Dynamic Load Balancing and Robustness In An Amorphous Self−Regulating Sequential Producer−Consumer Process

Jonathan Ari Goler
jagoler@mit.edu

ABSTRACT. The goal of this project is to show that a general system of smart cells can be designed to cooperate to perform an arbitrarily complex, sequential, producer−consumer process that automatically performs adaptive load balancing. In more plain terms, a bunch of mobile agents can be programmed so that they can automatically turn themselves into an assembly line. That assembly line can repair itself if some parts fail, without central control or human interference. To demonstrate this goal, we shall build chemical factories (in simulation) using smart cells. It is difficult, through normal human−controlled chemical means, to reliably, precisely and efficiently construct molecules such as hydrocarbons, proteins and other small molecules. In this system, the product of one stage of reactions is needed to execute the second stage, but the two cannot occur at the same cell. While the ability to control such reactions within cells is not something that we can do in the wet−lab yet, it is nevertheless likely to be something that is doable in the next several years. The cells in this system will have the ability to signal each other by sending chemicals out into their environment. They will also be able to monitor and modify their own state. We would like to see if these smart cells would develop into sensible morphologies and whether or not those morphologies would restore themselves should cells die.

1.1. Introduction

One of the main differences in approaching amorphous computing from the centralized control models of computing is the lack of coordination between individual cells in the system. While at first it seems that the lack of control might be detrimental to programming and performance, it is in fact, an advantage for solving certain problems when trying to build systems that can automatically recover themselves. In the case of a sequential process, where each stage of the process requires the product of a previous step, or an initial primer, the ability for cells to detect, in essence, what needs to be done, and then convert to the appropriate process architecture enables efficient and robust processing.

The process of chemical factories is a natural choice to demonstrate this ability, because we can understand more intuitively that enzymes are specialized to particular processes, and require complex processes to produce, thus, specialization is preferred. At the same time, regulation of these systems can be thought of as flipping switches in the DNA. While the wet lab technology is not this advanced yet, it will be in the near future.

Chemistry has been very successful at discovering how to synthesize very small molecules and very large molecules that do not require an exact size. However, it is in the middle of this spectrum: requiring very specific molecular construction for molecules ranging from 100 Daltons to tens of K Daltons. It is this range that living cells are really talented, and it is in this area that we hope to find answers to problems in biology that might require creating and

synthesizing specialized proteins. Such problems include solutions that might be customized proteins or polysaccharides. New drugs such as T−20, an HIV fusion inhibitor, are synthetic peptides. We cannot yet harness cells to produce it for us. Thus the chemical process to manufacture the drug is highly wasteful, risky and unreliable, increasing the cost of already expensive drugs.

We need to figure out how to harness the ability of cells that permits them to do complicated chemical reactions and do them in an orderly fashion. Because of the nature of many chemical processes, it is hard or impossible for certain combinations of reactions to occur in the same place, due to pH differences or other effects. Just as plants develop layers that perform different chemical functions in the energy systems of the plant, we require different reactions to take place in different cells. Another useful analogy is to think of each cell as a worker in a factory. If they are focused on doing exactly one operation, they can operate much faster than they could if they were doing many different tasks.

2.

The central idea that is necessary to understand how such cells operate is the idea of Gene Networks. Gene networks allow us to formalize a method for cells to regulate themselves and communicate with others. In order for cells to act efficiently as a group, such as "wolf−packs" of cells that specialize tasks in a group of motile cells in order to more effectively kill and feed on prey [4]. Such cooperative behavior is inspiration for formalizing the idea of cellular communication. Ron Weiss and Tom Knight discuss using digital intracellular genetic circuits to combine the abilities of digital engineering and the natural abilities of cells to engineer multicellular structures [3].

There are many important distinctions between biological components and silicon components. One of the most important distinctions is the way "wires" work. In silicon, a wire connects the output of one or more devices to the input of other. Such wires conduct electrons, and they can be engineered so that there is little cross talk (interference from other wires) and a very small delay from when a voltage appears on the source until it appears on the sink. In biology, except for highly specialized cells designed for signal propagation, signals are chemicals conducted through an aqueous medium. These signals produce additional variability in their transmission time, and reliability. They are also far more analog in nature than a wire voltage, as much more variability results from the transmission. However, cells possess the ability to distinguish between levels of a chemical signals or gradient. If they are motile, cells could follow the gradient level, or travel tangentially to it. In addition, cells have a propensity to die, either due to parts wearing out or because of a toxin; thus, our system must be cell−death tolerant.

