COMPUCELL Beta Version 1.6 - User Guide

Jesús A. Izaguirre
Department of Computer Science
University of Notre Dame
USA
izaguirr@cse.nd.edu

Rajiv Chaturvedi
Department of Computer Science
University of Notre Dame
USA
rchaturv@cse.nd.edu

`http://www.nd.edu/~lcls`
compucell@cse.nd.edu

Authors:

Trevor Cickovski
Chengbang Huang

Contributors to COMPUCELL from University of Notre Dame:

Jeffrey Goett
Thomas Keeley
Hugo Martinez
Thierry Matthey
Ryan Pohlman
Thomas Slabach
Patrick Virtue

July 10, 2002

Abstract

This user guide is intended to provide details on the operation and features of COMPUCELL , released with the intention of being a useful tool for conducting simulations of biocomplexity problems. COMPUCELL combines ease of use with high efficiency and accuracy with simple and easily-edited file input, the extended Potts model (an improvement over traditional Potts) and efficient Reaction-Diffusion equation solvers.

This user guide starts with a section describing some of the useful features of COMPUCELL and then will provide the appropriate steps with regard to running the software. Included also are compiling instructions for different platforms, instructions on how to change information for different simulations, and finally a section describing the future of COMPUCELL .

Acknowledgements

- Special thanks to the Center For Applied Mathematics at the University of Notre Dame for helping to fund researchers on the project.
- Parts of the development were supported by a National Science Foundation Biocomplexity grant (PHY-0083653) and a National Science Foundation ACI CAREER award (ACI 01-35195).
- Other significant contributions to front end design were made by Thierry Matthey and Thomas Slabach.

Contents

Chapter 1

Introduction

Biocomplexity is defined as "the study of the unique complex structures and behaviors that arise from the interaction of biological entities (molecules, cells, or organisms". [2] Often the simplest biological phenomena undergoes highly complex physical and chemical processes. Computer software plays an important role in simulating and giving us a clearer view on how these processes work. What COMPUCELL provides is an object-oriented framework designed to maximize efficiency, software flexibility, ease of extension and maintenance, and ease of use.

The hope is that the program can be used to simulate several different types of biological phenomena, taking into account the interaction of all natural processes that play a role, some of which may be very complex. COMPUCELL is currently part of a larger project on bioinformatics that is already underway in the Interdisciplinary Center for the Study of Biocomplexity at the University of Notre Dame.

As mentioned, COMPUCELL has been designed with ease of use being one of the goals at the forefront. It currently employs a file-based front end that allows the user to easily change from one complex simulation to the next without much difficulty. At this time COMPUCELL is centered around celluar processes, and its computational engine currently consists of two main portions: the first are efficient Schnakenberg Reaction Diffusion equation solvers that enable the establishment of a known chemical concentration field, and the second an implementation of the extended Potts model that is used to determine cell sorting and clustering overtime [3]. Cells respond to this concentration field and form patterns. These patterns may resemble many possibilities, for example, a growing chicken limb.

There are several useful features of the current COMPUCELL framework. They are described in more detail in the next section.

1.1 Useful Features

1.1.1 File Input

By allowing for initial input to come from files, the COMPUCELL framework makes it much easier for the user with regard to changing simulations, especially as opposed to console input. The user can easily change from one complex simulation to another completely unrelated yet still complex simulation by changing just a few parameters in the configuration file. All input files are structured in a straightforward manner.

1.1.2 Extended Potts Model Simulations

The Potts Model is a popular technique for observing changes in celluar patterns in given environments. This Extended Potts Model, which COMPUCELL uses, builds on this but allow there to be more than one cell spin, constrains cell size and allows for different surface energies between different cell spins. [3]

1.1.3 Monte Carlo Sampling

The Monte Carlo algorithm is currently a popular sampling technique. COMPUCELL has been made to support this algorithm.

1.1.4 Ability to interact with VTK

VTK (stands for Visualization ToolKit) is a popular system designed for graphics and visualization in three-dimensions. COMPUCELL is conveniently able to interact with this useful program to establish a visual interface that the user may clearly view the behavior of cells and the concentrations in the reaction-diffusion field. For more information, see:

`http://www.kitware.com/vtk/`

1.1.5 Implementation of a Moving Potts

By integrating the visualization described above with the extended Potts model, a viewable “moving Potts” is created. This “moving Potts” can be viewed on a local machine or can be easily rendered into a viewable internet movie.

Chapter 2

Getting Started

This chapter will introduce the commands and settings needed to run COMPUCELL . Included are the exact formats of the command line on a UNIX machine and the configuration file containing all initial information to run the simulation. In Chapter 4 we will show three sample configuration files.

2.1 Command Line

The name of the COMPUCELL framework application is, conveniently, `CompuCell`. At a UNIX prompt, a user types `CompuCell` followed by an alternating list of keywords and arguments (see section 2.2.3 for a list of keywords). Thus, the general format for the COMPUCELL execution command is the following:

`CompuCell [--keyword1` *value1* `] [--keyword2` *value2* `] [--keyword3` *value3* `] .......`

Note that keywords must be preceded by two dashes, where a value can be a list of values - e.g. for the keyword PottsLatticeSize. Also note that any keyword-value pair specified on the command line overrides any according pair in the configuration file.

