A Parallelized Framework for Evolutionary Computation

Geoffrey Saxton Long (*260403840*)
McGill University, Quebec
Geoffrey.Long@mail.mcgill.ca

Abstract

Evolutionary algorithms are a common approach to problems with indeterminate strategies or lengthy computation times when exact results are not necessary. The goal of this project is to implement an extensible framework which allows for parallelization of an evolutionary algorithm. Although computation speed is a primary goal, I would also like to see the outcomes where "populations" of individuals are allowed to evolve in partial, or complete, isolation from one another. Each one of these populations would be implemented on a multithreaded Beowulf cluster, exposure to other populations would occur via MPI message passing. Within each cluster the populations would be evolved through different mutation, crossover, and fitness evaluation methods. This variance in operators would ensure that the populations diverge.

This framework will implement a genetic algorithm. Although the algorithm will be tested with the Travelling Salesperson Algorithm, it will be designed to work with a wide variety of problems. The overall performance of the framework will be evaluated on the results and speedup compared to the sequential version.

1. Introduction

Evolutionary Computation is a branch of computational intelligence commonly used to solve problems with multiple parameters or complex relationships between the parameters, multiple local optima, or no known approach to solving the problem. The term Evolutionary computation covers several different implementations. These are evolutionary programming, genetic programming, genetic algorithms, and evolution strategies. Although the approaches to each of these frameworks is slightly different, they all have the same general structure and themes. Central to all of them is the adherance to Darwinian principles.

The darwinian principles central to evolutionary computation are those of evolution by natural selection. This is commonly broken into four main themes:

1. More individuals are produced each generation than can survive
2. Variation exists amoung individuals, this variation is inhertiable
3. Those individuals with inherited traits better suited to the environment will survive
4. When reproductive isolation occurs a new species will form

survival of the fittest (natural selection), mutation, mating, As the name suggests, Evolutionary Computation methods follow the common themes of evolution as described by Darwin.

1.1. Terminology

In Evolutionary Computation you have a set of *individuals*, also known as a *population*. Each of these individuals is a possible solution to the problem. The way in which the solution is encoded is often unique to the problem and the implementation of the problem. At its core though, the solution must be formed in such a way that it can be changed by the evolutionary operators in the framework.

The evolutionary operators most commonly used are *mutation* and *crossover*. Although mutation is present in all evolutionary computation methods, crossover is often specific to genetic algorithms. Mutation is the alteration of an individual

via a slight change in their genetic makeup. This typically happens by changing a value of one of the individual's alleles or by switching two or more alleles. As a result, the resultant individual after mutation is slightly different than the original. Crossover involves the creation of a new individual by the combination of two "parent" individuals. Parents are often selected by their genetic fitness, though selection can be random. These crossover operators usually involve combining the two parents to create one or more new individuals.

2. Background

3. Implementation

4. Results

5. Conclusions

6. Sources