BIG DATA APPLICATIONS

Geoffrey C. Fox
Gregor von Laszewski
Fugang Wang

laszewski@gmail.com

Big Data Applications

Editor Geoffrey C. Fox, Gregor von Laszewski, Fugang Wang

Big Data Applications

1 Preface

1.1 Version

Date: Thu Jan 10 15:56:18 2019 -0500

This document can be downloaded from

https://github.com/cloudmesh-community/book/blob/master/vonLaszewski-bigdata-application.epub?raw=true

1.2 Corrections

The material presented here is all managed in github.com/cloudmesh-community/book. In case you see an error or like to make a contribution of your own section, you can do so in github via pull requests.

The easiest way to for example fix an error is to read the ePub and click on the cloud ☁ symbol in a heading where you see the error. This will bring you to an editable document (if you are signed into github). You can directly fix the error in the web browser and create there a pull request.

The great thing about doing it this way is that the authors will be integrated from github the next time we compile the material. Thus even if you have a single spelling error corrected, you will be acknowledged.

1.3 Contributors

Contributors are sorted by the first letter of their "combined Firtsname and Lastname and if not available by their github ID. Please note that the authors are identified through git logs in addition to some contributors added by hand. The git repository contains more than the documents included in this section. Thus not everyone in this list may have directly contributed to this document. However if you find someone missing that has contributed (they may not have used this particular git) please let us know. We will add you. The contributors that we are aware of include:

> Anand Sriramulu, Ankita Rajendra Alshi, Arnav, Averill Cate, Jr, Bertolt Sobolik, Bo Feng, Brad Pope, Dave DeMeulenaere, De'Angelo Rutledge, Eliyah Ben Zayin, Fugang Wang, Geoffrey C. Fox, Gerald Manipon, Gregor von Laszewski, Hyungro Lee, Ian Sims, IzoldaIU, Javier Diaz, Jonathan Branam, Juliette Zerick, Mani Kagita, Miao Jiang, Mihir Shanishchara, Min Chen, Murali Cheruvu, Orly Esteban, Pulasthi Supun, Pulasthi Supun Wickramasinghe, Pulkit Maloo, Qianqian Tang, Ravinder Lambadi, Richa Rastogi, Ritesh Tandon, Saber Sheybani, Sachith Withana, Sandeep Kumar Khandelwal, Silvia Karim, Swarnima H. Sowani, Tim Whitson, Tyler Balson, Vafa Andalibi, Vibhatha Abeykoon, Vineet Barshikar, Yu Luo, ahilgenkamp, aralshi, bfeng, btpope, harshadpitkar, isims1, janumudvari, karankotz, qianqian tang, rirasto, shilpasingh21, swsachith, varunjoshi01, vineetb-gh, xianghang mi

1.4 Creating the ePubs from source

Although you will never likely to create the epub from source, we have included this section for our most advanced contributors and those that update the epub on github.

Please note that you must have at least Pandoc version 2.2.3 installed. You will also need Python version 3.7.1 to run the scripts needed to assamble the document. Earlier versions will not work. You can check the versions with

```
$ pandoc --version
$ python --version
```

1.4.1 OSX Requirements

This is just a guess I for got how to install all of this, it may be documented in another md file, grep -R for brew

```
$ brew install graphviz
# needs version >2.2.3 of pandoc see travis.yml for
# proper install if brew does not work
$ brew install pandoc

$ brew install pandoc-citeproc
$ brew install node
$ npm install --global mermaid-filter
$ npm install --global pandoc-index
$ git clone https://github.com/tomduck/pandoc-fignos.git
$ cd pandoc-fignos/
$ pip install .
```

1.4.2 Ubuntu requirements

```
$ sudo apt-get update
$ sudo apt-get install graphviz
$ wget https://hackage.haskell.org/package/pandoc-2.2.3.2/pandoc-2.2.3.2.tar.gz
$ git clone https://github.com/jgm/pandoc-citeproc.git
$ wget -qO- https://get.haskellstack.org/ | sh
$ tar xvzf pandoc-2.2.3.2.tar.gz
$ cd pandoc-2.2.3.2
$ stack setup
$ stack install
$ cd ..
$ cd pandoc-citeproc
$ stack setup
$ stack install
$ npm install --global mermaid-filter
$ npm install --global pandoc-index
```

1.4.3 Creating a book

First you have to check out the book source from github with:

```
git clone git@github.com:cloudmesh-community/book.git
```

Books are organize in directories. We currently have

```
./book/cloud/
./book/big-data-applications/
./book/pi
./book/writing
./book/222
```

To compile a book go to the directory and make it. Let us assume you like to create the cloud book for 516

```
$ git clone https://github.com/cloudmesh-community/book.git
$ cd cloud
$ make new
```

To view it you say

```
$ make view
```

After you have done modifications, you need to do one of two things. In case you add new images you need to use

```
$ make new
```

otherwise you can just use

```
$ make
```

The structure of the books is maintained in chapters.yaml.

In case you add a new chapter, you have to say

```
$ make update
$ make new
$ make view
```

1.4.4 Publishing the book to github

⚠ This task should only be done by with direct write privileges. and never be part of a pull request. Please discuss with Gregor von Laszewski the details of your update first. Typically Gregor is the one who publishes it.

To publish the book say

```
$ make publish
```

1.4.5 Creating Drafts that are not yet published

Developers of the manual can modify the `Makefile` and locate the variable `DRAFT=` to add additional sections and chapters they work on, but should not yet been distributed with the main publication. Simply add them to the list and say

```
$ make draft
$ make view
```

to create the draft sections only and view them.

To conveniently call them in a lazy fashion in a terminal you could use the following two aliases.

```
alias m='make; make view'
alias d='make draft; make view'
```

This allows you to typ `m` for the main volume and `q` for the draft. Please note that all artifacts are written into the dest folder.

1.4.6 Creating a new book

Let us assume you like to create a new book. The easiest way to start is to copy from an existing book. However, make sure not to copy old files in dest. Let us assume you like to call the book gregor and you coppy from the 222 directory.

You have to do the following

```
$ cd 222
$ make clean
$ cd ..
$ cp -r 222 gregor
```

Now edit the file chapters.yaml and copy the section with `BOOK_222=` to `BOOK_gregor`. Make modifications to the outline as you see fit.

Now you can create the book with

```
$ cd gregor
$ make update
$ make new
```

1.5 Github Issues

⚠ The issues are automatically created from Github Issues. Please change them directly in github.

Do not modify the table. or this file

To file new issues, please go to:

- https://github.com/cloudmesh-community/book/issues

Additionally, we use the following notation to coordinate sections and chapters that are open, conducted by students. This is indicated by a prefix to the issue summary.

Prefix	Description
Internal:	Assigned to a staff member and done internally by them
Open:	A task is open and can be executed also by students.
Assigned:	A task is assigned and is currently been executed by the assignee
Done:	The task is done and needs review.

Assignees can change the summary to done. We will experiment with possibly additional labels to communicate the state via labels. Please stay tuned.

.

Count	Number	Title	Assignee
1	227	internal: Virtual machine .md file is missing	tbalson
2	225	created puppet.bib	None
3	223	Open: Section CircleCi	None
4	211	some refs missing	pulasthi
5	190	internal: Hadoop 3.1.1	bfeng
6	189	Open: bibtex entries for FaaS Sections	bfeng
7	186	internal: figure refernces {#fig:label}	bfeng
8	184	fix bibtex entries	None
9	180	E534 Cloud Section lecture videos merged into E-books	None
10	173	internal: biber installation	bfeng
		internal: Upload lecture videos from Prof Fox's Google Drive to	

11	152	Youtube	laszewsk
12	148	Open: IU GUEST, IU Secure & Raspberry Pi	bfeng
13	146	Open: ssh port forwarding section	pulasthi
14	145	Open: manage vm guests with virsh	pulasthi
15	144	internal: bigdata missing improvements	laszewsk
16	139	Open: Codecov	pulasthi
17	134	internal: bibtex erros in technologies	bfeng
18	126	bibtex: 523	bfeng
19	115	Check and redo Docker Hadoop 3.0.1 and Docker for Hadoop section.	bfeng
20	110	replace images with text	bfeng
21	77	add mapreduce to syllabus	laszewsk
22	76	openapi demo	laszewsk
23	68	internal: bib: create bibtex entries for go-intro.md	bfeng
24	51	Open: OpenAPI in GO	bfeng
25	50	Open: REST in go	bfeng
26	49	Open: Develop a tutorial on the Parallel language constructs in Go	bfeng
27	48	Open: Develop an introduction to Go	bfeng
28	47	Open: Hot cold isles in data center	laszewsk
29	9	Internal: Slides to video are not included in the introduction	laszewsk

.

1.6 epub Readers

This document is distributed in epub format. Every OS will have a suitable epub reader to view the document. Such readers can even be integrated into the browser you use, and when you click on an epub publication in github it could pop up the document in your reader. The way we create the document is through scalable markup, that iss you can zoom in and out of a page while the content will beautomatically rendered for the selected page size. If you ever see a content that does not fit on a page we recommend you zoom out with your reader to make sure you can see the entire content.

1.7 Notation

☁ `:cloud:`

> If you click on the ☁ in a heading, you can go directly to the document in github that contains the next content. This is convenient to fix errors or make additions to the content.

$

> Content in bash is marked with verbatim text and a dollar sign

```
$ This is a bash text
```

[1]

> References are indicated with a number and are included in the reference chapter [1]

⭕ `:o:`

> Chapters marked with this emoji are not yet complete or have some issue that we know about. These chapters need to be fixed. If you like to help us fixing this section, please let us know.

REST 36:02

> Example for a video with the `:clapper:` emoji

Slides 10

> Example for slides with the `:scroll:` emoji. These slides may or may not include audio.

Slides 10

Slides without any audio. They may be faster to download.

🎓

> A set of learning objectives with the `:mortar_board:` emoji.

☑

> A section is release when it is marked with this emoji in the syllabus.

❓

Indicates sections that are worked on by contributors

Sections marked by the contributor with this emoji when they are ready to be reviewed.

Sections that need modifications are indicated with this emoji.

⚠

A warning that we need to look at in more detail.

Notes are indicated with a bulb and are written in italic and surrounded by bars using the `:bulb:` emoji

---

Figures have a caption and can be refereed to in the epub simple with a number. We show such a refernce pointer while refering to fig. 1.

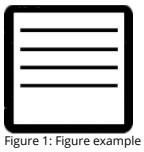


Figure 1: Figure example

Figures must be written in the md as

```
![Figure example](images/code.png){#fig:code-example width=1in}
```

You can refer to them with `+@fig:code-example`. Please note in order for numbering to work figure refernces must include the `#fig:` followed by a unique identifier. Please note that identifiers must be realy unique and that identifies such as `#fig:cloud` or similar simple identifiers are a poor choice and will likely not work. To check, please list all lines with an identifier such as

```
$ grep -R "#fig:" chapters
```

and see if your identifier is truly unique.

Other emojis

Other emojis can be found at https://gist.github.com/rxaviers/7360908

⚠ Please note that there is currently a bug when our document is exported to html or to PDF, as emojis are for sme reason not properly embedded. Hence to read the document we recommend that you use an ePub reader.

Abbreviations ⭕

THis does not yet work

Abbreviations can be stored in the file chapters/dbase and used as follows while using the video abbreviation

```
+video
```

+video

It uses the filter defined at https://github.com/scokobro/pandoc-abbreviations

1.8 Organization

This class is an online class. Online classes require you to be very disciplined in order to execute the tasks necessary for the class in time. It is your responsibility to organize the lessons so that you can complete them not only by the end of the semester, but also in time for conducting your assignments. This is a great opportunity for you to structure the class based on your availability. The classes are attended by two different set of students. One set are remote online students, while to other are residential students. For the residential students we have a mandatory in person meeting that takes place at the posted location and hours once a week. For pure online students we have weekly online hours that we will identify based on our availability and a doodle poll.

Figure Components of the Class i523, i423, e534 showcases the different parts of the class. If you have taken a previous class with us you are able to continue your previous project upon approval. It must however be a significant improvement. Please note that the in i523 and i524 the project and it's report can be substituted by a longer term paper that does not require programming. As this is a significant reduction in work and goals, for that class, the maximum grade in this case for the entire class can only be an A-.

There will not be any bonus projects or tasks to improve grades. Instead make sure your deliverables of the few assignments are truly outstanding.

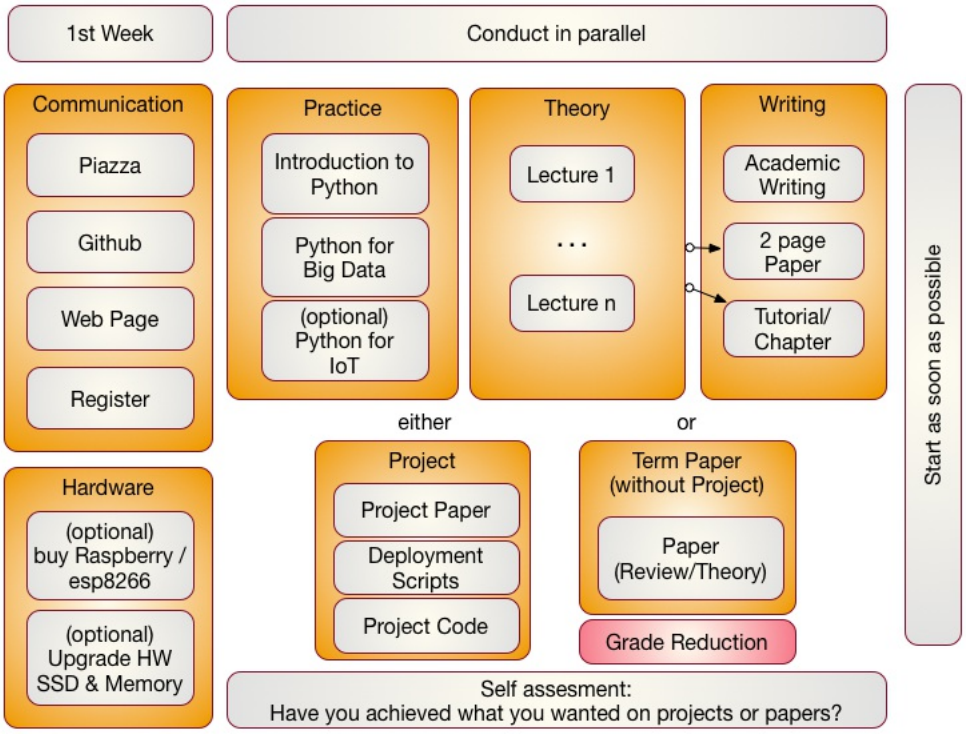


Figure: Components of the Class i523, i423, e534

The content for this class will be available through a series of documents that will be regularly updated and are linked from this document. All communication is done with Piazza. issues.

1.8.1 First Week

In the first week we will be introducing you how we communicate to you. Naturally you need to register for the class. Once you register you need to set up a number of services.

1.8.2 Access to Clouds

As part of the course you will also need access to a cloud. We will try our best to provide you with access to suitable computers for the class, but do be reminded that the amount of time and access to supercomputers and clouds we offer is limited. Our class policy is to use the compute resources only when you really need them. Thus you **must** shut down your VMs when they are not in use. It would be a violation of class policy if we would find out through an analysis of the cloud logs that you unnecessarily keep your VMs running. Thus we will implement a **strict policy** that you must record yourself how many hours you run VM's and provide this information to us. We will than compare that time with the time recorded by the computer system as well as with your target application and will deduct points form your project if you can not justify why you have not shut down your VMs. A resource section needs to be added to your report justifying the used resources.

Why is this such a big deal you may ask? For example we estimate if every student in class violates this policy it would cost about $200000 to rent the time for this on a public cloud. Due to this high cost, we no longer tolerate deliberate violations of the policy and will terminate your account. Furthermore, violators will have to find alternative resources to conduct their projects while not using our resources. In our case the problem is even beyond the issue of cost as our allocation on the clouds would be terminated due to abuse and **no student**, including those that follow policies, could use the cloud. It may take weeks to reestablish cloud access and would effect every student in class.

We will provide clarification for accessing cloud resources and teach you how to avoid getting in such a situation. I am sure that a future employer of yours will be real happy if you have a deep understanding of resource vs. cost estimate.

Listing the used computer time for your project is part of your report.

1.8.3 Using Your Own Computer

In many cases however you could and are recommended to use your own personal computer, but make sure the computer is up-to-date. We also like to make sure that you do not use a work computer as you need to avoid that when you develop a cloud program you do not by accident introduce a security risk on your machine. This does not mean that you need to buy a new computer, or need to upgrade it. However, if you consider an upgrade of an older machine please consider the following.

These days we recommend that your computer has a solid-state drive and fast memory (put as much memory in your machine as is supported). We recommend 16 GB off main memory which gives you enough space to run containers, virtual machines and naturally the main operating system. We fond that students with only 8GB could do the work but it was slow. In some cases the memory to conduct their projects was not sufficient. Make sure you follow your upgrade guide to your computer and by suitable memory chips. In most cases you have to buy them in **pairs** and make sure all chips in your computer are the same. When it comes to buying a solid-state drive, make sure that you buy one that is compatible with motherboards bus speed. As you may want to reuse your solid-state drive at a later time I suggest to get a 6GB/s SSD and not a 3GB/s.

In case of Windows, your could also get yourself a UBS stick or external SSD drive and place ubuntu on it. You could than use your bios to boot of from that drive. This way you do not have to modify anything on your computer. This method works very well for most computers and allows you to use the maximum memory while for example using ubuntu.

Students that only had a chromebook and took this class gave us the feedback that they are too inconvenient as they do not allow you to program directly in python on them and the ssh terminals to login to other computer although working are not supporting the GUI tools.

Another option is (if money is an issue) you can buy a Raspberry Pi and edit your programs there and when satisfied run them on a cloud. However a PI is small and has only very limited memory and processing power.

We also like to remind you that this course does not require you to purchase expensive text books, thus the money you safe on this could be used in upgrading your hardware or renting yourself from your own money time on AWS. However, be careful with the cloud its easy to spend lots of money there if you are not careful.

1.8.3.1 Self Discipline

As this class has no graded tests and only few graded homework, we like that you deliver an **exceptional** project report or paper. Instead of focusing on preparing for tests we provide you with the opportunity to **explore** without the pressure of grades. However you should not give up or take the easy way out or it will effect you in your project execution. Also, to achieve your best do not just say: We do not have a test, so let me not do this weeks assignment, let me do it next week. After a couple of times with this attitude you will be in big trouble. All this requires discipline. For example, if you believe you are so good that you can do a project within one week before deadline, you will **certainly fail**. To avoid this and to introduce discipline, you will also be monitored on progress and we check your github for activities which will be part of the

participation grade.

It will be up to you to assess what you want to deliver before handing it in to us. Self assessment or a check with other students is a real good way to do that. You should not expect to get an A if you yourself are not convinced about your project or are unsure about it. Common sense prevails.

1.8.3.2 Fun

I hope you have fun and are able to integrate in the projects your own thoughts and interests.

We have quotes form students such as

> "This is the best class I have taken ..."

or

> "I really enjoyed taking this class and having maximum flexibility to schedule the lectures."

or

> "The lessons learned from this class were adopted within my company."

Furthermore you should know that the way we teach the class has also been adopted in STEM classes. As a result a team coached by Gregor von Laszewski won an award at the FLL Robotics World Championship. The certainly had lots of fun and integrated their own ideas into the project that won the award.

1.8.3.3 Uniqueness

We will try to have every project or paper to be non overlapping with another topic, If there are overlaps we may ask you to modify your focus.

1.8.3.4 Continuation

If you like to put additional effort in the project, the report could be made to a conference or workshop paper. Dr. von Laszewski is happy to help as co-author.

1.8.4 Parallel Tracks

In this class we have three parallel tracks.

1.8.4.1 Track 1: Practice

Track 1 introduces you to using python for Big Data. We recommend that you do know a programming language for any of our courses. Learning a programming language is not part of the hours you spend for this class. It is an additional time requirement that you must plan for. Maybe you want to take for example a python programming language class at the same time. This can also be done in self study. Although you do not need to know any programming language, it is certainly useful as it will make this course much easier for you. We had students that had no prior programming knowledge and successfully completed the course. So we know it can be done. The course is designed in such a fashion, that there is enough time to learn programming and do a project.

We provide you with a general introduction to Python. This includes enough knowledge so you can conduct a project with it. We will build on these technologies to introduce you to python libraries that can be used for big data. Not every section in the Python chapter will need to be used in this class. At minimum you must understand python classes, and the map reduce function.

1.8.4.2 Track 2: Theory

The theory track includes a number of online lectures that introduces you to a variety of topics related to Big Data. You have especially the opportunity to become part of a project that would contribute to the understanding and the development of a Big Data Architecture developed in collaboration with NIST. Other topics that are covered include IoT, Health Care, Physics, Science, Biology, Genomics, and so forth. We will update the Theory track and will release lectures in the specified areas. Some lectures may be used in multiple classes.

1.8.4.3 Track 3: Writing

You have a choice in this class between writing a two page review paper about a big data technology or application (area), or contribute a chapter to this document. We explain next the difference:

Review Paper: A review paper will introduce your into how to write an academic paper and conduct proper bibliography management. Knowing how to write is a preparation for your term project.

You will be writing a paper that is 2 pages long (in a particular format, typically ACM) possibly within a team. In case you work in a team you have to produce as many papers as you have team members. We like to avoid that all students take the same topic. We will use github to avoid that everyone chooses same topic. Knowing how to write is a preparation fo your project or term paper.

We noticed a curious observation in previous classes. Any paper written in MSWord was inferior. Thus we no longer provide the choice to write papers in MSWord in order for you to achieve your best. Papers and Project reports must be written in LaTeX or markdown. For the classes starting in 2018 we do prefer markdown and may restrict all document to this format.

Chapter: A is chapter to a review paper, but is written in markdown and can be added to the lecture notes. A chapter should be formulated in a consistent form and is equivalent in length (number of words) to those of the 2 page paper. Bibliography management is conducted in bibtex and can be used in the markdown document.

Important in both cases is that you stay focused. You can assume that if you write a document about "Big Data in Baseball", you do not spend 1.5 pages to describe what big data is and only half a page where baseball fits in. What you should do is focus on the topic. A chapter could also include some practical lessons with real programming lessons.

1.8.4.4 Track 4: Term Paper/Project

The major deliverable of the course is a term project or paper. The exact details will be posted on the Web page in this document. The important part is that you start on this project once you are sufficiently familiar with Track 1-3. However you can also use the project to for example learn python and engage in a goal oriented learning activity while working towards implementing your project and integrating the python lessons that you encounter. The same is valid for the theory.

It is **expected** that you identify a suitable analysis and data set for the project and that you learn how to apply this analysis as well as justify it. It is part of the learning outcome that you determine this instead of us giving you a topic.

Furthermore, it is also important to note that if you do not do a project (this is your option) the maximum grade for the entire class is limited to an A-. This is achieved simply by reducing the grade of your term report by a full grade due to the distribution of the grade this will result in a fractional grade reduction and limits the maximum grade to an A-.

Starting in 2018 the paper format will be Markdown.

1.8.5 Plagiarism

In the first week(s) of class you will need to read the information about plagiarism. If there are any questions about plagiarism we require you to take a course offered from the IU educational department.

Warning:

> If we find cheating or plagiarism, your assignment will be receiving an F. This especially includes copying text without proper attribution. We are required to follow IU policy and report your case to the dean of students who may elect to expel you form the university. Please understand that it is your doing and the instructors have no choice as to follow university policies. Thus, please do not blame the instructors for your actions. Excuses such as "I missed the lecture on plagiarism", "I forgot to include the original reference as I ran out of time", "I did not understand what plagiarism is" do not apply as we explicitly make the policies clear. This applies to all material prepared for class including assignments, exercises, code, sections, tutorials, papers, and projects. If there is no time, do not submit and instead of an F ask for an incomplete. In fact if you know you have plagiarized, do not even have us review your paper.

For more information on this topic please see:

- https://studentaffairs.indiana.edu/student-conduct/misconduct-charges/academic-misconduct.shtml

Furthermore you are supposed to review our lecture material on plagiarism and take the plagiarism test. The information is located at:

- Scientific Writing with Markdown

In Piazza a form will be posted that will ask you for your passing ID. If the form is not yet posted, please be patient till it is.

1.9 Course Policies

We describe briefly some class policies.

1.9.1 Discussion via Piazza

1. All communication is done in Piazza. It is an IU approved communication tool and superior to CANVAS discussion list
2. CANVAS is only used with students that are not in Piazza. The only messages they will get is to activate Piazza and use the class Piazza
3. You are allowed to use whatever calendar system you like.
4. We will not use CANVAS calendar, however you can manage that yourself. CANVASS allows you to add events such as assignment deadlines.
5. Piazza is FERPA compliant https://piazza.com/legal/ferpa

1.9.2 Managing Your Own Calendar

From time to time we get the question from a very small number of students why we are not using or uploading the assignment deadlines and the assignment descriptions to CANVAS. The reason for this is manifold. First, our class has different deadlines for different students within the same class. This is not supported by CANVAS and if we would use CANVAS leads to confusion and clearly shows the limitation of CANVAS. Second, we are teaching cloud computing. CANVAS is not a tool that you likely will use after graduating. Thus we are providing you the ability to explore industry standard tools such as github to maintaining your own tasks and deadlines, while for example using github issues (see the section about github). We highly recommend that you explore this as part of this class and you will see that managing the assignments in github is **superior** to CANVAS. Naturally you can not make that assessment if you are not trying it. Thus we like you to do so and it is part of any assignment in your class to use github issues to manage your assignments for this class.

However, if you still want to manage your tasks in CANVAS, you can do so. CANVAS allows you to create custom events, so if you see an assignment in piazza or the handbook, you are more than welcome to add that task yourself to your own CANVAS tasks. As we have only a very small number of assignments this will not pose a problem either for graduate or undergraduate. Being able to organize your deadlines and assignment with industry accepted tools is part of your general learning experience at IU.

Obviously, this makes it also possible to use any other task or calendar system that you may use such as google calendar, jira, microsoft project, and others.

As you can see through this strategy we provide the most flexible system for any student of the class, while giving each student the ability to chose the system they prefer for managing their assignment deadlines. It is obvious that this strategy is superior to CANVAS as it is much more general.

1.9.3 Online and Office Hours

To support you we have established an open policy of sharing information not only as part of the class material, but also as part of how we conduct support. We establish the following principals:

- in case of doubt how to communicate address this early in class and attend online hours;
- all office hours if not of personal nature are open office hours meaning that any student in class can be joined by other students of the class and all meeting times are posted publicly. This includes in person office hours with TAs. Other students are allowed to listen in and participate.
- it is in your responsibility to attend in person classes and online hours as we found that those that do get better grades. For residential students participation in the residential classes may be mandatory. International students may need to check university policies.
- instructors of this class will attempt within reason to find suitable times for you to attend an online hour in case you are an online student.

1.9.3.1 Office Hour Calendar

Online Students:

- Online hours are prioritized for online students, residential students should attend the residential meetings.

Residential Students:

- Residential students participate in the official meeting times. If additional times are required, they have to be done by appointment. Office hours will be announced publicly. All technical office hours are public and can be attended by any student. Online hours are not an excuse not to come to the residential class.

  However Residential students can in addition to the residential class use the online student meeting times. However, in that case online students will be served first. It is probably good to check into the zoom meeting and identify if the TA has time. They will be in zoom.

Meeting times and phone numbers are posted in your piazza in the `Resources` section

1.9.4 Class Material

As the class material will evolve during the semester it is obvious that some content will be improved and material will be added. This benefits everyone. To stay up to date, please, revisit this document on weekly basis. This is practice in any class.

1.9.5 HID

You will be assigned an hid (Homework IDentifier) which allows us to easily communicate with you and does allow us to not use your university ID to communicate with you.

You will receive the HID within the first couple of weeks of the semester by the TA's.

1.9.6 Class Directory

You will get a class directory on github.com and not the iu github. For that reason you will be asked to give us a github id so we can create a openly accessible directory for you in which you can collaborate with the students of this class. The directories are only used to store the artifacts of the class. As all artifacts are supposed to be open source github.com provides us with the service that millions of professionals and researchers use for their work.

1.9.7 Notebook

All students are required to maintain a class notebook in github in which they summarize their weekly activities for this course in bullet form. This includes a self maintained list of which lecture material they viewed and what they worked on in each week of the class.

The notebook is maintained in the class github.com in your hid project folder. It is a file called `notebook.md` that uses markdown as format. Notebooks are expected to be set up as soon as the git repository was created.

You will be responsible to set up and maintain the `notebook.md` and update it accordingly. We suggest that you prepare sections such as: Logistic, Theory, Practice, Writing and put in bullet form what you have done into these sections during the week. We can see from the github logs when you changed the notebook.md file to monitor progress. The management of the notebook will be part of your discussion grade.

The format of the notebook is very simple markdown format and must follow these rules:

- use headings with the # character and have a space after the # Use `# Week X: mm/dd/yyyy - mm/dd/yyyy` as the subject line for each week
- use bullets in each topic.
- Do not refer to section numbers form the epub in your notebook as they can change. Instead use the section name or headline and possibly a URL. When using URLS in md format they must be enclosed in <> or `[text](URL)`

Please examine carefully the sample note book is available at:

- https://github.com/cloudmesh-community/hid-sample/blob/master/notebook.md

The notebook.md is not a blog and should only contain a summary of what you have done.

1.9.8 Blog

You can maintain your own optional blog. However, the blog will not be used for grading. Do not include sensitive information in either the blog or the notebook. A blog is not a replacement for the notebook. If something does not go so well, do not focus on the negative things, but focus on how that experience can be overcome and how you turn it to a positive experience. Be positive in general.

1.9.9 Waitlist

The waitlist contains students that are unable to enroll in a section of a course. Students choose to add themselves to the waitlist. They are not automatically added, but choose to do so intentionally based on the status of the course. There are two reasons for students to be on the waitlist. The first, and primary, reason is that the class is already at the scheduled, maximum capacity. Since there are no seats available, the student can elect to add themselves to the waitlist. The second reason is that the students' own schedule has a time conflict. This occurs when they are trying to enroll in a class that overlaps with the time of a class they are already enrolled in.

Students are moved from the waitlist to the regular section during a daily batch process, and not in real time. The process is not in realtime because the registrar receives many requests to increase capacity, decrease capacity, and change rooms. If the process were real time there would be a catastrophe of conflicts.

Students are moved from the waitlist in chronological order that they added themselves to the waitlist. If you are still on the waitlist there are no spaces free, the batch process has not run for the day, or the student in question has a schedule conflict.

Faculty are not able to selectively choose students from the waitlist.

How long does the waitlist process stay active?: The automated processing of the waitlist ends on Thursday of the first week of class At this time the waitlist will no longer be processed. As the residential class starts on Friday, this may cause issues. Either talk to the department on Thursday or show up on Friday. Most likely there will be spaces left. Students on the waitlist at that time will remain on the waitlist, but remain there until the student decides to change their registration. Students may not do that, because they get assessed a change schedule fee.

Students tell me they still want to enroll after the first week of classes. How do they do this?

Beginning Monday, after the first week of class students begin to use the eAdd process to do a late addition of the course. The request is routed to the professor of record on an eDoc and the faculty will be notified via email. Faculty can deny or approve based on whatever criteria they wish to apply. If the faculty member approves, the eDoc is electronically forwarded to the Academic Operations office and we will approve the

late add **if the room capacity** allows the addition, otherwise we must deny the addition because of fire marshal regulations. Many times, there are seats in a classroom/discussion/lab, but because other students have not officially dropped, enrollment is still at capacity.

After everything, a student that was unable to enroll in the class attended all year and completed all course work as if they had enrolled. Can the student get credit and can I give the student a grade?

Yes. There is a provision for a late registration - contact our office if this occurs. Students will be assessed a tuition fee at the time of late or retroactive registration.

1.9.10 Registration

The Executive Associate Dean for Academic Affairs requires starting Spring 2018 that students that are not officially enrolled, can not register at the end of the class if they in-officially took the class. Please make sure that within the first month you have enrolled. If we do not see in CANVAS, you are not in the class. In case you are on a waitlist it is your responsibility to work with the administration after the waitlist is over to be added to the class by getting permission from the School.

1.9.11 Auditing the class

We no longer allow students to audit the class because:

- Seating in the lecture room is limited and we want foster students that enroll full time first.
- The best way to take the class is to conduct a project. As this can not be achieved without taking the class full time and as auditing the class does not provide the full value of the class, e.g. not more than 10% of the class. Hence, we do not think it is useful to audit the class.
- Accounts and services have to be set up and require considerable resources that are not accessible to students that audit the class.

1.9.12 Resource restrictions

- It is not allowed to use our services we offer as part of the class for profit (e.g. just enrolling in the class to use our clouds).
- In case of abuse of available compute time on our clouds the student is aware that we will terminate the computer account on our clouds and the student may have to conduct the project on a public cloud or his own computer under own cost. There will be no guarantee that cloud services we offer will be available after the semester is over. Projects can be conducted as part of the class that do not require access to the cloud.

1.9.13 Incomplete

Incompletes have to be explicitly requested in piazza through a private mail to instructors. All incompletes have to be filed by DATE TO BE ANNOUNCED.

Incomplete's will receive a fractional Grade reduction: `A` will become `A-`, `A-` will become `B+`, and so forth. There is enough time in the course to complete all assignments without getting an incomplete.

Why do we have such a policy? As we teach state-of-the-art software this software is subject to change, not only within the course, but also after the course. As we may offer some services and only have access to the TA's during the semester it is obvious that we like all class projects and homework assignments to be completed within a semester. Services that were offered during the semester may no longer be available after the semester is over and could adversely effect your planing. It will be in the students responsibility to identify such services and provide alternatives if they become unavailable. We try hard to avoid this but we can not guarantee it.

Furthermore, once an incomplete is requested, you will have 10 month to complete it. We will need 2 month to grade. No grading will be conducted over breaks including the summer. This may effect those that require student loans. Please plan ahead and avoid an incomplete.

The incomplete request needs to be off the following format in piazza:

```
Subject:
    INCOMPLETE REQUEST: HID000: Lastname, Firstname

Body:
    Firstname: TBD
    Lastname: TBD
    HID: TBD
    Semester: TBD
    Course: TBD
    Online: yes/no

    URL notebook: TBD
    URL assignment1:
    URL assignment2: TBD
    ....
    URL paper1: TBD
    URL project: TBD

    URL other1: TBD
```

Please make sure that the links are clickable in piazza. Also as classes have different assignments, make sure to include whatever is relevant for that class and add the appropriate artifacts.

In case of an incomplete you may be asked to do additional assignments and assignments that have been adapted based on experience from the class. Please note also that we could reject an assignment if it is identified to no longer reflect the state-of-the-art. All previously submitted assignments such as papers, sections, and so on will be reviewed on this criterion. For example, let us assume you developed a tutorial on technology visit version x. Let us assume that since you completed this task a version x+1 comes out. It will be your obligation to update the deliverable. This is also true if the tutorial has been graded previously. The incomplete and the change of the software have at this time negated the originally assigned grade. In most cases the changes may be small. In other cases the changes could be substantial. Hence avoid an incomplete.

Here is the process for how to deal with incompletes at IU are documented:

- http://registrar.indiana.edu/grades/grade-values/grade-of-incomplete.shtml

1.9.13.1 Exercises

E.Policy.1

Take the Plagiarism test, See the Scientific Writing I epub for more details.

1.10 Course Description

1.10.1 Big Data Applications and Big Data Applications Analytics

- Indiana University
- Fall 2018
- Course Numbers: E534, I523, I423
- Faculty: Dr. Geoffrey C. Fox
- Credits: 3
- Prerequisite(s): Knowledge of a programming language, the ability to pick up other programming languages as needed, willingness to enhance your knowledge from online resources and additional literature. You will need access to a "modern" computer that allows using virtual machines and/or containers. Knowledge of material taught by e516 is desirable and will make project execution easier. e516 and this class can be taken in parallel.
- This page is maintained and updated at e534-i523: Big Data Applications and Big Data Applications Analytics
- Course Description URL: https://github.com/cloudmesh-community/book/blob/master/chapters/class/e534-i523.md
- Registrar information and Other related classes

1.10.2 Course Objectives

This class investigates the use of clouds running data analytics collaboratively for processing Big Data to solve problems in Big Data Applications and Analytics. Case studies such as Netflix recommender systems, Genomic data, Sports, Health, and more will be discussed.

The course has the following objectives:

- Provide an introduction to Big Data
- Provide an introduction to Big Data Analytics
- Provide overviews of different Big Data Application areas
- Explore state-of-the-art big data and cloud technologies and services while providing a write up about it and exploring it practically with a section that you develop
- Enforce the theoretical knowledge with a project that you conduct in one of the application areas.

1.10.3 Learning Outcomes

- Be able to explain the concepts of the big data paradigm including its paradigm shift, its characteristics, and the advantages. Contrast them with the challenges and disadvantages.
- Be able to identify bigdata applications and analytical methods needed to support real world applications.

- Be able to implement a real world application in big data.
- Be able to conduct sophisticated analysis of big data.
- Be able to communicate the results through sections, chapters, manuals, and reports.
- Be able to work in a team to develop collaboratively software or contribute collaboratively to develop sections using clouds.

1.10.4 Course Syllabus

- Release classes are marked with ☑
- Classes that will be improved are marked with ⭘

	Date[a]	Unit	Title	Description
☑	08/24	1	Fundamentals	Introduction to Big data
☑	08/24	1	Introduction	Intoduction to Clouds ⭘
☑	08/24	1	Introduction	Overview of Big Data ⭘
☑	09/03			Big Data Use Cases
☑	10/15	2	Basic Math	Minimal Statistics
☑	09/17		Applications	Physics and Astronomy
☑	09/24	3		Lifestyle and e-Commerce
☑	10/15			kNN and Clustering
☑	10/15			k-means
☑	10/08			Web Search
☑	10/08			Sports
☑	10/15			Health
☑	10/22			Sensor
☑	10/29			Radar
⭘	11/05	4	Basic Cloud	Cloud Computing for Big Data I ⭘ (outdated)
⭘	11/19			Cloud Computing for Big Data II ⭘
⭘	12/03	5	Applications	Edge Computing and the Cloud ⭘
☑	09/03-11/02*	5	Technology Summaries	Contribute your assigned technology summaries while not plagiarizing.
☑	09/03-11/02*	5	Example	Contribute a significant technical example related to Big Data technologies. Do not develop redundant or duplicated content.
☑	09/03-11/02*	5	Chapter	Contribute a significant chapter while not plagiarizsing that may use your example to the class documentation. Do not develop redundant or duplicated content.
☑	09/03-11/26*	6	Project Type A	Build a Big Data Application Project (maximum grade for class A or A+)
☑	09/03-11/26*		Project Type B	Write a comprehensive Report without programming (maximum grade for entire class A-)

Students need only to do one project. The project is conducted thought the entire semester.

a. Dates may change as the semester evolves

(*) The project is a long term assignment (and are ideally worked on weekly by residential students). It is the major part of the course grade.

(*) Sections prepare you for documenting a technical aspect related to cloud computing. It is a preparation for a document that explains how to execute your project in a reproducible manner to others.

- all times are in EST

Additional Lectures will be added that allow easy management of the project. These lectures can be taken any time when needed

1.10.5 Assessment

This course is focusing on the principal Learning by Doing which is assessed by simple graded and non-graded activities. The assessment may include comprehension of the material taught, programming assignments, participation in online discussion forums, or the contribution of additional material to the class showcasing your comprehension.

The comprehension is also measured by the development of one or more sections in markdown that can be distributed and replicated to other students. This is done in preparation for the project that must include a simple deployment and runtime instruction set.

The main deliverable of the class is a project. The project is assessed through the following artifacts:

- a. Deployment and install instructions,
- b. Project report (typically 2-3 pages per month, sections and chapters can be reused if possible),
- c. Working project code that can be installed and executed in reproducible manner by a third party
- d. Code developed by the project team distributed in github.com
- e. Project progress notes checked into github throughout the semester. Each week the project progress is reported and will be integrated into the final grade.
- f. three discussions or progress reports with the instructors about your project

The grade distribution is as follows

- 10% Comprehension Activities
- 10% Section
- 10% Chapter
- 70% Project

As the project is the main deliverable of the course it is obvious that those starting a week before the deadline will not succeed in this class. The project will take a significant amount of time and fosters the principal of "Learning by Doing" at all stages throughout the semester.

The class will not have a regular midterm, but it is expected that you have worked on your project and can provide a snapshot of the progress outlining the goals of the project and how you will achieve these goals till the end of the semester.

The final Project is due Dec. 1st. Issues with your project ought to have been discussed before this deadline with the TA's The TAs will in the next 14 days go over the projects and evaluate major and minor issues that you may be able to fix without penalty. Larger changes will receive a grade penalty. The last fix (upon approval) possible will be Dec 7th.

1.10.5.1 Incomplete

Please see the university regulations for getting an incomplete. However, as this class uses state-of-the-art technology that changes frequently, you must expect that an incomplete may result in significant additional work on your behalf as your project may need significant updates on infrastructure, technology, or even programming models used. It is best to complete the course within one semester.

1.10.5.2 Calendar

Assignment #	Event		Date
	Full Term		16 Weeks
	Begins		Mon 08/20
1, 2	**Bio, Notebook**	assigned	Mon 08/20
1, 2	**Bio, Notebook**	due	Mon 08/27 9am
3	**Section**	assigned	Mon 09/03
4	**Chapter**	assigned	Mon 09/03
5	**Project**	selection or proposal	Mon 09/03
	Labor Day		Mon 09/03
5	**Project**	update	10/05 9am EST
	Fall Break		10/05 - 10/07
	Auto W		Sun 10/21
5	**Project**	update	11/02 9am EST
3	**Section**	due	11/02 9am EST
4	**Chapter**	due	11/02 9am EST
	Thanksgiving		11/18 - 11/25
5	**Project**	due	11/26 9am EST
5	**Project** (no penalty)	improvements	11/26 - 12/03 9am EST
5	**Project** (with penalty)	improvements	12/04 - 12/07 9am EST
	Final Deliverables due		12/07 9am EST
	Grading		12/01 - 12/14
	Ends		Fri 12/14
	grade submission to school		Fri 12/14
	grade posting by registrar		12/31

- TA's must be available till all grades have been submitted.
- Bio: a formal 3 paragraph Bio
- Notebook: a markdown in which you record your progress of this class in bullet form
- All times are in EST
- Dependent on class progress Comprehension Assignments may be added

1.11 Example Artifacts

Learning Objectives

- Identify what other students have done previously.
- Look at previous chapters, which are collected as technology reviews.
- Look at previous project reports.
- Looking at the documents provides you with an initial overview of the scope for the artifacts.

As part of this class you will be delivering some artifacts that are being graded. Some of them include writing a chapter that can be contributed to the class lecture and a project. To showcase you some example artifacts take a closer look at the documents listed in this section. Please also note that you can not duplicate or replicate a students previous work without significant improvements. All material listed here is available online, including all source code.

1.11.1 Technology Summaries

We are maintaining a large list of technologies related to clouds and Big Data at

- https://github.com/cloudmesh/technologies

This repository generates the following epub

- https://github.com/cloudmesh/technologies/blob/master/vonLaszewski-cloud-technologies.epub

Students that have to contribute as part of their class Technology summaries are asked to produce meaningful, advertisement free summaries of the technology and indicate in some cases if not obvious show they relate to cloud or big data. The length of such summaries is about 300 words. Students of E516 do not have to contribute to this and will instead focus on programming. Students of I423, I523 and other sections must contribute to it and will get an assignment related to it. We post here the existence of this document also for 516 students. They can voluntarily improve or add sections if they like which will go into their discussion credit.

Please use the following indicators to mark the progress of summaries that you are working on.

😀 ready for review

selected by student so others do not select it and we know what is worked on

needs revision (only assigned by ta, after smiley)

The signs are put as follows. You can view an example at https://github.com/cloudmesh/technologies/blob/master/chapters/tech/bioconductor.md

Ex - Title Of Summary fa18-xxx-xx

1.11.2 Chapters

Previously we asked students to write a 2 page paper on a topic related to bigdata analytics or cloud technologies (dependent on the course). Example papers are listed bellow

- Use Cases in Big Data Software and Analytics Vol. 1, Gregor von Laszewski, Fall 2017, http://cyberaide.org/papers/vonLaszewski-i523-v1.pdf
- Use Cases in Big Data Software and Analytics Vol. 2, Gregor von Laszewski, Fall 2017, http://cyberaide.org/papers/vonLaszewski-i523-v2.pdf
- Big Data Software Vol 1., Gregor von Laszewski, Spring 2017, https://github.com/cloudmesh/sp17-i524/blob/master/paper1/proceedings.pdf
- Big Data Software Vol 2., Gregor von Laszewski, Spring 2017, https://github.com/cloudmesh/sp17-i524/blob/master/paper2/proceedings.pdf
- Vol 8, Gregor von Laszewski, Spring 2018, http://cyberaide.org/papers/vonLaszewski-cloud-vol-8.pdf

This has however resulted in a large number of duplicated material especially in the introductions and motivations. Thus we like this year to have you more focused on the topic and do not write a large introduction on what big data or cloud computing is. Therefore we renamed the 2 paper to a chapter, while you could assume certain things that have already been taught to you and you do not have to repeat it.

1.11.3 Project Reports

The goal of the class is to use open source technology to also write your technical reports. As a beneficial side product, we are able to distribute all previous reports from students to you. In your reports you will be doing a similar report, but will not use the same topic, without a significant improvement from a report already delivered in that area. For big data we have more than 1000 data sets we point to. I am sure you can do a unique project. For engineering cloud there are recently so many new technologies that there is not much chance of an overlap. TA's will review your project proposal, but it is your responsibility to make sure they are unique.

Please note that we do not make any quality assumptions to the published papers. It is up to you to identify outstanding papers.

- Use Cases in Big Data Software and Analytics Vol. 3, Gregor von Laszewski, Fall 2017, http://cyberaide.org/papers/vonLaszewski-i523-v3.pdf
- Big Data Projects, Gregor von Laszewski, Spring 2017, https://github.com/cloudmesh/sp17-i524/blob/master/project/projects.pdf
- Vol 9, Gregor von Laszewski, Spring 2018, http://cyberaide.org/papers/vonLaszewski-cloud-vol-9.pdf

1.12 Datasets

Below are links to collections of datasets that may be of use for homework assignments or projects.

- https://www.data.gov/
- https://github.com/caesar0301/awesome-public-datasets
- https://aws.amazon.com/public-data-sets/
- https://www.kaggle.com/datasets
- https://cloud.google.com/bigquery/public-data/github
- https://www.quora.com/Where-can-I-find-large-datasets-open-to-the-public

For NIST Projects:

- NIST Special Database 27A - 4GB
- INRIA Person Dataset
- Healthcare data from CMS
- Uber Ride Sharing GPS Data
- Census Data

1.13 Assignments

1.13.1 Due dates

For due dates see the calendar section.

1.13.2 Terminology

Dependent on the class you need to do different assignments. The assignments will be clearly posted. In case of questions, we will update this document to provide clarifications if needed. We use the following terminology:

License:

All projects are developed under an open source license such as Apache 2.0 License. You will be required to add a LICENCE.txt file and if you use other software identify how it can be reused in your project. If your project uses different licenses, please add in a README.md file which packages are used and which license these packages have.

Sections:

Sections are written in markdown and include information on a particular technical issue that is in general helpful for other students. Sections must be about a substantial topic and include an introduction a section that teaches a reader a significant issue, as well as practical code examples. Multiple small sections can lead to a substantial contribution. We expect that the sections are of high quality and can be included in our handbooks. Please be careful of plagiarism and do not just copy the sections from tutorials or code or from elsewhere.

Technology or Review Paper :

A technology paper is a summary paper about a technology, application, or topic that is not yet covered in other technology papers delivered by previous students of this class. A review paper is a paper that reviews a specific topic related to this class.

In either case includes useful information that provides an overview of what you are trying to describe and analyses its relationship to the class topic. Be mindful about plagiarism. The paper is written in LaTeX or Markdown and uses bibtex for bibliography management. It uses the same format as your report paper. The format is discussed in the Section Report Format.

A technology paper is 2 pages long. This will make it between 2000-2400 words.

Note: that for the 2018 we decided to just us Markdown and not LaTeX. We will calculate the exact number of words needed.

Project:

We refer with the term project to the major activity that you chose as part of your class. The default case is an implementation project that requires a project report and project code. In case you have issues with code development you can also chose a term paper as project.

Term Paper:

A term paper is an enhanced topic paper (only available for I523). The difference is in length and depth of coverage. Comparative or review papers can also become term papers. In case you chose the term paper, you or your team will pick a topic relevant for the class. Term papers should have the quality to be publishable either in a workshop or as part of the handbook. Not all classes allow you to do a term paper, but require you to do a project. Please confirm with your class. For the classes listed here the term paper wil result in a quarter reduction in grade for the entire class not just the paper. Remember tables and figures do not count towards the paper length. A term paper has the following length.

- 8 pages, one student in the project
- 10 pages, two student in the project
- 12 pages, three student in the project

We estimate that a single page is between 1000-1200 words. Please note that for 2018 the format will be markdown, so the word count will be used instead.

Project Report:

A project report is an enhanced topic paper that includes not just the analysis of a topic, but an actual code, with **benchmark** and demonstrated application use. Obviously it is longer than a term paper and includes descriptions about reproducibility of the application. A README is provided that describes in a section how others can reproduce your project and run it. Term papers should have the quality to be publishable either in a workshop or as part of the handbook. The format is discussed in the Section Report Format. Remember tables and figures do not count towards the paper length. The following length is required:

- 6 pages, one student in the project
- 8 pages, two students in the project
- 10 pages, three students in the project

We estimate that a single page is between 1000-1200 words. Please note that for 2018 the format will be markdown, so the word count will be used instead.

Project Code:

This is the **documented** and **reproducible** code and scripts that allows a TA do replicate the project. In case you use images they must be created from scratch locally and may not be uploaded to services such as dockerhub. You can however reuse vendor uploaded images such as from ubuntu or centos. All code, scripts, and documentation must be uploaded to github.com under the class specific github directory.

Data:

Data is to be hosted on IUs google drive if needed. If you have larger data, it should be downloaded from the internet. It is in your responsibility to develop a download program. The data **must** not be stored in github. You will be expected to write a python program that downloads the data.

Work Breakdown:

This is an appendix to the document that describes in detail who did what in the project. This section comes in a new page after the references. It does not count towards the page length of the document. It also includes explicit URLs to the git history that documents the statistics to demonstrate not only one student has worked on the project. If you can not provide such a statistic or all check-ins have been made by a single student, the project has shown that they have not properly used git. Thus points will be deducted from the project. Furthermore, if we detect that a student has not contributed to a project we may invite the student to give a detailed presentation of the project.

Bibliography:

All bibliography has to be provided in a jabref/bibtex file. This is regardless if you use LaTeX or Word. There is **NO EXCEPTION** to this rule. Please be advised doing references right takes some time so you want to do this early. Please note that exports of Endnote or other bibliography management tools do not lead to properly formatted bibtex files, despite they claiming to do so. You will have to clean them up and we recommend to do it the other way around. Manage your bibliography with jabref, and if you like to use it import them to endnote or other tools. Naturally you may have to do some cleanup to. If you use LaTeX and jabref, you have naturally much less work to do. What you chose is up to you.

1.13.2.1 Project Deliverables

The objective of the project is to define a clear problem statement and create a framework to address that problem as it relates to big data your project must address the reproducibility of the deployment and the application. A dataset must be chosen and you can analyze the data. YOu must make sure your project can be deployed on the TAs computer through scripts that make your project reproducible.

You have plenty of time to make this choice and if you find you struggle with programming you may want to consider a term paper instead of a project.

In case you chose a project your maximum grade for the entire class could be an A+. However, an A+ project must be truly outstanding and include an exceptional project report. Such a project and report will have the potential quality of being able to be published in a conference or workshop/

In case you chose a term paper your maximum grade for the entire class will be an A-.

1.13.2.1.0.1 Deliverables

- Find a data set with reasonable size (this may depend on your resources and needs to include a benchmark in your paper for justification).
- Clean up the data set or make it smaller or find a bigger data set
- Identify existing algorithms and tools and technologies that you can use to analyze your data
- Provide benchmarks.
- Take results in two different cloud services and your local PC (ex: Chameleon Cloud, echo kubernetes). Make sure your system can be created and deployed based on your documentation.
- Create a Makefile with the tags deploy, run, kill, view, clean that deploys your environment, runs application, kills it, views the result and cleans up after wards. You are allowed to have different makefiles for the different clouds and different directories. Keep the code and directory structure clean and document how to reproduce your results.
- For python use a requirements.txt file also
- For docker use a Dockerfile also
- Write a report that includes the following sections
    - Abstract
    - Introduction
    - Architecture
    - Implementation
        - Technologies Used
    - Design
    - Implementation
    - Results
        - Deployment Benchmarks

        - Application Benchmarks
    - (Limitations)
    - Conclusion
    - (Work Breakdown)

- Your paper will not have a future work section as this implies that you will do work in future, instead you can use an optional limitations section.

1.13.2.2 Group work

You are allowed to work on any assignment in class in groups up to 3 team members. We will not allow more team members as previous examples showed that more team members do not result in better projects than thise delivered with 3 team members.

The assignment is only to be added into github by one team member. Please make sure that you do pull requests to the repository of that team member. If your team likes direct access to the repo form the lead, please communicate this in a private post to piazza with the github user names and we will add the team members. The lea should be aware that in this case all team members have access to all files from the team leader, not only that assignment. If the team leader does not like this, the team should stick with pull requests that the team lead coordinates and integrates.

Groups are expected to have significantly better artifacts than a single student. It is not the goal of the group to deliver a project or paper that is done by n people but could have been done by a single person. Therefore the requirements of all group projects are increased accordingly. Typically this is not an issue. Make sure all team members contribute.

Group requirements Technology summaries:

If you have n team members together you need to have at least 4 * n technologies. The technologies that have been assigned to each team member will have to be completed. YOu can work in collaboration on the technologies, but you need to place all hids that worked on the technology in the headline.

Group requirements 2-page Technology or Review paper:

Option multiple papers. If you have n team members you need to write n different papers each of which has 2 pages. The n team members can be authoring the n papers jointly. Put your hids and names in the paper

Option one large paper. If you have n team members you need to write one large paper with 2 * n pages. The paper must be well written and integrated and not just the concatenation of 2 pages from each author.

Group requirements project paper:

The requirements are clearly stated in another section of the epub.

2 Introduction to Big Data Applications

This is an overview course of Big Data Applications covering a broad range of problems and solutions. It covers cloud computing technologies and includes a project. Also, algorithms are introduced and illustrated.

2.1 General Remarks Including Hype cycles

This is Part 1 of the introduction. We start with some general remarks and take a closer look at the emerging technology hype cycles.

1.a Gartner's Hypecycles and especially those for emerging technologies between 2016 and 2018

1.b Gartner's Hypecycles with Emerging technologies hypecycles and the priority matrix at selected times 2008-2015

1.a Gartner's Hypecycles and especially those for emerging technologies between 2016 and 2018

1.b Gartner's Hypecycles with emerging technologies hypecycles and the priority matrix at selected times 2008-2015

1.a + 1.b

- Technology trends
- Industry reports

2.2 Data Deluge

This is Part 2 of the introduction.

2.a Business usage patterns from NIST

2.b Cyberinfrastructure and AI

2.a Business usage patterns from NIST

2.b Cyberinfrastructure and AI

2.a + 2.b

- Several examples of rapid data and information growth in different areas
- Value of data and analytics

2.3 Jobs

This is Part 3 of the introduction.

3. Jobs

3. Jobs

3. Jobs

- Jobs opportunities in the areas: data science, clouds and computer science and computer engineering
- Jobs demands in different countries and companies.
- Trends and forecast of jobs demands in the future.

2.4 Industry Trends

This is Part 4 of the introduction.

4a. Industry Trends: Technology Trends by 2014

4b. Industry Trends: 2015 onwards

An older set of trend slides is available from:

4a. Industry Trends: Technology Trends by 2014

A current set is available at:

4b. Industry Trends: 2015 onwards

4b. Industry Trends: 2015 onwards

4c. Industry Trends: Voice and HCI, cars,Deep learning

- Many technology trends through end of 2014 and 2015 onwards, examples in different fields
- Voice and HCI, Cars Evolving and Deep learning

2.5 Digital Disruption and Transformation

This is Part 5 of the introduction.

5. Digital Disruption and Transformation

5. Digital Disruption and Transformation

5. Digital Disruption and Transformation

- The past displaced by digital disruption

2.6 Computing Model

This is Part 6 of the introduction.

6a. Computing Model: earlier discussion by 2014

6a. Computing Model: earlier discussion by 2014

6a. Computing Model: earlier discussion by 2014

6b. Computing Model: developments after 2014 including Blockchain

6b. Computing Model: developments after 2014 including Blockchain

6b. Computing Model: developments after 2014 including Blockchain

- Industry adopted clouds which are attractive for data analytics, including big companies, examples are Google, Amazon, Microsoft and so on.
- Some examples of development: AWS quarterly revenue, critical capabilities public cloud infrastructure as a service.
- Blockchain: ledgers redone, blockchain consortia.

2.7 Research Model

This is Part 7 of the introduction.

7. Research Model: 4th Paradigm; From Theory to Data driven science?

7. Research Model: 4th Paradigm; From Theory to Data driven science?

7. Research Model: 4th Paradigm; From Theory to Data driven science?

- The 4 paradigm of scientific research: Theory,Experiment and observation,Simulation of theory or model,Data-driven.

2.8 Data Science Pipeline

This is Part 8 of the introduction.

8. Data Science Pipeline

8. Data Science Pipeline

- DIKW process:Data, Information, Knowledge, Wisdom and Decision.
- Example of Google Maps/navigation.
- Criteria for Data Science platform.

2.9 Physics as an Application Example

This is Part 9 of the introduction.

9. Physics as an Application Example

- Physics as an application example.

2.10 Technology Example

This is Part 10 of the introduction.

10. Technology Example: Recommender Systems I

- Overview of many informatics areas, recommender systems in detail.
- NETFLIX on personalization, recommendation, datascience.

2.11 Exploring Data Bags and Spaces

This is Part 11 of the introduction.

11. Exploring data bags and spaces: Recommender Systems II

11. Exploring data bags and spaces: Recommender Systems II

- Distances in funny spaces, about "real" spaces and how to use distances.

2.12 Another Example: Web Search Information Retrieval

This is Part 12 of the introduction.

12. Another Example: Web Search Information Retrieval

12. Another Example: Web Search Information Retrieval

2.13 Cloud Application in Research

This is Part 13 of the introduction discussing cloud applications in research.

13. Cloud Applications in Research: Science Clouds and Internet of Things

2.14 Software Ecosystems: Parallel Computing and MapReduce

This is Part 14 of the introduction discussing the software ecosystem

14. Software Ecosystems: Parallel Computing and MapReduce

2.15 Conclusions

This is Part 15 of the introduction with some concluding remarks.

15. Conclusions

15. Conclusions

15. Conclusions

3 Overview of Data Science

What is Big Data, Data Analytics and X-Informatics?

We start with X-Informatics and its rallying cry. The growing number of jobs in data science is highlighted. The first unit offers a look at the phenomenon described as the Data Deluge starting with its broad features. Data science and the famous DIKW (Data to Information to Knowledge to Wisdom) pipeline are covered. Then more detail is given on the flood of data from Internet and Industry applications with eBay and General Electric discussed in most detail.

In the next unit, we continue the discussion of the data deluge with a focus on scientific research. He takes a first peek at data from the Large Hadron Collider considered later as physics Informatics and gives some biology examples. He discusses the implication of data for the scientific method which is changing with the data-intensive methodology joining observation, theory and simulation as basic methods. Two broad classes of data are the long tail of sciences: many users with individually modest data adding up to a lot; and a myriad of Internet connected devices – the Internet of Things.

We give an initial technical overview of cloud computing as pioneered by companies like Amazon, Google and Microsoft with new centers holding up to a million servers. The benefits of Clouds in terms of power consumption and the environment are also touched upon, followed by a list of the most critical features of Cloud computing with a comparison to supercomputing. Features of the data deluge are discussed with a salutary example where more data did better than more thought. Then comes Data science and one part of it ~~ data analytics ~~ the large algorithms that crunch the big data to give big wisdom. There are many ways to describe data science and several are discussed to give a good composite picture of this emerging field.

3.1 Data Science generics and Commercial Data Deluge

We start with X-Informatics and its rallying cry. The growing number of jobs in data science is highlighted. This unit offers a look at the phenomenon described as the Data Deluge starting with its broad features. Then he discusses data science and the famous DIKW (Data to Information to Knowledge to Wisdom) pipeline. Then more detail is given on the flood of data from Internet and Industry applications with eBay and General Electric discussed in most detail.

Commercial Data Deluge (45)

3.1.1 What is X-Informatics and its Motto

This discusses trends that are driven by and accompany Big data. We give some key terms including data, information, knowledge, wisdom, data analytics and data science. We discuss how clouds running Data Analytics Collaboratively processing Big Data can solve problems in X-Informatics. We list many values of X you can defined in various activities across the world.

- X Informatics (9:49)

3.1.2 Jobs

Big data is especially important as there are some many related jobs. We illustrate this for both cloud computing and data science from reports by Microsoft and the McKinsey institute respectively. We show a plot from LinkedIn showing rapid increase in the number of data science and analytics jobs as a function of time.

- Jobs (2:58)

3.1.3 Data Deluge: General Structure

We look at some broad features of the data deluge starting with the size of data in various areas especially in science research. We give examples from real world of the importance of big data and illustrate how it is integrated into an enterprise IT architecture. We give some views as to what characterizes Big data and why data science is a science that is needed to interpret all the data.

- Data Deluge (13:04)

3.1.4 Data Science: Process

We stress the DIKW pipeline: Data becomes information that becomes knowledge and then wisdom, policy and decisions. This pipeline is illustrated with Google maps and we show how complex the ecosystem of data, transformations (filters) and its derived forms is.

- Data Science Process (4:27)

3.1.5 Data Deluge: Internet

We give examples of Big data from the Internet with Tweets, uploaded photos and an illustration of the vitality and size of many commodity applications.

- Internet (3:42)

3.1.6 Data Deluge: Business

We give examples including the Big data that enables wind farms, city transportation, telephone operations, machines with health monitors, the banking, manufacturing and retail industries both online and offline in shopping malls. We give examples from ebay showing how analytics allowing them to refine and improve the customer experiences.

- Business I (6:00)
- Business II (7:34)
- Business III (9:37)

3.1.7 Resources

- http://www.microsoft.com/en-us/news/features/2012/mar12/03-05CloudComputingJobs.aspx
- http://www.mckinsey.com/mgi/publications/big_data/index.asp
- Tom Davenport
- Anjul Bhambhri
- Jeff Hammerbacher
- http://www.economist.com/node/15579717
- http://cs.metrostate.edu/~sbd/slides/Sun.pdf
- http://jess3.com/geosocial-universe-2/
- Bill Ruh
- http://www.hsph.harvard.edu/ncb2011/files/ncb2011-z03-rodriguez.pptx
- Hugh Williams

3.2 Data Deluge and Scientific Applications and Methodology

3.2.1 Overview of Data Science

We continue the discussion of the data deluge with a focus on scientific research. He takes a first peek at data from the Large Hadron Collider considered later as physics Informatics and gives some biology examples. He discusses the implication of data for the scientific method which is changing with the data-intensive methodology joining observation, theory and simulation as basic methods. We discuss the long tail of sciences; many users with individually modest data adding up to a lot. The last lesson emphasizes how everyday devices ~~ the Internet of Things ~~ are being used to create a wealth of data.

Methodology (22)

3.2.2 Science and Research

We look into more big data examples with a focus on science and research. We give astronomy, genomics, radiology, particle physics and discovery of Higgs particle (Covered in more detail in later lessons), European Bioinformatics Institute and contrast to Facebook and Walmart.

- Science and Research (11:27)
- Science and Research (11:49)

3.2.3 Implications for Scientific Method

We discuss the emergencies of a new fourth methodology for scientific research based on data driven inquiry. We contrast this with third ~~ computation or simulation based discovery - methodology which emerged itself some 25 years ago.

- Scientific Methods (5:07)

3.2.4 Long Tail of Science

There is big science such as particle physics where a single experiment has 3000 people collaborate!.Then there are individual investigators who do not generate a lot of data each but together they add up to Big data.

- Long Tail of Science (2:10)

3.2.5 Internet of Things

A final category of Big data comes from the Internet of Things where lots of small devices ~~ smart phones, web cams, video games collect and disseminate data and are controlled and coordinated in the cloud.

- Internet of Things (5:45)

3.2.6 Resources

- http://www.economist.com/node/15579717
- Geoffrey Fox and Dennis Gannon Using Clouds for Technical Computing To be published in Proceedings of HPC 2012 Conference at Cetraro, Italy June 28 2012
- http://grids.ucs.indiana.edu/ptliupages/publications/Clouds_Technical_Computing_FoxGannonv2.pdf
- http://grids.ucs.indiana.edu/ptliupages/publications/Where%20does%20all%20the%20data%20come%20from%20v7.pdf
- http://www.genome.gov/sequencingcosts/
- http://www.quantumdiaries.org/2012/09/07/why-particle-detectors-need-a-trigger/atlasmgg
- http://salsahpc.indiana.edu/dlib/articles/00001935/
- http://en.wikipedia.org/wiki/Simple_linear_regression
- http://www.ebi.ac.uk/Information/Brochures/
- http://www.wired.com/wired/issue/16-07
- http://research.microsoft.com/en-us/collaboration/fourthparadigm/
- CSTI General Assembly 2012, Washington, D.C., USA Technical Activities Coordinating Committee (TACC) Meeting, Data Management, Cloud Computing and the Long Tail of Science October 2012 Dennis Gannon

3.3 Clouds and Big Data Processing; Data Science Process and Analytics

3.3.1 Overview of Data Science

We give an initial technical overview of cloud computing as pioneered by companies like Amazon, Google and Microsoft with new centers holding up to a million servers. The benefits of Clouds in terms of power consumption and the environment are also touched upon, followed by a list of the most critical features of Cloud computing with a comparison to supercomputing.

He discusses features of the data deluge with a salutary example where more data did better than more thought. He introduces data science and one part of it ~~ data analytics ~~ the large algorithms that crunch the big data to give big wisdom. There are many ways to describe data science and several are discussed to give a good composite picture of this emerging field.

Clouds (35)

3.3.2 Clouds

We describe cloud data centers with their staggering size with up to a million servers in a single data center and centers built modularly from shipping containers full of racks. The benefits of Clouds in terms of power consumption and the environment are also touched upon, followed by a list of the most critical features of Cloud computing and a comparison to supercomputing.

- Clouds (16:04){MP4}

3.3.3 Aspect of Data Deluge

Data, Information, intelligence algorithms, infrastructure, data structure, semantics and knowledge are related. The semantic web and Big data are compared. We give an example where “More data usually beats better algorithms”. We discuss examples of intelligent big data and list 8 different types of data deluge

- Data Deluge (8:02)
- Data Deluge (6:24)

3.3.4 Data Science Process

We describe and critique one view of the work of a data scientists. Then we discuss and contrast 7 views of the process needed to speed data through the DIKW pipeline.

- Scientific Process (11:28)

3.3.5 Data Analytics

Data Analytics (30) We stress the importance of data analytics givi ng examples from several fields. We note that better analytics is as important as better computing and storage capability. In the second video we look at High Performance Computing in Science and Engineering: the Tree and the Fruit.

- Data Analytics (7:28)
- Data Analytics (6:51)

3.3.6 Resources

- CSTI General Assembly 2012, Washington, D.C., USA Technical Activities Coordinating Committee (TACC) Meeting, Data Management, Cloud Computing and the Long Tail of Science October 2012 Dennis Gannon
- Dan Reed Roger Barga Dennis Gannon Rich Wolski http://research.microsoft.com/en-us/people/barga/sc09_cloudcomp_tutorial.pdf
- http://www.datacenterknowledge.com/archives/2011/05/10/uptime-institute-the-average-pue-is-1-8/
- http://loosebolts.wordpress.com/2008/12/02/our-vision-for-generation-4-modular-data-centers-one-way-of-getting-it-just-right/
- http://www.mediafire.com/file/zzqna34282frr2f/koomeydatacenterelectuse2011finalversion.pdf
- Bina Ramamurthy
- Jeff Hammerbacher
- Jeff Hammerbacher
- Anjul Bhambhri
- http://cs.metrostate.edu/~sbd/slides/Sun.pdf
- Hugh Williams
- Tom Davenport
- http://www.mckinsey.com/mgi/publications/big_data/index.asp
- http://cra.org/ccc/docs/nitrdsymposium/pdfs/keyes.pdf

4 Physics

This section starts by describing the LHC accelerator at CERN and evidence found by the experiments suggesting existence of a Higgs Boson. The huge number of authors on a paper, remarks on histograms and Feynman diagrams is followed by an accelerator picture gallery. The next unit is devoted to Python experiments looking at histograms of Higgs Boson production with various forms of shape of signal and various background and with various event totals. Then random variables and some simple principles of statistics are introduced with explanation as to why they are relevant to Physics counting experiments. The unit introduces Gaussian (normal) distributions and explains why they seen so often in natural phenomena. Several Python illustrations are given. Random Numbers with their Generators and Seeds lead to a discussion of Binomial and Poisson Distribution. Monte-Carlo and accept-reject methods. The Central Limit Theorem concludes discussion.