It is important to note, that while HLSim [6] was considered as a simulator, it was rejected because of the need to support fully mobile agents. While state can be moved around in HLSim to different cells, thus the cells are merely pixels, it is an awkward way of programming.

In order to explore the possibility of engineering cell systems, it is necessary to simulate their behavior to cut the necessary wet−lab experiments. This project seeks to develop a simulator for engineered multi−stage cellular processing, and to verify that it behaves in a sensible way.

2. Methodology

3.Capabilities of the Cells

Cells are designed to mimic their biological counterparts in many areas. Cells in our simulation have only local state: they can detect the chemistry of their current environment, and they have the ability to have local internal state. The internal state that real cells have is enabled by many mechanisms, such as phosphorylation of enzymes, methylation of DNA or histone loops. We define the base cell in Cell.java (See Appendix B) Any cell that extend the Cell class can be placed in the simulation. In this experiment, we introduce the FactoryCell, which acts to produce a 3 stage production with self−repairing ability.

Cells do not need to have much raw computation power, but only simple combinational logic as described by Ron Weiss[1] and Knight/Sussman [5], and Maimon et al [2]. In addition to the state awareness, cells have the ability to dump chemical products into the environment via a Gradient construct. Initiating things like starting points is achieved through a random number generator, again, since motion of chemicals in a cell is achieved by a mostly random walk, it is reasonable to assume that cells have the ability to generate randomness.

Finally, we give the cells the ability to move. In the default cell implementation, cells move in a random walk, keeping track of the direction they are moving and the level of a chemical of interest, until they can figure out which way they should go in response to a gradient they detect. In this manner, we simulate chemotaxis. They then move along that gradient, and periodically adjust their course. This type of movement is observed in paramecia, which move around using their flagella to move, detect the level of food present and then pick a new direction if the location is unsuitable.

4.Environment Characteristics

The environment is modeled as a solution that is capable of supporting the cells in it. Additionally, it provides a medium for gradients to diffuse. We are assuming that the gradients will be ideally distributed, so they are radially symmetrical. This is a reasonable assumption as the gradients will most likely be distributed in a somewhat circular way, but the flows of the solution would naturally perturb that configuration. Unfortunately, it is much more difficult to model that situation.

5.The Cell Program

6.

cells built for this project, FactoryCells are designed to produce a chemical that takes 3 stages. Cells begin in an uninitialized state, waiting for a leader (the first product) to appear. In the absence of a leader, they wander around, and every several cycles, throw dice and decide if they will become a leader. Eventually, one or more cells becomes a leader, and begins producing leader (first product.) Once the other cells detect the gradient, they move towards its source by essentially wiggling around to determine what direction they should travel for a while. Cells continually monitor the level of the gradient, if it reaches a threshold that they are looking for, and then they switch modes. So, if a cell in "wander mode" gets close to a producer, it changes to a mode where it consumes the first product and produces the second stage product. The cell consumes the first product, so if there is greater consumption than production, then some cells will be unable to carry out the second reaction, and will revert to wander mode, if they can't find anything, they will convert to the first stage producers with some probability. A similar procedure happens with the later stage reactions.

7.

8.3. Hypothesis

9.

This program will result in two different "good" topologies that may be combined. In addition, these cells, if killed in groups, or randomly, will restore similar topologies and operate properly.

The two topologies that are predicted are the bulls eye topology and a linear layered topology. A combination of these topologies can be seen forming in Figure 1.

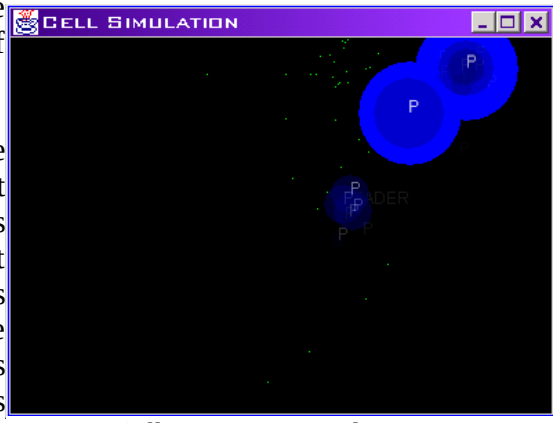


Figure 1. Cells running a staged−processing program forming a combination of linear and bull's eye topology

These topologies are "good" because they efficiently use the available resources, as well as the gradient transport phenomenon. Thus, as one product is created, it is consumed by the next stage processor at the fastest possible rate. The bull's eyes form when the cells surrounding the leader consume the product, and more cells surround those cells. The linear arrangement is formed as several leaders output product and the cells surrounding them are consuming from more than one leader, like an assembly line.

