StarLogo: An Environment for Decentralized Modeling and Decentralized Thinking

Mitchel Resnick
MIT Media Laboratory
20 Ames Street
Cambridge, MA 02139 USA
+1 617 253 9783
mres@media.mit.edu

ABSTRACT

StarLogo is programmable modeling environment designed to help nonexpert users (in particular, precollege students) model and explore decentralized systems, such as ant colonies and market economies. People often have difficulty understanding the workings of such systems. By using StarLogo, people can move beyond the "centralized mindset"—that is, they begin to understand how patterns can arise through decentralized interactions, not from the dictates of a centralized authority.

Keywords
Educational applications, end-user programming, modeling

INTRODUCTION

In recent years, computer scientists have developed and experimented with a wide range of new programming paradigms: object orientation, logic, constraints, parallelism. Each of these paradigms offers new design possibilities—that is, new ways to create things with computers. But perhaps more important, each new paradigm also offers new epistemological possibilities—that is, new ways to think about computation and other phenomena in the world. An old saying goes something like this: If a person has only a hammer, the whole world looks like a nail. Adding new tools to the carpenter's toolkit changes the way the carpenter looks at the world. So, too, with computational paradigms: new paradigms can change the way computer users think about the world.

StarLogo is new modeling environment based on the paradigm of massive parallelism [3]. StarLogo is designed to help nonexpert users (such as precollege students) model the workings of decentralized systems, such as ant colonies and market economies. Users can write simple rules for thousands of objects, then observe the patterns that arise from the interactions. In doing so, users develop new ways of thinking about decentralized systems.

PARALLELISM

In most parallel-computing research, the primary goal is to improve the speed of computation. Many people see parallelism as a "necessary evil" in order to improve the speed at which programs execute. Indeed, some language developers try to hide parallelism from the user through "parallelizing compilers."

In creating StarLogo, I had a very different set of goals. I was not particularly concerned with performance or speed. Rather, I was interested in providing new ways for users to model, control, and think about actions that actually happen in parallel. Many things in the world really do act in parallel; the most natural way to model such situations is with a parallel programming language. In these cases, parallelism isn't a "trick" to improve performance; it is the most natural way of expressing the desired behavior.

THE CENTRALIZED MINDSET

StarLogo is especially designed for modeling *decentralized* systems—that is, systems in which patterns are determined not by some central authority but by local interactions among many (parallel) components. The world is full of such systems. As ants forage for food, their trail patterns are determined not by the dictates of the queen ant, but by local interactions among thousands of worker ants. Macroeconomic patterns arise from local interactions among millions of buyers and sellers. In immune systems, armies of antibodies seek out bacteria in a systematic, coordinated attack—without any "generals" organizing the overall battle plan.

In recent years, there has been a growing scientific interest in decentralized systems [4]. But most people continue to have great difficulty reasoning about and understanding such situations. When people see patterns in the world, they tend to assume some type of centralized control, even where it doesn't exist. When people see a flock of birds, for example, they typically assume that the bird in the front is leading and the others are following. But most bird flocks don't have leaders at all. Rather, each bird follows a set of simple rules, reacting to the movements of the birds nearby. Orderly flock patterns arise from these simple, local interactions [1]. StarLogo is intended to help people model such systems—and, in the process, move beyond the "centralized mindset."

STARLOGO

StarLogo is an extension of the Logo programming language [2]. In traditional versions of Logo, students create pictures and animations by giving commands to a graphic "turtle." StarLogo extends Logo in three major ways:

First, *StarLogo has many more turtles.* While traditional versions of Logo typically have only a few turtles, StarLogo has *thousands* of turtles—and all of the turtles can perform their actions at the same time, in parallel. For many colony-type explorations, having hundreds of turtles is a necessity. The behavior of a colony can change qualitatively when the number of turtles is increased. An ant colony with 10 ants might not be able to make a stable pheromone trail to a food source, whereas a colony with 100 ants (following the exact same rules) might.