The user may also specify any of these keywords and values in the configuration file that he or she is using. However, *either the exact pathname of the configuration file being used for running* COMPUCELL , *or the exact pathname of the prefix of all initial data files which includes the configuration file, potts initial file, visualization file, and cell parameter file* (all four are described in further detail in the section entitled **Supported File Formats**. For example, the configuration file can be specified one of the following ways:

```
CompuCell [--configfile] <pathname/myconfigfilename> ......
```

The `--configfile` *keyword can be omitted if the configuration file is the first thing specified.*

```
CompuCell .....  --configfile <pathname/myconfigfilename> .....
```

Or the input file prefix can be used:

```
CompuCell --inputfileprefix <pathname/prefix>
```

Expecting the following initial data files:

<pathname/prefix.conf>	(the configuration file)
<pathname/prefix.pif>	(the potts initial file)
<pathname/prefix.viz>	(the visualization file)
<pathname/prefix.cpf>	(the cell parameter file)

At least one of these two options must be specified on the command line. Note: if both are specified, the configuration file will override the input file prefix.

2.1.1 Help Option

A user may type one of the following two possibilities at a UNIX prompt for help with the CompuCell command line:

1. `CompuCell -h`
2. `CompuCell --help`

2.1.2 Version Option

The actual version of CompuCell :

1. `CompuCell -v`
2. `CompuCell --version`

2.1.3 Keyword Option

Output of all actually supported keywords, their default values and their types in the configuration file:

1. `CompuCell -k`
2. `CompuCell --keywords`

2.2 Configuration File

The configuration file is a text file containing a collection of keyword-value pairs specifying the simulation configuration, and I/O files and formats. If an input file prefix is specified on the commandline, the configuration file will be assumed to be that file prefix concatenated with the <.conf>suffix. When using the full name extensions, <.conf>is the suffix of the configuration file.

2.2.1 Format of CompuCell keywords in the configuration file

The configuration file format for CompuCell is quite simple, making it convenient for creation and modification of the file. The advantage of straightforward modification of the configuration file is that the user can very easily switch between simulating the same cell patterns under different initial conditions, or even switch to a different pattern without much trouble. The general format for a CompuCell configuration file is a list of keywords and values, with whitespace between each keyword and value and a newline between each new keyword-value pair:

keyword1	*value1*
keyword2	*value2*
keyword3	*value3*
.	
.	
.	

A list of CompuCell *keywords, possible values and defaults can be found in section 2.2.3*

2.2.2 Sample Configuration File Used For The Simulation of the Growth of a Chicken Limb

```
cellparameterfile chicklimb.cpf
visualizationfile chicklimb.viz
rdinputfile /afs/nd.edu/user28/izaguirr/Research/biocomp/RDdata/
contGammaSmallWindow/assemblyData.dat

pottslatticedim 2
pottsdim 150 150 0
pottslatticesize 150 150 0
temperature 1.0
uniformcelldistribution true
initialcellsize 12
boundaryborder 3
pottsnumsteps 1200

dovisualization false

domainfill 0.99
rdupdatefreq 5

pottsstepsperwindowmovement 90
pottswindowincrementbottom 0 1 0
pottswindowincrementtop 0 15 0
pottswindowsize 15 15 0
```

2.2.3 COMPUCELL keywords and descriptions

Keyword	Type	Description
inputfileprefix	string	Contains the full pathname of the file prefix for all initial data files. If this is specified, there is no need to specify any initial data files as long as each of them can be defined as this file prefix concatenated with the appropriate flag (.conf, .pif, .viz, .cpf).
configfile	string	Contains the full pathname of the configuration file to be used in the simulation. Note: this keyword would only be used on the command line.
pottsinitialfile	string	Contains the full pathname of the potts initial file. This file specifies initial positions in pixels for each cell type and cell spin, including the surrounding medium. This is only necessary if uniformcelldistribution is set to false (meaning the user desires to control the cell distribution). Note: as of now it is not needed at all, COMPUCELL can only handle a uniform cell distribution - though will be implemented in a future version.
visualizationfile	string	Contains the full pathname of the visualization file. This file specifies each cell spin and essential information to run the simulation in VTK. Note this is only necessary if dovisualization is set to true.
cellparameterfile	string	Contains the full pathname of the cell parameter file. This file contains specific information about cell models and types involved in the simulation, including Hamiltonian variables for the Extended Potts Model.
rdinputfile	string	Contains the full pathname of the file containing the initial activator concentrations. A necessary parameter.
dovisualization	boolean (default: false)	Specifies if the user desires visualization. VTK must be installed if this is the case.
uniformcelldistribution	boolean (default: true)	For now, this must be set to true. This will be implemented in a future version of COMPUCELL .
initialcellsize	int	Since a uniform cell distribution is assumed right now, this is a necessary parameter. This specifies the length of a side of the square cells in pixels.
boundaryborder	int	This is also necessary for the same reason of the uniform cell distribution assumption. This specifies the separation distance between cells in pixels.