4. Results

The cell program was executed, with complete functionality, ten times. Raw screenshots of the resulting topologies are included as Appendix B. In 10 cases, we observe bull's eye topology, and in 1 we see the linear topology, and in only 1 we observe both.

In each case, we removed cells by two methods: random selection and block destruction. The blocks were taken out of the area in which the good topology appeared. The existing cells recovered topology as follows:

Table 1: Time to recovery vs. block removed

Trial	Time to First Form	Kill	Recovery	Area Killed(pix)	# of cells	
1	30	63	111	50x50	100	
2	39	60	117	50x50	100	
3	15	81	243	75x75	100	
4	13	57	360	100x100	100	
5	15	156	DNR&	100x100	200	& The sample did not recover the topology within 600 time units.
6	18	57	87	100x100	200	
7	18	342	366	100x100	200	
8	21	87	102	100x100	300	* Tighter threshold (1/5 rather than 1/25 of the required concentration)
9*	12	60	69	100x100	300	
10*	12	66	81	100x100	300	

Time is in cycles, reported with every repaint (3 cycles)

For the images, the green dots are cells that are in motion, red cells are final stage, and they output product, indicated by the green circles. The yellow dots are the leader cells, and some of them emit the red circle, while the blue dots are the stage 1 cells, note that they do not show much product. The brightness and size of the circles indicate how much and how long they have been in the environment, respectively.

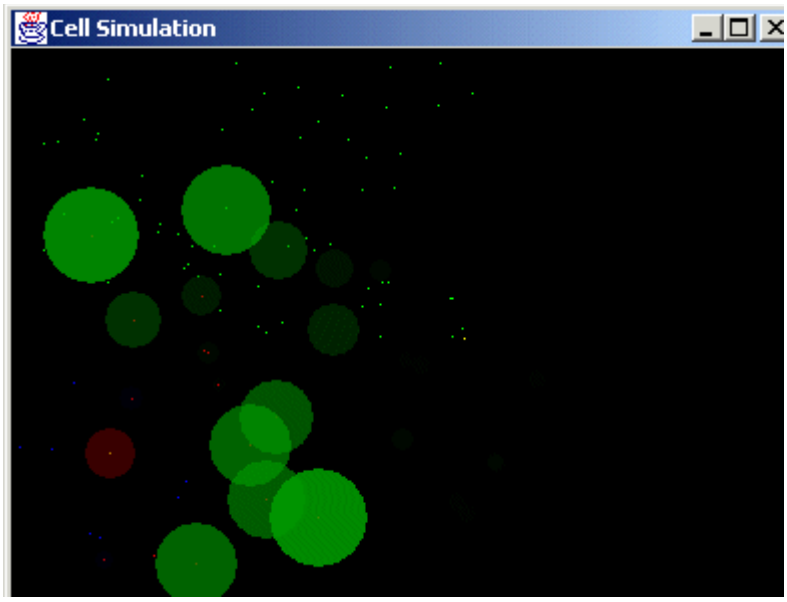


Figure 2. A Good example of a Bull's Eye.