4.1 Looking for Higgs Particles

4.1.1 Bumps in Histograms, Experiments and Accelerators

This unit is devoted to Python and Java experiments looking at histograms of Higgs Boson production with various forms of shape of signal and various background and with various event totals. The lectures use Python but use of Java is described.

- Higgs (20)
- https://github.com/cloudmesh/book/tree/master/examples/physics/mr-higgs/higgs-classI-sloping.py

4.1.2 Particle Counting

We return to particle case with slides used in introduction and stress that particles often manifested as bumps in histograms and those bumps need to be large enough to stand out from background in a statistically significant fashion.

- Discovery of Higgs Particle (13:49)

We give a few details on one LHC experiment ATLAS. Experimental physics papers have a staggering number of authors and quite big budgets. Feynman diagrams describe processes in a fundamental fashion.

- Looking for Higgs Particle and Counting Introduction II (7:38)

4.1.3 Experimental Facilities

We give a few details on one LHC experiment ATLAS. Experimental physics papers have a staggering number of authors and quite big budgets. Feynman diagrams describe processes in a fundamental fashion.

- Looking for Higgs Particle Experiments (9:29)

4.1.4 Accelerator Picture Gallery of Big Science

This lesson gives a small picture gallery of accelerators. Accelerators, detection chambers and magnets in tunnels and a large underground laboratory used fpr experiments where you need to be shielded from background like cosmic rays.

- Accelerator Picture Gallery of Big Science (11:21)

4.1.5 Resources

- http://grids.ucs.indiana.edu/ptliupages/publications/Where%20does%20all%20the%20data%20come%20from%20v7.pdf [2]
- http://www.sciencedirect.com/science/article/pii/S037026931200857X [3]
- http://www.nature.com/news/specials/lhc/interactive.html

Looking for Higgs Particles: Python Event Counting for Signal and Background (Part 2)

This unit is devoted to Python experiments looking at histograms of Higgs Boson production with various forms of shape of signal and various background and with various event totals.

- Higgs II (29)

Files:

- https://github.com/cloudmesh/book/tree/master/examples/physics/mr-higgs/higgs-classI-sloping.py
- https://github.com/cloudmesh/book/tree/master/examples/physics/number-theory/higgs-classIII.py
- https://github.com/cloudmesh/book/tree/master/examples/physics/mr-higgs/higgs-classII-uniform.py

4.1.6 Event Counting

We define event counting data collection environments. We discuss the python and Java code to generate events according to a particular scenario (the important idea of Monte Carlo data). Here a sloping background plus either a Higgs particle generated similarly to LHC observation or one observed with better resolution (smaller measurement error).

- Event Counting (7:02)

4.1.7 Monte Carlo

This uses Monte Carlo data both to generate data like the experimental observations and explore effect of changing amount of data and changing measurement resolution for Higgs.

- With Python examples of Signal plus Background (7:33) This lesson continues the examination of Monte Carlo data looking at effect of change in number of Higgs particles produced and in change in shape of background.
- Change shape of background & num of Higgs Particles (7:01)

4.1.8 Resources

- Python for Data Analysis: Agile Tools for Real World Data By Wes McKinney, Publisher: O'Reilly Media, Released: October 2012, Pages: 472. [4]
- http://jwork.org/scavis/api/ [5]
- https://en.wikipedia.org/wiki/DataMelt **???**

4.1.9 Random Variables, Physics and Normal Distributions

We introduce random variables and some simple principles of statistics and explains why they are relevant to Physics counting experiments. The unit introduces Gaussian (normal) distributions and explains why they seen so often in natural phenomena. Several Python illustrations are given. Java is currently not available in this unit.

- Higgs (39)
- https://github.com/cloudmesh/book/tree/master/examples/physics/number-theory/higgs-classIII.py

4.1.10 Statistics Overview and Fundamental Idea: Random Variables

We go through the many different areas of statistics covered in the Physics unit. We define the statistics concept of a random variable.

- Random variables and normal distributions (8:19)

4.1.11 Physics and Random Variables

We describe the DIKW pipeline for the analysis of this type of physics experiment and go through details of analysis pipeline for the LHC ATLAS experiment. We give examples of event displays showing the final state particles seen in a few events. We illustrate how physicists decide whats going on with a plot of expected Higgs production experimental cross sections (probabilities) for signal and background.

- Physics and Random Variables I (8:34)
- Physics and Random Variables II (5:50)

4.1.12 Statistics of Events with Normal Distributions

We introduce Poisson and Binomial distributions and define independent identically distributed (IID) random variables. We give the law of large numbers defining the errors in counting and leading to Gaussian distributions for many things. We demonstrate this in Python experiments.

- Statistics of Events with Normal Distributions (11:25)

4.1.13 Gaussian Distributions

We introduce the Gaussian distribution and give Python examples of the fluctuations in counting Gaussian distributions.

- Gaussian Distributions (9:08)

4.1.14 Using Statistics

We discuss the significance of a standard deviation and role of biases and insufficient statistics with a Python example in getting incorrect answers.

- Using Statistics (14:02)

4.1.15 Resources

- http://indico.cern.ch/event/20453/session/6/contribution/15?materialId=slides
- ~~http://www.atlas.ch/photos/events.html~~ (this link is outdated)
- https://cms.cern/ [6]

4.1.16 Random Numbers, Distributions and Central Limit Theorem

We discuss Random Numbers with their Generators and Seeds. It introduces Binomial and Poisson Distribution. Monte-Carlo and accept-reject methods are discussed. The Central Limit Theorem and Bayes law concludes discussion. Python and Java (for student - not reviewed in class) examples and Physics applications are given.

- Higgs III (44)

Files:

- https://github.com/cloudmesh/book/tree/master/examples/physics/calculated-dice-roll/higgs-classIV-seeds.py

4.1.16.1 Generators and Seeds

We define random numbers and describe how to generate them on the computer giving Python examples. We define the seed used to define to specify how to start generation.

- Higgs Particle Counting Errors (6:28)
- Generators and Seeds II (7:10)

4.1.16.2 Binomial Distribution

We define binomial distribution and give LHC data as an example of where this distribution valid.

- Binomial Distribution: (12:38)

4.1.16.3 Accept-Reject

We introduce an advanced method **accept/reject** for generating random variables with arbitrary distributions.

- Accept-Reject (5:54)

4.1.16.4 Monte Carlo Method

We define Monte Carlo method which usually uses accept/reject method in typical case for distribution.

- Monte Carlo Method (2:23)

4.1.16.5 Poisson Distribution

We extend the Binomial to the Poisson distribution and give a set of amusing examples from Wikipedia.

- Poisson Distribution (4:37)

4.1.16.6 Central Limit Theorem

We introduce Central Limit Theorem and give examples from Wikipedia.

- Central Limit Theorem (4:47)

4.1.16.7 Interpretation of Probability: Bayes v. Frequency

This lesson describes difference between Bayes and frequency views of probability. Bayes's law of conditional probability is derived and applied to Higgs example to enable information about Higgs from multiple channels and multiple experiments to be accumulated.

- Interpretation of Probability (12:39)

4.1.16.8 Resources

4.2 SKA – Square Kilometer Array

Professor Diamond, accompanied by Dr. Rosie Bolton from the SKA Regional Centre Project gave a presentation at SC17 "into the deepest reaches of the observable universe as they describe the SKA's international partnership that will map and study the entire sky in greater detail than ever before."

- http://sc17.supercomputing.org/presentation/?id=inspkr101&sess=sess263

A summary article about this effort is available at:

- https://www.hpcwire.com/2017/11/17/sc17-keynote-hpc-powers-ska-efforts-peer-deep-cosmos/ The video is hosted at
- http://sc17.supercomputing.org/presentation/?id=inspkr101&sess=sess263 Start at about 1:03:00 (e.g. the one hour mark)

5 e-Commerce and LifeStyle

Recommender systems operate under the hood of such widely recognized sites as Amazon, eBay, Monster and Netflix where everything is a recommendation. This involves a symbiotic relationship between vendor and buyer whereby the buyer provides the vendor with information about their preferences, while the vendor then offers recommendations tailored to match their needs. Kaggle competitions h improve the success of the Netflix and other recommender systems. Attention is paid to models that are used to compare how changes to the systems affect their overall performance. It is interesting that the humble ranking has become such a dominant driver of the world's economy. More examples of recommender systems are given from Google News, Retail stores and in depth Yahoo! covering the multi-faceted criteria used in deciding recommendations on web sites.

The formulation of recommendations in terms of points in a space or bag is given where bags of item properties, user properties, rankings and users are useful. Detail is given on basic principles behind recommender systems: user-based collaborative filtering, which uses similarities in user rankings to predict their interests, and the Pearson correlation, used to statistically quantify correlations between users viewed as points in a space of items. Items are viewed as points in a space of users in item-based collaborative filtering. The Cosine Similarity is introduced, the difference between implicit and explicit ratings and the k Nearest Neighbors algorithm. General features like the curse of dimensionality in high dimensions are discussed. A simple Python k Nearest Neighbor code and its application to an artificial data set in 3 dimensions is given. Results are visualized in Matplotlib in 2D and with Plotviz in 3D. The concept of a training and a testing set are introduced with training set pre labeled. Recommender system are used to discuss clustering with k-means based clustering methods used and their results examined in Plotviz. The original labelling is compared to clustering results and extension to 28 clusters given. General issues in clustering are discussed including local optima, the use of annealing to avoid this and value of heuristic algorithms.

5.1 Recommender Systems

We introduce Recommender systems as an optimization technology used in a variety of applications and contexts online. They operate in the background of such widely recognized sites as Amazon, eBay, Monster and Netflix where everything is a recommendation. This involves a symbiotic relationship between vendor and buyer whereby the buyer provides the vendor with information about their preferences, while the vendor then offers recommendations tailored to match their needs, to the benefit of both.

There follows an exploration of the Kaggle competition site, other recommender systems and Netflix, as well as competitions held to improve the success of the Netflix recommender system. Finally attention is paid to models that are used to compare how changes to the systems affect their overall performance. It is interesting how the humble ranking has become such a dominant driver of the world's economy.

Lifestyle Recommender (45)

5.1.1 Recommender Systems as an Optimization Problem

We define a set of general recommender systems as matching of items to people or perhaps collections of items to collections of people where items can be other people, products in a store, movies, jobs, events, web pages etc. We present this as "yet another optimization problem".

Recommender Systems I (8:06)

5.1.2 Recommender Systems Introduction

We give a general discussion of recommender systems and point out that they are particularly valuable in long tail of tems (to be recommended) that are not commonly known. We pose them as a rating system and relate them to information retrieval rating systems. We can contrast recommender systems based on user profile and context; the most familiar collaborative filtering of others ranking; item properties; knowledge and hybrid cases mixing some or all of these.

Recommender Systems Introduction (12:56)

5.1.3 Kaggle Competitions

We look at Kaggle competitions with examples from web site. In particular we discuss an Irvine class project involving ranking jokes.

Kaggle Competitions: (3:36)

⚠ Please not that we typically do not accept any projects using kaggle data for this classes. This class is not about winning a kaggle competition and if done wrong it does not fullfill the minimum requiremnt for this class. Please consult with the instructor.

5.1.4 Examples of Recommender Systems

We go through a list of 9 recommender systems from the same Irvine class.

Examples of Recommender Systems (1:00)

5.1.5 Netflix on Recommender Systems

We summarize some interesting points from a tutorial from Netflix for whom everything is a recommendation. Rankings are given in multiple categories and categories that reflect user interests are especially important. Criteria used include explicit user preferences, implicit based on ratings and hybrid methods as well as freshness and diversity. Netflix tries to explain the rationale of its recommendations. We give some data on Netflix operations and some methods used in its recommender systems. We describe the famous Netflix Kaggle competition to improve its rating system. The analogy to maximizing click through rate is given and the objectives of optimization are given.

Netflix on Recommender Systems (14:20)

Next we go through Netflix's methodology in letting data speak for itself in optimizing the recommender engine. An example iis given on choosing self produced movies. A/B testing is discussed with examples showing how testing does allow optimizing of sophisticated criteria. This lesson is concluded by comments on Netflix technology and the full spectrum of issues that are involved including user interface, data, AB testing, systems and architectures. We comment on optimizing for a household rather than optimizing for individuals in household.

Consumer Data Science (13:04)

5.1.6 Other Examples of Recommender Systems

We continue the discussion of recommender systems and their use in e-commerce. More examples are given from Google News, Retail stores and in depth Yahoo! covering the multi-faceted criteria used in deciding recommendations on web sites. Then the formulation of recommendations in terms of points in a space or bag is given.

Here bags of item properties, user properties, rankings and users are useful. Then we go into detail on basic principles behind recommender systems: user-based collaborative filtering, which uses similarities in user rankings to predict their interests, and the Pearson correlation, used to statistically quantify correlations between users viewed as points in a space of items.

Lifestyle Recommender (49)

We start with a quick recap of recommender systems from previous unit; what they are with brief examples.

Recap and Examples of Recommender Systems (5:48)

5.1.6.1 Examples of Recommender Systems

We give 2 examples in more detail: namely Google News and Markdown in Retail.

Examples of Recommender Systems (8:34)

5.1.6.2 Recommender Systems in Yahoo Use Case Example

We describe in greatest detail the methods used to optimize Yahoo web sites. There are two lessons discussing general approach and a third lesson examines a particular personalized Yahoo page with its different components. We point out the different criteria that must be blended in making decisions; these criteria include analysis of what user does after a particular page is clicked; is the user satisfied and cannot that we quantified by purchase decisions etc. We need to choose Articles, ads, modules, movies, users, updates, etc to optimize metrics such as relevance score, CTR, revenue, engagement.These lesson stress that if though we have big data, the recommender data is sparse. We discuss the approach that involves both batch (offline) and on-line (real time) components.

Recap of Recommender Systems II (8:46)

Recap of Recommender Systems III (10:48)

Case Study of Recommender systems (3:21)

5.1.6.3 User-based nearest-neighbor collaborative filtering

Collaborative filtering is a core approach to recommender systems. There is user-based and item-based collaborative filtering and here we discuss the user-based case. Here similarities in user rankings allow one to predict their interests, and typically this quantified by the Pearson correlation, used to statistically quantify correlations between users.

User-based nearest-neighbor collaborative filtering I (7:20)

User-based nearest-neighbor collaborative filtering II (7:29)

5.1.6.4 Vector Space Formulation of Recommender Systems

We go through recommender systems thinking of them as formulated in a funny vector space. This suggests using clustering to make recommendations.

Vector Space Formulation of Recommender Systems new (9:06)

5.1.7 Resources

- http://pages.cs.wisc.edu/~beechung/icml11-tutorial/

5.2 Item-based Collaborative Filtering and its Technologies

We move on to item-based collaborative filtering where items are viewed as points in a space of users. The Cosine Similarity is introduced, the difference between implicit and explicit ratings and the k Nearest Neighbors algorithm. General features like the curse of dimensionality in high dimensions are discussed.

Lifestyle Filtering (18)

5.2.1 Item-based Collaborative Filtering

We covered user-based collaborative filtering in the previous unit. Here we start by discussing memory-based real time and model based offline (batch) approaches. Now we look at item-based collaborative filtering where items are viewed in the space of users and the cosine measure is used to quantify distances. WE discuss optimizations and how batch processing can help. We discuss different Likert ranking scales and issues with new items that do not have a significant number of rankings.

Item Based Filtering (11:18)

k Nearest Neighbors and High Dimensional Spaces (7:16)

5.2.2 k-Nearest Neighbors and High Dimensional Spaces

We define the k Nearest Neighbor algorithms and present the Python software but do not use it. We give examples from Wikipedia and describe performance issues. This algorithm illustrates the curse of dimensionality. If items were a real vectors in a low dimension space, there would be faster solution methods.

k Nearest Neighbors and High Dimensional Spaces (10:03)

5.2.2.1 Recommender Systems - K-Neighbors

Next we provide some sample Python code for the k Nearest Neighbor and its application to an artificial data set in 3 dimensions. Results are visualized in Matplotlib in 2D and with Plotviz in 3D. The concept of training and testing sets are introduced with training set pre-labelled. This lesson is adapted from the Python k Nearest Neighbor code found on the web associated with a book by Harrington on Machine Learning [??]. There are two data sets. First we consider a set of 4 2D vectors divided into two categories (clusters) and use k=3 Nearest Neighbor algorithm to classify 3 test points. Second we consider a 3D dataset that has already been classified and show how to normalize. In this lesson we just use Matplotlib to give 2D plots.

The lesson goes through an example of using k NN classification algorithm by dividing dataset into 2 subsets. One is training set with initial classification; the other is test point to be classified by k=3 NN using training set. The code records fraction of points with a different classification from that input. One can experiment with different sizes of the two subsets. The Python implementation of algorithm is analyzed in detail.

5.2.2.2 Plotviz

The clustering methods are used and their results examined in Plotviz. The original labelling is compared to clustering results and extension to 28 clusters given. General issues in clustering are discussed including local optima, the use of annealing to avoid this and value of heuristic algorithms.

5.2.2.3 Files

- https://github.com/cloudmesh-community/book/blob/master/examples/python/knn/kNN.py
- https://github.com/cloudmesh-community/book/blob/master/examples/python/knn/kNN_Driver.py
- https://github.com/cloudmesh-community/book/blob/master/examples/python/knn/dating_test_set2.txt
- https://github.com/cloudmesh-community/book/blob/master/examples/python/knn/clusterFinal-M3-C3Dating-ReClustered.pviz
- https://github.com/cloudmesh-community/book/blob/master/examples/python/knn/dating_rating_original_labels.pviz
- https://github.com/cloudmesh-community/book/blob/master/examples/python/knn/clusterFinal-M30-C28.pviz
- https://github.com/cloudmesh-community/book/blob/master/examples/python/plotviz/clusterfinal_m3_c3dating_reclustered.pviz
- https://github.com/cloudmesh-community/book/blob/master/examples/python/plotviz/fungi_lsu_3_15_to_3_26_zeroidx.pviz

5.2.3 Resources k-means

- http://www.slideshare.net/xamat/building-largescale-realworld-recommender-systems-recsys2012-tutorial [7]
- http://www.ifi.uzh.ch/ce/teaching/spring2012/16-Recommender-Systems_Slides.pdf [8]
- https://www.kaggle.com/ [9]
- http://www.ics.uci.edu/~welling/teaching/CS77Bwinter12/CS77B_w12.html [10]
- Jeff Hammerbacher[11]
- http://www.techworld.com/news/apps/netflix-foretells-house-of-cards-success-with-cassandra-big-data-engine-3437514/ [12]
- https://en.wikipedia.org/wiki/A/B_testing [13]
- http://www.infoq.com/presentations/Netflix-Architecture [14]

6 Sports

Sports sees significant growth in analytics with pervasive statistics shifting to more sophisticated measures. We start with baseball as game is built around segments dominated by individuals where detailed (video/image) achievement measures including PITCHf/x and FIELDf/x are moving field into big data arena. There are interesting relationships between the economics of sports and big data analytics. We look at Wearables and consumer sports/recreation. The importance of spatial visualization is discussed. We look at other Sports: Soccer, Olympics, NFL Football, Basketball, Tennis and Horse Racing.

6.1 Basic Sabermetrics

This unit discusses baseball starting with the movie Moneyball and the 2002-2003 Oakland Athletics. Unlike sports like basketball and soccer, most baseball action is built around individuals often interacting in pairs. This is much easier to quantify than many player phenomena in other sports. We discuss Performance-Dollar relationship including new stadiums and media/advertising. We look at classic baseball averages and sophisticated measures like Wins Above Replacement.

Overview (40)

6.1.1 Introduction and Sabermetrics (Baseball Informatics) Lesson

Introduction to all Sports Informatics, Moneyball The 2002-2003 Oakland Athletics, Diamond Dollars economic model of baseball, Performance - Dollar relationship, Value of a Win.

Introduction and Sabermetrics (Baseball Informatics) Lesson (31:4)

6.1.2 Basic Sabermetrics

Different Types of Baseball Data, Sabermetrics, Overview of all data, Details of some statistics based on basic data, OPS, wOBA, ERA, ERC, FIP, UZR.

Basic Sabermetrics (26:53)

6.1.3 Wins Above Replacement

Wins above Replacement WAR, Discussion of Calculation, Examples, Comparisons of different methods, Coefficient of Determination, Another, Sabermetrics Example, Summary of Sabermetrics.

Wins Above Replacement (30:43)

6.2 Advanced Sabermetrics

This unit discusses 'advanced sabermetrics' covering advances possible from using video from PITCHf/X, FIELDf/X, HITf/X, COMMANDf/X and MLBAM.

Sporta II (41)

6.2.1 Pitching Clustering

A Big Data Pitcher Clustering method introduced by Vince Gennaro, Data from Blog and video at 2013 SABR conference.

Pitching Clustering (20:59)

6.2.2 Pitcher Quality

Results of optimizing match ups, Data from video at 2013 SABR conference.

Pitcher Quality (10:02)

6.3 PITCHf/X

Examples of use of PITCHf/X.

PITCHf/X (10:39)

6.3.1 Other Video Data Gathering in Baseball

FIELDf/X, MLBAM, HITf/X, COMMANDf/X.

Other Video Data Gathering in Baseball (18:5) Other Sports

We look at Wearables and consumer sports/recreation. The importance of spatial visualization is discussed. We look at other Sports: Soccer, Olympics, NFL Football, Basketball, Tennis and Horse Racing.

Sport Sports III (44)

6.3.2 Wearables

Consumer Sports, Stake Holders, and Multiple Factors.

Wearables (22:2)

6.3.3 Soccer and the Olympics

Soccer, Tracking Players and Balls, Olympics.

Soccer and the Olympics (8:28)

6.3.4 Spatial Visualization in NFL and NBA

NFL, NBA, and Spatial Visualization.

Spatial Visualization in NFL and NBA (15:19)

6.3.5 Tennis and Horse Racing

Tennis, Horse Racing, and Continued Emphasis on Spatial Visualization.

Tennis and Horse Racing (8:52)

6.3.6 Resources

- http://www.slideshare.net/Tricon_Infotech/big-data-for-big-sports [15]
- http://www.slideshare.net/BrandEmotivity/sports-analytics-innovation-summit-data-powered-storytelling **???**
- http://www.slideshare.net/elew/sport-analytics-innovation [16]
- http://www.wired.com/2013/02/catapault-smartball/ [17]
- http://www.sloansportsconference.com/wp-content/uploads/2014/06/Automated_Playbook_Generation.pdf [18]
- http://autoscout.adsc.illinois.edu/publications/football-trajectory-dataset/ [19]
- http://www.sloansportsconference.com/wp-content/uploads/2012/02/Goldsberry_Sloan_Submission.pdf [20]
- http://gamesetmap.com/ [21]
- http://www.slideshare.net/BrandEmotivity/sports-analytics-innovation-summit-data-powered-storytelling [22]
- http://www.sloansportsconference.com/ [23]
- http://sabr.org/ [24]
- http://en.wikipedia.org/wiki/Sabermetrics [25]
- http://en.wikipedia.org/wiki/Baseball_statistics [26]
- http://m.mlb.com/news/article/68514514/mlbam-introduces-new-way-to-analyze-every-play [27]
- http://www.fangraphs.com/library/offense/offensive-statistics-list/ [28]
- http://en.wikipedia.org/wiki/Component_ERA [29]
- http://www.fangraphs.com/library/pitching/fip/ **???**
- http://en.wikipedia.org/wiki/Wins_Above_Replacement [30]
- http://www.fangraphs.com/library/misc/war/ [31]
- http://www.baseball-reference.com/about/war_explained.shtml [32]

- http://www.baseball-reference.com/about/war_explained_comparison.shtml [33]
- http://www.baseball-reference.com/about/war_explained_position.shtml [34]
- http://www.baseball-reference.com/about/war_explained_pitch.shtml [35]
- http://www.fangraphs.com/leaders.aspx?pos=all&stats=bat&lg=all&qual=y&type=8&season=2014&month=0&season1=1871&ind=0 [36]
- http://battingleadoff.com/2014/01/08/comparing-the-three-war-measures-part-ii/ [37]
- http://en.wikipedia.org/wiki/Coefficient_of_determination [38]
- http://www.sloansportsconference.com/wp-content/uploads/2014/02/2014_SSAC_Data-driven-Method-for-In-game-Decision-Making.pdf [39]
- https://courses.edx.org/courses/BUx/SABR101x/2T2014/courseware/10e616fc7649469ab4457ae18df92b20/
- http://vincegennaro.mlblogs.com/ [40]
- https://www.youtube.com/watch?v=H-kx-x_d0Mk **???**
- http://www.baseballprospectus.com/article.php?articleid=13109 [41]
- http://baseball.physics.illinois.edu/FastPFXGuide.pdf [42]
- http://baseball.physics.illinois.edu/FieldFX-TDR-GregR.pdf [43]
- http://regressing.deadspin.com/mlb-announces-revolutionary-new-fielding-tracking-syste-1534200504 [44]
- http://grantland.com/the-triangle/mlb-advanced-media-play-tracking-bob-bowman-interview/ [45]
- https://www.youtube.com/watch?v=YkjtnuNmK74 [46]

resources below do not exsit:

- ~~http://www.sportvision.com/baseball~~
- ~~http://www.sportvision.com/media/pitchfx-how-it-works~~
- ~~http://www.sportvision.com/baseball/fieldfx~~
- ~~http://www.sportvision.com/baseball/hitfx~~
- ~~http://www.trakus.com/technology.asp#tNetText~~
- ~~http://www.sloansportsconference.com/?page_id=481&sort_cate=Research%20Paper~~
- ~~http://www.liveathos.com/apparel/app~~

7 Cloud Computing

We describe the central role of Parallel computing in Clouds and Big Data which is decomposed into lots of Little data running in individual cores. Many examples are given and it is stressed that issues in parallel computing are seen in day to day life for communication, synchronization, load balancing and decomposition. Cyberinfrastructure for e-moreorlessanything or moreorlessanything-Informatics and the basics of cloud computing are introduced. This includes virtualization and the important as a Service components and we go through several different definitions of cloud computing.

Gartner's Technology Landscape includes hype cycle and priority matrix and covers clouds and Big Data. Two simple examples of the value of clouds for enterprise applications are given with a review of different views as to nature of Cloud Computing. This IaaS (Infrastructure as a Service) discussion is followed by PaaS and SaaS (Platform and Software as a Service). Features in Grid and cloud computing and data are treated. We summarize the 21 layers and almost 300 software packages in the HPC-ABDS Software Stack explaining how they are used.

Cloud (Data Center) Architectures with physical setup, Green Computing issues and software models are discussed followed by the Cloud Industry stakeholders with a 2014 Gartner analysis of Cloud computing providers. This is followed by applications on the cloud including data intensive problems, comparison with high performance computing, science clouds and the Internet of Things. Remarks on Security, Fault Tolerance and Synchronicity issues in cloud follow. We describe the way users and data interact with a cloud system. The Big Data Processing from an application perspective with commercial examples including eBay concludes section after a discussion of data system architectures.

7.1 Parallel Computing (Outdated)

We describe the central role of Parallel computing in Clouds and Big Data which is decomposed into lots of "Little data" running in individual cores. Many examples are given and it is stressed that issues in parallel computing are seen in day to day life for communication, synchronization, load balancing and decomposition.

Parallel Computing (33)

7.1.1 Decomposition

We describe why parallel computing is essential with Big Data and distinguishes parallelism over users to that over the data in problem. The general ideas behind data decomposition are given followed by a few often whimsical examples dreamed up 30 years ago in the early heady days of parallel computing. These include scientific simulations, defense outside missile attack and computer chess. The basic problem of parallel computing – efficient coordination of separate tasks processing different data parts – is described with MPI and MapReduce as two approaches. The challenges of data decomposition in irregular problems is noted.

- Decomposition (8:51)
- Examples of Application (13:22)
- Decomposition Strategies (9:22)

7.1.2 Parallel Computing in Society

This lesson from the past notes that one can view society as an approach to parallel linkage of people. The largest example given is that of the construction of a long wall such as that (Hadrian's wall) between England and Scotland. Different approaches to parallelism are given with formulae for the speed up and efficiency. The concepts of grain size (size of problem tackled by an individual processor) and coordination overhead are exemplified. This example also illustrates Amdahl's law and the relation between data and processor topology. The lesson concludes with other examples from nature including collections of neurons (the brain) and ants.

- Parallel Computing in Society I (8:24)
- Parallel Computing in Society II (8:01)

7.1.3 Parallel Processing for Hadrian's Wall

This lesson returns to Hadrian's wall and uses it to illustrate advanced issues in parallel computing. First We describe the basic SPMD – Single Program Multiple Data – model. Then irregular but homogeneous and heterogeneous problems are discussed. Static and dynamic load balancing is needed. Inner parallelism (as in vector instruction or the multiple fingers of masons) and outer parallelism (typical data parallelism) are demonstrated. Parallel I/O for Hadrian's wall is followed by a slide summarizing this quaint comparison between Big data parallelism and the construction of a large wall.

- Processing for Hadrian's Wall (9:24)

7.1.4 Resources

- Solving Problems in Concurrent Processors-Volume 1, with M. Johnson, G. Lyzenga, S. Otto, J. Salmon, D. Walker, Prentice Hall, March 1988.
- Parallel Computing Works!, with P. Messina, R. Williams, Morgan Kaufman (1994).
- The Sourcebook of Parallel Computing book edited by Jack Dongarra, Ian Foster, Geoffrey Fox, William Gropp, Ken Kennedy, Linda Torczon, and Andy White, Morgan Kaufmann, November 2002.
- Geoffrey Fox Computational Sciences and Parallelism to appear in Encyclopedia on Parallel Computing edited by David Padua and published by Springer.

7.2 Introduction

We discuss Cyberinfrastructure for e-moreorlessanything or moreorlessanything-Informatics and the basics of cloud computing. This includes virtualization and the important 'as a Service' components and we go through several different definitions of cloud computing.Gartner's Technology Landscape includes hype cycle and priority matrix and covers clouds and Big Data. The unit concludes with two simple examples of the value of clouds for enterprise applications. Gartner also has specific predictions for cloud computing growth areas.

Introduction (45)

7.2.1 Cyberinfrastructure for E-Applications

This introduction describes Cyberinfrastructure or e-infrastructure and its role in solving the electronic implementation of any problem where e-moreorlessanything is another term for moreorlessanything-Informatics and generalizes early discussion of e-Science and e-Business.

- Cloud Computing Introduction Part1 (13:34)

7.2.2 What is Cloud Computing: Introduction

Cloud Computing is introduced with an operational definition involving virtualization and efficient large data centers that can rent computers in an elastic fashion. The role of services is essential – it underlies capabilities being offered in the cloud. The four basic aaS's – Software (SaaS), Platform (Paas), Infrastructure (IaaS) and Network (NaaS) – are introduced with Research aaS and other capabilities (for example Sensors aaS are discussed later) being built on top of these.

- What is Cloud Computing Intro (12:01)

7.2.3 What and Why is Cloud Computing: Other Views I

This lesson contains 5 slides with diverse comments on "what is cloud computing" from the web.

- Other Views I (5:25)
- Other Views II (6:41)
- Other Views III (7:27)

7.2.4 Gartner's Emerging Technology Landscape for Clouds and Big Data

This lesson gives Gartner's projections around futures of cloud and Big data. We start with a review of hype charts and then go into detailed Gartner analyses of the Cloud and Big data areas. Big data itself is at the top of the hype and by definition predictions of doom are emerging. Before too much excitement sets in, note that spinach is above clouds and Big data in Google trends.

- Gartners Emerging Technology Landscape (11:26)

7.2.5 Simple Examples of use of Cloud Computing

This short lesson gives two examples of rather straightforward commercial applications of cloud computing. One is server consolidation for multiple Microsoft database applications and the second is the benefits of scale comparing gmail to multiple smaller installations. It ends with some fiscal comments.

- Examples (3:26)

7.2.6 Value of Cloud Computing

Some comments on fiscal value of cloud computing.

- Value of Cloud Computing (4:20)

7.2.7 Resources

- http://www.slideshare.net/woorung/trend-and-future-of-cloud-computing
- http://www.slideshare.net/JensNimis/cloud-computing-tutorial-jens-nimis
- https://setandbma.wordpress.com/2012/08/10/hype-cycle-2012-emerging-technologies/
- http://insights.dice.com/2013/01/23/big-data-hype-is-imploding-gartner-analyst-2/
- http://research.microsoft.com/pubs/78813/AJ18_EN.pdf
- http://static.googleusercontent.com/media/www.google.com/en//green/pdfs/google-green-computing.pdf

7.3 Software and Systems

We cover different views as to nature of architecture and application for Cloud Computing. Then we discuss cloud software for the cloud starting at virtual machine management (IaaS) and the broad Platform (middleware) capabilities with examples from Amazon and academic studies. We summarize the 21 layers and almost 300 software packages in the HPC-ABDS Software Stack explaining how they are used.

Software and Systems (32)

7.3.1 What is Cloud Computing

This lesson gives some general remark of cloud systems from an architecture and application perspective.

- What is Cloud Computing (6:20)

7.3.2 Introduction to Cloud Software Architecture: IaaS and PaaS I

We discuss cloud software for the cloud starting at virtual machine management (IaaS) and the broad Platform (middleware) capabilities with examples from Amazon and academic studies. We cover different views as to nature of architecture and application for Cloud Computing. Then we discuss cloud software for the cloud starting at virtual machine management (IaaS) and the broad Platform (middleware) capabilities with examples from Amazon and academic studies. We summarize the 21 layers and almost 300 software packages in the HPC-ABDS Software Stack explaining how they are used.

- Intro to IaaS and PaaS I (7:42)
- Intro to IaaS and PaaS II (6:42)

We discuss cloud software for the cloud starting at virtual machine management (IaaS) and the broad Platform (middleware) capabilities with examples from Amazon and academic studies. We cover different views as to nature of architecture and application for Cloud Computing. Then we discuss cloud software for the cloud starting at virtual machine management (IaaS) and the broad Platform (middleware) capabilities with examples from Amazon and academic studies. We summarize the 21 layers and almost 300 software packages in the HPC-ABDS Software Stack explaining how they are used.

- Software Architecture: (7:42)
- IaaS and PaaS II: (6:43)

7.3.3 Using the HPC-ABDS Software Stack

Using the HPC-ABDS Software Stack.

- ABDS (27:50)

7.3.4 Resources

- http://www.slideshare.net/JensNimis/cloud-computing-tutorial-jens-nimis
- http://research.microsoft.com/en-us/people/barga/sc09_cloudcomp_tutorial.pdf
- http://research.microsoft.com/en-us/um/redmond/events/cloudfutures2012/tuesday/Keynote_OpportunitiesAndChallenges_Yousef_Khalidi.pdf
- http://cloudonomic.blogspot.com/2009/02/cloud-taxonomy-and-ontology.html

7.4 Architectures, Applications and Systems

We start with a discussion of Cloud (Data Center) Architectures with physical setup, Green Computing issues and software models. We summarize a 2014 Gartner analysis of Cloud computing providers. This is followed by applications on the cloud including data intensive problems, comparison with high performance computing, science clouds and the Internet of Things. Remarks on Security, Fault Tolerance and Synchronicity issues in cloud follow.

scroll: Architectures (64)

7.4.1 Cloud (Data Center) Architectures

Some remarks on what it takes to build (in software) a cloud ecosystem, and why clouds are the data center of the future are followed by pictures and discussions of several data centers from Microsoft (mainly) and Google. The role of containers is stressed as part of modular data centers that trade scalability for fault tolerance. Sizes of cloud centers and supercomputers are discussed as is "green" computing.

- Coud Architecture (8:38)
- Cloud Data Center Architecture (9:59)

7.4.2 Analysis of Major Cloud Providers

Gartner 2014 Analysis of leading cloud providers.

- Analysis of Major Cloud Providers (21:40)

7.4.3 Commercial Cloud Storage Trends

Use of Dropbox, iCloud, Box etc.

- Commercial Storage Trends (3:07)

7.4.4 Cloud Applications I

This short lesson discusses the need for security and issues in its implementation. Clouds trade scalability for greater possibility of faults but here clouds offer good support for recovery from faults. We discuss both storage and program fault tolerance noting that parallel computing is especially sensitive to faults as a fault in one task will impact all other tasks in the parallel job.

- Cloud Applications I (7:57)
- Cloud Applications II (7:44)

7.4.5 Science Clouds

Science Applications and Internet of Things.

- Science Clouds (19:26)

7.4.6 Security

This short lesson discusses the need for security and issues in its implementation.

- Security (2:34)

7.4.7 Comments on Fault Tolerance and Synchronicity Constraints

Clouds trade scalability for greater possibility of faults but here clouds offer good support for recovery from faults. We discuss both storage and program fault tolerance noting that parallel computing is especially sensitive to faults as a fault in one task will impact all other tasks in the parallel job.

- Comments on Fault Tolerance and Synchronicity Constraints (8:55)

7.4.8 Resources

- http://www.slideshare.net/woorung/trend-and-future-of-cloud-computing
- http://www.eweek.com/c/a/Cloud-Computing/AWS-Innovation-Means-Cloud-Domination-307831
- CSTI General Assembly 2012, Washington, D.C., USA Technical Activities Coordinating Committee (TACC) Meeting, Data Management, Cloud Computing and the Long Tail of Science October 2012 Dennis Gannon.
- http://research.microsoft.com/en-us/um/redmond/events/cloudfutures2012/tuesday/Keynote_OpportunitiesAndChallenges_Yousef_Khalidi.pdf
- http://www.datacenterknowledge.com/archives/2011/05/10/uptime-institute-the-average-pue-is-1-8/
- https://loosebolts.wordpress.com/2008/12/02/our-vision-for-generation-4-modular-data-centers-one-way-of-getting-it-just-right/
- http://www.mediafire.com/file/zzqna34282frr2f/koomeydatacenterelectuse2011finalversion.pdf
- http://www.slideshare.net/JensNimis/cloud-computing-tutorial-jens-nimis
- http://www.slideshare.net/botchagalupe/introduction-to-clouds-cloud-camp-columbus

- http://www.venus-c.eu/Pages/Home.aspx
- Geoffrey Fox and Dennis Gannon Using Clouds for Technical Computing To be published in Proceedings of HPC 2012 Conference at Cetraro, Italy June 28 2012
- https://berkeleydatascience.files.wordpress.com/2012/01/20120119berkeley.pdf
- Taming The Big Data Tidal Wave: Finding Opportunities in Huge Data Streams with Advanced Analytics, Bill Franks Wiley ISBN: 978-1-118-20878-6
- Anjul Bhambhri, VP of Big Data, IBM
- Conquering Big Data with the Oracle Information Model, Helen Sun, Oracle
- Hugh Williams VP Experience, Search & Platforms, eBay
- Dennis Gannon, Scientific Computing Environments
- http://research.microsoft.com/en-us/um/redmond/events/cloudfutures2012/tuesday/Keynote_OpportunitiesAndChallenges_Yousef_Khalidi.pdf
- http://www.datacenterknowledge.com/archives/2011/05/10/uptime-institute-the-average-pue-is-1-8/
- https://loosebolts.wordpress.com/2008/12/02/our-vision-for-generation-4-modular-data-centers-one-way-of-getting-it-just-right/
- http://www.mediafire.com/file/zzqna34282frr2f/koomeydatacenterelectuse2011finalversion.pdf
- http://searchcloudcomputing.techtarget.com/feature/Cloud-computing-experts-forecast-the-market-climate-in-2014
- http://www.slideshare.net/botchagalupe/introduction-to-clouds-cloud-camp-columbus
- http://www.slideshare.net/woorung/trend-and-future-of-cloud-computing
- http://www.venus-c.eu/Pages/Home.aspx
- http://www.kpcb.com/internet-trends

7.5 Data Systems

We describe the way users and data interact with a cloud system. The unit concludes with the treatment of data in the cloud from an architecture perspective and Big Data Processing from an application perspective with commercial examples including eBay.

Data Systems (49)

7.5.1 The 10 Interaction scenarios (access patterns) I

The next 3 lessons describe the way users and data interact with the system.

- The 10 Interaction scenarios I (10:26)

7.5.2 The 10 Interaction scenarios. Science Examples

This lesson describes the way users and data interact with the system for some science examples.

- The 10 Interaction scenarios. Science Examples (16:34)

7.5.3 Remaining general access patterns

This lesson describe the way users and data interact with the system for the final set of examples.

Access Patterns (11:36)

7.5.4 Data in the Cloud

Databases, File systems, Object Stores and NOSQL are discussed and compared. The way to build a modern data repository in the cloud is introduced.

Data in the Cloud (10:24)

7.5.5 Applications Processing Big Data

This lesson collects remarks on Big data processing from several sources: Berkeley, Teradata, IBM, Oracle and eBay with architectures and application opportunities.

Processing Big Data (8:45)

7.6 Resources

- http://bigdatawg.nist.gov/_uploadfiles/M0311_v2_2965963213.pdf
- https://dzone.com/articles/hadoop-t-etl
- http://venublog.com/2013/07/16/hadoop-summit-2013-hive-authorization/
- https://indico.cern.ch/event/214784/session/5/contribution/410
- http://asd.gsfc.nasa.gov/archive/hubble/a_pdf/news/facts/FS14.pdf
- http://blogs.teradata.com/data-points/announcing-teradata-aster-big-analytics-appliance/
- http://wikibon.org/w/images/2/20/Cloud-BigData.png
- http://hortonworks.com/hadoop/yarn/
- https://berkeleydatascience.files.wordpress.com/2012/01/20120119berkeley.pdf
- http://fisheritcenter.haas.berkeley.edu/Big_Data/index.html

8 Big Data Use Cases Survey

This section covers 51 values of X and an overall study of Big data that emerged from a NIST (National Institute for Standards and Technology) study of Big data. The section covers the NIST Big Data Public Working Group (NBD-PWG) Process and summarizes the work of five subgroups: Definitions and Taxonomies Subgroup, Reference Architecture Subgroup, Security and Privacy Subgroup, Technology Roadmap Subgroup and the Requirements andUse Case Subgroup. 51 use cases collected in this process are briefly discussed with a classification of the source of parallelism and the high and low level computational structure. We describe the key features of this classification.

8.1 NIST Big Data Public Working Group

This unit covers the NIST Big Data Public Working Group (NBD-PWG) Process and summarizes the work of five subgroups: Definitions and Taxonomies Subgroup, Reference Architecture Subgroup, Security and Privacy Subgroup, Technology Roadmap Subgroup and the Requirements and Use Case Subgroup. The work of latter is continued in next two units.

Overview (45)

8.1.1 Introduction to NIST Big Data Public Working

The focus of the (NBD-PWG) is to form a community of interest from industry, academia, and government, with the goal of developing a consensus definitions, taxonomies, secure reference architectures, and technology roadmap. The aim is to create vendor-neutral, technology and infrastructure agnostic deliverables to enable big data stakeholders to pick-and-choose best analytics tools for their processing and visualization requirements on the most suitable computing platforms and clusters while allowing value-added from big data service providers and flow of data between the stakeholders in a cohesive and secure manner.

Introduction (13:02)

8.1.2 Definitions and Taxonomies Subgroup

The focus is to gain a better understanding of the principles of Big Data. It is important to develop a consensus-based common language and vocabulary terms used in Big Data across stakeholders from industry, academia, and government. In addition, it is also critical to identify essential actors with roles and responsibility, and subdivide them into components and sub-components on how they interact/ relate with each other according to their similarities and differences.

For Definitions: Compile terms used from all stakeholders regarding the meaning of Big Data from various standard bodies, domain applications, and diversified operational environments. For Taxonomies: Identify key actors with their roles and responsibilities from all stakeholders, categorize them into components and subcomponents based on their similarities and differences. In particular data Science and Big Data terms are discussed.

Taxonomies (7:42)

8.1.3 Reference Architecture Subgroup

The focus is to form a community of interest from industry, academia, and government, with the goal of developing a consensus-based approach to orchestrate vendor-neutral, technology and infrastructure agnostic for analytics tools and computing environments. The goal is to enable Big Data stakeholders to pick-and-choose technology-agnostic analytics tools for processing and visualization in any computing platform and cluster while allowing value-added from Big Data service providers and the flow of the data between the stakeholders in a cohesive and secure manner. Results include a reference architecture with well defined components and linkage as well as several exemplars.

Architecture (10:05)

8.1.4 Security and Privacy Subgroup

The focus is to form a community of interest from industry, academia, and government, with the goal of developing a consensus secure reference architecture to handle security and privacy issues across all stakeholders. This includes gaining an understanding of what standards are available or under development, as well as identifies which key organizations are working on these standards. The Top Ten Big Data Security and Privacy Challenges from the CSA (Cloud Security Alliance) BDWG are studied. Specialized use cases include Retail/Marketing, Modern Day Consumerism, Nielsen Homescan, Web Traffic Analysis, Healthcare, Health Information Exchange, Genetic Privacy, Pharma Clinical Trial Data Sharing, Cyber-security, Government, Military and Education.

Security (9:51)

8.1.5 Technology Roadmap Subgroup

The focus is to form a community of interest from industry, academia, and government, with the goal of developing a consensus vision with recommendations on how Big Data should move forward by performing a good gap analysis through the materials gathered from all other NBD subgroups. This includes setting standardization and adoption priorities through an understanding of what standards are available or under development as part of the recommendations. Tasks are gather input from NBD subgroups and study the taxonomies for the actors' roles and responsibility, use cases and requirements, and secure reference architecture; gain understanding of what standards are available or under development for Big Data; perform a thorough gap analysis and document the findings; identify what possible barriers may delay or prevent adoption of Big Data; and document vision and recommendations.

Technology (4:14)

8.1.6 Interfaces Subgroup

This subgroup is working on the following document: NIST Big Data Interoperability Framework: Volume 8, Reference Architecture Interface.

This document summarizes interfaces that are instrumental for the interaction with Clouds, Containers, and HPC systems to manage virtual clusters to support the NIST Big Data Reference Architecture (NBDRA). The Representational State Transfer (REST) paradigm is used to define these interfaces allowing easy integration and adoption by a wide variety of frameworks. . This volume, Volume 8, uses the work performed by the NBD-PWG to identify objects instrumental for the NIST Big Data Reference Architecture (NBDRA) which is introduced in the NBDIF: Volume 6, Reference Architecture.

This presentation was given at the 2nd NIST Big Data Public Working Group (NBD-PWG) Workshop in Washington DC in June 2017. It explains our thoughts on deriving automatically a reference architecture form the Reference Architecture Interface specifications directly from the document.

The workshop Web page is located at

- https://bigdatawg.nist.gov/workshop2.php

The agenda of the workshop is as follows:

- https://bigdatawg.nist.gov/2017_NIST_Big_Data_PWG_WorkshopAgenda_with_Speakers_Bio.pdf

The Web cas of the presentation is given bellow, while you need to fast forward to a particular time

- Webcast: Interface subgroup: https://www.nist.gov/news-events/events/2017/06/2nd-nist-big-data-public-working-group-nbd-pwg-workshop
  - see: Big Data Working Group Day 1, part 2 Time start: 21:00 min, Time end: 44:00
- Slides: https://github.com/cloudmesh/cloudmesh.rest/blob/master/docs/NBDPWG-vol8.pptx?raw=true
- Document: https://github.com/cloudmesh/cloudmesh.rest/raw/master/docs/NIST.SP.1500-8-draft.pdf

You are welcome to view other presentations if you are interested.

8.1.7 Requirements and Use Case Subgroup

The focus is to form a community of interest from industry, academia, and government, with the goal of developing a consensus list of Big Data requirements across all stakeholders. This includes gathering and understanding various use cases from diversified application domains.Tasks are gather use case input from all stakeholders; derive Big Data requirements from each use case; analyze/prioritize a list of challenging general requirements that may delay or prevent adoption of Big Data deployment; develop a set of general patterns capturing the essence of use cases (not done yet) and work with Reference Architecture to validate requirements and reference architecture by explicitly implementing some patterns based on use cases. The progress of gathering use cases (discussed in next two units) and requirements systemization are discussed.

Requirements (27:28)

8.2 51 Big Data Use Cases

This units consists of one or more slides for each of the 51 use cases - typically additional (more than one) slides are associated with pictures. Each of the use cases is identified with source of parallelism and the high and low level computational structure. As each new classification topic is introduced we briefly discuss it but full discussion of topics is given in following unit.

51 Use Cases (100)

8.2.1 Government Use Cases

This covers Census 2010 and 2000 - Title 13 Big Data; National Archives and Records Administration Accession NARA, Search, Retrieve, Preservation; Statistical Survey Response Improvement (Adaptive Design) and Non-Traditional Data in Statistical Survey Response Improvement (Adaptive Design).

Government Use Cases (17:43)

8.2.2 Commercial Use Cases

This covers Cloud Eco-System, for Financial Industries (Banking, Securities & Investments, Insurance) transacting business within the United States; Mendeley - An International Network of Research; Netflix Movie Service; Web Search; IaaS (Infrastructure as a Service) Big Data Business Continuity & Disaster Recovery (BC/DR) Within A Cloud Eco-System; Cargo Shipping; Materials Data for Manufacturing and Simulation driven Materials Genomics.

Commercial Use Cases (17:43)

8.2.3 Defense Use Cases

This covers Large Scale Geospatial Analysis and Visualization; Object identification and tracking from Wide Area Large Format Imagery (WALF) Imagery or Full Motion Video (FMV) - Persistent Surveillance and Intelligence Data Processing and Analysis.

Defense Use Cases (15:43)

8.2.4 Healthcare and Life Science Use Cases

This covers Electronic Medical Record (EMR) Data; Pathology Imaging/digital pathology; Computational Bioimaging; Genomic Measurements; Comparative analysis for metagenomes and genomes; Individualized Diabetes Management; Statistical Relational Artificial Intelligence for Health Care; World Population Scale Epidemiological Study; Social Contagion Modeling for Planning, Public Health and Disaster Management and Biodiversity and LifeWatch.

Healthcare and Life Science Use Cases (30:11)

8.2.5 Deep Learning and Social Networks Use Cases

This covers Large-scale Deep Learning; Organizing large-scale, unstructured collections of consumer photos; Truthy: Information diffusion research from Twitter Data; Crowd Sourcing in the Humanities as Source for Bigand Dynamic Data; CINET: Cyberinfrastructure for Network (Graph) Science and Analytics and NIST Information Access Division analytic technology performance measurement, evaluations, and standards.

Deep Learning and Social Networks Use Cases (14:19)

8.2.6 Research Ecosystem Use Cases

DataNet Federation Consortium DFC; The 'Discinnet process', metadata -big data global experiment; Semantic Graph-search on Scientific Chemical and Text-based Data and Light source beamlines.

Research Ecosystem Use Cases (9:09)

8.2.7 Astronomy and Physics Use Cases

This covers Catalina Real-Time Transient Survey (CRTS): a digital, panoramic, synoptic sky survey; DOE Extreme Data from Cosmological Sky Survey and Simulations; Large Survey Data for Cosmology; Particle Physics: Analysis of LHC Large Hadron Collider Data: Discovery of Higgs particle and Belle II High Energy Physics Experiment.

Astronomy and Physics Use Cases (17:33)

8.2.8 Environment, Earth and Polar Science Use Cases

EISCAT 3D incoherent scatter radar system; ENVRI, Common Operations of Environmental Research Infrastructure; Radar Data Analysis for CReSIS Remote Sensing of Ice Sheets; UAVSAR Data Processing, DataProduct Delivery, and Data Services; NASA LARC/GSFC iRODS Federation Testbed; MERRA Analytic Services MERRA/AS; Atmospheric Turbulence - Event Discovery and Predictive Analytics; Climate Studies using the Community Earth System Model at DOE's NERSC center; DOE-BER Subsurface Biogeochemistry Scientific Focus Area and DOE-BER AmeriFlux and FLUXNET Networks.

Environment, Earth and Polar Science Use Cases (25:29)

8.2.9 Energy Use Case

This covers Consumption forecasting in Smart Grids.

Energy Use Case (4:01)

8.3 Features of 51 Big Data Use Cases

This unit discusses the categories used to classify the 51 use-cases. These categories include concepts used for parallelism and low and high level computational structure. The first lesson is an introduction to all categories and the further lessons give details of particular categories.

Features (43)

8.3.1 Summary of Use Case Classification

This discusses concepts used for parallelism and low and high level computational structure. Parallelism can be over People (users or subjects), Decision makers; Items such as Images, EMR, Sequences; observations, contents of online store; Sensors – Internet of Things; Events; (Complex) Nodes in a Graph; Simple nodes as in a learning network; Tweets, Blogs, Documents, Web Pages etc.; Files or data to be backed up, moved or assigned metadata; Particles/cells/mesh points. Low level computational types include PP (Pleasingly Parallel); MR (MapReduce); MRStat; MRIter (Iterative MapReduce); Graph; Fusion; MC (Monte Carlo) and Streaming. High level computational types include Classification; S/Q (Search and Query); Index; CF (Collaborative Filtering); ML (Machine Learning); EGO (Large Scale Optimizations); EM (Expectation maximization); GIS; HPC; Agents. Patterns include Classic Database; NoSQL; Basic processing of data as in backup or metadata; GIS; Host of Sensors processed on demand; Pleasingly parallel processing; HPC assimilated with observational data; Agent-based models; Multi-modal data fusion or Knowledge Management; Crowd Sourcing.

Summary of Use Case Classification (23:39)

8.3.2 Database(SQL) Use Case Classification

This discusses classic (SQL) database approach to data handling with Search&Query and Index features. Comparisons are made to NoSQL approaches.

Database (SQL) Use Case Classification (11:13)

8.3.3 NoSQL Use Case Classification

This discusses NoSQL (compared in previous lesson) with HDFS, Hadoop and Hbase. The Apache Big data stack is introduced and further details of comparison with SQL.

NoSQL Use Case Classification (11:20)

8.3.4 Other Use Case Classifications

This discusses a subset of use case features: GIS, Sensors. the support of data analysis and fusion by streaming data between filters.

Use Case Classifications I (12:42) This discusses a subset of use case features: Pleasingly parallel, MRStat, Data Assimilation, Crowd sourcing, Agents, data fusion and agents, EGO and security.

Use Case Classifications II (20:18)

This discusses a subset of use case features: Classification, Monte Carlo, Streaming, PP, MR, MRStat, MRIter and HPC(MPI), global and local analytics (machine learning), parallel computing, Expectation Maximization, graphs and Collaborative Filtering.

Use Case Classifications III (17:25)

8.3.5 Resources

- NIST Big Data Public Working Group (NBD-PWG) Process
- Big Data Definitions
- Big Data Taxonomies
- Big Data Use Cases and Requirements
- Big Data Security and Privacy
- Big Data Architecture White Paper Survey
- Big Data Reference Architecture
- Big Data Standards Roadmap

Some of the links bellow may be outdated. Please let us know the new links and notify us of the outdated links.

- DCGSA Standard Cloud
- On line 51 Use Cases

- Summary of Requirements Subgroup
- ~~Use Case 6 Mendeley~~ **(this link does not exist any longer)**
- ~~Use Case 7 Netflix~~
- Use Case 8 Search
  - http://www.slideshare.net/kleinerperkins/kpcb-internet-trends-2013,
  - http://webcourse.cs.technion.ac.il/236621/Winter2011-2012/en/ho_Lectures.html,
  - http://www.ifis.cs.tu-bs.de/teaching/ss-11/irws,
  - http://www.slideshare.net/beechung/recommender-systems-tutorialpart1intro,
  - http://www.worldwidewebsize.com/
- Use Case 9 IaaS (Infrastructure as a Service) Big Data Business Continuity & Disaster Recovery (BC/DR) Within A Cloud Eco-System provided by Cloud Service Providers (CSPs) and Cloud Brokerage Service Providers (CBSPs)
- Use Case 11 and Use Case 12 Simulation driven Materials Genomics
- Use Case 13 Large Scale Geospatial Analysis and Visualization
  - http://www.opengeospatial.org/standards
  - http://geojson.org/
  - http://earth-info.nga.mil/publications/specs/printed/CADRG/cadrg.html
- Use Case 14 Object identification and tracking from Wide Area Large Format Imagery (WALF) Imagery or Full Motion Video (FMV) - Persistent Surveillance
  - http://www.militaryaerospace.com/topics/m/video/79088650/persistent-surveillance-relies-on-extracting-relevant-data-points-and-connecting-the-dots.htm,
  - http://www.defencetalk.com/wide-area-persistent-surveillance-revolutionizes-tactical-isr-45745/
- Use Case 15 Intelligence Data Processing and Analysis
  - http://www.afcea-aberdeen.org/files/presentations/AFCEAAberdeen_DCGSA_COLWells_PS.pdf,
  - http://stids.c4i.gmu.edu/STIDS2011/papers/STIDS2011_CR_T1_SalmenEtAl.pdf,
  - ~~http://stids.c4i.gmu.edu/papers/STIDSPapers/STIDS2012/_T14/_SmithEtAl/_HorizontalIntegrationOfWarfighterIntel.pdf~~,
  - ~~https://www.youtube.com/watch?v=l4Qii7TBzeg~~
  - ~~http://dcgsa.apg.army.mil/~~
- Use Case 16 Electronic Medical Record (EMR) Data:
  - Regenstrief Institute
  - Logical observation identifiers names and codes
  - Indiana Health Information Exchange
  - ~~Institute of Medicine Learning Healthcare System~~
- Use Case 17
  - Pathology Imaging/digital pathology
  - https://web.cci.emory.edu/confluence/display/HadoopGIS
- Use Case 19 Genome in a Bottle Consortium:
  - ~~www.genomeinabottle.org~~
- Use Case 20 Comparative analysis for metagenomes and genomes
- Use Case 25
  - Biodiversity
  - LifeWatch
- Use Case 26 Deep Learning: Recent popular press coverage of deep learning technology:
  - http://www.nytimes.com/2012/11/24/science/scientists-see-advances-in-deep-learning-a-part-of-artificial-intelligence.html
  - http://www.nytimes.com/2012/06/26/technology/in-a-big-network-of-computers-evidence-of-machine-learning.html
  - http://www.wired.com/2013/06/andrew_ng/,
  - ~~A recent research paper on HPC for Deep Learning~~
  - Widely-used tutorials and references for Deep Learning:
    - http://ufldl.stanford.edu/wiki/index.php/Main_Page
    - http://deeplearning.net/
- Use Case 27 Organizing large-scale, unstructured collections of consumer photos
- Use Case 28
  - Truthy: Information diffusion research from Twitter Data
  - http://cnets.indiana.edu/groups/nan/truthy/
  - http://cnets.indiana.edu/groups/nan/despic/
- ~~Use Case 30 CINET: Cyberinfrastructure for Network (Graph) Science and Analytics~~
- Use Case 31 NIST Information Access Division analytic technology performance measurement, evaluations, and standards
- Use Case 32
  - DataNet Federation Consortium DFC: The DataNet Federation Consortium,
  - iRODS
- Use Case 33 The 'Discinnet process', big data global experiment
- Use Case 34 Semantic Graph-search on Scientific Chemical and Text-based Data
  - http://www.eurekalert.org/pub_releases/2013-07/aiop-ffm071813.php
  - http://xpdb.nist.gov/chemblast/pdb.pl
- Use Case 35 Light source beamlines
  - http://www-als.lbl.gov/
  - https://www1.aps.anl.gov/
- Use Case 36
  - CRTS survey
  - CSS survey
  - For an overview of the classification challenges
- Use Case 37 DOE Extreme Data from Cosmological Sky Survey and Simulations
  - http://www.lsst.org/lsst/
  - http://www.nersc.gov/
  - http://www.nersc.gov/assets/Uploads/HabibcosmosimV2.pdf
- Use Case 38 Large Survey Data for Cosmology
  - http://desi.lbl.gov/
  - http://www.darkenergysurvey.org/
- Use Case 39 Particle Physics: Analysis of LHC Large Hadron Collider Data: Discovery of Higgs particle
  - http://grids.ucs.indiana.edu/ptliupages/publications/Where%20does%20all%20the%20data%20come%20from%20v7.pdf,
  - http://www.es.net/assets/pubs_presos/High-throughput-lessons-from-the-LHC-experience.Johnston.TNC2013.pdf
- ~~Use Case 40 Belle II High Energy Physics Experiment~~ **(old link does not exist, new link: https://www.belle2.org)**
- Use Case 41 EISCAT 3D incoherent scatter radar system
- Use Case 42 ENVRI, Common Operations of Environmental Research Infrastructure
  - ENVRI Project website
  - ~~ENVRI Reference Model~~
  - ~~ENVRI deliverable D3.2 : Analysis of common requirements of Environmental Research Infrastructures~~
  - ICOS,
  - Euro-Argo
  - EISCAT 3D
  - LifeWatch
  - EPOS
  - EMSO
- Use Case 43 Radar Data Analysis for CReSIS Remote Sensing of Ice Sheets
- Use Case 44 UAVSAR Data Processing, Data Product Delivery, and Data Services
  - http://uavsar.jpl.nasa.gov/,
  - http://www.asf.alaska.edu/program/sdc,
  - http://geo-gateway.org/main.html
- Use Case 47 Atmospheric Turbulence - Event Discovery and Predictive Analytics
  - http://oceanworld.tamu.edu/resources/oceanography-book/teleconnections.htm
  - http://www.forbes.com/sites/toddwoody/2012/03/21/meet-the-scientists-mining-big-data-to-predict-the-weather/
- Use Case 48 Climate Studies using the Community Earth System Model at DOE's NERSC center
  - http://www-pcmdi.llnl.gov/
  - http://www.nersc.gov/
  - http://science.energy.gov/ber/research/cesd/
  - http://www2.cisl.ucar.edu/
- Use Case 50 DOE-BER AmeriFlux and FLUXNET Networks
  - http://ameriflux.lbl.gov/
  - http://www.fluxdata.org/default.aspx
- Use Case 51 Consumption forecasting in Smart Grids
  - ~~http://smartgrid.usc.edu/~~ **(old link does not exsit, new link: http://dslab.usc.edu/smartgrid.php)**
  - http://ganges.usc.edu/wiki/Smart_Grid

- https://www.ladwp.com/ladwp/faces/ladwp/aboutus/a-power/a-p-smartgridla?_afrLoop=157401916661989&_afrWindowMode=0&_afrWindowId=null#%40%3F_afrWindowId%3Dnull%26_afrLoop%3D157401916661989%26_afrWindowMode%3D0%26_adf.ctrl-state%3Db7yulr4rl_17
- http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6475927

9 Sensors

We start with the Internet of Things IoT giving examples like monitors of machine operation, QR codes, surveillance cameras, scientific sensors, drones and self driving cars and more generally transportation systems. We give examples of robots and drones. We introduce the Industrial Internet of Things IIoT and summarize surveys and expectations Industry wide. We give examples from General Electric. Sensor clouds control the many small distributed devices of IoT and IIoT. More detail is given for radar data gathered by sensors; ubiquitous or smart cities and homes including U-Korea; and finally the smart electric grid.

Sensor I (31)

Sensor II (44)

9.1 Internet of Things

There are predicted to be 24-50 Billion devices on the Internet by 2020; these are typically some sort of sensor defined as any source or sink of time series data. Sensors include smartphones, webcams, monitors of machine operation, barcodes, surveillance cameras, scientific sensors (especially in earth and environmental science), drones and self driving cars and more generally transportation systems. The lesson gives many examples of distributed sensors, which form a Grid that is controlled by a cloud.

Internet of Things (12:36)

9.2 Robotics and IoT

Examples of Robots and Drones.

Robotics and IoT Expectations (8:05)

9.3 Industrial Internet of Things

We summarize surveys and expectations Industry wide.

Industrial Internet of Things (24:02)

9.4 Sensor Clouds

We describe the architecture of a Sensor Cloud control environment and gives example of interface to an older version of it. The performance of system is measured in terms of processing latency as a function of number of involved sensors with each delivering data at 1.8 Mbps rate.

Sensor Clouds (4:40)

9.5 Earth/Environment/Polar Science data gathered by Sensors

This lesson gives examples of some sensors in the Earth/Environment/Polar Science field. It starts with material from the CReSIS polar remote sensing project and then looks at the NSF Ocean Observing Initiative and NASA's MODIS or Moderate Resolution Imaging Spectroradiometer instrument on a satellite.

Earth/Environment/Polar Science data gathered by Sensors (4:58)

9.6 Ubiquitous/Smart Cities

For Ubiquitous/Smart cities we give two examples: Iniquitous Korea and smart electrical grids.

Ubiquitous/Smart Cities (1:44)

9.7 U-Korea (U=Ubiquitous)

Korea has an interesting positioning where it is first worldwide in broadband access per capita, e-government, scientific literacy and total working hours. However it is far down in measures like quality of life and GDP. U-Korea aims to improve the latter by Pervasive computing, everywhere, anytime i.e. by spreading sensors everywhere. The example of a 'High-Tech Utopia' New Songdo is given.

U-Korea (U=Ubiquitous) (2:49)

9.8 Smart Grid

The electrical Smart Grid aims to enhance USA's aging electrical infrastructure by pervasive deployment of sensors and the integration of their measurement in a cloud or equivalent server infrastructure. A variety of new instruments include smart meters, power monitors, and measures of solar irradiance, wind speed, and temperature. One goal is autonomous local power units where good use is made of waste heat.

Smart Grid (6:04)

9.9 Resources

- http://www.accenture.com/SiteCollectionDocuments/PDF/Accenture-Industrial-Internet-Changing-Competitive-Landscape-Industries.pdf [47]
- http://www.gesoftware.com/ge-predictivity-infographic [48]
- http://www.getransportation.com/railconnect360/rail-landscape [49]
- http://www.gesoftware.com/sites/default/files/GE-Software-Modernizing-Machine-to-Machine-Interactions.pdf **???**

resources below do not exsit:

- ~~https://www.gesoftware.com/minds-and-machines~~
- ~~https://www.gesoftware.com/predix~~
- ~~https://www.gesoftware.com/sites/default/files/the-industrial-internet/index.html~~
- ~~https://developer.cisco.com/site/eiot/discover/overview/~~

10 Radar

The changing global climate is suspected to have long-term effects on much of the world's inhabitants. Among the various effects, the rising sea level will directly affect many people living in low-lying coastal regions. While the ocean-s thermal expansion has been the dominant contributor to rises in sea level, the potential contribution of discharges from the polar ice sheets in Greenland and Antarctica may provide a more significant threat due to the unpredictable response to the changing climate. The Radar-Informatics unit provides a glimpse in the processes fueling global climate change and explains what methods are used for ice data acquisitions and analysis.

Radar (58)

10.1 Introduction

This lesson motivates radar-informatics by building on previous discussions on why X-applications are growing in data size and why analytics are necessary for acquiring knowledge from large data. The lesson details three mosaics of a changing Greenland ice sheet and provides a concise overview to subsequent lessons by detailing explaining how other remote sensing technologies, such as the radar, can be used to sound the polar ice sheets and what we are doing with radar images to extract knowledge to be incorporated into numerical models.

- Radar Informatics (3:31)

10.2 Remote Sensing

This lesson explains the basics of remote sensing, the characteristics of remote sensors and remote sensing applications. Emphasis is on image acquisition and data collection in the electromagnetic spectrum.

- Remote Sensing (6:43)

10.3 Ice Sheet Science

This lesson provides a brief understanding on why melt water at the base of the ice sheet can be detrimental and why it's important for sensors to sound the bedrock.

- Ice Sheet Science (1:00)

10.4 Global Climate Change

This lesson provides an understanding and the processes for the greenhouse effect, how warming effects the Polar Regions, and the implications of a rise in sea level.

- Global Climate Change (2:51)

10.5 Radio Overview

This lesson provides an elementary introduction to radar and its importance to remote sensing, especially to acquiring information about Greenland and Antarctica.

- Radio Overview (4:16)

10.6 Radio Informatics

This lesson focuses on the use of sophisticated computer vision algorithms, such as active contours and a hidden markov model to support data analysis for extracting layers, so ice sheet models can accurately forecast future changes in climate.