Keyword	Type	Description
vizfromfile	boolean (default: false)	Specifies if the user would like Potts model visualization information to be read from files. dovisualization must be set to true if this is the case.
datafileprefix	string	The prefix for Potts model visualization input files, if desired.
rdvizfromfile	boolean (default: false)	Specifies if the user would like Reaction-Diffusion visualization information to be read from files. Again, dovisualization must be set to true.
rddatafileprefix	string	The prefix for reaction-diffusion visualization input files, if desired.
temperatureoutputfreq	integer (default: 1)	Specifies the frequency in outputfreq of the temperature trajectory file to be written, if desired.
temperaturefile	string	Contains the full pathname of the temperature trajectory file, if so desired.
dotemperaturelfile	boolean (default: false)	Specifies if the user would like a temperature trajectory file to be written.
concentrationfile	string	Contains the full pathname of the concentration output file, if desired.
doconcentrationfile	boolean (default: false)	Specifies if the user would like a concentration file to be written.
latticefile	string	Contains the full pathname of the lattice file, if desired. This file can be used as visualization input for the Potts model for a future run.
dolatticefile	boolean (default: false)	Specifies if the user would like a lattice file to be written.
logfile	string	Contains the full pathname of the log file to be written, if desired.
dologfile	boolean (default: false)	Specifies if the user would like a log file to be written.
finpif	string	Contains the full pathname of the final Potts Initial File to be written, if desired. Note: right now this will write the initial information (not final), though it is written at the end of the simulation. This is because having the initial information is useful for a uniform cell distribution in knowing where all of the cells are located. Once CompuCell expands to allowing the user to set cell positions in the lattice, this will change.
dofinpif	boolean (default: false)	Specifies if the user would like a final Potts Initial File to be written. See above.
fincpf	string	Contains the full pathname of the final Cell Parameter File to be written, if desired.
dofincpf	boolean (default: false)	Specifies if the user would like an final Cell Parameter File to be written.
allenergiesfile	string	Contains the full pathname of the energies file to be written, again if desired.

Keyword	Type	Description
doallenergiesfile	boolean (default: false)	Specifies if the user would like an energies file to be written, with all energies (chemotaxis, surface, volume, interaction, total), plus temperature in one file.
splitenergiesfile	string	Contains the full pathname of the split energies file prefix, if split energy files are desired. In this case, there will be different files generated for different energies attached with the appropriate flag (chemotaxis = .chemotaxis.dat, surface = .surface.dat, volume = .volume.dat, interaction = .interaction.dat, total = .total.dat, temperature = .temperature.dat).
dosplitenergiesfile	boolean (default: false)	Specifies if the user desires split energy output files.
restartfile	string	Contains the file prefix of the restart output files, if the user desires them. This option allows a configuration file (prefix.conf), a Potts Initial File (prefix.pif), a Visualization file (prefix.viz - only if visualization was desired) and a Cell Parameter File (prefix.cpf) to be printed out on several occasions throughout the simulation, the exact frequency specified by restartfreq (below). This way the user can view more closely how the molecule changes throughout the simulation, rather than simply viewing initial and final data.
restartfreq	integer (default: 24000)	Specifies the frequency in timesteps for restart files to be written, if desired. This applies only if restart files are desired, but must be specified if they are.
dorestartfiles	boolean (default: false)	Specifies if the user desires restart files to be written.
pauseafterrestart	boolean (default: false)	Specifies if the user would like to pause the simulation everytime restart files are written, to have time to view them before continuing the simulation from where it left off.
outputfreq	integer	Specifies the frequency in timesteps for the writing of energy data to the console.
pottsnumsteps	integer	Specifies the number of steps for the Potts simulation to run.
pottsfirststep	integer (default: 0)	Specifies the number of the initial timestep for the Potts simulation.
pottsdim	coordinates	Specifies the size in pixels for each dimension of the Potts simulation.
pottslatticedim	integer (default: 2)	Specifies 2 or 3 dimensions for the Potts simulation. Right now this is automatically set to 2.
pottslatticesize	coordinates	Specifies the number of divisions in each dimension of the Potts simulation.

Keyword	Type	Description
pottswindowsize	coordinates	The size of the Potts simulation window, in pixels.
pottswindowincrement	coordinates	The amount (in pixels) to move the active zone window up at each increment point.
pottswindowincrementbottom	coordinates	This should be used if the bottom and top of the active zone should increment at different rates. This particular parameter is the amount to move the bottom of the active zone window in pixels.
pottswindowincrementtop	coordinates	The amount in pixels to move the top of the active zone, if it should be different from the bottom.
pottsstepsperwindowmovement	integer	The frequency in Potts steps to move the active zone window.
numneighborsused	integer (default: 4)	For Extended Potts, specifies the number of neighbors to take into account.
numrdsections	integer	Specifies the size of the histogram.
rdupdatefreq	integer	Specifies the frequency at which the reaction-diffusion concentration window should be updated.
domainfill	real (default: 0.99) The percentage of the y-range of the lattice to be filled by cells.	

- `Xfile` defines the path and the filename of the output `X`, where `doXfile` is the flag to turn on/off the output. The default of `doXfile` is true, if `Xfile` is defined, otherwise false.

2.3 Required Parameters

As mentioned earlier, the command line must contain either the full pathname of the configuration file or an input file prefix. These are the only restrictions specifically applied to the command line.

