CINET: A CyberInfrastructure for Network Science

Sherif Elmeligy Abdelhamid[1], Richard Alo[2], S. M. Arifuzzaman[1], Pete Beckman[3], Md Hasanuzzaman Bhuiyan[1], Keith Bisset[1], Edward A. Fox[1], Geoffrey C. Fox[4], Kevin Hall[1], S.M.Shamimul Hasan[1], Anurodh Joshi[1], Maleq Khan[1], Chris J. Kuhlman[1], Spencer Lee[1], Jonathan P. Leidig[1], Hemanth Makkapati[1], Madhav V. Marathe[1], Henning S. Mortveit[1], Judy Qiu[4], S.S. Ravi[5], Zalia Shams[1], Ongard Sirisaengtaksin[2], Rajesh Subbiah[1], Samarth Swarup[1], Nick Trebon[3], Anil Vullikanti[1], and Zhao Zhao[1]

[1]Virginia Tech, Blacksburg, VA
[2]University of Houston, Houston, TX
[3]Argonne National Laboratory, Chicago, IL
[4]Indiana University, Bloomington, IN
[5]University at Albany, SUNY, Albany, NY

***Abstract*—Network science is increasingly used in academia and industry to solve problems in many fields. We describe a newly built and deployed cyberinfrastructure for network science (CINET) that is publicly available. CINET contains two applications: (i) Granite, which generates graphs and computes measures on generated graphs and also on realistic networks that are stored in a graph library; and (ii) GDSC, which computes dynamics on graphs. The foundation of CINET is a cyberinfrastructure consisting of a User Interface, Digital Library, and Backend. These three components expose functionality to the user and handle all aspects of producing, managing, and fulfilling user requests; directing and controlling computations on high performance computing clusters; and data management. The system and illustrative use case descriptions focus on the user community and thereby also serve as a user introduction to CINET.**

I. Introduction

A. Motivation

Network science research has been expanding at an ever quickening pace since the mid 1990s, as indicated by the numbers of publications related to complex networks [1]. This is not surprising, as the list of application areas employing graph abstractions, theory, and algorithms, and/or (agent-based) modeling and simulation, includes biology [18], ecology [4], cell immunology [16], social sciences (e.g., collective action, mass movements, revolutions, repression, emotions, technology adoption, drug use, drinking, obesity) [29], [20], [13], [35], health sciences [36], economics [10], computer networks [31], epidemiology [32], statistical physics [6], and language evolution [5]. Clearly, network science is useful for understanding system properties and behaviors. Hence, software that can perform graph-based computations is of great value. In this paper, we describe a newly released, free web-based cyberinfrastructure for network science (CINET) for performing graph-based computations.

B. System Overview

Figure 1 describes the system at a high level. A user logs onto the system using a web browser. Broadly speaking, two classes of computations can be performed. First,

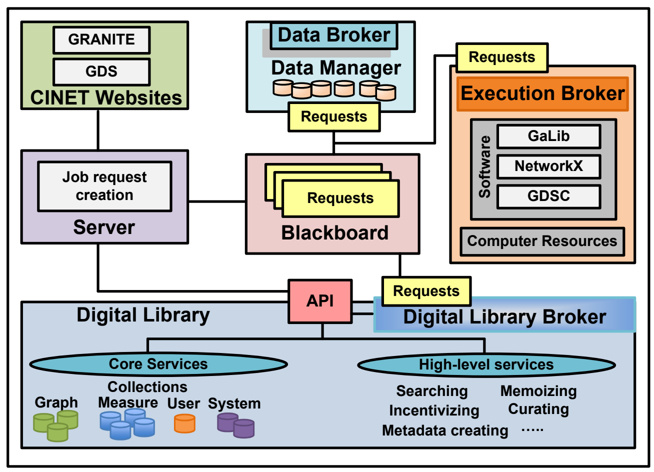

Fig. 1. High level overview of CINET system components and interactions.

the Granite system (with GaLib and NetworkX computation engines) generates graphs, computes a host of graph measures, and finds subgraph motifs. It has over 60 such operations. Many realistic (i.e., mined) graphs from the literature are also provided. Second, the graph dynamical systems calculator (GDSC) system (with the GDSC compute engine) computes dynamics on networks. The cyberinfrastructure (CI) is represented by the rest of Figure 1 and includes the Web-Based User Interface (UI), a Digital Library (DL), and a multi-component Backend (e.g., Job Request Server, Blackboard, Data Broker, and Execution Broker) for job and data management and high performance computing (HPC). Both the applications and the CI are described later in more detail. This system is hosted by Virginia Tech and is free for public use.

C. Example Use Cases

Two use cases are presented that illustrate the utility of CINET. A user here may be a student, teacher, researcher, or practitioner.

Case 1: A user selects a pre-existing graph and a group of graph measures such as distributions of all-pairs shortest paths and of clustering coefficent, and finding pentagon subgraphs in the graph. She clicks a submit button and CINET provides status updates as computations progress. A notification appears when computations complete. Depending on measures selected, graphs may be up to a few billion nodes in size. Output is given in two forms: (i) the raw data (distributions and counts), and (ii) plots (in *.pdf or *.eps format). Both forms of output are displayed on-screen. Data and plots may be downloaded to a user's machine.

Case 2: A user is interested in dynamics on a graph. She selects a type of graph (e.g., circle, lattice) and number of nodes in the network. She specifies a local function (such as the *nor* function) that describes how nodes change state based on influence from adjacent (neighboring) nodes. The state of a node can be either 0 or 1. A choice of update sequence is made; either all nodes execute their local functions simultaneously or sequentially. The sequential option is chosen and the submit button is clicked. All user inputs (except number of nodes) are specified from drop-down lists. Job status is displayed as GDSC computes all state transitions. Results are details of the long-term graph dynamics, and these data can be viewed on-screen and downloaded as a text file.