- Radio Informatics (3:35)

11 Web Search and Text Mining

This section starts with an overview of data mining and puts our study of classification, clustering and exploration methods in context. We examine the problem to be solved in web and text search and note the relevance of history with libraries, catalogs and concordances. An overview of web search is given describing the continued evolution of search engines and the relation to the field of Information.

The importance of recall, precision and diversity is discussed. The important Bag of Words model is introduced and both Boolean queries and the more general fuzzy indices. The important vector space model and revisiting the Cosine Similarity as a distance in this bag follows. The basic TF-IDF approach is dis cussed. Relevance is discussed with a probabilistic model while the distinction between Bayesian and frequency views of probability distribution completes this unit.

We start with an overview of the different steps (data analytics) in web search and then goes key steps in detail starting with document preparation. An inverted index is described and then how it is prepared for web search. The Boolean and Vector Space approach to query processing follow. This is followed by Link Structure Analysis including Hubs, Authorities and PageRank. The application of PageRank ideas as reputation outside web search is covered. The web graph structure, crawling it and issues in web advertising and search follow. The use of clustering and topic models completes the section.

11.1 Web Search and Text Mining

The unit starts with the web with its size, shape (coming from the mutual linkage of pages by URL's) and universal power laws for number of pages with particular number of URL's linking out or in to page. Information retrieval is introduced and compared to web search. A comparison is given between semantic searches as in databases and the full text search that is base of Web search. The origin of web search in libraries, catalogs and concordances is summarized. DIKW – Data Information Knowledge Wisdom – model for web search is discussed. Then features of documents, collections and the important Bag of Words representation. Queries are presented in context of an Information Retrieval architecture. The method of judging quality of results including recall, precision and diversity is described. A time line for evolution of search engines is given.

Boolean and Vector Space models for query including the cosine similarity are introduced. Web Crawlers are discussed and then the steps needed to analyze data from Web and produce a set of terms. Building and accessing an inverted index is followed by the importance of term specificity and how it is captured in TF-IDF. We note how frequencies are converted into belief and relevance.

Web Search and Text Mining (56)

11.1.1 The Problem

Text Mining (9:56)

This lesson starts with the web with its size, shape (coming from the mutual linkage of pages by URL's) and universal power laws for number of pages with particular number of URL's linking out or in to page.

11.1.2 Information Retrieval

Information Retrieval (6:06)

Information retrieval is introduced A comparison is given between semantic searches as in databases and the full text search that is base of Web search. The ACM classification illustrates potential complexity of ontologies. Some differences between web search and information retrieval are given.

11.1.3 History

Web Search History (5:48)

The origin of web search in libraries, catalogs and concordances is summarized.

11.1.4 Key Fundamental Principles

Principles (9:30)

This lesson describes the DIKW – Data Information Knowledge Wisdom – model for web search. Then it discusses documents, collections and the important Bag of Words representation.

11.1.5 Information Retrieval (Web Search) Components

Fundamental Principals of Web Search (5:06)

This describes queries in context of an Information Retrieval architecture. The method of judging quality of results including recall, precision and diversity is described.

11.2 Search Engines

Search Engines (3:08)

This short lesson describes a time line for evolution of search engines. The first web search approaches were directly built on Information retrieval but in 1998 the field was changed when Google was founded and showed the importance of URL structure as exemplified by PageRank.

11.2.1 Boolean and Vector Space Models

Boolean and Vector Space Model (6:17)

This lesson describes the Boolean and Vector Space models for query including the cosine similarity.

11.2.2 Web crawling and Document Preparation

Web crawling and Document Preparation (4:55)

This describes a Web Crawler and then the steps needed to analyze data from Web and produce a set of terms.

11.2.3 Indices

Indices (5:44)

This lesson describes both building and accessing an inverted index. It describes how phrases are treated and gives details of query structure from some early logs.

11.2.4 TF-IDF and Probabilistic Models

TF-IDF and Probabilistic Models (3:57)

It describes the importance of term specificity and how it is captured in TF-IDF. It notes how frequencies are converted into belief and relevance.

11.3 Topics in Web Search and Text Mining

Text Mining (33)

We start with an overview of the different steps (data analytics) in web search. This is followed by Link Structure Analysis including Hubs, Authorities and PageRank. The application of PageRank ideas as reputation outside web search is covered. Issues in web advertising and search follow. his leads to emerging field of computational advertising. The use of clustering and topic models completes unit with Google News as an example.

11.3.1 Data Analytics for Web Search

Web Search and Text Mining II (6:11)

This short lesson describes the different steps needed in web search including: Get the digital data (from web or from scanning); Crawl web; Preprocess data to get searchable things (words, positions); Form Inverted Index mapping words to documents; Rank relevance of documents with potentially sophisticated techniques; and integrate technology to support advertising and ways to allow or stop pages artificially enhancing relevance.

11.3.2 Link Structure Analysis including PageRank

Related Applications (17:24)

The value of links and the concepts of Hubs and Authorities are discussed. This leads to definition of PageRank with examples. Extensions of PageRank viewed as a reputation are discussed with journal rankings and university department rankings as examples. There are many extension of these ideas which are not discussed here although topic models are covered briefly in a later lesson.

11.3.3 Web Advertising and Search

Web Advertising and Search (9:02)

Internet and mobile advertising is growing fast and can be personalized more than for traditional media. There are several advertising types Sponsored search, Contextual ads, Display ads and different models: Cost per viewing, cost per clicking and cost per action. This leads to emerging field of computational advertising.

11.3.4 Clustering and Topic Models

Clustering and Topic Models (6:21)

We discuss briefly approaches to defining groups of documents. We illustrate this for Google News and give an example that this can give different answers from word-based analyses. We mention some work at Indiana University on a Latent Semantic Indexing model.

11.3.5 Resources

All resources accessed March 2018.

- http://saedsayad.com/data_mining_map.htm
- http://webcourse.cs.technion.ac.il/236621/Winter2011-2012/en/ho_Lectures.html
- The Web Graph: an Overviews
- Jean-Loup Guillaume and Matthieu Latapy
- Constructing a reliable Web graph with information on browsing behavior, Yiqun Liu, Yufei Xue, Danqing Xu, Rongwei Cen, Min Zhang, Shaoping Ma, Liyun Ru
- http://www.ifis.cs.tu-bs.de/teaching/ss-11/irws
- https://en.wikipedia.org/wiki/PageRank
- Meeker/Wu May 29 2013 Internet Trends D11 Conference

12 Health Informatics

Health Informatics (131)

This section starts by discussing general aspects of Big Data and Health including data sizes, different areas including genomics, EBI, radiology and the Quantified Self movement. We review current state of health care and trends associated with it including increased use of Telemedicine. We summarize an industry survey by GE and Accenture and an impressive exemplar Cloud-based medicine system from Potsdam. We give some details of big data in medicine. Some remarks on Cloud computing and Health focus on security and privacy issues.

We survey an April 2013 McKinsey report on the Big Data revolution in US health care; a Microsoft report in this area and a European Union report on how Big Data will allow patient centered care in the future. Examples are given of the Internet of Things, which will have great impact on health including wearables. A study looks at 4 scenarios for healthcare in 2032. Two are positive, one middle of the road and one negative. The final topic is Genomics, Proteomics and Information Visualization.

12.1 Big Data and Health

This lesson starts with general aspects of Big Data and Health including listing subareas where Big data important. Data sizes are given in radiology, genomics, personalized medicine, and the Quantified Self movement, with sizes and access to European Bioinformatics Institute.

Big Data and Health (10:02)

12.2 Status of Healthcare Today

This covers trends of costs and type of healthcare with low cost genomes and an aging population. Social media and government Brain initiative.

Status of Healthcare Today (16:09)

12.3 Telemedicine (Virtual Health)

This describes increasing use of telemedicine and how we tried and failed to do this in 1994.

Telemedicine (8:21)

12.4 Medical Big Data in the Clouds

An impressive exemplar Cloud-based medicine system from Potsdam.

Medical Big Data in the Clouds (15:02)

12.4.1 Medical image Big Data

Medical Image Big Data (6:33)

12.4.2 Clouds and Health

Clouds and Health (4:35)

12.4.3 McKinsey Report on the big-data revolution in US health care

This lesson covers 9 aspects of the McKinsey report. These are the convergence of multiple positive changes has created a tipping point for

innovation; Primary data pools are at the heart of the big data revolution in healthcare; Big data is changing the paradigm: these are the value pathways; Applying early successes at scale could reduce US healthcare costs by $300 billion to $450 billion; Most new big-data applications target consumers and providers across pathways; Innovations are weighted towards influencing individual decision-making levers; Big data innovations use a range of public, acquired, and proprietary data

types; Organizations implementing a big data transformation should provide the leadership required for the associated cultural transformation; Companies must develop a range of big data capabilities.

McKinsey Report (14:53)

12.4.4 Microsoft Report on Big Data in Health

This lesson identifies data sources as Clinical Data, Pharma & Life Science Data, Patient & Consumer Data, Claims & Cost Data and Correlational Data. Three approaches are Live data feed, Advanced analytics and Social analytics.

Microsoft Report on Big Data in Health (2:26)

12.4.5 EU Report on Redesigning health in Europe for 2020

This lesson summarizes an EU Report on Redesigning health in Europe for 2020. The power of data is seen as a lever for change in My Data, My decisions; Liberate the data; Connect up everything; Revolutionize health; and Include Everyone removing the current correlation between health and wealth.

EU Report on Redesigning health in Europe for 2020 (5:00)

12.4.6 Medicine and the Internet of Things

The Internet of Things will have great impact on health including telemedicine and wearables. Examples are given.

Medicine and the Internet of Things (8:17)

12.4.7 Extrapolating to 2032

A study looks at 4 scenarios for healthcare in 2032. Two are positive, one middle of the road and one negative.

Extrapolating to 2032 (15:13)

12.4.8 Genomics, Proteomics and Information Visualization

A study of an Azure application with an Excel frontend and a cloud BLAST backend starts this lesson. This is followed by a big data analysis of personal genomics and an analysis of a typical DNA sequencing analytics pipeline. The Protein Sequence Universe is defined and used to motivate Multi dimensional Scaling MDS. Sammon's method is defined and its use illustrated by a metagenomics example. Subtleties in use of MDS include a monotonic mapping of the dissimilarity function. The application to the COG Proteomics dataset is discussed. We note that the MDS approach is related to the well known chisq method and some aspects of nonlinear minimization of chisq (Least Squares) are discussed.

Genomics, Proteomics and Information Visualization (6:56)

Next we continue the discussion of the COG Protein Universe introduced in the last lesson. It is shown how Proteomics clusters are clearly seen in the Universe browser. This motivates a side remark on different clustering methods applied to metagenomics. Then we discuss the Generative Topographic Map GTM method that can be used in dimension reduction when original data is in a metric space and is in this case faster than MDS as GTM computational complexity scales like N not N squared as seen in MDS.

Examples are given of GTM including an application to topic models in Information Retrieval. Indiana University has developed a deterministic annealing improvement of GTM. 3 separate clusterings are projected for visualization and show very different structure emphasizing the importance of visualizing results of data analytics. The final slide shows an application of MDS to generate and visualize phylogenetic trees.

Genomics, Proteomics and Information Visualization I (10:33)

Genomics, Proteomics and Information Visualization: II (7:41)

Proteomics and Information Visualization (131)

12.4.9 Resources

- https://wiki.nci.nih.gov/display/CIP/CIP+Survey+of+Biomedical+Imaging+Archives [50]
- http://grids.ucs.indiana.edu/ptliupages/publications/Where\%20does\%20all\%20the\%20data\%20come\%20from\%20v7.pdf [2]
- ~~http://www.ieee-icsc.org/IC5C2010/Tony\%20Hey\%20-\%2020100923.pdf~~(this link does not exist any longer)
- http://quantifiedself.com/larry-smarr/ [51]
- http://www.ebi.ac.uk/Information/Brochures/ [52]
- http://www.kpcb.com/internet-trends **???**

- http://www.slideshare.net/drsteventucker/wearable-health-fitness-trackers-and-the-quantified-self [53]
- http://www.siam.org/meetings/sdm13/sun.pdf **???** –big-data-analytics-healthcare
- http://en.wikipedia.org/wiki/Calico_\%28company\%29 [54]
- http://www.slideshare.net/GSW_Worldwide/2015-health-trends **???** trends
- http://www.accenture.com/SiteCollectionDocuments/PDF/Accenture-Industrial-Internet-Changing-Competitive-Landscape-Industries.pdf [55]
- http://www.slideshare.net/schappy/how-realtime-analysis-turns-big-medical-data-into-precision-medicine [56]
- http://medcitynews.com/2013/03/the-body-in-bytes-medical-images-as-a-source-of-healthcare-big-data-infographic/ [57]
- ~~http://healthinformatics.wikispaces.com/file/view/cloud_computing.ppt~~ (this link does not exist any longer)
- https://www.mckinsey.com/~/media/mckinsey/industries/healthcare%20systems%20and%20services/our%20insights/the%20big%20data%20revolution%20in%20us%20health%20care/the_big_data_revol **???**
- ~~https://partner.microsoft.com/download/global/40193764~~ (this link does not exist any longer)
- https://ec.europa.eu/eip/ageing/file/353/download_en?token=8gECi1RO
- ~~http://www.liveathos.com/apparel/app~~
- http://debategraph.org/Poster.aspx?aID=77 [58]
- ~~http://www.oerc.ox.ac.uk/downloads/presentations-from-events/microsoftworkshop/gannon~~(this link does not exist any longer)
- ~~http://www.delsall.org~~ (this link does not exist any longer)
- http://salsahpc.indiana.edu/millionseq/mina/16SrRNA_index.html [59]
- http://www.geatbx.com/docu/fcnindex-01.html **???**

13 BIGDATA TECHNOLOGIES AND ALGORITHMS

In this section we will introduce you to additional technologies and algorithms frequently associated with Big Data. Previously we already introduced you to topics such as

- Recommender SYstems as shown in Section Recommender Systems
- k Nearest Neighbour as shown in Section kNN

We introduce you here to plotviz that allows us to display large numbers of points in 3D, as well as k-means.

13.1 STATISTICS

We assume that you are familiar with elementary statistics including

- mean, minimum, maximum
- standard deviation
- probability
- distribution
- frequency distribution
- Gaussian distribution
- bell curve
- standard normal probabilities
- tables (z table)
- Regression
- Correlation

Some of these terms are explained in various sections throughout our application discussion. This includes especially the Physics section. However these terms are so elementary that any undergraduate or highschool book will provide you with a good introduction.

It is expected from you to identify these terms and you can contribute to this section with non plagiarized subsections explaining these topics for credit.

O Topics identified by a :?: can be contributed by students. If you are interested, use piazza for announcing your willingness to do so.

Mean, minimum, maximum:

O ?

Standard deviation:

O ?

Probability:

O ?

Distribution:

O ?

Frequency distribution:

O ?

Gaussian distribution:

O ?

Bell curve:

O ?

Standard normal probabilities:

O ?

Tables (z-table):

O ?

Regression:

O ?

Correlation:

O ?

13.1.1 Exercise

E.Statistics.1:

> Pick a term from above and define it while not plagiarizing. Create a pull request. Coordinate on piazza as to not duplicate someone else's contribution. Also look into outstanding pull requests.

E.Statistics.2:

> Pick a term above and develop a python program demonstrating it and create a pull request for a contribution into the examples directory. Make links to the github location. Coordinate on piazza as to not duplicate someone else's contribution. Also look into outstanding pull requests.

13.2 PRACTICAL K-MEANS, MAP REDUCE, AND PAGE RANK FOR BIG DATA APPLICATIONS AND ANALYTICS

We use the K-means Python code in SciPy package to show real code for clustering. After a simple example we generate 4 clusters of distinct centers and various choice for sizes using Matplotlib tor visualization. We show results can sometimes be incorrect and sometimes make different choices among comparable solutions. We discuss the hill between different solutions and rationale for running K-means many times and choosing best answer. Then we introduce MapReduce with the basic architecture and a homely example. The discussion of advanced topics includes an extension to Iterative MapReduce from Indiana University called Twister and a generalized Map Collective model. Some measurements of parallel performance are given. The SciPy K-means code is modified to support a MapReduce execution style. This illustrates the key ideas of mappers and reducers. With appropriate runtime this code would run in parallel but here the parallel maps run sequentially. This simple 2 map version can be generalized to scalable parallelism. Python is used to Calculate PageRank from Web Linkage Matrix showing several different formulations of the basic matrix equations to finding leading eigenvector. The unit is concluded by a calculation of PageRank for general web pages by extracting the secret from Google.

K-Means I (11:42)

K-Means II (11:54)

13.2.1 K-means in Practice

We introduce the k means algorithm in a gentle fashion and describes its key features including dangers of local minima. A simple example from Wikipedia is examined.

We use the K-means Python code in SciPy package to show real code for clustering. After a simple example we generate 4 clusters of distinct centers and various choice for sizes using Matplotlib tor visualization. We show results can sometimes be incorrect and sometimes make different choices among comparable solutions. We discuss the hill between different solutions and rationale for running K-means many times and choosing best answer.

Files:

- https://github.com/cloudmesh-community/book/blob/master/examples/python/kmeans/xmean.py
- https://github.com/cloudmesh-community/book/blob/master/examples/python/kmeans/sample.csv
- https://github.com/cloudmesh-community/book/blob/master/examples/python/kmeans/parallel-kmeans.py
- https://github.com/cloudmesh-community/book/blob/master/examples/python/kmeans/kmeans-extra.py

13.2.1.1 K-means in Python

We use the K-means Python code in SciPy package to show real code for clustering and applies it a set of 85 two dimensional vectors – officially sets of weights and heights to be clustered to find T-shirt sizes. We run through Python code with Matplotlib displays to divide into 2-5 clusters. Then we discuss Python to generate 4 clusters of varying sizes and centered at corners of a square in two dimensions. We formally give the K means algorithm better than before and make definition consistent with code in SciPy.

13.2.1.2 Analysis of 4 Artificial Clusters

We present clustering results on the artificial set of 1000 2D points described in previous lesson for 3 choices of cluster sizes small large and very large. We emphasize the SciPy always does 20 independent K means and takes the best result – an approach to avoiding local minima. We allow this number of independent runs to be changed and in particular set to 1 to generate more interesting erratic results. We define changes in our new K means code that also has two measures of quality allowed. The slides give many results of clustering into 2 4 6 and 8 clusters (there were only 4 real clusters). We show that the very small case has two very different solutions when clustered into two clusters and use this to discuss functions with multiple minima and a hill between them. The lesson has both discussion of already produced results in slides and interactive use of Python for new runs.

13.2.2 Parallel K-means

We modify the SciPy K-means code to support a MapReduce execution style and runs it in this short unit. This illustrates the key ideas of mappers and reducers. With appropriate runtime this code would run in parallel but here the parallel maps run sequentially. We stress that this simple 2 map version can be generalized to scalable parallelism.

Files:

- https://github.com/cloudmesh-community/book/blob/master/examples/python/kmeans/parallel-kmeans.py

13.2.3 PageRank in Practice

We use Python to Calculate PageRank from Web Linkage Matrix showing several different formulations of the basic matrix equations to finding leading eigenvector. The unit is concluded by a calculation of PageRank for general web pages by extracting the secret from Google.

Files:

- https://github.com/cloudmesh-community/book/blob/master/examples/python/page-rank/pagerank1.py
- https://github.com/cloudmesh-community/book/blob/master/examples/python/page-rank/pagerank2.py

13.2.4 Resources

- https://en.wikipedia.org/wiki/Kmeans
- http://grids.ucs.indiana.edu/ptliupages/publications/DACIDR_camera_ready_v0.3.pdf
- http://salsahpc.indiana.edu/millionseq/
- http://salsafungiphy.blogspot.com/
- https://en.wikipedia.org/wiki/Heuristic

13.3 Plotviz

We introduce Plotviz, a data visualization tool developed at Indiana University to display 2 and 3 dimensional data. The motivation is that the human eye is very good at pattern recognition and can see structure in data. Although most Big data is higher dimensional than 3, all can be transformed by dimension reduction techniques to 3D. He gives several examples to show how the software can be used and what kind of data can be visualized. This includes individual plots and the manipulation of multiple synchronized plots.Finally, he describes the download and software dependency of Plotviz.

13.3.1 Using Plotviz Software for Displaying Point Distributions in 3D

We introduce Plotviz, a data visualization tool developed at Indiana University to display 2 and 3 dimensional data. The motivation is that the human eye is very good at pattern recognition and can see structure in data. Although most Big data is higher dimensional than 3, all can be transformed by dimension reduction techniques to 3D. He gives several examples to show how the software can be used and what kind of data can be visualized. This includes individual plots and the manipulation of multiple synchronized plots. Finally, he describes the download and software dependency of Plotviz.

Plotviz (34)

Files:

- https://github.com/cloudmesh-community/book/blob/master/examples/python/plotviz/fungi-lsu-3-15-to-3-26-zeroidx.pviz
- https://github.com/cloudmesh-community/book/blob/master/examples/python/plotviz/datingrating-originallabels.pviz
- https://github.com/cloudmesh-community/book/blob/master/examples/python/plotviz/clusterFinal-M30-C28.pviz
- https://github.com/cloudmesh-community/book/blob/master/examples/python/plotviz/clusterfinal-m3-c3dating-reclustered.pviz

13.3.1.1 Motivation and Introduction to use

The motivation of Plotviz is that the human eye is very good at pattern recognition and can see structure in data. Although most Big data is higher dimensional than 3, all data can be transformed by dimension reduction techniques to 3D and one can check analysis like clustering and/or see structure missed in a computer analysis. The motivations shows some Cheminformatics examples. The use of Plotviz is started in slide 4 with a discussion of input file which is either a simple text or more features (like colors) can be specified in a rich XML syntax. Plotviz deals with points and their classification (clustering). Next the protein sequence browser in 3D shows the basic structure of Plotviz interface. The next two slides explain the core 3D and 2D manipulations respectively. Note all files used in examples are available to students.

Motivation (7:58)

13.3.1.2 Example of Use I: Cube and Structured Dataset

Initially we start with a simple plot of 8 points – the corners of a cube in 3 dimensions – showing basic operations such as size/color/labels and Legend of points. The second example shows a dataset (coming from GTM dimension reduction) with significant structure. This has .pviz and a .txt versions that are compared.

Example I (9:45)

13.3.1.3 Example of Use II: Proteomics and Synchronized Rotation

This starts with an examination of a sample of Protein Universe Browser showing how one uses Plotviz to look at different features of this set of Protein sequences projected to 3D. Then we show how to compare two datasets with synchronized rotation of a dataset clustered in 2 different ways; this dataset comes from k Nearest Neighbor discussion.

Proteomics and Synchronized Rotation (9:14)

13.3.1.4 Example of Use III: More Features and larger Proteomics Sample

This starts by describing use of Labels and Glyphs and the Default mode in Plotviz. Then we illustrate sophisticated use of these ideas to view a large Proteomics dataset.

Larger Proteomics Sample (8:37)

13.3.1.5 Example of Use IV: Tools and Examples

This lesson starts by describing the Plotviz tools and then sets up two examples – Oil Flow and Trading – described in PowerPoint. It finishes with the Plotviz viewing of Oil Flow data.

Plotviz I (10:17)

13.3.1.6 Example of Use V: Final Examples

This starts with Plotviz looking at Trading example introduced in previous lesson and then examines solvent data. It finishes with two large biology examples with 446K and 100K points and each with over 100 clusters. We finish remarks on Plotviz software structure and how to download. We also remind you that a picture is worth a 1000 words.

Plotviz II (14:58)

13.3.2 Resources

Download

14 Development Tools and Services

14.1 Refcards

Learning Objectives

- Obtain quickly information about technical aspects with the help of reference cards.

We present you with a list of useful short reference cards. This cards can be extremely useful to remind yourself about some important commands and features. Having them could simplify your interaction with the systems, We not only collected here some refcards about Linux, but also about other useful tools and services.

If you like to add new topics, let us know via your contribution (see the contribution section).

CheatSheets

- CheatSheets

Editors

- Emacs
- Vi
- Vim

Documentation

- LaTeX
- RST

Linux

- Linux
- Makefile
- Git

Cloud/Virtualization

- Openstack
- Openstack
- vagrant

SQL

- SQL

Languages

- R

Python

- Python
- PythonData
- Numpy/Pandas
- PythonTutorial
- Python
- Python
- PythonAPIIndex
- Python3

14.2 Virtual Box

For development purposes we recommend that you use for this class an Ubuntu virtual machine that you set up with the help of virtualbox. We recommend that you use the current version of ubuntu and do not install or reuse a version that you have set up years ago.

As access to cloud resources requires some basic knowledge of linux and security we will restrict access to our cloud services to those that have demonstrated responsible use on their own computers. Naturally as it is your own computer you must make sure you follow proper security. We have seen in the past students carelessly working with virtual machines and introducing security vulnerabilities on our clouds just because “it was not their computer.” Hence, we will allow using of cloud resources only if you have demonstrated that you responsibly use a linux virtual machine on your own computer. Only after you have successfully used ubuntu in a virtual machine you will be allowed to use virtual machines on clouds.

A cloud drivers license test will be conducted. Only after you pass it we wil let you gain access to the cloud infrastructure. We will announce this test. Before you have not passed the test, you will not be able to use the clouds. Furthermore, you do not have to ask us for join requests to cloud projects before you have not passed the test. Please be patient. Only students enrolled in the class can get access to the cloud.

If you however have access to other clouds yourself you are welcome to use the, However, be reminded that projects need to be reproducible, on our cloud. This will require you to make sure a TA can replicate it.

Let us now focus on using virtual box.

14.2.1 Installation

First you will need to install virtualbox. It is easy to install and details can be found at

- https://www.virtualbox.org/wiki/Downloads

After you have installed virtualbox you also need to use an image. For this class we will be using ubuntu Desktop 16.04 which you can find at:

- http://www.ubuntu.com/download/desktop

Please note some hardware you may have may be too old or has too little resources to be useful. We have heard from students that the following is a minimal setup for the desktop machine:

- multi core processor or better allowing to run hypervisors
- 8 GB system memory
- 50 GB of free hard drive space

For virtual machines you may need multiple, while the minimal configuration may not work for all cases.

As configuration we often use

minimal

: 1 core, 2GB Memory, 5 GB disk

latex

: 2 core, 4GB Memory, 25 GB disk

A video to showcase such an install is available at:

Using Ubuntu in Virtualbox (8:08)

⚠ Please note that the video shows the version 16.04. You should however use the newest version which at this time is 18.04.

If you specify your machine too small you will not be able to install the development environment. Gregor used on his machine 8GB RAM and 25GB diskspace.

Please let us know the smallest configuration that works.

14.2.2 Guest additions

The virtual guest additions allow you to easily do the following tasks:

- Resize the windows of the vm
- Copy and paste content between the Guest operating system and the host operating system windows.

This way you can use many native programs on you host and copy contents easily into for example a terminal or an editor that you run in the Vm.

A video is located at

Virtualbox (4:46)

Please reboot the machine after installation and configuration.

On OSX you can once you have enabled bidirectional copying in the Device tab with

OSX to Vbox:

command c shift CONTRL v

Vbox to OSX:

shift CONTRL v shift CONTRL v

On Windows the key combination is naturally different. Please consult your windows manual. If you let us know TAs will add the information here.

14.2.3 Exercises

E.Virtualbox.1:

Install ubuntu desktop on your computer with guest additions.

E.Virtualbox.2:

Make sure you know how to paste and copy between your host and guest operating system.

E.Virtualbox.3:

Install the programs defined by the development configuration.

E.Virtualbox.4:

Provide us with the key combination to copy and paste between Windows and Vbox.

14.3 VAGRANT

Learning Objectives

- Be able to experiment with virtual machines on your computer before you go on a cloud.
- Simulate a virtual cluster with multiple VMs running on your computer if it is big enough.

A convenient tool to interface with Virtual Box is vagrant.Vagrant allows us to manage virtual machines directly from the commandline. It support also other providers and can be used to start virtual machines and even containers. The latest version of vagrant includes the ability to automatically fetch a virtual machine image and start it on your local computer. It assumes that you have virtual box installed. Some key concepts and advertisement are located at

- https://www.vagrantup.com/intro/index.html:

Detailed documentation for it is located

- https://www.vagrantup.com/docs/index.html

A list of boxes is available from

- https://app.vagrantup.com/boxes/search

One image we will typically use is Ubuntu 18.04. Please note that older version may not be suitable for class and we will not support any questions about them. This image is located at

- https://app.vagrantup.com/ubuntu/boxes/bionic64

14.3.1 Installation

Vagrant is easy to install. You can go to the download page and download and install the appropriate version:

- https://www.vagrantup.com/downloads.html

14.3.1.1 macOS

On MacOS, download the dmg image, and click on it. You will find a pkg in it that you double click. After installation vagrant is installed in

- `/usr/local/bin/vagrant`

Make sure `/usr/local/bin` is in your `PATH` Start a new terminal to verify this.

Check it with

```
echo $PATH
```

If it is not in the path put

export PATH=/usr/local/bin:$PATH

in the terminal command or in your `~/.bash_profile`

14.3.1.2 Windows

students contribute

14.3.1.3 Linux

students contribute

14.3.2 Usage

To download, start and login into install the 18.04:

```
host$ vagrant init ubuntu/bionic64
host$ vagrant up
host$ vagrant ssh
```

Once you are logged in you can test the version of python with

```
vagrant@ubuntu-bionic:~$ sudo apt-get update
vagrant@ubuntu-bionic:~$ python3 --version
Python 3.6.5
```

To install a newer version of python, and pip you can use

```
vagrant@ubuntu-bionic:~$ sudo apt-get install python3.7
vagrant@ubuntu-bionic:~$ sudo apt-get install python3-pip
```

To install the light weight idle development environment in case you do not want o use pyCharm, please use

```
vagrant@ubuntu-bionic:~$ sudo apt-get install idle-python
```

So that you do not have to always use the number 3, you can also set an alias with

```
alias python=python3
```

When you exit the virtual machine with the

```
exit command
```

It does not terminate the VM. You can use from your host system the commands such as

```
host$ vagrant status
host$ vagrant destroy
host$ vagrant suspend
host$ vagrant resume
```

to manage the vm.

14.4 Packer

Packer is an open source tool for creating identical machine images for multiple platforms from a single source configuration. Packer runs on every major operating system, and creates machine images for multiple platforms in parallel form configuration specufications.

Some key concepts are located at

- https://www.packer.io/intro/index.html

Detailed documentation is located at

- https://www.packer.io/docs/index.html

Use cases for packer is located at

- https://www.packer.io/intro/use-cases.html

14.4.1 Installation

Installation instructions for all platforms is located at

- https://www.packer.io/intro/getting-started/install.html

14.4.2 Usage

In the Section vagrant we use vagrant to start up an Ubuntu 18.04 virtual machine. Once the VM was up and running, vagrant allowed the user to log in and setup the VM according to the user's requirements. In that example, the user ran commands to install and upgrade software dependencies:

1. upgrade from Python 3.6.5 to Python 3.7
2. installing python3-pip and idle-python
3. alias `python` to `python3`

Let us assume that the VM is now in a desirable state for the purpose of doing development on a large number of virtual machines and you want to distribute it to the rest of your team or community so that all are using the same environment. You could simply send your team members a copy of your Ubuntu 18.04 VirtualBox VM assuming they will be developing on VMs using VirtualBox. However, let us assume one community member wants to develop on Google Cloud Platform, another on AWS and another on OpenStack. In this case, they will each need to figure out how to import a VirtualBox VM into the respective cloud vendor they're utilizing. Packer can help this situation by codifying the state of the development environment with a single configuration file which can then be used to create images in different cloud environments.

Assuming packer has been installed, let's create a packer JSON file that will build an Ubuntu 18.04 image and provision it as we did manually using Vagrant. In this example, we will create the image in Google Compute Platform.

First download your Google Cloud credentials according to the documentation at

- https://www.packer.io/docs/builders/googlecompute.html#running-without-a-compute-engine-service-account

Save the credential file as `accounts.json`. Also, determine the project ID you will use in your Google Cloud Platform account. In this example, we will use `my_project_id` for our project ID.

Next save the following JSON to a file named `e516.json`:

```
{
  "variables": {
    "google_project_id": null
  },
  "builders": [
    {
      "type":         "googlecompute",
      "account_file": "account.json",
      "project_id":   "{{ user `google_project_id` }}",
      "image_name":   "ubuntu-1804-dev-e516",
      "source_image": "ubuntu-1804-bionic-v20180911",
      "ssh_username": "packer",
      "zone":         "us-central1-a"
    }
  ],
  "provisioners": [
    {
      "type": "shell",
      "expect_disconnect": true,
      "inline": [
        "sudo apt-get update -y",
        "sudo apt-get install -y python3.7 python3-pip idle-python3.7",
        "echo \"alias python='python3'\" > .bash_aliases"
      ]
    }
  ]
}
```

The packer file format specifies 3 sections, `variables`, `builders` and `provisioners`. The `variables` section allows you to declare variables that are to be used in the rest of the document. By declaring a variable in this section, for example `google_project_id`, it allows the user to pass in the value of that variable via the packer command line.

The `builders` section allows you to declare the builders for any cloud vendor supported by packer. The list of supported vendors can be found here:

- https://www.packer.io/docs/builders/index.html

In our example, we define the builder for Google Cloud Platform which requires our credential file (account.json), our project ID, base image name, ssh username and zone.

Finally, the `provisioners` section allows the user to customize the base image defined in the `builders` section. In our example, we simply use the `shell` provisioner which allows us to type in shell commands to provision the image as we want it. Here we install python3.7, python3-pip and idle-python3.7. We also write out an aliases file so that upon login, the user can access python3.7 using the python alias.

To build the image, we now run packer:

```
$ packer build -var 'google_project_id=my_project_id' e516.json
```

You will see output that shows the progress of packer as it starts up and provisions the instance. Upon success, packer will create an image from the instance and clean up after itself:

```
$ googlecompute output will be in this color.

==> googlecompute: Checking image does not exist...
==> googlecompute: Creating temporary SSH key for instance...
==> googlecompute: Using image: ubuntu-1804-bionic-v20180911
```

```
==> googlecompute: Creating instance...
    googlecompute: Loading zone: us-central1-a
    googlecompute: Loading machine type: n1-standard-1
    googlecompute: Requesting instance creation...
    googlecompute: Waiting for creation operation to complete...
    googlecompute: Instance has been created!
==> googlecompute: Waiting for the instance to become running...
    googlecompute: IP: 104.154.21.240
==> googlecompute: Waiting for SSH to become available...
==> googlecompute: Connected to SSH!
==> googlecompute: Provisioning with shell script: /var/folders/rm/g1h4bhf54x750jzjyryckmnc000lxd/T/packer-shell210916201
    googlecompute: Get:1 http://archive.canonical.com/ubuntu bionic InRelease [10.2 kB]
...
    googlecompute: Setting up idle-python3.7 (3.7.0-1~18.04) ...
    googlecompute: Processing triggers for libc-bin (2.27-3ubuntu1) ...
    googlecompute: Processing triggers for ureadahead (0.100.0-20) ...
    googlecompute: Processing triggers for systemd (237-3ubuntu10.3) ...
==> googlecompute: Deleting instance...
    googlecompute: Instance has been deleted!
==> googlecompute: Creating image...
==> googlecompute: Deleting disk...
    googlecompute: Disk has been deleted!
Build 'googlecompute' finished.

==> Builds finished. The artifacts of successful builds are:
--> googlecompute: A disk image was created: ubuntu-1804-dev-e516
```

You can now click on the list of images in the Google Compute Platform console to see your new image. The new image is ready to use for development.

Next, let's add a builder for an AWS AMI. Before we do that, setup your AWS credentials using the AWS CLI according to the documentation here:

- https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-started.html

Ensure your `default` profile is saved under `~/.aws/credentials`.

Update the `e516.json` so that the contents is as follows:

```
{
  "variables": {
    "google_project_id": null,
    "image_name":   "ubuntu-1804-dev-e516",
    "ssh_username": "packer"
  },
  "builders": [
    {
      "type":         "googlecompute",
      "account_file": "account.json",
      "ssh_username": "{{ user `ssh_username` }}",
      "project_id":   "{{ user `google_project_id` }}",
      "image_name":   "{{ user `image_name` }}",
      "source_image": "ubuntu-1804-bionic-v20180911",
      "zone":         "us-central1-a"
    },
    {
      "type":         "amazon-ebs",
      "ssh_username": "{{ user `ssh_username` }}",
      "profile":      "default",
      "ami_name":   "{{ user `image_name` }}",
      "source_ami":   "ami-0bbe6b35405ecebdb",
      "instance_type":"t2.micro",
      "region":       "us-west-2"
    }
  ],
  "provisioners": [
    {
      "type": "shell",
      "expect_disconnect": true,
      "inline": [
        "sudo apt-get update -y",
        "sudo apt-get install -y python3.7 python3-pip idle-python3.7",
        "echo \"alias python='python3'\" > .bash_aliases"
      ]
    }
  ]
}
```

Note that we've added the AWS builder in the `builders` section and that we've refactored the `ssh_username` and `image_name` to the `variables` section since those variable hold values that can be reused in both the Google Compute and AWS builders.

Let's rerun packer:

```
packer build -var 'google_project_id=my_project_id' e516.json
```

You will see output that states the image already exists in your Google Compute account and so packer smartly skips building that image. The output also shows the progress of packer as it starts up and provisions the instance in AWS. Upon success, packer will create an AMI from the instance and clean up after itself:

```
amazon-ebs output will be in this color.
googlecompute output will be in this color.

==> googlecompute: Checking image does not exist...
==> amazon-ebs: Prevalidating AMI Name: ubuntu-1804-dev-e516
==> googlecompute: Image ubuntu-1804-dev-e516 already exists.
==> googlecompute: Use the force flag to delete it prior to building.
Build 'googlecompute' errored: Image ubuntu-1804-dev-e516 already exists.
Use the force flag to delete it prior to building.
    amazon-ebs: Found Image ID: ami-0bbe6b35405ecebdb
==> amazon-ebs: Creating temporary keypair:
    packer_5bad9d99-f631-1778-1e83-afd19ad0d5cc
==> amazon-ebs: Creating temporary security group for this instance:
    packer_5bad9d9b-38c5-252d-0368-74aa75bfb286
==> amazon-ebs: Authorizing access to port 22 from 0.0.0.0/0
    in the temporary security group...
==> amazon-ebs: Launching a source AWS instance...
==> amazon-ebs: Adding tags to source instance
    amazon-ebs: Adding tag: "Name": "Packer Builder"
    amazon-ebs: Instance ID: i-0d0383f9f84b54051
```

You can now click on the list of images in the AWS EC2 console to see your new AMI. The new AMI is ready to use for development.

14.5 Ubuntu on an USB stick

In case you cannot install any programs on your development computer most often the easiest way is to use the hardware but boot the OS from a USB stick. Make sure you have access to the Bios or your system to actually boot from a USB device before you start this activity.

14.5.1 Ubuntu on an USB stick for macOS via Command Line

The easiest way to create an ubuntu distribution that can be booted from an USB stick is done via command line. The original Web page for this method is available at this [link].

We have copied some of the information from this Web page but made enhancements to it. Currently all images are copied form that Web page.

⚠ Pleas test it out and improve if it does not work

Our goal is to create a USB stick that has either Ubuntu 18.04 LTS that can be downloaded from this [link]. You will need a USB stick/flash drive. We recommend a 8GB or larger. Please let us know if it works for you on larger than 8GB drives.

We assume that you downloaded the iso from ubuntu to a folder called /iso. Next we open a terminal and cd into the folder /iso. Now we need to convert the is to an image file. This is done as follows and you need to

execute the command for the version of ubuntu you like to use.

Your folder will look something like this

```
$ ls -1

   ubuntu-18.04-desktop-amd64.iso
```

You will need to generate an image with the following command

```
$ hdiutil convert ubuntu-18.04-desktop-amd64.iso -format UDRW -o ubuntu-18.04-desktop-amd64.img
```

macOS will append a .dmg behind the name. At this time **do not** plug in your usb stick. Just issue the command

```
$ diskutil list
```

Observe the output. Now plug in the USB stick. Wait till the USB stick registers in the Finder. If this does not work find a new USB stick or format it. Execute the command

```
$ diskutil list
```

and observer the output again. Another device will register and you will see something like

```
/dev/disk2 (external, physical):
#:     TYPE NAME              SIZE       IDENTIFIER
0:     FDisk_partition_scheme *8.2 GB     disk2
1:     DOS_FAT_32 NO NAME      8.2 GB     disk2s1
```

Please note in this example the device path and number is recognized as

```
/dev/disk2
```

It also says external, which is a good sign as the USB stick is external. Next, we need to unmount the device with

```
$ diskutil unmountDisk /dev/diskN
```

where you replace the number N with the disk number that you found for the device. In our example it would be 2. If you see the error “Unmount of diskN failed: at least one volume could not be unmounted”, start Disk Utility.app and unmount the volume (do not eject). If it was successful, you will see

```
Unmount of all volumes on disk2 was successful
```

The next step is dangerous and you need to make sure you follow it. So please do not copy and paste, but read first, reflect and only if you understand it execute it. We know we say this all the time, but better saying it again instead of you destroying your system. This command also requires sudo access so you will either have to be in the sudo group, or use

```
$ su <your administrator name>
```

login and than execute the command under root.

```
$ sudo dd if=ubuntu-18.04-desktop-amd64.img.dmg of=/dev/diskN bs=1m
```

(Not tested: Using /dev/rdisk instead of /dev/disk may be faster according to the ubuntu documentation)

Ubuntu’s Web page also gives the following tips:

- “If you see the error dd: Invalid number ‘1m’, you are using GNU dd. Use the same command but replace bs=1m with bs=1M.”
- “If you see the error dd: /dev/diskN: Resource busy, make sure the disk is not in use. Start Disk Utility.app and unmount the volume (do not eject).”

You will see an error window popping up telling you: **The disk inserted was not readable by this compute**. Please, leave the window as is and instead type in on the terminal.

```
$ diskutil eject /dev/diskN
```

Now remove the flash drive, and press in the error window **Ignore**
Now you have a flash drive with ubuntu installed and you can boot from it. To do so, please

restart your Mac and press option key

while the Mac is restarting to choose the USB-Stick

You will need a plug for USB keyboard, USB mouse, and network cable.

There are some issue from this point on.

```
$ sudo apt-get update
```

Add universe to the window for application updates

see https://help.ubuntu.com/community/Repositories/Ubuntu

```
$ sudo apt-get install vnc4server
```

Start the server and set up a password

```
$ vncserver
```

The next section is untested and needs verification.

14.5.1.1 Boot from the USB Stick

To boot from the USB stick, you need to restart or power-on the Mac with the USB stick inserted while you press the Option/alt key.

The launch Startup Manager will be started showing a list of bootable devices connected to the machine. Your USB stick should appear as gold/yellow and labelled EFI Boot. Use your cursor keys to move to the most right EFI boot device in that list (likely the USB stick) and press ENTER. YOu can also use the mouse.

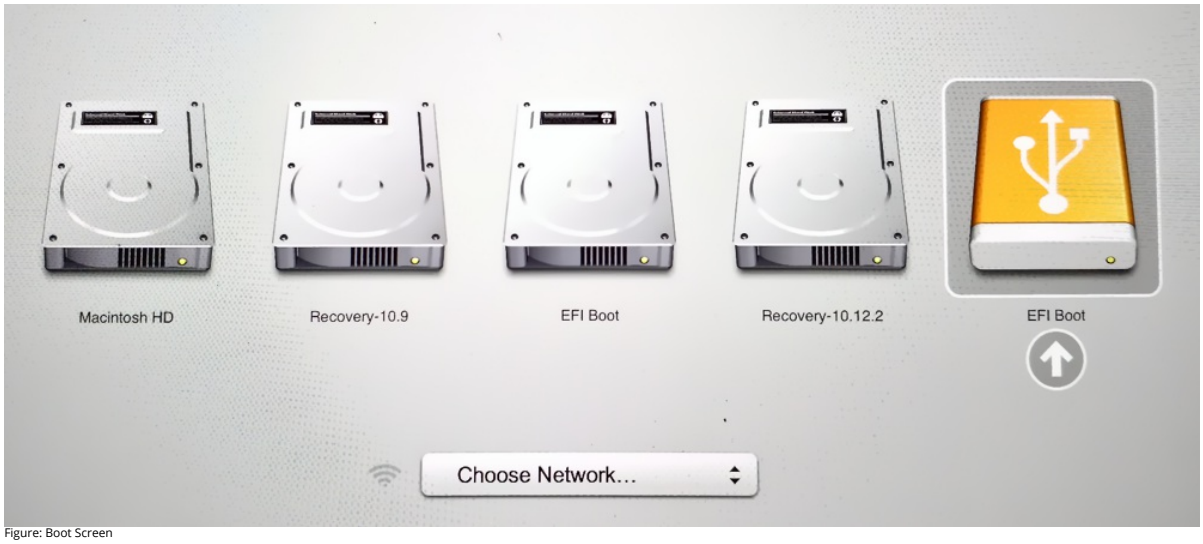


Figure: Boot Screen

A boot menu will shortly start up and after you press again ENTER your machine will boot into Ubuntu.

For more information on how to setup ubuntu see:

- https://tutorials.ubuntu.com/tutorial/tutorial-install-ubuntu-desktop#0

After you have booted and logged in, you need to update the distribution. We recommend that you switch on Universe in the applications settings.

Next you need to issue in the command terminal

```
$ sudo apt-get update
```

You will likely see some warnings with number 95 which you can ignore. Please report your experience and we update this page based on your feedback.

14.5.2 Ubuntu on an USB stick for macOS via GUI

An alternative to the Command Line solution to create an USB stick with bootable Ubuntu on is to use the macOS GUI. This method is more complex than the command line solution. In addition as we are learning about cloud computing in this book, it is of advantage to learn how to do this from commandline as the replication of the approach via commandline is easier and more scalable. However for completeness, we have also included here the GUI-based method.

The material in this section was copied and modified from

- https://tutorials.ubuntu.com/tutorial/tutorial-create-a-usb-stick-on-macos

You will need a USB stick/flash drive. We recommend a 8GB or larger. Please let us know if it works for you on larger than 8GB drives.

14.5.2.1 Install Etcher

Etcher is a tool that allows you to easily write an ISO onto a USB stick. Etcher is integrated in the macOS GUI environment and allows to drag the iso into it for burning. Etcher can be found at

- https://etcher.io/

As this is an application from unidentified developers (not registered in the apple store), you need to enable it after downloading. To do so, you can enable the App Store and identified developers in the Security and Privacy pane in the System Preferences. IN case you get a warning about running the application, click Open Anyway in the same pane.

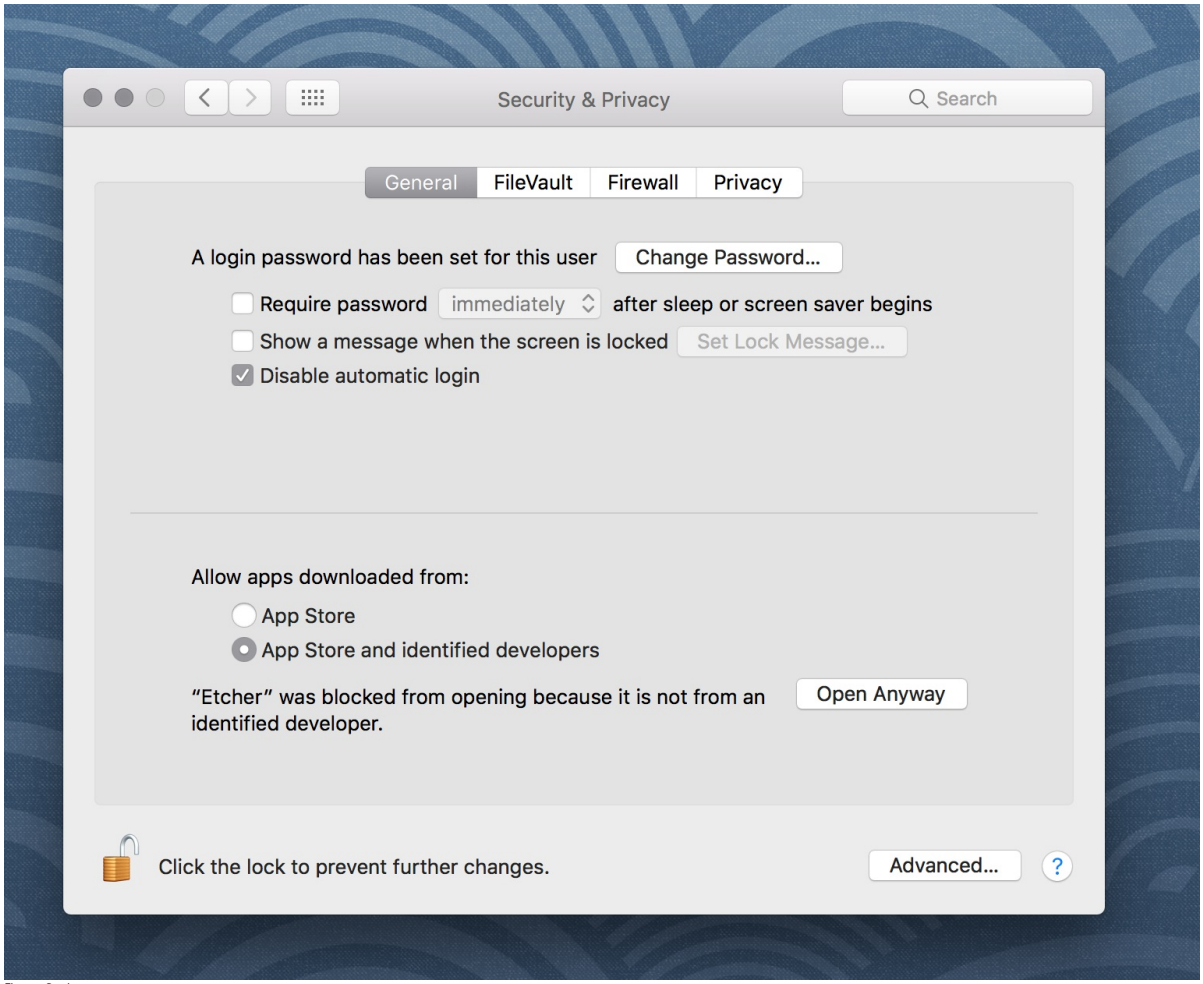


Figure: Setting

14.5.2.2 Prepare the USB stick

The Disk Utility needs to be used with caution as selecting the wrong device or partition can result in data loss.

Next you need to conduct the following steps which we copied from the Ubuntu Web page:

- Launch Disk Utility from Applications>Utilities or Spotlight search
- Insert your USB stick and observe the new device added to Disk Utility
- Select the USB stick device and select Erase from the tool bar (or right-click menu)
- Set the format to MS-DOS (FAT) and the scheme to GUID Partition Map Check you've chosen the correct device and click Erase

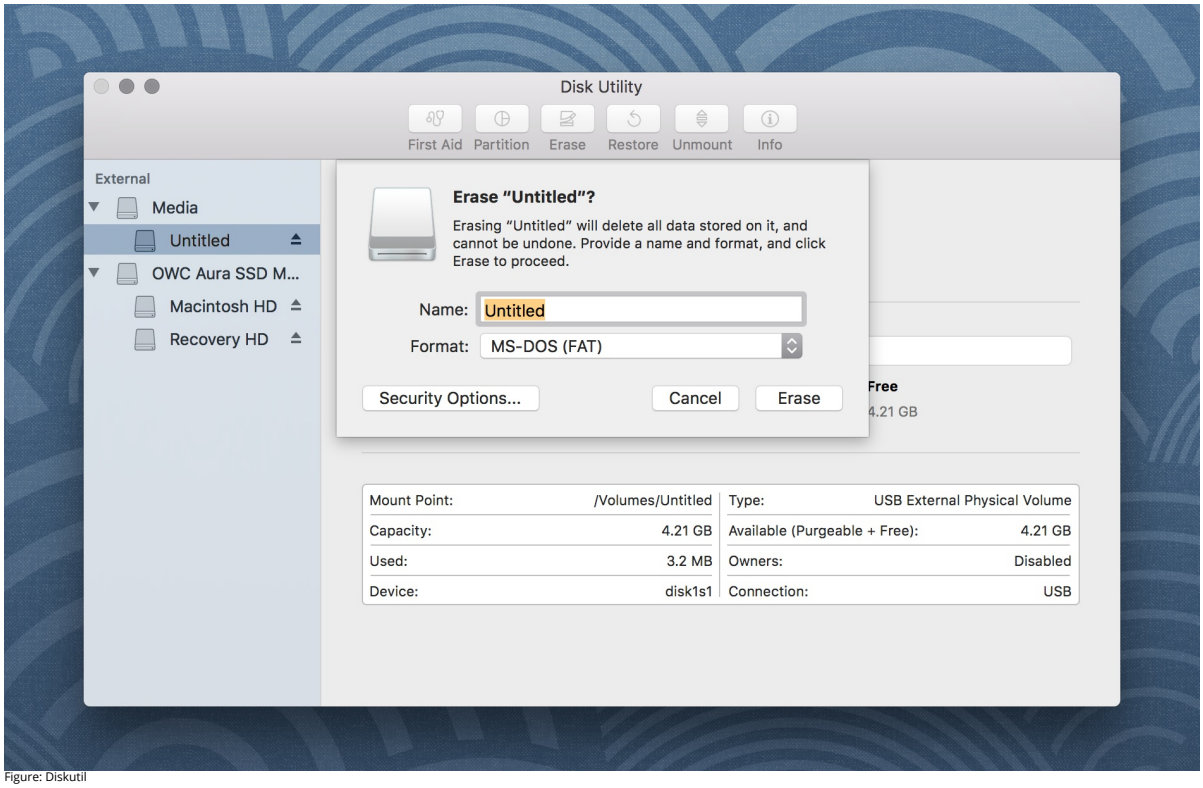


Figure: Diskutil

14.5.2.3 Etcher configuration

Next we use Etcher to configure and write to your USB device as follows (copied form the Ubuntu Web page):

- Select image will open a file requester from which should navigate to and select the ISO file downloaded previously. By default, the ISO file will be in your Downloads folder.
- Select drive, replaced by the name of your USB device if one is already attached, lets you select your target device. You will be warned if the storage space is too small for your selected ISO.
- Flash! will activate when both the image and the drive have been selected. As with Disk Utility, Etcher needs low-level access to your storage hardware and will ask for your password after selection.

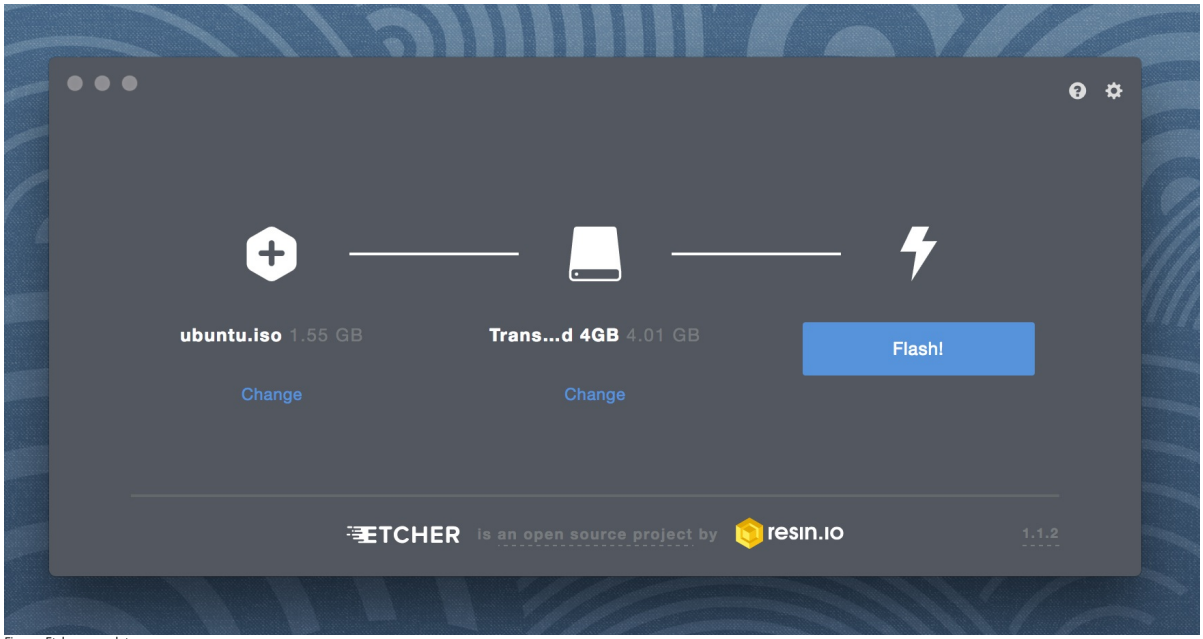


Figure: Etcher complete message

14.5.2.4 Write to the USB stick

When writing to the USB, Etcher will ask you for your password. It will write the ISO file, once you confirmed the password.

You will see the progress reported to the Etcher window. Once it has finished, Etcher will report on the successful process.

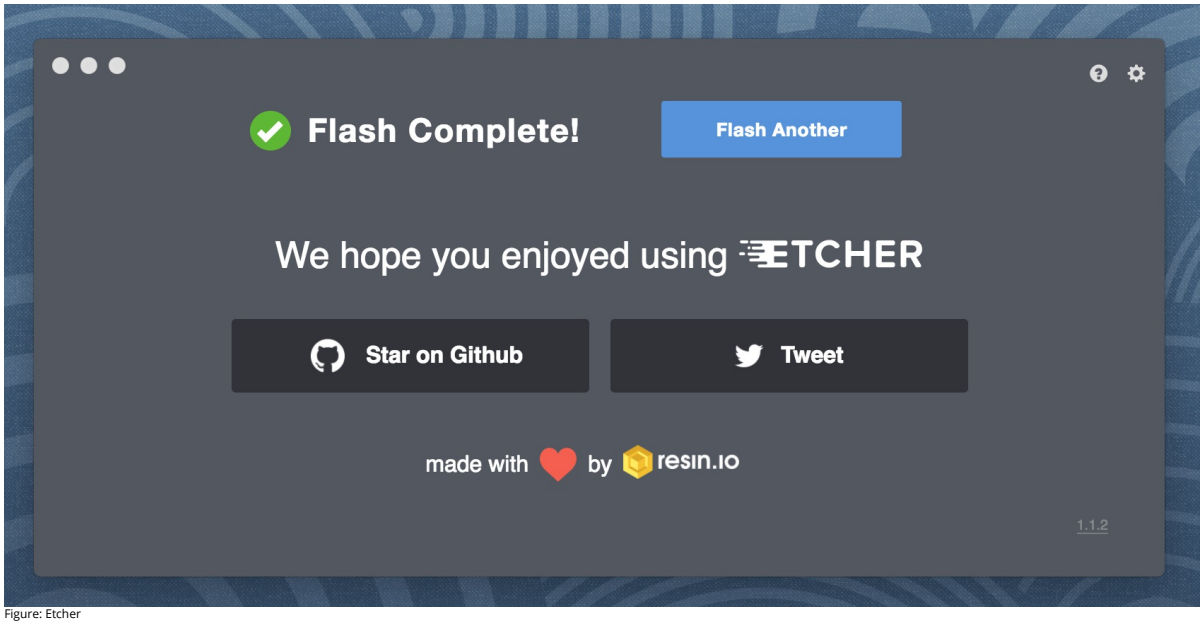


Figure: Etcher

After the write process has completed, macOS may inform you that *The disk you inserted was not readable by this computer*. Do not select Initialise. Instead, select Eject and remove the USB device.

14.5.3 Ubuntu on an USB stick for Windows 10

See exercise Development.Server.1

Material for this directions were taken from a detailed tutorial [link]

First you will need to install Rufus, which is a free program to create bootable USB drives on windows. Rufus is available at

- https://rufus.akeo.ie/

Next you need to launch Rufus, insert the USB stick, and observe that it is added to Rufus. Select the Device on which you like to place ubuntu. Be careful that you do not bya accident use a wrong device.

Select the partition scheme and target system type set as MBR partition scheme for UEFI. (in case you have older hardware try MBR Partition Scheme for BIOS or UEFI instead).

Select the ubuntu iso file.

Next press the `Start` button so we activate the write process. This will take quite a while. Select `Write in ISO Image mode (Recommended)`

Once the process is completed, try booting from it. How to activate the boot in your system depends on your hardware and vendor. Please consult with your documentation.

14.5.4 Exercise

Development.Server.1

> If you are in need to but from a USB stick in Windows, please verify and expand on our section similar to the one provided by macOS. It does not matter if you chose a GUI or a commandline option via gitbash.

14.6 Github

Learning Objectives

- Be able to use the github cloud sevices to collaborately develop contents and programs.
- Be able to use github as part of an open source project.

In some classes the material may be openly shared in code repositories. This includes class material, papers and project. Hence, we need some mechanism to share content with a large number of students.

First, we like to introduce you to git and github.com (Section 1.1). Next, we provide you with the basic commands to interact with git from the commandline (Section 1.12). Than we will introduce you how you can contribute to this set of documentations with pull requests.

14.6.1 Overview

Github is a code repository that allows the development of code and documents with many contributors in a distributed fashion. There are many good tutorials about github. Some of them can be found on the github Web page. An interactive tutorial is for example available at

- https://try.github.io/

However, although these tutorials are helpful in many cases they do not address some cases. For example, you have already a repository set up by your organization and you do not have to completely initialize it. Thus do not just replicate the commands in the tutorial, or the once we present here before not evaluating their consequences. In general make sure you verify if the command does what you expect **before** you execute it.

A more extensive list of tutorials can be found at

- https://help.github.com/articles/what-are-other-good-resources-for-learning-git-and-github

The github foundation has a number of excellent videos about git. If you are unfamiliar with git and you like to watch videos in addition to reading the documentation we recommend these videos

- https://www.youtube.com/user/GitHubGuides/videos

Next, we introduce some important concepts used in github.

14.6.2 Upload Key

Before you can work with a repository in an easy fashion you need to upload a public key in order to access your repository. Naturally, you need to generate a key first which is explained in the section about ssh key generation (TODO: lessons-ssh-generate-key include link) before you upload one. Copy the contents of your `.ssh/id_rsa.pub` file and add them to your github keys.

More information on this topic can be found on the github Web page.

14.6.3 Fork

Forking is the first step to contributing to projects on GitHub. Forking allows you to copy a repository and work on it under your own account. Next, creating a branch, making some changes, and offering a pull request to the original repository, rounds out your contribution to the open source project.

Git 1:41 Fork

14.6.4 Rebase

When you start editing your project, you diverge from the original version. During your developing, the original version may be updated, or other developers may have some of their branches implementing good features that you would like to include in your current work. That is when Rebase becomes useful. When you Rebase to certain points, could be a newer Master or other custom branch, consider you graft all your on-going work right to that point.

Rebase may fail, because some times it is impossible to achieve what we just described as conflicts may exist. For example, you and the to-be-rebased copy both edited some common text section. Once this happens, human intervention needs to take place to resolve the conflict.

Git 4:20 Rebase

14.6.5 Remote

Collaborating with others involves managing the remote repositories and pushing and pulling data to and from them when you need to share work. Managing remote repositories includes knowing how to add remote repositories, remove remotes that are no longer valid, manage various remote branches and define them as being tracked or not, and more.

Though out this semester, you will typically work on two remote repos. One is the office class repo, and another is the repo you forked from the class repo. The class repo is used as the centralized, authority and final version of all student submissions. The repo under your own Github account is for your personal storage. To show progress on a weekly basis you need to commit your changes on a weekly basis. However make sure that things in the master branch are working. If not, just use another branch to conduct your changes and merge at a later time. We like you to call your development branch dev.

- https://git-scm.com/book/en/v2/Git-Basics-Working-with-Remotes

14.6.6 Pull Request

Pull requests are a means of starting a conversation about a proposed change back into a project. We will be taking a look at the strength of conversation, integration options for fuller information about a change, and cleanup strategy for when a pull request is finished.

Git 4:26 Pull Request

14.6.7 Branch

Branches are an excellent way to not only work safely on features or experiments, but they are also the key element in creating Pull Requests on GitHub. Lets take a look at why we want branches, how to create and delete branches, and how to switch branches in this episode.

Git 2:25 Branch

14.6.8 Checkout

Change where and what you are working on with the checkout command. Whether we are switching branches, wanting to look at the working tree at a specific commit in history, or discarding edits we want to throw away, all of these can be done with the checkout command.

Git 3:11 Checkout

14.6.9 Merge

Once you know branches, merging that work into master is the natural next step. Find out how to merge branches, identify and clean up merge conflicts or avoid conflicts until a later date. Lastly, we will look at combining the merged feature branch into a single commit and cleaning up your feature branch after merges.

Git 3:11 Merge

14.6.10 GUI

Using Graphical User Interfaces can supplement your use of the command line to get the best of both worlds. GitHub for Windows and GitHub for Mac allow for switching to command line, ease of grabbing repositories from GitHub, and participating in a particular pull request. We will also see the auto-updating functionality helps us stay up to date with stable versions of Git on the command line.

Git 3:47 GUI

There are many other git GUI tools available that directly integrate into your operating system finders, windows, ..., or PyCharm. It is up to you to identify such tools and see if they are useful for you. Most of the people we work with us git from the command line, even if they use PyCharm, eclipse, or other tools that have build in git support. You can identify a tool that works best for you.

14.6.11 Windows

This is a quick tour of GitHub for Windows. It offers GitHub newcomers a brief overview of what this feature-loaded version control tool and an equally powerful web application can do for developers, designers, and managers using Windows in both the open source and commercial software worlds. More: http://windows.github.com

Git 1:25 Windows

14.6.12 Git from the Commandline

Although github.com provides a powerful GUI and other GUI tools are available to interface with github.com, the use of git from the commandline can often be faster and in many cases may be simpler.

Git commandline tools can be easily installed on a variety of operating systems including Linux, macOS, and Windows. Many great tutorials exist that will allow you to complete this task easily. We found the following two tutorials sufficient to get the task accomplished:

- https://git-scm.com/book/en/v2/Getting-Started-Installing-Git
- https://www.atlassian.com/git/tutorials/install-git

Although the later is provided by an alternate repository to github. The installation instructions are very nice and are not impacted by it. Once you have installed git you need to configure it.

14.6.13 Configuration

Once you installed Git, you can need to configure it properly. This includes setting up your username, email address, line endings, and color, along with the settings' associated configuration scopes.

Git 2:47 Configuration

It is important that make sure that use the `git config` command to initialize git for the first time on each new computer system or virtual machine you use. This will ensure that you use on all resources the same name and e-mail so that git history and log will show consistently your checkins across all devices and computers you use. If you do not do this, your checkins in git do not show up in a consistent fashion as a single user. Thus on each computer execute the following commands:

```
$ git config --global user.name "Albert Zweistein"
$ git config --global user.email albert.zweistein@gmail.com
```

where you replace the information with the information related to you. You can set the editor to emacs with:

```
$ git config --global core.editor emacs
```

Naturally if you happen to want to use other editors you can configure them by specifying the command that starts them up. You will also need to decide if you want to push branches individually or all branches at the same time. It will be up to you to make what will work for you best. We found that the following seems to work best:

```
git config --global push.default matching
```

More information about a first time setup is documented at:

```
* http://git-scm.com/book/en/Getting-Started-First-Time-Git-Setup
```

To check your setup you can say:

```
$ git config --list
```

One problem we observed is that students often simply copy and paste instructions, but do not read carefully the error that is reported back and do not fix it. Overlooking the proper set of the push.default is often overlooked. Thus we remind you: **Please read the information on the screen when you set up**.

14.6.14 Upload your public key

Please upload your public key to the repository as documented in github, while going to your account and find it in settings. There you will find a panel SSH key that you can click on which brings you to the window allowing you to add a new key. If you have difficulties with this find a video from the github foundation that explains this.

14.6.15 Working with a directory that will be provided for you

In case your course provided you with a github directory, starting and working in it is going to be real simple. Please wait till an announcement to the class is send before you ask us questions about it.

If you are the only student working on this you still need to make sure that papers or programs you manage in the repository work and do not interfere with scripts that instructors may use to check your assignments. Thus it is god to still create a branch, work in the branch and than merge the branch into the master once you verified things work. After you merged you can push the content to the github repository.

Tip: Please use only **lowercase** characters in the directory names and no special characters such as `@ ; / _` and spaces. In general we recommend that you avoid using directory names with capital letters spaces and _ in them. This will simplify your documentation efforts and make the URLs from git more readable. Also while on some OS's the directories MyDirectory is different from mydirectory on macOS it is considered the same and thus renaming from capital to lower case can not be done without first renaming it to another directory.

