Concurrent GARTH (Genetic AlgoRiTHms) in C++11: A Framework for Lock-free GAs

J. Caleb Wherry
Virginia Tech
Department of ESM
Blacksburg, Virginia
cwherry@vt.edu

ABSTRACT

Genetic algorithms (GAs) are a fasinating area of optimization theory that draw from ideas set forth by Charles Darwin in *The Orgin of Species*. GAs have been shown to improve optimizations that we as humans cannot obtain on our own because of our inherint linear thinking. Non-linear optimization problems benefit most from GAs since GAs are able to search the non-linear space much farther and find optimal solutions much quicker than traditional optimization methods.

In this project we would like to take the typical framework that GAs are built on and make it concurrent and lock-free using the new C++11 standard. This will involve multiple stages in the development cycle that deal with designing the concurrent object for the GA threads to work on, implementing the GA framework using the new concurrent object, and then benchmarking the concurrent framework on some real-world applications.

We have done previous work in sequential GAs and have developed a framework in Java that runs on a uniprocessor. We will draw from this knowledge when developing our new Lock-free GA framework. After the concurrent framework has been developed, we will benchmark the framework based on the number of threads that work on the GA. We will then be able to analyze the tradeoffs in using more threads and how much speedup we obtain using the concurrent object.

As of right now, the real-world application will be a physics problem that is non-trivial in quantum condensed matter physics: the optimal placement of charges on sphere with the lowest energy state[1]. GAs have been shown to optimize this problem more efficiently than tarditional methods when the number of particles exceeds 100.

The main goal of this project is to apply a concurrency model to GAs for real-world problems. The difficulty here will be in integrating the Lock-free object into the framework such that we see a gain in speed when using more threads, not a speed lose that is occured in using the lock-free object.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

1. SCHEDULE

1. February:
   (a) Architect and design GA framework (Note: almost done).
   (b) Research lock-free objects that will best fit the GA process (Lock-free queue or Lock-free B-tree).
   (c) Implement lock-free object that is decided upon from above research.
2. March:
   (a) Finish last details for lock-free object.
   (b) Implement GA framework to work with new concurrent object.
   (c) Formulate quantum CMP application requirements.
   (d) Submit mid-semester report.
3. April:
   (a) Complete GA framework.
   (b) Apply framework to quantum application.
   (c) Run benchmarks based on varying numbers of threads.
   (d) Write final proposal and create presentation.
4. May:
   (a) Submit final report and give presentation!

2. INTRODUCTION

This is the introduction...

3. SOMETHING ELSE...

And another section...

3.1 A subsection...
How about a subsection?

3.2 And another...
Or another?

4. CONCLUSIONS
And finally conclusions!

5. REFERENCES
[1] T. Pang. *Introduction to Computational Physics*. Cambridge University Press, Cambridge, 2006.