D. Contributions

The major contributions of this work follow.

1. A free web-based HPC large-graph analysis tool. We envision a user base of students, teachers, domain experts, and investigators that do not have the means to build such systems or lack access to HPC resources for large graph analyses. Properties of pre-existing and user-generated graphs can be computed, as can dynamics on selected graphs. Particularly for Granite, large graphs (those with a few billions of nodes) can be analyzed. The system is intended to significantly reduce the turn-around time for computing answers to questions.

2. An extensible system. The CINET system is extensible in many dimensions. For the Granite system, mined or realistic networks can be added by the user community. Software to compute new graph measures and new graph generators can be added (by users). For the GDSC system, new local functions that describe dynamics can be added, as can additional node update schemes, and state spaces. (For GDSC functionality, system administrators must currently add new features, but the system is designed for such enhancements.) These application-based extensions are made with no alterations in the infrastructure. There are several avenues for extending the CI; e.g., adding computing resources and adding DL features. The CI can also support new applications.

3. Promotion of remote interdisciplinary collaboration. Users from different geographic regions can collaborate with CINET. Moreover, this tool facilitates investigations by multidisciplinary teams whose areas of expertise often reside in different organizations.

4. Common repository. As demonstrated above, the network science-related literature spans many fields. As this trend is expected to continue, it becomes increasingly more difficult for researchers to keep abreast of the progress of others. Tools such as CINET offer a common site where people can go to produce and share results, and contribute functionality to the user community.

E. Test Drive

Reviewers can try the CINET Granite system at `http://ndssl.vbi.vt.edu/granite/` and the CINET GDSC system at `http://ndssl.vbi.vt.edu/gdscalc/` For both systems, the username and password are `demo` and `demo`, respectively.

Organization. The remainder of the paper is organized as follows. Related work is addressed in Section II. The system architecture is explained in Section III and ties together all other sections. Applications—Granite and GDSC—are described in Section IV. Two of the three main components of CI—the UI and DL—are described in Sections V and VI, respectively. Owing to space limitations, the low-level details of the third CI component, the backend, are omitted. However, a high level treatment is provided in the system architecture description. Section VII concludes the paper.

II. Related Work

Numerous workflow management systems exist including Kepler [26], Taverna [30], Triana [27], and Pegasus [8], but are not tailored specifically for simulation workflows.

Existing digital library packages are not capable of supporting large-scale scientific research environments. Scientific digital libraries are an emerging approach related to modeling, managing, analyzing, supporting, and understanding scientific research systems. The eScience and cyberinfrastructure research communities are actively attempting to improve scientific data management practices. Examples of scientific data management projects include earthquake simulation repositories [17], embedded sensor network DLs [3], community earth systems [9], D4Science II [23], mathematical-based retrieval [38], chemistry systems [25], national research data plans [19], and science portals [28]. However, existing systems do not provide the types of services required to support modeling and simulation. These inadequacies motivated the development of our simulation-supporting digital library.

Several network analysis tools exist including the Stanford Network Analysis Project (SNAP) [24] which is a general purpose network analysis and graph mining library. SNAP is also available through NodeXL which is a graphical front-end that integrates network analysis into Microsoft Office and Excel. Another toolkit, Network Workbench [34], has been used for biomedical, social science, and physics research. Network Workbench provides an online portal for researchers, educators, and practitioners. PEGASUS [11] is a peta-scale, distributed graph mining system that runs on clouds. PEGASUS also provides large scale algorithms for important graph mining tasks. Pajek [2] is a tool for the analysis and visualization of networks having thousands or millions of vertices. NetworkX [15] is an open source software package for the generation and study of complex networks. NetworkX contains various graph algorithms.

From a dynamics perspective, [37] provides a web-based tool for computing trajectories and phase spaces. Local functions can be combinations of logical and, or, and not, and can also be additive and multiplicative.

Overall, few systems can match CINET in high performance computing, few systems provide CINET's number of features, and we are not aware of any system that matches CINET in both dimensions. Furthermore, because we host the system, analysts benefit from the use of HPC resource usage. To exploit other publicly available software, users must have access to and install the software on HPC machines.

III. System Architecture

The CI of CINET (also referred to simply as CINET) is a distributed system that constitutes a set of well-defined processes/services that coordinate to perform a given request and its associated tasks. CINET embraces a JavaSpaces-based architecture that relies on persistent object exchange for loosely-coupled coordination between services. This supports extensibility and component change-out. Figure 1 depicts the high-level architecture of the CINET framework with key components that are discussed in this and the following sections in greater detail.

A. Blackboard

The blackboard is the central communication and coordination mechanism for CINET. It is currently implemented with a JavaSpace. It provides asynchronous, loose coupling of system components. Components do not need to be aware of the existence of the other components in the system. They simply put requests onto the blackboard and wait for them to be fulfilled.

Requests are Java objects that contain details about how the request is to be fulfilled, in the form of an embedded workflow. These are active requests, not simply collections of data. For instance, an analysis request contains not only the parameters to run the analysis, but a runner object that contains the workflow to run the analysis, including pre- and post-processing, and validation of the output produced.

