Evolving Machines

Project Proposal

Group members

- Adeola Bannis - abannis@andrew.cmu.edu
- Mark Whiting - mwhiting@andrew.cmu.edu

Project mentor

Aniruddha Basak (TA) - aniruddha.basak@sv.cmu.edu

Proposal Summary

Despite a long history of genetic and evolutionary approaches to computational problems, one of the most valuable aspects of nature's evolution remains largely absent from these efforts - effective motivation of emergent complexity. We will build an approach for evolving mechanical systems that meet simple i/o requirements as a stepping stone to evolve more complex behaviour. Our eventual goal is to use these system to evolve solutions to more complex requirements as an example of a systems of systems that could exhibit emergent behaviour.

Optimization Application

First, we will use an algorithm to evolve solutions to a number of well defined i/o problems, essentially evolving simulated mechanical logic gates and processors. At later stages we will put these in systems with more complex fitness functions in an attempt to evolve more sophisticated behaviours such as emergence.

Application 1: given an input fixture and an output fixture, create a machine between them that produces a desired mechanical output signal from a mechanical input signal. We will optimize for fewer parts, and for closeness to the desired output signal.

Application 2: given a similar setup with more complex input and output signals, use results from application 1 in addition to the initial simple parts to generate desired outputs. Again, the optimization will aim at reducing the number of parts and accuracy of the output but we recursively add complexity to promote emergence.

Evolutionary Algorithms

We aim to adapt Genetic programing or Neuroevolution to treat simulated physical mechanisms as species with logical function. Using both mutation and crossover initially we are open to introducing better adjustment methodologies as system complexity requires it, i.e. Meta-Genetic Programming. This approach affords us a simple, straightforward representation with direct relevance to the behaviour of the outcomes and, increasingly fast and appropriate generational improvements that can accommodate high level structural modification. Parameters are not set by external datasets, as we are not using any, so we will incorporate them as learnt aspects of the EA where possible.

Project Plan

We are implementing this project in c, as an OSX application, using our personal laptops. Our implementation uses the Chipmunk2D physics engine[1] to simulate the physical environment. Due to our problem setup we will not use any major dataset. Once the system is functioning we will run ongoingly as we continue to iterate it for the secondary application. The success of the experiment will be assessed based on the final complexity achieved, measured using an approach based adapted from physics [2, 3] and evolutionary computation[4] literature.

Project Schedule

Date	Task
September 26	Set up the simulation environment
September 30	Proposal & Demonstrating Working Environment
October 7	Application 1 - Building EA and setting up an I/O schema
October 14	Application 1 - Testing and dealing with bugs
October 21	Application 1 - Producing output
October 28	Application 2 - Setting up EA for more complex I/O
November 4	Application 2 - Testing and dealing with bugs
November 11	Application 2 - Producing output
November 18	Experimenting, iterating and dealing with bugs
November 25	Improving UI, running final experiments and dealing with bugs

[1] http://chipmunk-physics.net

[2] Bennett, C. "On the Nature and Origin of Complexity in Discrete, Homogeneous, Locally-Interacting Systems," Foundations of Physics 16 (1986) 585-592

[3] Grassberger, P. "Toward a Quantitative Theory of Self-Generated Complexity," International Journal of Theoretical Physics 25 (1986) 907-938

[4] Teo, J. & Abbass, H. A. "Multiobjectivity and Complexity in Embodied Cognition", IEEE Transactions on Evolutionary Computation 9 (2005) 337-360