An Abstraction of Intercellular Communication

Thomas E. Portegys

Illinois State University
Campus Box 5150
Normal, Illinois, USA 61790-5150
portegys@ilstu.edu

Abstract

Multi-cellular organisms exhibit many mechanisms of intercellular communication for control and developmental purposes. Neurotransmitters, hormones, and signals mediating such processes as apoptosis and angiogenesis are but a few of these mechanisms. An abstraction of intercellular communication is presented here to provide a basis for constructing artificial life forms and self-organizing systems. Although the investigational vehicle is a cellular automaton, a cell in this context can be viewed in a broader sense. An intercellular signal is defined as a parameterized entity, a powerful paradigm that expresses not only lateral but also hierarchical and even recursive control structures. "Game of Life", L-System, Turing machine and other examples are included.

Introduction

A few years ago I found myself in the familiar surroundings of a coffee house, reading about a tragic disease in which the internal organs of the body fail to form proper left-right asymmetry (Belmonte 1999). The cause of this disease was discovered through its relatively abundant frequency among victims of cystic fibrosis, both diseases the result of defective cilia, the hair-like projections that cells use to produce mechanical motion in their environment. In the case of the symmetry problem, this caused an abnormal distribution of a chemical signal during early development.

It is commonplace knowledge that cells communicate and collaborate in myriad ways, yet for some reason this article particularly inspired me. Hence the paper you are reading. And of course, since for a computer scientist nature's creations are but suggestions and not mandates for artificial systems, the resulting work represents what nature would have designed, had she only the proper education.

This work was done using a two-dimensional cellular automaton, although, as will become apparent, the abstraction does not pertain only to this model. Cellular automata (von Neumann 1966) are inherently parallel computers (Talia, 2000), and are popular in artificial life research projects such as *Avida* (Adami and Brown 1994) and *Amoeba* (Pargellis 1996). Extensively analyzed (Langton 1992 and Wolfram 1984), cellular automata exhibit fascinating self-organizing behavior with the simplest of rules, such as those in Conway's classic *Game of Life* (Berlekamp, Conway, and Guy 1982).

In a conventional cellular automaton, the state of a cell at time t is typically determined by the states of a neighborhood of nearby cells at time $t-1$. An inherent assumption is that a cell must be "aware" of its neighbors. A cell may affect a distant (not in its neighborhood) cell only by, at least temporarily, changing the states of intervening cells. The drawback of this method is that it requires additional states and carefully orchestrated rules, especially if the intervening cells are to return to their former states.

The abstraction proposed here is that a cell's state is determined by the *signals* it receives, regardless of their origin. The naturally reasonable notion of locality is preserved, as signals are in fact transferred from cell to cell. However, a signal can be constructed

analogously to a nerve impulse, so as to use intervening cells as transport vehicles, arriving and taking effect only in an intended destination cell. In addition, interacting signals from different sources can be "focused" on an intended cell or group of cells. But it is not just the concept of a signal that enables this and, as will be demonstrated, other complex intercellular communication mechanisms, but also that a signal is defined as a parameterized entity, expressing not only lateral but also hierarchical and even recursive control structures.

The following examples are presented to illustrate the abstraction:

- Fractal spiral.
- Spiraling "Game of Life" glider guns.
- Lindenmayer-System (L-System).
- Turing machine.
- Bug.

The examples are arranged in rough order of complexity, so proceeding in sequence is recommended.

Description

The cellular automaton consists of a two-dimensional array in which each cell can exchange signals directly with itself and the eight adjacent cells (Moore neighborhood). A cell state represents a set of properties, such as color, orientation, etc. It is produced by *morphogenic* functions over the set of input signals from the environment:

$$state_i^{(t+1)} = morphogenic_j(Input\text{-}signals^{(t)})$$

A cell may also produce directional output signals that are a function of the input signals:

$$output\text{-}signal_i^{(t+1)} = morphogenic_j(Input\text{-}signals^{(t)})$$

A signal is a parameterized entity that may recursively contain other signals to be subsequently unloaded and issued:

<signal> := <signal type>[<parameter>$_0$...]
<parameter> := <signal>|<number>|<other>

A morphogenic function is an arbitrary, unconstrained function that uses combinations of signal types and parameters as input. As an example, in the Turing Machine described below, a sense signal arriving at a cell in which a tape signal is circulating results in the production of a read signal directed toward the cell representing the current state. A set of related functions and signals are called a *morphogen*.

Examples

The following examples are presented to demonstrate the capabilities of the abstraction. All cells initially reside in a quiescent state, producing no signals. The center cell is stimulated by an INITIAL signal to begin, causing the automaton to exhibit patterns determined by its morphogen.