B. Brokers

A broker is a component that is responsible for providing a service. It does this by monitoring the blackboard for specific requests that it can fulfill. It takes these requests from the space and executes the workflow embedded in the request. In the process of executing the workflow, it may put requests for other services onto the blackboard for other brokers to fulfill. CINET framework has primarily the following brokers:

- Execution Broker: The execution broker is responsible for identifying execution requests and running each on a specific machine. It does this by constructing system-specific job submission scripts and monitoring the progress of the execution. Results generated by execution are typically transferred with the help of the Data (Management) Broker.
- Data (Management) Broker: A data manager is responsible for managing the data resources that reside on a system. A resource may be a path to a file, or a "fully qualified" database (DB) query. The data manager will also request the transfer of non-local datasets, and the creation of non-existent datasets. It may also make decisions about transfer versus regeneration, purging unused datasets, prefetching data, and the like.
- Digital Library Broker: Digital Library Broker is responsible for communicating with DLs and fetching appropriate information that is required for the end-to-end execution of an analysis request. Some of the key functions of Digital Library Broker are: add/remove graph(s)/measure(s), add/retrieve/update information about graph(s)/measure(s), add/retrieve/remove execution results, and so forth. The DL is described more fully in Section VI.

C. CINET Interface

The CINET Interface consists of user interfaces and web applications that enables a user to submit analysis requests, add graph measure software, add graphs and/or perform administrative tasks. The CINET interface is discussed in Section V.

D. Compute Resource

Compute Resources are the physical resources on which jobs are executed. Current resources are two HPC Linux clusters at Virginia Tech. Potential compute resources include traditional HPC clusters, compute grids (e.g., Open Science Grid) and clouds (e.g., Amazon Web Services), volunteer computing platforms (e.g., BOINC), or dedicated servers. A typical compute resource runs NetworkX and GaLib components (which constitute the Granite compute engine) and the GDSC engine. These components contain the binaries for performing graph analyses. Also, execution and data management brokers run on compute resources to supervise the generation and management of results.

E. Applications

Current applications are Granite and GDSC.

IV. Graph Analysis and Modeling

In this section, we overview system capabilities for producing and evaluating structure properties of graphs, and for modeling dynamics on them.

A. Graph Algorithm for Static Analysis of Networks

Static analysis of networks are meant to compute various static measures associated with networks (e.g. number of triangles, diameter, BFS tree). GaLib and NetwrokX are the computation engines that provide CINET with necessary computing capabilities for structural analysis of networks and

TABLE I
NUMBER OF GALIB AND NETWORKX MODULES IN CINET

Type	GaLib	NetworkX
Graph Generator	11	11
Centrality	14	11
Shortest path and connectivity	15	8
Subgraph/motif counting	8	2
Others	15	3
Total measures	63	35

TABLE II
PERFORMANCE COMPARISON BETWEEN GALIB AND NETWORKX FOR THREE MEASURES ON G(500K, 20) GRAPH; NETWORKX IS NOT ABLE TO RUN ON LARGE GRAPHS

Measures	Runtime (sec)		Memory (GB)	
	GaLib	NetworkX	GaLib	NetworkX
Single Source Shortest Path	12	275	0.22	3.6
Check Connectivity	14	360	0.22	3.6
Counting Triangles	20	480	0.22	3.6

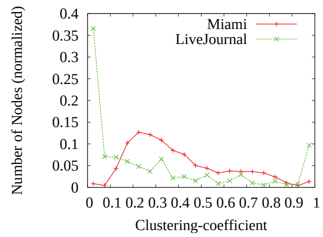

Fig. 2. Clustering Coefficient distribution of two graphs, Miami and LiveJournal [24]

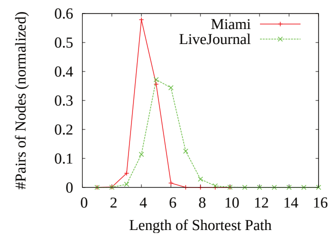

Fig. 3. Shortest path distribution of two graphs, Miami and LiveJournal [24]

determining various metrics of interests associated with real-world and artificial networks.

GaLib, a Graph Algorithm Library written in C++, is developed at Network Dynamics and Simulation Science Laboratory, Virginia Tech. Though there are some graph libraries already developed by various groups and labs (e.g. SNAP [24], Pajek [2], Network Workbench [34]), challenges arisen from the analysis of practical large graphs demand new capabilities and consideration. Firstly, the graphs we consider (especially social contact graphs) have millions of nodes and edges, which leads to significant space-time challenges. Secondly, another prominent characteristic of the graphs we study is that there are labeled nodes having attributes such as demographic information (e.g., age, immunity level, gender, and household income), and edges having attributes such as contact time, nature of contact, location, etc. Finally, methods in literature do not scale easily. GaLib, the new Graph Algorithm Library, is intended for analyzing large scale labeled graphs. GaLib provides efficient implementations of various classical and new graph measures that are motivated by the analysis of social contact graphs and disease dynamics on such graphs, including (i) analysis of the subgraph structure and relational labeled graph queries and (ii) graph measures related to disease dynamics on social contact networks. For some of these algorithms, GaLib implements sampling based approximation algorithms known in literature, but with error guarantees that can be controlled by the users, leading to significant speedups. Moreover, GaLib has impnlemented many parallel algorithms, e.g., subgraph counting (induced and non-induced), generalized clique counting, triangle counting, computing of clustering coefficient. Data structures and algorithms included in GaLib are carefully tuned to be capable for running on large graphs containing up to millions of nodes.

NetworkX [15] is an open source software package for the generation and study of the structure and dynamics of various complex networks. It has been developed at Los Alamos National Laboratory and was first released in 2005 for public use. NetworkX uses Python because it is a very powerful language having rapid prototyping and large number of built-in data structures. NetworkX supports various functions for graph algorithms that make it simple and flexible to use it for the representation and manipulation of many complex networks, including undirected, directed and multi-graphs, in different types of data format. NetworkX has a large collection of standard graph algorithms' implementation and several types of graph generators (i.e., classic, random graphs and synthetic networks) with the ability of graph drawing. NetworkX is very useful for the study, analysis of properties, structures and dynamics of different types of complex networks (i.e., social, biological, infrastructure, contact networks), for classroom use and for analysis of newly developed algorithms performance. For each NetworkX function available in CINET, a Python script has been written that is used by the CINET front end.