Your homework for submission should be organized according to folders in your clone repository. To submit a particular assignment, you must first add it using:

```
git add <name of the file you are adding>
```

Afterwards, commit it using:

```
git commit -m "message describing your submission"
```

Then push it to your remote repository using:

```
git push
```

If you want to modify your submission, you only need to:

```
git commit -m "message relating to updated file"
```

afterwards:

```
git push
```

If you lose any documents locally, you can retrieve them from your remote repository using:

```
git pull
```

14.6.16 README.yml and notebook.md

In case you take classes e516 and e616 with us you will have to create a README.yaml and notebook.md file in the top most directory of your repository. It serves the purpose of identifying your submission for homework and information about yourself.

It is important to follow the format precisely. As it is yaml it is an easy homework to write a 4 line python script that validates if the README.yaml file is valid. In addition you can use programs such as `yamllint` which is documented at

- https://yamllint.readthedocs.io/en/latest/

This file is used to integrate your assignments into a proceedings. An example is provided at

- https://github.com/cloudmesh-community/hid-sample/blob/master/README.yml

Any derivation from this format will not allow us to see your homework as our automated scripts will use the README.yml to detect them. Make sure the file does not contain ay TABs. Please also mind that all filenames of all homework and the main directory must be **lowercase** and do not include spaces. This will simplify your task of managing the files across different operating systems.

In case you work in a team, on a submission, the document will only be submitted in the author and hid that is listed first. All other readme files, will have for that particular artifact a `duplicate: yes` entry to indicate that this submission is managed elsewhere. The team will be responsible to manage their own pull requests, but if the team desires we can grant access for all members to a repository by a user. Please be aware that you must make sure you coordinate with your team.

We will not accept submission of homework as pdf documents or tar files. All assignments must be submitted as code and the reports in native latex and in github. We have a script that will automatically create the PDF and include it in a proceedings. There is no exception from this rule and all reports not compilable will be returned without review and if not submitted within the deadline receive a penalty.

Please check with your instructor on the format of the README.yaml file as it could be different for your class.

To see an example for the notebook.md file, you can visit our sample hid, and browse to the notebook.md file. Alternatively you can visit the following link

- https://github.com/cloudmesh-community/hid-sample/blob/master/notebook.md

The purpose of the notebook md file is to record what you did in the class to us. We will use this file at the end of the class to make sure you have recorded on a weekly basis what you did for the class. Inactivity is a valid response. Not updating the notebook, is not.

The sample directory contains other useful directories and samples, that you may want to investigate in more detail. One of the most important samples is the github issues (see Section 1.19). There is even a video in that section about this and showcases you how to organize your tasks within this class, while copying the assignments from piazza into one or more github issues. As we are about cloud computing, using the services offered by a prominent cloud computing service such as github is part of the learning experience of this course.

14.6.17 Contributing to the Document

It is relatively easy to contribute to the document if you understand how to use github. The first thing you will need to do is to create a fork of the repository. The easiest way to do this is to visit the URL

- https://github.com/cloudmesh-community/book

Towards the upper right corner you will find a link called **Fork**. Click on it and chose into which account you like to fork the original repository. Next you will create a colne from your corked directory. You will see in your fork a green clone button. You will see a URL that you can copy into your terminal. If the links does not include your username, it is the wrong link.

In your terminal you now say

```
git colne  https://github.com/<yourusername>/book
```

Now cd into this directory and make your changes.

```
$ cd book
```

Use the usual git commands such as `git add`, `git commit`, `git push`

Note you will push into your local directory.

14.6.17.1 Stay up to date with the original repo

Form time to time you will see that others are contributing to the original repo. To stay up to date you want to not only sync from your local copy, but also from the original repo. To link your repo with what is called the upstream you need to do the following once, so you can issue `git pull` tha also pulls from the upstream

Make sure you have upstream repo defined:

```
$ git remote add upstream \
      https://github.com/cloudmesh-community/book
```

Now Get latest from upstream:

```
$ git rebase upstream/master
```

In this step, the conflicting file shows up (in my case it was refs.bib):

```
$ git status
```

should show the name of the conflicting file:

```
$ git diff <file name>
```

should show the actual differences. May be in some cases, It is easy to simply take latest version from upstream and reapply your changes.

So you can decide to checkout one version earlier of the specific file. At this stage, the re-base should be complete. So, you need to commit and push the changes to your fork:

```
$ git commit
$ git rebase origin/master
$ git push
```

Then reapply your changes to refs.bib - simply use the backed up version and use the editor to redo the changes.

At this stage, only refs.bib is changed:

```
$ git status
```

should show the changes only in refs.bib. Commit this change using:

```
$ git commit -a -m "new:usr: <message>"
```

And finally push the last committed change:

```
$ git push
```

The changes in the file to resolve merge conflict automatically goes to the original pull request and the pull request can be merged automatically.

You still have to issue the pull request from the Github Web page so it is registered with the upstream repository.

14.6.17.2 Resources

- Pro Git book
- Official tutorial
- Official documentation
- TutorialsPoint on git
- Try git online
- GitHub resources for learning git Note: this is for github and not for gitlab. However as it is for gt the only thing you have to do is replace github, for gitlab.
- Atlassian tutorials for git

In addition the tutorials from atlassian are a good source. However remember that you may not use bitbucket as the repository, so ignore those tutorials. We found the following useful

- What is git: https://www.atlassian.com/git/tutorials/what-is-git
- Installing git: https://www.atlassian.com/git/tutorials/install-git
- git config: https://www.atlassian.com/git/tutorials/setting-up-a-repository#git-config
- git clone: https://www.atlassian.com/git/tutorials/setting-up-a-repository#git-clone
- saving changes: https://www.atlassian.com/git/tutorials/saving-changes
- collaborating with git: https://www.atlassian.com/git/tutorials/syncing

14.6.18 Exercises

E.Github.1:

How do you set your favorite editor as a default with github config

E.Github.2:

What is the differencebetween merge and rebase?

E.Github.3:

Assume you have made a change in your local fork, however other users have since committed to the master branch, how can you make sure your commit works off from the latest information in the master branch?

E.Github.4:

Find a spelling error in the Web page or a contribution and create a pull request for it.

E.Gitlab.5:

Create a README.yml in your github account directory provided for you for class.

14.6.19 Github Issues

Github 8:29 Issues

When we work in teams or even if we work by ourselves, it is prudent to identify a system to coordinate your work. While conduction projects that use a variety of cloud services, it is important to have a system that enables us to have a cloud service that enables us to facilitate this coordination. Github provides such a feature through its issue service that is embedded in each repository.

Issues allow for the coordination of tasks, enhancements, bugs, as well as self defined labeled activities. Issues are shared within your team that has access to your repository. Furthermore, in an open source project the issues are visible to the community, allowing to easily communicate the status, as well as a roadmap to new features.

This enables the community to participate also in reporting of bugs. Using such a system transforms the development of software from the traditional closed shop development toa truly open source development encouraging contributions from others. Furthermore it is also used as bug tracker in which not only you, but the community can communicate bugs to the project.

A good resource for learning more about issues is provided at

- https://guides.github.com/features/issues/

14.6.19.1 Git Issue Features

A git issue has the following features:

title

– a short description of what the issue is about

description

a more detailed description. Descriptions allow also to conveniently add check-boxed todo's.

label

a color enhanced label that can be used to easily categorize the issue. You can define your own labels.

milestone

a milestone so you can identify categorical groups issues as well as their due date. You can for example group all tasks for a week in a milestone, or you could for example put all tasks for a topic such as developing a paper in a milestone and provide a deadline for it.

assignee

an assignee is the person that is responsible for making sure the task is executed or on track if a team works on it. Often projects allow only one assignee, but in certain cases it is useful to assign a group, and the group identifies if the task can be split up and assigns them through check-boxed todo's.

comments

allow anyone with access to provide feedback via comments.

14.6.19.2 Github Markdown

Github uses markdown which we introduce you in Section [S:markdown].

As github has its own flavor of markdown we however also point you to

as a reference. We like to mention the special enhancements fo github's markdown that integrate well to support project management.

14.6.19.2.1 Task lists

Taks lists can be added to any description or comment in github issues To create a task list you can add to any item `[ ]`. This includes a task to be done. To make it as complete simple change it to `[x]`. Whoever the great feature of tasks is that you do not even have to open the editor but you can simply check the task on and of via a mouse click. An example of a task list could be

```
Post Bios

* [x] Post bio on piazza
* [ ] Post bio on google docs
* [ ] Post bio on github
* [ ] \(optional) integrate image in google docs bio
```

In case you need to use a (have at the beginning ot the task text, you need to escape it with a \

14.6.19.2.2 Team integration

A person or team on GitHub can be mentioned by typing the username proceeded by the `@` sign. When posting the text in the issue, it will trigger a notification to them and allow them to react to it. It is even possible to notify entire teams, which are described in more detail at

- https://help.github.com/articles/about-teams/

14.6.19.2.3 Referencing Issues and Pull requests

Each issue has a number. If you use the `#` followed by the issue number you can refer to it in the text which will also automatically include a hyperlink to the task. The same is valid for pull requests.

14.6.19.2.4 Emojis

Although github supports emojis such as `:+1:` we do not use them typically in our class.

14.6.19.3 Notifications

Github allows you to set preferences on how you lik to receive notifications. You can receive them either via e-mail or the Web. This is controlled by configuring it in your settings, where you can set the preferences for participating projects as well as projects you decide to watch. To access the notifications you can simply look at them in the notification screen. In this screen when you press the `?` you will see a number of commands that allow you to control the notification when pressing on one of them.

14.6.19.4 cc

To carbon copy users in your issue text, simply use `/cc` followed by the `@` sign and their github user name.

14.6.19.5 Interacting with issues

Github has the ability to search issues with a search query and a search language that you can find out more about it at

https://guides.github.com/features/issues/#search

A dashboard gives convenient overviews of the issues including a pulse that lists todo's status if you use them in the issue description.

14.6.20 Git Pull Request

14.6.20.1 Introduction

Git pull requests allow developers to submit work or changes they have done to a repository, The developers can then check the changes that have been proposed in the pull request, discuss and make changes if needed. After the content off the pull request has been agreed upon it can be merged to the repository to add the information or changes in the pull request into the repository.

14.6.20.2 How to create a pull request

In this document we will see how we can create a pull request for the Cloudmesh technologies repo that is located at

- https://github.com/cloudmesh/technologies

However if you do pull request on other directories, you just have to replace the url with that of the repository you like to use. A common one four our classes is also

- https://github.com/cloudmesh-community/book

Which contains this book.

You can either create a pull request through a branch or through a fork. In this document we will be looking at how we can create a pull request through a fork.

14.6.20.3 Fork the original repository

First you need to create a fork of the original repository. A fork is your own copy of the repository to which you can make changes to. To fork the Cloudmesh technologies goto Cloudmesh technologies repo and click on the Fork button on the top right corner. Now you can notice that instead of `cloudmesh/technologies` the name of the repo says `YOURGITUSERNAME/technologies`, where `YOURGITUSERANAME` is indeed your github user name. That is because you are now in your own copy of the `cloudmesh/technologies` repository. n our case the user name will be `pulashti`.

14.6.20.4 Clone your copy

Now that you have your fork created, we can go ahead and clone it into our machine. Instructions on how to clone a repository can be found in the Github documentation - Cloning a repository. Make sure that you clone your version of the technologies repo.

14.6.20.5 Adding an upstream

Before we can start working on our copy of the git repo it is good to add an upstream (a link to the original repo) so that we can get all the latest changes in the original repository into our copy. Use the following commands to add an upstream to `cloudmesh/technologies`. First go into the folder which contains your git repo that you cloned and execute the following command.

```
$ git remote add upstream https://github.com/cloudmesh/technologies.git'
```

To make sure you have added it correctly execute the following command

```
$ git remote -v
```

You should see something similar to the following as the output

```
origin  https://github.com/pulasthi/technologies.git (fetch)
origin  https://github.com/pulasthi/technologies.git (push)
upstream    https://github.com/cloudmesh/technologies.git (fetch)
upstream    https://github.com/cloudmesh/technologies.git (push)
```

14.6.20.6 Making changes

Now you can make changes to your repo as with any normal git repository. However to make sure you have the latest copy from the original execute the following command before you start making changes. This will pull the latest changes from the original `cloudmesh/technologies` into your local copy

```
$ git pull upstream master
```

Now make the needed changes commit and push, the changes will be pushed to your copy of the repo i Github, not the `cloudmesh/technologies` repo.

14.6.20.7 Creating a pull request

Once we have changes pushed, you can go into your repository in Github to create a pull request. As seen in +@#fig:button-pullrequest, you have an button named `Pull request`

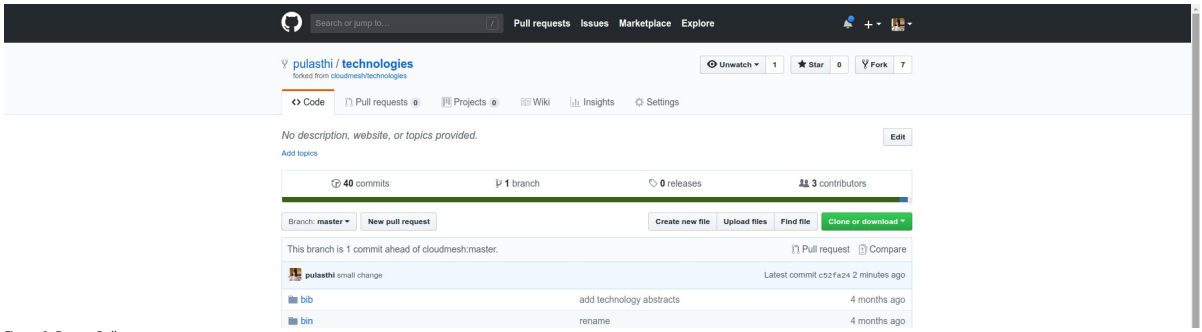


Figure 2: Button Pull request

Once you click on that button you will be taken to a page to create the pull request, which will look similar to fig. 3.

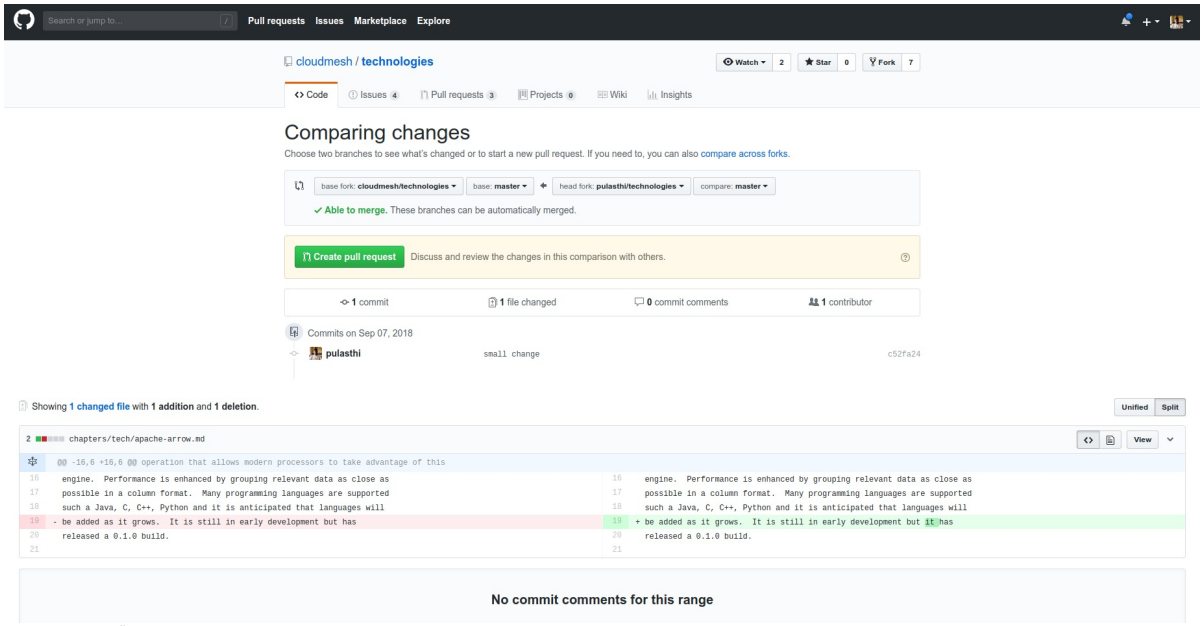


Figure 3: Create a pull request

Once you click on the `Create pull request` button you will be given an option to add a title and a comment for the pull request. Once you complete the details and submit the pull request will appear in the original `cloudmesh/technologies` repo.

Note: Make sure you see the `Able to merge` sign before you submit the pull request, otherwise your pull will not be able to directly merged to the original repo. If you do not see this that means you have not properly done the `git pull upstream master` command before you made the changes

14.7 Linux

Learning Objectives

- Be able to know the basic commands to work in a Linux terminal.
- Get familiar with Linux Commands

Now that you have Linux or a Linux like environment (such as `gitbash`) on your computer it is time to learn a number of useful commands to interact with the system.

In order for this task to enhance your knowledge you are encouraged to find additional material and are required to complete the table of useful Linux commands. You will do this as team and create pull requests improving and completing this documentation. The TAs will provide a mapping between students and commands to be documented. If you find additional commands that aught to be listed here, please add.

14.7.1 History

LINUX is a reimplementation by the community of UNIX which was developed in 1969 by Ken Thompson and Dennis Ritchie of Bell Laboratories and rewritten in C. An important part of UNIX is what is called the kernel which allows the software to talk to the hardware and utilize it.

In 1991 Linus Torvalds started developing a Linux Kernel that was initially targeted for PC's. This made it possible to run it on Laptops and was later on further developed by making it a full Operating system replacement for UNIX.

14.7.2 Shell

One of the most important features for us will be to access the computer with the help of a shell. The shell is typically run in what is called a terminal and allows interaction to the computer with commandline programs.

There are many good tutorials out there that explain why one needs a linux shell and not just a GUI. Randomly we picked the first one that came up with a google query. This is not an endorsement for the material we point to, but could be a worth while read for someone that has no experience in Shell programming:

http://linuxcommand.org/lc3_learning_the_shell.php

Certainly you are welcome to use other resources that may suite you best. We will however summarize in table form a number of useful commands that you may als find even as a RefCard.

http://www.cheat-sheets.org/#Linux

We provide in the next table a number of useful commands that you want to explore. For more information simply type man and the name of the command.

.

Command	Description
man command	manual page for the command
apropos text	list all commands that have text in it
ls	Directory listing
ls -lisa	list details
tree	list the directories in graphical form
cd dirname	Change directory to dirname
mkdir dirname	create the directory
rmdir dirname	delete the directory
pwd	print working directory
rm file	remove the file
cp a b	copy file a to b
mv a b	move/rename file a to b
cat a	print content of filea
cat -n filename	print content of filea with
	line numbers
less a	print paged content of file a
head -5 a	Display first 5 lines of file a
tail -5 a	Display last 5 lines of file a
`du -hs .`	show in human readable form the space used by the current directory
`df -h`	show the details of the disk file system
wc filename	counts the word in a file
sort filename	sorts the file
uniq filename	displays only uniq entries in the file
tar -xvf dir	tars up a compressed version of the directory
rsync	faster, flexible replacement for rcp
gzip filename	compresses the file
gunzip filename	compresses the file
bzip2 filename	compresses the file with
	block-sorting
bunzip2 filename	uncompresses the file with block-sorting
clear	clears the terminal screen
touch filename	change file access and modification times or if file does not exist creates file
who	displays a list of users that are currently logged on, for each user the login name, date and time of login, tty name, and hostname if not local are displayed

whoami	displays the users effective id see also id
echo -n string	write specified arguments to standard output
date	displays or sets date & time, when invoked without arguments the current date and time are displayed
logout	exit a given session
exit	when issued at the shell prompt the shell will exit and terminate any running jobs within the shell
kill	terminate or signal a process by sending a signal to the specified process usually by the pid
ps	displays a header line followed by all processes that have controlling terminals
sleep	suspends execution for an interval of time specified in seconds
uptime	displays how long the system has been running
time command	times the command execution in seconds
find / [-name] file-name.txt	searches a specified path or directory with a given expression that tells the find utility what to find, if used as shown the find utility would search the entire drive for a file named file-name.txt
diff	compares files line by line
hostname	prints the name of the current host system
which	locates a program file in the users path
tail	displays the last part of the file
head	displays the first lines of a file
top	displays a sorted list of system processes
locate filename	finds the path of a file
grep ‘word’ filename	finds all lines with the word in it
grep -v ‘word’ filename	finds all lines without the word in it
chmod ug+rw filename	change file modes or Access Control Lists. In this example user and group are changed to read and write
chown	change file owner and group
history	a build-in command to list the past commands
sudo	execute a command as another user
su	substitute user identity
uname	print the operating system name

set -o emacs	tells the shell to use Emacs commands.
chmod go-rwx file	changes the permission of the file
chown username file	changes the ownership of the file
chgrp group file	changes the group of a file
fgrep text filename	searches the text in the given file
grep -R text .	recursively searches for xyz in all files
find . -name *.py	find all files with .py at the end
ps	list the running processes
kill -9 1234	kill the process with the id 1234
at	que commands for later execution
cron	daemon to execute scheduled commands
crontab	manage the time table for execution commands with cron
mount /dev/cdrom /mnt/cdrom	mount a filesystem from a cd rom to /mnt/cdrom
users	list the logged in users
who	display who is logged in
whoami	print the user id
dmesg	display the system message buffer
last	indicate last logins of users and ttys
uname	print operating system name
date	prints the current date and time
time command	prints the sys, real and user time
shutdown -h “shut down”	shutdown the computer
ping	ping a host
netstat	show network status
hostname	print name of current host system
traceroute	print the route packets take to network host
ifconfig	configure network interface parameters
host	DNS lookup utility
whois	Internet domain name and network number directory service
dig	DNS lookup utility
wget	non-interactive network downloader
curl	transfer a URL
ssh	remote login program
scp	remote file copy program
sftp	secure file transfer program

watch command	run any designated command at regular intervals
awk	program that you can use to select particular records in a file and perform operations on them
sed	stream editor used to perform basic text transformations
xargs	program that can be used to build and execute commands from STDIN
cat some_file.json | python -m json.tool	quick and easy JSON validator

14.7.3 Multi-command execution

One of the important features is that one can execute multiple commands in the shell.

To execute command 2 once command 1 has finished use

```
command1; command2
```

To execute command 2 as soon as command 1 forwards output to stdout use

```
command1; command2
```

To execute command 1 in the background use

```
command1 &
```

14.7.4 Keyboard Shortcuts

These shortcuts will come in handy. Note that many overlap with emacs short cuts.

.

Keys	Description
Up Arrow	Show the previous command
Ctrl + z	Stops the current command
	Resume with `fg` in the foreground
	Resume with `bg` in the background
Ctrl + c	Halts the current command
Ctrl + l	Clear the screen
Ctrl + a	Return to the start of the line
Ctrl + e	Go to the end of the line
Ctrl + k	Cut everything after the cursor to a special clipboard
Ctrl + y	Paste from the special clipboard
Ctrl + d	Logout of current session, similar to exit

14.7.5 bashrc and bash_profile

Usage of a particular command and all the attributes associated with it, use `man` command. Avoid using `rm -r` command to delete files recursively. A good way to avoid accidental deletion is to include the following in your `.bash_profile` file:

```
alias e=open_emacs
alias rm='rm -i'
alias mv='mv -i'
alias h='history'
```

More Information

https://cloudmesh.github.io/classes/lesson/linux/refcards.html

14.7.6 Makefile

Makefiles allow developers to coordinate the execution of code compilations. This not only includes C or C++ code, but any translation from source to a final format. For us this could include the creation of PDF files from latex sources, creation of docker images, and the creation of cloud services and their deployment through simple workflows represented in makefiles, or the coordination of execution targets.

As makefiles include a simple syntax allowing structural dependencies they can easily adapted to fulfill simple activities to be executed in repeated fashion by developers.

An example of how to use Makefiles for docker is provided at http://jmkhael.io/makefiles-for-your-dockerfiles/.

An example on how to use Makefiles for LaTeX is provided at https://github.com/cloudmesh/book/blob/master/Makefile.

Makefiles include a number of rules that are defined by a target name. Let us define a target called hello that prints out the string "Hello World".

```
    hello:
```

```
	@echo "Hello World"
```

Important to remember is that the commands after a target are not indented just by spaces, but actually by a single TAB character. Editors such as emacs will be ideal to edit such Makefiles, while allowing syntax highlighting and easy manipulation of TABs. Naturally other editors will do that also. Please chose your editor of choice. One of the best features of targets is that they can depend on other targets. Thus, iw we define

```
hallo: hello
    @echo "Hallo World"
```

our makefile will first execute hello and than all commands in hallo. As you can see this can be very useful for defining simple dependencies.

In addition we can define variables in a makefile such as

```
HELLO="Hello World"

hello:
    @echo $(HELLO)
```

and can use them in our text with $ invocations.

Moreover, in sophisticated Makefiles, we could even make the targets dependent on files and a target rules could be defined that only compiles those files that have changed since our last invocation of the Makefile, saving potentially a lot of time. However, for our work here we just use the most elementary makefiles.

For more information we recommend you to find out about it on the internet. A convenient reference card sis available at http://www.cs.jhu.edu/~joanne/unixRC.pdf.

14.7.6.1 Makefiles on Windows

Makefiles can easily be accessed also on windows while installing gitbash. Please reed to the internet or search in this handbook for more information about gitbash.

14.7.7 Exercises

E.Linux.1

Familiarize yourself with the commands

E.Linux.2

Find more commands that you find useful and add them to this page.

E.Linux.3

Use the sort command to sort all lines of a file while removing duplicates.

E.Linux.4

Should there be other commands listed in the table with the Linux commands If so which? Create a pull request for them.

E.Linux.5

Write a section explaining chmod. Use letters not numbers

E.Linux.6

Write a section explaining chown. Use letters not numbers

E.Linux.7

Write a section explaining su and sudo

E.Linux.8

Write a section explaining cron, at, and crontab

14.8 Secure Shell

🎓 **Learning Objectives**

- This is one of the most important sections of the book, studdy it carefully.
- learn how to use SSH keys
- Learn how to use ssh-add and ssh-keycahin so you only have to type in your password once
- Understand that each computer needs its own ssh key

Secure Shell is a network protocol allowing users to securely connect to remote resources over the internet. In many services we need to use SSH to assure that we protect he messages send between the communicating entities. Secure Shell is based on public key technology requiring to generate a public-private key pair on the computer. The public key will than be uploaded to the remote machine and when a connection is established during authentication the public private key pair is tested. If they match authentication is granted. As many users may have to share a computer it is possible to add a list of public keys so that a number of computers can connect to a server that hosts such a list. This mechanism builds the basis for networked computers.

In this section we will introduce you to some of the commands to utilize secure shell. We will reuse this technology in other sections to for example create a network of workstations to which we can log in from your laptop. For more information please also consult with the SSH Manual.

⚠ Whatever others tell you, the private key should never be copied to another machine. You almost always want to have a passphrase protecting your key.

14.8.1 ssh-keygen

The first thing you will need to do is to create a public private key pair. Before you do this check whether there are already keys on the computer you are using:

```
ls ~/.ssh
```

If there are files named id_rsa.pub or id_dsa.pub, then the keys are set up already, and we can skip the generating keys step. However you must know the passphrase of the key. If you forgot it you will need to recreate the key. However you will lose any ability to connect with the old key to the resources to which you uploaded the public key. So be careful.

To generate a key pair use the command ssh-keygen. This program is commonly available on most UNIX systems and most recently even Windows 10.

To generate the key, please type:

```
$ ssh-keygen -t rsa -C <comment>
```

The comment will remind you where the key has been created, you could for example use the hostname on which you created the key.

In the following text we will use localname to indicate the username on your computer on which you execute the command.

The command requires the interaction of the user. The first question is:

```
Enter file in which to save the key (/home/localname/.ssh/id_rsa):
```

We recommend using the default location ~/.ssh/ and the default name id_rsa. To do so, just press the enter key.

The second and third question is to protect your ssh key with a passphrase. This passphrase will protect your key because you need to type it when you want to use it. Thus, you can either type a passphrase or press enter to leave it without passphrase. To avoid security problems, you **MUST** chose a passphrase.

It will ask you for the location and name of the new key. It will also ask you for a passphrase, which you **MUST** provide. Please use a strong passphrase to protect it appropriately. Some may advise you (including teachers and TA's) to not use passphrases. This is **WRONG** as it allows someone that gains access to your computer to also gain access to all resources that have the public key. Only for some system related services you may create passwordless keys, but such systems need to be properly protected.

⚠ Not using passphrases poses a security risk!

Make sure to not just type return for an empty passphrase:

```
Enter passphrase (empty for no passphrase):
```

and:

```
Enter same passphrase again:
```

If executed correctly, you will see some output similar to:

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/localname/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/localname/.ssh/id_rsa.
Your public key has been saved in /home/localname/.ssh/id_rsa.pub.
The key fingerprint is:
34:87:67:ea:c2:49:ee:c2:81:d2:10:84:b1:3e:05:59 localname@indiana.edu

+--[ RSA 2048]----+
|.+...Eo= .       |
| ..=.o + o +o    |
|O.  = ......     |
| = .   . .       |
+-----------------+
```

Once, you have generated your key, you should have them in the `.ssh` directory. You can check it by:

```
$ cat ~/.ssh/id_rsa.pub
```

If everything is normal, you will see something like:

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCXJH2iG2FMHqC6T/U7uB8kt
6KlRh4kUOjgw9sc4Uu+Uwe/kshuispauhfsjhfm,anf6787sjgdkjsgl+EwD0
thkoamyi0VvhTVZhj61pTdhyl1t8hlkoLi9JVnVBPP5kIN3wVyNAJjYBrAUNW
4dXKXtmfkXp98T3OW4mxAtTH434MaT+QcPTcxims/hwsUeDAVKZY7UgZhEbiE
xxkejtnRBHTipi0W03W05TOUGRW7EuKf/4ftNVPilCO4DpfY44NFG1xPwHeim
Uk+t9h48pBQj16FrUCp0rS02Pj+4/9dNeS1kmNJu5ZYS8HVRhvuoTXuAY/UVc
ynEPUegkp+qYnR user@myemail.edu
```

The directory `~/.ssh` will also contain the private key `id_rsa` which you must not share or copy to another computer.

⚠ Never, copy your private key to another machine or check it into a repository!

To see what is in the .ssh directory, please use

```
$ ls ~/.ssh
```

Typically you will se a list of files such as

```
authorized_keys
id_rsa
id_rsa.pub
known_hosts
```

In case you need to change your change passphrase, you can simply run `ssh-keygen -p` command. Then specify the location of your current key, and input (old and) new passphrases. There is no need to re-generate keys:

```
ssh-keygen -p
```

You will see the following output once you have completed that step:

```
Enter file in which the key is (/home/localname/.ssh/id_rsa):
Enter old passphrase:
Key has comment '/home/localname/.ssh/id_rsa'
Enter new passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved with the new passphrase.
```

14.8.2 ssh-add

Often you wil find wrong information about passphrases on the internet and people recommending you not to use one. However it is in almost all cases better to create a key pair and use `ssh-add` to add the key to the current session so it can be used in behalf of you. This is accomplished with an agent.

The `ssh-add` command adds SSH private keys into the SSH authentication agent for implementing single sign-on with SSH. ssh-add allows the user to use any number of servers that are spread across any number of organizations, without having to type in a password every time when connecting between servers. This is commonly used by system administrators to login to multiple server.

`ssh-add` can be run without arguments. When run without arguments, it adds the following default files if they do exist:

- `~/.ssh/identity` - Contains the protocol version 1 RSA authentication identity of the user.
- `~/.ssh/id_rsa` - Contains the protocol version 1 RSA authentication identity of the user.
- `~/.ssh/id_dsa` - Contains the protocol version 2 DSA authentication identity of the user.
- `~/.ssh/id_ecdsa` - Contains the protocol version 2 ECDSA authentication identity of the user.

To add a key you can provide the path of the key file as an argument to ssh-add. For example,

```
ssh-add ~/.ssh/id_rsa
```

would add the file `~/.ssh/id_rsa`

If the key being added has a passphrase, `ssh-add` will run the `ssh-askpass` program to obtain the passphrase from the user. If the `SSH_ASKPASS` environment variable is set, the program given by that environment variable is used instead.

Some people use the `SSH_ASKPASS` environment variable in scripts to provide a passphrase for a key. The passphrase might then be hard-coded into the script, or the script might fetch it from a password vault.

The command line options of `ssh-add` are as follows:

Option	Description
`-c`	Causes a confirmation to be requested from the user every time the added identities are used for authentication. The confirmation is requested using ssh-askpass.
`-D`	Deletes all identities from the agent.
`-d`	Deletes the given identities from the agent. The private key files for the identities to be deleted should be listed on the command line.
`-e` pkcs11	Remove key provided by pkcs11

`-L`	Lists public key parameters of all identities currently represented by the agent.
`-l`	Lists fingerprints of all identities currently represented by the agent.
`-s` pkcs11	Add key provided by pkcs11.
`-t` life	Sets the maximum time the agent will keep the given key. After the timeout expires, the key will be automatically removed from the agent. The default value is in seconds, but can be suffixed for m for minutes, h for hours, d for days, or w for weeks.
`-X`	Unlocks the agent. This asks for a password to unlock.
`-x`	Locks the agent. This asks for a password; the password is required for unlocking the agent. When the agent is locked, it cannot be used for authentication.

14.8.3 SSH Add and Agent

To not always type in your password, you can use `ssh-add` as previously discussed

It prompts the user for a private key passphrase and add it to a list of keys managed by the ssh-agent. Once it is in this list, you will not be asked for the passphrase as long as the agent is running.with your public key. To use the key across terminal shells you can start an ssh agent.

To start the agent please use the following command:

```
eval `ssh-agent`
```

or use

```
eval "$(ssh-agent -s)"
```

It is important that you use the backquote, located under the tilde (US keyboard), rather than the single quote. Once the agent is started it will print a PID that you can use to interact with later

To add the key use the command

```
ssh-add
```

To remove the agent use the command

```
kill $SSH_AGENT_PID
```

To execute the command upon logout, place it in your `.bash_logout` (assuming you use bash).

On OSX you can also add the key permanently to the keychain if you do toe following:

```
ssh-add -K ~/.ssh/id_rsa
```

Modify the file `.ssh/config` and add the following lines:

```
Host *
  UseKeychain yes
  AddKeysToAgent yes
  IdentityFile ~/.ssh/id_rsa
```

14.8.3.1 Using SSH on Mac OS X

Mac OS X comes with an ssh client. In order to use it you need to open the `Terminal.app` application. Go to `Finder`, then click `Go` in the menu bar at the top of the screen. Now click `Utilities` and then open the `Terminal` application.

14.8.3.2 Using SSH on Linux

All Linux versions come with ssh and can be used right from the terminal.

14.8.3.3 Using SSH on Raspberry Pi 3

SSH is available on Raspbian. However, to ssh into the PI you have to activate it via the configuration menu. For a more automated configuration, we will provide more information in the Raspberry PI section.

14.8.3.4 SSH on Windows

⚠ This section is outdated and should be replaced with information from SSH in powershell and the new ubuntu running in windows

- https://www.howtogeek.com/336775/how-to-enable-and-use-windows-10s-built-in-ssh-commands/

In case you need access to ssh Microsoft has fortunately updated their software to be able to run it directly from the Windows commandline including PowerShell.

However it is as far as we know not activated by default so you need to follow some setup scripts. Also this software is considered beta and its development and issues can be found at

https://github.com/PowerShell/Win32-OpenSSH https://github.com/PowerShell/Win32-OpenSSH/issues What you have to do is to install it by going to

```
Settings > Apps
```

and click

```
Manage optional features
```

under

```
Apps & features
```

Next, Click on the `Add feature`. You will be presented with a list in which you scroll down, till you find `OpenSSH Client (Beta)`. Click on it and invoke `Install`.

After the install has completed, you can use the `ssh` command. Just type it in the commandshell or PowerShell

```
PS C:\Users\gregor> ssh
```

Naturally you can now use it just as on Linux or OSX. and use it to login to other resources

```
PS C:\Users\gregor> ssh myname@example.com
```

14.8.4 SSH and putty

We no longer recommend the use of putty and instead you should be using SSH over Powershell for this class.

14.8.4.1 Access a Remote Machine

Once the key pair is generated, you can use it to access a remote machine. To dod so the public key needs to be added to the `authorized_keys` file on the remote machine.

The easiest way to do tis is to use the command `ssh-copy-id`.

```
$ ssh-copy-id user@host
```

Note that the first time you will have to authenticate with your password.

Alternatively, if the ssh-copy-id is not available on your system, you can copy the file manually over SSH:

```
$ cat ~/.ssh/id_rsa.pub | ssh user@host 'cat >> .ssh/authorized_keys'
```

Now try:

```
$ ssh user@host
```

and you will not be prompted for a password. However, if you set a passphrase when creating your SSH key, you will be asked to enter the passphrase at that time (and whenever else you log in in the future). To avoid typing in the password all the time we use the ssh-add command that we described earlier.

```
$ ssh-add
```

14.8.5 SSH Port Forwarding ⭕ ❓

⚠ this section has not been vetted yet

TODO: Add images to illustrate the concepts

SSH Port forwarding (SSH tunneling) creates an encrypted secure connection between a local computer and a remote computer through which services can be relayed. Because the connection is encrypted, SSH tunneling is useful for transmitting information that uses an unencrypted protocol.

14.8.5.1 Prerequisites

- Before you begin, you need to check if forwarding is allowed on the SSH server you will connect to.
- You also need to have a SSH client on the computer you are working on.

If you are using the OpenSSH server:

```
$ vi /etc/ssh/sshd_config
```

and look and change the following:

```
AllowTcpForwarding = Yes
GatewayPorts = Yes
```

Set the `GatewaysPorts` variable only if you are going to use remote port forwarding (discussed later in this tutorial). Then, you need to restart the server for the change to take effect.

14.8.5.2 How to Restart the Server

If you are on:

- Linux, depending upon the init system used by your distribution, run:

  ```
  $ sudo systemctl restart sshd
  $ sudo service sshd restart
  ```

  Note that depending on your distribution, you may have to change the service to ssh instead of sshd.

- Mac, you can restart the server using:

  ```
  $ sudo launchctl unload /System/Library/LaunchDaemons/ssh.plist
  $ sudo launchctl load -w /System/Library/LaunchDaemons/ssh.plist
  ```

- Windows and want to set up a SSH server, have a look at MSYS2 or Cygwin.

14.8.5.3 Types of Port Forwarding

There are three types of SSH Port forwarding:

14.8.5.4 Local Port Forwarding

Local port forwarding lets you connect from your local computer to another server. It allows you to forward traffic on a port of your local computer to the SSH server, which is forwarded to a destination server. To use local port forwarding, you need to know your destination server, and two port numbers.

Example 1:

```
$ ssh -L 8080:www.cloudcomputing.org:80 <host>
```

Where `<host>` should be replaced by the name of your laptop. The -L option specifies local port forwarding. For the duration of the SSH session, pointing your browser at `http://localhost:8080/` would send you to `http://cloudcomputing.com`

Example 2:

This example opens a connection to the www.cloudcomputing.com jump server, and forwards any connection to port 80 on the local machine to port 80 on `intra.example.com`.

```
$ ssh -L 80:intra.example.com:80 www.cloudcomputing.com
```

Example 3:

By default, anyone (even on different machines) can connect to the specified port on the SSH client machine. However, this can be restricted to programs on the same host by supplying a bind address:

```
$ ssh -L 127.0.0.1:80:intra.example.com:80 www.cloudcomputing.com
```

Example 4:

```
$ ssh -L 8080:www.Cloudcomputing.com:80 -L 12345:cloud.com:80 <host>
```

This would forward two connections, one to `www.cloudcomputing.com`, the other to `www.cloud.com`. Pointing your browser at `http://localhost:8080/` would download pages from www.cloudcomputing.com, and pointing your browser to `http://localhost:12345/` would download pages from www.cloud.com.

Example 5:

The destination server can even be the same as the SSH server.

```
$ ssh -L 5900:localhost:5900 <host>
```

The LocalForward option in the OpenSSH client configuration file can be used to configure forwarding without having to specify it on command line.

14.8.5.5 Remote Port Forwarding

Remote port forwarding is the exact opposite of local port forwarding. It forwards traffic coming to a port on your server to your local computer, and then it is sent to a destination. The first argument should be the remote port where traffic will be directed on the remote system. The second argument should be the address and port to point the traffic to when it arrives on the local system.

```
$ ssh -R 9000:localhost:3000 user@clodcomputing.com
```

SSH does not by default allow remote hosts to forwarded ports. To enable remote forwarding add the following to: `/etc/ssh/sshd_config`

```
GatewayPorts yes
```

```
$ sudo vim /etc/ssh/sshd_config
```

and restart SSH

```
$ sudo service ssh restart
```

After above steps you should be able to connect to the server remotely, even from your local machine. `ssh -R` first creates an SSH tunnel that forwards traffic from the server on port 9000 to your local machine on port 3000.

14.8.5.6 Dynamic Port Forwarding

Dynamic port forwarding turns your SSH client into a SOCKS proxy server. SOCKS is a little-known but widely-implemented protocol for programs to request any Internet connection through a proxy server. Each program that uses the proxy server needs to be configured specifically, and reconfigured when you stop using the proxy server.

```
$ ssh -D 5000 user@clodcomputing.com
```

The SSH client creates a SOCKS proxy at port 5000 on your local computer. Any traffic sent to this port is sent to its destination through the SSH server.

Next, you'll need to configure your applications to use this server. The Settings section of most web browsers allow you to use a SOCKS proxy.

14.8.5.7 ssh config

Defaults and other configurations can be added to a configuration file that is placed in the system. The ssh program on a host receives its configuration from

- the command line options
- a user-specific configuration file: `~/.ssh/config`
- a system-wide configuration file: `/etc/ssh/ssh_config`

Next we provide an example on how to use a config file

14.8.5.8 Tips

Use SSH keys

- You will need to use ssh keys to access remote machines

No blank passphrases

- In most cases you must use a passphrase with your key. In fact if we find that you use passwordless keys to futuresystems and to chameleon cloud resources, we may elect to give you an F for the assignment in question. There are some exceptions, but they will be clearly communicated to you in class. You will as part of your cloud drivers license test explain how you gain access to futuresystems and chameleon to explicitly explain this point and provide us with reasons what you can not do.

A key for each server

- Under no circumstances copy the same private key on multiple servers. This violates security best practices. Create for each server a new private key and use their public keys to gain access to the appropriate server.

Use SSH agent

- So as to not to type in all the time the passphrase for a key, we recommend using ssh-agent to manage the login. This will be part of your cloud drivers license.

  But shut down the ssh-agent if not in use

keep an offline backup, put encrypt the drive

- You may for some of our projects need to make backups of private keys on other servers you set up. If you like to make a backup you can do so on a USB stick, but make sure that access to the stick is encrypted. Do not store anything else on that key and look it in a safe place. If you lose the stick, recreate all keys on all machines.

14.8.5.9 References

- The Secure Shell: The Definitive Guide, 2 Ed (O'Reilly and Associates)

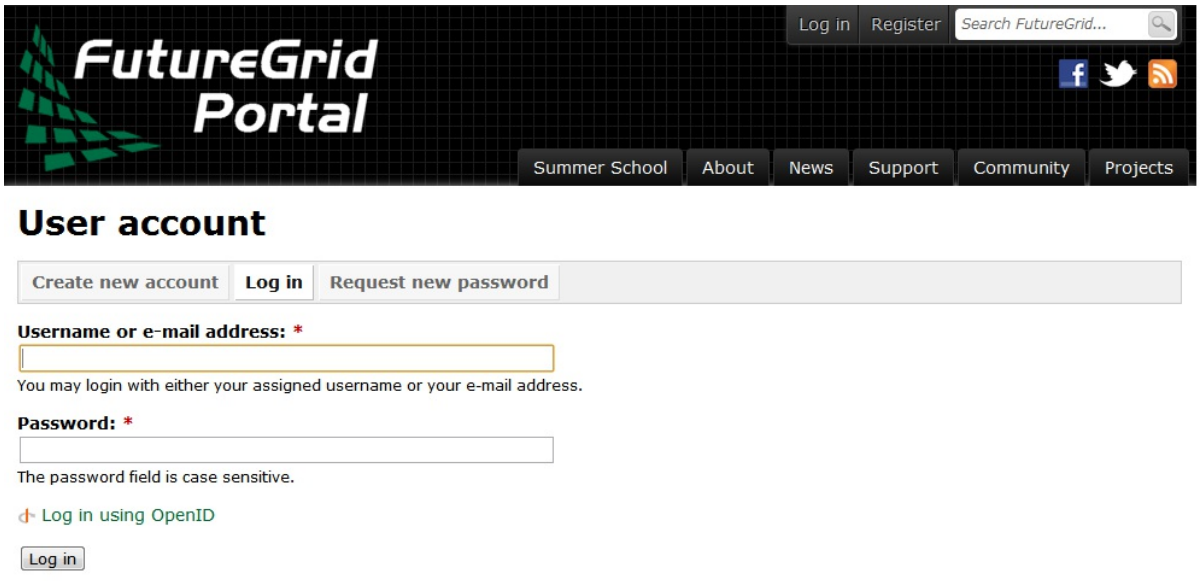
14.8.6 SSH to FutureSystems Resources

Learning Objectives

- Obtain a Future system account so you can use kubernetes or dockerswarm or other services offered by FutureSystems.
- Note that we no longer support OpenStack in FutureSystems.

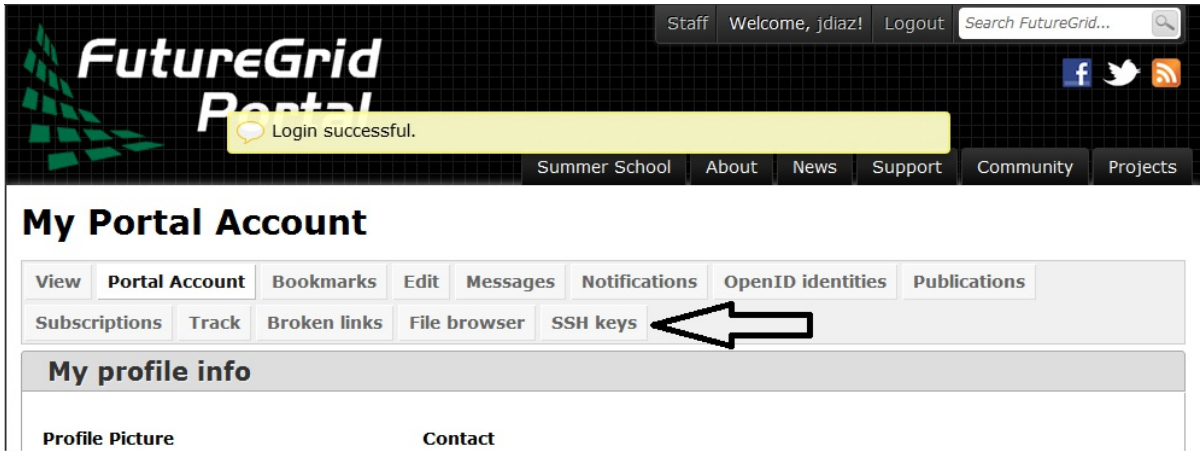
Next, you need to upload the key to the portal. You must be logged into the portal to do so.

Step 1: Log into the portal



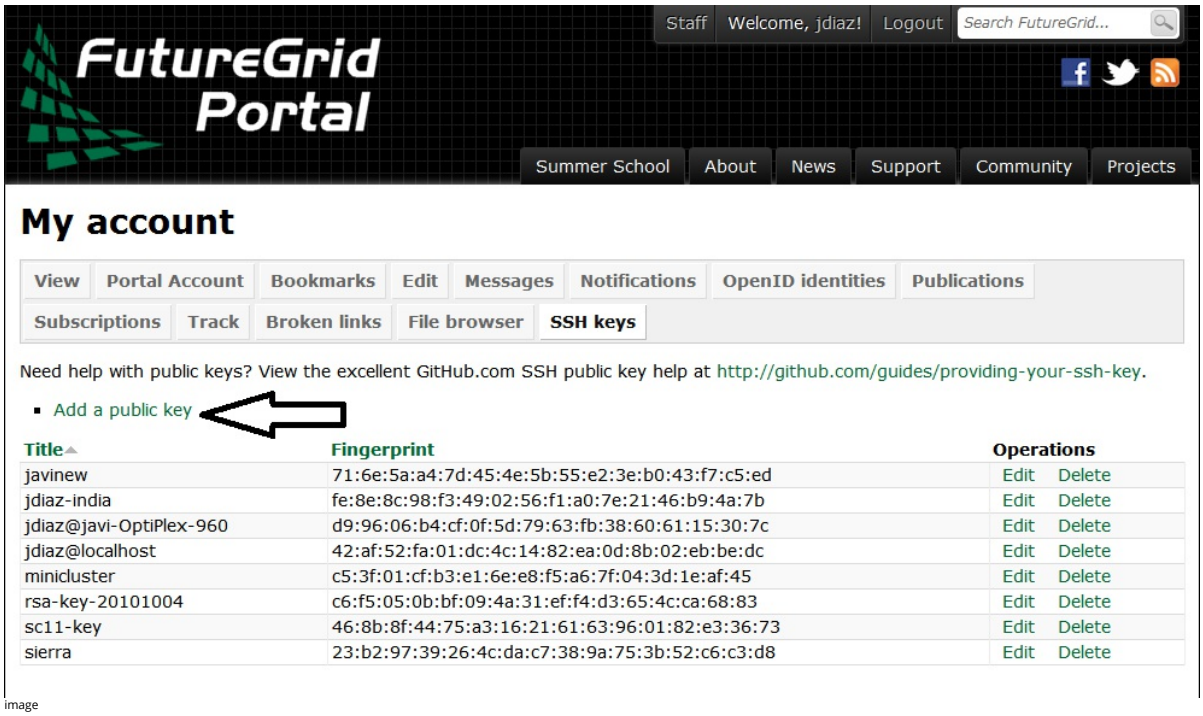
image

Step 2: Click in the "ssh key" button or go directly to https://portal.futuresystems.org/my/ssh-keys



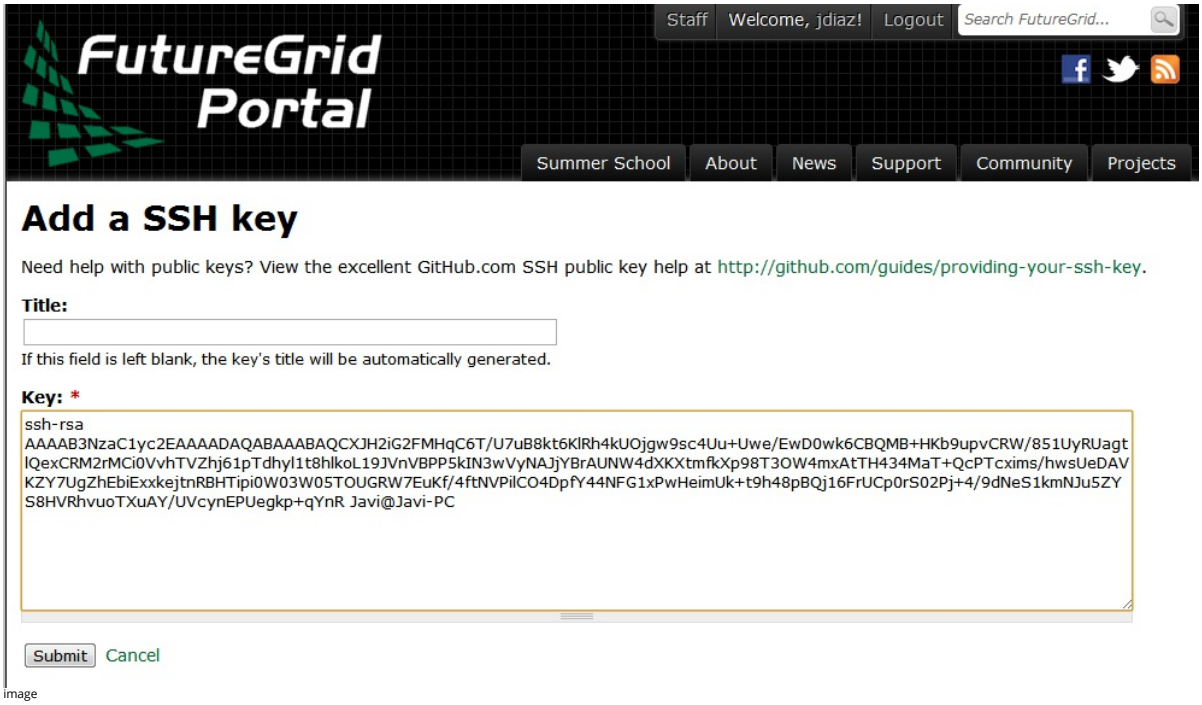
image

Step 3: Click in the "add a public key" link.



image

Step 4: Paste your ssh key into the box marked Key. Use a text editor to open the `id_rsa.pub`. Copy the entire contents of this file into the ssh key field as part of your profile information. Many errors are introduced by users in this step as they do not paste and copy correctly.



image

Step 5: Click the submit button. **IMPORTANT**: Leave the Title field blank. Make sure that when you paste your key, it does not contain newlines or carriage returns that may have been introduced by incorrect pasting and copying. If so, please remove them.

At this point, you have uploaded your key. However, you will still need to wait till all accounts have been set up to use the key, or if you did not have an account till it has been created by an administrator. Please, check your email for further updates. You can also refresh this page and see if the boxes in your account status information are all green. Then you can continue.

14.8.6.1 Testing your FutureSystems ssh key

If you have had no FutureSystem account before, you need to wait for up to two business days so we can verify your identity and create the account. So please wait. Otherwise, testing your new key is almost instantaneous on india. For other clusters like it can take around 30 minutes to update the ssh keys.

To log into india simply type the usual ssh command such as:

```
$ ssh portalname@india.futuresystems.org
```

The first time you ssh into a machine you will see a message like this:

```
The authenticity of host 'india.futuresystems.org (192.165.148.5)' cannot be established.
RSA key fingerprint is 11:96:de:b7:21:eb:64:92:ab:de:e0:79:f3:fb:86:dd.
Are you sure you want to continue connecting (yes/no)? yes
```

You have to type yes and press enter. Then you will be logging into india. Other FutureSystem machines can be reached in the same fashion. Just replace the name india, with the appropriate FutureSystems resource name.

14.8.7 Exercises

E.SSH.1:

Create an SSH key pair

E.SSH.2:

Upload the public key to git repository you use. Create a fork in git and use your ssh key to clone and commit to it section/ssh.tex

E.SSH.3:

The images in the futuresystems ssh section are a bit outdated. Please update them. Make sure to blend out your username and fingerprints in the images. or invent some ...

E.SSH.4

Get an account on futuresystems.org (if you are authorized to do so). Upload your key to https://futuresystems.org. Login to india.futuresystems.org. Note that this could take some time as administrators need to approve you. Be patient.

15 Python

15.1 Introduction to Python

Learning Objectives

- Learn quickly Python under the assumption you know a programming language
- Work with modules
- Understand docopts and cmd
- Contuct some python examples to refresh your python knpwledge
- Learn about the `map` function in Python
- Learn how to start subprocesses and rederect their output
- Learn more advanced constructs such as multiprocessing and Queues
- Understand why we do not use `anaconda`
- Get familiar with `pyenv`

Portions of this lesson have been adapted from the official Python Tutorial copyright Python Software Foundation.

Python is an easy to learn programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's simple syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, https://www.python.org/, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation. The Python interpreter can be extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

Python is an interpreted, dynamic, high-level programming language suitable for a wide range of applications.

The philosophy of python is summarized in The Zen of Python as follows:

- Explicit is better than implicit
- Simple is better than complex
- Complex is better than complicated
- Readability counts

The main features of Python are:

- Use of indentation whitespace to indicate blocks
- Object orient paradigm
- Dynamic typing
- Interpreted runtime
- Garbage collected memory management
- a large standard library
- a large repository of third-party libraries

Python is used by many companies (such as Google, Yahoo!, CERN, NASA) and is applied for web development, scientific computing, embedded applications, artificial intelligence, software development, and information security, to name a few.

The material collected here introduces the reader to the basic concepts and features of the Python language and system. After you have worked through the material you will be able to:

- use Python
- use the interactive Python interface
- understand the basic syntax of Python
- write and run Python programs stored in a file
- have an overview of the standard library
- install Python libraries using pyenv or if it is not available virtualenv

This section does not attempt to be comprehensive and cover every single feature, or even every commonly used feature. Instead, it introduces many of Python's most noteworthy features, and will give you a good idea of the language's flavor and style. After reading it, you will be able to read and write Python modules and programs, and you will be ready to learn more about the various Python library modules.

In order to conduct this lesson you need

- A computer with Python 2.7.15 or 3.7.0
- Familiarity with command line usage
- A text editor such as PyCharm, emacs, vi or others. You should identity which works best for you and set it up.

15.1.1 References

Some important additional information can be found on the following Web pages.

- Python
- Pip
- Virtualenv
- NumPy
- SciPy
- Matplotlib
- Pandas
- pyenv
- PyCharm

Python module of the week is a Web site that provides a number of short examples on how to use some elementary python modules. Not all modules are equally useful and you should decide if there are better alternatives. However for beginners this site provides a number of good examples

- Python 2: https://pymotw.com/2/
- Python 3: https://pymotw.com/3/

15.2 Python Installation

Python is easy to install and very good instructions for most platforms can be found on the python.org Web page. We will be using Python 2.7.15 and/or Python 3.7 in our activities.

To manage python modules, it is useful to have pip package installation tool on your system.

We assume that you have a computer with python installed. The version of python however may not be the newest version. Please check with

```
$ python --version
```

which version of python you run. If it is not the newest version, we recommend that you install pyenv to install a newer version so you do not effect the default version of python from your system.

While in other classes yo may have been taught to use anaconda, this is not a tool that ought to be used in a cloud class. The reason for this is that it installs many packages that you are likely not to use. In fact installing anaconda on your VM will waste space and time and you should look into other installs.

However, real cloud engineers with the most flexibility in python versions want to install python via pyenv.

Note: whenever possible please use for the newest version of Python 2 or 3. In order not to effect your OS we will use pyenv.

15.2.1 Managing custom Python installs

Often you have your own computer and you do not like to change its environment to keep it in pristine condition. Python comes with many libraries that could for example conflict with libraries that you have installed. To avoid this it is bets to work in an isolated python we can use tools such as virtualenv, pyenv or pyenv for 3.7.1. Which you use depends on you, but we highly recommend pyenv if you can.

15.2.1.1 Managing Multiple Python Versions with Pyenv

Python has several versions that are used by the community. This includes Python 2 and Python 3, but all different management of the python libraries. As each OS may have their own version of python installed. It is not recommended that you modify that version. Instead you may want to create a localized python installation that you as a user can modify. To do that we recommend pyenv. Pyenv allows users to switch between multiple versions of Python (https://github.com/yyuu/pyenv). To summarize:

- users to change the global Python version on a per-user basis;
- users to enable support for per-project Python versions;
- easy version changes without complex environment variable management;
- to search installed commands across different python versions;
- integrate with tox (https://tox.readthedocs.io/).

15.2.1.1.1 Installation without pyenv

If you need to have more than one python version installed and do not want or can use pyenv, we recommend you download and install python 2.7.15 and 3.7.1 from python.org (https://www.python.org/downloads/)

15.2.1.1.2 Disabling wrong python installs on macOS

While working with students we have seen at times that they take other classes either at universities or online that teach them how to program in python. Unfortunately, although they seem to do that they often ignore to teach you how to properly install python. I just recently had a students that had installed python 7 times on his macOS machine, while another student had 3 different installations, all of which conflicted with each other as they were not set up properly.

We recommend that you inspect if you have a files such as `~/.bashrc` or `~/.bashrc_profile` in your home directory and identify if it activates various versions of python on your computer. If so you could try to deactivate them while out-commenting the various versions with the # character at the beginning of the line, start a new terminal and see if the terminal shell still works. Than you can follow our instructions here while using an install on pyenv.

15.2.1.1.3 Install pyenv on macOS from git

This is our recommended way to install pyenv on macOS:

```
$ git clone https://github.com/pyenv/pyenv.git ~/.pyenv
$ git clone https://github.com/pyenv/pyenv-virtualenv.git ~/.pyenv/plugins/pyenv-virtualenv
$ git clone https://github.com/yyuu/pyenv-virtualenvwrapper.git ~/.pyenv/plugins/pyenv-virtualenvwrapper
$ echo 'export PYENV_ROOT="$HOME/.pyenv"' >> ~/.bash_profile
$ echo 'export PATH="$PYENV_ROOT/bin:$PATH"' >> ~/.bash_profile
```

15.2.1.1.4 Installation of Homebrew

Before installing anything on your computer make sure you have enough space. Use in the terminal the command:

```
$ df -h
```

which gives your an overview of your file system. If you do not have enough space, please make sure you free up unused files from your drive.

In many occasions it is beneficial to use readline as it provides nice editing features for the terminal and xz for completion. First, make sure you have xcode installed:

```
$ xcode-select --install
```

On Mojave you will get an error that zlib is not installed. THis is due to that the header files are not properly installed. To do this you can say

```
$ sudo installer -pkg /Library/Developer/CommandLineTools/Packages/macOS_SDK_headers_for_macOS_10.14.pkg -target /
```

Next install homebrew, pyenv, pyenv-virtualenv and pyenv-virtualwrapper. Additionally install readline and some compression tools:

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
brew update
brew install readline xz
```

15.2.1.1.5 Install pyenv on macOS with Homebrew

This is the recommended way of installing pyenv on macOS High Sierra. This method should also be considered if you get the following error: “ERROR: The Python ssl extension was not compiled. Missing the OpenSSL lib?”

We describe here a mechanism of installing pyenv with homebrew. Other mechanisms can be found on the pyenv documentation page (https://github.com/yyuu/pyenv-installer). You must have homebrew installed as discussed in the previous section.

To install pyenv with homebrew execute in the terminal:

```
brew install pyenv pyenv-virtualenv pyenv-virtualenvwrapper
```

15.2.1.1.6 Install pyenv on Ubuntu

The following steps will install pyenv in a new ubuntu 18.04 distribution.

Start up a terminal and execute in the terminal the following commands. We recommend that you do it one command at a time so you can observe if the command succeeds:

```
$ sudo apt-get update
$ sudo apt-get install git python-pip make build-essential libssl-dev
$ sudo apt-get install zlib1g-dev libbz2-dev libreadline-dev libsqlite3-dev
$ sudo pip install virtualenvwrapper

$ git clone https://github.com/yyuu/pyenv.git ~/.pyenv
$ git clone https://github.com/pyenv/pyenv-virtualenv.git ~/.pyenv/plugins/pyenv-virtualenv
$ git clone https://github.com/yyuu/pyenv-virtualenvwrapper.git ~/.pyenv/plugins/pyenv-virtualenvwrapper

$ echo 'export PYENV_ROOT="$HOME/.pyenv"' >> ~/.bashrc
$ echo 'export PATH="$PYENV_ROOT/bin:$PATH"' >> ~/.bashrc
```

You can also install pyenv using curl command in following way:

```
curl -L https://raw.githubusercontent.com/yyuu/pyenv-installer/master/bin/pyenv-installer | bash
```

Then install its dependencies:

```
sudo apt-get update && sudo apt-get upgrade
sudo apt-get install -y make build-essential libssl-dev zlib1g-dev libbz2-dev libreadline-dev libsqlite3-dev wget curl llvm libncurses5-dev git
```

Now that you have installed pyenv it is not yet activated in your current terminal. The easiest thing to do is to start a new terminal and typ in:

```
which pyenv
```

If you see a response pyenv is installed and you can proceed with the next steps.

Please remember whenever you modify `.bashrc` or `.bash_profile` you need to start a new terminal.

15.2.1.1.7 Install Different Python Versions

Pyenv provides a large list of different python versions. To see the entire list please use the command:

```
$ pyenv install -l
```

However, for us we only need to worry about python 2.7.15 and python 3.7.1. You can now install different versions of python into your local environment with the following commands:

```
$ pyenv update
$ pyenv install 2.7.15
$ pyenv install 3.7.1
```

You can set the global python default version with:

```
$ pyenv global 3.7.1
```

Type the following to determine which version you activated:

```
$ pyenv version
```

Type the following to determine which versions you have available:

```
$ pyenv versions
```

Associate a specific environment name with a certain python version, use the following commands:

```
$ pyenv virtualenv 2.7.15 ENV2
$ pyenv virtualenv 3.7.1 ENV3
```

In the example, ENV2 would represent python 2.7.15 while ENV3 would represent python 3.7.1. Often it is easier to type the alias rather than the explicit version.

15.2.1.1.8 Set up the Shell

To make all work smoothly from your terminal, you can include the following in your `.bashrc` files:

```
export PYENV_VIRTUALENV_DISABLE_PROMPT=1
eval "$(pyenv init -)"
eval "$(pyenv virtualenv-init -)"

__pyenv_version_ps1() {
  local ret=$?;
  output=$(pyenv version-name)
  if [[ ! -z $output ]]; then
    echo -n "($output)"
  fi
  return $ret;
}

PS1="\$(__pyenv_version_ps1) ${PS1}"
```

We recommend that you do this towards the end of your file.

15.2.1.1.9 Switching Environments

After setting up the different environments, switching between them is now easy. Simply use the following commands:

```
(2.7.15) $ pyenv activate ENV2
(ENV2) $ pyenv activate ENV3
(ENV3) $ pyenv activate ENV2
(ENV2) $ pyenv deactivate ENV2
(2.7.15) $
```

To make it even easier, you can add the following lines to your `.bash_profile` file:

```
alias ENV2="pyenv activate ENV2"
alias ENV3="pyenv activate ENV3"
```

If you start a new terminal, you can switch between the different versions of python simply by typing:

```
$ ENV2
$ ENV3
```

15.2.2 Updating Python Version List

Pyenv maintains locally a list of available python versions. To see the list use the command

```
pyenv update
pyenv install -l
```

You will see the updated list.

15.2.3 Updating to a new version of Python with pyenv

Naturally python itself evolves and new versions will become available via pyenv. To facilitate such a new version you need to first install it into pyenv. Let us assume you had an old version of python installed onto the ENV3 environment. Than you need to execute the following steps:

```
pyenv deactivate
pyenv uninstall ENV3
pyenv install 3.7.1
pyenv virtualenv 3.7.1 ENV3
ENV3
pip install pip -U
```

With the pi install command, we make sure we have the newest version of pip. In case you get an error, you may have to update xcode as follows and try again:

```
  xcode-select --install
```

After you installed it you can activate it by typing `ENV3`. Naturally this requires that you added it to your bash environment as discussed in Section 1.1.1.8.

15.2.4 Pyenv in a docker container

We provide a simple docker container on docker hub that is based on ubuntu 18.04 that has pyenv, python 2.7.15 and python 3.7.1 installed. Using this image is as simple as downloading it and running it.

To run the container and loginto the command prompt please use

```
$ docker run --rm -it cloudmesh/pyenv:1.0  /bin/bash
```

To switch between the python versions use the command

```
container> ENV2
container> ENV3
```

where container indicates that the command is executed ### Creating the container locally

This section is only needed if you like to recreate the image or modify the Dockerfile.

The information about how we create the image is provided at in a repository. You can download the code in the directory and can create the image from the Docker file while using the Makefile as follows:

```
$ mkdir cloudmesh-community
$ cd cloudmesh-community
$ git clone https://github.com/cloudmesh-community/images.git
$ cd images/pyenv
$ make image
```

This will create an image locally. with

```
$ make login
```

you can login to the shell. Typically you will only need the docker command as descripbed in the prvious section.

15.2.5 Installation without pyenv

If you need to have more than one python version installed and do not want or can use pyenv, we recommend you download and install python 2.7.15 and 3.7.1 from python.org (https://www.python.org/downloads/)

15.2.5.1 Make sure pip is up to date

As you will want to install other packages, make sure pip is up to date:

```
pip install pip -U
```

pyenv virtualenv anaconda3-4.3.1 ANA3 pyenv activate ANA3

15.2.6 Anaconda and Miniconda

We do not recommend that you use anaconda or miniconda as it may

> interfere with your default python interpreters and setup.

Please note that beginners to python should always use anaconda or miniconda only after they have installed pyenv and use it. For this class neither anaconda nor miniconda is required. In fact we do not recommend it. We keep this section as we know that other classes at IU may use anaconda. We are not aware if these classes teach you the right way to install it, with pyenv.

15.2.6.1 Miniconda

⚠ This section about miniconda is experimental and has not been tested. We are looking for contributors that help completing it. If you use anaconda or miniconda we recommend to manage it via pyenv.

To install mini conda you can use the following commands:

```
$ mkdir ana
$ cd ana
$ pyenv install miniconda3-latest
$ pyenv local miniconda3-latest
$ pyenv activate miniconda3-latest
$ conda create -n ana anaconda
```

To activate use:

```
$ source activate ana
```

To deactivate use:

```
$ source deactivate
```

To install cloudmesh cmd5 please use:

```
$ pip install cloudmesh.cmd5
$ pip install cloudmesh.sys
```

15.2.6.2 Anaconda

⚠ This section about anaconda is experimental and has not been tested. We are looking for contributors that help completing it.

You can add anaconda to your pyenv with the following commands:

```
pyenv install anaconda3-4.3.1
```

To switch more easily we recommend that you use the following in your `.bash_profile` file:

```
alias ANA="pyenv activate anaconda3-4.3.1"
```

Once you have done this you can easily switch to anaconda with the command:

```
$ ANA
```

Terminology in anaconda could lead to confusion. Thus we like to point out that the version number of anaconda is unrelated to the python version. Furthermore, anaconda uses the term root not for the root user, but for the originating directory in which the anaconda program is installed.

In case you like to build your own conda packages at a later time we recommend that you install the conda-build package:

```
$ conda install conda-build
```

When executing:

```
pyenv versions
```

you will see after the install completed the anaconda versions installed:

```
pyenv versions
system
2.7.15
2.7.15/envs/ENV2
3.7.1
3.7.1/envs/ENV3
ENV2
ENV3
* anaconda3-4.3.1 (set by PYENV_VERSION environment variable)
```

Let us now create virtualenv for anaconda:

```
$ pyenv virtualenv anaconda3-4.3.1 ANA
```

To activate it you can now use:

```
$ pyenv ANA
```

However, anaconda may modify your `.bashrc` or `.bash_profile` files and may result in incompatibilities with other python versions. For this reason we recommend not to use it. If you find ways to get it to work reliably with other versions, please let us know and we update this tutorial.

To install cloudmesh cmd5 please use:

```
$ pip install cloudmesh.cmd5
$ pip install cloudmesh.sys
```

15.2.6.3 virtualenv

Documentation about it can be found at:

```
* https://virtualenv.pypa.io
```

The installation is simple once you have pip installed. If it is not installed you can say:

```
$ easy_install pip
```

After that you can install the virtual env with:

```
$ pip install virtualenv
```

To setup an isolated environment for example in the directory ~/ENV please use:

```
$ virtualenv ~/ENV
```

To activate it you can use the command:

```
$ source ~/ENV/bin/activate
```

you can put this command in your `.bashrc` or `.bash_profile` files so you do not forget to activate it. Instructions for this can be found in our lesson on Linux `bashrc`.

15.2.6.3.1 Exercises

E.Python.Install.0:

> Write installation instructions for an operating system of your choice and add to this documentation.

E.Python.Install.1:

> Replicate the steps to install pyenv, so you can type in ENV2 and ENV3 in your terminals to switch between python 2 and 3.

E.Python.Install.3:

> Why do you not want to use generally anaconda for cloud computing? When is it ok to use anaconda?

15.3 Interactive Python

Python can be used interactively. You can enter the interactive mode by entering the interactive loop by executing the command:

```
$ python
```

You will see something like the following:

```
python
Python 3.7.1 (default, Nov 24 2018, 14:27:15)
[Clang 10.0.0 (clang-1000.11.45.5)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

The `>>>` is the prompt used by the interpreter. This is similar to bash where commonly `$` is used.