The following parameters MUST be specified on either the command line or in the configuration file specified on the command line (you may view Chapter 3 for a list of supported CompuCell files and their formats):

- One of either a configuration file, or an input file prefix.
- One of a potts initial file, an input file prefix, or a uniform cell distribution set to true.
- One of either a visualization file, an input file prefix, or dovisualization set to false.
- One of either a cell parameter file or an input file prefix.
- An initial rd file for concentrations.
- If vizfromfile is set to true, datafileprefix must be specified.
- If rdvizfromfile is set to true, rddatafileprefix must be specified.
- If the user specifies any output files that they want written, an output frequency must be specified.
- If the user specifies that restart files are desired, both a restart frequency and restart file prefix must be specified.
- If the user specifies any specific output files that should be written, they must specify a filename.
- The number of steps for the Potts simulation.
- If pauseafterrestart is true, restart files must have been specified. The same criteria for restart files applies.

Chapter 3

Input and Output File Types and Formats

3.1 Supported File Formats

3.1.1 PIF: Potts Initial File

COMPUCELL supports initial, intermediate and final cell position files in PIF format. This format is very simple. It consists of rows of information containing a cell type, a cell spin, and the coordinates (in pixels) at which this cell type of this spin is located:

[type1] [spin1] [x1](-[x2]) [y1](-[y2]) [z1](-[z2])
[type2] [spin2] [x1](-[x2]) [y1](-[y2]) [z1](-[z2])
[type3] [spin3] [x1](-[x2]) [y1](-[y2]) [z1](-[z2])
.
.
.
.

The types are all strings, the spins and coordinates are all integers. Note that you can specify a range of coordinates in any dimension using the '-' operator followed by another integer, or you may just specify one. Also, the z coordinates are optional if in the configuration file 2 dimensions was specified for the Potts simulation. Once again please note that this is not essential at the moment because a uniform distribution of cells is implicitly used.

3.1.2 Viz: Visualization File

For visualization information to be supplied to VTK for each specific cell spin, COMPUCELL uses the .viz input file. Note that this file is only necessary if visualization is desired. The format consists of one or more entries of the following:

ITEM
VALUE [spin1] ([spin2])
OPACITY [opacity]
SPECULAR [specular]
SPECPOWER [specpower]
COLOR [red] [green] [blue]
CUT [cut]

All values are of type integer except SPECULAR which is of type Real. Note again here that a range of spins can be specified with spin1 and spin2, if all spins between spin1 and spin2 have the same visualization properties.

3.1.3 CPF: Cell Parameter File

COMPUCELL supports Extended Potts Model Hamiltonian Terms and values in CPF format. This file follows the following format:

CellModel [modelname]{

[variable1] ¡([parametertype1])¿ [variable type] ([default value]) ([condition])
[variable2] ....
}

CellTypes {

[type1],
[type2],
.....

Type [type] : Model [model] {
[variable1] ([parameter1]) ([value1])
[variable2] ([parameter1]) ([value2])
[variable3] ([parameter1]) ([value3])
}

This file enables the user to declare a given number of cell models and cell types. Cell types follow a specific cell model, and that is declared in the third section. Every type specified in the third section must be declared in the second section, and any model specified in the third section must be declared in the first section. Each model has its own set of variables that the cell types automatically inherit when they are declared to be of a given model. Note that in the third section if a parameter is specified it must be of the correct type corresponding to the model declaration (first section), and all values must obey the condition specified. If no value is specified in the third section for a variable declared to be a part of that particular model, the default value is used (if no default is specified the program errors).

The format for a CPF condition consists of one of the traditional condition symbols ($>=$, $>$, $<$, $<=$, $==$, $!=$) followed immediately (no whitespace) by a value. An example will be given in the next chapter, *Examples of* COMPUCELL *input files*.

Valid CPF types for both variables and parameters include: int/integer, double/real, str/string, type/celltype, none, or bool.

Important: as of now, there can only be two types used, R0 and med, the first being the type of a cell and the second being the surrounding medium. ALL CELLS NOW HAVE THE SAME TYPE. This will be changed in the future though.

3.1.4 COMPUCELL Chemical Concentration File

This file specifies the activator concentration for each window of the active zone throughout a simulation. The current representation consists of repeated entries of the following format:
[starting row for active window 1]
[ending row for active window 1]

[activator concentration for pixel (starting row, 0)]
[activator concentration for pixel (starting row, 1)]
......
[activator concentration for pixel (ending row, lattice size)]

As can be seen, this format is not preferable. If the simulation window is large and there are many window increments (meaning many active zones), this file could be huge. Shrinking the potential size of this file is believed be a future implementation.

An initial file of this format is required, but the file that COMPUCELL can output upon request follows this format but will only output this information for the current active zone.

3.1.5 Temperature Trajectory Files

COMPUCELL is able to write temperature trajectory files, consisting of multiple rows with the following simple format:

[step number] [temperature at this step]

This file will be written every `outputfreq` timesteps, with this values specified in the configuration file.

3.1.6 COMPUCELL All Energy Output File

A COMPUCELL all energy file includes all different types of energy in the same file. Each row in the all energies file represents one timestep. There may be several rows of data depending on the output frequency specified in the configuration file, since that specifies the number of timesteps between data writes.

The format of one row is as follows, with each entry separated by whitespace:

<step><total energy><chemotaxis energy><surface energy><volume energy><interaction energy><temperature>

3.1.7 COMPUCELL Split Energy Output File

If COMPUCELL split energy files are desired, ten separate files will be generated, one for each different type of energy shown above in the COMPUCELL All Energy Output File. Each file will have the same format, composed of rows of the following depending on the output frequency:

<time><value>

where <value>is the value of the appropriate energy parameter at the given time.