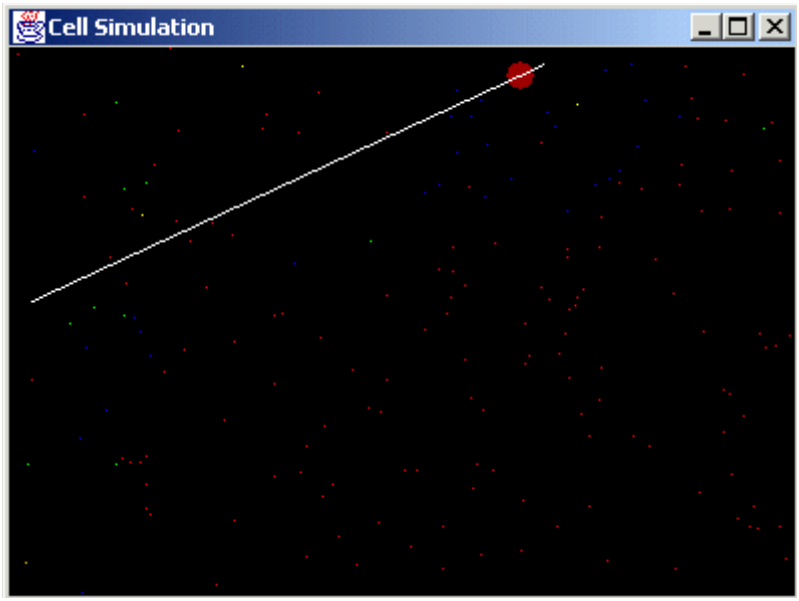


Figure 3. An example of a Linear pattern.

5. Conclusions

The cells were very robust in their ability to generate similar topologies given the 3−stage program. In all cases of running the trials, the bull's eye topology was observed. Only when the threshold for changing modes was tightened so that cells had to get very close to the source of their raw material was the linear pattern observed.

The time it took to recover the topology seems to be proportional to the area removed, thus larger areas took longer to recover. In addition, increasing the number of cells decreased the time it took to recover a good topology. Finally, tightening the threshold for stopping decreased the time to recover, presumably the cells would be quicker to pick up and move around, thus making them more mobile. The results seem consistent with the way the cells should work.

6. Future Work and Alternatives

In its present state, the simulator is only useful to approximate the behavior of mobile cells in a static culture. The method for programming the cells is currently to extend the Cell java class, which of course requires knowledge of Java, but only the very simple constructs.

One of the most important aspects of future work is to try to add different features to the simulation environment itself. Currently, one could use a specialized cell to introduce external control, but it would be convenient to permit injection of chemicals into the environment. Another important addition would be to add some notions of fluid dynamics, so that the gradients are not circular/spherical but rather are irregular due to flows of the culture medium. One could then 'stir' the medium. It might also be fruitful to add a built in notion of the cell cycle that might affect their ability to sense and operate, as well as multiply. A construct that was added from the beginning but never implemented is the notion of a field, such as electric, magnetic, or gravitational, which are often used by cells to orient themselves.

The simulator may be useful in the BioJade toolkit that is currently being designed to enable the design of gene networks for computation. The toolkit will include the ability to design digital circuits from a library of genes, and regulatory sequences, such as the ones included in the BioBricks toolkit.

References:

[1] R. Weiss. Cellular Computation and Communications using Engineered Genetic Regulatory Networks. 2001. Thesis.

[2] R. Maimon, S. Browning. GNS Biotech. Diagrammatic Notation and Computational Structure of Gene Networks.

[3] R Weiss, T. Knight. Engineered Communications for Microbial Robotics. DNA6. Sixth International Meeting on DNA based computers.

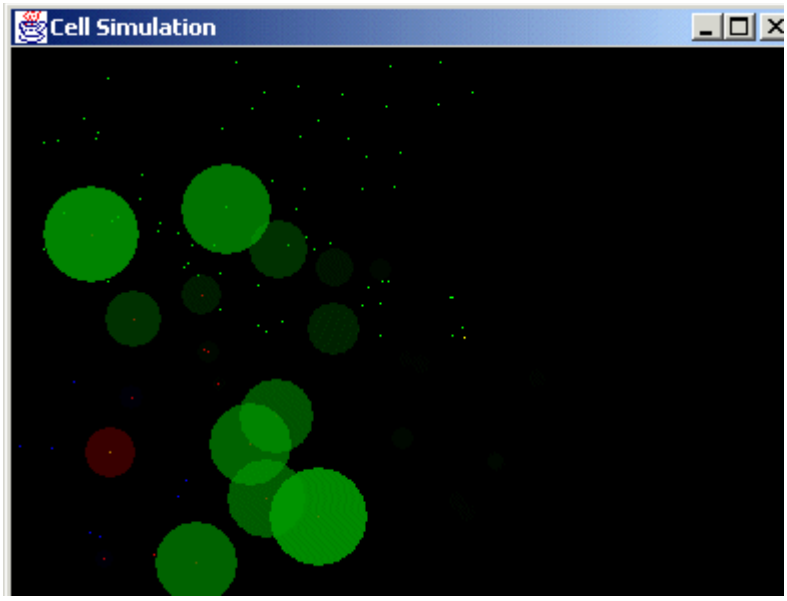
[4] M Dworkin, Cell−cell interactions in the myxobacteria. Symp. Soc. Gen. Microbiol., vol. 25, 1973, pp 135−147.

