Editorial
Volume 13:6, 2020
DOI: 10.37421/jcsb.2020.13.325

Journal of Computer Science & Systems Biology

ISSN: 0974-7230 Open Access

Editorial on Digital Organism

Chinthala Mounica*
Department of Computer Science, Osmania University, India

Editorial Note

An advanced creature is a self-duplicating PC program that changes and develops. Advanced creatures are utilized as an apparatus to contemplate the elements of Darwinian development, and to test or check explicit speculations or numerical models of development. The investigation of computerized creatures is firmly identified with the region of counterfeit life.

Digital organisms can be traced back to the game Darwin, developed in 1961 at Bell Labs, in which computer programs had to compete with each other by trying to stop others from executing. A similar implementation that followed this was the game Core War. In Core War, it turned out that one of the winning strategies was to replicate as fast as possible, which deprived the opponent of all computational resources. Programs in the Core War game were also able to mutate themselves and each other by overwriting instructions in the simulated "memory" in which the game took place. This allowed competing programs to embed damaging instructions in each other that caused errors (terminating the process that read it), "enslaved processes" (making an enemy program work for you), or even change strategies mid-game and heal themselves.

Steen Rasmussen at Los Alamos National Laboratory took the idea from Core War one step further in his core world system by introducing a genetic algorithm that automatically wrote programs. However, Rasmussen did not observe the evolution of complex and stable programs. It turned out that the programming language in which core world programs were written was very brittle, and more often than not mutations would completely destroy the functionality of a program.

The first to solve the issue of program brittleness was Thomas S. Ray with his Tierra system, which was similar to core world. Ray made some key changes to the programming language such that mutations were much less likely to destroy a program. With these modifications, he observed for the first time computer programs that did indeed evolve in a meaningful and complex way.

Later, Chris Adami, Titus Brown, and Charles Ofria started developing their Avida system which was inspired by Tierra but again had some crucial differences. In Tierra, all programs lived in the same address space and could potentially execute or otherwise interfere with each other's code. In Avida, on the other hand, each program lives in its own address space. Because of this modification, experiments with Avida became much cleaner and easier to interpret than those with Tierra. With Avida, digital organism research has begun to be accepted as a valid contribution to evolutionary biology by a growing number of evolutionary biologists. Evolutionary biologist Richard Lenski of Michigan State University has used Avida extensively in his work. Lenski, Adami, and their colleagues have published in journals such as Nature and the Proceedings of the National Academy of Sciences (USA). Digital organisms can be traced back to the game Darwin, developed in 1961 at Bell Labs, in which computer programs had to compete with each other by trying to stop others from executing. A similar implementation that followed this was the game Core War. In Core War, it turned out that one of the winning strategies was to replicate as fast as possible, which deprived the opponent of all computational resources. Programs in the Core War game were also able to mutate themselves and each other by overwriting instructions in the simulated "memory" in which the game took place. This allowed competing programs to embed damaging instructions in each other that caused errors (terminating the process that read it), "enslaved processes" (making an enemy program work for you), or even change strategies mid-game and heal themselves.

Steen Rasmussen at Los Alamos National Laboratory took the idea from Core War one step further in his core world system by introducing a genetic algorithm that automatically wrote programs. However, Rasmussen did not observe the evolution of complex and stable programs. It turned out that the programming language in which core world programs were written was very brittle, and more often than not mutations would completely destroy the functionality of a program.

The first to solve the issue of program brittleness was Thomas S. Ray with his Tierra system, which was similar to core world. Ray made some key changes to the programming language such that mutations were much less likely to destroy a program. With these modifications, he observed for the first time computer programs that did indeed evolve in a meaningful and complex way.

Later, Chris Adami, Titus Brown, and Charles Ofria started developing their Avida system, which was inspired by Tierra but again had some crucial differences. In Tierra, all programs lived in the same address space and could potentially execute or otherwise interfere with each other's code. In Avida, on the other hand, each program lives in its own address space. Because of this modification, experiments with Avida became much cleaner and easier to interpret than those with Tierra. With Avida, digital organism research has begun to be accepted as a valid contribution to evolutionary biology by a growing number of evolutionary biologists. Evolutionary biologist Richard Lenski of Michigan State University has used Avida extensively in his work. Lenski, Adami, and their colleagues have published in journals such as Nature and the Proceedings of the National Academy of Sciences (USA).

How to cite this article: Chinthala Mounica. "Editorial on Digital Organism". *J Comput Sci Syst Biol* 13 (2020) doi: 10.37421/jcsb.2020.13.325

***Address for Correspondence:** Chinthala Mounica, Department of Computer Science, Osmania University, India, E-mail: chinthalamounica93@gmail.com*

Received 20 October 2020; **Accepted** 25 October 2020; **Published** 30 October 2020