3.1.8 Lattice File

This file provides a three-dimensional scene description of the simulation. It can be read by COMPUCELL as an input to the visualization framework and can generate the appropriate cell structure in the visualization window by reading it. That is the main purpose of enabling COMPUCELL to write this type of file.

3.1.9 Log File

This file contains information that is output as the simulation runs - values of certain variables at certain points in the execution, where in the execution the program currently is, etc.

3.2 Input Files

Input files are specified on the command line and/or the configuration file. The following guidelines need to be followed:

Potts Initial File:	PIF format
Visualization File:	Viz format
Cell Parameter File:	CPF format
Chemical Concentration File:	COMPUCELL Concentration File format

3.3 Restart Output Files

If the user desires restart files to be printed throughout the simulation, here is what will be printed every certain number of timesteps, given by the restart frequency:

Potts Initial File:	PIF format - filename <restartprefix.pif>
Visualization File:	Viz format - filename <restartprefix.viz>
Cell Paramter File:	CPF format - filename <restartprefix.cpf>
Configuration File:	COMPUCELL configuration file format - filename <restartprefix.conf>

3.4 User-Specified Output Files

There are several supported types for output files, but the advantage here is that the user can have the same information printed in multiple formats if multiple formats are supported for this information, unlike the input files where there could only be one file for each different initial information type. the following are the supported formats:

Final Potts Initial File:	PIF format
Final Cell Parameter File:	CPF format
Lattice File:	COMPUCELL lattice file format
Log File:	text format
Temperature Trajectory File:	COMPUCELL temperature file format
All/Split Energies File:	COMPUCELL energies file format
Concentration File:	COMPUCELL concentration file format (one entry)

Chapter 4

Examples of COMPUCELL Input Files

4.1 An Example Potts Initial File

Following is an example of the fi rst few lines of a Potts Initial File. This fi le was a fi nal PIF generated when simulating chicken limb growth.

There is only one cell type in this simulation, two including the surrounding medium (typically the medium is declared as a cell type in the CPF, so it is counted as one when discussing general topics). Each cell has a different spin, the medium is counted as one "cell" with a spin of 0. This is a two-dimensional system so all z-coordinates are 0. Columns 3 and 4 represent ranges of x and y-coordinates in pixels in which the cell with the type (column 1) and spin (column 2) occupies. For example, the cell with a spin of 1 and type R0 occupies pixels (1,1,0), (1,2,0), (1,3,0)....(2,1,0), (2,2,0), ...(7,7,0). It is thus a square cell with a side length of 7 pixels. The cell with a spin of 2 is also a square cell but is 3 pixels above the cell with spin 1 in the y direction. And everything else afterwards follows this same format. Note that the medium is placed where the cells are not, be it along the edges (the four beginning entries) or between cells.

```
med 0 0 0-136 0
med 0 1-50 136 0
med 0 50 0-135 0
med 0 1-49 0 0
RO 1 1-7 1-7 0
med 0 1-7 8-9 0
RO 2 1-7 10-16 0
med 0 1-7 17-18 0
RO 3 1-7 19-25 0
med 0 1-7 26-27 0
RO 4 1-7 28-34 0
med 0 1-7 35-36 0
RO 5 1-7 37-43 0
med 0 1-7 44-45 0
RO 6 1-7 46-52 0
med 0 1-7 53-54 0
RO 7 1-7 55-61 0
med 0 1-7 62-63 0
RO 8 1-7 64-70 0
med 0 1-7 71-72 0
RO 9 1-7 73-79 0
med 0 1-7 80-81 0
RO 10 1-7 82-88 0
med 0 1-7 89-90 0
```

4.2 A Sample Visualization File

Following is a sample Viz input file. Note that the number of entries in this file will correspond exactly to the number of cells in the simulation. Each cell will have a unique spin (specified as VALUE in this file). The cell with a spin of 1 has an opacity of 1, specular of .3, specpower of 10, is purely red and has no cut. The cells with spins 2 through 4 (note the range specification) all have the same characteristics, they are blue with opacity of 1, specular of .3 and specpower of 10, and again no cut. The rest of the entries all follow this same format.

```
ITEM
VALUE 1
OPACITY 1
SPECULAR .3
SPECPOWER 10
COLOR 1 0 0
CUT 0

ITEM
VALUE 2 4
OPACITY 1
SPECULAR .3
SPECPOWER 10
COLOR 0 0 1
CUT 0

ITEM
VALUE 5
OPACITY 1
SPECULAR .3
SPECPOWER 10
COLOR 1 0 0
CUT 0

ITEM
VALUE 6
OPACITY 1
SPECULAR .5
SPECPOWER 10
COLOR 1 0 0
CUT 0

ITEM
VALUE 7
OPACITY 1
SPECULAR .3
SPECPOWER 10
COLOR 0 1 0
CUT 0
```

4.3 Example Cell Parameter File

Here is the exact Cell Parameter File that was used in our simulation of the chicken limb growth. As can be seen in the file, there is one model (named Chicklimb) and two cell types, R0 and the medium. R0 and the medium both follow the Chicklimb model.