The programs are written in the Java programming language (Flanagan 1997) as web-publishable applets. All morphogen objects are derived from a base object, and may be used as building blocks to construct new morphogens. The cell object contains state information, currently consisting of cell color, shape, and orientation. Cell orientation provides for relative directionality, including mirroring. Signal types are uniquely scoped to morphogens, and by convention each morphogen contains an INITIAL signal type for initialization purposes. The source code is available at www.acs.ilstu.edu/faculty/portegys/research/morphone.zip.

Fractal Spiral

The cellular automaton in this example produces the fractal pattern of square-cornered spirals shown in Figure 1, demonstrating the use of parameters to achieve this effect. There are three levels of scale to this fractal, the black lines representing the largest scale, proceeding to lighter shades of gray lines for the smaller scales. The fractal is constructed by planting a smaller scale spiral at each turn of a larger spiral.

The INITIAL signal initiates the process by creating a SPIRAL signal oriented northward. The SPIRAL signal contains the following parameters:

```
{ initialLength, lengthIncrement,
lengthCount, numberTurns, turnCount,
  turnAngle, orientation, color }
```

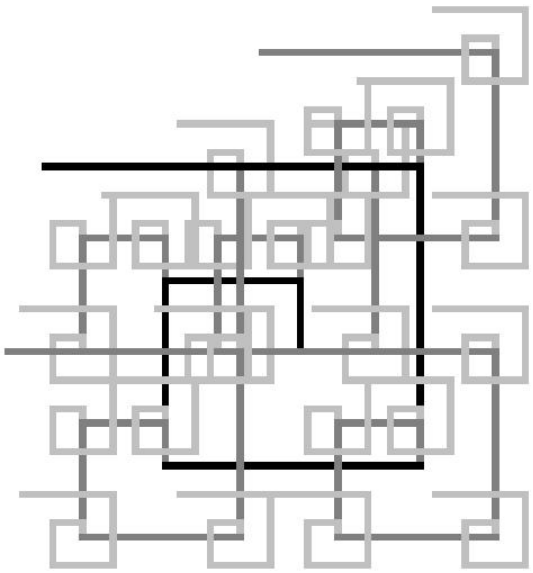

Figure 1 – Fractal Spiral

The length parameters serve to draw the straight sides of a spiral. The turn, angle, and orientation parameters count turns and specify side directions. When a cell receives a SPIRAL signal, its morphogen performs as follows:

```
cell.color = signal.color;
if (signal.lengthCount > 1)
{
  // Continue building side.
  signal.lengthCount = signal.lengthCount-1;
  output signal in received direction;
} else {
  // Turn spiral and lengthen next side.
  signal.turnCount = signal.turnCount+1;
  if (signal.turnCount > signal.numberTurns)
    return;
  signal.rotate(signal.turnAngle);
  signal.length = signal.length +
    signal.lengthIncrement;
  output signal in computed direction;
  // Create smaller scale spiral.
  fractalSignal = new signal object;
  fractalSignal.color =
    lighter(signal.color);
  fractalSignal.initialLength =
    signal.initialLength-fixed amount;
  if (fractalSignal.initialLength > 0)
  {
    output fractalSignal;
  }
}
```

The fractal spiral can be viewed at www.acs.ilstu.edu/faculty/portegys/programs/morphone/fractalspiral.html.

Spiraling Glider Guns

This example repeats the spiral pattern of the previous example for the purpose of demonstrating orientation as a state variable. In this case, the spiral deposits "Game of Life" *glider guns* at each turn juncture. A glider gun is an oscillating pattern that producing moving patterns called *gliders*. Figure 2 shows four glider guns oriented in the four cardinal directions, each emitting a stream of gliders.

Figure 2 – Spiraling Glider Guns

At each turn, the Spiral morphogen changes the cell orientation to match that of the incoming signal. This will cause any subsequent signals output from the cell to be relatively oriented. A simple modulo addition of the signal orientation and the cell orientation, starting with a value of 0 for the north direction and incrementing clockwise, does this. So for example, if the cell is oriented east, a northeast signal will proceed in

the southeast direction. The Spiral morphogen then issues a GliderGun INITIAL signal to create the glider gun.