Statistics related to different measures implemented by GaLib and NetworkX as a part of CINET is shown in Table I. Improved performance of GaLib measures in terms of runtime and memory consumption is shown in Table II. Here, G(500K, 20) is an Erdös-Rényi random graph with 500,000 nodes and average degree 20, and Miami is a synthetic, but realistic, social contact network for Miami city with 2,092,147 nodes and average degree 50.38. A typical analysis using GaLib is provided in Fig. 2 and 3. A CINET user can work with any network. Users can add own networks to the system and analyze them using CINET. For convenience, 36 real-word and various random networks have been included in the system, and more such networks will be added in the future.

B. GDS Calculator

1) Social Sciences As A Case Study: We use social sciences to demonstrate the utility of the GDS Calculator (GDSC). Often in the social sciences, human populations are represented as graphs, where vertices (nodes) denote agents and edges correspond to interactions betweeen them. Threshold systems have been used for decades to model collective action and the influence of one (human) agent on another [14], [33], [7]. In its most basic form, in a two-state system, an agent v will change from the non-participating state (e.g., state 0) to the participating state (e.g., state 1) when a threshold θ_v number of its neighbors are in the participating state. GDSC is used to explore (human) interactions in various domains.

2) Illustrative Results: Selected inputs and outputs are displayed graphically in Figure 4 for the BITHRESHOLD vertex function and $\{0, 1\}$ vertex state space, where for each vertex (agent) v in a population, $\theta_{v,up} = 1$ and $\theta_{v,down} = 3$. Thus, for v in state 0, if at least $\theta_{v,up}$ neighbors are in state 1, then v transitions to state 1; otherwise v's state x_v remains 0. For v in state 1, if less than $\theta_{v,down}$ vertices in its closed neighborhood (i.e., including the state of v) are in state 1, then v will transition to state 0; otherwise it remains in state 1. The graph X, at the left side of Figure 4, is a 4-vertex square, known formally as Circle-4 (Circ_4). The right graphic is the phase space (i.e., all 16 system state transitions) for synchronous vertex (or local) function update; i.e., all vertices update their states simultaneously. There are three 2-cycles in the phase space (identified by the green arrows). Suppose we have a system conforming to these conditions, where a vertex in state 0 means that the agent is not participating in a revolt, and state 1 means that an agent is participating. Then there are many interesting observations to be made about this system, but we give only one. For any initial system state other than $(0, 0, 0, 0)$ or $(1, 1, 1, 1)$, the three 2-cycles indicate that in the long term, two agents will be participating at any one time (i.e., exactly two vertices have states of 1 on the 2-cycles). In GDSC, a user may select from different graph classes and numbers of vertices, vertex functions, and vertex function update schemes.

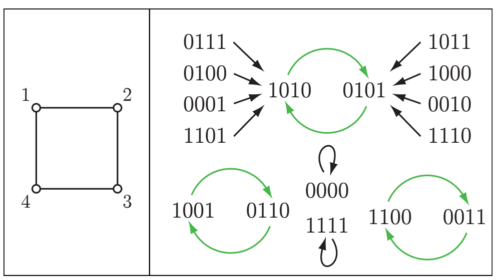

Fig. 4. (Left) graph Circ_4; and (right) phase space for synchronous BITHRESHOLD GDS map with 2-cycles shown in green.

V. USER INTERFACE

The Granite system also refers to the web application of CI that supports the interactive UI. It allows users to work with a wide range of graphs and measures. The objective of this system is to move beyond the traditional database and information management system to integrate content, community, and services. The Granite system interacts with the user through a tab-based interface as shown in Figure 5. Smart GWT is used for Granite development. The following is a brief description of the Granite UI.

A. Workspace Tab

The workspace tab allows a user to select one or more graphs within CINET. These include instances of random graphs (RN), social networks (SN) generated by various groups, and several of the networks from the SNAP repository [24]. Also the workspace tab provides a list of available measures and generators that can be applied to selected graphs.

B. Results Tab

Once the user clicks on the analyze button, the system will start to process the requested jobs. The Results tab shows the progress of each job and allows the user to view output.

C. Add New Graph Tab

The UI for adding new graphs provides for a two-step process. A user needs to upload a graph and then needs to complete a metadata form for the new graph. After uploading a graph, subservices (graph conversion, graph consistency checking, and basic metadata extraction) are sequentially initiated and performed. The new graph is converted to all other available graph formats, and then a GaLib measure is used to check graph consistency. If a graph fails the consistency check, the system rejects the graph and informs the user with error messages returned by the GaLib measure. Otherwise, the system automatically extracts basic properties of the graph, which are normally called degree statistics such as numbers of nodes and edges, degree distribution, etc., and then transfers the graph file to the production repository to make it available in the list of graphs of the main Granite UI.

D. Add Measure Tab

The Add Measure service provides a UI to add a new measure and associated information to CINET. See Figure 6. By "adding a new measure," we mean adding an executable to CINET, along with required parameters. To add a measure, a user needs to insert information for measure name, description, tool type (Galib or NetworkX), output files name(s), executable file name, measure parameter and runtime information of the measure. Based on user input, the system dynamically creates different types of widgets to collect information about parameters and output file names. It allows a user to insert information about parameter pype, name, and value. Also several data validation checks are performed by this service. If a measure is successfully added to the system then it will appear in the measure list. Currently, this service is only available for an administrative user. This is to ensure that new measure codes are validated before being exposed to the user community.