The Chicklimb model has six different Hamiltonian variables declared. Each of these variables are needed for the extended Potts model. Variable J (in the Potts model this represents the surface energy between two cell types) accepts one parameter of type 'type', which means celltype. It is of type double, with a default value of 0 and has the additional requirement that it must be greater than or equal to 0.

Note that in both cell type 'R0' and cell type 'med', J has been defined accepting one available cell type (RO or med) and has been given a value that satisfies the type of double and the condition that it must be greater than 0. The parameter 'lambdavolume' of model Chicklimb accepts no arguments and also has a type of double, and the declarations of R0 and med correspond correctly. Similar cases for all other variables.

```
CellModel Chicklimb{
        J <type> double 0 >=0
        lambdavolume <> double
        targetvolume <> int
        lambdasurface <> double
        targetsurface <> int
        muchemotax <> double
}

CellTypes {

R0,
med

}


Type R0 : Model Chicklimb
{
        J R0 0.5
        J med 4.0
        lambdavolume 3.0
        targetvolume 144
        lambdasurface 0.0
        targetsurface 1
        muchemotax 70.0
}

Type med : Model Chicklimb
{
        J med 0
        J R0 4.0
        lambdavolume 0.0
        targetvolume 0
        lambdasurface 0
        targetsurface 0
}
```

Chapter 5

Running COMPUCELL

You may run COMPUCELL as specified in the chapter on *Getting Started* with the appropriate command line arguments. If you specified that you wanted visualization, a simulation window will appear. After some time, this window will be split into initial Potts and Reaction-Diffusion windows on the left and right, respectively. The Potts window will demonstrate cell clustering and sorting in accordance with the extended Potts model, and the Reaction-Diffusion window will show a color field demonstrating activator concentration level in a ROYGBIV form (blue is high, red is low). This way you may view how each window changes over time. Accompaning this will be a box consisting of five buttons: "Next" (which will advance the simulation one step), "Run 10" (which will advance the simulation ten steps), "Run" (which will run the entire simulation), "Run and Exit" (which will run the entire simulation, finalize output and automatically exit), 'and "Exit" (which will finalize output and exit). Click on any of these initially and at any point at which the simulation has paused, be it for restart files if that option was set to true or if the simulation has run the appropriate amount of steps that was specified by the button last clicked.

Next a display will come up in the console showing many initial values, and the program will immediately start running for the appropriate number of steps. A line of output will be displayed at every step, and these lines are of the format:

```
Step : <step #>, TE : <total energy> [units], V : <total volume> [units]
```

Chapter 6

Availability and Installation of CompuCell

6.1 How to Download CompuCell

The download site for CompuCell is a link from the page for the Laboratory for Computational Life Sciences at the University of Notre Dame: `http://www.nd.edu/˜lcls/compucell`. Executable code is available online, and source code will be available upon request. An agreement to a Non-Exclusive, Non-Commercial Use License is required (this exact license can be found in the appendix of this user guide, appendix B) for download.

6.2 Platforms that CompuCell is Able to Run On

CompuCell executables are available for the following three platforms:

1. Sun OS 5.8
2. Linux 2.4

6.3 Compiling CompuCell

Because executables are available online, compiling CompuCell should not be necessary unless the user would like to change features. Here are the steps for generating the appropriate makefi les on different platforms:

1. Change to the main **CompuCell1.6** directory.
2. View the `README` fi le for specifi c directions on how to compile CompuCell correctly on the apporpriate platform.

Once the makefi les have been generated, the user types "`make`" from the protomol directory. A "`make clean`" will clear all object fi les and executables recursively throughout the directory tree, if necessary.

After everything has compiled successfully, the user may run CompuCell from the `CompuCell1.6/applications/compucell-app` directory as described in Chapter 2.

6.4 Documentation

Available documentation for CompuCell can be downloaded from:

`http://www.nd.edu/˜lcls/compucell`

Chapter 7

The Future of COMPUCELL

Further expansion of COMPUCELL is under way now. In the future we expect to add new algorithms and hamiltonians to the back end. It is also desirable to have COMPUCELL take advantage of parallel computing considering the complexity and size of systems that it will likely be expected to handle - that is a future implementation.

In addition, we would like to increase user flexibility by implementing factory design methods in the front end that will allow the user to declare his or her own cell types without the low-level code needing to know these types. This way the user will be able as many types as desired and name them anything. Also, we would like the program to be able to run in 3 dimensions and for the user to be able to specify the initial locations of cells, their types and spins (an initial PIF), since at the moment a uniform distribution of cells in 2-dimensions is assumed.

We also hope to include a graphical user interface in a future release of COMPUCELL , which will help in the area of ease of use. Work is also being done in making the software compatible to Windows, and this should be hopefully be released in the near future.

Appendix A

Background

A.1 Extended Potts Model [3]

Reference: "Simulation of Biological Cell Sorting Using a Two-Dimensional Extended Potts Model", Francois Graner and James A. Glazier, Physical Review Letters Volume 69, Number 13 - September 28, 1992.

Cell sorting is involved with the reaggregation of dissociated and randomly mixed embryonic cells - in fact it is a very key step. Though it involves rearrangement of the cell positions in space, the cells do not divide or differentiate. The results of cell sorting are determined by differences in surface energy between neighboring cells or between a cell and the surrounding medium.