The Game of Life rules demand that the glider gun configuration must start synchronously. To accomplish this, the GliderGun morphogen utilizes two service morphogens: Transporter and Timer. The TRANSPORTER signal for the Transporter morphogen carries a destination cell "address" in the form of x and y displacements, and a payload signal as parameters. The TIMER signal for the Timer morphogen carries a countdown timer and a payload signal as parameters. The GliderGun morphogen nests its START signals within TIMER signals, which are in turn nested within TRANSPORTER signals, bound for the cells that form the glider gun configuration. At each intervening cell en route, the x and y displacements direct the TRANSPORTER signal toward its destination, whereupon it issues the TIMER signal it carries. The TIMER signal "cycles" within the cell, counting down until the GliderGun START signal is issued.

The GliderGun morphogen consists of several signals and associated functions. In the Game of Life, each cell must know how many neighbor cells are "alive" in order to determine whether it is alive or dead. This is accomplished by cycling an ALIVE signal in a live cell. The ALIVE signal causes the morphogen to issue NEIGHBOR signals to neighboring cells. Counting the NEIGHBOR signals allows the morphogen to decide whether the cell is alive or dead.

The spiraling glider guns can be viewed at www.acs.ilstu.edu/faculty/portegys/programs/morphone/gliderguns.html.

L-System

A Lindenmayer-System or L-System is a grammar for modeling cellular growth using a string-rewriting scheme (Prusinkiewicz and Lindenmayer 1990). L-Systems also exhibit fractal properties of interest in mathematics and computer graphics. In an L-System grammar, every non-terminal symbol in a production is replaced for an arbitrary number of iterations. The resulting string is then interpreted as a program to draw an object such as that shown in Figure 3.

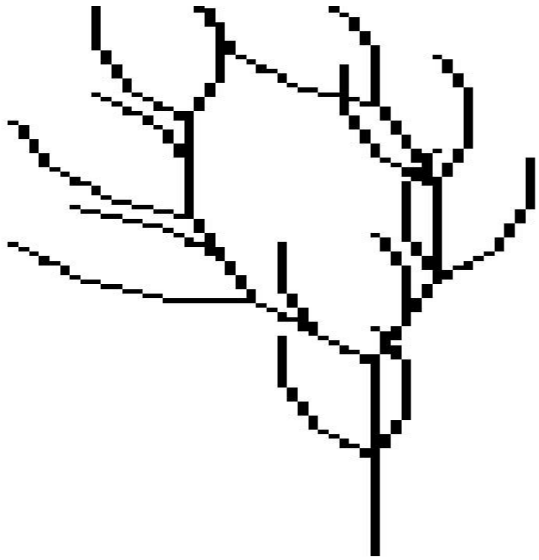

Figure 3 – L-System

For this example, the grammar consists of:

$$F \rightarrow FF+[+F-F-F]-[-F+F+F]$$

One iteration of this production yields the string used to draw the figure. The 'F' symbol is interpreted as a command to move forward, drawing a line. The '+' symbol is a command to turn clockwise by a certain number of degrees; conversely the '-' causes a counter-clockwise turn. The '[' and ']' symbols respectively push and pop a "subroutine" module consisting of the string that they enclose.

The LSYTEM signal for the Lsystem morphogen carries two parameters: a production string and an angle. The morphogen processes the production string by executing the symbols in order. The 'F' command causes an EDGE signal to be issued, which draws a line to a given point, determined by the current angle. The LSYSTEM signal is then packed into a TRANSPORTER signal and sent to the cell at the end of the drawn line. '+' or '-' commands increment or decrement the angle. '[' indicates the beginning of a substring module, which is extracted and

independently executed via a separate LSYSTEM signal.

The L-System can be viewed at www.acs.ilstu.edu/faculty/portegys/programs/morphone/lsystem.html.

Turing Machine

A Turing machine is a well-known, fundamental computer (Turing 1936) often used as a benchmark of computing capability. In this example, a binary adder is implemented, shown in Figure 4 in its starting configuration. The cells in the bottom row comprise the tape, containing the binary numbers to be added, 101 and 11. Other symbols are: 's' for start, 'b' for blank, '%' and '#' for separators. The answer will be written to the left of the leftmost '%'. The column of circles comprises the states of the machine; the filled circle representing the current state. The tape "read/write head" is over the cell beneath the column.

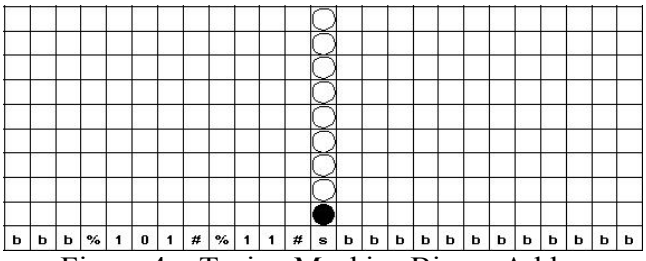

Figure 4 – Turing Machine Binary Adder

The Turing morphogen contains the signals shown in Table 1.