VI. DIGITAL LIBRARY

Digital libraries (DLs) have greatly advanced since the community was formed in the early 1990s. However, efforts into supporting scientific and simulation-baed research have been minimal until now. As a part of CINET, we have developed a simulation-supporting digital library (SimDL) [21], [22]. SimDL is a framework for producing DL instances that support large-scale simulation-based infrastructures. Formal definitions using the 5S formal framework were produced that precisely describe the services required to support this community and identify functionality absent in current open-source DL software [12]. See Figure 7 for the metamodel produced by these formalizations. From these functional definitions, the minimal set of simulation-supporting services were identified and implemented to produce a software toolkit. The final

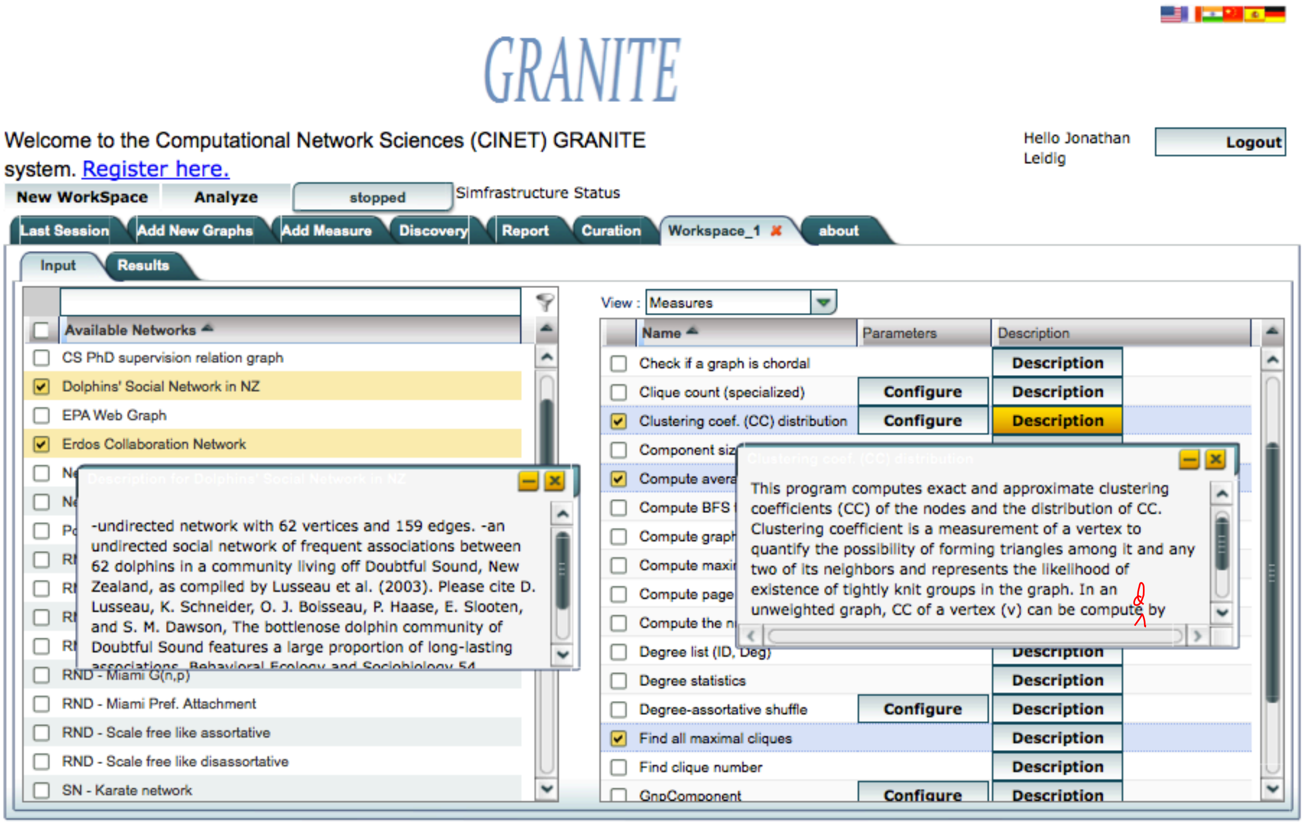

Fig. 5. Granite System User Interface.