A.1.1 Compared to Regular Potts Model

The regular Potts model was useful in simulating the growth of cells that was driven by surface energy differences. It assumes one cell type with one single surface energy. The extended Potts model assumes at least two cell types with different surface energies. Also, the extended version is oriented towards cell *sorting*, which deals with the movement of cells, as opposed to cell *growth*.

The Regular Potts model, more commonly known as the *large-Q* Potts model, defi nes cells by spin - with each cell having its own spin value between 1 and N for N cells. The following Hamiltonian is solved for energy:

$$H_{Potts} = \sum_{(i,j),(i',j')neighbors} 1 - \delta_{\sigma(i,j),\sigma(i',j')} \tag{A.1}$$

where the neighbors may be of any desired range on the lattice, which can be square or hexagonal. A random point in the lattice is selected at each step and with Monte Carlo probability its spin gets flipped from σ to σ ', depending on the change in energy ΔH that would result and the current temperature T. The following probability equation is used to determine whether or not to flip the spin:

$$P(\sigma(i,j) \rightarrow \sigma'(i,j)) = \begin{cases} \{exp(-\Delta H/kt), & \Delta H > 0, T > 0 \\ 1, & \Delta H \leq 0, T > 0 \\ 0, & \Delta H > 0, T = 0 \\ 0.5, & \Delta H = 0, T = 0 \\ 1, & \Delta H < 0, T = 0 \end{cases} \tag{A.2}$$

The idea with cell sorting is that the overall cell pattern's energy is reduced by cell motion causes contact energy differences between cells with different types. Thus in the Extended Potts Model we are introduced to this new concept of cell type, and denote it by a τ symbol. The concept of cell spin still exists and is the same as for large-Q potts, but cells of different spins may have the same type. Cells of different type will have different Hamiltonian terms. The modifi ed Hamiltonian equation looks like this:

$$H_{sort} = \sum_{(i,j),(i',j')neighbors} J[\tau(\sigma(i,j)), \tau(\sigma(i',j'))][1 - \delta_{\sigma(i,j),\sigma(i',j')}] + \lambda \sum_{spintypes,\sigma} [a(\sigma) - A_{\tau(\sigma)}]^2 \theta(A_{\tau(\sigma)}) \tag{A.3}$$

Where:

$\tau(\sigma)$	=	The type associated with the cell with spin σ
$J(\tau,\tau')$	=	The surface energy between cells with type τ and τ'
λ	=	Lagrange multiplier - indicates the strength of the area constraint.
$a(\sigma)$	=	Area of the cell with spin σ
A_τ	=	Target area of cells of type τ
$\theta(x)$	=	Area constraint: 0 if x $<$ 0, 1 if x $>$ 0

For the surrounding medium, we defi ne it as one cell typically with a spin of 0, and its own type, energies and volume, with target area negative. A requirement is set that a point in the lattice must flip to a spin of a neighboring cell. Potts-model dynamics are used with Monte Carlo steps to simulate the cell sorting.

A.2 Schnackenberg Reaction-Diffusion [1]

Schnackenberg Reaction-Diffusion equations are solved to obtain an activator concentration fi eld throughout the simulation. The dimensionless Schnakenberg equation is:

$$\frac{\delta u}{\delta t} = \gamma(a - u + u^2 v) + \nabla^2 u = \gamma f(u,v) + \nabla^2 u \tag{A.4}$$

$$\frac{\delta v}{\delta t} = \gamma(b - u^2 v) + d\nabla^2 u = \gamma g(u,v) + d\nabla^2 u \tag{A.5}$$

Appendix B

COMPUCELL Biocomplexity Software - Non-Exclusive, Non-Commercial Use License

B.1 Introduction

Upon execution of this Agreement by the party identifi ed below ("Licensee"), The Board of Trustees of the University of Notre Dame ("Notre Dame"), on behalf of Laboratory for Computational Life Sciences ("LCLS") will provide the COMPUCELL molecular dynamics software ("COMPUCELL ") in Executable Code and/or Source Code form ("Software") to Licensee, subject to the following terms and conditions. For purposes of this Agreement, Executable Code is the compiled code, which is ready to run on Licensee's computer. Source code consists of a set of fi les which contain the actual program commands that are compiled to form the Executable Code.

B.2 Conditions and Regulations

1. The Software is intellectual property owned by Notre Dame, and all right, title and interest, including copyright, remain with Notre Dame. Notre Dame grants, and Licensee hereby accepts, a restricted, non-exclusive, non-transferable license to use the Software for academic, research and internal business purposes only, e.g. not for commercial use (see Clause 7 below), without a fee.

2. Licensee may, at its own expense, create and freely distribute complimentary works that interoperate with the Software, directing others to the LCLS server to license and obtain the Software itself. Licensee may, at its own expense, modify the Software to make derivative works. Except as explicitly provided below, this License shall apply to any derivative work as it does to the original Software distributed by Notre Dame. Any derivative work should be clearly marked and renamed to notify users that it is a modifi ed version and not the original Software distributed by Notre Dame. Licensee agrees to reproduce the copyright notice and other proprietary markings on any derivative work and to include in the documentation of such work the acknowledgement:

"This software includes code developed by the Laboratory for Computational Life Sciences in the Department of Computer Science and Engineering at the University of Notre Dame."