Sometimes it is convenient to show the prompt when illustrating an example. This is to provide some context for what we are doing. If you are following along you will not need to type in the prompt.

This interactive python process does the following:

- read your input commands
- evaluate your command
- print the result of evaluation
- loop back to the beginning.

This is why you may see the interactive loop referred to as a **REPL**: **R**ead-**E**valuate-**P**rint-**L**oop.

15.3.1 REPL (Read Eval Print Loop)

There are many different types beyond what we have seen so far, such as **dictionaries**, **list**s, **set**s. One handy way of using the interactive python is to get the type of a value using type():

```
>>> type(42)
<type 'int'>
>>> type(hello)
<type 'str'>
>>> type(3.14)
<type 'float'>
```

You can also ask for help about something using help():

```
>>> help(int)
>>> help(list)
>>> help(str)
```

Using help() opens up a help message within a pager. To navigate you can use the spacebar to go down a page w to go up a page, the arrow keys to go up/down line-by-line, or q to exit.

15.3.2 Interpreter

Although the interactive mode provides a convenient tool to test things out you will see quickly that for our class we want to use the python interpreter from the commandline. Let us assume the program is called `prg.py`. Once you have written it in that file you simply can call it with

```
$ python prg.py
```

It is important to name the program with meaningful names.

15.3.3 Python 3 Features in Python 2

In this course we want to be able to seamlessly switch between python 2 and python 3. Thus it is convenient from the start to use python 3 syntax when it is supported also in python 2. One of the most used functions is the print statement that has in python 3 parentheses. To enable it in python 2 you just need to import this function:

```
from __future__ import print_function, division
```

The first of these imports allows us to use the print function to output text to the screen, instead of the print statement, which Python 2 uses. This is simply a design decision that better reflects Python's underlying philosophy.

Other functions such as the division also behave differently. Thus we use

```
from __future__ import division
```

This import makes sure that the division operator behaves in a way a newcomer to the language might find more intuitive. In Python 2, division / is floor division when the arguments are integers, meaning that the following

```
(5 / 2 == 2) is True
```

In Python 3, division / is a floating point division, thus

```
(5 / 2 == 2.5) is True
```

15.4 Editors

This section is meant to give an overview of the python editing tools needed for doing for this course. There are many other alternatives, however, we do recommend to use PyCharm.

15.4.1 Pycharm

PyCharm is an Integrated Development Environment (IDE) used for programming in Python. It provides code analysis, a graphical debugger, an integrated unit tester, integration with git.

Python 8:56 Pycharm

15.4.2 Python in 45 minutes

An additional community video about the Python programming language that we found on the internet. Naturally there are many alternatives to this video, but the video is probably a good start. It also uses PyCharm which we recommend.

Python 43:16 PyCharm

How much you want to understand of python is actually a bit up to you. While its good to know classes and inheritance, you may be able for this class to get away without using it. However, we do recommend that you learn it.

PyCharm Installation: Method 1: PyCharm Installation on ubuntu using umake

```
sudo add-apt-repository ppa:ubuntu-desktop/ubuntu-make
sudo apt-get update
sudo apt-get install ubuntu-make
```

Once umake command is run, use below command to install Pycharm community edition:

```
umake ide pycharm
```

If you want to remove PyCharm installed using umake command, use this:

```
umake -r ide pycharm
```

Method 2: PyCharm installation on ubuntu using PPA

```
sudo add-apt-repository ppa:mystic-mirage/pycharm
sudo apt-get update
sudo apt-get install pycharm-community
```

PyCharm also has a Professional (paid) version which can be installed using following command:

```
sudo apt-get install pycharm
```

Once installed, go to your VM dashboard and search for PyCharm.

15.5 Language

15.5.1 Statements and Strings

TODO: some of the python examples assume REPL, but its better to use a print statement instead as more general, please fix} Let us explore the syntax of Python. Type into the interactive loop and press Enter:

```
print("Hello world from Python!")
```

This will print on the terminal

```
Hello world from Python
```

What happened? The print function was given a **string** to process. A string is a sequence of characters. A **character** can be a alphabetic (A through Z, lower and upper case), numeric (any of the digits), white space (spaces, tabs, newlines, etc), syntactic directives (comma, colon, quotation, exclamation, etc), and so forth. A string is just a sequence of the character and typically indicated by surrounding the characters in double quotes.

Standard output is discussed in the Section Linux.

So, what happened when you pressed Enter? The interactive Python program read the line `print ("Hello world from Python!")`, split it into the print statement and the `"Hello world from Python!"` string, and then executed the line, showing you the output.

15.5.2 Variables

You can store data into a **variable** to access it later. For instance, instead of:

```
print('Hello world from Python!')
```

which is a lot to type if you need to do it multiple times, you can store the string in a variable for convenient access:

```
hello = 'Hello world from Python!'
print(hello)
```

This will print again

```
Hello world from Python!
```

15.5.3 Data Types

15.5.3.1 Booleans

A **boolean** is a value that indicates truthness of something. You can think of it as a toggle: either “on” or “off”, “one” or “zero”, “true” or “false”. In fact, the only possible values of the **boolean** (or bool) type in Python are:

- True
- False

You can combine booleans with **boolean operators**:

- and
- or

```
print(True and True)
## True

print(True and False)
## False

print(False and False)
## False

print(True or True)
## True

print(True or False)
## True

print(False or False)
## False
```

15.5.3.2 Numbers

The interactive interpreter can also be used as a calculator. For instance, say we wanted to compute a multiple of 21:

```
print(21 * 2)
## 42
```

We saw here the print statement again. We passed in the result of the operation 21 * 2. An **integer** (or **int**) in Python is a numeric value without a fractional component (those are called **floating point** numbers, or **float** for short).

The mathematical operators compute the related mathematical operation to the provided numbers. Some operators are:

Operator	Function
*	multiplication
/	division
+	addition
-	subtraction
**	exponent

Exponentiation x^y is written as x**y is x to the yth power.

You can combine **float**s and **int**s:

```
print(3.14 * 42 / 11 + 4 - 2)
## 13.9890909091

print(2**3)
## 8
```

Note that **operator precedence** is important. Using parenthesis to indicate affect the order of operations gives a difference results, as expected:

```
print(3.14 * (42 / 11) + 4 - 2)
## 11.42

print(1 + 2 * 3 - 4 / 5.0)
## 6.2

print( (1 + 2) * (3 - 4) / 5.0 )
```

```
## -0.6
```

15.5.4 Module Management

A module allows you to logically organize your Python code. Grouping related code into a module makes the code easier to understand and use. A module is a Python object with arbitrarily named attributes that you can bind and reference. A module is a file consisting of Python code. A module can define functions, classes and variables. A module can also include runnable code.

15.5.4.1 Import Statement

When the interpreter encounters an import statement, it imports the module if the module is present in the search path. A search path is a list of directories that the interpreter searches before importing a module. The from...import Statement Python's from statement lets you import specific attributes from a module into the current namespace. It is preferred to use for each import its own line such as:

```
import numpy
import matplotlib
```

When the interpreter encounters an import statement, it imports the module if the module is present in the search path. A search path is a list of directories that the interpreter searches before importing a module.

15.5.4.2 The from ... import Statement

Python's from statement lets you import specific attributes from a module into the current namespace. The from ... import has the following syntax:

```
from datetime import datetime
```

15.5.5 Date Time in Python

The datetime module supplies classes for manipulating dates and times in both simple and complex ways. While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation. For related functionality, see also the time and calendar modules.

The import Statement You can use any Python source file as a module by executing an import statement in some other Python source file.

```
from datetime import datetime
```

This module offers a generic date/time string parser which is able to parse most known formats to represent a date and/or time.

```
from dateutil.parser import parse
```

pandas is an open source Python library for data analysis that needs to be imported.

```
import pandas as pd
```

Create a string variable with the class start time

```
fall_start = '08-21-2018'
```

Convert the string to datetime format

```
datetime.strptime(fall_start, '%m-%d-%Y') \#
datetime.datetime(2017, 8, 21, 0, 0)
```

Creating a list of strings as dates

```
class_dates = ['8/25/2017', '9/1/2017', '9/8/2017', '9/15/2017', '9/22/2017', '9/29/2017']
```

Convert Class_dates strings into datetime format and save the list into variable a

```
a = [datetime.strptime(x, '%m/%d/%Y') for x in class_dates]
```

Use parse() to attempt to auto-convert common string formats. Parser must be a string or character stream, not list.

```
parse(fall_start)
## datetime.datetime(2017, 8, 21, 0, 0)
```

Use parse() on every element of the Class_dates string.

```
[parse(x) for x in class_dates]
## [datetime.datetime(2017, 8, 25, 0, 0),
##   datetime.datetime(2017, 9, 1, 0, 0),
##   datetime.datetime(2017, 9, 8, 0, 0),
##   datetime.datetime(2017, 9, 15, 0, 0),
##   datetime.datetime(2017, 9, 22, 0, 0),
##   datetime.datetime(2017, 9, 29, 0, 0)]
```

Use parse, but designate that the day is first.

```
parse (fall_start, dayfirst=True)
## datetime.datetime(2017, 8, 21, 0, 0)
```

Create a dataframe.A DataFrame is a tabular data structure comprised of rows and columns, akin to a spreadsheet, database table. DataFrame as a group of Series objects that share an index (the column names).

```
import pandas as pd
data = {
  'dates': [
    '8/25/2017 18:47:05.069722',
    '9/1/2017 18:47:05.119994',
    '9/8/2017 18:47:05.178768',
    '9/15/2017 18:47:05.230071',
    '9/22/2017 18:47:05.230071',
    '9/29/2017 18:47:05.280592'],
  'complete': [1, 0, 1, 1, 0, 1]}
df = pd.DataFrame(
  data,
  columns = ['dates','complete'])
print(df)
##                  dates  complete
## 0  8/25/2017 18:47:05.069722 1
## 1   9/1/2017 18:47:05.119994 0
## 2   9/8/2017 18:47:05.178768 1
## 3  9/15/2017 18:47:05.230071 1
## 4  9/22/2017 18:47:05.230071 0
## 5  9/29/2017 18:47:05.280592 1
```

Convert `df[`date`]` from string to datetime

```
import pandas as pd
pd.to_datetime(df['dates'])
## 0   2017-08-25 18:47:05.069722
## 1   2017-09-01 18:47:05.119994
## 2   2017-09-08 18:47:05.178768
## 3   2017-09-15 18:47:05.230071
## 4   2017-09-22 18:47:05.230071
## 5   2017-09-29 18:47:05.280592
## Name: dates, dtype: datetime64[ns]
```

15.5.6 Control Statements

15.5.6.1 Comparison

Computer programs do not only execute instructions. Occasionally, a choice needs to be made. Such as a choice is based on a condition. Python has several conditional operators:

Operator	Function

>	greater than
<	smaller than
==	equals
!=	is not

Conditions are always combined with variables. A program can make a choice using the if keyword. For example:

```
x = int(input("Guess x:"))
if x == 4:
   print('Correct!')
```

In this example, You guessed correctly! will only be printed if the variable x equals to four. Python can also execute multiple conditions using the elif and else keywords.

```
x = int(input("Guess x:"))
if x == 4:
    print('Correct!')
elif abs(4 - x) == 1:
    print('Wrong, but close!')
else:
    print('Wrong, way off!')
```

15.5.6.2 Iteration

To repeat code, the for keyword can be used. For example, to display the numbers from 1 to 10, we could write something like this:

```
for i in range(1, 11):
   print('Hello!')
```

The second argument to range, 11, is not inclusive, meaning that the loop will only get to 10 before it finishes. Python itself starts counting from 0, so this code will also work:

```
for i in range(0, 10):
   print(i + 1)
```

In fact, the range function defaults to starting value of 0, so it is equivalent to:

```
for i in range(10):
   print(i + 1)
```

We can also nest loops inside each other:

```
for i in range(0,10):
    for j in range(0,10):
        print(i,' ',j)
```

In this case we have two nested loops. The code will iterate over the entire coordinate range (0,0) to (9,9)

15.5.7 Datatypes

15.5.7.1 Lists

see: https://www.tutorialspoint.com/python/python_lists.htm

Lists in Python are ordered sequences of elements, where each element can be accessed using a 0-based index.

To define a list, you simply list its elements between square brackets '[ ]':

```
names = [
  'Albert',
  'Jane',
  'Liz',
  'John',
  'Abby']
## access the first element of the list
names[0]
## 'Albert'
## access the third element of the list
names[2]
## 'Liz'
```

You can also use a negative index if you want to start counting elements from the end of the list. Thus, the last element has index -1, the second before last element has index -2 and so on:

```
## access the last element of the list
names[-1]
## 'Abby'
## access the second last element of the list
names[-2]
## 'John'
```

Python also allows you to take whole slices of the list by specifying a beginning and end of the slice separated by a colon

```
## the middle elements, excluding first and last
names[1:-1]
## ['Jane', 'Liz', 'John']
```

As you can see from the example, the starting index in the slice is inclusive and the ending one, exclusive.

Python provides a variety of methods for manipulating the members of a list.

You can add elements with append':

```
names.append('Liz')
names
## ['Albert', 'Jane', 'Liz',
##  'John', 'Abby', 'Liz']
```

As you can see, the elements in a list need not be unique.

Merge two lists with 'extend':

```
names.extend(['Lindsay', 'Connor'])
names
## ['Albert', 'Jane', 'Liz', 'John',
##  'Abby', 'Liz', 'Lindsay', 'Connor']
```

Find the index of the first occurrence of an element with 'index':

```
names.index('Liz') \## 2
```

Remove elements by value with 'remove':

```
names.remove('Abby')
names
## ['Albert', 'Jane', 'Liz', 'John',
##  'Liz', 'Lindsay', 'Connor']
```

Remove elements by index with 'pop':

```
names.pop(1)
## 'Jane'
names
## ['Albert', 'Liz', 'John',
##  'Liz', 'Lindsay', 'Connor']
```

Notice that pop returns the element being removed, while remove does not.

If you are familiar with stacks from other programming languages, you can use insert and 'pop':

```
names.insert(0, 'Lincoln')
names
## ['Lincoln', 'Albert', 'Liz',
##  'John', 'Liz', 'Lindsay', 'Connor']
names.pop()
## 'Connor'
names
## ['Lincoln', 'Albert', 'Liz',
##  'John', 'Liz', 'Lindsay']
```

The Python documentation contains a full list of list operations.

To go back to the range function you used earlier, it simply creates a list of numbers:

```
range(10)
## [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
range(2, 10, 2)
## [2, 4, 6, 8]
```

15.5.7.2 Sets

Python lists can contain duplicates as you saw previously:

```
names = ['Albert', 'Jane', 'Liz',
         'John', 'Abby', 'Liz']
```

When we do not want this to be the case, we can use a set:

```
unique_names = set(names)
unique_names
## set(['Lincoln', 'John', 'Albert', 'Liz', 'Lindsay'])
```

Keep in mind that the set is an unordered collection of objects, thus we can not access them by index:

```
unique_names[0]
## Traceback (most recent call last):
##   File "<stdin>", line 1, in <module>
##   TypeError: 'set' object does not support indexing
```

However, we can convert a set to a list easily:

```
unique_names = list(unique_names)
unique_names [`Lincoln', `John', `Albert', `Liz', `Lindsay']
unique_names[0]
## `Lincoln'
```

Notice that in this case, the order of elements in the new list matches the order in which the elements were displayed when we create the set. We had

```
set(['Lincoln', 'John', 'Albert', 'Liz', 'Lindsay'])
```

and now we have

```
['Lincoln', 'John', 'Albert', 'Liz', 'Lindsay'])
```

You should not assume this is the case in general. That is, do not make any assumptions about the order of elements in a set when it is converted to any type of sequential data structure.

You can change a set's contents using the add, remove and update methods which correspond to the append, remove and extend methods in a list. In addition to these, set objects support the operations you may be familiar with from mathematical sets: union, intersection, difference, as well as operations to check containment. You can read about this in the Python documentation for sets.

15.5.7.3 Removal and Testing for Membership in Sets

One important advantage of a `set` over a `list` is that **access to elements is fast**. If you are familiar with different data structures from a Computer Science class, the Python list is implemented by an array, while the set is implemented by a hash table.

We will demonstrate this with an example. Let's say we have a list and a set of the same number of elements (approximately 100 thousand):

```
import sys, random, timeit
nums_set = set([random.randint(0, sys.maxint) for _ in range(10**5)])
nums_list = list(nums_set)
len(nums_set)
## 100000
```

We will use the timeit Python module to time 100 operations that test for the existence of a member in either the list or set:

```
timeit.timeit('random.randint(0, sys.maxint) in nums',
              setup='import random; nums=%s' % str(nums_set), number=100)
## 0.0004038810729980469
timeit.timeit('random.randint(0, sys.maxint) in nums',
              setup='import random; nums=%s' % str(nums_list), number=100)
## 0.398054122924804
```

The exact duration of the operations on your system will be different, but the take away will be the same: searching for an element in a set is orders of magnitude faster than in a list. This is important to keep in mind when you work with large amounts of data.

15.5.7.4 Dictionaries

One of the very important data structures in python is a dictionary also referred to as dict.

A dictionary represents a key value store:

```
person = {
  'Name': 'Albert',
  'Age': 100,
  'Class': 'Scientist'
  }
print("person['Name']: ", person['Name'])
## person['Name']:  Albert
print("person['Age']: ", person['Age'])
## person['Age']:  100
```

A convenient for to print by named attributes is

```
print("{Name} {Age}'.format(**data))
```

This form of printing with the format statement and a reference to data increases readability of the print statements.

You can delete elements with the following commands:

```
del person['Name'] ## remove entry with key 'Name'
person
## {'Age': 100, 'Class': 'Scientist'}
person.clear()     ## remove all entries in dict
## person
## {}
del person         ## delete entire dictionary
person
## Traceback (most recent call last):
```

```
##  File "<stdin>", line 1, in <module>
##  NameError: name 'person' is not defined
```

You can iterate over a dict:

```
person = {
  'Name': 'Albert',
  'Age': 100,
  'Class': 'Scientist'
  }
for item in person:
  print(item, person[item])

## Age 100
## Name Albert
## Class Scientist
```

15.5.7.5 Dictionary Keys and Values

You can retrieve both the keys and values of a dictionary using the keys() and values() methods of the dictionary, respectively:

```
person.keys()
## ['Age', 'Name', 'Class']
person.values()
## [100, 'Albert', 'Scientist']
```

Both methods return lists. Notice, however, that the order in which the elements appear in the returned lists (Age, Name, Class) is different from the order in which we listed the elements when we declared the dictionary initially (Name, Age, Class). It is important to keep this in mind:

> ⚠ you cannot make any assumptions about the order in which the elements of a dictionary will be returned by the keys() and values() methods**.

However, you can assume that if you call `keys()` and `values()` in sequence, the order of elements will at least correspond in both methods. In the example Age corresponds to 100, Name to `Albert`, and Class to Scientist, and you will observe the same correspondence in general as long as *`keys()`* and *`values()`* are called one right after the other.

15.5.7.6 Counting with Dictionaries

One application of dictionaries that frequently comes up is counting the elements in a sequence. For example, say we have a sequence of coin flips:

```
import random
die_rolls = [
  random.choice(['heads', 'tails']) for _ in range(10)
]
## die_rolls
## ['heads', 'tails', 'heads',
##  'tails', 'heads', 'heads',
   'tails', 'heads', 'heads', 'heads']
```

The actual list die_rolls will likely be different when you execute this on your computer since the outcomes of the die rolls are random.

To compute the probabilities of heads and tails, we could count how many heads and tails we have in the list:

```
counts = {'heads': 0, 'tails': 0}
for outcome in coin_flips:
   assert outcome in counts
   counts[outcome] += 1
print('Probability of heads: %.2f' % (counts['heads'] / len(coin_flips)))
## Probability of heads: 0.70

print('Probability of tails: %.2f' % (counts['tails'] / sum(counts.values())))
## Probability of tails: 0.30
```

In addition to how we use the dictionary counts to count the elements of coin_flips, notice a couple things about this example:

1. We used the assert outcome in counts statement. The assert statement in Python allows you to easily insert debugging statements in your code to help you discover errors more quickly. assert statements are executed whenever the internal Python `__debug__` variable is set to True, which is always the case unless you start Python with the -O option which allows you to run optimized Python.
2. When we computed the probability of tails, we used the built-in sum function, which allowed us to quickly find the total number of coin flips. sum is one of many built-in function you can read about here.

15.5.8 Functions

You can reuse code by putting it inside a function that you can call in other parts of your programs. Functions are also a good way of grouping code that logically belongs together in one coherent whole. A function has a unique name in the program. Once you call a function, it will execute its body which consists of one or more lines of code:

```
def check_triangle(a, b, c):
return \
    a < b + c and a > abs(b - c) and \
    b < a + c and b > abs(a - c) and \
    c < a + b and c > abs(a - b)

    print(check_triangle(4, 5, 6))
```

The def keyword tells Python we are defining a function. As part of the definition, we have the function name, check_triangle, and the parameters of the function – variables that will be populated when the function is called.

We call the function with arguments 4, 5 and 6, which are passed in order into the parameters a, b and c. A function can be called several times with varying parameters. There is no limit to the number of function calls.

It is also possible to store the output of a function in a variable, so it can be reused.

```
def check_triangle(a, b, c):
  return \
     a < b + c and a > abs(b - c) and \
     b < a + c and b > abs(a - c) and \
     c < a + b and c > abs(a - b)

    result = check_triangle(4, 5, 6)
    print(result)
```

15.5.9 Classes

A class is an encapsulation of data and the processes that work on them. The data is represented in member variables, and the processes are defined in the methods of the class (methods are functions inside the class). For example, let's see how to define a Triangle class:

```
class Triangle(object):

  def __init__(self, length, width,
               height, angle1, angle2, angle3):
     if not self._sides_ok(length, width, height):
         print('The sides of the triangle are invalid.')
     elif not self._angles_ok(angle1, angle2, angle3):
         print('The angles of the triangle are invalid.')

     self._length = length
     self._width = width
     self._height = height

     self._angle1 = angle1
     self._angle2 = angle2
     self._angle3 = angle3

  def _sides_ok(self, a, b, c):
     return \
```

```
        a < b + c and a > abs(b - c) and \
        b < a + c and b > abs(a - c) and \
        c < a + b and c > abs(a - b)

 def _angles_ok(self, a, b, c):
     return a + b + c == 180

triangle = Triangle(4, 5, 6, 35, 65, 80)
```

Python has full object-oriented programming (OOP) capabilities, however we can not cover all of them in this section, so if you need more information please refer to the Python docs on classes and OOP.

15.5.10 Modules

Now write this simple program and save it:

```
from __future__ import print_statement, division
print("Hello world!")
```

As a check, make sure the file contains the expected contents on the command line:

```
$ cat hello.py
from __future__ import print_statement, division
print("Hello world!")
```

To execute your program pass the file as a parameter to the python command:

```
$ python hello.py
Hello world!
```

Files in which Python code is stored are called **module**s. You can execute a Python module form the command line like you just did, or you can import it in other Python code using the import statement.

Let's write a more involved Python program that will receive as input the lengths of the three sides of a triangle, and will output whether they define a valid triangle. A triangle is valid if the length of each side is less than the sum of the lengths of the other two sides and greater than the difference of the lengths of the other two sides.:

```
"""Usage: check_triangle.py [-h] LENGTH WIDTH HEIGHT

Check if a triangle is valid.

Arguments:
  LENGTH     The length of the triangle.
  WIDTH      The width of the traingle.
  HEIGHT     The height of the triangle.

Options:
-h --help
"""
from __future__ import print_function, division
from docopt import docopt

if __name__ == '__main__':
  arguments = docopt(__doc__)
  a, b, c = int(arguments['LENGTH']),
            int(arguments['WIDTH']),
            int(arguments['HEIGHT'])
  valid_triangle = \
      a < b + c and a > abs(b - c) and \
      b < a + c and b > abs(a - c) and \
      c < a + b and c > abs(a - b)
  print('Triangle with sides %d, %d and %d is valid: %r' % (
      a, b, c, valid_triangle
  ))
```

Assuming we save the program in a file called `check_triangle.py`, we can run it like so:

```
$ python check_triangle.py 4 5 6
Triangle with sides 4, 5 and 6 is valid: True
```

Let us break this down a bit.

1. We are importing the print_function and division modules from python 3 like we did earlier in this section. It's a good idea to always include these in your programs.
2. We've defined a boolean expression that tells us if the sides that were input define a valid triangle. The result of the expression is stored in the valid_triangle variable. inside are true, and False otherwise.
3. We've used the backslash symbol \ to format are code nicely. The backslash simply indicates that the current line is being continued on the next line.
4. When we run the program, we do the check if `__name__ == '__main__'`. `__name__` is an internal Python variable that allows us to tell whether the current file is being run from the command line (value `__name__`), or is being imported by a module (the value will be the name of the module). Thus, with this statement we're just making sure the program is being run by the command line.
5. We are using the docopt module to handle command line arguments. The advantage of using this module is that it generates a usage help statement for the program and enforces command line arguments automatically. All of this is done by parsing the docstring at the top of the file.
6. In the print function, we are using Python's string formatting capabilities to insert values into the string we are displaying.

15.5.11 Lambda Expressions

As oppose to normal functions in Python which are defined using the `def` keyword, lambda functions in Python are anonymous functions which do not have a name and are defined using the `lambda` keyword. The generic syntax of a lambda function is in form of`lambda arguments: expression`, as shown in the following example:

```
greeter = lambda x: print('Hello %s!'%x)
print(greeter('Albert'))
```

As you could probably guess, the result is:

```
Hello Albert!
```

Now consider the following examples:

```
power2 = lambda x: x ** 2
```

The `power2` function defined in the expression, is equivalent to the following definition:

```
def power2(x):
    return x ** 2
```

Lambda functions are useful for when you need a function for a short period of time. Note that they can also be very useful when passed as an argument with other built-in functions that take a function as an argument, e.g. `filter()` and `map()`. In the next example we show how a lambda function can be combined with the `filer` function. Consider the array `all_names` which contains five words that rhyme together. We want to filter the words that contain the word `name`. To achieve this, we pass the function `lambda x: 'name' in x` as the first argument. This lambda function returns `True` if the word `name` exists as a sub-string in the string `x`. The second argument of `filter` function is the array of names, i.e. `all_names`.

```
all_names = ['surname', 'rename', 'nickname', 'acclaims', 'defame']
filtered_names = list(filter(lambda x: 'name' in x, all_names))
print(filtered_names)
## ['surname', 'rename', 'nickname']
```

As you can see, the names are successfully filtered as we expected.

In Python3, filter function returns a filter object or the iterator which gets lazily evaluated which means neither we can access the elements of the filter object with index nor we can use len() to find the length of the filter object.

```
list_a = [1, 2, 3, 4, 5]
filter_obj = filter(lambda x: x % 2 == 0, list_a)
## Convert the filer obj to a list
even_num = list(filter_obj)
print(even_num)
## Output: [2, 4]
```

In Python, we can have a small usually a single liner anonymous function called Lambda function which can have any number of arguments just like a normal function but with only one expression with no return statement. The result of this expression can be applied to a value.

Basic Syntax:

```
lambda arguments : expression
```

For an example: a function in python

```
def multiply(a, b):
    return a*b

#call the function
multiply(3*5) #outputs: 15
```

Same function can written as Lambda function. This function named as multiply is having 2 arguments and returns their multiplication.

Lambda equivalent for this function would be:

```
multiply = Lambda a, b : a*b

print(multiply(3, 5))
## outputs: 15
```

Here a and b are the 2 arguments and a*b is the expression whose value is returned as an output.

Also we don't need to assign Lambda function to a variable.

```
(lambda a, b : a*b)(3*5)
```

Lambda functions are mostly passed as parameter to a function which expects a function objects like in map or filter.

15.5.11.1 map

The basic syntax of the map function is

```
map(function_object, iterable1, iterable2,...)
```

map functions expects a function object and any number of iterables like list or dictionary. It executes the function_object for each element in the sequence and returns a list of the elements modified by the function object.

Example:

```
def multiply(x):
    return x * 2

map(multiply2, [2, 4, 6, 8])
## Output [4, 8, 12, 16]
```

If we want to write same function using Lambda

```
map(lambda x: x*2, [2, 4, 6, 8])
## Output [4, 8, 12, 16]
```

15.5.11.2 dictionary

Now, lets see how we can interate over a dictionary using map and lambda Lets say we have a dictionary object

```
dict_movies = [
    {'movie': 'avengers', 'comic': 'marvel'},
    {'movie': 'superman', 'comic': 'dc'}]
```

We can iterate over this dictionary and read the elements of it using map and lambda functions in following way:

```
map(lambda x : x['movie'], dict_movies)  ## Output: ['avengers', 'superman']
map(lambda x : x['comic'],  dict_movies)  ## Output: ['marvel', 'dc']
map(lambda x : x['movie'] == "avengers", dict_movies)
## Output: [True, False]
```

In Python3, map function returns an iterator or map object which gets lazily evaluated which means neither we can access the elements of the map object with index nor we can use len() to find the length of the map object. We can force convert the map output i.e. the map object to list as shown below:

```
map_output = map(lambda x: x*2, [1, 2, 3, 4])
print(map_output)
## Output: map object: <map object at 0x04D6BAB0>
list_map_output = list(map_output)
print(list_map_output) ## Output: [2, 4, 6, 8]
```

15.5.12 Iterators

In Python, an iterator protocol is defined using two methods: `__iter()__` and `next()`. The former returns the iterator object and latter returns the next element of a sequence. Some advantages of iterators are as follows:

- Readability
- Supports sequences of infinite length
- Saving resources

There are several built-in objects in Python which implement iterator protocol, e.g. string, list, dictionary. In the following example, we create a new class that follows the iterator protocol. We then use the class to generate `log2` of numbers:

```
from math import log2

class LogTwo:
    "Implements an iterator of log two"

    def __init__(self,last = 0):
        self.last = last

    def __iter__(self):
        self.current_num = 1
        return self

    def __next__(self):
        if self.current_num <= self.last:
            result = log2(self.current_num)
            self.current_num += 1
            return result
        else:
            raise StopIteration

L = LogTwo(5)
i = iter(L)
print(next(i))
print(next(i))
print(next(i))
print(next(i))
```

As you can see, we first create an instance of the class and assign its `__iter()__` function to a variable called `i`. Then by calling the `next()` function four times, we get the following output:

```
$ python iterator.py
0.0
1.0
1.584962500721156
2.0
```

As you probably noticed, the lines are `log2()` of 1, 2, 3, 4 respectively.

15.5.13 Generators

Before we go to Generators, please understand Iterators. Generators are also Iterators but they can only be interated over once. Thats because Generators do not store the values in memory instead they generate

the values on the go. If we want to print those values then we can either simply iterate over them or use the for loop.

15.5.13.1 Generators with function

For example: we have a function named as multiplyBy10 which prints all the input numbers multiplied by 10.

```
def multiplyBy10(numbers):
    result = []
    for i in numbers:
        result.append(i*10)
    return result

new_numbers = multiplyBy10([1,2,3,4,5])

print new_numbers  #Output: [10, 20, 30, 40 ,50]
```

Now, if we want to use Generators here then we will make following changes.

```
def multiplyBy10(numbers):
    for i in numbers:
        yield(i*10)

new_numbers = multiplyBy10([1,2,3,4,5])

print new_numbers  #Output: Generators object
```

In Generators, we use yield() function in place of return(). So when we try to print new_numbers list now, it just prints Generators object. The reason for this is because Generators dont hold any value in memory, it yields one result at a time. So essentially it is just waiting for us to ask for the next result. To print the next result we can just say print next(new_numbers) , so how it is working is its reading the first value and squaring it and yielding out value 1. Also in this case we can just print next(new_numbers) 5 times to print all numbers and if we do it for 6th time then we will get an error StopIteration which meanns Generators has exausted its limit and it has no 6th element to print.

```
print next(new_numbers)  #Output: 1
```

15.5.13.2 Generators using for loop

If we now want to print the complete list of squared values then we can just do:

```
def multiplyBy10(numbers):
    for i in numbers:
        yield(i*10)

new_numbers = multiplyBy10([1,2,3,4,5])

for num in new_numbers:
    print num
```

The output will be:

```
10
20
30
40
50
```

15.5.13.3 Generators with List Comprehension

Python has something called List Comprehension, if we use this then we can replace the complete function def with just:

```
new_numbers = [x*10 for x in [1,2,3,4,5]]
print new_numbers  #Output: [10, 20, 30, 40 ,50]
```

Here the point to note is square brackets [] in line 1 is very important. If we change it to () then again we will start getting Generators object.

```
new_numbers = (x*10 for x in [1,2,3,4,5])
print new_numbers  #Output: Generators object
```

We can get the individual elements again from Generators if we do a for loop over new_numbers like we did previously. Alternatively, we can convert it into a list and then print it.

```
new_numbers = (x*10 for x in [1,2,3,4,5])
print list(new_numbers)  #Output: [10, 20, 30, 40 ,50]
```

But here if we convert this into a list then we loose on performance, which we will just see next.

15.5.13.4 Why to use Generators?

Generators are better with Performance because it does not hold the values in memory and here with the small examples we provide its not a big deal since we are dealing with small amount of data but just consider a scenario where the records are in millions of data set. And if we try to convert millions of data elements into a list then that will definitely make an impact on memory and performance because everything will in memory.

Lets see an example on how Generators help in Performance. First, without Generators, normal function taking 1 million record and returns the result[people] for 1 million.

```
names = ['John', 'Jack', 'Adam', 'Steve', 'Rick']
majors = ['Math', 'CompScience', 'Arts', 'Business', 'Economics']

## prints the memory before we run the function
memory = mem_profile.memory_usage_resource()
print ('Memory (Before): {memory}Mb'.format(memory=memory))

def people_list(people):
    result = []
    for i in range(people):
        person = {
                'id' : i,
                'name' : random.choice(names),
                'major' : randon.choice(majors)
                }
        result.append(person)
    return result

t1 = time.clock()
people = people_list(10000000)
t2 = time.clock()

## prints the memory after we run the function
memory = mem_profile.memory_usage_resource()
print ('Memory (After): {memory}Mb'.format(memory=memory))
print ('Took {time} seconds'.format(time=t2-t1))

#Output
Memory (Before): 15Mb
Memory (After): 318Mb
Took 1.2 seconds
```

I am just giving approximate values to compare it with next execution but we just try to run it we will see a serious consumption of memory with good amount of time taken.

```
names = ['John', 'Jack', 'Adam', 'Steve', 'Rick']
majors = ['Math', 'CompScience', 'Arts', 'Business', 'Economics']

## prints the memory before we run the function
memory = mem_profile.memory_usage_resource()
print ('Memory (Before): {memory}Mb'.format(memory=memory))
def people_generator(people):
    for i in xrange(people):
        person = {
                'id' : i,
```

```
                'name' : random.choice(names),
                'major' : random.choice(majors)
                }
        yield person

t1 = time.clock()
people = people_list(10000000)
t2 = time.clock()

## prints the memory after we run the function
memory = mem_profile.memory_usage_resource()
print ('Memory (After): {memory}Mb'.format(memory=memory))print ('Took {time} seconds'.format(time=t2-t1))

#Output
Memory (Before): 15Mb
Memory (After): 15Mb
Took 0.01 seconds
```

Now after running the same code using Generators, we will see a significant amount of performance boost with alomost 0 Seconds. And the reason behind this is that in case of Generators, we do not keep anything in memory so system just reads 1 at a time and yields that.

15.5.14 Non Blocking Threads

Students can contribute this section

15.5.15 Subprocess

A module which allows us to start a new process and connect to their input, output, error nodes and get the return values is called a subprocess.

15.5.15.1 Popen Class

The most important class in Python to start a new process is Popen class. The other functions like call, check_output, and check_call use Popen internally. Signature of this class is as follows:

```
class subprocess.Popen(args, bufsize=0, executable=None, stdin=None,
stdout=None, stderr=None, preexec_fn=None, close_fds=False, shell=False,
cwd=None, env=None, universal_newlines=False, startupinfo=None,
creationflags=0)
```

Following program starts the Unix program 'cat' and the second parameter is the argument.

```
from subprocess import Popen, PIPE

process = Popen(['cat', 'test.py'], stdout=PIPE, stderr=PIPE)
stdout, stderr = process.communicate()
print stdout
```

process.communicate() reads the input and output from the process. stderr will only get populated if there is some error. stdout is the output for this process.

15.5.15.2 Popen.communicate()

The communicate() method returns a tuple (stdoutdata, stderrdata). Popen.communicate() interacts with process: Send data to stdin. Read data from stdout and stderr, until end-of-file is reached.

Wait for process to terminate.

The optional input argument should be a string to be sent to the child process, or None, if no data should be sent to the child.

Basically, when you use communicate() it means that you want to execute the command

15.5.15.3 Subprocess call()

The most recommended way to launch a process is to use following function with arguments and this will also have a returncode attribute:

```
subprocess.call(args, *, stdin=None, stdout=None, stderr=None, shell=False)
## Run the command described by args.
## Wait for command to complete, then return the returncode attribute.
```

The behaviour of the shell argument can sometimes be confusing so I'll try to clear it a bit here.

Firstly, lets consider the case where shell is set to False, the default. In this case, if args is a string, it is assumed to be the name of the executable file. Even if it contains spaces. Consider the following.

```
subprocess.call('ls -l')
```

This won't work because subprocess is looking for an executable file called ls -l, but obviously can't find it. However, if args is a list, then the first item in this list is considered as the executable and the rest of the items in the list are passed as command line arguments to the program.

```
subprocess.call(['ls', '-l'])
```

does what you think it will.

Second case, with shell set to True, the program that actually gets executed is the OS default shell, /bin/sh on Linux and cmd.exe on windows. This can be changed with the executable argument.

When using the shell, args is usually a string, something that will be parsed by the shell program. The args string is passed as a command line argument to the shell (with a -c option on Linux) such that the shell will interpret it as a shell command sequence and process it accordingly. This means you can use all the shell builtins and goodies that your shell offers.

```
subprocess.call('ls -l', shell=True)
```

is similar to

```
$ /bin/sh -c 'ls -l'
```

In the same vein, if you pass a list as args with shell set to True, all items in the list are passed as command line arguments to the shell.

```
subprocess.call(['ls', '-l'], shell=True)
```

is similar to

```
$ /bin/sh -c ls -l
```

which is the same as

```
$ /bin/sh -c ls
```

since /bin/sh takes just the argument next to -c as the command line to execute.

Example 2:

```
>>> subprocess.call("exit 1", shell=True)
1

**Warning: Using shell=True can cause some security issues. When we have shell=True then it will executed in shell. This can be useful if you are using Python primarily for the enhanced control f

If we execute shell command which takes in unsanitized input from an untrusted source, it can make the program prone to shell-injection, this can be a serious security risk. For this reason, the

>>> from subprocess import call
>>> filename = input("What file would you like to display?\n")
What file would you like to display?
non_existent; rm -rf / #
>>> call("cat " + filename, shell=True) ## Uh-oh. This will end badly...

shell=False disables all shell based features, but does not suffer from this vulnerability.
```

15.5.15.4 Save process output (stdout)

We can get the program output using check_output and store it in a string which we can later print. Method definition is as follows:

```
subprocess.check_output(args, *, stdin=None, stderr=None, shell=False, universal_newlines=False)
## Run command with arguments and return its output as a byte string.
```

Example:

```
>>> import subprocess
>>>
>>> s = subprocess.check_output(["echo", "Hello World!"])
>>> print("s = " + s)

'Hello World!\n'
```

If we want to get the standard error output, use stderr = subprocess.STDOUT

```
>>> subprocess.check_output("ls non_existent_file; exit 0", stderr=subprocess.STDOUT, shell=True)
'ls: non_existent_file: No such file or directory\n'
```

15.5.15.5 Getting the return code (OR exit status)

If we get a non-zero return code, then it will raise a CalledProcessError. This object will have return code in returncode attribute and output will be in output attribute.

```
>>> subprocess.check_output("exit 1", shell=True)
Traceback (most recent call last):
```

subprocess.CalledProcessError: Command 'exit 1' returned non-zero exit status 1

Exception subprocess.CalledProcessError

Exception raised when a process run by check_call() or check_output() returns a non-zero exit status.

returncode Exit status of the child process.

cmd Command that was used to spawn the child process.

output Output of the child process if this exception is raised by check_output(). Otherwise, None.

subprocess.PIPE Special value that can be used as the stdin, stdout or stderr argument to Popen and indicates that a pipe to the standard stream should be opened. Most useful with Popen.communicate().

subprocess.STDOUT Special value that can be used as the stderr argument to Popen and indicates that standard error should go into the same handle as standard output.

```
**Note: Do not use stdout=PIPE or stderr=PIPE with this function as that can deadlock based on the child process output volume. Use Popen with the communicate() method when you need pipes.
```

15.5.15.6 Popen Constructor

The process creation and its management is handled by this class - Popen. Its signature is as follows:

```
class subprocess.Popen(args, bufsize=0, executable=None, stdin=None, stdout=None, stderr=None, preexec_fn=None, close_fds=False, shell=False, cwd=None, env=None, universal_newlines=False, startup
```

This will execute a child program in a new process. The arguments to Popen is as follows:

```
args are a sequence of program arguments or it can be a single string.
```

If the arguments is a sequence, then by default, the first item in args is the program to execute. If args is a string, the interpretation is platform-dependent which will see next. Unless stated specifically, it is recommended to pass args as a sequence.

On Unix, if args is a string, the string is interpreted as the name or path of the program to execute. However, this can only be done if not passing arguments to the program.

```
Note shlex.split() can be useful when determining the correct tokenization for args, especially in complex cases:
>>> import shlex, subprocess
>>> command_line = raw_input()
/bin/vikings -input eggs.txt -output "spam spam.txt" -cmd "echo '$MONEY'"
>>> args = shlex.split(command_line)
>>> print args
['/bin/vikings', '-input', 'eggs.txt', '-output', 'spam spam.txt', '-cmd', "echo '$MONEY'"]
>>> p = subprocess.Popen(args) ## Success!
Note in particular that options (such as -input) and arguments (such as eggs.txt) that are separated by whitespace in the shell go in separate list elements, while arguments that need quoting or
```

On Windows, if args is a sequence then it will be converted to a string. This is because the underlying CreateProcess() operates on strings. Parsing the string after conversion uses the following rules:

1. Arguments are delimited by white space, which is either a space or a tab.
2. A string surrounded by double quotation marks is interpreted as a single argument, regardless of white space contained within. A quoted string can be embedded in an argument.
3. A double quotation mark preceded by a backslash is interpreted as a literal double quotation mark.
4. Backslashes are interpreted literally, unless they immediately precede a double quotation mark.
5. If backslashes immediately precede a double quotation mark, every pair of backslashes is interpreted as a literal backslash. If the number of backslashes is odd, the last backslash escapes the next double quotation mark as described in rule 3.

The shell argument is by default set to False, this argument specifies whether to use the shell as the program to execute. If shell is True, it is recommended to pass args as a string rather than as a sequence.

15.5.15.7 Exceptions in Subprocess

If a child process raises any exception before the new program starts, that exception will be raised again in the parent process. Additionally, the exception object will have one extra attribute called child_traceback, which is a string containing traceback information from the child's point of view.

OSError - This occurs, for example, when trying to execute a non-existent file. Applications should prepare for OSError exceptions.

ValueError - This will be raised if Popen is called with invalid arguments.

CalledProcessError - check_call() and check_output() will raise CalledProcessError if the called process returns a non-zero return code.

15.5.15.8 Security

Its very important for the application to handle security aspect explicitly.

15.5.15.9 Popen Objects

Popen.poll() Check if child process has terminated. Set and return returncode attribute.

Popen.wait() Wait for child process to terminate. Set and return returncode attribute.

```
**Warning This will deadlock when using stdout=PIPE and/or stderr=PIPE and the child process generates enough output to a pipe such that it blocks waiting for the OS pipe buffer to accept more da
```

Popen.communicate(input=None) Interact with process: Send data to stdin. Read data from stdout and stderr, until end-of-file is reached. Wait for process to terminate. The optional input argument should be a string to be sent to the child process, or None, if no data should be sent to the child. communicate() returns a tuple (stdoutdata, stderrdata).

```
**Note that if you want to send data to the process's stdin, you need to create the Popen object with stdin=PIPE. Similarly, to get anything other than None in the result tuple, you need to give

**Note The data read is buffered in memory, so do not use this method if the data size is large or unlimited.
```

Popen.send_signal(signal) Sends the signal signal to the child.

```
**Note On Windows, SIGTERM is an alias for terminate(). CTRL_C_EVENT and CTRL_BREAK_EVENT can be sent to processes started with a creationflags parameter which includes CREATE_NEW_PROCESS_GROUP.
```

New in version 2.6.

Popen.terminate() Stop the child. On Posix OSs the method sends SIGTERM to the child. On Windows the Win32 API function TerminateProcess() is called to stop the child.

New in version 2.6.

Popen.kill() Kills the child. On Posix OSs the function sends SIGKILL to the child. On Windows kill() is an alias for terminate().

New in version 2.6.

The following attributes are also available:

```
**Warning Use communicate() rather than .stdin.write, .stdout.read or .stderr.read to avoid deadlocks due to any of the other OS pipe buffers filling up and blocking the child process.
```

Popen.stdin If the stdin argument was PIPE, this attribute is a file object that provides input to the child process. Otherwise, it is None.

Popen.stdout If the stdout argument was PIPE, this attribute is a file object that provides output from the child process. Otherwise, it is None.

Popen.stderr If the stderr argument was PIPE, this attribute is a file object that provides error output from the child process. Otherwise, it is None.

Popen.pid The process ID of the child process.

```
**Note that if you set the shell argument to True, this is the process ID of the spawned shell.
```

Popen.returncode The child return code, set by poll() and wait() (and indirectly by communicate()). A None value indicates that the process hasn't terminated yet.

A negative value -N indicates that the child was terminated by signal N (Unix only).

15.5.16 Queue

⭕ Students can contribute this section

see: * https://docs.python.org/3/library/queue.html

15.5.17 Scheduler

⭕ Students can contribute this section

see: * https://docs.python.org/3/library/sched.html

15.5.18 Python SSL

⭕ Students can contribute this section

see:

- https://docs.python.org/3/library/ssl.html
- also demonstrate how you could just use supprocess ... to contarst

15.6 Python Modules

Often you may need functionality that is not present in Python's standard library. In this case you have two option:

- implement the features yourself
- use a third-party library that has the desired features.

Often you can find a previous implementation of what you need. Since this is a common situation, there is a service supporting it: the Python Package Index (or PyPi for short).

Our task here is to install the autopep8 tool from PyPi. This will allow us to illustrate the use if virtual environments using the pyenv or virtualenv command, and installing and uninstalling PyPi packages using pip.

15.6.1 Updating Pip

It is important that you have the newest version of pip installed for your version of python. Let us assume your python is registered with python and you use pyenv, than you can update pip with

```
pip install -U pip
```

without interfering with a potential system wide installed version of p ip that may be needed by the system default version of python. See the section about pyenv for more details

15.6.2 Using pip to Install Packages

Let's now look at another important tool for Python development: the Python Package Index, or PyPI for short. PyPI provides a large set of third-party python packages. If you want to do something in python, first check pypi, as odd are someone already ran into the problem and created a package solving it.

In order to install package from PyPI, use the pip command. We can search for PyPI for packages:

```
$ pip search --trusted-host pypi.python.org autopep8 pylint
```

It appears that the top two results are what we want so install them:

```
$ pip install --trusted-host pypi.python.org autopep8 pylint
```

This will cause pip to download the packages from PyPI, extract them, check their dependencies and install those as needed, then install the requested packages.

You can skip '–trusted-host pypi.python.org' option if you have

patched urllib3 on Python 2.7.9.

15.6.3 GUI

15.6.3.1 GUIZero

Install guizero with the following command:

```
sudo pip install guizero
```

For a comprehensive tutorial on guizero, click here.

15.6.3.2 Kivy

You can install Kivy on macOS as follows:

```
brew install pkg-config sdl2 sdl2_image sdl2_ttf sdl2_mixer gstreamer
pip install -U Cython
pip install kivy
pip install pygame
```

A hello world program for kivy is included in the cloudmesh.robot repository. Which you can fine here

- https://github.com/cloudmesh/cloudmesh.robot/tree/master/projects/kivy

To run the program, please download it or execute it in cloudmesh.robot as follows:

```
cd cloudmesh.robot/projects/kivy
python swim.py
```

To create stand alone packages with kivy, please see:

```
-  https://kivy.org/docs/guide/packaging-osx.html
```

15.6.4 Formatting and Checking Python Code

First, get the bad code:

```
$ wget --no-check-certificate http://git.io/pXqb -O bad_code_example.py
```

Examine the code:

```
$ emacs bad_code_example.py
```

As you can see, this is very dense and hard to read. Cleaning it up by hand would be a time-consuming and error-prone process. Luckily, this is a common problem so there exist a couple packages to help in this situation.

15.6.5 Using autopep8

We can now run the bad code through autopep8 to fix formatting problems:

```
$ autopep8 bad_code_example.py >code_example_autopep8.py
```

Let us look at the result. This is considerably better than before. It is easy to tell what the example1 and example2 functions are doing.

It is a good idea to develop a habit of using autopep8 in your python-development workflow. For instance: use autopep8 to check a file, and if it passes, make any changes in place using the -i flag:

```
$ autopep8 file.py    # check output to see of passes
$ autopep8 -i file.py # update in place
```

If you use pyCharm you have the ability to use a similar function while pressing on Inspect Code.

15.6.6 Writing Python 3 Compatible Code

To write python 2 and 3 compatible code we recommend that you take a look at: http://python-future.org/compatible_idioms.html

15.6.7 Using Python on FutureSystems

This is only important if you use Futuresystems resources.

In order to use Python you must log into your FutureSystems account. Then at the shell prompt execute the following command:

```
$ module load python
```

This will make the python and virtualenv commands available to you.

The details of what the module load command does are described in the future lesson modules.

15.6.8 Ecosystem

15.6.8.1 pypi

The Python Package Index is a large repository of software for the Python programming language containing a large number of packages, many of which can be found on pypi. The nice thing about pypi is that many packages can be installed with the program 'pip'.

To do so you have to locate the <package_name> for example with the search function in pypi and say on the commandline:

```
$ pip install <package_name>
```

where `package_name` is the string name of the package. an example would be the package called cloudmesh_client which you can install with:

```
$ pip install cloudmesh_client
```

If all goes well the package will be installed.

15.6.8.2 Alternative Installations

The basic installation of python is provided by python.org. However others claim to have alternative environments that allow you to install python. This includes

- Canopy
- Anaconda
- IronPython

Typically they include not only the python compiler but also several useful packages. It is fine to use such environments for the class, but it should be noted that in both cases not every python library may be available for install in the given environment. For example if you need to use cloudmesh client, it may not be available as conda or Canopy package. This is also the case for many other cloud related and useful python libraries. Hence, we do recommend that if you are new to python to use the distribution form python.org, and use pip and virtualenv.

Additionally some python version have platform specific libraries or dependencies. For example coca libraries, `.NET` or other frameworks are examples. For the assignments and the projects such platform dependent libraries are not to be used.

If however you can write a platform independent code that works on Linux, macOS and Windows while using the python.org version but develop it with any of the other tools that is just fine. However it is up to you to guarantee that this independence is maintained and implemented. You do have to write requirements.txt files that will install the necessary python libraries in a platform independent fashion. The homework assignment PRG1 has even a requirement to do so.

In order to provide platform independence we have given in the class a minimal python version that we have tested with hundreds of students: python.org. If you use any other version, that is your decision. Additionally some students not only use python.org but have used iPython which is fine too. However this class is not only about python, but also about how to have your code run on any platform. The homework is designed so that you can identify a setup that works for you.

However we have concerns if you for example wanted to use chameleon cloud which we require you to access with cloudmesh. cloudmesh is not available as conda, canopy, or other framework package. Cloudmesh client is available form pypi which is standard and should be supported by the frameworks. We have not tested cloudmesh on any other python version then python.org which is the open source community standard. None of the other versions are standard.

In fact we had students over the summer using canopy on their machines and they got confused as they now had multiple python versions and did not know how to switch between them and activate the correct version. Certainly if you know how to do that, than feel free to use canopy, and if you want to use canopy all this is up to you. However the homework and project requires you to make your program portable to python.org. If you know how to do that even if you use canopy, anaconda, or any other python version that is fine. Graders will test your programs on a python.org installation and not canopy, anaconda, ironpython while using virtualenv. It is obvious why. If you do not know that answer you may want to think about that every time they test a program they need to do a new virtualenv and run vanilla python in it. If we were to run two installs in the same system, this will not work as we do not know if one student will cause a side effect for another. Thus we as instructors do not just have to look at your code but code of hundreds of students with different setups. This is a non scalable solution as every time we test out code from a student we would have to wipe out the OS, install it new, install an new version of whatever python you have elected, become familiar with that version and so on and on. This is the reason why the open source community is using python.org. We follow best practices. Using other versions is not a community best practice, but may work for an individual.

We have however in regards to using other python version additional bonus projects such as

- deploy run and document cloudmesh on ironpython
- deploy run and document cloudmesh on anaconda, develop script to generate a conda package form github
- deploy run and document cloudmesh on canopy, develop script to generate a conda package form github
- deploy run and document cloudmesh on ironpython
- other documentation that would be useful

15.6.9 Resources

If you are unfamiliar with programming in Python, we also refer you to some of the numerous online resources. You may wish to start with Learn Python or the book Learn Python the Hard Way. Other options include Tutorials Point or Code Academy, and the Python wiki page contains a long list of references for learning as well. Additional resources include:

- https://virtualenvwrapper.readthedocs.io
- https://github.com/yyuu/pyenv
- https://amaral.northwestern.edu/resources/guides/pyenv-tutorial
- https://godjango.com/96-django-and-python-3-how-to-setup-pyenv-for-multiple-pythons/
- https://www.accelebrate.com/blog/the-many-faces-of-python-and-how-to-manage-them/
- http://ivory.idyll.org/articles/advanced-swc/
- http://python.net/~goodger/projects/pycon/2007/idiomatic/handout.html
- http://www.youtube.com/watch?v=0vJJlVBVTFg
- http://www.korokithakis.net/tutorials/python/
- http://www.afterhoursprogramming.com/tutorial/Python/Introduction/
- http://www.greenteapress.com/thinkpython/thinkCSpy.pdf
- https://docs.python.org/3.3/tutorial/modules.html
- https://www.learnpython.org/en/Modules/_and/_Packages
- https://docs.python.org/2/library/datetime.html
- https://chrisalbon.com/python/strings/_to/_datetime.html

A very long list of useful information are also available from

- https://github.com/vinta/awesome-python
- https://github.com/rasbt/python_reference

This list may be useful as it also contains links to data visualization and manipulation libraries, and AI tools and libraries. Please note that for this class you can reuse such libraries if not otherwise stated.

15.6.9.1 Jupyter Notebook Tutorials

A Short Introduction to Jupyter Notebooks and NumPy To view the notebook, open this link in a background tab https://nbviewer.jupyter.org/ and copy and paste the following link in the URL input area https://cloudmesh.github.io/classes/lesson/prg/Jupyter-NumPy-tutorial-I523-F2017.ipynb Then hit Go.

15.6.10 Exercises

E.Python.Lib.1:

Write a python program called iterate.py that accepts an integer n from the command line. Pass this integer to a function called iterate.

The iterate function should then iterate from 1 to n. If the i-th number is a multiple of three, print multiple of 3, if a multiple of 5 print multiple of 5, if a multiple of both print multiple of 3 and 5, else print the value.

E:Python.Lib.2:

1. Create a pyenv or virtualenv ~/ENV
2. Modify your ~/.bashrc shell file to activate your environment upon login.
3. Install the docopt python package using pip
4. Write a program that uses docopt to define a commandline program. Hint: modify the iterate program.
5. Demonstrate the program works.

15.6.11 Data Management

Obviously when dealing with big data we may not only be dealing with data in one format but in many different formats. It is important that you will be able to master such formats and seamlessly integrate in your analysis. Thus we provide some simple examples on which different data formats exist and how to use them.

15.6.11.1 Formats

15.6.11.1.1 Pickle

Python pickle allows you to save data in a python native format into a file that can later be read in by other programs. However, the data format may not be portable among different python versions thus the format is often not suitable to store information. Instead we recommend for standard data to use either json or yaml.

```
import pickle

flavor = {
    "small": 100,
    "medium": 1000,
    "large": 10000
    }

pickle.dump( flavor, open( "data.p", "wb" ) )
```

To read it back in use

```
flavor = pickle.load( open( "data.p", "rb" ) )
```

15.6.11.1.2 Text Files

To read text files into a variable called content you can use

```
content = open('filename.txt', 'r').read()
```

You can also use the following code while using the convenient `with` statement

```
with open('filename.txt','r') as file:
    content = file.read()
```

To split up the lines of the file into an array you can do

```
with open('filename.txt','r') as file:
    lines = file.read().splitlines()
```

This cam also be done with the build in `readlines` function

```
lines = open('filename.txt','r').readlines()
```

In case the file is too big you will want to read the file line by line:

```
with open('filename.txt','r') as file:
    line = file.readline()
    print (line)
```

15.6.11.1.3 CSV Files

Often data is contained in comma separated values (CSV) within a file. To read such files you can use the csv package.

```
import csv
with open('data.csv', 'rb') as f:
   contents = csv.reader(f)
for row in content:
    print row
```

Using pandas you can read them as follows.

```
import pandas as pd
df = pd.read_csv("example.csv")
```

There are many other modules and libraries that include CSV read functions. IN case you need to split a single line by comma, you may also use the `split` function. However, remember it swill split at every comma, including those contained in quotes. SO this method although looking originally convenient has limitations.

15.6.11.1.4 Excel spread sheets

Pandas contains a method to read Excel files

```
import pandas as pd
filename = 'data.xlsx'
data = pd.ExcelFile(file)
df = data.parse('Sheet1')
```

15.6.11.1.5 YAML

YAML is a very important format as it allows you easily to structure data in hierarchical fields It is frequently used to coordinate programs while using yaml as the specification for configuration files, but also data files. To read in a yaml file the following code can be used

```
import yaml
with open('data.yaml', 'r') as f:
    content = yaml.load(f)
```

The nice part is that this code can also be used to verify if a file is valid yaml. To write data out we can use

```
with open('data.yml', 'w') as f:
    yaml.dump(data, f, default_flow_style=False)
```

The flow style set to false formats the data in a nice readable fashion with indentations.

15.6.11.1.6 JSON

```
import json
with open('strings.json') as f:
    content = json.load(f)
```

15.6.11.1.7 XML ⭕

Please contribute a XML python section

15.6.11.1.8 RDF

To read RDF files you will need to install RDFlib with

```
$ pip install rdflib
```

This will than allow you to read RDF files

```
from rdflib.graph import Graph
g = Graph()
g.parse("filename.rdf", format="format")
for entry in g:
   print(entry)
```

Good examples on using RDF are provided on the RDFlib Web page at https://github.com/RDFLib/rdflib

From the Web page we showcase also how to directly process RDF data from the Web

```
import rdflib
g=rdflib.Graph()
g.load('http://dbpedia.org/resource/Semantic_Web')

for s,p,o in g:
    print s,p,o
```

15.6.11.1.9 PDF

The Portable Document Format (PDF) has been made available by Adobe Inc. royalty free. This has enabled PDF to become a world wide adopted format that also has been standardized in 2008 (ISO/IEC 32000-1:2008, https://www.iso.org/standard/51502.html). A lot of research is published in papers making PDF one of the de-facto standards for publishing. However, PDF is difficult to parse and is focused on high quality output instead of data representation. Nevertheless, tools to manipulate PDF exist:

PDFMiner

: https://pypi.python.org/pypi/pdfminer/ allows the simple translation of PDF into text that than can be further mined. The manual page helps to demonstrate some examples http://euske.github.io/pdfminer/index.html.

pdf-parser.py

: https://blog.didierstevens.com/programs/pdf-tools/ parses pdf documents and identifies some structural elements that can than be further processed.

If you know about other tools, let us know.

15.6.11.1.10 HTML

A very powerful library to parse HTML Web pages is provided with https://www.crummy.com/software/BeautifulSoup/

More details about it are provided in the documentation page https://www.crummy.com/software/BeautifulSoup/bs4/doc/

⭕ TODO: Students can contribute a section

15.6.11.1.11 ConfigParser

⭕ TODO: Students can contribute a section

- https://pymotw.com/2/ConfigParser/

15.6.11.1.12 ConfigDict

- https://github.com/cloudmesh/cloudmesh.common/blob/master/cloudmesh/common/ConfigDict.py

15.6.11.2 Encryption

Often we need to protect the information stored in a file. This is achieved with encryption. There are many methods of supporting encryption and even if a file is encrypted it may be target to attacks. Thus it is not only important to encrypt data that you do not want others to se but also to make sure that the system on which the data is hosted is secure. This is especially important if we talk about big data having a potential large effect if it gets into the wrong hands.

To illustrate one type of encryption that is non trivial we have chosen to demonstrate how to encrypt a file with an ssh key. In case you have openssl installed on your system, this can be achieved as follows.

```
    #! /bin/sh

    # Step 1. Creating a file with data
    echo "Big Data is the future." > file.txt

    # Step 2. Create the pem
    openssl rsa -in ~/.ssh/id_rsa -pubout  > ~/.ssh/id_rsa.pub.pem

    # Step 3. look at the pem file to illustrate how it looks like (optional)
    cat ~/.ssh/id_rsa.pub.pem

    # Step 4. encrypt the file into secret.txt
    openssl rsautl -encrypt -pubin -inkey ~/.ssh/id_rsa.pub.pem -in file.txt -out secret.txt

    # Step 5. decrypt the file and print the contents to stdout
    openssl rsautl -decrypt -inkey ~/.ssh/id_rsa -in secret.txt
```

Most important here are Step 4 that encrypts the file and Step 5 that decrypts the file. Using the Python os module it is straight forward to implement this. However, we are providing in cloudmesh a convenient class that makes the use in python very simple.

```
from cloudmesh.common.ssh.encrypt import EncryptFile

e = EncryptFile('file.txt', 'secret.txt')
e.encrypt()
e.decrypt()
```

In our class we initialize it with the locations of the file that is to be encrypted and decrypted. To initiate that action just call the methods `encrypt` and `decrypt`.

15.6.11.3 Database Access

⭕ TODO: Students: define conventional database access section

see: https://www.tutorialspoint.com/python/python_database_access.htm

15.6.11.4 SQLite

;o: TODO: Students can contribute to this section

https://www.sqlite.org/index.html

https://docs.python.org/3/library/sqlite3.html

15.6.11.4.1 Exercises ⭘

E:Encryption.1:

> Test the shell script to replicate how this example works

E:Encryption.2:

Test the cloudmesh encryption class

E:Encryption.3:

What other encryption methods exist. Can you provide an example and contribute to the section?

E:Encryption.4:

> What is the issue of encryption that make it challenging for Big Data

E:Encryption.5:

> Given a test dataset with many files text files, how long will it take to encrypt and decrypt them on various machines. Write a benchmark that you test. Develop this benchmark as a group, test out the time it takes to execute it on a variety of platforms.

