EvoPopcorn: A Web-Native Distributed Artificial Life Simulation

Luis Zaman
BEACON Center for the Study of Evolution in Action
Michigan State University
luis.zaman@gmail.com

JavaScript is on the way to being the ideal programing language. Despite its many shortcomings, the JavaScript community is growing astonishingly fast. The ubiquity of browsers capable of running JavaScript on every new cellphone, television, and even some kitchen appliances make it a potentially transformative technology that is nearly 20 years old[i]. In other words, the technology was clearly ahead of its time, but its time has finally come. Thus, it is fascinating to consider how a ubiquitous computing environment like the World Wide Web enables new artificial life applications.

To explore web native artificial life, we developed a simple simulation of 2-dimensional rigid-body organisms living in a simple "cage" with food blocks that can be manipulated by the user[ii]. The entire simulation is carried out natively in the browser through Canvas and the PhysicJS 2D physics engine[iii]. Each organism is composed of four circular bodies attached by rigid constraints. The circles' radii and angular velocities, as well as the lengths of the rigid constraints determine the morphology as well as the behavior of the organisms. Small and slowly rotating circles generally produce organisms that settle towards the bottom of the cage, while large and quickly moving circles generate erratically bouncing organisms.

We seed initial populations with randomly generated individuals, but allowed the traits that determine morphology and behavior (radii, angular velocity, and rigid constraint length) to mutate with a small probability. In every update, a random organism is removed and the winner of a tournament (tournament size of 4) produces an offspring to fill the vacant spot. Fitness in this simulation is simply the number of times an organism has collided with food blocks. Thus, the populations will evolve better-suited solutions to obtain food from their particular environment. How the environment shapes selection is most evident when food blocks are on opposite ends of the cage (Figure 1).

Considering the ubiquity of browsers and an artificial life simulation running natively within them, harnessing the interconnectedness of the World Wide Web is a potentially transformative paradigm in artificial life. To begin exploring these technologies, I extended the simulation presented above with the ability to serialize and transmit organisms through the WebSocket API (specifically, through Socket.io[iv]). This technology allows each connected browser to establish a persistent socket to a central server. Both the client and server can emit and subscribe to socket *events*, which enables real-time communication between browsers. Thus, by adding a small probability that a vacancy in the population is filled by an organism from the server rather than the offspring of a local organism, we have a rudimentary implementation of migration between browsers. In a sense, this is also a simple mechanism of distributed computation within browsers.

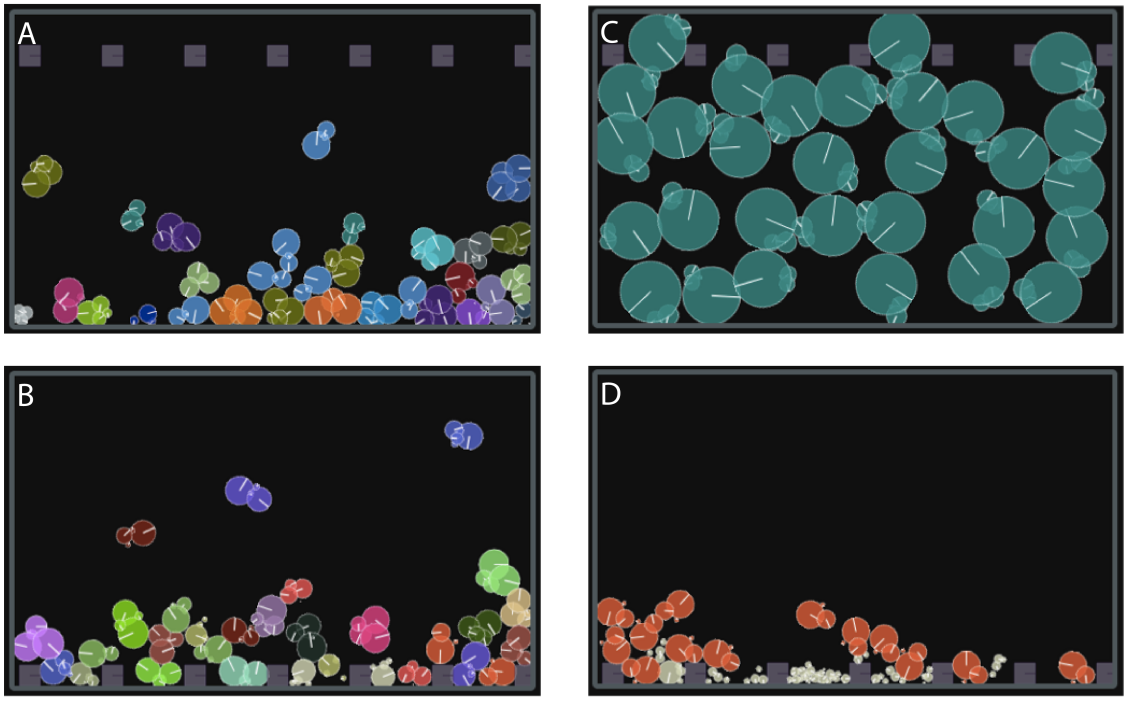

Figure 1: Evolution of different morphologies and behaviors depending on the environment. A and B show initially random populations in alternative environments (either low or high food). C and D show the resulting evolution after only a few minutes.

In the simulation, migrated organisms are temporarily colored bright pink when they are first born. After a short period of time they are returned to their original color. The source code for both the server and client are open-source and available on figshare[v]. Currently, each population is seeded with a random color pallet with the hopes that migrants will be easily identified. The physical connections between browsers are arranged in a star topology (all browsers connected to a central server), but the migration topology can take on any form from rings to small-world networks. In the future, more advanced tracking and "scoring" of individual populations will be added. This will allow us to measure the fitness of individuals across a range of environments, and even identify winning lineages that eventually overtake other populations. With these additions, it would be simple to create a game out of web based artificial life, where users get points based on how many invasions their organisms successfully carryout, and how many they prevent.

---

[i] http://en.wikipedia.org/wiki/ECMAScript
[ii] http://humpty.mmg.msu.edu:8080/
[iii] http://wellcaffeinated.net/PhysicsJS/
[iv] http://socket.io/
[v] Zaman, Luis (2014): EvoPopcorn - WebA'14. figshare. http://dx.doi.org/10.6084/m9.figshare.1009782