Redistribution and use of COMPUCELL in source and executable formats, with or without modifi cation, are permitted provided that the following conditions are met:

a. Redistributions of source code must retain the above copyright notice, this list of conditions, and a citation of

the following:

”Thierry Matthey, Jesus A. Izaguirre, CompuCell: A Molecular Dynamics Framework with Incremental Parallelization, Proc. 10th SIAM Conference on Parallel Processing for Scientifi c Computing, March 2001.” Available electronically at `http://www.nd.edu/~lcls/lcls_publications.html`

b. Redistributions of executable code must reproduce the above copyright notice, this list of conditions, and a citation to the above reference in the documentation and/or other materials provided with the documentation.

3. Except as expressly set forth in this Agreement, THIS SOFTWARE IS PROVIDED ”AS IS” AND NOTRE DAME MAKES NO REPRESENTATIONS AND EXTENDS NO WARRANTIES OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO WARRANTIES OR MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, OR THAT THE USE OF THE SOFTWARE WILL NOT INFRINGE ANY PATENT, TRADEMARK, OR OTHER RIGHTS. LICENSEE ASSUMES THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE SOFTWARE AND/OR ASSOCIATED MATERIALS. LICENSEE AGREES THAT UNIVERSITY SHALL NOT BE HELD LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, OR INCIDENTAL DAMAGES WITH RESPECT TO ANY CLAIM BY LICENSEE OR ANY THIRD PARTY ON ACCOUNT OF OR ARISING FROM THIS AGREEMENT OR USE OF THE SOFTWARE AND/OR ASSOCIATED MATERIALS, INCLUDING BUT NOT LIMITED TO PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION; HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE), EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

4. Licensee understands the Software is proprietary to Notre Dame. Licensee agrees to take all reasonable steps to insure that the Software is protected and secured from unauthorized disclosure, use, or release and will treat it with at least the same level of care as Licensee would use to protect and secure its own proprietary computer programs and/or information, but using no less than a reasonable standard of care. Licensee agrees to provide the Software only to any other person or entity who has registered with Notre Dame. If licensee is not registering as an individual but as an institution or corporation each member of the institution or corporation who has access to or uses Software must agree to and abide by the terms of this license. If Licensee becomes aware of any unauthorized licensing, copying or use of the Software, Licensee shall promptly notify Notre Dame in writing. Licensee expressly agrees to use the Software only in the manner and for the specifi c uses authorized in this Agreement.

5. By using or copying this Software, Licensee agrees to abide by the copyright law and all other applicable laws of the U.S. including, but not limited to, export control laws and the terms of this license. Notre Dame shall have the right to terminate this license immediately by written notice upon Licensee’s breach of, or non-compliance with, any terms of the license. Licensee may be held legally responsible for any copyright infringement that is caused or encouraged by its failure to abide by the terms of this license. Upon termination, Licensee agrees to destroy all copies of the Software in its possession and to verify such destruction in writing.

6. The user agrees that any reports or published results obtained with the Software will acknowledge its use by the appropriate citation as follows:

"COMPUCELL was developed by the Laboratory for Computational Life Sciences in the Department of Computer Science and Engineering at the University of Notre Dame."

Any published work which utilizes COMPUCELL shall include the previous reference.

Electronic documents will include a direct link to the official COMPUCELL page at
`http://www.nd.edu/~lcls/compucell`

7. Commercial use of the Software, or derivative works based thereon, REQUIRES A COMMERCIAL LICENSE. Should Licensee wish to make commercial use of the Software, Licensee will contact Notre Dame to negotiate an appropriate license for such use. Commercial use includes: (1) integration of all or part of the Software into a product for sale, lease or license by or on behalf of Licensee to third parties, or (2) distribution of the Software to third parties that need it to commercialize product sold or licensed by or on behalf of Licensee.

8. COMPUCELL is being distributed as a research and teaching tool and as such, LCLS encourages contributions from users of the code that might, at the sole discretion of Notre Dame, be used or incorporated to make the basic operating framework of the Software a more stable, flexible, and/or useful product. Licensees who contribute their code to become an internal portion of the Software agree that such code may be distributed by Notre Dame under the terms of this License and may be required to sign an "Agreement Regarding Contributory Code for COMPUCELL Software" before Notre Dame can accept it (contact *compucell@cse.nd.edu* for a copy).

UNDERSTOOD AND AGREED.

B.3 Contact Information

The best contact path for licensing issues is by e-mail to *compucell@cse.nd.edu* or send correspondence to:

COMPUCELL Team
c/o Prof. Jesús A. Izaguirre
Laboratory for Computational Life Sciences
Department of Computer Science and Engineering
University of Notre Dame
384 Fitzpatrick Hall of Engineering
Notre Dame, Indiana 46556 USA

Bibliography

[1] Laboratory for Computational Life Sciences. `http://www.nd.edu/~lcls/compucell`, 2002. CompuCell home page.

[2] Interdisciplinary Center for the Study of Biocomplexity. `http://www.nd.edu/~icsb`, 2002. Homepage for the ICSB at the University of Notre Dame.

[3] James A. Glazier and Francois Graner. Simulation of biological cell sorting using a two-dimenional extended potts model. *Physical Review Letters*, 69:2013–2016, September 1992.