[5] Thomas F. Knight Jr. and Gerald Jay Sussman, Cellular gate technology. First International Conference on Unconventional Models of Computation. (CS Calude, J Casti, and MJ Dinneen, eds.), Springer−Verlag, 1998, pp. 257−272.

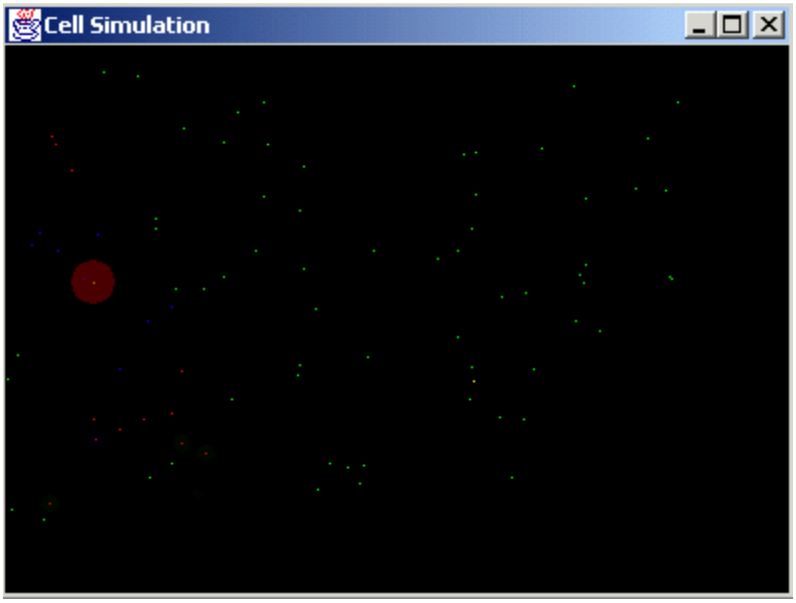
[6] S. Adams. A High Level Simulator for Gunk. http://www.swiss.ai.mit.edu/projects/amorphous/hlsim/doc/hlsim.html

Appendix A: Screenshots

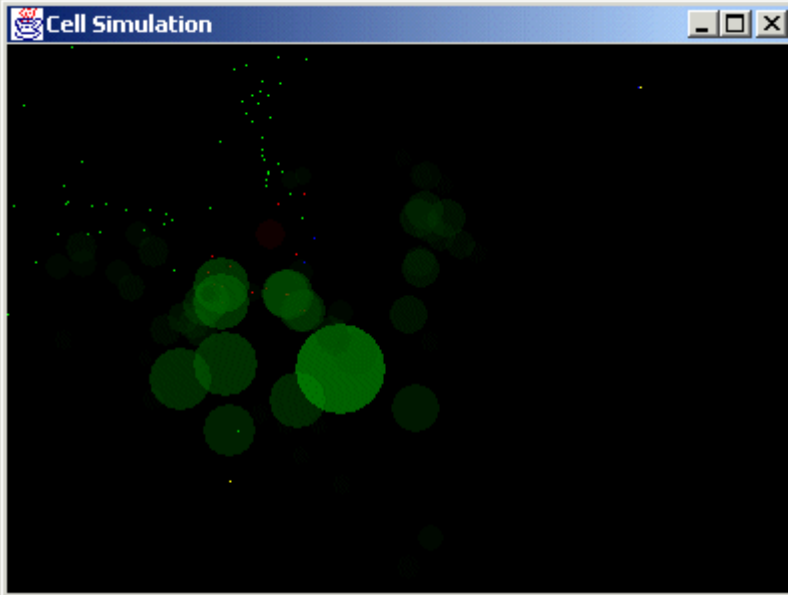
Bull's Eye with text.



A good bull’s eye.



Another good bulls eye.



A Recovered BullsEye (lower right killed)