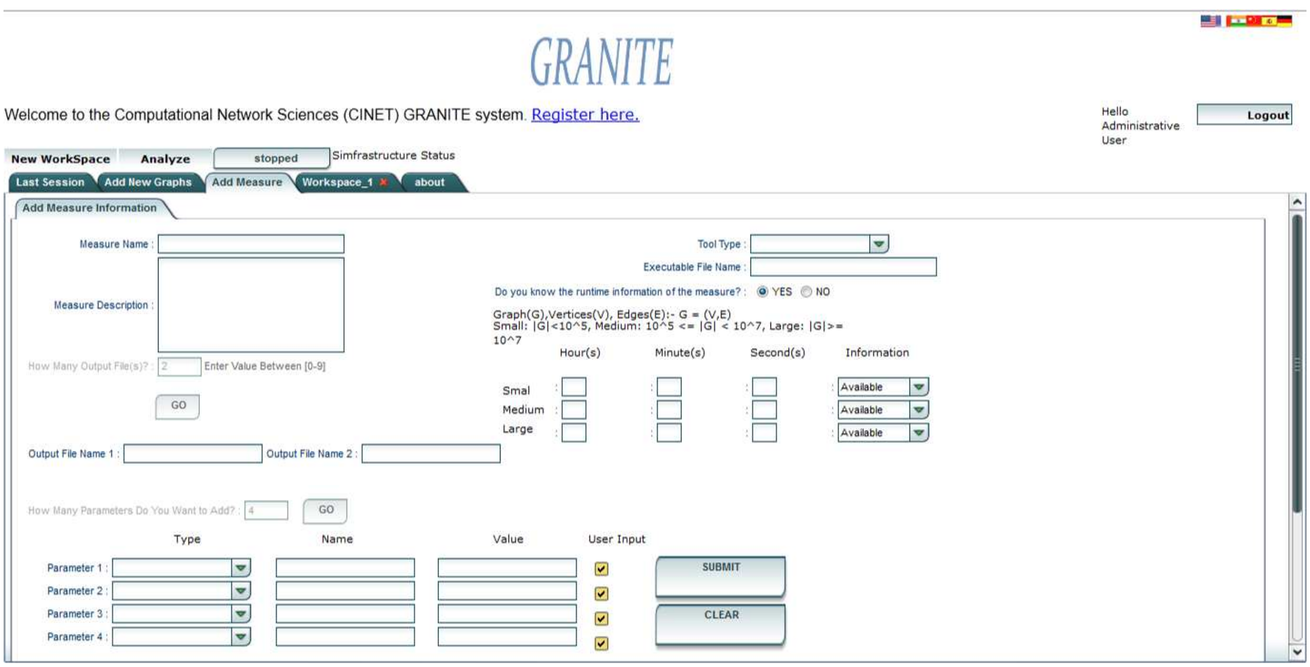

Fig. 6. UI for adding a new graph measure.

result was a software package that may be deployed in various simulation research infrastructures. The SimDL instance utilized in this work includes core information storage and retrieval services to manage graphs, measures, and results, as well as higher-level simulation supporting services to support infrastructure and scientific tasks.

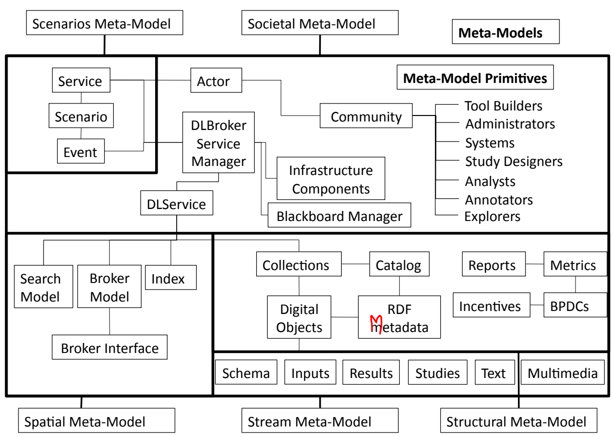

Fig. 7. Metamodel leading to SimDL's architectural design.

A. Core Functionality

The digital library component provides the CINET system with core services for retrieving and storing various metadata for network graphs, network analysis tools and their measures, hosts, users, etc. as shown in Figure 1. CINET uses the metadata provided by the digital library to create job requests which are compositions of graph, measures, and High Performance Resource (HPC) information in the form of shell scripts. The system also receives decisions of tool supportabilities based on sizes and estimated running times of graphs and measures selected by users.

Table III shows the primary metadata for each major data type used by the CINET system. These are the fundamental data used by the core DL services mentioned above and the simulation supporting services which will be described in the next section.

B. Simulation Supporting Services

Simulation supporting services aim to provide incentives for using the system and efficient data management services for infrastructure components. Each of these services described here was designed and implemented to provide scalable functionality appropriate for highly numeric, data-intensive scientific content. While other DL systems often provide these services in other contexts, e.g., full-text publications, existing implementations do not scale well for automatic indexing and support of large quantities of simulation-produced digital objects. SimDL utilizes a DL broker to communicate with other components, e.g., UIs, HPC systems, and simulation models. Through this broker, other components can contribute, register, and query collections of new datasets, simulation

TABLE III
PRIMARY METADATA OF EACH MAJOR DATA TYPE.

Graph	Measure	CINET logic	User
Name	Name	Network tool supportability	ID
Description	Description	Estimated running time based on graph size	Real name
Number of Node/Edge	Parameters		Group
File path	command		password
Network type	category		
Graph format			
Date of addition			
Owner/right			
Source			

studies, metadata records, etc. The broker is also utilized to request information from the following set of simulation-specific service implementations. The formal definitions leading to the design of high-level services are overviewed in Figure 8.

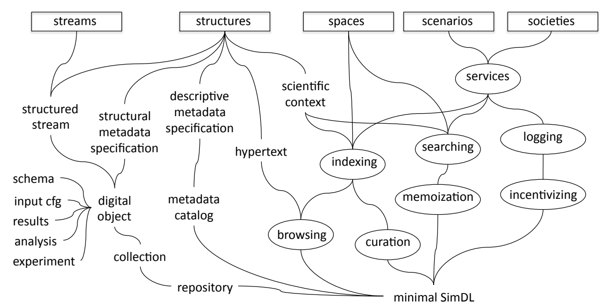

Fig. 8. Formalisation of the minimal simulation-supporting digital library.

To incentivize researchers and educators to contribute to and utilize CINET and the infrastructure, components log the submission and usage of graphs, measures, HPC systems, and user behaviors. Thus, we have a logging system that tracks what types of activities and data products are related to users. A DL incentivization service then generates reports detailing statistics to each class of user, e.g., content provider, software provider, student, and HPC system administrator. The memoization service maintains existing simulation results and returns these to the UI if a deterministic simulation study is designed and submitted multiple times. This minimizes unnecessary workloads on contributed systems and provides quicker access time to end-users. The curation service recommends when digital objects should be archived, preserved, migrated, and deleted from the DL based on administrator-defined rules. Additional services execute query searches, filter content lists, support automated metadata indexing, and tracking provenance through the simulation workflow. This set of minimal SimDL services supports UIs, workflows, HPC systems, simulation models, and user tasks, using automated, scalable, and domain-free software implementations.