15.6.12 Plotting with matplotlib

A brief overview of plotting with matplotlib along with examples is provided. First matplotlib must be installed, which can be accomplished with pip install as follows:

```
pip install matplotlib
```

We will start by plotting a simple line graph using built in numpy functions for sine and cosine. This first step is to import the proper libraries shown below.

```
import numpy as np
import matplotlib.pyplot as plt
```

Next we will define the values for the x axis, we do this with the linspace option in numpy. The first two parameters are the starting and ending points, these must be scalars. The third parameter is optional and defines the number of samples to be generated between the starting and ending points, this value must be an integer. Additional parameters for the linspace utility can be found here:

```
x = np.linspace(-np.pi, np.pi, 16)
```

Now we will use the sine and cosine functions in order to generate y values, for this we will use the values of x for the argument of both our sine and cosine functions i.e. $cos(x)$.

```
cos = np.cos(x)
sin = np.sin(x)
```

You can display the values of the three parameters we have defined by typing them in a python shell.

```
x
array([-3.14159265, -2.72271363, -2.30383461, -1.88495559, -1.46607657,
    -1.04719755, -0.62831853, -0.20943951, 0.20943951, 0.62831853,
    1.04719755, 1.46607657, 1.88495559, 2.30383461, 2.72271363,
    3.14159265])
```

Having defined x and y values we can generate a line plot and since we imported matplotlib.pyplot as plt we simply use plt.plot.

```
plt.plot(x,cos)
```

We can display the plot using plt.show() which will pop up a figure displaying the plot defined.

```
plt.show()
```

Additionally we can add the sine line to out line graph by entering the following.

```
plt.plot(x,sin)
```

Invoking plt.show() now will show a figure with both sine and cosine lines displayed. Now that we have a figure generated it would be useful to label the x and y axis and provide a title. This is done by the following three commands below:

```
plt.xlabel("X - label (units)")
plt.ylabel("Y - label (units)")
plt.title("A clever Title for your Figure")
```

Along with axis labels and a title another useful figure feature may be a legend. In order to create a legend you must first designate a label for the line, this label will be what shows up in the legend. The label is defined in the initial plt.plot(x,y) instance, below is an example.

```
plt.plot(x,cos, label="cosine")
```

Then in order to display the legend the following command is issued:

```
plt.legend(loc='upper right')
```

The location is specified by using upper or lower and left or right. Naturally all these commands can be combined and put in a file with the .py extension and run from the command line.

```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-np.pi, np.pi, 16)
cos = np.cos(x)
sin = np.sin(x)
plt.plot(x,cos, label="cosine")
plt.plot(x,sin, label="sine")

plt.xlabel("X - label (units)")
plt.ylabel("Y - label (units)")
plt.title("A clever Title for your Figure")

plt.legend(loc='upper right')

plt.show()
```

⭘ link error

An example of a bar chart is preceded below using data from [T:fast-cars].

```
import matplotlib.pyplot as plt

x = [' Toyota Prius', 'Tesla Roadster ', ' Bugatti Veyron', ' Honda Civic ', ' Lamborghini Aventador ']
horse_power = [120, 288, 1200, 158, 695]

x_pos = [i for i, _ in enumerate(x)]

plt.bar(x_pos, horse_power, color='green')
plt.xlabel("Car Model")
plt.ylabel("Horse Power (Hp)")
plt.title("Horse Power for Selected Cars")

plt.xticks(x_pos, x)

plt.show()
```

You can customize plots further by using plt.style.use(), in python 3. If you provide the following command inside a python command shell you will see a list of available styles.

```
print(plt.style.available)
```

An example of using a predefined style is shown below.

```
plt.style.use('seaborn')
```

Up to this point we have only showcased how to display figures through python output, however web browsers are a popular way to display figures. One example is Bokeh, the following lines can be entered in a python shell and the figure is outputted to a browser.

```
from bokeh.io import show
from bokeh.plotting import figure

x_values = [1, 2, 3, 4, 5]
y_values = [6, 7, 2, 3, 6]

p = figure()
p.circle(x=x_values, y=y_values)
show(p)
```

15.6.13 DocOpts

When we want to design commandline arguments for python programs we have many options. However, as our approach is to create documentation first, docopts provides also a good apprach for Python. The code for it is located at

- https://github.com/docopt/docopt

It can be installed with

```
$ pip install docopt
```

A sample programs are located at

- https://github.com/docopt/docopt/blob/master/examples/options_example.py

A sample program of using doc opts for our purposes loks as follows

```
"""Cloudmesh VM management

Usage:
  cm-go vm start NAME [--cloud=CLOUD]
  cm-go vm stop NAME [--cloud=CLOUD]
  cm-go set --cloud=CLOUD
  cm-go -h | --help
  cm-go --version

Options:
  -h --help     Show this screen.
  --version     Show version.
  --cloud=CLOUD  The name of the cloud.
  --moored      Moored (anchored) mine.
  --drifting    Drifting mine.

ARGUMENTS:
  NAME      The name of the VM`
"""
from docopt import docopt

if __name__ == '__main__':
    arguments = docopt(__doc__, version='1.0.0rc2')
    print(arguments)
```

Another good feature of using docopts is that we can use the same verbal description in other programming languages as showcased in this book.

15.6.14 Cloudmesh Command Shell

15.6.14.1 CMD5

Python's CMD (https://docs.python.org/2/library/cmd.html) is a very useful package to create command line shells. However it does not allow the dynamic integration of newly defined commands. Furthermore, additions to CMD need to be done within the same source tree. To simplify developing commands by a number of people and to have a dynamic plugin mechanism, we developed cmd5. It is a rewrite on our earlier efforts in cloudmesh client and cmd3.

15.6.14.1.1 Resources

The source code for cmd5 is located in github:

- https://github.com/cloudmesh/cmd5

15.6.14.1.2 Creating a Python Development Environment

We recommend that you use a virtualenv either with virtualenv or pyenv. This is in detail documented in the Section [S:managing-multiple-python-versions-with-pyenv].

15.6.14.1.3 Installation from source

Cmd5 can be easily deployed with pip:

```
$ pip install cloudmesh.cmd5
```

In case you would like to generate easily new cmd5 commands we also recommend you install the cloudmesh sys command with:

```
$ pip install cloudmesh.sys
```

In case you like to work with the source please clone the following directories from github:

```
mkdir -p ~/github
cd ~/github

git clone https://github.com/cloudmesh/cloudmesh.common.git
git clone https://github.com/cloudmesh/cloudmesh.cmd5.git
git clone https://github.com/cloudmesh/cloudmesh.sys.git

cd ~/github/cloudmesh.common
python setup.py install
pip install .

cd ~/github/cloudmesh.cmd5
python setup.py install
pip install .

cd ~/github/cloudmesh.sys
python setup.py install
pip install .
```

The common directory contains some useful libraries, the cmd5 repository >contains the shell, while the sys directory contains a command to generate extensions to cloudmesh.

15.6.14.1.4 Execution

To run the shell you can activate it with the cms command. cms stands for cloudmesh shell:

```
(ENV2) $ cms
```

It will print the banner and enter the shell:

```
+-------------------------------------------------------------+
|  ____ _                 _                           _       |
| / ___| | ___  _   _  __| |_ __ ___   ___  ___| |__    |
| | |   | |/ _ \| | | |/ _` | '_ ` _ \ / _ \/ __| '_ \  |
| | |___| | (_) | |_| | (_| | | | | | |  __/\__ \ | | | |
|  \____|_|\___/ \__,_|\__,_|_| |_| |_|\___||___/_| |_| |
+-------------------------------------------------------------+
|                  Cloudmesh CMD5 Shell                       |
+-------------------------------------------------------------+

cms>
```

To see the list of commands you can say:

```
cms> help
```

To see the manual page for a specific command, please use:

```
help COMMANDNAME
```

15.6.14.1.5 Create your own Extension

One of the most important features of CMD5 is its ability to extend it with new commands. This is done via packaged name spaces. We recommend you name is cloudmesh.mycommand, where mycommand is the name of the command that you like to create. This can easily be done while using the sys command:

```
$ cms sys command generate mycommand
```

It will download a template from cloudmesh called cloudmesh.bar and generate a new directory cloudmesh.mycommand with all the needed files to create your own command and register it dynamically with cloudmesh. All you have to do is to cd into the directory and install the code:

```
$ cd cloudmesh.mycommand
$ python setup.py install
# pip install .
```

Adding your own command is easy. It is important that all objects are defined in the command itself and that no global variables be use in order to allow each shell command to stand alone. Naturally you should develop API libraries outside of the cloudmesh shell command and reuse them in order to keep the command code as small as possible. We place the command in:

```
cloudmsesh/mycommand/command/mycommand.py
```

An example for the bar command is presented at:

- https://github.com/cloudmesh/cloudmesh.bar/blob/master/cloudmesh/bar/command/bar.py

It shows how simple the command definition is (bar.py):

```
from __future__ import print_function
from cloudmesh.shell.command import command
from cloudmesh.shell.command import PluginCommand


class BarCommand(PluginCommand):

    @command
    def do_bar(self, args, arguments):
        """
        ::
          Usage:
                command -f FILE
                command FILE
                command list
          This command does some useful things.
          Arguments:
              FILE   a file name
          Options:
              -f      specify the file
        """
        print(arguments)
```

An important difference to other CMD solutions is that our commands can leverage (besides the standard definition), docopts as a way to define the manual page. This allows us to use arguments as dict and use simple if conditions to interpret the command. Using docopts has the advantage that contributors are forced to think about the command and its options and document them from the start. Previously we did not use but argparse and click. However we noticed that for our contributors both systems lead to commands that were either not properly documented or the developers delivered ambiguous commands that resulted in confusion and wrong usage by subsequent users. Hence, we do recommend that you use docopts for documenting cmd5 commands. The transformation is enabled by the @command decorator that generates a manual page and creates a proper help message for the shell automatically. Thus there is no need to introduce a separate help method as would normally be needed in CMD while reducing the effort it takes to contribute new commands in a dynamic fashion.

15.6.14.1.6 Exercises

E.CMD5.1

Install cmd5 on your computer.

E. CMD5.2

Write a new command with your firstname as the command name.

E.CMD5.3

Write a new command and experiment with docopt syntax and argument interpretation of the dict with if conditions.

E.CMD5.4

If you have useful extensions that you like us to add by default, please work with us.

E.CMD5.5

At this time one needs to quote in some commands the " in the shell command line. Develop and test code that fixes this.

15.6.15 cmd Module

If you consider using this module, you may instead want to use cloudmesh cmd5 instead as it provides some very nice features that are not included in cmd. However to do the basics, cmd will do.

The Python cmd module is useful for any more involved command-line application. It is used in the Cloudmesh Project, for example, and students have found it helpful in their projects to develop quickly high quality command line tools with documentation so that others can replicate and use the programs. The Python cmd module contains a public class, Cmd, designed to be used as a base class for command processors such as interactive shells and other command interpreters.

15.6.15.1 Hello, World with cmd

This example shows a very simple command interpreter that simply responds to the greet command.

In order to demonstrate commands provided by cmd, let's save the following program in a file called helloworld.py.

```
from __future__ import print_function, division
import cmd


class HelloWorld(cmd.Cmd):
    '''Simple command processor example.'''

    def do_greet(self, line):
        if line is not None and len(line.strip()) > 0:
            print('Hello, %s!' % line.strip().title())
        else:
            print('Hello!')
```

```
    def do_EOF(self, line):
        print('bye, bye')
        return True


if __name__ == '__main__':
    HelloWorld().cmdloop()
```

A session with this program might look like this:

```
$ python helloworld.py

(Cmd) help

Documented commands (type help <topic>):
========================================
help

Undocumented commands:
======================
EOF  greet

(Cmd) greet
Hello!
(Cmd) greet albert
Hello, Albert!
<CTRL-D pressed>
(Cmd) bye, bye
```

The Cmd class can be used to customize a subclass that becomes a user-defined command prompt. After you have executed your program, commands defined in your class can be used. Take note of the following in this example:

- The methods of the class of the form do_xxx implement the shell commands, with xxx being the name of the command. For example, in the `HelloWorld` class, the function do_greet maps to the greet on the command line.
- The EOF command is a special command that is executed when you press CTRL-D on your keyboard.
- As soon as any command method returns True the shell application exits. Thus, in this example the shell is exited by pressing CTRL-D, since the do_EOF method is the only one that returns True.
- The shell application is started by calling the cmdloop method of the class.

15.6.15.2 A More Involved Example

Let's look at a little more involved example. Save the following code in a file called calculator.py.

```
from __future__ import print_function, division
import cmd


class Calculator(cmd.Cmd):
 prompt = 'calc >>> '
 intro = 'Simple calculator that can do addition, subtraction, multiplication and division.'

 def do_add(self, line):
     args = line.split()
     total = 0
     for arg in args:
         total += float(arg.strip())
     print(total)

 def do_subtract(self, line):
     args = line.split()
     total = 0
     if len(args) > 0:
         total = float(args[0])
     for arg in args[1:]:
         total -= float(arg.strip())
     print(total)

 def do_EOF(self, line):
     print('bye, bye')
     return True


if __name__ == '__main__':
 Calculator().cmdloop()
```

A session with this program might look like this:

```
$ python calculator.py
Simple calculator that can do addition, subtraction, multiplication and division.
calc >>> help

Documented commands (type help <topic>):
========================================
help

Undocumented commands:
======================
EOF  add  subtract

calc >>> add
0
calc >>> add 4 5 6
15.0
calc >>> subtract
0
calc >>> subtract 10 2
8.0
calc >>> subtract 10 2 20
-12.0
calc >>> bye, bye
```

In this case we are using the prompt and intro class variables to define what the default prompt looks like and a welcome message when the command interpreter is invoked.

In the `add` and `subtract` commands we are using the strip and split methods to parse all arguments. If you want to get fancy, you can use Python modules like getopts or argparse for this, but this is not necessary in this simple example.

15.6.15.3 Help Messages

Notice that all commands presently show up as undocumented. To remedy this, we can define help_ methods for each command:

```
from __future__ import print_function, division
import cmd


class Calculator(cmd.Cmd):
  prompt = 'calc >>> '
  intro = 'Simple calculator that can do addition, subtraction, multiplication and division.'

  def do_add(self, line):
      args = line.split()
```

```
        total = 0
        for arg in args:
            total += float(arg.strip())
        print(total)

    def help_add(self):
        print('\n'.join([
            'add [number,]',
            'Add the arguments together and display the total.'
        ]))

    def do_subtract(self, line):
        args = line.split()
        total = 0
        if len(args) > 0:
            total = float(args[0])
        for arg in args[1:]:
            total -= float(arg.strip())
        print(total)

    def help_subtract(self):
        print('\n'.join([
            'subtract [number,]',
            'Subtract all following arguments from the first argument.'
        ]))

    def do_EOF(self, line):
        print('bye, bye')
        return True


if __name__ == '__main__':
    Calculator().cmdloop()
```

Now, we can obtain help for the add and subtract commands:

```
$ python calculator.py
Simple calculator that can do addition, subtraction, multiplication and division.
calc >>> help

Documented commands (type help <topic>):
========================================
add  help  subtract

Undocumented commands:
======================
EOF

calc >>> help add
add [number,]
Add the arguments together and display the total.
calc >>> help subtract
subtract [number,]
Subtract all following arguments from the first argument.
calc >>> bye, bye
```

15.6.15.4 Useful Links

- cms Python 2 Docs
- cmd Python 3 Docs
- Python Module of the Week: cmd – Create line-oriented command processors
- Python Module of the Week: cmd – Create line-oriented command processors

15.6.16 OpenCV

Learning Objectives

- Provide some simple calculations so we can test cloud services.
- Show case some elementary OpenCV functions
- Show an environmental image analysis application using Secchi disks

OpenCV (Open Source Computer Vision Library) is a library of thousands of algorithms for various applications in computer vision and machine learning. It has C++, C, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. In this section, we will explain basic features of this library, including the implementation of a simple example.

15.6.16.1 Overview

OpenCV has countless functions for image and videos processing. The pipeline starts with reading the images, low-level operations on pixel values, preprocessing e.g. denoising, and then multiple steps of higher-level operations which vary depending on the application. OpenCV covers the whole pipeline, especially providing a large set of library functions for high-level operations. A simpler library for image processing in Python is Scipy's multi-dimensional image processing package (scipy.ndimage).

15.6.16.2 Installation

OpenCV for Python can be installed on Linux in multiple ways, namely PyPI(Python Package Index), Linux package manager (apt-get for Ubuntu), Conda package manager, and also building from source. You are recommended to use PyPI. Here's the command that you need to run:

```
$ pip install opencv-python
```

This was tested on Ubuntu 16.04 with a fresh Python 3.6 virtual environment. In order to test, import the module in Python command line:

```
import cv2
```

If it does not raise an error, it is installed correctly. Otherwise, try to solve the error.

For installation on Windows, see:

- https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_setup/py_setup_in_windows/py_setup_in_windows.html#install-opencv-python-in-windows

Note that building from source can take a long time and may not be feasible for deploying to limited platforms such as Raspberry Pi.

15.6.16.3 A Simple Example

In this example, an image is loaded. A simple processing is performed, and the result is written to a new image.

15.6.16.3.1 Loading an image

```
%matplotlib inline
import cv2

img = cv2.imread('images/opencv/4.2.01.tiff')
```

The image was downloaded from USC standard database:

http://sipi.usc.edu/database/database.php?volume=misc&image=9

15.6.16.3.2 Displaying the image

The image is saved in a numpy array. Each pixel is represented with 3 values (R,G,B). This provides you with access to manipulate the image at the level of single pixels. You can display the image using imshow function as well as Matplotlib's imshow function.

You can display the image using imshow function:

```
cv2.imshow('Original',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

or you can use Matplotlib. If you have not installed Matplotlib before, install it using:

```
$ pip install matplotlib
```

Now you can use:

```
import matplotlib.pyplot as plt
plt.imshow(img)
```

which results in

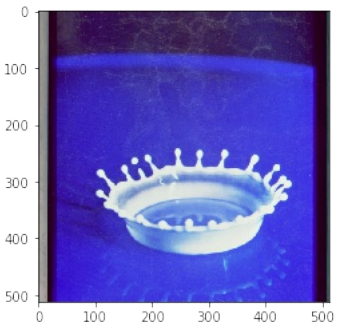


Figure: Image display

15.6.16.3.3 Scaling and Rotation

Scaling (resizing) the image relative to different axis

```
res = cv2.resize(img,
                 None,
                 fx=1.2,
                 fy=0.7,
                 interpolation=cv2.INTER_CUBIC)
plt.imshow(res)
```

which results in

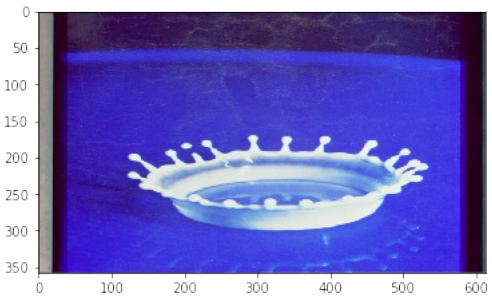
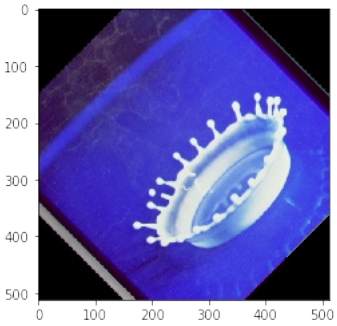


Figure: Scaling and rotation

Rotation of the image for an angle of t

```
rows,cols,_ = img.shape
t = 45
M = cv2.getRotationMatrix2D((cols/2,rows/2),t,1)
dst = cv2.warpAffine(img,M,(cols,rows))

plt.imshow(dst)
```

which results in



image

15.6.16.3.4 Gray-scaling

```
img2 = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
plt.imshow(img2, cmap='gray')
```

which results in

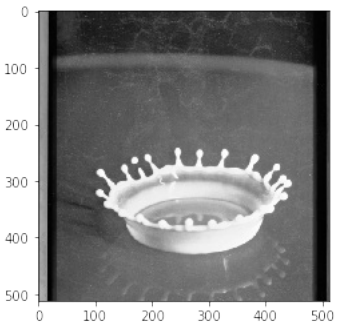


Figure: Gray sacling

15.6.16.3.5 Image Thresholding

```
ret,thresh =    cv2.threshold(img2,127,255,cv2.THRESH_BINARY)
plt.subplot(1,2,1), plt.imshow(img2, cmap='gray')
plt.subplot(1,2,2), plt.imshow(thresh, cmap='gray')
```

which results in

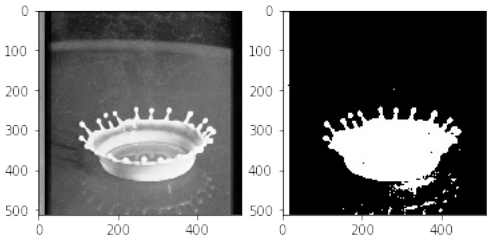


Figure: Image Thresholding

15.6.16.3.6 Edge Detection

Edge detection using Canny edge detection algorithm

```
edges = cv2.Canny(img2,100,200)

plt.subplot(121),plt.imshow(img2,cmap = 'gray')
plt.subplot(122),plt.imshow(edges,cmap = 'gray')
```

which results in

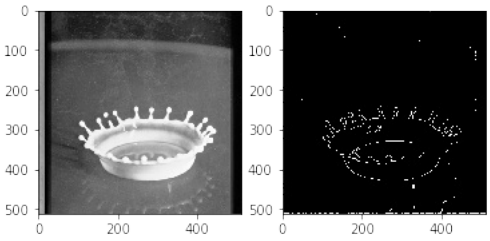


Figure: Edge detection

15.6.16.4 Additional Features

OpenCV has implementations of many machine learning techniques such as KMeans and Support Vector Machines, that can be put into use with only a few lines of code. It also has functions especially for video analysis, feature detection, object recognition and many more. You can find out more about them in their website

[OpenCV](https://docs.opencv.org/3.0-beta/index.html was initially developed for C++ and still has a focus on that language, but it is still one of the most valuable image processing libraries in Python.

15.6.17 Secchi Disk

We are developing an autonomous robot boat that you can be part of developing within this class. The robot bot is actually measuring turbidity or water clarity. Traditionally this has been done with a Secchi disk. The use of the Secchi disk is as follows:

1. Lower the Secchi disk into the water.
2. Measure the point when you can no longer see it
3. Record the depth at various levels and plot in a geographical 3D map

One of the things we can do is take a video of the measurement instead of a human recording them. Than we can analyse the video automatically to see how deep a disk was lowered. This is a classical image analysis program. You are encouraged to identify algorithms that can identify the depth. The most simplest seems to be to do a histogram at a variety of depth steps, and measure when the histogram no longer changes significantly. The depth at that image will be the measurement we look for.

Thus if we analyse the images we need to look at the image and identify the numbers on the measuring tape, as well as the visibility of the disk.

To show case how such a disk looks like we refer to the image showcasing different Secchi disks. For our purpose the black-white contrast Secchi disk works well.

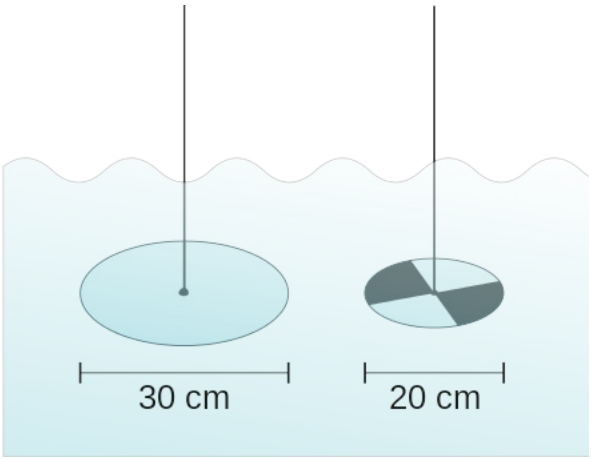


Figure: Secchi disk types. A marine style on the left and the freshwater version on the right wikipedia.

More information about Secchi Disk can be found at:

- https://en.wikipedia.org/wiki/Secchi_ disk

We have included next a couple of examples while using some obviously useful OpenCV methods. Surprisingly, the use of the edge detection that comes in mind first to identify if we still can see the disk, seems to complicated to use for analysis. We at this time believe the histogram will be sufficient.

Please inspect our examples.

15.6.17.1 Setup for OSX

First lest setup the OpenCV environment for OSX. Naturally you will have to update the versions based on your versions of python. When we tried the install of OpenCV on MacOS, the setup was slightly more complex than other packages. This may have changed by now and if you have improved instructions, pleas elt us know. However we do not want to install it via Anaconda out of the obvious reason that anaconda installs to many other things.

```
import os, sys
from os.path import expanduser
os.path
home = expanduser("~")
sys.path.append('/usr/local/Cellar/opencv/3.3.1_1/lib/python3.6/site-packages/')
sys.path.append(home + '/.pyenv/versions/OPENCV/lib/python3.6/site-packages/')
import cv2
cv2.__version__
! pip install numpy > tmp.log
! pip install matplotlib >> tmp.log
%matplotlib inline
```

15.6.17.2 Step 1: Record the video

Record the video on the robot

We have actually done this for you and will provide you with images and videos if you are interested in analyzing them.

15.6.17.3 Step 2: Analyse the images from the Video

For now we just selected 4 images from the video

```
import cv2
import matplotlib.pyplot as plt

img1 = cv2.imread('secchi/secchi1.png')
img2 = cv2.imread('secchi/secchi2.png')
img3 = cv2.imread('secchi/secchi3.png')
img4 = cv2.imread('secchi/secchi4.png')

figures = []
fig = plt.figure(figsize=(18, 16))
for i in range(1,13):
    figures.append(fig.add_subplot(4,3,i))
count = 0
for img in [img1,img2,img3,img4]:
    figures[count].imshow(img)

    color = ('b','g','r')
    for i,col in enumerate(color):
        histr = cv2.calcHist([img],[i],None,[256],[0,256])
        figures[count+1].plot(histr,color = col)

    figures[count+2].hist(img.ravel(),256,[0,256])

    count += 3

print("Legend")
print("First column = image of Secchi disk")
print("Second column = histogram of colors in image")
print("Third column = histogram of all values")

plt.show()
```

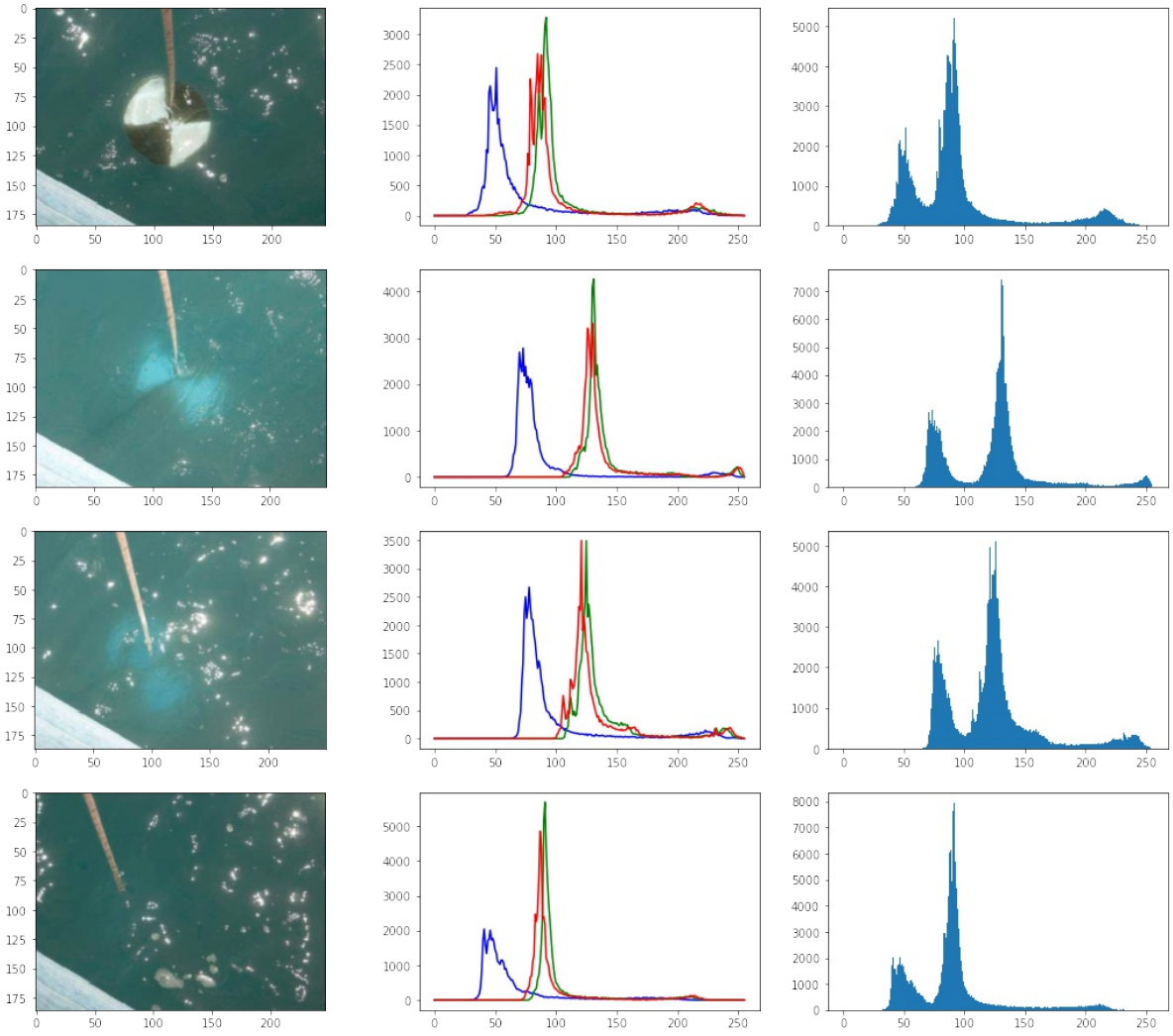


Figure: Histogram

15.6.17.3.1 Image Thresholding

```
def threshold(img):
    ret,thresh = cv2.threshold(img,150,255,cv2.THRESH_BINARY)
    plt.subplot(1,2,1), plt.imshow(img, cmap='gray')
    plt.subplot(1,2,2), plt.imshow(thresh, cmap='gray')

threshold(img1)
```

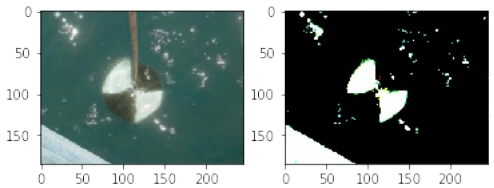


Figure: Threshold 1

```
threshold(img2)
```

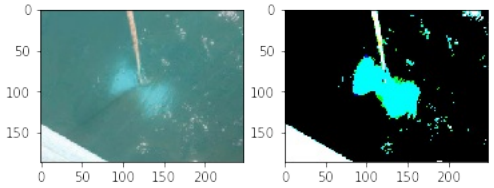


Figure: Threshold 2

```
threshold(img3)
```

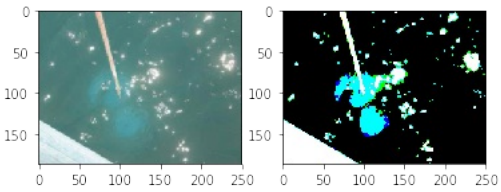


Figure: Threshold 3

```
threshold(img4)
```

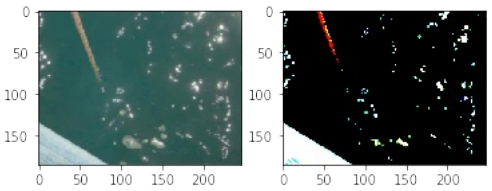


Figure: Threshold 4

15.6.17.3.2 Edge Detection

Edge detection using Canny edge detection algorithm

```
def find_edge(img):
    edges = cv2.Canny(img,50,200)
    plt.subplot(121),plt.imshow(img,cmap = 'gray')
    plt.subplot(122),plt.imshow(edges,cmap = 'gray')

find_edge(img1)
```

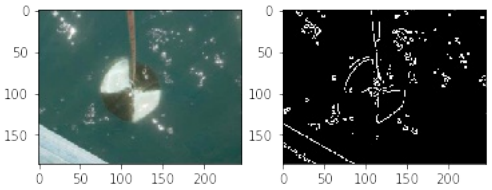


Figure: Edge Detection 1

```
find_edge(img2)
```

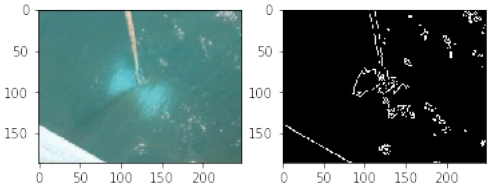


Figure: Edge Detection 2

```
find_edge(img3)
```

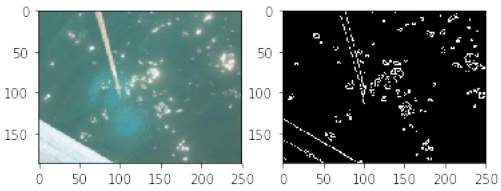


Figure: Edge Detection 3

```
find_edge(img4)
```

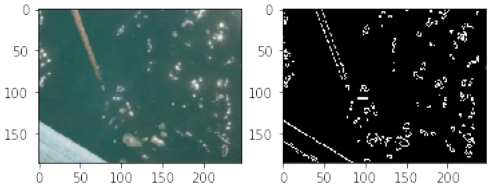


Figure: Edge Detection 4

15.6.17.3.3 Black and white

```
bw1 = cv2.cvtColor(img1, cv2.COLOR_BGR2GRAY)
plt.imshow(bw1, cmap='gray')
```

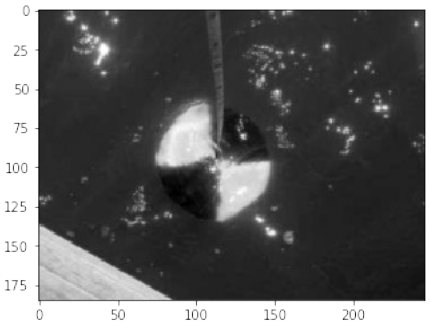


Figure: Back White conversion

15.6.18 Data Libraries

15.6.18.1 Data Formats

15.6.18.1.1 YAML

The term YAML stand for "YAML Ainot Markup Language". According to the Web Page at

- http://yaml.org/

"YAML is a human friendly data serialization standard for all programming languages." There are multiple versions of YAML existing and one needs to take care of that your software supports the right version. The current version is YAML 1.2.

YAML is often used for configuration and in many cases can also be used as XML replacement. Important is tat YAM in contrast to XML removes the tags while replacing them with indentation. This has naturally the advantage that it is mor easily to read, however, the format is strict and needs to adhere to proper indentation. Thus it is important that you check your YAML files for correctness, either by writing for example a python program that read your yaml file, or an online YAML checker such as provided at

- http://www.yamllint.com/

An example on how to use yaml in python is provided in our next example. Please note that YAML is a superset of JSON. Originally YAML was designed as a markup language. However as it is not document oriented but data oriented it has been recast and it does no longer classify itself as markup language.

```
import os
import sys
import yaml

try:
    yamlFilename = os.sys.argv[1]
    yamlFile = open(yamlFilename, "r")
except:
    print("filename does not exist")
    sys.exit()
try:
   yaml.load(yamlFile.read())
except:
   print("YAML file is not valid.")
```

Resources:

- http://yaml.org/
- https://en.wikipedia.org/wiki/YAML
- http://www.yamllint.com/

15.6.18.1.2 JSON

The term JSON stand for JavaScript Object Notation. It is targeted as an open-standard file format that emphasizes on integration of human-readable text to transmit data objects. The data objects contain attribute value pairs. Although it originates from JavaScript, the format itself is language independent. It uses brackets to allow organization of the data. PLease note that YAML is a superset of JSON and not all YAML documents can be converted to JSON. Furthermore JSON does not support comments. For these reasons we often prefer to us YAMl instead of JSON. However JSON data can easily be translated to YAML as well as XML.

Resources:

- https://en.wikipedia.org/wiki/JSON
- https://www.json.org/

15.6.18.1.3 XML

XML stands for Extensible Markup Language. XML allows to define documents with the help of a set of rules in order to make it machine readable. The emphasize here is on machine readable as document in XML can become quickly complex and difficult to understand for humans. XML is used for documents as well as data structures.

A tutorial about XML is available at

- https://www.w3schools.com/xml/default.asp

Resources:

- https://en.wikipedia.org/wiki/XML

15.6.18.2 MongoDB in Python

🎓 **Learning Objectives**

- Introduction to basic MongoDB knowledge
- Use of MOngoDB via PyMongo
- Use of MongoEngine MongoEngine and Object-Document mapper,
- Use of Flask-Mongo

In today's era, NoSQL databases have developed an enormous potential to process the unstructured data efficiently. Modern information is complex, extensive, and may not have pre-existing relationships. With the advent of the advanced search engines, machine learning, and Artificial Intelligence, technology expectations to process, store, and analyze such data have grown tremendously [60]. The NoSQL database engines such as MongoDB, Redis, and Cassandra have successfully overcome the traditional relational database challenges such as scalability, performance, unstructured data growth, agile sprint cycles, and growing needs of processing data in real-time with minimal hardware processing power [61]. The NoSQL databases are a new generation of engines that do not necessarily require SQL language and are sometimes also called Not Only SQL databases. However, most of them support various third-party open connectivity drivers that can map NoSQL queries to SQL's. It would be safe to say that although NoSQL databases are still far from replacing the relational databases, they are adding an immense value when used in hybrid IT environments in conjunction with relational databases, based on the application specific needs [61]. In this paper, we will be covering the MongoDB technology, its driver PyMongo, its object-document mapper MongoEngine, and the Flask-PyMongo micro-web framework that make MongoDB more attractive and user-friendly.

15.6.18.2.1 MongoDB

Today MongoDB is one of leading NoSQL database which is fully capable of handling dynamic changes, processing large volumes of complex and unstructured data, easily using object-oriented programming features; as well as distributed system challenges [62]. At its core, MongoDB is an open source, cross-platform, document database mainly written in C++ language.

15.6.18.2.1.1 Installation

MongoDB can be installed on various Unix Platforms, including Linux, Ubuntu, Amazon Linux, etc [63]. This section focuses on installing MongoDB on Ubuntu 18.04 Bionic Beaver used as a standard OS for a virtual machine used as a part of Big Data Application Class during the 2018 Fall semester.

15.6.18.2.1.1.1 Installation procedure

Before installing, it is recommended to configure the non-root user and provide the administrative privileges to it, in order to be able to perform general MongoDB admin tasks. This can be accomplished by login as the root user in the following manner [64].

```
$ adduser mongoadmin
$ usermod -aG sudo sammy
```

When logged in as a regular user, one can perform actions with superuser privileges by typing sudo before each command [64].

Once the user set up is completed, one can login as a regular user (mongoadmin) and use the following instructions to install MongoDB.

To update the Ubuntu packages to the most recent versions, use below command:

```
$ sudo apt update
```

To install the MongoDB package:

```
$ sudo apt install -y mongodb
```

To check the service and database status:

```
$ sudo systemctl status mongodb
```

Verifying the status of a successful MongoDB installation can be confirmed with an output similar to this:

```
$ mongodb.service - An object/document-oriented database
    Loaded: loaded (/lib/systemd/system/mongodb.service; enabled; vendor preset: enabled)
    Active: **active** (running) since Sat 2018-11-15 07:48:04 UTC; 2min 17s ago
      Docs: man:mongod(1)
  Main PID: 2312 (mongod)
     Tasks: 23 (limit: 1153)
    CGroup: /system.slice/mongodb.service
            └─2312 /usr/bin/mongod --unixSocketPrefix=/run/mongodb --config /etc/mongodb.conf
```

To verify the configuration, more specifically the installed version, server, and port, use the following command:

```
$ mongo --eval 'db.runCommand({ connectionStatus: 1 })'
```

Similarly, to restart MongoDB, use the following:

```
$ sudo systemctl restart mongodb
```

To allow access to MongoDB from an outside hosted server one can use the following command which opens the fire-wall connections [63].

```
$ sudo ufw allow from your_other_server_ip/32 to any port 27017
```

Status can be verified by using:

```
$ sudo ufw status
```

Other MongoDB configurations can be edited through the /etc/mongodb.conf files such as port and hostnames, file paths.

```
$ sudo nano /etc/mongodb.conf
```

Also, to complete this step, a server's IP address must be added to the bindIP value [63].

```
$ logappend=true

  bind_ip = 127.0.0.1,your_server_ip
  *port = 27017*
```

MongoDB is now listening for a remote connection that can be accessed by anyone with appropriate credentials [63].

15.6.18.2.1.2 Collections and Documents

Each database within Mongo environment contains collections which in turn contain documents. Collections and documents are analogous to tables and rows respectively to the relational databases. The document structure is in a key-value form which allows storing of complex data types composed out of field and value pairs. Documents are objects which correspond to native data types in many programming languages, hence a well defined, embedded document can help reduce expensive joins and improve query performance. The `_id` field helps to identify each document uniquly [61].

MongoDB offers flexibility to write records that are not restricted by column types. The data storage approach is flexible as it allows one to store data as it grows and to fulfill varying needs of applications and/or users. It supports JSON like binary points known as BSON where data can be stored without specifying the type of data. Moreover, it can be distributed to multiple machines at high speed. It includes a sharding feature that partitions and spreads the data out across various servers. This makes MongoDB an excellent choice for cloud data processing. Its utilities can load high volumes of data at high speed which ultimately provides greater flexibility and availability in a cloud-based environment [60].

The dynamic schema structure within MongoDB allows easy testing of the small sprints in the Agile project management life cycles and research projects that require frequent changes to the data structure with minimal downtime. Contrary to this flexible process, modifying the data structure of relational databases can be a very tedious process [60].

15.6.18.2.1.2.1 Collection example:

The following collection example for a person named Corey includes additional information such as age, status, and group [65].

```
{
 name: "Corey"
 age: "21"
 status: "Open"
 group: ["AI" , "Machine Learning"]
}
```

15.6.18.2.1.2.2 Document structure:

```
{
   field1: value1,
   field2: value2,
   field3: value3,
   ...
   fieldN: valueN
}
```

15.6.18.2.1.2.3 Collection Operations

If collection does not exists, MongoDB database will create a collection by default.

```
> db.myNewCollection1.insertOne( { x: 1 } )
> db.myNewCollection2.createIndex( { y: 1 } )
```

15.6.18.2.1.3 MongoDB Querying

The data retrieval patterns, the frequency of data manipulation statements such as insert, updates, and deletes may demand for the use of indexes or incorporating the sharding feature to improve query performance and efficiency of MongoDB environment [61]. One of the significant difference between relational databases and NoSQL databases are joins. In the relational database, one can combine results from two or more tables using a common column, often called as key. The native table contains the primary key column while the referenced table contains a foreign key. This mechanism allows one to make changes in a single row instead of changing all rows in the referenced table. This action is referred to as normalization. MongoDB is a document database and mainly contains denormalized data which means the data is repeated instead of indexed over a specific key. If the same data is required in more than one table, it needs to be repeated. This constraint has been eliminated in MongoDB's new version 3.2. The new release introduced a $lookup feature which more likely works as a left-outer-join. Lookups are restricted to aggregated functions which means that data usually need some type of filtering and grouping operations to be conducted beforehand. For this reason, joins in MongoDB require more complicated querying compared to the traditional relational database joins. Although at this time, lookups are still very far from replacing joins, this is a prominent feature that can resolve some of the relational data challenges for MongoDB [66]. MongoDB queries support regular expressions as well as range asks for specific fields that eliminate the need of returning entire documents [61]. MongoDB collections do not enforce document structure like SQL databases which is a compelling feature. However, it is essential to keep in mind the needs of the applications[60].

15.6.18.2.1.3.1 Mongo Queries examples:

The queries can be executed from Mongo shell as well as through scripts.

To query the data from a MongoDB collection, one would use MongoDB's find() method.

```
> db.COLLECTION_NAME.find()
```

The output can be formatted by using the pretty() command.

```
> db.mycol.find().pretty()
```

The MongoDB insert statements can be performed in the following manner:

```
> db.COLLECTION_NAME.insert(document)
```

> "The $lookup command performs a left-outer-join to an unsharded collection in the same database to filter in documents from the joined collection for processing" [67].

```
$ {
    $lookup:
      {
        from: <collection to join>,
        localField: <field from the input documents>,
        foreignField: <field from the documents of the "from" collection>,
        as: <output array field>
      }
  }
```

This operation is equivalent to the following SQL operation:

```
$ SELECT *, <output array field>
  FROM collection
  WHERE <output array field> IN (SELECT *
                                 FROM <collection to join>
                                 WHERE <foreignField> = <collection.localField>);`
```

To perform a Like Match (Regex), one would use the following command:

```
> db.products.find( { sku: { $regex: /789$/ } } )
```

15.6.18.2.1.4 MongoDB Basic Functions

When it comes to the technical elements of MongoDB, it posses a rich interface for importing and storage of external data in various formats. By using the Mongo Import/Export tool, one can easily transfer contents from JSON, CSV, or TSV files into a database. MongoDB supports CRUD (create, read, update, delete) operations efficiently and has detailed documentation available on the product website. It can also query the geospatial data, and it is capable of storing geospatial data in GeoJSON objects. The aggregation operation of the MongoDB process data records and returns computed results. MongoDB aggregation framework is modeled on the concept of data pipelines [68].

15.6.18.2.1.4.1 Import/Export functions examples:

To import JSON documents, one would use the following command:

```
$ mongoimport --db users --collection contacts --file contacts.json
```

The CSV import uses the input file name to import a collection, hence, the collection name is optional [68].

```
$ mongoimport --db users --type csv --headerline --file /opt/backups/contacts.csv
```

> "Mongoexport is a utility that produces a JSON or CSV export of data stored in a MongoDB instance" [68].

```
$ mongoexport --db test --collection traffic --out traffic.json
```

15.6.18.2.1.5 Security Features

Data security is a crucial aspect of the enterprise infrastructure management and is the reason why MongoDB provides various security features such as ole based access control, numerous authentication options, and encryption. It supports mechanisms such as SCRAM, LDAP, and Kerberos authentication. The administrator can create role/collection-based access control; also roles can be predefined or custom. MongoDB can audit activities such as DDL, CRUD statements, authentication and authorization operations [69].

15.6.18.2.1.5.1 Collection based access control example:

A user defined role can contain the following privileges [69].

```
$ privileges: [
   { resource: { db: "products", collection: "inventory" }, actions: [ "find", "update"] },
   { resource: { db: "products", collection: "orders" },  actions: [ "find" ] }
]
```

15.6.18.2.1.6 MongoDB Cloud Service

In regards to the cloud technologies, MongoDB also offers fully automated cloud service called Atlas with competitive pricing options. Mongo Atlas Cloud interface offers interactive GUI for managing cloud resources and deploying applications quickly. The service is equipped with geographically distributed instances to ensure no single point failure. Also, a well-rounded performance monitoring interface allows users to promptly detect anomalies and generate index suggestions to optimize the performance and reliability of the database. Global technology leaders such as Google, Facebook, eBay, and Nokia are leveraging MongoDB and Atlas cloud services making MongoDB one of the most popular choices among the NoSQL databases [70].

15.6.18.2.2 PyMongo

PyMongo is the official Python driver or distribution that allows work with a NoSQL type database called MongoDB [71]. The first version of the driver was developed in 2009 [72], only two years after the development of MongoDB was started. This driver allows developers to combine both Python's versatility and MongoDB's flexible schema nature into successful applications. Currently, this driver supports MongoDB versions 2.6, 3.0, 3.2, 3.4, 3.6, and 4.0 [73]. MongoDB and Python represent a compatible fit considering that BSON (binary JSON) used in this NoSQL database is very similar to Python dictionaries, which makes the collaboration between the two even more appealing [74]. For this reason, dictionaries are the recommended tools to be used in PyMongo when representing documents [75].

15.6.18.2.2.1 Installation

Prior to being able to exploit the benefits of Python and MongoDB simultaneously, the PyMongo distribution must be installed using pip. To install it on all platforms, the following command should be used [76]:

```
$ python -m pip install pymongo
```

Specific versions of PyMongo can be installed with command lines such as in our example where the 3.5.1 version is installed [76].

```
$ python -m pip install pymongo==3.5.1
```

A single line of code can be used to upgrade the driver as well [76].

```
$ python -m pip install --upgrade pymongo
```

Furthermore, the installation process can be completed with the help of the easy_install tool, which requires users to use the following command [76].

```
$ python -m easy_install pymongo
```

To do an upgrade of the driver using this tool, the following command is recommended [76]:

```
$ python -m easy_install -U pymongo
```

There are many other ways of installing PyMongo directly from the source, however, they require for C extension dependencies to be installed prior to the driver installation step, as they are the ones that skim through the sources on GitHub and use the most up-to-date links to install the driver [76].

To check if the installation was completed accurately, the following command is used in the Python console [77].

```
import pymongo
```

If the command returns zero exceptions within the Python shell, one can consider for the PyMongo installation to have been completed successfully.

15.6.18.2.2.2 Dependencies

The PyMongo driver has a few dependencies that should be taken into consideration prior to its usage. Currently, it supports CPython 2.7, 3.4+, PyPy, and PyPy 3.5+ interpreters [73]. An optional dependency that requires some additional components to be installed is the GSSAPI authentication [73]. For the Unix based machines, it requires pykerberos, while for the Windows machines WinKerberos is needed to fullfill this requirement [73]. The automatic installation of this dependency can be done simultaneously with the driver installation, in the following manner:

```
$ python -m pip install pymongo[gssapi]
```

Other third-party dependencies such as ipaddress, certifi, or wincerstore are necessary for connections with help of TLS/SSL and can also be simultaneously installed along with the driver installation [73].

15.6.18.2.2.3 Running PyMongo with Mongo Deamon

Once PyMongo is installed, the Mongo deamon can be run with a very simple command in a new terminal window [77].

```
$ mongod
```

15.6.18.2.2.4 Connecting to a database using MongoClient

In order to be able to establish a connection with a database, a MongoClient class needs to be imported, which sub-sequentially allows the MongoClient object to communicate with the database [77].

```
from pymongo import MongoClient
client = MongoClient()
```

This command allows a connection with a default, local host through port 27017, however, depending on the programming requirements, one can also specify those by listing them in the client instance or use the same information via the Mongo URI format [77].

15.6.18.2.2.5 Accessing Databases

Since MongoClient plays a server role, it can be used to access any desired databases in an easy way. To do that, one can use two different approaches. The first approach would be doing this via the attribute method where the name of the desired database is listed as an attribute, and the second approach, which would include a dictionary-style access [77]. For example, to access a database called cloudmesh_community, one would use the following commands for the attribute and for the dictionary method, respectively.

```
db = client.cloudmesh_community
db = client['cloudmesh_community']
```

15.6.18.2.2.6 Creating a Database

Creating a database is a straight forward process. First, one must create a MongoClient object and specify the connection (IP address) as well as the name of the database they are trying to create [78]. The example of this command is presented in the followng section:

```
import pymongo
client = pymongo.MongoClient('mongodb://localhost:27017/')
db = client['cloudmesh']
```

15.6.18.2.2.7 Inserting and Retrieving Documents (Querying)

Creating documents and storing data using PyMongo is equally easy as accessing and creating databases. In order to add new data, a collection must be specified first. In this example, a decision is made to use the cloudmesh group of documents.

```
cloudmesh = db.cloudmesh
```

Once this step is completed, data may be inserted using the insert_one() method, which means that only one document is being created. Of course, insertion of multiple documents at the same time is possible as well with use of the insert_many() method [77]. An example of this method is as follows:

```
course_info = {
     'course': 'Big Data Applications and Analytics',
     'instructor': ' Gregor von Laszewski',
     'chapter': 'technologies'
}
result = cloudmesh.insert_one(course_info)`
```

Another example of this method would be to create a collection. If we wanted to create a collection of students in the cloudmesh_community, we would do it in the following manner:

```
student = [ {'name': 'John', 'st_id': 52642},
    {'name': 'Mercedes', 'st_id': 5717},
    {'name': 'Anna', 'st_id': 5654},
    {'name': 'Greg', 'st_id': 5423},
    {'name': 'Amaya', 'st_id': 3540},
    {'name': 'Cameron', 'st_id': 2343},
    {'name': 'Bozer', 'st_id': 4143},
    {'name': 'Cody', 'price': 2165} ]

client = MongoClient('mongodb://localhost:27017/')

with client:
    db = client.cloudmesh
    db.students.insert_many(student)
```

Retrieving documents is equally simple as creating them. The find_one() method can be used to retrieve one document [77]. An implementation of this method is given in the following example.

```
gregors_course = cloudmesh.find_one({'instructor':'Gregor von Laszewski'})
```

Similarly, to retieve multiple documents, one would use the find() method instead of the find_one(). For example, to find all courses thought by professor von Laszewski, one would use the following command:

```
gregors_course = cloudmesh.find({'instructor':'Gregor von Laszewski'})
```

One thing that users should be cognizant of when using the find() method is that it does not return results in an array format but as a cursor object, which is a combination of methods that work together to help with data querying [77]. In order to return individual documents, iteration over the result must be completed [77].

15.6.18.2.2.8 Limiting Results

When it comes to working with large databases it is always useful to limit the number of query results. PyMongo supports this option with its limit() method [78]. This method takes in one parameter which specifies the number of documents to be returned [78]. For example, if we had a collection with a large number of cloud technologies as individual documents, one could modify the query results to return only the top 10 technologies. To do this, the following example could be utilized:

```
client = pymongo.MongoClient('mongodb://localhost:27017/')
    db = client['cloudmesh']
    col = db['technologies']
    topten = col.find().limit(10)
```

15.6.18.2.2.9 Updating Collection

Updating documents is very similar to inserting and retrieving the same. Depending on the number of documents to be updated, one would use the update_one() or update_many() method [78]. Two parameters need to be passed in the update_one() method for it to successfully execute. The first argument is the query object that specifies the document to be changed, and the second argument is the object that specifies the new value in the document. An example of the update_one() method in action is the following:

```
myquery = { 'course': 'Big Data Applications and Analytics' }
newvalues = { '$set': { 'course': 'Cloud Computing' } }
```

Updating all documents that fall under the same criteria can be done with the update_many method [78]. For example, to update all documents in which course title starts with letter B with a different instructor information, we would do the following:

```
client = pymongo.MongoClient('mongodb://localhost:27017/')
db = client['cloudmesh']
```

```
col = db['courses']
query = { 'course': { '$regex': '^B' } }
newvalues = { '$set': { 'instructor': 'Gregor von Laszewski' } }

edited = col.update_many(query, newvalues)
```

15.6.18.2.2.10 Counting Documents

Counting documents can be done with one simple operation called count_documents() instead of using a full query [79]. For example, we can count the documents in the cloudmesh_commpunity by using the following command:

```
cloudmesh = count_documents({})
```

To create a more specific count, one would use a command similar to this:

```
cloudmesh = count_documents({'author': 'von Laszewski'})
```

This technology supports some more advanced querying options as well. Those advanced queries allow one to add certain contraints and narrow down the results even more. For example, to get the courses thought by professor von Laszewski after a certain date, one would use the following command:

```
d = datetime.datetime(2017, 11, 12, 12)
for course in cloudmesh.find({'date': {'$lt': d}}).sort('author'):
    pprint.pprint(course)
```

15.6.18.2.2.11 Indexing

Indexing is a very important part of querying. It can greately improve query performance but also add functionality and aide in storing documents [79].

> "To create a unique index on a key that rejects documents whose value for that key already exists in the index" [79].

We need to firstly create the index in the following manner:

```
result = db.profiles.create_index([('user_id', pymongo.ASCENDING)],
unique=True)
sorted(list(db.profiles.index_information()))
```

This command acutally creates two different indexes. The first one is the *_id* , created by MongoDB automatically, and the second one is the user_id, created by the user.

The purpose of those indexes is to cleverly prevent future additions of invalid user_ids into a collection.

15.6.18.2.2.12 Sorting

Sorting on the server-side is also avaialable via MongoDB. The PyMongo sort() method is equivalent to the SQL order by statement and it can be performed as pymongo.ascending and pymongo.descending [80]. This method is much more efficient as it is being completed on the server-side, compared to the sorting completed on the client side. For example, to return all users with first name Gregor sorted in descending order by birthdate we would use a command such as this:

```
users = cloudmesh.users.find({'firstname':'Gregor'}).sort(('dateofbirth', pymongo.DESCENDING))
for user in users:
   print user.get('email')
```

15.6.18.2.2.13 Aggregation

Aggregation operations are used to process given data and produce summarized results. Aggregation operations collect data from a number of documents and provide collective results by grouping data. PyMongo in its documentation offers a separate framework that supports data aggregation. This aggregation framework can be used to

> "provide projection capabilities to reshape the returned data" [81].

In the aggregation pipeline, documents pass through multiple pipeline stages which convert documents into result data. The basic pipeline stages include filters. Those filters act like document transformation by helping change the document output form. Other pipelines help group or sort documents with specific fields. By using native operations from MongoDB, the pipeline operators are efficient in aggregating results.

The addFields stage is used to add new fields into documents. It reshapes each document in stream, similarly to the project stage. The output document will contain existing fields from input documents and the newly added fields 82]. The following example shows how to add student details into a document.

```
 db.cloudmesh_community.aggregate([
{
       $addFields: {
       "document.StudentDetails": {
       $concat:['$document.student.FirstName', '$document.student.LastName']
           }
       }
   } ])
```

The bucket stage is used to categorize incoming documents into groups based on specified expressions. Those groups are called buckets [82]. The following example shows the bucket stage in action.

```
db.user.aggregate([
{ "$group": {
  "_id": {
    "city": "$city",
    "age": {
      "$let": {
        "vars": {
 "age": { "$subtract" :[{ "$year": new Date() },{ "$year": "$birthDay" }] }},
        "in": {
          "$switch": {
            "branches": [
              { "case": { "$lt": [ "$$age", 20 ] }, "then": 0 },
              { "case": { "$lt": [ "$$age", 30 ] }, "then": 20 },
              { "case": { "$lt": [ "$$age", 40 ] }, "then": 30 },
              { "case": { "$lt": [ "$$age", 50 ] }, "then": 40 },
              { "case": { "$lt": [ "$$age", 200 ] }, "then": 50 }
            ] }  }  } } },
  "count": { "$sum": 1 }}})
```

In the bucketAuto stage, the boundaries are automatically determined in an attempt to evenly distribute documents into a specified number of buckets. In the following operation, input documents are grouped into four buckets according to the values in the price field [82].

```
db.artwork.aggregate( [
  {
    $bucketAuto: {
        groupBy: "$price",
        buckets: 4
    }
  }
] )
```

The collStats stage returns statistics regarding a collection or view [82].

```
db.matrices.aggregate( [ { $collStats: { latencyStats: { histograms: true } }
 } ] )
```

The count stage passes a document to the next stage that contains the number documents that were input to the stage [82].

```
db.scores.aggregate(  [    {
   $match: {        score: {          $gt: 80    } }  },
{      $count: "passing_scores"  } ])
```

The facet stage helps process multiple aggregation pipelines in a single stage [82].

```
db.artwork.aggregate( [ {
   $facet: {  "categorizedByTags": [    { $unwind: "$tags" },
       { $sortByCount: "$tags" }  ],  "categorizedByPrice": [
```

```
      // Filter out documents without a price e.g., _id: 7
       { $match: { price: { $exists: 1 } } },
      { $bucket: { groupBy: "$price",
          boundaries: [  0, 150, 200, 300, 400 ],
          default: "Other",
          output: { "count": { $sum: 1 },
            "titles": { $push: "$title" }
          } }        }], "categorizedByYears(Auto)": [
      { $bucketAuto: { groupBy: "$year",buckets: 4 }
      } ]}}])
```

The geoNear stage returns an ordered stream of documents based on the proximity to a geospatial point. The output documents include an additional distance field and can include a location identifier field [82].

```
db.places.aggregate([
 {    $geoNear: {
      near: { type: "Point", coordinates: [ -73.99279 , 40.719296 ] },
      distanceField: "dist.calculated",
      maxDistance: 2,
      query: { type: "public" },
      includeLocs: "dist.location",
      num: 5,
      spherical: true
   }  }])
```

The graphLookup stage performs a recursive search on a collection. To each output document, it adds a new array field that contains the traversal results of the recursive search for that document [82].

```
db.travelers.aggregate( [
 {
    $graphLookup: {
       from: "airports",
       startWith: "$nearestAirport",
       connectFromField: "connects",
       connectToField: "airport",
       maxDepth: 2,
       depthField: "numConnections",
       as: "destinations"
    }
 }
] )
```

The group stage consumes the document data per each distinct group. It has a RAM limit of 100 MB. If the stage exceeds this limit, the group produces an error [82].

```
db.sales.aggregate(
 [
    {
      $group : {
         _id : { month: { $month: "$date" }, day: { $dayOfMonth: "$date" },
         year: { $year: "$date" } },
         totalPrice: { $sum: { $multiply: [ "$price", "$quantity" ] } },
         averageQuantity: { $avg: "$quantity" },
         count: { $sum: 1 }
       }
    }
 ]
)
```

The indexStats stage returns statistics regarding the use of each index for a collection [82].

```
db.orders.aggregate( [ { $indexStats: { } } ] )
```

The limit stage is used for controlling the number of documents passed to the next stage in the pipeline [82].

```
db.article.aggregate(
  { $limit : 5 }
)
```

The listLocalSessions stage gives the session information currently connected to mongos or mongod instance [82].

```
db.aggregate( [  { $listLocalSessions: { allUsers: true } } ] )
```

The listSessions stage lists out all session that have been active long enough to propagate to the system.sessions collection [82].

```
use config
db.system.sessions.aggregate( [  { $listSessions: { allUsers: true } } ] )
```

The lookup stage is useful for performing outer joins to other collections in the same database [82].

```
{
   $lookup:
     {
       from: <collection to join>,
       localField: <field from the input documents>,
       foreignField: <field from the documents of the "from" collection>,
       as: <output array field>
     }
}
```

The match stage is used to filter the document stream. Only matching documents pass to next stage [82].

```
db.articles.aggregate(
    [ { $match : { author : "dave" } } ]
)
```

The project stage is used to reshape the documents by adding or deleting the fields.

```
db.books.aggregate( [ { $project : { title : 1 , author : 1 } } ] )
```

The redact stage reshapes stream documents by restricting information using information stored in documents themselves [82].

```
 db.accounts.aggregate(
 [
   { $match: { status: "A" } },
   {
     $redact: {
       $cond: {
         if: { $eq: [ "$level", 5 ] },
         then: "$$PRUNE",
         else: "$$DESCEND"
       }      }    }  ]);
```

The replaceRoot stage is used to replace a document with a specified embedded document [82].

```
 db.produce.aggregate( [
  {
    $replaceRoot: { newRoot: "$in_stock" }
  }
] )
```

The sample stage is used to sample out data by randomly selecting number of documents form input [82].

```
 db.users.aggregate(
  [ { $sample: { size: 3 } } ]
)
```

The skip stage skips specified initial number of documents and passes remaining documents to the pipeline [82].

```
db.article.aggregate(
   { $skip : 5 }
);
```

The sort stage is useful while reordering document stream by a specified sort key [82].

```
db.users.aggregate(
   [
     { $sort : { age : -1, posts: 1 } }
   ]
)
```

The sortByCounts stage groups the incoming documents based on a specified expression value and counts documents in each distinct group [82].

```
db.exhibits.aggregate(
[ { $unwind: "$tags" },  { $sortByCount: "$tags" } ] )
```

The unwind stage deconstructs an array field from the input documents to output a document for each element [82].

```
db.inventory.aggregate( [ { $unwind: "$sizes" } ] )
db.inventory.aggregate( [ { $unwind: { path: "$sizes" } } ] )
```

The out stage is used to write aggregation pipeline results into a collection. This stage should be the last stage of a pipeline [82].

```
db.books.aggregate( [
                  { $group : { _id : "$author", books: { $push: "$title" } } },
                     { $out : "authors" }
                  ] )
```

Another option from the aggregation operations is the Map/Reduce framework, which essentially includes two different functions, map and reduce. The first one provides the key value pair for each tag in the array, while the latter one

> "sums over all of the emitted values for a given key" [81].

The last step in the Map/Reduce process it to call the map_reduce() function and iterate over the results [81]. The Map/Reduce operation provides result data in a collection or returns results in-line. One can perform subsequent operations with the same input collection if the output of the same is written to a collection [83]. An operation that produces results in a in-line form must provide results with in the BSON document size limit. The current limit for a BSON document is 16 MB. These types of operations are not supported by views [83]. The PyMongo's API supports all features of the MongoDB's Map/Reduce engine [84]. Moreover, Map/Reduce has the ability to get more detailed results by passing full_response=True argument to the map_reduce() function [84].

15.6.18.2.2.14 Deleting Documents from a Collection

The deletion of documents with PyMongo is fairly straight forward. To do so, one would use the remove() method of the PyMongo Collection object [80]. Similarly to the reads and updates, specification of documents to be removed is a must. For example, removal of the entire document collection with a score of 1, would required one to use the following command:

```
cloudmesh.users.remove({"score":1, safe=True})
```

The safe parameter set to True ensures the operation was completed [80].

15.6.18.2.2.15 Copying a Database

Copying databases within the same mongod instance or between different mongod servers is made possible with the command() method after connecting to the desired mongod instance [85]. For example, to copy the cloudmesh database and name the new database cloudmesh_copy, one would use the command() method in the following manner:

```
client.admin.command('copydb',
                     fromdb='cloudmesh',
                     todb='cloudmesh_copy')
```

There are two ways to copy a database between servers. If a server is not password-prodected, one would not need to pass in the credentials nor to authenticate to the admin database [85]. In that case, to copy a database one would use the following command:

```
client.admin.command('copydb',
                     fromdb='cloudmesh',
                     todb='cloudmesh_copy',
                     fromhost='source.example.com')
```

On the other hand, if the server where we are copying the database to is protected, one would use this command instead:

```
client = MongoClient('target.example.com',
                     username='administrator',
                     password='pwd')
client.admin.command('copydb',
                     fromdb='cloudmesh',
                     todb='cloudmesh_copy',
                     fromhost='source.example.com')
```

15.6.18.2.2.16 PyMongo Strengths

One of PyMongo strengths is that allows document creation and querying natively

> "through the use of existing language features such as nested dictionaries and lists" [80].

For moderately experienced Python developers, it is very easy to learn it and quickly feel comfortable with it.

> "For these reasons, MongoDB and Python make a powerful combination for rapid, iterative development of horizontally scalable backend applications" [80].

According to [80], MongoDB is very applicable to modern applications, which makes PyMongo equally valuable [80].

15.6.18.2.3 MongoEngine

> "MongoEngine is an Object-Document Mapper, written in Python for working with MongoDB" [86].

It is actually a library that allows a more advanced communication with MongoDB compared to PyMongo. As MongoEngine is technically considered to be an object-document mapper(ODM), it can also be considered to be

> "equivalent to a SQL-based object relational mapper(ORM)" [77].

The primary technique why one would use an ODM includes data conversion between computer systems that are not compatible with each other [87]. For the purpose of converting data to the appropriate form, a virtual object database must be created within the utilized programming language [87]. This library is also used to define schemata for documents within MongoDB, which ultimately helps with minimizing coding errors as well defining methods on existing fields [88]. It is also
very beneficial to the overall workflow as it tracks changes made to the documents and aids in the document saving process [89].

15.6.18.2.3.1 Installation

The installation process for this technology is fairly simple as it is considered to be a library. To install it, one would use the following command [90]:

```
$ pip install mongoengine
```

A bleeding-edge version of MongoEngine can be installed directly from GitHub by first cloning the repository on the local machine, virtual machine, or cloud.

15.6.18.2.3.2 Connecting to a database using MongoEngine

Once installed, MongoEngine needs to be connected to an instance of the mongod, similarly to PyMongo [91]. The connect() function must be used to successfully complete this step and the argument that must be used in this function is the name of the desired database [91]. Prior to using this function, the function name needs to be imported from the MongoEngine library.

```
from mongoengine import connect
connect('cloudmesh_community')
```

Similarly to the MongoClient, MongoEngine uses the local host and port 27017 by default, however, the connect() function also allows specifying other hosts and port arguments as well [91].

```
connect('cloudmesh_community', host='196.185.1.62', port=16758)
```

Other types of connections are also supported (i.e. URI) and they can be completed by providing the URI in the connect() function [91].

15.6.18.2.3.3 Querying using MongoEngine

To query MongoDB using MongoEngine an objects attribute is used, which is, technically, a part of the document class [92]. This attribute is called the QuerySetManager which in return

> "creates a new QuerySet object on access" [92].

To be able to access individual documents from a database, this object needs to be iterated over. For example, to return/print all students in the cloudmesh_community object (database), the following command would be used.

```
for user in cloudmesh_community.objects:
   print cloudmesh_community.student
```

MongoEngine also has a capability of query filtering which means that a keyword can be used within the called QuerySet object to retrieve specific information [92]. Let's say one would like to iterate over cloudmesh_community students that are natives of Indiana. To achieve this, one would use the following command:

```
indy_students = cloudmesh_community.objects(state='IN')
```

This library also allows the use of all operators except for the equality operator in its queries, and moreover, has the capability of handling string queries, geo queries, list querying, and querying of the raw PyMongo queries [92].

The string queries are useful in performing text operations in the conditional queries. A query to find a document exactly matching and with state ACTIVE can be performed in the following manner:

```
db.cloudmesh_community.find( State.exact("ACTIVE") )
```

The query to retrieve document data for names that start with a case sensitive AL can be written as:

```
db.cloudmesh_community.find( Name.startswith("AL") )
```

To perform an exact same query for the non-key-sensitive AL one would use the following command:

```
db.cloudmesh_community.find( Name.istartswith("AL") )
```

The MongoEngine allows data extraction of geographical locations by using Geo queries. The geo_within operator checks if a geometry is within a polygon.