Second, *StarLogo turtles have better "senses."* The traditional Logo turtle was designed primarily as a drawing turtle, for creating geometric shapes. But the StarLogo turtle is more of a behavioral turtle. StarLogo turtles come equipped with "senses." They can detect and distinguish other turtles nearby, and they can "sniff" scents in the world. Such turtle-turtle and turtle-world interactions are essential for creating and experimenting with decentralized phenomena. Parallelism alone is not enough. If each turtle just acts on its own, without any interactions, interesting colony-level behaviors will never arise.

Third, *StarLogo reifies the turtles' world.* In traditional versions of Logo, each pixel of the turtles' world has a single piece of state information—its color. StarLogo attaches a much higher status to the turtles' world. The world is divided into small square sections called *patches.* The patches have many of the same capabilities as turtles—except that they can not move. Each patch can hold an arbitrary variety of information. For example, each patch can keep track of the amount of "chemical" that has been released within its borders. Patches can also execute StarLogo commands, just as turtles do. For example, each patch could diffuse some of its "chemical" into neighboring patches, or it could grow "food" based on the level of chemical within its borders. Thus, the environment is given a status equal to that of the creatures inhabiting it.

StarLogo was originally implemented on the Connection Machine, a massively-parallel supercomputer. Recently, StarLogo was ported to the Macintosh. (To obtain a copy, write to `starlogo-request@media.mit.edu`)

EXAMPLE: SLIME-MOLD AGGREGATION

Figure 1 shows a StarLogo simulation inspired by the aggregation behavior of slime-mold cells. For many years, scientists believed that the aggregation process was coordinated by specialized slime-mold cells, known as "pacemaker" cells. But that is not the case. In fact, each slime-mold cell follows the same simple rules. In the StarLogo model, each "turtle" emits a chemical pheromone, while also following the gradient of the pheromone. The patches cause the pheromone to diffuse and evaporate. With this simple decentralized strategy, the creatures aggregate into clusters after several dozen time steps.

High-school students have used StarLogo in many similar projects, modeling the behaviors of traffic jams, termite colonies, and predator-prey ecosystems.

ACKNOWLEDGMENTS

Brian Silverman and Andy Begel implemented the Macintosh version of StarLogo and contributed many design ideas. Hal Abelson, Seymour Papert, Randy Sargent, and Uri Wilensky have provided encouragement and ideas for the StarLogo project. The LEGO Group and the National Science Foundation (grant 9358519-RED) have provided financial support for this research.

REFERENCES

1. Heppner, F., & Grenander, U. A Stochastic Nonlinear Model for Coordinated Bird Flocks. In *The Ubiquity of Chaos.* AAAS, Washington DC, 1990.
2. Papert, S. *Mindstorms.* Basic Books, New York, 1980.
3. Resnick, M. *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds.* MIT Press, Cambridge, MA, 1994.
4. Waldrop, M. *Complexity: The Emerging Science at the Edge of Order and Chaos.* Simon and Schuster, New York, 1992.

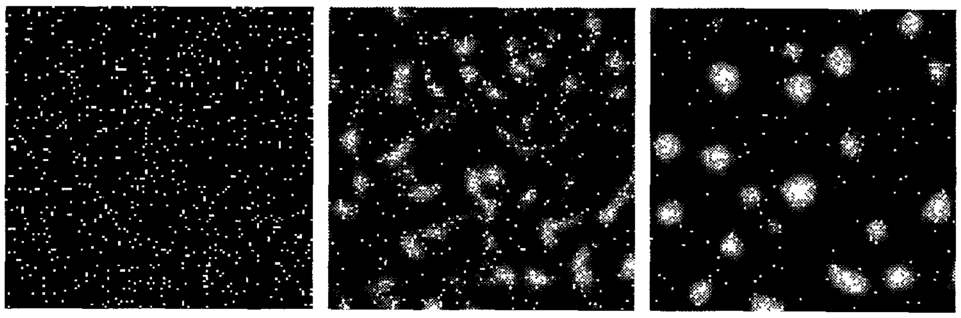

Figure 1
StarLogo simulation inspired by slime-mold aggregation.
Each "turtle" emits a chemical pheromone, while also following the gradient of the pheromone.