VII. Conclusions

We have described CINET, a cyberinfrastructure for analysis of large graphs (e.g., up to billions of nodes). The CI provides a middleware platform connecting high performance compute engines on the back end (that run on HPC clusters), with a DL and Data Managers that generate, decide the routing of, and control the flow of job submissions made by users through a UI. Currently, this CI supports two applications: (i) Granite, for computing over 60 measures on a host of predefined graphs and those produced with provided graph generators; and (ii) GDSC, for computing dynamics on graphs. The system–both the CI infrastructure and the applications–is extensible in many ways. For example, the CI can support other applications. The user base is envisioned to be students, teachers, domain experts, researchers, and practitioners from a variety of application areas. Free for public use, the system is intended to foster (remote, interdisciplinary) collaboration and provide a common repository for network science investigations and for users to share data and analysis methods.

The CINET website provides all information related to the CINet project(Cyberinfrastructure for Network Science). The website also acts as an online repository for educational materials related to network science that includes courses and notes, presentations and survey papers. The CINET website can be accessed at http://ndssl.vbi.vt.edu/cinet/cinet_project/.

Acknowledgment

This comes from the NDSSL website. Perhaps Madhav wants to delete some of these??

We thank our external collaborators and members of the Network Dynamics and Simulation Science Laboratory (NDSSL) for their suggestions and comments. This work has been partially supported by NSF HSD Grant SES-0729441, NSF PetaApps Grant OCI-0904844, NSF NETS Grant CNS-0831633, NSF CAREER Grant CNS-0845700, NSF NetSE Grant CNS-1011769, NSF SDCI Grant OCI-1032677, NSF ICES Grant CCF-1216000, DTRA R&D Grant HDTRA1-09-1-0017, DTRA Grant HDTRA1-11-1-0016, DTRA CNIMS Contract HDTRA1-11-D-0016-0001, DOE Grant DE-SC0003957, PSU/DOE 4345-VT-DOE-4261, US Naval Surface Warfare Center Grant N00178-09-D-3017 DEL ORDER 13, NIH MIDAS Grant 2U01GM070694-09, NIH MIDAS Grant 3U01FM070694-09S1, LLNL Fellowship SubB596713, and DOI Contract D12PC00337.

References

[1] ~~Anonymous.~~ National Research Council Committee on Network Science for Future Army Applications. In *Network Science*. The National Academies Press, 2005.

[2] M. A. Batagelj V. Pajek - Program for Large Network Analysis. *Connections*, 21(2):47–57, 1998.

[3] C. L. Borgman, J. C. Wallis, M. S. Mayernik, and A. Pepe. Drowning in data: digital library architecture to support scientific use of embedded sensor networks. In *Proc. JCDL 2007*, pages 269–277, 2007.

[4] C. Campbell, S. Yang, R. Albert, and K. Sheab. A network model for plantpollinator community assembly. *Proceedings of the National Academy of Sciences*, 108(1):197–202, 2011.

[5] A. Cangelosi and D. Parisi. Computer Simulation: A New Scientific Approach to the Study of Language Evolution. In A. Cangelosi and D. Parisi, editors, *Simulating the Evolution of Language*. Springer, 2001.

[6] C. Castellano, S. Fortunato, and V. Loreto. Statistical physics of social dynamics. *Rev. Mod. Phys.*, 81(2):591–646, 2009.

[7] D. Centola and M. Macy. Complex Contagions and the Weakness of Long Ties. *American J. Sociology*, 113(3):702–734, 2007.

[8] E. Deelman, J. Blythe, Y. Gil, C. Kesselman, S. Koranda, A. Lazzarini, G. Mehta, M. A. Papa, and K. Vahi. Pegasus and the Pulsar Search: From Metadata to Execution on the Grid. In *Applications Grid Workshop at the Fifth International Conference on Parallel Processing and Applied Mathematics (PPAM)*, pages 821–830, Czestochowa, Poland, 2003.

[9] R. Dunlap, L. Mark, S. Rugaber, V. Balaji, J. Chastang, L. Cinquini, C. DeLuca, D. Middleton, and S. Murphy. Earth system curator: metadata infrastructure for climate modeling. *Earth Science Informatics*, 1:131–149, 2008. 10.1007/s12145-008-0016-1.

[10] D. Easley and J. Kleinberg. *Networks, Crowds and Markets: Reasoning About A Highly Connected World*. Cambridge University Press, New York, NY, 2010.

[11] Faloutsos. Project Pegasus. http://www.cs.cmu.edu/~pegasus/, 2009. [Online; accessed 17-July-2012].

[12] M. A. Gonçalves, E. A. Fox, L. T. Watson, and N. A. Kipp. Streams, structures, spaces, scenarios, societies (5s): A formal model for digital libraries. *ACM Trans Inf Syst*, 22(2):270–312, 2004.

[13] S. Gonzalez-Bailon, J. Borge-Holthoefer, A. Rivero, and Y. Moreno. The Dynamics of Protest Recruitment Through an Online Network. *Nature Scientific Reports*, pages 1–7, 2011. DOI: 10.1038/srep00197.

[14] M. Granovetter. Threshold Models of Collective Behavior. *American J. Sociology*, 83(6):1420–1443, 1978.