```
cloudmesh_community.objects(
          point__geo_within=[[[40, 5], [40, 6], [41, 6], [40, 5]]])
cloudmesh_community.objects(
          point__geo_within={"type": "Polygon",
               "coordinates": [[[40, 5], [40, 6], [41, 6], [40, 5]]]})
```

The list query looks up the documents where the specified fields matches exactly to the given value. To match all pages that have the word coding as an item in the tags list one would use the following query:

```
class Page(Document):
   tags = ListField(StringField())

Page.objects(tags='coding')
```

Overall, it would be safe to say that MongoEngine has good compatibility with Python. It provides different functions to utilize Python easily with MongoDBand which makes this pair even more attractive to application developers.

15.6.18.2.4 Flask-PyMongo

> "Flask is a micro-web framework written in Python" [93].

It was developed after Django, and it is very pythonic in nature which implies that it is explicitly the targeting the Python user community. It is lightweight as it does not require additional tools or libraries and hence is classified as a Micro-Web framework. It is often used with MongoDB using PyMongo connector, and it treats data within MongoDB as searchable Python dictionaries. The applications such as Pinterest, LinkedIn, and the community web page for Flask are using the Flask framework. Moreover, it supports various features such as the RESTful request dispatching, secure cookies, Google app engine compatibility, and integrated support for unit testing, etc [93]. When it comes to connecting to a database, the connection details for MongoDB can be passed as a variable or configured in PyMongo constructor with additional arguments such as username and password, if required. It is important that versions of both Flask and MongoDB are compatible with each other to avoid functionality breaks [94].

15.6.18.2.4.1 Installation

Flask-PyMongo can be installed with an easy command such as this:

```
$ pip install Flask-PyMongo
```

PyMongo can be added in the following manner:

```
from flask import Flask
from flask_pymongo import PyMongo
app = Flask(__name__)
app.config["MONGO_URI"] = "mongodb://localhost:27017/cloudmesh_community"
mongo = PyMongo(app)
```

15.6.18.2.4.2 Configuration

There are two ways to configure Flask-PyMongo. The first way would be to pass a MongoDB URI to the PyMongo constructor, while the second way would be to

> "assign it to the MONGO_URI Flask confiuration variable" [94].

15.6.18.2.4.3 Connection to multiple databases/servers

Multiple PyMongo instances can be used to connect to multiple databases or database servers. To achieve this, once would use a command similar to the following:

```
app = Flask(__name__)
mongo1 = PyMongo(app, uri="mongodb://localhost:27017/cloudmesh_community_one")
mongo2 = PyMongo(app, uri="mongodb://localhost:27017/cloudmesh_community_two")
mongo3 = PyMongo(app, uri=
      "mongodb://another.host:27017/cloudmesh_community_Three")
```

15.6.18.2.4.4 Flask-PyMongo Methods

Flask-PyMongo provides helpers for some common tasks. One of them is the Collection.find_one_or_404 method shown in the following example:

```
@app.route("/user/<username>")
def user_profile(username):
    user = mongo.db.cloudmesh_community.find_one_or_404({"_id": username})
    return render_template("user.html", user=user)
```

This method is very similar to the MongoDB's find_one() method, however, instead of returning None it causes a 404 Not Found HTTP status [94].

Similarly, the PyMongo.send_file and PyMongo.save_file methods work on the file-like objects and save them to GridFS using the given file name [94].

15.6.18.2.4.5 Additional Libraries

Flask-MongoAlchemy and Flask-MongoEngine are the additional libraries that can be used to connect to a MongoDB database while using enhanced features with the Flask app. The Flask-MongoAlchemy is used as a proxy between Python and MongoDB to connect. It provides an option such as server or database based authentication to connect to MongoDB. While the default is set server based, to use a database-based authentication, the config value MONGOALCHEMY_SERVER_AUTH parameter must be set to False [95].

Flask-MongoEngine is the Flask extension that provides integration with the MongoEngine. It handles connection management for the apps. It can be installed through pip and set up very easily as well. The default configuration is set to the local host and port 27017. For the custom port and in cases where MongoDB is running on another server, the host and port must be explicitly specified in connect strings within the MONGODB_SETTINGS dictionary with app.config, along with the database username and password, in cases where a database authentication is enabled. The URI style connections are also supported and supply the URI as the host in the MONGODB_SETTINGS dictionary with app.config. There are various custom query sets that are available within Flask-Mongoengine that are attached to Mongoengine's default queryset [96].

15.6.18.2.4.6 Classes and Wrappers

Attributes such as cx and db in the PyMongo objects are the ones that help provide access to the MongoDB server [94]. To achieve this, one must pass the Flask app to the constructor or call init_app() [94].

> "Flask-PyMongo wraps PyMongo's MongoClient, Database, and Collection classes, and overrides their attribute and item accessors" [94].

This type of wrapping allows Flask-PyMongo to add methods to Collection while at the same time allowing a MongoDB-style dotted expressions in the code [94].

```
type(mongo.cx)
type(mongo.db)
type(mongo.db.cloudmesh_community)
```

Flask-PyMongo creates connectivity between Python and Flask using a MongoDB database and supports

> "extensions that can add application features as if they were implemented in Flask itself" [97],

hence, it can be used as an additional Flask functionality in Python code. The extensions are there for the purpose of supporting form validations, authentication technologies, object-relational mappers and framework related tools which ultimately adds a lot of strength to this micro-web framework [97]. One of the main reasons and benefits why it is frequently used with MongoDB is its capability of adding more control over databases and history [97].

15.6.18.3 Mongoengine

15.6.18.3.1 Introduction

MongoEngine is a document mapper for working with mongoldb with python. To be able to use mongo engine MongodD should be already installed and running.

15.6.18.3.2 Install and connect

Mongoengine can be installed by running:

```
    $ pip install mongo engine
```

This will install six, pymongo and mongoengine.

To connect to mongoldb use connect () function by specifying mongoldb instance name. You don't need to go to mongo shell but this can be done from unix shell or cmd line. In this case we are connecting to a database named student_db.

```
from mongo engine import * connect ('student_db')
```

If mongodb is running on a port different from default port , port number and host need to be specified. If mongoldb needs authentication username and password need to be specified.

15.6.18.3.3 Basics

Mongodb does not enforce schemas. Comparing to RDBMS, Row in mongoldb is called a "document" and table can be compared to Collection. Defining a schema is helpful as it minimizes coding error's. To define a schema we create a class that inherits from document.

```
from mongoengine import *

class Student(Document):
    first_name = StringField(max_length=50)
    last_name = StringField(max_length=50)
```

TODO: Can you fix the code sections and look at the examples we provided.

Fields are not mandatory but if needed, set the required keyword argument to True. There are multiple values available for field types. Each field can be customized by by keyword argument. If each student is sending text messages to Universities central database , these can be stored using Mongodb. Each text can have different data types, some might have images or some might have url's. So we can create a class text and link it to student by using Reference field (similar to foreign key in RDBMS).

```
class Text(Document):
    title = StringField(max_length=120, required=True)
    author = ReferenceField(Student)
    meta = {'allow_inheritance': True}

class OnlyText(Text):
    content = StringField()

class ImagePost(Text):
    image_path = StringField()

class LinkPost(Text):
    link_url = StringField()
```

MongoDb supports adding tags to individual texts rather then storing them separately and then having them referenced.Similarly Comments can also be stored directly in a Text.

```
class Text(Document):
    title = StringField(max_length=120, required=True)
    author = ReferenceField(User)
    tags = ListField(StringField(max_length=30))
    comments = ListField(EmbeddedDocumentField(Comment))
```

For accessing data: if we need to get titles.

```
for text in OnlyText.objects:
    print(text.title)
```

Searching texts with tags.

```
for text in Text.objects(tags='mongodb'):
    print(text.title)
```

15.7 Word Count with Parallel Python

We will demonstrate Python's `multiprocessing` API for parallel computation by writing a program that counts how many times each word in a collection of documents appear.

15.7.1 Generating a Document Collection

Before we begin, let us write a script that will generate document collections by specifying the number of documents and the number of words per document. This will make benchmarking straightforward.

To keep it simple, the vocabulary of the document collection will consist of random numbers rather than the words of an actual language:

```
'''Usage: generate_nums.py [-h] NUM_LISTS INTS_PER_LIST MIN_INT MAX_INT DEST_DIR

Generate random lists of integers and save them
as 1.txt, 2.txt, etc.

Arguments:
   NUM_LISTS      The number of lists to create.
   INTS_PER_LIST  The number of integers in each list.
   MIN_NUM        Each generated integer will be >= MIN_NUM.
   MAX_NUM        Each generated integer will be <= MAX_NUM.
   DEST_DIR       A directory where the generated numbers will be stored.

Options:
  -h --help
'''

from __future__ import print_function
import os, random, logging
from docopt import docopt


def generate_random_lists(num_lists,
                          ints_per_list, min_int, max_int):
    return [[random.randint(min_int, max_int) \
```

```
        for i in range(ints_per_list)] for i in range(num_lists)]


if __name__ == '__main__':
   args = docopt(__doc__)
   num_lists, ints_per_list, min_int, max_int, dest_dir = [
      int(args['NUM_LISTS']),
      int(args['INTS_PER_LIST']),
      int(args['MIN_INT']),
      int(args['MAX_INT']),
      args['DEST_DIR']
   ]

   if not os.path.exists(dest_dir):
      os.makedirs(dest_dir)

   lists = generate_random_lists(num_lists,
                                 ints_per_list,
                                 min_int,
                                 max_int)
   curr_list = 1
   for lst in lists:
      with open(os.path.join(dest_dir, '%d.txt' % curr_list), 'w') as f:
     f.write(os.linesep.join(map(str, lst)))
  curr_list += 1
   logging.debug('Numbers written.')
```

Notice that we are using the docopt module that you should be familiar with from the Section [Python DocOpts](#s-python-docopts} to make the script easy to run from the command line.

You can generate a document collection with this script as follows:

```
python generate_nums.py 1000 10000 0 100 docs-1000-10000
```

15.7.2 Serial Implementation

A first serial implementation of wordcount is straightforward:

```
'''Usage: wordcount.py [-h] DATA_DIR

Read a collection of .txt documents and count how many times each word
appears in the collection.

Arguments:
  DATA_DIR  A directory with documents (.txt files).

Options:
  -h --help
'''

from __future__ import division, print_function
import os, glob, logging
from docopt import docopt

logging.basicConfig(level=logging.DEBUG)


def wordcount(files):
   counts = {}
   for filepath in files:
      with open(filepath, 'r') as f:
     words = [word.strip() for word in f.read().split()]
     for word in words:
        if word not in counts:
           counts[word] = 0
        counts[word] += 1
   return counts


if __name__ == '__main__':
   args = docopt(__doc__)
   if not os.path.exists(args['DATA_DIR']):
      raise ValueError('Invalid data directory: %s' % args['DATA_DIR'])

   counts = wordcount(glob.glob(os.path.join(args['DATA_DIR'], '*.txt')))
   logging.debug(counts)
```

15.7.3 Serial Implementation Using map and reduce

We can improve the serial implementation in anticipation of parallelizing the program by making use of Python's `map` and `reduce` functions.

In short, you can use `map` to apply the same function to the members of a collection. For example, to convert a list of numbers to strings, you could do:

```
import random
nums = [random.randint(1, 2) for _ in range(10)]
print(nums)
[2, 1, 1, 1, 2, 2, 2, 2, 2, 2]
print(map(str, nums))
['2', '1', '1', '1', '2', '2', '2', '2', '2', '2']
```

We can use reduce to apply the same function cumulatively to the items of a sequence. For example, to find the total of the numbers in our list, we could use `reduce` as follows:

```
def add(x, y):
    return x + y

print(reduce(add, nums))
17
```

We can simplify this even more by using a lambda function:

```
print(reduce(lambda x, y: x + y, nums))
17
```

You can read more about Python's lambda function in the docs.

With this in mind, we can reimplement the wordcount example as follows:

```
'''Usage: wordcount_mapreduce.py [-h] DATA_DIR

Read a collection of .txt documents and count how
many times each word
appears in the collection.

Arguments:
   DATA_DIR  A directory with documents (.txt files).

Options:
   -h --help
'''

from __future__ import division, print_function
import os, glob, logging
from docopt import docopt

logging.basicConfig(level=logging.DEBUG)
```

```
def count_words(filepath):
   counts = {}
   with open(filepath, 'r') as f:
      words = [word.strip() for word in f.read().split()]

  for word in words:
     if word not in counts:
        counts[word] = 0
     counts[word] += 1
  return counts


def merge_counts(counts1, counts2):
   for word, count in counts2.items():
      if word not in counts1:
     counts1[word] = 0
  counts1[word] += counts2[word]
   return counts1


if __name__ == '__main__':
   args = docopt(__doc__)
   if not os.path.exists(args['DATA_DIR']):
      raise ValueError('Invalid data directory: %s' % args['DATA_DIR'])

      per_doc_counts = map(count_words,
                           glob.glob(os.path.join(args['DATA_DIR'],
                           '*.txt')))
   counts = reduce(merge_counts, [{}] + per_doc_counts)
   logging.debug(counts)
```

15.7.4 Parallel Implementation

Drawing on the previous implementation using `map` and `reduce`, we can parallelize the implementation using Python's `multiprocessing` API:

```
'''Usage: wordcount_mapreduce_parallel.py [-h] DATA_DIR NUM_PROCESSES

Read a collection of .txt documents and count, in parallel, how many
times each word appears in the collection.

Arguments:
   DATA_DIR       A directory with documents (.txt files).
   NUM_PROCESSES  The number of parallel processes to use.

Options:
   -h --help
'''

from __future__ import division, print_function
import os, glob, logging
from docopt import docopt
from wordcount_mapreduce import count_words, merge_counts
from multiprocessing import Pool

logging.basicConfig(level=logging.DEBUG)

if __name__ == '__main__':
   args = docopt(__doc__)
   if not os.path.exists(args['DATA_DIR']):
      raise ValueError('Invalid data directory: %s' % args['DATA_DIR'])
   num_processes = int(args['NUM_PROCESSES'])

   pool = Pool(processes=num_processes)

   per_doc_counts = pool.map(count_words,
                             glob.glob(os.path.join(args['DATA_DIR'],
                             '*.txt')))
   counts = reduce(merge_counts, [{}] + per_doc_counts)
   logging.debug(counts)
```

15.7.5 Benchmarking

To time each of the examples, enter it into its own Python file and use Linux's `time` command:

```
$ time python wordcount.py docs-1000-10000
```

The output contains the real run time and the user run time. real is wall clock time - time from start to finish of the call. user is the amount of CPU time spent in user-mode code (outside the kernel) within the process, that is, only actual CPU time used in executing the process.

15.7.6 Excersises

E.python.wordcount.1:

> Run the three different programs (serial, serial w/ map and reduce, parallel) and answer the following questions:
>
> 1. Is there any performance difference between the different versions of the program?
> 2. Does user time significantly differ from real time for any of the versions of the program?
> 3. Experiment with different numbers of processes for the parallel example, starting with 1. What is the performance gain when you goal from 1 to 2 processes? From 2 to 3? When do you stop seeing improvement? (this will depend on your machine architecture)

15.7.7 References

- Map, Filter and Reduce
- multiprocessing API

15.8 Numpy

NumPy is a popular library on that is used by many other python librariessuch as pandas, and SciPy. It provides simple to use array operations for data. This helps to accass arrays in a more intuitive fashion and introduces various matrix operations.

We provide a short introduction to Numpy.

First we import the modules needed for this introduction and abreviate them with the `as` feature of the import statement

```
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
```

Now we showcase some features of Numpy.

15.8.1 Float Range

`arange()` is like `range()`, but for floating-point numbers.

```
X = np.arange(0.2,1,.1)

print (X)
```

We use this function to generate parameter space that can then be iterated on.

```
P = 10.0 ** np.arange(-7,1,1)

print (P)

for x,p in zip(X,P):
    print ('%f, %f' % (x, p))
```

15.8.2 Arrays

To create one dimensional arrays you use

```
a = np.array([1, 2, 3])
```

To check some properties you can use the following

```
print(type(a))            # Prints "<class 'numpy.ndarray'>"

print(a.shape)            # Prints "(3,)"
```

The shape indicates that in the first dimension, there are 3 elements. To print the actual values you can use

```
print(a)                  # Prints the values of the array
print(a[0], a[1], a[2])   # Prints "1 2 3"
```

To change values you can use the index of the element or use any other python method to do so. IN our example we change the first element to `42`

```
a[0] = 42
print(a)
```

To create more dimensional arrays you use

```
b = np.array([[1,2,3],[4,5,6]])    # Create a 2 dimensional array
print(b.shape)                     # Prints "(2, 3)"
print(b[0, 0], b[0, 1], b[1, 0])   # Prints "1 2 4"
```

15.8.3 Array Operations

Let us first create some arrays with a predefined datatype

```
x = np.array([[1,2],[3,4]], dtype=np.float64)
y = np.array([[5,6],[7,8]], dtype=np.float64)

print (x)

print(y)
```

To add the numbers use

```
print(x+y)
```

Other functions such as `-`, `*`, `/` behave as expected using elementwise oerations:

```
print(x-y)

print(x*y)

print(x/y)
```

To apply functions such as 'sin' make sure you use the function provided by the numpy package such as `np.sin`. The list of functions is included in the manual at * https://docs.scipy.org/doc/numpy/reference/routines.math.html

```
print (np.sin(x))

print (np.sum(x))
```

Computations can also be applied to columns and rows

```
print(np.sum(x, axis=0)) # sum of each column
print(np.sum(x, axis=1)) # sum of each row
```

15.8.4 Linear Algebra

Linear algebra methods are also provided.

```
from numpy import linalg

A = np.diag((1,2,3))

w,v = linalg.eig(A)

print ('w =', w)
print ('v =', v)
```

15.8.5 Resources

- https://docs.scipy.org/doc/numpy/
- http://cs231n.github.io/python-numpy-tutorial/#numpy

15.9 Scipy

SciPy is a library built around numpy and has a number of off-the-shelf algorithms and operations implemented. These include algorithms from calculus (such as integration), statistics, linear algebra, image-processing, signal processing, machine learning.

To achieve this, SciPy bundels a number of useful open-source software for mathematics, science, and engineering. It includes the following packages:

NumPy,
: for managing N-dimensional arrays

SciPy library,
: to access fundamental scientific computing capabilities

Matplotlib,
: to conduct 2D plotting

IPython,
: for an Interactive console (see jupyter)

Sympy,
: for symbolic mathematics

pandas,
: for providing data structures and analysis

15.9.1 Introduction

First we add the usual scientific computing modules with the typical abbreviations, including sp for scipy. We could invoke scipy's statistical package as sp.stats, but for the sake of laziness we abbreviate that too.

```
import numpy as np # import numpy
import scipy as sp # import scipy
from scipy import stats # refer directly to stats rather than sp.stats
import matplotlib as mpl # for visualization
from matplotlib import pyplot as plt # refer directly to pyplot
                                     # rather than mpl.pyplot
```

Now we create some random data to play with. We generate 100 samples from a Gaussian distribution centered at zero.

```
s = sp.randn(100)
```

How many elements are in the set?

```
print ('There are',len(s),'elements in the set')
```

What is the mean (average) of the set?

```
print ('The mean of the set is',s.mean())
```

What is the minimum of the set?

```
print ('The minimum of the set is',s.min())
```

What is the maximum of the set?

```
print ('The maximum of the set is',s.max())
```

We can use the scipy functions too. What's the median?

```
print ('The median of the set is',sp.median(s))
```

What about the standard deviation and variance?

```
print ('The standard deviation is',sp.std(s),
       'and the variance is',sp.var(s))
```

Isn't the variance the square of the standard deviation?

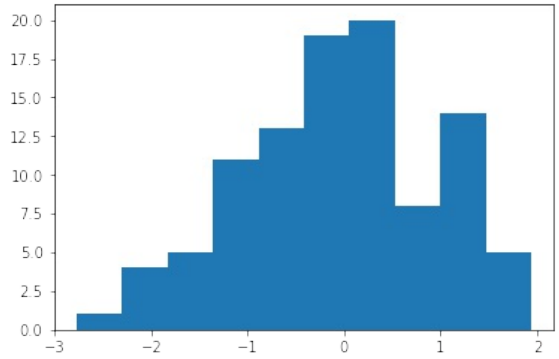
```
print ('The square of the standard deviation is',sp.std(s)**2)
```

How close are the measures? The differences are close as the following calculation shows

```
print ('The difference is',abs(sp.std(s)**2 - sp.var(s)))

print ('And in decimal form, the difference is %0.16f' %
       (abs(sp.std(s)**2 - sp.var(s))))
```

How does this look as a histogram?

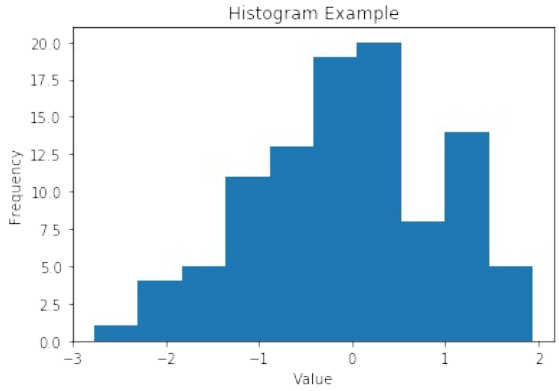
```
plt.hist(s) # yes, one line of code for a histogram
plt.show()
```



image

Let's add some titles.

```
plt.clf() # clear out the previous plot

plt.hist(s)
plt.title("Histogram Example")
plt.xlabel("Value")
plt.ylabel("Frequency")

plt.show()
```

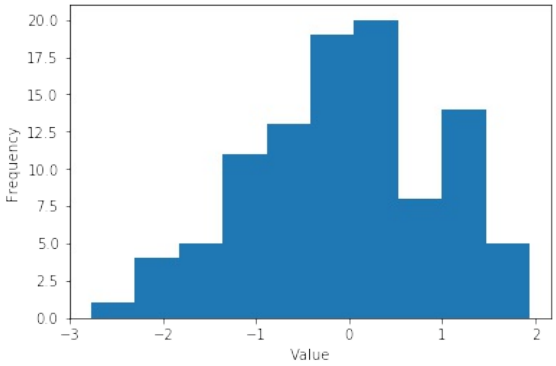


image

Typically we do not include titles when we prepare images for inclusion in LaTeX. There we use the caption to describe what the figure is about.

```
plt.clf() # clear out the previous plot
```

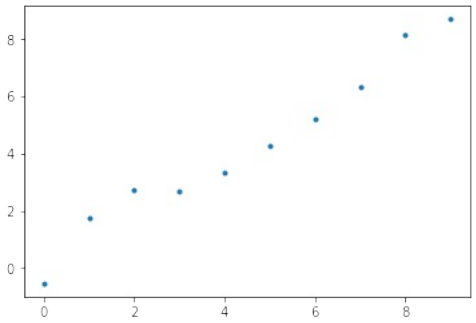
```
plt.hist(s)
plt.xlabel("Value")
plt.ylabel("Frequency")

plt.show()
```



image

Let's try out some linear regression, or curve fitting.

```
import random

def F(x):
    return 2*x - 2

def add_noise(x):
    return x + random.uniform(-1,1)

X = range(0,10,1)

Y = []
for i in range(len(X)):
    Y.append(add_noise(X[i]))

plt.clf() # clear out the old figure
plt.plot(X,Y,'.')
plt.show()
```



image

Now let's try linear regression to fit the curve.

```
m, b, r, p, est_std_err = stats.linregress(X,Y)
```

What is the slope and y-intercept of the fitted curve?

```
print ('The slope is',m,'and the y-intercept is', b)

def Fprime(x): # the fitted curve
    return m*x + b
```

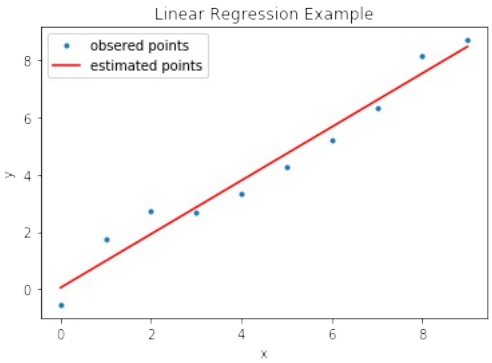
Now let's see how well the curve fits the data. We'll call the fitted curve *F'*.

```
X = range(0,10,1)

Yprime = []
for i in range(len(X)):
    Yprime.append(Fprime(X[i]))

plt.clf() # clear out the old figure

# the observed points, blue dots
plt.plot(X, Y, '.', label='observed points')

# the interpolated curve, connected red line
plt.plot(X, Yprime, 'r-', label='estimated points')

plt.title("Linear Regression Example") # title
plt.xlabel("x") # horizontal axis title
plt.ylabel("y") # vertical axis title
# legend labels to plot
plt.legend(['obsered points', 'estimated points'])

# comment out so that you can save the figure
#plt.show()
```

To save images into a PDF file for inclusion into LaTeX documents you can save the images as follows. Other formats such as png are also possible, but the quality is naturally not sufficient for inclusion in papers and documents. For that you certainly want to use PDF. The save of the figure has to occur before you use the `show()` command.

```
plt.savefig("regression.pdf", bbox_inches='tight')
```

```
plt.savefig('regression.png')

plt.show()
```



image

15.9.2 References

For more information about SciPy we recommend that you visit the following link

https://www.scipy.org/getting-started.html#learning-to-work-with-scipy

Additional material and inspiration for this section are from

- [O] "Getting Started guide" https://www.scipy.org/getting-started.html
- [O] Prasanth. "Simple statistics with SciPy." Comfort at 1 AU. February 28, 2011. https://oneau.wordpress.com/2011/02/28/simple-statistics-with-scipy/.
- [O] SciPy Cookbook. Lasted updated: 2015. http://scipy-cookbook.readthedocs.io/.

O create bibtex entries

15.10 Scikit-learn O

In this section we demonstrate how simple it is to use k-means in scikit learn.

15.10.1 Instalation

If you already have a working installation of numpy and scipy, the easiest way to install scikit-learn is using pip

```
$ pip install numpy
$ pip install scipy -U
$ pip install -U scikit-learn
```

15.10.2 Import

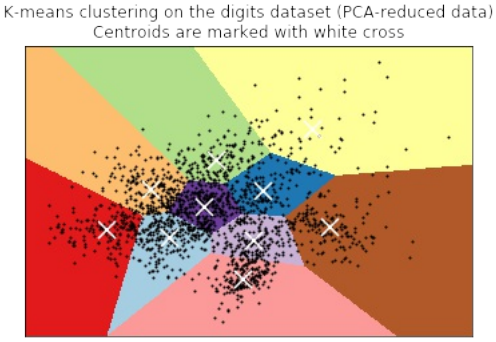
```
from time import time
import numpy as np
import matplotlib.pyplot as plt

from sklearn import metrics
from sklearn.cluster import KMeans
from sklearn.datasets import load_digits
from sklearn.decomposition import PCA
from sklearn.preprocessing import scale
```

15.10.3 Create samples

```
np.random.seed(42)

digits = load_digits()
data = scale(digits.data)

n_samples, n_features = data.shape
n_digits = len(np.unique(digits.target))
labels = digits.target

sample_size = 300

print("n_digits: %d, \t n_samples %d, \t n_features %d"
      % (n_digits, n_samples, n_features))


print(79 * '_')
print('% 9s' % 'init'
      '    time  inertia    homo   compl  v-meas     ARI AMI  silhouette')


def bench_k_means(estimator, name, data):
    t0 = time()
    estimator.fit(data)
    print('% 9s   %.2fs    %i   %.3f   %.3f   %.3f   %.3f   %.3f    %.3f'
          % (name, (time() - t0), estimator.inertia_,
             metrics.homogeneity_score(labels, estimator.labels_),
             metrics.completeness_score(labels, estimator.labels_),
             metrics.v_measure_score(labels, estimator.labels_),
             metrics.adjusted_rand_score(labels, estimator.labels_),
             metrics.adjusted_mutual_info_score(labels,  estimator.labels_),
             metrics.silhouette_score(data, estimator.labels_,
                                      metric='euclidean',
                                      sample_size=sample_size)))

bench_k_means(KMeans(init='k-means++', n_clusters=n_digits, n_init=10),
              name="k-means++", data=data)

bench_k_means(KMeans(init='random', n_clusters=n_digits, n_init=10),
              name="random", data=data)

# in this case the seeding of the centers is deterministic, hence we run the
# kmeans algorithm only once with n_init=1
pca = PCA(n_components=n_digits).fit(data)
bench_k_means(KMeans(init=pca.components_,
                     n_clusters=n_digits, n_init=1),
              name="PCA-based",
              data=data)
print(79 * '_')
```

15.11 Visualize

```
reduced_data = PCA(n_components=2).fit_transform(data)
kmeans = KMeans(init='k-means++', n_clusters=n_digits, n_init=10)
kmeans.fit(reduced_data)

# Step size of the mesh. Decrease to increase the quality of the VQ.
h = .02     # point in the mesh [x_min, x_max]x[y_min, y_max].

# Plot the decision boundary. For that, we will assign a color to each
x_min, x_max = reduced_data[:, 0].min() - 1, reduced_data[:, 0].max() + 1
y_min, y_max = reduced_data[:, 1].min() - 1, reduced_data[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))

# Obtain labels for each point in mesh. Use last trained model.
Z = kmeans.predict(np.c_[xx.ravel(), yy.ravel()])

# Put the result into a color plot
Z = Z.reshape(xx.shape)
plt.figure(1)
plt.clf()
plt.imshow(Z, interpolation='nearest',
           extent=(xx.min(), xx.max(), yy.min(), yy.max()),
           cmap=plt.cm.Paired,
           aspect='auto', origin='lower')

plt.plot(reduced_data[:, 0], reduced_data[:, 1], 'k.', markersize=2)
# Plot the centroids as a white X
centroids = kmeans.cluster_centers_
plt.scatter(centroids[:, 0], centroids[:, 1],
            marker='x', s=169, linewidths=3,
            color='w', zorder=10)
plt.title('K-means clustering on the digits dataset (PCA-reduced data)\n'
          'Centroids are marked with white cross')
plt.xlim(x_min, x_max)
plt.ylim(y_min, y_max)
plt.xticks(())
plt.yticks(())
plt.show()
```



image

15.12 Parallel Computing in Python

In this module we will review the available Python modules that can be used for parallel computing. The parallel computing can be in form of either multi-threading or multi-processing. In multi-threading approach, the threads run in the same shared memory heap whereas in case of multi-processing, the memory heaps of processes are separate and independent, therefore the communication between the processes are a little bit more complex.

15.12.1 Multi-threading in Python

Threading in Python is perfect for I/O operations where the process is expected to be idle regularly, e.g. web scraping. This is a very useful feature because several applications and script might spend the majority of their runtime on waiting for network or data I/O. In several cases, e.g. web scraping, the resources, i.e. downloading from different websites, are most of the time independent. Therefore the processor can download in parallel and join the result at the end.

15.12.1.1 `Thread` vs `Threading`

There are two built-in modules in Python that are related to threading, namely `thread` and `threading`. The former module is deprecated for sometime in `Python 2`, and in `Python 3` it is renamed to `_thread` for the sake of backwards incompatibilities. The `_thread` module provides low-level threading API for multi-threading in Python, whereas the module `threading` builds a high-level threading interface on top of it.

The `Thread()` is the main method of the `threading` module, the two important arguments of which are `target`, for specifying the callable object, and `args` to pass the arguments for the target callable. We illustrate these in the following example:

```
import threading

def hello_thread(thread_num):
    print ("Hello from Thread ", thread_num)

if __name__ == '__main__':
    for thread_num in range(5):
        t = threading.Thread(target=hello_thread,arg=(thread_num,))
        t.start()
```

This is the output of the previous example:

```
In [1]: %run threading.py
Hello from Thread  0
Hello from Thread  1
Hello from Thread  2
Hello from Thread  3
Hello from Thread  4
```

In case you are not familiar with the `if __name__ == '__main__:'` statement, what it does is basically making sure that the code nested under this condition will be run only if you run your module as a program and it will not run in case your module is imported in another file.

15.12.1.2 Locks

As mentioned prior, the memory space is shared between the threads. This is at the same time beneficial and problematic: it is beneficial in a sense that the communication between the threads becomes easy, however, you might experience strange outcome if you let several threads change same variable without caution, e.g. thread 2 changes variable `x` while thread 1 is working with it. This is when `lock` comes into play. Using `lock`, you can allow only one thread to work with a variable. In other words, only a single thread can hold the `lock`. If the other threads need to work with that variable, they have to wait until the other thread is done and the variable is "unlocked".

We illustrate this with a simple example:

```
import threading
```

```
global counter
counter = 0

def incrementer1():
    global counter
    for j in range(2):
        for i in range(3):
            counter += 1
            print("Greeter 1 incremented the counter by 1")
        print ("Counter is %d"%counter)

def incrementer2():
    global counter
    for j in range(2):
        for i in range(3):
            counter += 1
            print("Greeter 2 incremented the counter by 1")
        print ("Counter is now %d"%counter)


if __name__ == '__main__':
    t1 = threading.Thread(target = incrementer1)
    t2 = threading.Thread(target = incrementer2)

    t1.start()
    t2.start()
```

Suppose we want to print multiples of 3 between 1 and 12, i.e. 3, 6, 9 and 12. For the sake of argument, we try to do this using 2 threads and a nested for loop. Then we create a global variable called counter and we initialize it with 0. Then whenever each of the `incrementer1` or `incrementer2` functions are called, the `counter` is incremented by 3 twice (counter is incremented by 6 in each function call). If you run the previous code, you should be really lucky if you get the following as part of your output:

`bash Counter is now 3 Counter is now 6 Counter is now 9 Counter is now 12` The reason is the conflict that happens between threads while incrementing the `counter` in the nested for loop. As you probably noticed, the first level for loop is equivalent of adding 3 to the counter and the conflict that might happen is not effective on that level but the nested for loop. Accordingly, the output of the previous code is different in every run. This is an example output:

```
$ python3 lock_example.py
Greeter 1 incremented the counter by 1
Greeter 1 incremented the counter by 1
Greeter 1 incremented the counter by 1
Counter is 4
Greeter 2 incremented the counter by 1
Greeter 2 incremented the counter by 1
Greeter 1 incremented the counter by 1
Greeter 2 incremented the counter by 1
Greeter 1 incremented the counter by 1
Counter is 8
Greeter 1 incremented the counter by 1
Greeter 2 incremented the counter by 1
Counter is 10
Greeter 2 incremented the counter by 1
Greeter 2 incremented the counter by 1
Counter is 12
```

We can fix this issue using a `lock`: whenever one of the function is going to increment the value by 3, it will `acquire()` the lock and when it is done the function will `release()` the lock. This mechanism is illustrated in the following code:

```
import threading

increment_by_3_lock = threading.Lock()

global counter
counter = 0

def incrementer1():
    global counter
    for j in range(2):
        increment_by_3_lock.acquire(True)
        for i in range(3):
            counter += 1
            print("Greeter 1 incremented the counter by 1")
        print ("Counter is %d"%counter)
        increment_by_3_lock.release()

def incrementer2():
    global counter
    for j in range(2):
        increment_by_3_lock.acquire(True)
        for i in range(3):
            counter += 1
            print("Greeter 2 incremented the counter by 1")
        print ("Counter is %d"%counter)
        increment_by_3_lock.release()

if __name__ == '__main__':
    t1 = threading.Thread(target = incrementer1)
    t2 = threading.Thread(target = incrementer2)

    t1.start()
    t2.start()
```

No matter how many times you run this code, the output would always be in the correct order:

```
$ python3 lock_example.py
Greeter 1 incremented the counter by 1
Greeter 1 incremented the counter by 1
Greeter 1 incremented the counter by 1
Counter is 3
Greeter 1 incremented the counter by 1
Greeter 1 incremented the counter by 1
Greeter 1 incremented the counter by 1
Counter is 6
Greeter 2 incremented the counter by 1
Greeter 2 incremented the counter by 1
Greeter 2 incremented the counter by 1
Counter is 9
Greeter 2 incremented the counter by 1
Greeter 2 incremented the counter by 1
Greeter 2 incremented the counter by 1
Counter is 12
```

Using the `Threading` module increases both the overhead associated with thread management as well as the complexity of the program and that is why in many situations, employing `multiprocessing` module might be a better approach.

15.12.2 Multi-processing in Python

We already mentioned that multi-threading might not be sufficient in many applications and we might need to use `multiprocessing` sometime, or better to say most of the times. That is why we are dedicating this subsection to this particular module. This module provides you with an API for spawning processes the way you spawn threads using `threading` module. Moreover, there are some functionalities that are not even available in `threading` module, e.g. the `Pool` class which allows you to run a batch of jobs using a pool of worker processes.

15.12.2.1 Process

Similar to `threading` module which was employing `thread` (aka `_thread`) under the hood, `multiprocessing` employs the `Process` class. Consider the following example:

```
from multiprocessing import Process
```

```
import os

def greeter (name):
    proc_idx = os.getpid()
    print ("Process {0}: Hello {1}!".format(proc_idx,name))

if __name__ == '__main__':
    name_list = ['Harry', 'George', 'Dirk', 'David']
    process_list = []
    for name_idx, name in enumerate(name_list):
        current_process = Process(target=greeter, args=(name,))
        process_list.append(current_process)
        current_process.start()
    for process in process_list:
        process.join()
```

In this example, after importing the `Process` module we created a `greeter()` function that takes a `name` and greets that person. It also prints the `pid` (process identifier) of the process that is running it. Note that we used the `os` module to get the `pid`. In the bottom of the code after checking the `__name__='__main__'` condition, we create a series of `Process`es and `start` them. Finally in the last for loop and using the `join` method, we tell Python to wait for the processes to terminate. This is one of the possible outputs of the code:

```
$ python3 process_example.py
Process 23451: Hello Harry!
Process 23452: Hello George!
Process 23453: Hello Dirk!
Process 23454: Hello David!
```

15.12.2.2 Pool

Consider the `Pool` class as a pool of worker processes. There are several ways for assigning jobs to the `Pool` class and we will introduce the most important ones in this section. These methods are categorized as `blocking` or `non-blocking`. The former means that after calling the API, it blocks the thread/process until it has the result or answer ready and the control returns only when the call completes. In the `non-blockin` on the other hand, the control returns immediately.

15.12.2.2.1 Synchronous `Pool.map()`

We illustrate the `Pool.map` method by re-implementing our previous greeter example using `Pool.map`:

`python from multiprocessing import Pool import os

def greeter(name): pid = os.getpid() print("Process {0}: Hello {1}!".format(pid,name))

if **name** == '**main**': names = ['Jenna', 'David','Marry', 'Ted','Jerry','Tom','Justin'] pool = Pool(processes=3) sync_map = pool.map(greeter,names) print("Done!") `

As you can see, we have seven names here but we do not want to dedicate each greeting to a separate process. Instead we do the whole job of "greeting seven people" using "two processes". We create a pool of 3 processes with `Pool(processes=3)` syntax and then we map an iterable called `names` to the `greeter` function using `pool.map(greeter,names)`. As we expected, the greetings in the output will be printed from three different processes:

```
$ python poolmap_example.py
Process 30585: Hello Jenna!
Process 30586: Hello David!
Process 30587: Hello Marry!
Process 30585: Hello Ted!
Process 30585: Hello Jerry!
Process 30587: Hello Tom!
Process 30585: Hello Justin!
Done!
```

Note that `Pool.map()` is in `blocking` category and does not return the control to your script until it is done calculating the results. That is why `Done!` is printed after all of the greetings are over.

15.12.2.2.2 Asynchronous `Pool.map_async()`

As the name implies, you can use the `map_async` method, when you want assign many function calls to a pool of worker processes asynchronously. Note that unlike `map`, the order of the results is not guaranteed (as oppose to `map`) and the control is returned immediately. We now implement the previous example using `map_async`:

```
from multiprocessing import Pool
import os

def greeter(name):
    pid = os.getpid()
    print("Process {0}: Hello {1}!".format(pid,name))

if __name__ == '__main__':
    names = ['Jenna', 'David','Marry', 'Ted','Jerry','Tom','Justin']
    pool = Pool(processes=3)
    async_map = pool.map_async(greeter,names)
    print("Done!")
    async_map.wait()
```

As you probably noticed, the only difference (clearly apart from the `map_async` method name) is calling the `wait()` method in the last line. The `wait()` method tells your script to wait for the result of `map_async` before terminating:

```
$ python poolmap_example.py
Done!
Process 30740: Hello Jenna!
Process 30741: Hello David!
Process 30740: Hello Ted!
Process 30742: Hello Marry!
Process 30740: Hello Jerry!
Process 30741: Hello Tom!
Process 30742: Hello Justin!
```

Note that the order of the results are not preserved. Moreover, `Done!` is printer before any of the results, meaning that if we do not use the `wait()` method, you probably will not see the result at all.

15.12.2.3 Locks

The way `multiprocessing` module implements locks is almost identical to the way the `threading` module does. After importing `Lock` from `multiprocessing` all you need to do is to `acquire` it, do some computation and then `release` the lock. We will clarify the use of `Lock` by providing an example in next section about process communication.

15.12.2.4 Process Communication

Process communication in `multiprocessing` is one of the most important, yet complicated, features for better use of this module. As oppose to `threading`, the `Process` objects will not have access to any shared variable by default, i.e. no shared memory space between the processes by default. This effect is illustrated in the following example:

```
from multiprocessing import Process, Lock, Value
import time

global counter
counter = 0

def incrementer1():
    global counter
    for j in range(2):
        for i in range(3):
            counter += 1
        print ("Greeter1: Counter is %d"%counter)

def incrementer2():
    global counter
    for j in range(2):
        for i in range(3):
            counter += 1
        print ("Greeter2: Counter is %d"%counter)
```

```
if __name__ == '__main__':

    t1 = Process(target = incrementer1 )
    t2 = Process(target = incrementer2 )
    t1.start()
    t2.start()
```

Probably you already noticed that this is almost identical to our example in `threading` section. Now, take a look at the strange output:

```
$ python communication_example.py
Greeter1: Counter is 3
Greeter1: Counter is 6
Greeter2: Counter is 3
Greeter2: Counter is 6
```

As you can see, it is as if the processes does not see each other. Instead of having two processes one counting to 6 and the other counting from 6 to 12, we have two processes counting to 6.

Nevertheless, there are several ways that `Process`es from `multiprocessing` can communicate with each other, including `Pipe`, `Queue`, `Value`, `Array` and `Manager`. `Pipe` and `Queue` are appropriate for inter-process message passing. To be more specific, `Pipe` is useful for process-to-process scenarios while `Queue` is more appropriate for process**es**-toprocess**es** ones. `Value` and `Array` are both used to provide a synchronized access to a shared data (very much like shared memory) and `Managers` can be used on different data types. In the following sub-sections, we cover both `Value` and `Array` since they are both lightweight, yet useful, approaches.

15.12.2.4.1 Value

The following example re-implements the broken example in the previous section. We fix the strange output, by using both `Lock` and `Value`:

```
from multiprocessing import Process, Lock, Value
import time

increment_by_3_lock = Lock()


def incrementer1(counter):
    for j in range(3):
        increment_by_3_lock.acquire(True)
        for i in range(3):
            counter.value += 1
            time.sleep(0.1)
        print ("Greeter1: Counter is %d"%counter.value)
        increment_by_3_lock.release()

def incrementer2(counter):
    for j in range(3):
        increment_by_3_lock.acquire(True)
        for i in range(3):
            counter.value += 1
            time.sleep(0.05)
        print ("Greeter2: Counter is %d"%counter.value)
        increment_by_3_lock.release()


if __name__ == '__main__':

    counter = Value('i',0)
    t1 = Process(target = incrementer1, args=(counter,))
    t2 = Process(target = incrementer2 , args=(counter,))
    t2.start()
    t1.start()
```

The usage of `Lock` object in this example is identical to the example in `threading` section. The usage of `counter` is on the other hand the novel part. First, note that counter is not a global variable anymore and instead it is a `Value` which returns a `ctypes` object allocated from a shared memory between the processes. The first argument `'i'` indicates a signed integer, and the second argument defines the initialization value. In this case we are assigning a signed integer in the shared memory initialized to size 0 to the `counter` variable. We then modified our two functions and pass this shared variable as an argument. Finally, we change the way we increment the `counter` since counter is not an Python integer anymore but a `ctypes` signed integer where we can access its value using the `value` attribute. The output of the code is now as we expected:

```
$ python mp_lock_example.py
Greeter2: Counter is 3
Greeter2: Counter is 6
Greeter1: Counter is 9
Greeter1: Counter is 12
```

The last example related to parallel processing, illustrates the use of both `Value` and `Array`, as well as a technique to pass multiple arguments to a function. Note that the `Process` object does not accept multiple arguments for a function and therefore we need this or similar techniques for passing multiple arguments. Also, this technique can also be used when you want to pass multiple arguments to `map` or `map_async`:

```
from multiprocessing import Process, Lock, Value, Array
import time
from ctypes import c_char_p


increment_by_3_lock = Lock()


def incrementer1(counter_and_names):
    counter=  counter_and_names[0]
    names = counter_and_names[1]
    for j in range(2):
        increment_by_3_lock.acquire(True)
        for i in range(3):
            counter.value += 1
            time.sleep(0.1)
        name_idx = counter.value//3 -1
        print ("Greeter1: Greeting {0}! Counter is {1}".format(names.value[name_idx],counter.value))
        increment_by_3_lock.release()

def incrementer2(counter_and_names):
    counter=  counter_and_names[0]
    names = counter_and_names[1]
    for j in range(2):
        increment_by_3_lock.acquire(True)
        for i in range(3):
            counter.value += 1
            time.sleep(0.05)
        name_idx = counter.value//3 -1
        print ("Greeter2: Greeting {0}! Counter is {1}".format(names.value[name_idx],counter.value))
        increment_by_3_lock.release()


if __name__ == '__main__':
    counter = Value('i',0)
    names = Array (c_char_p,4)
    names.value = ['James','Tom','Sam', 'Larry']
    t1 = Process(target = incrementer1, args=((counter,names),))
    t2 = Process(target = incrementer2 , args=((counter,names),))
    t2.start()
    t1.start()
```

In this example we created a `multiprocessing.Array()` object and assigned it to a variable called `names`. As we mentioned before, the first argument is the `ctype` data type and since we want to create an array of strings with length of 4 (second argument), we imported the `c_char_p` and passed it as the first argument.

Instead of passing the arguments separately, we merged both the `Value` and `Array` objects in a tuple and passed the tuple to the functions. We then modified the functions to unpack the objects in the first two lines in the both functions. Finally we changed the print statement in a way that each process greets a particular name. The output of the example is:

```
$ python3 mp_lock_example.py
Greeter2: Greeting James! Counter is 3
Greeter2: Greeting Tom! Counter is 6
Greeter1: Greeting Sam! Counter is 9
Greeter1: Greeting Larry! Counter is 12
```

15.13 Dask

Dask is a python-based parallel computing library for analytics. Parallel computing is a type of computation in which many calculations or the execution of processes are carried out simultaneously. Large problems can often be divided into smaller ones, which can then be solved concurrently.

Dask is composed of two components:

1. Dynamic task scheduling optimized for computation. This is similar to Airflow, Luigi, Celery, or Make, but optimized for interactive computational workloads.
2. Big Data collections like parallel arrays, dataframes, and lists that extend common interfaces like NumPy, Pandas, or Python iterators to larger-than-memory or distributed environments. These parallel collections run on top of the dynamic task schedulers.

Dask emphasizes the following virtues:

- Familiar: Provides parallelized NumPy array and Pandas DataFrame objects.
- Flexible: Provides a task scheduling interface for more custom workloads and integration with other projects.
- Native: Enables distributed computing in Pure Python with access to the PyData stack.
- Fast: Operates with low overhead, low latency, and minimal serialization necessary for fast numerical algorithms
- Scales up: Runs resiliently on clusters with 1000s of cores
- Scales down: Trivial to set up and run on a laptop in a single process
- Responsive: Designed with interactive computing in mind it provides rapid feedback and diagnostics to aid humans

The section is structured in a number of subsections addressing the following topics:

Foundations:

an explanation of what Dask is, how it works, and how to use lower level primitives to set up computations. Casual users may wish to skip this section, although we consider it useful knowledge for all users.

Distributed Features:

information on running Dask on the distributed scheduler, which enables scale-up to distributed settings and enhanced monitoring of task operations. The distributed scheduler is now generally the recommended engine for executing task work, even on single workstations or laptops.

Collections:

convenient abstractions giving a familiar feel to big data.

Bags:

Python iterators with a functional paradigm, such as found in func/iter-tools and toolz - generalize lists/generators to big data; this will seem very familiar to users of PySpark's RDD

Array:

massive multi-dimensional numerical data, with Numpy functionality

Dataframe:

massive tabular data, with Pandas functionality

15.13.1 How Dask Works

Dask is computation tool for larger-than-memory datasets, parallel execution or delayed/background execution.

We can summarize the basics of Dask as follows:

- process data that does not fit into memory by breaking it into blocks and specifying task chains
- parallelize execution of tasks across cores and even nodes of a cluster
- move computation to the data rather than the other way around, to minimize communication overheads

We use for-loops to build basic tasks, Python iterators, and the Numpy (array) and Pandas (dataframe) functions for multi-dimensional or tabular data, respectively.

Dask allows us to construct a prescription for the calculation we want to carry out. A module named Dask.delayed lets us parallelize custom code. It is useful whenever our problem doesn't quite fit a high-level parallel object like dask.array or dask.dataframe but could still benefit from parallelism. Dask.delayed works by delaying our function evaluations and putting them into a dask graph. Here is a small example:

```
from dask import delayed

@delayed
def inc(x):
    return x + 1

@delayed
def add(x, y):
    return x + y
```

Here we have used the delayed annotation to show that we want these functions to operate lazily - to save the set of inputs and execute only on demand.

15.13.2 Dask Bag

Dask-bag excels in processing data that can be represented as a sequence of arbitrary inputs. We'll refer to this as "messy" data, because it can contain complex nested structures, missing fields, mixtures of data types, etc. The functional programming style fits very nicely with standard Python iteration, such as can be found in the itertools module.

Messy data is often encountered at the beginning of data processing pipelines when large volumes of raw data are first consumed. The initial set of data might be JSON, CSV, XML, or any other format that does not enforce strict structure and datatypes. For this reason, the initial data massaging and processing is often done with Python lists, dicts, and sets.

These core data structures are optimized for general-purpose storage and processing. Adding streaming computation with iterators/generator expressions or libraries like itertools or toolz let us process large volumes in a small space. If we combine this with parallel processing then we can churn through a fair amount of data.

Dask.bag is a high level Dask collection to automate common workloads of this form. In a nutshell

```
dask.bag = map, filter, toolz + parallel execution
```

You can create a Bag from a Python sequence, from files, from data on S3, etc..

```
# each element is an integer
import dask.bag as db
b = db.from_sequence([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])

# each element is a text file of JSON lines
import os
b = db.read_text(os.path.join('data', 'accounts.*.json.gz'))

# Requires `s3fs` library
# each element is a remote CSV text file
b = db.read_text('s3://dask-data/nyc-taxi/2015/yellow_tripdata_2015-01.csv')
```

Bag objects hold the standard functional API found in projects like the Python standard library, toolz, or pyspark, including map, filter, groupby, etc..

As with Array and DataFrame objects, operations on Bag objects create new bags. Call the .compute() method to trigger execution.

```
def is_even(n):
    return n % 2 == 0

b = db.from_sequence([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
c = b.filter(is_even).map(lambda x: x ** 2)
c

# blocking form: wait for completion (which is very fast in this case)
```

```
c.compute()
```

For more details on Dask Bag check https://dask.pydata.org/en/latest/bag.html

15.13.3 Concurrency Features

Dask supports a real-time task framework that extends Python's concurrent.futures interface. This interface is good for arbitrary task scheduling, like dask.delayed, but is immediate rather than lazy, which provides some more flexibility in situations where the computations may evolve over time. These features depend on the second generation task scheduler found in dask.distributed (which, despite its name, runs very well on a single machine).

Dask allows us to simply construct graphs of tasks with dependencies. We can find that graphs can also be created automatically for us using functional, Numpy or Pandas syntax on data collections. None of this would be very useful, if there weren't also a way to execute these graphs, in a parallel and memory-aware way. Dask comes with four available schedulers:

- `dask.threaded.get`: a scheduler backed by a thread pool
- `dask.multiprocessing.get`: a scheduler backed by a process pool
- `dask.async.get_sync`: a synchronous scheduler, good for debugging
- `distributed.Client.get`: a distributed scheduler for executing graphs on multiple machines.

Here is a simple program for dask.distributed library:

```
from dask.distributed import Client
client = Client('scheduler:port')

futures = []
for fn in filenames:
    future = client.submit(load, fn)
    futures.append(future)

summary = client.submit(summarize, futures)
summary.result()
```

For more details on Concurrent Features by Dask check https://dask.pydata.org/en/latest/futures.html

15.13.4 Dask Array

Dask arrays implement a subset of the NumPy interface on large arrays using blocked algorithms and task scheduling. These behave like numpy arrays, but break a massive job into tasks that are then executed by a scheduler. The default scheduler uses threading but you can also use multiprocessing or distributed or even serial processing (mainly for debugging). You can tell the dask array how to break the data into chunks for processing.

```
import dask.array as da
f = h5py.File('myfile.hdf5')
x = da.from_array(f['/big-data'], chunks=(1000, 1000))
x - x.mean(axis=1).compute()
```

For more details on Dask Array check https://dask.pydata.org/en/latest/array.html

15.13.5 Dask DataFrame

A Dask DataFrame is a large parallel dataframe composed of many smaller Pandas dataframes, split along the index. These pandas dataframes may live on disk for larger-than-memory computing on a single machine, or on many different machines in a cluster. Dask.dataframe implements a commonly used subset of the Pandas interface including elementwise operations, reductions, grouping operations, joins, timeseries algorithms, and more. It copies the Pandas interface for these operations exactly and so should be very familiar to Pandas users. Because Dask.dataframe operations merely coordinate Pandas operations they usually exhibit similar performance characteristics as are found in Pandas. To run the following code, save 'student.csv' file in your machine.

```
import pandas as pd
df = pd.read_csv('student.csv')
d = df.groupby(df.HID).Serial_No.mean()
print(d)
```

```
ID
101     1
102     2
104     3
105     4
106     5
107     6
109     7
111     8
201     9
202    10
Name: Serial_No, dtype: int64
```

```
import dask.dataframe as dd
df = dd.read_csv('student.csv')
dt = df.groupby(df.HID).Serial_No.mean().compute()
print (dt)
```

```
ID
101     1.0
102     2.0
104     3.0
105     4.0
106     5.0
107     6.0
109     7.0
111     8.0
201     9.0
202    10.0
Name: Serial_No, dtype: float64
```

For more details on Dask DataFrame check https://dask.pydata.org/en/latest/dataframe.html

15.13.6 Dask DataFrame Storage

Efficient storage can dramatically improve performance, particularly when operating repeatedly from disk.

Decompressing text and parsing CSV files is expensive. One of the most effective strategies with medium data is to use a binary storage format like HDF5.

```
# be sure to shut down other kernels running distributed clients
from dask.distributed import Client
client = Client()
```

Create data if we don't have any

```
from prep import accounts_csvs
accounts_csvs(3, 1000000, 500)
```

First we read our csv data as before.

CSV and other text-based file formats are the most common storage for data from many sources, because they require minimal pre-processing, can be written line-by-line and are human-readable. Since Pandas' read_csv is well-optimized, CSVs are a reasonable input, but far from optimized, since reading required extensive text parsing.

```
import os
filename = os.path.join('data', 'accounts.*.csv')
filename

import dask.dataframe as dd
df_csv = dd.read_csv(filename)
df_csv.head()
```

HDF5 and netCDF are binary array formats very commonly used in the scientific realm.

Pandas contains a specialized HDF5 format, HDFStore. The dd.DataFrame.to_hdf method works exactly like the pd.DataFrame.to_hdf method.

```
target = os.path.join('data', 'accounts.h5')
target
```

```
%time df_csv.to_hdf(target, '/data')

df_hdf = dd.read_hdf(target, '/data')
df_hdf.head()
```

For more information of Dask DataFrame Storage, click http://dask.pydata.org/en/latest/dataframe-create.html

15.13.7 Links

- https://dask.pydata.org/en/latest/
- http://matthewrocklin.com/blog/work/2017/10/16/streaming-dataframes-1
- http://people.duke.edu/~ccc14/sta-663-2017/18A_Dask.html
- https://www.kdnuggets.com/2016/09/introducing-dask-parallel-programming.html
- https://pypi.python.org/pypi/dask/
- https://www.hdfgroup.org/2015/03/hdf5-as-a-zero-configuration-ad-hoc-scientific-database-for-python/
- https://github.com/dask/dask-tutorial

15.14 Dask - Random Forest Feature Detection

15.14.1 Setup

First we need our tools. pandas gives us the DataFrame, very similar to R's DataFrames. The DataFrame is a structure that allows us to work with our data more easily. It has nice features for slicing and transformation of data, and easy ways to do basic statistics.

numpy has some very handy functions that work on DataFrames.

15.14.2 Dataset

We are using a dataset about the wine quality dataset, archived at UCI's Machine Learning Repository (http://archive.ics.uci.edu/ml/index.php).

```
import pandas as pd
import numpy as np
```

Now we'll load our data. pandas makes it easy!

```
# red wine quality data, packed in a DataFrame
red_df = pd.read_csv('winequality-red.csv',sep=';',header=0, index_col=False)

# white wine quality data, packed in a DataFrame
white_df = pd.read_csv('winequality-white.csv',sep=';',header=0,index_col=False)

# rose? other fruit wines? plum wine? :(
```

Like in R, there is a .describe() method that gives basic statistics for every column in the dataset.

```
# for red wines
red_df.describe()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	1
count	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1599.000000	1
mean	8.319637	0.527821	0.270976	2.538806	0.087467	15.874922	4
std	1.741096	0.179060	0.194801	1.409928	0.047065	10.460157	3
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6
25%	7.100000	0.390000	0.090000	1.900000	0.070000	7.000000	2
50%	7.900000	0.520000	0.260000	2.200000	0.079000	14.000000	3
75%	9.200000	0.640000	0.420000	2.600000	0.090000	21.000000	6
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	2

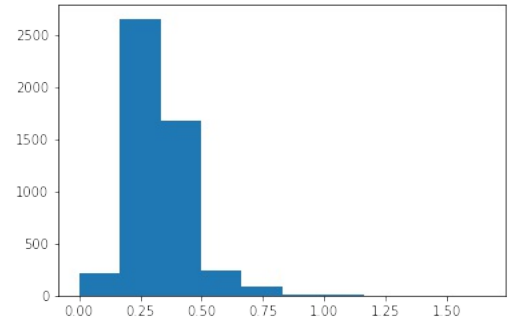
```
# for white wines
white_df.describe()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	4
count	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4898.000000	4
mean	6.854788	0.278241	0.334192	6.391415	0.045772	35.308085	1
std	0.843868	0.100795	0.121020	5.072058	0.021848	17.007137	4
min	3.800000	0.080000	0.000000	0.600000	0.009000	2.000000	9
25%	6.300000	0.210000	0.270000	1.700000	0.036000	23.000000	1
50%	6.800000	0.260000	0.320000	5.200000	0.043000	34.000000	1
75%	7.300000	0.320000	0.390000	9.900000	0.050000	46.000000	1
max	14.200000	1.100000	1.660000	65.800000	0.346000	289.000000	4

Sometimes it is easier to understand the data visually. A histogram of the white wine quality data citric acid samples is shown below. You can of course visualize other columns' data or other datasets. Just replace

the DataFrame and column name below.

```
import matplotlib.pyplot as plt

def extract_col(df,col_name):
    return list(df[col_name])

col = extract_col(white_df,'citric acid') # can replace with another dataframe or column
plt.hist(col)

#TODO: add axes and such to set a good example

plt.show()
```



png

15.14.3 Detecting Features

Let us try out a some elementary machine learning models. These models are not always for prediction. They are also useful to find what features are most predictive of a variable of interest. Depending on the classifier you use, you may need to transform the data pertaining to that variable.

15.14.3.1 Data Preparation

Let us assume we want to study what features are most correlated with pH. pH of course is real-valued, and continuous. The classifiers we want to use usually need labeled or integer data. Hence, we will transform the pH data, assigning wines with pH higher than average as `hi` (more basic or alkaline) and wines with pH lower than average as `lo` (more acidic).

```
# refresh to make Jupyter happy
red_df = pd.read_csv('winequality-red.csv',sep=';',header=0, index_col=False)
white_df = pd.read_csv('winequality-white.csv',sep=';',header=0,index_col=False)

#TODO: data cleansing functions here, e.g. replacement of NaN

# if the variable you want to predict is continuous, you can map ranges of values
# to integer/binary/string labels

# for example, map the pH data to 'hi' and 'lo' if a pH value is more than or
# less than the mean pH, respectively
M = np.mean(list(red_df['pH'])) # expect inelegant code in these mappings
Lf = lambda p: int(p < M)*'lo' + int(p >= M)*'hi' # some C-style hackery

# create the new classifiable variable
red_df['pH-hi-lo'] = map(Lf,list(red_df['pH']))

# and remove the predecessor
del red_df['pH']
```

Now we specify which dataset and variable you want to predict by assigning vlues to `SELECTED_DF` and `TARGET_VAR`, respectively.

We like to keep a parameter file where we specify data sources and such. This lets me create generic analytics code that is easy to reuse.

After we have specified what dataset we want to study, we split the training and test datasets. We then scale (normalize) the data, which makes most classifiers run better.

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import metrics

# make selections here without digging in code
SELECTED_DF = red_df # selected dataset
TARGET_VAR = 'pH-hi-lo' # the predicted variable

# generate nameless data structures
df = SELECTED_DF
target = np.array(df[TARGET_VAR]).ravel()
del df[TARGET_VAR] # no cheating

#TODO: data cleansing function calls here

# split datasets for training and testing
X_train, X_test, y_train, y_test = train_test_split(df,target,test_size=0.2)

# set up the scaler
scaler = StandardScaler()
scaler.fit(X_train)

# apply the scaler
X_train = scaler.transform(X_train)
X_test = scaler.transform(X_test)
```

Now we pick a classifier. As you can see, there are many to try out, and even more in scikit-learn's documentation and many examples and tutorials. Random Forests are data science workhorses. They are the go-to method for most data scientists. Be careful relying on them though–they tend to overfit. We try to avoid overfitting by separating the training and test datasets.

15.14.4 Random Forest

```
# pick a classifier

from sklearn.tree import DecisionTreeClassifier,DecisionTreeRegressor,ExtraTreeClassifier,ExtraTreeRegressor
from sklearn.ensemble import RandomForestClassifier,ExtraTreesClassifier

clf = RandomForestClassifier()
```

Now we will test it out with the default parameters.

Note that this code is boilerplate. You can use it interchangeably for most scikit-learn models.

```
# test it out

model = clf.fit(X_train,y_train)
pred = clf.predict(X_test)
```

```
conf_matrix = metrics.confusion_matrix(y_test,pred)

var_score = clf.score(X_test,y_test)

# the results
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1]
```

Now output the results. For Random Forests, we get a feature ranking. Relative importances usually exponentially decay. The first few highly-ranked features are usually the most important.

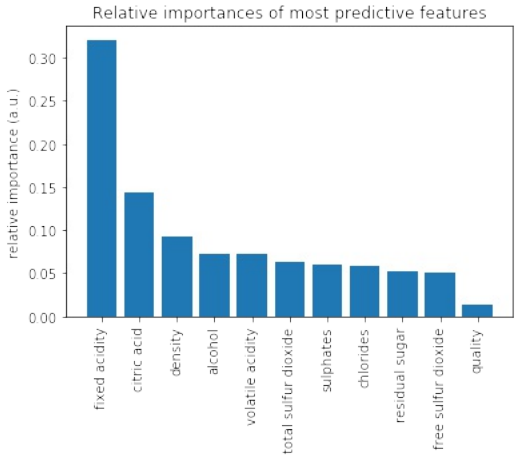
```
# for the sake of clarity
num_features = X_train.shape[1]
features = map(lambda x: df.columns[x],indices)
feature_importances = map(lambda x: importances[x],indices)

print 'Feature ranking:\n'

for i in range(num_features):
    feature_name = features[i]
    feature_importance = feature_importances[i]
    print '%s%f' % (feature_name.ljust(30), feature_importance)
```

Feature ranking:

fixed acidity 0.269778 citric acid 0.171337 density 0.089660 volatile acidity 0.088965 chlorides 0.082945 alcohol 0.080437 total sulfur dioxide 0.067832 sulphates 0.047786 free sulfur dioxide 0.042727 residual sugar 0.037459 quality 0.021075

Sometimes it's easier to visualize. We'll use a bar chart.

```
plt.clf()
plt.bar(range(num_features),feature_importances)
plt.xticks(range(num_features),features,rotation=90)
plt.ylabel('relative importance (a.u.)')
plt.title('Relative importances of most predictive features')
plt.show()
```



png

```
import dask.dataframe as dd

red_df = dd.read_csv('winequality-red.csv',sep=';',header=0)
white_df = dd.read_csv('winequality-white.csv',sep=';',header=0)
```

15.14.5 Acknowledgement

This notebook was developed by Juliette Zerick and Gregor von Laszewski

15.15 Fingerprint Matching

Python is a flexible and popular language for running data analysis pipelines. In this section we will implement a solution for a fingerprint matching.

15.15.1 Overview

Fingerprint recognition refers to the automated method for verifying a match between two fingerprints and that is used to identify individuals and verify their identity. Fingerprints (Figure 1) are the most widely used form of biometric used to identify individuals.

Fingerprints

The automated fingerprint matching generally required the detection of different fingerprint features (aggregate characteristics of ridges, and minutia points) and then the use of fingerprint matching algorithm, which can do both one-to- one and one-to- many matching operations. Based on the number of matches a proximity score (distance or similarity) can be calculated.

We use the following NIST dataset for the study:

Special Database 14 - NIST Mated Fingerprint Card Pairs 2. (http://www.nist.gov/itl/iad/ig/special_dbases.cfm)

15.15.2 Objectives

Match the fingerprint images from a probe set to a gallery set and report the match scores.

15.15.3 Prerequisites

For this work we will use the following algorithms:

- MINDTCT: The NIST minutiae detector, which automatically locates and records ridge ending and bifurcations in a fingerprint image. (http://www.nist.gov/itl/iad/ig/nbis.cfm)
- BOZORTH3: A NIST fingerprint matching algorithm, which is a minutiae based fingerprint-matching algorithm. It can do both one-to- one and one-to- many matching operations. (http://www.nist.gov/itl/iad/ig/nbis.cfm)

In order to follow along, you must have the NBIS tools which provide `mindtct` and `bozorth3` installed. If you are on Ubuntu 16.04 Xenial, the following steps will accomplish this:

```
$ sudo apt-get update -qq
$ sudo apt-get install -y build-essential cmake unzip
$ wget "http://nigos.nist.gov:8080/nist/nbis/nbis_v5_0_0.zip"
$ unzip -d nbis nbis_v5_0_0.zip
$ cd nbis/Rel_5.0.0
$ ./setup.sh /usr/local --without-X11
$ sudo make
```

15.15.4 Implementation

1. Fetch the fingerprint images from the web
2. Call out to external programs to prepare and compute the match scoreds
3. Store the results in a database
4. Generate a plot to identify likely matches.

```
from __future__ import print_function
```

```
import urllib
import zipfile
import hashlib
```

We'll be interacting with the operating system and manipulating files and their pathnames.

```
import os.path
import os
import sys
import shutil
import tempfile
```

Some general usefull utilities

```
import itertools
import functools
import types
from pprint import pprint
```

Using the `attrs` library provides some nice shortcuts to defining objects

```
import attr
```

```
import sys
```

We'll be randomly dividing the entire dataset, based on user input, into the probe and gallery stets

```
import random
```

We'll need to call out to the NBIS software. We'll also be using multiple processes to take advantage of all the cores on our machine

```
import subprocess
import multiprocessing
```

As for plotting, we'll use `matplotlib`, though there are many alternatives.