The machine works as follows: The cell containing the CURRENT (state) signal issues a SENSE command to the tape, which responds with a READ signal containing its value. If the current state calls for a write command, the WRITE signal is issued. Similarly, if the tape is to be moved, the MOVE signal is issued containing either a LEFT or RIGHT signal. Lastly, the current state sends the CURRENT signal to the cell containing the next state.

INITIAL	Initial external stimulus.
EGG	Lays out tape and states using transporter and timer morphogens.
VALUE	Tape value.
MOVE	Move tape command sent from state cell to tape; contains LEFT or RIGHT signal.
LEFT	Move tape left.
RIGHT	Move tape right.
SENSE	Command to sense value on tape.
READ	Response to sense command; contains value read.
WRITE	Command to write tape; contains value to write.
STATE	State information: read, write, move, and next state.
CURRENT	Current state.

Table 1 – Turing Morphogen Signals

The Turing machine can be viewed at www.acs.ilstu.edu/faculty/portegys/programs/morphone/turing.html.

Bug

The somewhat whimsical "bug" example shown in Figure 5 is intended to suggest the possibilities of the abstraction for artificial life research.

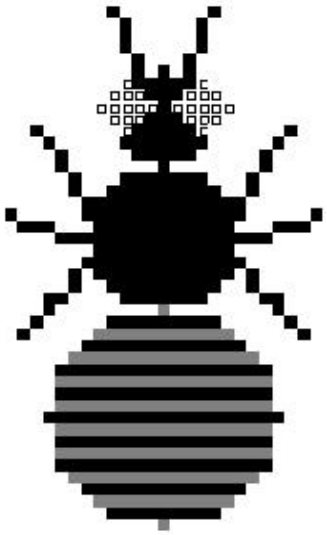

Figure 5 – Bug

The signals in the Bug morphogen are given in Table 2.

INITIAL	Initial stimulus; issues HEAD, THORAX, and ABDOMEN signals.
HEAD	Generates head; issues EYE signals.
EYE	Generates eye.
THORAX	Generates thorax; issues LEG signals.
LEG	Generates leg.
ABDOMEN	Generates abdomen.

Table 2 – Bug Morphogen Signals.

The Ball morphogen is used to create the head, thorax, and abdomen. The striping effect on the abdomen is an option of the BALL signal. Future enhancements of the bug could supply animation and developmental variables suitable for genetic algorithms.

The Bug can be viewed at www.acs.ilstu.edu/faculty/portegys/programs/morphone/bug.html.

Conclusion

The use of signals allows a cell's state transition processing to occur independently of the source of the signals, effectively making a cell's neighborhood a variable, dynamic entity. This feature also makes programming a cellular automaton significantly easier, as the examples should illustrate. In addition, cellular systems not based on the grid-like cellular automata motif may benefit from this abstraction; for example, a cluster of embryonic cells moving and replicating in a three-dimensional space.

References

Adami, C. and Brown, C.T. 1994, Evolutionary Learning in the 2D Artificial Life System "Avida", In Proceedings of Artificial Life IV, p. 337, Cambridge, Mass.: MIT Press.

Belmonte, J.C. 1999, How the Body Tells Left from Right, *Scientific American*, June 1999.

Berlekamp, E., Conway, J.H., and Guy, R. 1982, *Winning Ways for your Mathematical Pl*ays, New York: Academic Press.

Flanagan, D. 1997, *Java in a Nutshell: A Desktop Quick Reference*, O'Reilly & Associates.

Langton, C.G. 1992, Life at the Edge of Chaos, In Artificial Life II, p. 41, Redwood City, Calif.: Addison-Wesley.

Pargellis, A.N. 1996, The Spontaneous Generation of Digital Life, *Physica*, **D 91**, 86.

Prusinkiewicz, P. and Lindenmayer, A. 1990, *The Algorithmic Beauty of Plants*, New York: Springer-Verlag.

Talia, D. 2000, Cellular Processing Tools for High-Performance, *Computer*, 33(9): 44-52.

Turing, A.M. 1936, On Computable Numbers, with an Application to the Entscheidungsproblem In Proceedings of the London Mathematical Society, ser. 2. vol. 42 (1936-7), pp.230-265.

von Neumann, J. 1966, *Theory of Self Reproducing Automata*, Champaign, Ill.: University of Illinois Press.

Wolfram, S. 1984, Universality and Complexity in Cellular Automata, *Physica*, **D 10**, 1.