[15] A. A. Hagberg, D. A. Schult, and P. J. Swart. Exploring network structure, dynamics, and function using NetworkX. In *Proceedings of the 7th Python in Science Conference (SciPy2008)*, pages 11–15, Pasadena, CA USA, Aug. 2008.

[16] B. Hancioglu, D. Swigon, and G. Clermont. A dynamical model of human immune response to influenza a virus infection. *Journal of Theoretical Biology*, 246(1):70–86, 2007.

[17] T. H. Jordan. SCEC 2009 Annual Report. *Southern California Earthquake Center*, 2009.

[18] U. Karaoz, T. Murali, S. Letovsky, Y. Zheng, C. Ding, C. Cantor, and S. Kasif. Whole-genome annotation by using evidence integration in functional-linkage networks. *Proceedings of the National Academy of Sciences*, 101(9):2888–2893, 2004.

[19] S. Kethers, X. Shen, A. E. Treloar, and R. G. Wilkinson. Discovering Australia's research data. In *Proc. JCDL 2010*, pages 345–348, 2010.

[20] C. Kuhlman, V. Kumar, M. Marathe, S. Ravi, D. Rosenkrantz, S. Swarup, and G. Tuli. Inhibiting the Diffusion of Contagions in Bi-Threshold Systems: Analytical and Experimental Results. In *Proceedings of the AAAI Fall 2011 Symposium on Complex Adaptive Systems (CAS-AAAI 2011)*, pages 91–100, November 2011.

[21] J. Leidig, E. Fox, M. Marathe, and H. Mortveit. Epidemiology experiment and simulation management through schema-based digital libraries. In *Proceedings of the 2nd DL.org Workshop at ECDL*, pages 57–66, 2010.

[22] J. Leidig, E. A. Fox, K. Hall, M. Marathe, and H. Mortveit. SimDL: A Model Ontology Driven Digital Library for Simulation Systems. In *ACM/IEEE Joint Conference on Digital Libraries*, JCDL '11. ACM, 2011.

[23] P. P. Leonardo Candela, Donatella Castelli. D4Science: an e-infrastructure for supporting virtual research. In *Proceedings of IRCDL 2009 - 5th Italian Research Conference on Digital Libraries*, pages 166–169, 2009.

[24] J. Leskovec. Stanford Network Analysis Project. http://snap.stanford.edu/, 2009. [Online; accessed 17-July-2012].

[25] N. Li, L. Zhu, P. Mitra, K. Mueller, E. Poweleit, and C. L. Giles. oreChem ChemXSeer: a semantic digital library for chemistry. In *Proc. JCDL 2010*, pages 245–254, 2010.

[26] B. Lud?scher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, and Y. Zhao. Scientific Workflow Management and the Kepler System. In *Concurr. Comput. Pract. Exper*, page 2006, 2005.

[27] S. Majithia, M. S. Shields, I. J. Taylor, and I. Wang. Triana: A Graphical Web Service Composition and Execution Toolkit. In *Proceedings of the IEEE International Conference on Web Services (ICWS'04)*, pages 514–524. IEEE Computer Society, 2004.

[28] R. W. Moore, A. Rajasekar, M. Wan, Y. Katsis, D. Zhou, A. Deutsch, and Y. Papakonstantinou. Constraint-based Knowledge Systems for Grids, Digital Libraries, and Persistent Archives: Yearly Report. In *SDSC TR-2005-5*, 2005.

[29] D. J. Myers and P. E. Oliver. The opposing forces diffusion model: the initiation and repression of collective violence. *Dynamics of Asymmetric Conflict*, 1:164–189, 2008.

[30] T. Oinn, M. Greenwood, M. Addis, N. Alpdemir, J. Ferris, K. Glover, C. Goble, A. Goderis, D. Hull, D. Marvin, P. Li, P. Lord, M. Pocock, M. Senger, R. Stevens, A. Wipat, and C. Wroe. Taverna: lessons in creating a workflow environment for the life sciences. In *Concurrency and Computation: Practice and Experience*, volume 18, pages 1067–1100, 2006.

[31] R. Puzis, M. Tubi, Y. Elovici, C. Glezer, and S. Dolev. A Decision Support System for Placement of Intrusion Detection and Prevention Devices in Large-Scale Networks. *ACM Transactions on Modeling and Computer Simulation*, 22:1–2, 2011.

[32] T. C. Reluga, J. Medlock, and A. S. Perelson. Backward bifurcations and multiple equilibria in epidemic models with structured immunity. *Journal of Theoretical Biology*, 252:155–165, 2008.

[33] T. Schelling. *Micromotives and Macrobehavior*. W. W. Norton and Company, 1978.

[34] [illegible] Team. Network Workbench Tool. Indiana University, Northeastern University, and University of Michigan. http://nwb.cns.iu.edu, 2006. [Online; accessed 17-July-2012].

[35] J. Tsai, E. Bowring, S. Marsella, and M. Tambe. Empirical evaluation of computational emotional contagion models. In *Proceedings of the 11th International Conference on Intelligent Virtual Agents (IVA 2011)*, 2011.

[36] T. W. Valente. *Social Networks and Health: Models, Methods, and Applications*. Oxford University Press, 2010.

[37] H. Vastani, N. Eriksson, R. Laubenbacher, A. Jarrah, B. Stigler, and F. Hinkelmann. Discrete Visualizer of Dynamics (DVD) v1.0. http://dvd.vbi.vt.edu/cgi-bin/git/dvd.pl, 2012. [Online; accessed 17-July-2012].

[38] J. Zhao, M.-Y. Kan, and Y. L. Theng. Math information retrieval: user requirements and prototype implementation. In *Proceedings of JCDL '08*, pages 187–196, 2008.