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

Finally, we'll write the results to a database.

```
import sqlite3
```

15.15.5 Utility functions

Next, we'll define some utility functions:

```
def take(n, iterable):
    "Returns a generator of the first **n** elements of an iterable"
    return itertools.islice(iterable, n )


def zipWith(function, *iterables):
    "Zip a set of **iterables** together and apply **function** to each tuple"
    for group in itertools.izip(*iterables):
        yield function(*group)


def uncurry(function):
    "Transforms an N-arry **function** so that it accepts a single parameter of an N-tuple"
    @functools.wraps(function)
    def wrapper(args):
        return function(*args)
    return wrapper


def fetch_url(url, sha256, prefix='.', checksum_blocksize=2**20, dryRun=False):
    """Download a url.

    :param url: the url to the file on the web
    :param sha256: the SHA-256 checksum. Used to determine if the file was previously downloaded.
    :param prefix: directory to save the file
    :param checksum_blocksize: blocksize to used when computing the checksum
    :param dryRun: boolean indicating that calling this function should do nothing
    :returns: the local path to the downloaded file
    :rtype:

    """

    if not os.path.exists(prefix):
        os.makedirs(prefix)

    local = os.path.join(prefix, os.path.basename(url))

    if dryRun: return local

    if os.path.exists(local):
        print ('Verifying checksum')
        chk = hashlib.sha256()
        with open(local, 'rb') as fd:
            while True:
                bits = fd.read(checksum_blocksize)
```

```
                if not bits: break
                chk.update(bits)
        if sha256 == chk.hexdigest():
            return local

    print ('Downloading', url)

    def report(sofar, blocksize, totalsize):
        msg = '{}%\r'.format(100 * sofar * blocksize / totalsize, 100)
        sys.stderr.write(msg)

    urllib.urlretrieve(url, local, report)

    return local
```

15.15.6 Dataset

We'll now define some global parameters

First, the fingerprint dataset

```
DATASET_URL = 'https://s3.amazonaws.com/nist-srd/SD4/NISTSpecialDatabase4GrayScaleImagesofFIGS.zip'
DATASET_SHA256 = '4db6a8f3f9dc14c504180cbf67cdf35167a109280f121c901be37a80ac13c449'
```

We'll define how to download the dataset. This function is general enough that it could be used to retrieve most files, but we'll default it to use the values from above.

```
def prepare_dataset(url=None, sha256=None, prefix='.', skip=False):
    url = url or DATASET_URL
    sha256 = sha256 or DATASET_SHA256
    local = fetch_url(url, sha256=sha256, prefix=prefix, dryRun=skip)

    if not skip:
        print ('Extracting', local, 'to', prefix)
        with zipfile.ZipFile(local, 'r') as zip:
            zip.extractall(prefix)

    name, _ = os.path.splitext(local)
    return name


def locate_paths(path_md5list, prefix):
    with open(path_md5list) as fd:
        for line in itertools.imap(str.strip, fd):
            parts = line.split()
            if not len(parts) == 2: continue
            md5sum, path = parts
            chksum = Checksum(value=md5sum, kind='md5')
            filepath = os.path.join(prefix, path)
            yield Path(checksum=chksum, filepath=filepath)


def locate_images(paths):

    def predicate(path):
        _, ext = os.path.splitext(path.filepath)
        return ext in ['.png']

    for path in itertools.ifilter(predicate, paths):
        yield image(id=path.checksum.value, path=path)
```

15.15.7 Data Model

We'll define some classes so we have a nice API for working with the dataflow. We set `slots=True` so that the resulting objects will be more space-efficient.

15.15.7.1 Utilities

15.15.7.1.1 Checksum

The checksum consists of the actual hash value (`value`) as well as a string representing the hashing algorithm. The validator enforces that the algorith can only be one of the listed acceptable methods

```
@attr.s(slots=True)
class Checksum(object):
  value = attr.ib()
  kind = attr.ib(validator=lambda o, a, v: v in 'md5 sha1 sha224 sha256 sha384 sha512'.split())
```

15.15.7.1.2 Path

`Path`s refer to an image's filepath and associated `Checksum`. We get the checksum "for "free" since the MD5 hash is provided for each image in the dataset.

```
@attr.s(slots=True)
class Path(object):
    checksum = attr.ib()
    filepath = attr.ib()
```

15.15.7.1.3 Image

The start of the data pipeline is the image. An `image` has an `id` (the md5 hash) and the path to the image.

```
@attr.s(slots=True)
class image(object):
    id = attr.ib()
    path = attr.ib()
```

15.15.7.2 Mindtct

The next step in the pipeline is to apply the `mindtct` program from NBIS. A `mindtct` object therefore represents the results of applying `mindtct` on an `image`. The `xyt` output is needed fo r the next step, and the `image` attribute represents the image id.

```
@attr.s(slots=True)
class mindtct(object):
    image = attr.ib()
    xyt = attr.ib()

    def pretty(self):
        d = dict(id=self.image.id, path=self.image.path)
        return pprint(d)
```

We need a way to construct a `mindtct` object from an `image` object. A straightforward way of doing this would be to have a `from_image` `@staticmethod` or `@classmethod`, but that doesn't work well with `multiprocessing` as top-level functions work best as they need to be serialized.

```
def mindtct_from_image(image):
    imgpath = os.path.abspath(image.path.filepath)
    tempdir = tempfile.mkdtemp()
    oroot = os.path.join(tempdir, 'result')

    cmd = ['mindtct', imgpath, oroot]

    try:
        subprocess.check_call(cmd)

        with open(oroot + '.xyt') as fd:
```

```
            xyt = fd.read()

        result = mindtct(image=image.id, xyt=xyt)
        return result

    finally:
        shutil.rmtree(tempdir)
```

15.15.7.3 Bozorth3

The final step in the pipeline is running the `bozorth3` from NBIS. The `bozorth3` class represents the match being done: tracking the ids of the probe and gallery images as well as the match score.

Since we'll be writing these instance out to a database, we provide some static methods for SQL statements. While there are many Object-Relational-Model (ORM) libraries available for Python, this approach keeps the current implementation simple.

```
@attr.s(slots=True)
class bozorth3(object):
    probe = attr.ib()
    gallery = attr.ib()
    score = attr.ib()

    @staticmethod
    def sql_stmt_create_table():
        return 'CREATE TABLE IF NOT EXISTS bozorth3' \
             + '(probe TEXT, gallery TEXT, score NUMERIC)'

    @staticmethod
    def sql_prepared_stmt_insert():
        return 'INSERT INTO bozorth3 VALUES (?, ?, ?)'

    def sql_prepared_stmt_insert_values(self):
        return self.probe, self.gallery, self.score
```

In order to work well with `multiprocessing`, we define a class representuing the input paramaters to `bozorth3` and a helper function to run `bozorth3`. This way the pipeline definition can be kept simple to a `map` to create the input and then a `map` to run the program.

As NBIS `bozorth3` can be called to compare one-to-one or one-to-many, we'll also dynamically choose between these approaches depending on if the gallery attribute is a list or a single object.

```
@attr.s(slots=True)
class bozorth3_input(object):
    probe = attr.ib()
    gallery = attr.ib()

    def run(self):
        if isinstance(self.gallery, mindtct):
            return bozorth3_from_one_to_one(self.probe, self.gallery)
        elif isinstance(self.gallery, types.ListType):
            return bozorth3_from_one_to_many(self.probe, self.gallery)
        else:
            raise ValueError('Unhandled type for gallery: {}'.format(type(gallery)))
```

The next is the top-level function to running `bozorth3`. It accepts an instance of `bozorth3_input`. The is implemented as a simple top-level wrapper so that it can be easily passed to the `multiprocessing` library.

```
def run_bozorth3(input):
    return input.run()
```

15.15.7.3.1 Running Bozorth3

There are two cases to handle: 1. One-to-one probe to gallery sets 1. One-to-many probe to gallery sets

Both approaches are implemented below. The implementations follow the same pattern: 1. Create a temporary directory within with to work 1. Write the probe and gallery images to files in the temporary directory 1. Call the `bozorth3` executable 1. The match score is written to `stdout` which is captured and then parsed. 1. Return a `bozorth3` instance for each match 1. Make sure to clean up the temporary directory

15.15.7.3.1.1 One-to-one

```
def bozorth3_from_one_to_one(probe, gallery):
    tempdir = tempfile.mkdtemp()
    probeFile = os.path.join(tempdir, 'probe.xyt')
    galleryFile = os.path.join(tempdir, 'gallery.xyt')

    with open(probeFile,   'wb') as fd: fd.write(probe.xyt)
    with open(galleryFile, 'wb') as fd: fd.write(gallery.xyt)

    cmd = ['bozorth3', probeFile, galleryFile]

    try:
        result = subprocess.check_output(cmd)
        score = int(result.strip())
        return bozorth3(probe=probe.image, gallery=gallery.image, score=score)
    finally:
        shutil.rmtree(tempdir)
```

15.15.7.3.1.2 One-to-many

```
def bozorth3_from_one_to_many(probe, galleryset):
    tempdir = tempfile.mkdtemp()
    probeFile = os.path.join(tempdir, 'probe.xyt')
    galleryFiles = [os.path.join(tempdir, 'gallery%d.xyt' % i)
                    for i,_ in enumerate(galleryset)]

    with open(probeFile, 'wb') as fd: fd.write(probe.xyt)
    for galleryFile, gallery in itertools.izip(galleryFiles, galleryset):
        with open(galleryFile, 'wb') as fd: fd.write(gallery.xyt)

    cmd = ['bozorth3', '-p', probeFile] + galleryFiles

    try:
        result = subprocess.check_output(cmd).strip()
        scores = map(int, result.split('\n'))
        return [bozorth3(probe=probe.image, gallery=gallery.image, score=score)
                for score, gallery in zip(scores, galleryset)]
    finally:
        shutil.rmtree(tempdir)
```

15.16 Plotting

For plotting we'll operate only on the database. We'll select a small number of probe images and plot the score between them and the rest of the gallery images.

The `mk_short_labels` helper function will be defined below.

```
def plot(dbfile, nprobes=10):
    conn = sqlite3.connect(dbfile)
    results = pd.read_sql(
        "SELECT DISTINCT probe FROM bozorth3 ORDER BY score LIMIT '%s'" % nprobes,
        con=conn
    )
    shortlabels = mk_short_labels(results.probe)
    plt.figure()

    for i, probe in results.probe.iteritems():
```

```
        stmt = 'SELECT gallery, score FROM bozorth3 WHERE probe = ? ORDER BY gallery DESC'
        matches = pd.read_sql(stmt, params=(probe,), con=conn)
        xs = np.arange(len(matches), dtype=np.int)
        plt.plot(xs, matches.score, label='probe %s' % shortlabels[i])

    plt.ylabel('Score')
    plt.xlabel('Gallery')
    plt.legend(bbox_to_anchor=(0, 0, 1, -0.2))
    plt.show()
```

The image ids are long hash strings. In ordere to minimize the amount of space on the figure the labels occupy, we provide a helper function to create a short label that still uniquely identifies each probe image in the selected sample

```
def mk_short_labels(series, start=7):
    for size in xrange(start, len(series[0])):
        if len(series) == len(set(map(lambda s: s[:size], series))):
            break
    return map(lambda s: s[:size], series)
```

15.17 Putting it all Together

First, set up a temporary directory in which to work:

```
pool = multiprocessing.Pool()
prefix = '/tmp/fingerprint_example/'
if not os.path.exists(prefix):
    os.makedirs(prefix)
```

Next we download and extract the fingerprint images from NIST:

```
%%time
dataprefix = prepare_dataset(prefix=prefix)
```

```
Verifying checksum Extracting
/tmp/fingerprint_example/NISTSpecialDatabase4GrayScaleImagesofFIGS.zip
to /tmp/fingerprint_example/ CPU times: user 3.34 s, sys: 645 ms,
total: 3.99 s Wall time: 4.01 s
```

Next we'll configure the location of of the MD5 checksum file that comes with the download

```
md5listpath = os.path.join(prefix, 'NISTSpecialDatabase4GrayScaleImagesofFIGS/sd04/sd04_md5.lst')
```

Load the images from the downloaded files to start the analysis pipeline

```
%%time
print('Loading images')
paths = locate_paths(md5listpath, dataprefix)
images = locate_images(paths)
mindtcts = pool.map(mindtct_from_image, images)
print('Done')
```

```
Loading images Done CPU times: user 187 ms, sys: 17 ms, total: 204 ms
Wall time: 1min 21s
```

We can examine one of the loaded image. Note that `image` is refers to the MD5 checksum that came with the image and the `xyt` attribute represents the raw image data.

```
print(mindtcts[0].image)
print(mindtcts[0].xyt[:50])
```

```
98b15d56330cb17f1982ae79348f711d 14 146 214 6 25 238 22 37 25 51 180 20
30 332 214
```

For example purposes we'll only a use a small percentage of the database, randomly selected, for pur probe and gallery datasets.

```
perc_probe = 0.001
perc_gallery = 0.1
```

```
%%time
print('Generating samples')
probes  = random.sample(mindtcts, int(perc_probe   * len(mindtcts)))
gallery = random.sample(mindtcts, int(perc_gallery * len(mindtcts)))
print('|Probes| =', len(probes))
print('|Gallery|=', len(gallery))
```

```
Generating samples = 4 = 400 CPU times: user 2 ms, sys: 0 ns, total: 2
ms Wall time: 993 µs
```

We can now compute the matching scores between the probe and gallery sets. This will use all cores available on this workstation.

```
%%time
print('Matching')
input = [bozorth3_input(probe=probe, gallery=gallery)
         for probe in probes]
bozorth3s = pool.map(run_bozorth3, input)
```

```
Matching CPU times: user 19 ms, sys: 1 ms, total: 20 ms Wall time: 1.07
s
```

`bozorth3s` is now a `list` of `lists` of `bozorth3` instances.

```
print('|Probes|  =', len(bozorth3s))
print('|Gallery| =', len(bozorth3s[0]))
print('Result:', bozorth3s[0][0])
```

```
= 4 = 400 Result: bozorth3(probe='caf9143b268701416fbed6a9eb2eb4cf',
gallery='22fa0f24998eaea39dea152e4a73f267', score=4)
```

Now add the results to the database

```
dbfile = os.path.join(prefix, 'scores.db')
conn = sqlite3.connect(dbfile)
cursor = conn.cursor()
cursor.execute(bozorth3.sql_stmt_create_table())
```

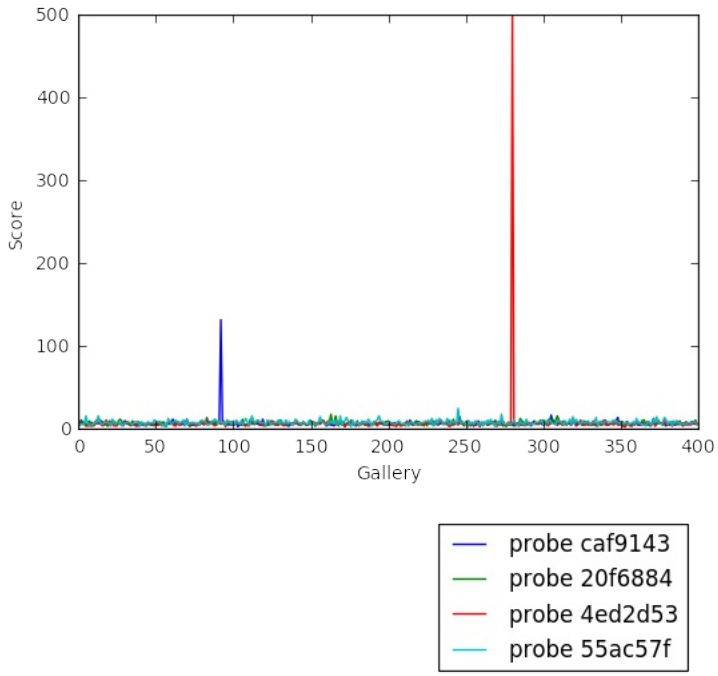
```
<sqlite3.Cursor at 0x7f8a2f677490>
```

```
%%time
for group in bozorth3s:
    vals = map(bozorth3.sql_prepared_stmt_insert_values, group)
    cursor.executemany(bozorth3.sql_prepared_stmt_insert(), vals)
    conn.commit()
    print('Inserted results for probe', group[0].probe)
```

```
Inserted results for probe caf9143b268701416fbed6a9eb2eb4cf Inserted
results for probe 55ac57f711eba081b9302eab74dea88e Inserted results for
probe 4ed2d53db3b5ab7d6b216ea0314beb4f Inserted results for probe
20f68849ee2dad02b8fb33ecd3ece507 CPU times: user 2 ms, sys: 3 ms, total:
5 ms Wall time: 3.57 ms
```

We now plot the results.

```
plot(dbfile, nprobes=len(probes))
```



image

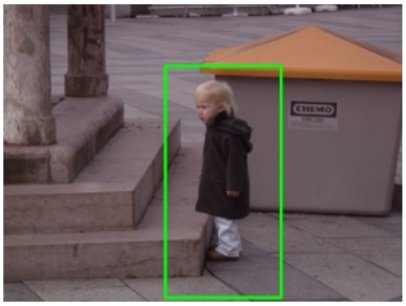
```
cursor.close()
```

15.18 NIST Pedestrian and Face Detection

Pedestrian and Face Detection uses OpenCV to identify people standing in a picture or a video and NIST use case in this document is built with Apache Spark and Mesos clusters on multiple compute nodes.

The example in this tutorial deploys software packages on OpenStack using Ansible with its roles.

Original



Pedestrian Detected

Original

Pedestrian and Face/eyes Detected

15.18.0.1 Introduction

Human (pedestrian) detection and face detection have been studied during the last several years and models for them have improved along with Histograms of Oriented Gradients (HOG) for Human Detection [1]. OpenCV is a Computer Vision library including the SVM classifier and the HOG object detector for pedestrian detection and INRIA Person Dataset [2] is one of popular samples for both training and testing purposes. In this document, we deploy Apache Spark on Mesos clusters to train and apply detection models from OpenCV using Python API.

15.18.0.1.1 INRIA Person Dataset

This dataset contains positive and negative images for training and test purposes with annotation files for upright persons in each image. 288 positive test images, 453 negative test images, 614 positive training images and 1218 negative training images are included along with normalized 64x128 pixel formats. 970MB dataset is available to download [3].

15.18.0.1.2 HOG with SVM model

Histogram of Oriented Gradient (HOG) and Support Vector Machine (SVM) are used as object detectors and classifiers and built-in python libraries from OpenCV provide these models for human detection.

15.18.0.1.3 Ansible Automation Tool

Ansible is a python tool to install/configure/manage software on multiple machines with JSON files where system descriptions are defined. There are reasons why we use Ansible:

- Expandable: Leverages Python (default) but modules can be written in any language
- Agentless: no setup required on managed node
- Security: Allows deployment from user space; uses ssh for authentication
- Flexibility: only requires ssh access to privileged user
- Transparency: YAML Based script files express the steps of installing and configuring software
- Modularity: Single Ansible Role (should) contain all required commands and variables to deploy software package independently
- Sharing and portability: roles are available from source (github, bitbucket, gitlab, etc) or the Ansible Galaxy portal

We use Ansible roles to install software packages for Humand and Face Detection which requires to run OpenCV Python libraries on Apache Mesos with a cluster configuration. Dataset is also downloaded from the web using an ansible role.

15.18.0.2 Deployment by Ansible

Ansible is to deploy applications and build clusters for batch-processing large datasets towards target machines e.g. VM instances on OpenStack and we use ansible roles with include directive to organize layers of big data software stacks (BDSS). Ansible provides abstractions by Playbook Roles and reusability by Include statements. We define X application in X Ansible Role, for example, and use include statements to combine with other applications e.g. Y or Z. The layers exist in sub directories (see below) to add modularity to your Ansible deployment. For example, there are five roles used in this example that are Apache Mesos in a scheduler layer, Apache Spark in a processing layer, a OpenCV library in an application layer, INRIA Person Dataset in a dataset layer and a python script for human and face detection in an analytics layer. If you have an additional software package to add, you can simply add a new role in a main ansible playbook with include directive. With this, your Ansible playbook maintains simple but flexible to add more roles without having a large single file which is getting difficult to read when it deploys more applications on multiple layers. The main Ansible playbook runs Ansible roles in order which look like:

```
```
include: sched/00-mesos.yml
include: proc/01-spark.yml
include: apps/02-opencv.yml
include: data/03-inria-dataset.yml
Include: anlys/04-human-face-detection.yml
```
```

Directory names e.g. sched, proc, data, or anlys indicate BDSS layers like: - sched: scheduler layer - proc: data processing layer - apps: application layer - data: dataset layer - anlys: analytics layer and two digits in the filename indicate an order of roles to be run.

15.18.0.3 Cloudmesh for Provisioning

It is assumed that virtual machines are created by cloudmesh, the cloud management software. For example on OpenStack,

```
cm cluster create -N=6
```

command starts a set of virtual machine instances. The number of machines and groups for clusters e.g. namenodes and datanodes are defined in the Ansible inventory file, a list of target machines with groups, which will be generated once machines are ready to use by cloudmesh. Ansible roles install software and dataset on virtual clusters after that stage.

15.18.0.4 Roles Explained for Installation

Mesos role is installed first as a scheduler layer for masters and slaves where mesos-master runs on the masters group and mesos-slave runs on the slaves group. Apache Zookeeper is included in the mesos role therefore mesos slaves find an elected mesos leader for the coordination. Spark, as a data processing layer, provides two options for distributed job processing, batch job processing via a cluster mode and real-time processing via a client mode. The Mesos dispatcher runs on a masters group to accept a batch job submission and Spark interactive shell, which is the client mode, provides real-time processing on any node in the cluster. Either way, Spark is installed later to detect a master (leader) host for a job submission. Other roles for OpenCV, INRIA Person Dataset and Human and Face Detection Python applications are followed by.

The following software are expected in the stacks according to the github:

- mesos cluster (master, worker)
- spark (with dispatcher for mesos cluster mode)
- openCV
- zookeeper
- INRIA Person Dataset
- Detection Analytics in Python
- [1] Dalal, Navneet, and Bill Triggs. "Histograms of oriented gradients for human detection." 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Vol. 1. IEEE,

  2005. [pdf]
- [2] http://pascal.inrialpes.fr/data/human/
- [3] ftp://ftp.inrialpes.fr/pub/lear/douze/data/INRIAPerson.tar
- [4] https://docs.python.org/2/library/configparser.html

15.18.0.4.1 Server groups for Masters/Slaves by Ansible inventory

We may separate compute nodes in two groups: masters and workers therefore Mesos masters and zookeeper quorums manage job requests and leaders and workers run actual tasks. Ansible needs group definitions in their inventory therefore software installation associated with a proper part can be completed.

Example of Ansible Inventory file (inventory.txt)

```
[masters]
10.0.5.67
10.0.5.68
10.0.5.69
[slaves]
10.0.5.70
10.0.5.71
10.0.5.72
```

15.18.0.5 Instructions for Deployment

The following commands complete NIST Pedestrian and Face Detection deployment on OpenStack.

15.18.0.5.1 Cloning Pedestrian Detection Repository from Github

Roles are included as submodules which require `--recursive` option to checkout them all.

```
$ git clone --recursive https://github.com/futuresystems/pedestrian-and-face-detection.git
```

Change the following variable with actual ip addresses:

```
sample_inventory="""[masters]
10.0.5.67
10.0.5.68
10.0.5.69
[slaves]
10.0.5.70
10.0.5.71
10.0.5.72"""
```

Create a `inventory.txt` file with the variable in your local directory.

```
!printf "$sample_inventory" > inventory.txt
!cat inventory.txt
```

Add `ansible.cfg` file with options for ssh host key checking and login name.

```
ansible_config="""[defaults]
host_key_checking=false
remote_user=ubuntu"""
!printf "$ansible_config" > ansible.cfg
!cat ansible.cfg
```

Check accessibility by ansible ping like:

```
!ansible -m ping -i inventory.txt all
```

Make sure that you have a correct ssh key in your account otherwise you may encounter 'FAILURE' in the ping test above.

15.18.0.5.2 Ansible Playbook

We use a main ansible playbook to deploy software packages for NIST Pedestrian and Face detection which includes: - mesos - spark -zookeeper - opencv - INRIA Person dataset - Python script for the detection

```
!cd pedestrian-and-face-detection/ && ansible-playbook -i ../inventory.txt site.yml
```

The installation may take 30 minutes or an hour to complete.

15.18.0.6 OpenCV in Python

Before we run our code for this project, let's try OpenCV first to see how it works.

15.18.0.6.1 Import cv2

Let's import opencv python module and we will use images from the online database image-net.org to test OpenCV image recognition.

```
import cv2
```

Let's download a mailbox image with a red color to see if opencv identifies the shape with a color. The example file in this tutorial is:

```
$ curl http://farm4.static.flickr.com/3061/2739199963_ee78af76ef.jpg > mailbox.jpg
```

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 167k 100 167k    0     0   686k      0 --:--:-- --:--:-- --:--:--  684k
```

```
%matplotlib inline

from IPython.display import Image
mailbox_image = "mailbox.jpg"
Image(filename=mailbox_image)
```

image

You can try other images. Check out the image-net.org for mailbox images: http://image-net.org/synset?wnid=n03710193

15.18.0.6.2 Image Detection

Just for a test, let's try to detect a red color shaped mailbox using opencv python functions.

There are key functions that we use: * cvtColor: to convert a color space of an image * inRange: to detect a mailbox based on the range of red color pixel values * np.array: to define the range of red color using a Numpy library for better calculation * findContours: to find a outline of the object * bitwise_and: to black-out the area of contours found

```
import numpy as np
import matplotlib.pyplot as plt

# imread for loading an image
img = cv2.imread(mailbox_image)
# cvtColor for color conversion
hsv = cv2.cvtColor(img,cv2.COLOR_BGR2HSV)

# define range of red color in hsv
lower_red1 = np.array([0, 50, 50])
upper_red1 = np.array([10, 255, 255])
lower_red2 = np.array([170, 50, 50])
upper_red2 = np.array([180, 255, 255])

# threshold the hsv image to get only red colors
mask1 = cv2.inRange(hsv, lower_red1, upper_red1)
mask2 = cv2.inRange(hsv, lower_red2, upper_red2)
mask = mask1 + mask2

# find a red color mailbox from the image
im2, contours,hierarchy = cv2.findContours(mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

# bitwise_and to remove other areas in the image except the detected object
res = cv2.bitwise_and(img, img, mask = mask)

# turn off - x, y axis bar
plt.axis("off")
# text for the masked image
cv2.putText(res, "masked image", (20,300), cv2.FONT_HERSHEY_SIMPLEX, 2, (255,255,255))
# display
plt.imshow(cv2.cvtColor(res, cv2.COLOR_BGR2RGB))
plt.show()
```

image

The red color mailbox is left alone in the image which we wanted to find in this example by opencv functions. You can try other images with different colors to detect the different shape of objects using findContours and inRange from opencv.

For more information, see the useful links below.

- contours features: http://docs.opencv.org/3.1.0/dd/d49/tutorial/_py/_contour/_features.html
- contours: http://docs.opencv.org/3.1.0/d4/d73/tutorial/_py/_contours/_begin.html
- red color in hsv: http://stackoverflow.com/questions/30331944/finding-red-color-using-python-opencv
- inrange: http://docs.opencv.org/master/da/d97/tutorial/_threshold/_inRange.html

- inrange: http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_colorspaces/py_colorspaces.html
- numpy: http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_core/py_basic_ops/py_basic_ops.html

15.18.0.7 Human and Face Detection in OpenCV

15.18.0.7.1 INRIA Person Dataset

We use INRIA Person dataset to detect upright people and faces in images in this example. Let's download it first.

```
$ curl ftp://ftp.inrialpes.fr/pub/lear/douze/data/INRIAPerson.tar > INRIAPerson.tar
```

```
  100  969M  100  969M    0     0  8480k      0  0:01:57  0:01:57 --:--:-- 12.4M
```

```
$ tar xvf INRIAPerson.tar > logfile && tail logfile
```

15.18.0.7.2 Face Detection using Haar Cascades

This section is prepared based on the opencv-python tutorial: http://docs.opencv.org/3.1.0/d7/d8b/tutorial_py_face_detection.html#gsc.tab=0

There is a pre-trained classifier for face detection, download it from here:

```
$ curl https://raw.githubusercontent.com/opencv/opencv/master/data/haarcascades/haarcascade_frontalface_default.xml > haarcascade_frontalface_default.xml
```

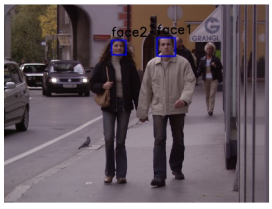
```
  100  908k  100  908k    0     0  2225k      0 --:--:-- --:--:-- --:--:-- 2259k
```

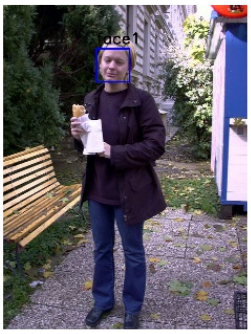
This classifier XML file will be used to detect faces in images. If you like to create a new classifier, find out more information about training from here: http://docs.opencv.org/3.1.0/dc/d88/tutorial_traincascade.html

15.18.0.7.3 Face Detection Python Code Snippet

Now, we detect faces from the first five images using the classifier.

```
# import the necessary packages
from __future__ import print_function
import numpy as np
import cv2
from os import listdir
from os.path import isfile, join
import matplotlib.pyplot as plt

mypath = "INRIAPerson/Test/pos/"
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

onlyfiles = [join(mypath, f) for f in listdir(mypath) if isfile(join(mypath, f))]

cnt = 0
for filename in onlyfiles:
    image = cv2.imread(filename)
    image_grayscale = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(image_grayscale, 1.3, 5)
    if len(faces) == 0:
        continue

    cnt_faces = 1
    for (x,y,w,h) in faces:
        cv2.rectangle(image,(x,y),(x+w,y+h),(255,0,0),2)
        cv2.putText(image, "face" + str(cnt_faces), (x,y-10), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,0), 2)
        plt.figure()
        plt.axis("off")
        plt.imshow(cv2.cvtColor(image[y:y+h, x:x+w], cv2.COLOR_BGR2RGB))
        cnt_faces += 1
    plt.figure()
    plt.axis("off")
    plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
    cnt = cnt + 1
    if cnt == 5:
        break
```



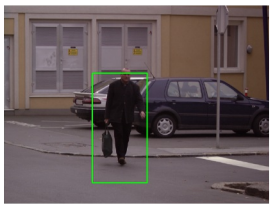


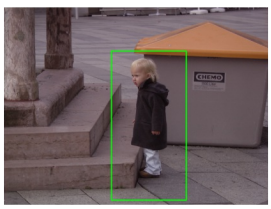
15.18.0.8 Pedestrian Detection using HOG Descriptor

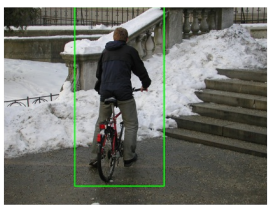
We will use Histogram of Oriented Gradients (HOG) to detect a upright person from images.

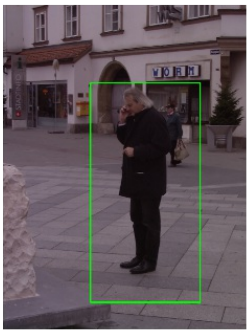
15.18.0.8.1 Python Code Snippet

```
# initialize the HOG descriptor/person detector
hog = cv2.HOGDescriptor()
hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())

cnt = 0
for filename in onlyfiles:
    img = cv2.imread(filename)
    orig = img.copy()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # detect people in the image
    (rects, weights) = hog.detectMultiScale(img, winStride=(8, 8),
    padding=(16, 16), scale=1.05)

    # draw the final bounding boxes
    for (x, y, w, h) in rects:
        cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)

    plt.figure()
    plt.axis("off")
    plt.imshow(cv2.cvtColor(orig, cv2.COLOR_BGR2RGB))
    plt.figure()
    plt.axis("off")
    plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    cnt = cnt + 1
    if cnt == 5:
        break
```









15.18.0.9 Processing by Apache Spark

INRIA Person dataset provides 100+ images and Spark can be used for image processing in parallel. We load 288 images from "Test/pos" directory.

Spark provides a special object 'sc' to connect between a spark cluster and functions in python code. Therefore, we can run python functions in parallel to detet objects in this example.

- map function is used to process pedestrian and face detection per image from the parallelize() function of 'sc' spark context.
- collect fonction merges results in an array.

def apply_batch(imagePath): import cv2 import numpy as np # initialize the HOG descriptor/person detector hog = cv2.HOGDescriptor() hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector()) image = cv2.imread(imagePath) # detect people in the image (rects, weights) = hog.detectMultiScale(image, winStride=(8, 8), padding=(16, 16), scale=1.05) # draw the final bounding boxes for (x, y, w, h) in rects: cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2) return image

15.18.0.9.1 Parallelize in Spark Context

The list of image files is given to parallelize.

```
pd = sc.parallelize(onlyfiles)
```

15.18.0.9.2 Map Function (apply_batch)

The 'apply_batch' function that we created above is given to map function to process in a spark cluster.

```
pdc = pd.map(apply_batch)
```

15.18.0.9.3 Collect Function

The result of each map process is merged to an array.

```
result = pdc.collect()
```

15.18.0.10 Results for 100+ images by Spark Cluster

```
for image in result:
    plt.figure()
    plt.axis("off")
    plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
```

16 FAQ

16.1 FAQ: General

In this section TA's and students can add FAQ's from the piazza. As the material especially the programming related one is so useful that it is shared by now in multiple classes, however they use different piazza's sharing the information in an FAQ in the handbook allows us to quickly disseminate the relevant information between classes. If an FAQ is only for one class we will be especially mark it.

16.1.1 Can I assume that all information is in the FAQ to do the class?

No. The class book will be our main source of information not just a collection of FAQ's.

16.1.2 Piazza

16.1.2.1 Why are some FAQs that are on piazza not here?

Two reason:

1. some of them need not to be in this FAQ.
2. The TAs will evaluate the FAQs every day at the end of the day and integrate those that need to be in this list at that time. Hence it may take up to 24 hours for FAQs to appear here.

Once an FAQ is in the book answered (it may actually be part of another section, TA's will mark the FAQ in piazza, so you can make sure which FAQs are already in the book. We recommend to look in the book as there could be information in it that you otherwise missed.

16.1.3 How do I find all FAQ's in Piazza?

Two ways exist

a. Please visit your class piazza. You will find a "faq" tag in your piazza window. Click on it and all posts marked with FAQ will show up,

b. In the search field type in FAQ. All posts with the text FAQ in it will be listed.

16.1.4 Has SOIC computers I can use remotely?

See: https://uisapp2.iu.edu/confluence-prd/pages/viewpage.action?pageId=114491559

16.1.5 When contributing to the book my name is not listed properly or not at all

The following reasons exist:

1. if its not listed at all your contribution may be in a different repository, please contact Gregor
2. if it does not show up correctly and only shows your github name, which you can see in the contributor section or with

```
$ git shortlog -s -e
2  laszewsk <laszewski@gmail.com>
...
```

You need to do two things.

First, add your name to the file

- https://github.com/cloudmesh-community/book/blob/master/.mailmap

Second, complete the set up your git on the machine you work with in case you use a commandline tool with git init (see our notes on this)

If you use the GUI you may need to go to the account settings and associate a first name lastname, I however do not know ho to do that, so if you kwon reply ti this

16.1.6 How to read the technical sections of the lecture notes

We will add throughout the semester some technical lecture notes. These notes contain information on how to install and run certain programs on a computer. What we have seen in the past with some students is that they do not read the text between the sections. Instead they just execute things without reading or understanding assuming that they can juste copy and paste. These sections include valuable information that you **must** read before you execute any code in them.

Here is the workflow on how to read such technical sections

1. Do not execute anything yet
2. Read the entire section including the lines between the gray boxes
3. Step back and reflect on what you read
4. Reread the section, if a section needs more information google for it (things could be overnight updated on the internet, please remember we are just presenting a snapshot in time here)
5. Once you have obtained knowledge, decide if the section is relevant for you (e.g. windows sections may not be relevant for MacOS users)
6. Carefully execute the relevant portions for you

⚠ AS ALWAYS THERE IS NO GUARANTEE THAT WHAT THE CODE WORKS OR COULD NOT DESTROY SOMETHING. MAKE SURE TO HAVE A BACKUP. IF IN DOUBT RUN IN A VIRTUAL MACHINE IF YOU CAN.

16.1.7 How to check if a yaml file is valid?

In case you need to check an open source public YAML file you can use the following

The easiest is to use yamllint:

```
$ pip install yamllint
$ yamllint README.yml
```

Using yamllint is our preferred method.

A python script to check it is available at

- https://github.com/cloudmesh-community/book/tree/master/examples/yaml-validation/validate_yml.py

This python script depends on `ruamel.yaml` package. We can install it using following command:

```
$ pip install ruamel.yaml
```

It accepts file path as an argument. This script will load `YAML` file and dump its content on console. For invalid syntax it will throw an error.

To execute python script you need to run following command after you clone book repository.

```
$ cd examples/yaml-validation
$ chmod +x validate_yml.py
$ ./validate_yml.py <path to yaml file to validate>
```

Online checkers are available at

- https://codebeautify.org/yaml-validator

A ruby script can do

```
$ ruby-e "require 'yaml';puts YAML.load_file('./README.yml')"
```

YAML validation in visual studio can be achieved also

- https://marketplace.visualstudio.com/items?itemName=redhat.vscode-yaml

16.1.8 Download the epub ferquently

Please be reminded that the epub is updated frequently and we recommend that you download it before you read.

I myself have integrated an epub reader in my Web browser so that every time I click on the View Raw in github, I get the most up to date version.

I use ibooks on OSX, calibre is a good system on Windows and Linux, MS also has Microsoft Edge. However on Microsoft edge you will need the lates version which starts with 42

16.1.9 Spelling of filenames in github

Most of our scripts require proper spellingg including proper capitalization. The spelling of `notebook.md` is `notebook.md`

not

`Notebook.md` or `NOTEBOOK.md` or other spelling

Please, correct if you did not use lower case

The spelling of `README.yml` is`README.yml` and not

`README.md` (which needs to be removed) or `readme.yaml`

please correct if needed. We will not grade any assignments if your README.yaml or notebook.md is misspelled or missing or is not following our simple format.

16.1.10 How to open the epub from Github?

If you see thw `View Raw`, you need to click on it. It will download the file. Than you can open that.

However, If you use edge or integrated your epub viewer in your browser and clicking on it will automatically open your epub browser.

16.1.11 Assignment Summary

⭕ outdated

a. The assignment is discussed in Chapter 1 of the lecture notes

b. Examples of what other students have done are in the Example Artifacts section

Please look at both sections

In this class we addressed 3 assignment that related to your grade

Tech summaries - they have been assigned to you in https://piazza.com/class/jl6rxey6w413gi?cid=89 to show to the TAs that you work on them use the nomenclature that is discussed in the preface of the technology handbook. Put yours hid in the "headline" and a smiley when done, If you work on it put in a hand. Project - look at examples in the example artifact sections A paper has typically the following sections

Theory Implementation (e.g. Python) Benchmark

A more detailed outline is * Paper * Title * Abstract * Introductions * Requirements * Architecture * Implementation * Benchmark * Conclusions * Bibliography * Work breakdown

16.1.12 Auto 80 char

⭕ outdated

Those that use emcas could experiment with the following. I do not know if this works well yet.

The following will autoformat an entire file to 80 chars. The reason i put it in test.md is that I do not know if it reliably works on all md files, just inspect the output and decide for yourself. some md files you may not want to manipulate with this though

```
cp file.md test.md
emacs -batch test.md --eval '(fill-region (point-min) (point-max))' -f save-buffer
```

16.1.13 Useful FAQs for residential and online students

⭕ this is outdated.

You will know if this post applies to you.

This class does not have a high volume on posts via Piazza

What we find is that some students create a high overhead on themselves by not following our FAQs or documentation on the technology summaries. When we observe something we just post it in an FAQ in piazza that we expect you look at. Yet we find some students that keep on resubmitting their technology summaries while not integrating our tips from the FAQs that cost less then a minute to do. Those that do not read and follow the FAQ make their work unnecessary complicated. We even start now noticing students that remove bibliography entries instead of just fixing them. Also, we saw recently students that had perfectly geat entries, other than the authors (see FAQ) and instead of fixing the authors in case it was a company or organization, to fix random fields such as the titles and thus creating even more work on themselves.

We have lots of online hours during the week There are 4 hours you can attend, Mo, We Thu, so if you do have something you do not understand, I recommend that you use these hours. In case you are a residential student you also have Fridays. To start, I would review our FAQs

Interestingly we see these issues more with residential students than with online students. This may indicate that the residential students in question forget to read the posts in piazza?

16.1.14 What if i committed a wrong file to github, a.g. a private key?

The answer to this question is more complicated than you think. Thus the best way to deal with it is to

AVOID IT:

a. first do github adds file by fill with git add. Avoid using adds on AND DO NOT USE

```
git add .                 # <<<<<<< DO NOT USE
```

b. only use ssh keys in ~/.ssh **NEVER** place keys in directories that are managed by git

YOU CAN NOT EASILY DELETE FILES FROM GIT:

c. as you may already know despite you deleting a file from git it is still in the git history. Also there are bad characters out there so if you checked in you ssh private key just for a second

you must assume your private key is now compromised and all machines that use it are compromised.

d. although Glt allows you to delete the file, it is still in the githistory, which can be mined so despite you pressing delete its still there and can be found. This is not a bug in git but this is you having git not used right.

e. There are ways to purge such files, but it would imply that everyone that did a fork needs to do a new fork which is naturally a big issue, so we do not do this during the semester.

NOW WHAT?

f. every machine on which you used the public key of this private key is to be considered now compromised.

g. put them off from the network while plugging out the network plug

h. if the machines are not owned by you but for example, IU, notify the people that own the machine to ask for help with mitigation.

i. if you are lucky, replace the key, this is the case for example for services such as github. Make sure to inspect the configurations and see if your account has not been hijacked.

j. We will immediately remove you from services such as future systems and chameleon cloud as a precaution or deactivate your membership in our cloud accounts.

k. if you used the keys on other services, including IU, it is up to you to identify how to deal with this,

l. definitely create a new key and use that from now on.

m. you can call Gregors office number or use piazza to set up a call to identify what the impact is as this is typically an emergency use 812 856 1311. Do not leave a msg, but instead send e-mail.with your phone number so we can call you back to assess the situation.

n. if you use them on public clouds that cost money, shut down all machines that use them. I would not start them again but instead use new once. It may be time to drop everything and do this first. Sorry for making you now panic.

16.2 FAQ: 423/523 AND OTHERS COLOCATED WITH THEM

This section contains FAQs relevant for 523.

16.2.1 Bibtex tips for consitency across contributors

Congratulations, the majority of the bitex entries were done correctly. However, there are some few and small issues you could improve. This helps consistency across all contributors

a. use camel case in all titles consistently

b. see the FAQ on authors and keys

c. allowed are

howpublished={Web page},

howpublished={Blog},

howpublished={Presentation},

howpublished={Github},

howpublished={Bitbucket},

in miscs are allowed

not allowed are: webpage Webpage website Website,

any author or organization name

in case of wikipedia, the author ie author = {{Wikipedia}},

all misc labels ought to have a www- in it if they are online resources

c. we do not use **???** which is specific to some publishers and not universal

d. if a online citation has a publishing date we use month and year not date, you can not mix date with month and year, use month and year instead, If the publication date is not known use month and date for access and put note={Accessed}, in the entry

e. Use camel case for authors. Note that some authors have strange last name such as mine, so my author name is von Laszewski, Gregor

f. do not uses utf-8 chars such as “u”a and so on use instead {"u} {"a} and so on, see LaTeX bibtex manuals for details

Please remember this is not a lot to change for you.

A TA is assigned to help on this.

16.2.2 Misc entries require an author or key

Misc entries do require either an author or a key. The author lastname or the key is used for sorting. An entry without either is invalid.

You have time to fix this for a month. Most of you added an author.

In case the author is a company it must be in double brackets, example: author={{My Company}}, keys do naturally do not need double brackets.

16.2.3 TODO list location

I still haven’t received any feedback from my earlier question this morning about locating the todo list in the epub so that I may see what is wrong with my technology summaries. I’ve also checked that I do not have any outstanding pull requests. Could someone please help me find this so I may fix my summaries by tonight?

The Todo is in the epub of the technologies its a section header

16.2.4 Video on how to find the error reports for Technology Summaries

Those with pull request errors may want to look at

https://www.youtube.com/watch?v=FDqlKtQcy1U

16.2.5 only one url in url=

There can only be one url in the bibtex url filed, IF you need multiple, each one must have its own citation entry.

16.2.6 Incomplete analysis of your technologies

While reviewing some of your technologies we found that some students checked on their technologies with a smiley so we started looking at them. The good news is that many are good improvements.

However, we have a couple of suggestions so you can achieve your best.

a. we see that some students have missing bibtex entries or use labels wrong

b. it is in the student’s responsibility to fix all duplicated bibtex entries in all technologies. For example we only need one www-google bibtex and not Google-web page Google, and other labels, all labels in all technologies should be changed to a single entry

c. We see that when we assigned you a technology you do not cross check if other entries use your technology. Naturally, if we assign you a technology and the entry is duplicated you nee dto discuss with us what to do. In most cases you also have to fix the other entry for which you get also credit for.

d. students do not use linux tools such as grep because they have not yet switched to using command line tools for git.

As an example I like to provide what you what I would do if i were to improve the entry “Flume”

1. I would grep for it:

```
grep -n -R -i flume chapters bib
```

As a result I get

```
chapters/tech/flume.md:## Flume

chapters/tech/flume.md:| title    | Flume          |

chapters/tech/flume.md:Flume is distributed, reliable and available service for efficiently

chapters/tech/flume.md:data [@apache-flume]. Flume was created to allow you to flow data

chapters/tech/flume.md:from a source into your Hadoop environment.  In Flume, the entities

chapters/tech/flume.md:be any data source, and Flume has many predefined source adapters. A

chapters/tech/flume.md:or removing pieces of information, and more [@ibm-flume].

chapters/tech/google-flumejava.md:## Google FlumeJava :hand: fa18-523-83

chapters/tech/google-flumejava.md:| title    | Google FlumeJava       |

chapters/tech/google-flumejava.md:FlumeJava is a Java library that is built based on the concepts of MapReduce to simplify the development, testing, and execution of dataparallel pipelines [fa18-

chapters/tech/google-flumejava.md:FlumeJava is an easier-to-use version of MapReduce, make it simplier to build operation that process data. FlumeJava can also be integrated with other applicatio
```

```
chapters/tech/google-flumejava.md:FlumeJava was able to optimize MapReduce tasks and decrease execution time of MapReduce by allowing roll-back failed job instead of restarting [fa18-523-83-@flumejava-bp3].

bib/references.bib:@Misc{apache-flume,

bib/references.bib:  Title =        {Apache Flume},

bib/references.bib:  Url =          {https://flume.apache.org/index.html}

bib/references.bib:@Misc{ibm-flume,

bib/references.bib:  Title =        {What is Flume?},

bib/references.bib:                 {https://www-01.ibm.com/software/data/infosphere/hadoop/flume/}

bib/references.bib:@Inproceedings{flumejava-paper,

bib/references.bib:  Title =        {FlumeJava: Easy, Efficient Data-Parallel Pipelines},

bib/references.bib:@Misc{www-flumejava-google,

bib/references.bib:  Key =          {FlumeJava Google research},

bib/references.bib:@Misc{apache-flume,

bib/references.bib:  Title =        {Apache Flume},

bib/references.bib:  Url =          {https://flume.apache.org/index.html}

bib/references.bib:@Misc{ibm-flume,

bib/references.bib:  Title =        {What is Flume?},

bib/references.bib:                 {https://www-01.ibm.com/software/data/infosphere/hadoop/flume/}

bib/references.bib:@Inproceedings{flumejava-paper,

bib/references.bib:  Title =        {FlumeJava: Easy, Efficient Data-Parallel Pipelines},

bib/references.bib:@Misc{www-flumejava-google,

bib/references.bib:  Key =          {FlumeJava Google research},
```

Now I see the following

a. I get two entries that relate to fume. So I need to look at both of them and potentially fix both of them or if it makes sense merging them

b. I see a howl bunch of bib tex entries. In fact it looks like that many are duplicated. Thus I need to make sure that I reduce the number of bibtex entries, but I must be careful, as the once that I would be deleting could be used elsewhere and if I delete them or change the label I need to change them elsewhere also. This includes files that may use the bibtex that have nothing to do with my technology. It is easy, I just check with grep for all entries label that I remove or rename and change the corresponding time.

This all takes a minute in the command line. I am unaware that this task even can be done in the GUI and if it would take hours.

So if you spend hours on doing things in the GUI you do something wrong and must attend the online hour so TAs can teach you how to do it right. Naturally, we have all that information in the handbook also and many students do it right.

c. due to the nature of students may needing to change labels staying in sync with upstream is much easier on the commandline. as documented in the handbook.

So what would I do for Flume in summary

1. merge the entries while flume java becomes a subsection on the 3rd level (assuming the flume we talk about are the same)
2. remove all duplicated entries and use new labels I define
3. for an old entry i just leave the bibtex label and use that
4. in case I need to use new citations I prepend my hid
5. all www resources have www- in it. if not I rename the label

Based on this analysis the Flume entry does not pass our review

I have made significant changes that require a new fork or an update to the fork. Please do so. This could even be done from the GUI, however, the commandline is easier, so we do not teach you how to do it in the GUI.

16.2.7 The pull requests of technology summaries

Hi professor,

I have updated four summaries in the github and got the reply that “use in addition to” also >“, I have no idea about the meaning of that, could you please clarify it? My hid is fa18-523-85, the four technologies are“blaze“,”daal (intel)“,”lxc“,”OSGi".

Looking forward to hearing from you.

look at markdown documentation for >

look at epub and find a technology report from a student with a gray bar

16.2.8 REMINDER: quotes for technologies

In the technologies, we like you to make the quoting style consistent among all entries that are assigned to you. Please fix them in your entries. please see the following example:

> "This is a quote over multiple lines
>
> for the technologies" **???**.

Please be reminded to use straight quotes not left and right quotes and use the greater sign at the beginning. Please remember that you are only allowed to use 30% of quotes and that the technologies typically have a 300 word minimum.

16.2.9 Headings

a. please do not use all caps for heading

wrong:

```
## HEADING
```

correct

```
## Heading
```

The title is the only heading that has only one #

16.2.10 Quote characters in markdown

Some editors such as word try to be extra smart and do replace the quotes with a left and a right quote. However, markdown is designed to just use straight quotes.

use proper quotes which are " not left and right quote. Markdown however as we use it uses only one quote and that is “quote” if your editor puts left and right quotes in automatically, find a different editor such as emacs or pycharm

16.2.11 Tech Summaries. Puntuation, citations. Please read.

We see several check-ins that have good content but do not follow the rules for citations.

a. a technology section does not have a References section at the end. All technologies just use bibtex. The bibtex is automatically inserted where the label is, so you do not have to worry about managing a references section

b. a citation must be in the same sentence before the sentence ends.

```
This is wrong. [@label]

"This is also wrong." [@label]



This is the right way to cite [@label].
```

As citations are an important placement please check all your entries that you are responsible for as you only have 4 that should be an issue of minutes not hours.

IF we find such punctuation errors, your entry will be downgraded to a B if nothing else is wrong. We will also keep it marked with a red circle. The same rules apply to your 2 page paper and the project report.

Why so strict? Citation rules are strict and must be done correctly I had professional editors that would reject a submission with citation errors such as this.

16.2.12 use of underscore for em and bf

As we do some translations of the markdown, we noticed that whan you use _ instead of * in your markdown this may lead to issues. please use

```
*italic*

and

**bold**
```

17 Glossary

17.1 VM and Container

Dom0

TBD

Hypervisor

TBD

KVM

TBD

Virtual Machine

TBD

Virtual Machine Manager

TBD

XEN

TBD

cgroups

TBD

chroot

TBD

container

TBD

Kernel namespace

TBD

17.2 Network

Bridged Networking

TBD

External Network

TBD

Internal Network

TBD

Local Bridge

TBD

Network Address Translation (NAT)

TBD

17.3 Storage

Block Device

TBD

Virtual Disk

TBD

Raw Disk

TBD

Refernces

[1] G. von Laszewski, F. Wang, H. Lee, H. Chen, and G. C. Fox, "Accessing Multiple Clouds with Cloudmesh," in Proceedings of the 2014 acm international workshop on software-defined ecosystems, 2014, pp. 21–28 [Online]. Available: http://doi.acm.org/10.1145/2609441.2609638

[2] G. Fox, T. Hey, and A. Trefethen, "Where does all the data come from," Data-Intensive Science. Chapman and Hall/CRC, pp. 15–51, 2011.

[3] G. Aad et al., "Observation of a new particle in the search for the standard model higgs boson with the atlas detector at the lhc," Physics Letters B, vol. 716, no. 1, pp. 1–29, 2012 [Online]. Available: http://www.sciencedirect.com/science/article/pii/S037026931200857X

[4] W. McKinney, Python for data analysis: Data wrangling with pandas, numpy, and ipython. O'Reilly Media, Inc., 2012.

[5] jwork, "Welcome to datamelt." 2018 [Online]. Available: http://jwork.org/scavis/api/

[6] cms.cern, "Observation of higgs boson decay to bottom quakers." Web page, Aug-2018 [Online]. Available: https://cms.cern/

[7] X. Amatriain, "Building large-scale real-world recommender systems - recsys2012 tuto." Web page, Sep-2012 [Online]. Available: https://www.slideshare.net/xamat/building-largescale-realworld-recommender-systems-recsys2012-tutorial

[8] S. Seuken, "PowerPoint presentation." Web page, Oct-2012 [Online]. Available: https://www.ifi.uzh.ch/ce/teaching/spring2012/16-Recommender-Systems_Slides.pdf

[9] Kaggle Inc, "Kaggle: Your home for data science." Kaggle website, 2018 [Online]. Available: https://www.kaggle.com/

[10] M. Welling, "ICS175winter11." Web page, 2012 [Online]. Available: https://www.ics.uci.edu/~welling/teaching/CS77Bwinter12/CS77B_w12.html

[11] J. Hammerbacher, "Introduction to data science." Web page, Jan-2012 [Online]. Available: https://berkeleydatascience.files.wordpress.com/2012/01/20120117berkeley1.pdf

[12] S. Curtis, "Netflix foretells 'house of cards' success with cassandra big data engine | apps & wearables | techworld." Web page, Mar-2013 [Online]. Available: https://www.techworld.com/news/apps-wearables/netflix-foretells-house-of-cards-success-with-cassandra-big-data-engine-3437514/

[13] Wikipedia, "A/b testing." Web page, 2018 [Online]. Available: https://en.wikipedia.org/wiki/A/B_testing

[14] A. Cockcroft, "Architectural patterns for high availability." Web page, Apr-2013 [Online]. Available: https://www.infoq.com/presentations/Netflix-Architecture

[15] T. Infotech, "Big data for big sports." Web page, Aug-2014 [Online]. Available: https://www.slideshare.net/Tricon_Infotech/big-data-for-big-sports

[16] E. Lewallen, "Sport analytics innovation summit." Web page, Dec-2013 [Online]. Available: https://www.slideshare.net/elew/sport-analytics-innovation

[17] J. Beckham, "SmartBall keeps an eye inside the ball." Web page, Feb-2013 [Online]. Available: https://www.wired.com/2013/02/catapault-smartball/

[18] J. Varadarajan, "Automated playbook generation in football through videos." presentation, Jun-2014 [Online]. Available: http://www.sloansportsconference.com/wp-content/uploads/2014/06/Automated_Playbook_Generation.pdf

[19] A. D. S. Center, "Football trajectory dataset - interactive digital media: Semantic analysis of video." Web page, Oct-2018 [Online]. Available: http://autoscout.adsc.illinois.edu/publications/football-trajectory-dataset/

[20] K. Goldsberry, "CourtVision: New visual and spatial analytics for the nba." Presentation, Feb-2012 [Online]. Available: http://www.sloansportsconference.com/wp-content/uploads/2012/02/Goldsberry_Sloan_Submission.pdf

[21] GameSetMap, "GameSetMap acquired by golden set analytics." Web page, Nov-2017 [Online]. Available: http://gamesetmap.com/

[22] N. Santos, "Sports analytics innovation summit - data powered storytelling." Web page, Sep-2013 [Online]. Available: https://www.slideshare.net/BrandEmotivity/sports-analytics-innovation-summit-data-powered-storytelling

[23] ESPN, "MIT sloan sports analytics conference." Web page, Mar-2019 [Online]. Available: http://www.sloansportsconference.com/

[24] SABR, "Society for american baseball research." Web page, Oct-2018 [Online]. Available: https://sabr.org/

[25] Wikimedia Foundation, "Sabermetrics." Web page, 2018 [Online]. Available: https://en.wikipedia.org/wiki/Sabermetrics

[26] Wikimedia Foundation, "Baseball statistics." Web page, Oct-2018 [Online]. Available: https://en.wikipedia.org/wiki/Baseball_statistics

[27] M. Newman, "MLBAM introduces new way to analyze every play." Web page, Mar-2014 [Online]. Available: https://www.mlb.com/news/mlbam-introduces-new-way-to-analyze-every-play/c-68514514

[28] FANGRAPHS, "Complete list (offense)." Web page, Oct-2018 [Online]. Available: https://www.fangraphs.com/library/offense/offensive-statistics-list/

[29] Wikimedia Foundation, "Component era." Web page, Dec-2016 [Online]. Available: https://en.wikipedia.org/wiki/Component_ERA

[30] Wikimedia Foundation, "Wins above replacement." Web page, Sep-2016 [Online]. Available: https://en.wikipedia.org/wiki/Wins_Above_Replacement

[31] FANGRAPHS, "What is war." Web page, Oct-2018 [Online]. Available: https://www.fangraphs.com/library/misc/war/

[32] Sports Reference LLC, "Baseball-reference.com war explained." Web page, Oct-2018 [Online]. Available: https://www.baseball-reference.com/about/war_explained.shtml

[33] Sports Reference LLC, "War cpmparison chart." Web page, Oct-2018 [Online]. Available: https://www.baseball-reference.com/about/war_explained_comparison.shtml

[34] Sports Reference LLC, "Position player war calculations and details." Web page, Oct-2018 [Online]. Available: https://www.baseball-reference.com/about/war_explained_position.shtml

[35] Sports Reference LLC, "Pitcher war calculations and details." Web page, Oct-2018 [Online]. Available: https://www.baseball-reference.com/about/war_explained_pitch.shtml

[36] FANGRAPHS, "2018 fans' scouting report." Web page, Oct-2018 [Online]. Available: https://www.fangraphs.com/leaders.aspxpos=all&stats=bat&lg=all&qual=y&type=8&season=2014&month=0&season1=1871&ind=0

[37] Batting Leadoff, "Coffee as energizer of baseball players?" Web page, Aug-2018 [Online]. Available: http://battingleadoff.com/

[38] Wikimedia Foundation, "Coefficient of determination." Web page, Sep-2018 [Online]. Available: https://en.wikipedia.org/wiki/coefficient_of_determination

[39] G. Ganeshapillai and J. Guttag, "A data-driven method for in-game decision making in mlb," in Sport analytics conf, 2014.

[40] MLB Advanced Media, "Opening day, mike trout and 2014 season expectations." Web page, Mar-2014 [Online]. Available: http://vincegennaro.mlblogs.com/

[41] M. Fast, "Spinning yarn: How accurate is pitchtrax." Web page, Mar-2011 [Online]. Available: https://www.baseballprospectus.com/news/article/13109/spinning-yarn-how-accurate-is-pitchtrax/

[42] M. Fast, "What the heck is pitchf/x," illinois university of urbana-champaign; report, 2010 [Online]. Available: http://baseball.physics.illinois.edu/FastPFXGuide.pdf

[43] K. McSurley and G. Rybarczyk, "An introduction to fieldf/x," illinois university of urbana-champaign, 2011 [Online]. Available: http://baseball.physics.illinois.edu/FieldFX-TDR-GregR.pdf

[44] K. Wagner, "MLB announces revolutionary new fielding-tracking system." Web page, Mar-2014 [Online]. Available: https://deadspin.com/mlb-announces-revolutionary-new-fielding-tracking-syste-1534200504

[45] J. Keri, "Q&A: MLB advanced media's bob bowman discusses revolutionary new play-tracking system." Web page, Mar-2014 [Online]. Available: http://grantland.com/the-triangle/mlb-advanced-media-play-tracking-bob-bowman-interview/

[46] A. Andres, "The science of a home run: Andy andres at tedxyouth@BeaconStreet." Apr-2013 [Online]. Available: https://www.youtube.com/watch?v=YkjtnuNmK74

[47] Accenture, "Winning with the industrial internet of things." Web page, Oct-2018 [Online]. Available: https://www.accenture.com/us-en/insight-industrial-internet-of-things

[48] GE digital, "No unplanned downtime." Web page, Oct-2015 [Online]. Available: https://www.predix.com/ge-industrial-internet-infographic

[49] GE transportation, "Driving the digital transformation of transportation." Web page, 2016 [Online]. Available: http://www.getransportation.com/digital-solutions

[50] J. (. [. Freymann, "CIP survey of biomedical imaging archives." Web page, Oct-2016 [Online]. Available: https://wiki.nci.nih.gov/display/CIP/CIP+Survey+of+Biomedical+Imaging+Archives

[51] E. Ramirez, "Larry smarr archives - quantified self." Web page, Feb-2013 [Online]. Available: http://quantifiedself.com/larry-smarr/

[52] E. B. Institute, "About us." Web page, 2018 [Online]. Available: https://www.ebi.ac.uk/about

[53] S. Tucker, "Wearable health, fitness trackers, and the quantified self." Web page, Feb-2014 [Online]. Available: https://www.slideshare.net/drsteventucker/wearable-health-fitness-trackers-and-the-quantified-self

[54] Wikipedia, "Calico(company)." Web page, Sep-2018 [Online]. Available: https://en.wikipedia.org/wiki/Calico_%28company%29

[55] accenture, "Winning with the indusrial internet of things." Web page, 2018 [Online]. Available: https://www.accenture.com/us-en/insight-industrial-internet-of-things

[56] M. Schapranow, “How real-time analysis turns big medical data into precision medicine.” Web page, Sep-2014 [Online]. Available: https://www.slideshare.net/schappy/how-realtime-analysis-turns-big-medical-data-into-precision-medicine

[57] D. Pogorelc, “The body in bytes: Medical images as a source of healthcare big data (infographic).” Web page, Mar-2013 [Online]. Available: https://medcitynews.com/2013/03/the-body-in-bytes-medical-images-as-a-source-of-healthcare-big-data-infographic/

[58] debategraph, “RWJF symposium – june 2012.” Web page, Jun-2012 [Online]. Available: https://debategraph.org/Poster.aspx?aID=77

[59] Indiana university Bloomington, “Million sequence clustering.” Web page, 2008 [Online]. Available: http://salsahpc.indiana.edu/millionseq/mina/16SrRNA_index.html

[60] C. WODEHOUSE, “Should you use mongodb? A look at the leading nosql database.” Web Page, 2018 [Online]. Available: https://www.upwork.com/hiring/data/should-you-use-mongodb-a-look-at-the-leading-nosql-database/

[61] Guru99, “Introduction to mongodb.” Web Page, 2018 [Online]. Available: https://www.guru99.com/mongodb-tutorials.html#1

[62] MongoDB, “Https://www.mongodb.com/.” Web Page, 2018 [Online]. Available: https://docs.mongodb.com/manual/introduction/

[63] M. Papiernik, “How to install mongodb on ubuntu 18.04.” Web Page, Jun-2018 [Online]. Available: https://www.digitalocean.com/community/tutorials/how-to-install-mongodb-on-ubuntu-18-04

[64] J. Ellingwood, “Initial server setup with ubuntu 18.04.” Web Page, Apr-2018 [Online]. Available: https://www.digitalocean.com/community/tutorials/initial-server-setup-with-ubuntu-18-04

[65] MongoDB, Databases and collections, 4.0 ed. New York, New York, USA: MongoDB Inc, 2008 [Online]. Available: https://docs.mongodb.com/manual/core/databases-and-collections/

[66] J. M. Craig Buckler, “Using joins in mongodb nosql databases.” Web Page, Sep-2016 [Online]. Available: https://www.sitepoint.com/using-joins-in-mongodb-nosql-databases/

[67] MongoDB, Lookup (aggregation), 3.2 ed. New York City, New York, United States: MongoDB Inc, 2008 [Online]. Available: https://docs.mongodb.com/manual/reference/operator/aggregation/lookup/

[68] MongoDB, MongoDB package components - mongoexport, 4.0 ed. New York City, New York, United States: MongoDB Inc, 2008 [Online]. Available: https://docs.mongodb.com/manual/reference/program/mongoexport/

[69] MongoDB, Security, 4.0 ed. New York City, New York, United States: MongoDB Inc, 2008 [Online]. Available: https://docs.mongodb.com/manual/security/

[70] MongoDB, “MongoDB atlas.” Web Page, 2018 [Online]. Available: https://www.mongodb.com/cloud/atlas

[71] I. MongoDB, “PyMongo 3.7.1 documentation.” Web Page, 2008 [Online]. Available: https://api.mongodb.com/python/current/api

[72] A. J. J. Davis, “Announcing pymongo3.” Web Page, Apr-2015 [Online]. Available: https://emptysqua.re/blog/announcing-pymongo-3/

[73] M. Dirolf, “PyMongo.” Web Page, Jul-2018 [Online]. Available: https://github.com/mongodb/mongo-python-driver

[74] N. Leite, “MongoDB and python.” Web Page, Mar-2015 [Online]. Available: https://www.slideshare.net/NorbertoLeite/mongodb-and-python

[75] V. Oleynik, “How do you use mongodb with python?” Web Page, Mar-2017 [Online]. Available: https://gearheart.io/blog/how-do-you-use-mongodb-with-python/

[76] I. MongoDB, “Installing / upgrading.” Web pages, 2008 [Online]. Available: http://api.mongodb.com/python/current/installation.html

[77] R. Python, “Introduction to mongodb and python.” Web Page, 2016 [Online]. Available: https://realpython.com/introduction-to-mongodb-and-python/

[78] W3Schools, “Python mongodb create database.” Web Page, 1999 [Online]. Available: https://www.w3schools.com/python/python_mongodb_create_db.asp

[79] I. MongoDB, “PyMongo 3.7.1 documentation.” Web Page, 2008 [Online]. Available: https://api.mongodb.com/python/current/tutorial.html

[80] N. O’Higgins, PyMongo & python. O’Reilly, 2011 [Online]. Available: http://img105.job1001.com/upload/adminnew/2015-04-07/1428393873-MHKX3LN.pdf

[81] I. MongoDB, “PyMongo 3.7.1 documentation.” Web Page, 2008 [Online]. Available: https://api.mongodb.com/python/current/examples/aggregation.html

[82] MongoDB, “PyMongo 3.7.2 documenation.” Web Page, 2008 [Online]. Available: https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/

[83] MongoDB, “PyMongo 3.7.2 documenation.” Web Page, 2008 [Online]. Available: https://docs.mongodb.com/manual/core/map-reduce/

[84] MongoDB, “PyMongo v2.0 documentation.” Web Page, 2008 [Online]. Available: https://api.mongodb.com/python/2.0/examples/map_reduce.html

[85] MongoDB, “PyMongo 3.7.2 documenation.” Web Page, 2008 [Online]. Available: https://api.mongodb.com/python/current/examples/copydb.html

[86] MongoEngine, “MongoEngine user documentation.” Web Page, 2009 [Online]. Available: http://docs.mongoengine.org/

[87] Wikipedia, “Object-relational mapping.” Web Page, May-2009 [Online]. Available: https://en.wikipedia.org/wiki/Object-relational_mapping

[88] MongoDB, “Flask-mongoengine.” Web Page, 2008 [Online]. Available: http://docs.mongoengine.org/guide/defining-documents.html

[89] MongoEngine, “User guide: Document instances.” Web Page, 2009 [Online]. Available: http://docs.mongoengine.org/guide/document-instances.html

[90] MongoEngine, “2.1 installing mongoengine.” Web Page, 2009 [Online]. Available: http://docs.mongoengine.org/guide/installing.html

[91] MongoEngine, “2.2 connection to mongodb.” Web Page, 2009 [Online]. Available: http://docs.mongoengine.org/guide/connecting.html

[92] MongoEngine, “User guide 2.5. Querying the database.” Web Page, 2009 [Online]. Available: http://docs.mongoengine.org/guide/querying.html

[93] wikipedia, “Flask (web framework).” Web Page, 2010 [Online]. Available: https://en.wikipedia.org/wiki/Flask_(web_framework)

[94] MongoDB, “Flask-pymongo.” Web Page, 2008 [Online]. Available: https://flask-pymongo.readthedocs.io/en/latest/

[95] MongoDB, “Flask mongoalchemy.” Web Page, 2008 [Online]. Available: https://pythonhosted.org/Flask-MongoAlchemy/

[96] MongoDB, “Flask-mongoengine.” Web Page, 2008 [Online]. Available: http://docs.mongoengine.org/projects/flask-mongoengine/en/latest/

[97] Wikipedia, “Flask (web framework).” Web Page, Oct-2018 [Online]. Available: https://en.wikipedia.org/wiki/Flask_(web_framework)

Table of Contents