MRPSO: MapReduce Particle Swarm Optimization

Andrew W. McNabb
Brigham Young University
Computer Science Dept.
3361 TMCB, Provo, UT 84602
a@cs.byu.edu

Christopher K. Monson
Google, Inc.
4720 Forbes Ave., Lower Level
Pittsburgh, PA 15213
c@cs.byu.edu

Kevin D. Seppi
Brigham Young University
Computer Science Dept.
3361 TMCB, Provo, UT 84602
k@cs.byu.edu

ABSTRACT

In optimization problems involving large amounts of data, Particle Swarm Optimization (PSO) must be parallelized because individual function evaluations may take minutes or even hours. However, large-scale parallelization is difficult because programs must communicate efficiently, balance workloads and tolerate node failures.

To address these issues, we present MapReduce Particle Swarm Optimization (MRPSO), a PSO implementation based on Google's MapReduce parallel programming model.

Categories and Subject Descriptors

G.1.6 [**Numerical Analysis**]: Optimization—*nonlinear programming, unconstrained optimization*

General Terms

Algorithms

Keywords

Swarm intelligence, Parallelization, Optimization

1. INTRODUCTION

Particle Swarm Optimization (PSO) is an optimization algorithm that was inspired by experiments with simulated bird flocking [1]. Many functions, especially those involving large amounts of data, take a long time to evaluate. To optimize such functions, PSO must be parallelized.

A parallel implementation of PSO must address a variety of issues. Inefficient communication or poor load balancing can hinder scalability. Once a program successfully scales, it must still address the issue of failing nodes. If a node fails on average once a year, then the probability of at least one node failing during a 24-hour job is 50.5% on a 256-node cluster and 93.6% on a 1000-node cluster.

Google faced these same problems in large-scale parallelization and created a common system called MapReduce to simplify its hundreds of specialized data processing programs [2]. MapReduce programs, which are formulated as a map function and a reduce function, automatically benefit from advanced mechanisms for communication, load balancing and fault tolerance.

GECCO'07, July 7–11, 2007, London, England, United Kingdom.
ACM 978-1-59593-697-4/07/0007.

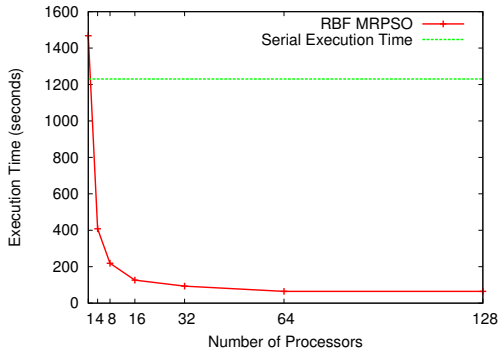

Figure 1: RBF execution times with 10,000 points

MapReduce Particle Swarm Optimization (MRPSO) is a parallel implementation of PSO for computationally intensive functions. MRPSO is simple, flexible, scalable and robust because it is designed in the MapReduce parallel programming model.

2. RESULTS

We evaluate MRPSO with a machine learning problem involving large amounts of data, specifically, training a radial basis function (RBF) network by minimizing error. We implemented MRPSO in Python and ran experiments on BYU's Marylou4 supercomputer using Hadoop, an open-source implementation of MapReduce. Figure 1 shows the performance of MRPSO through 128 processors on the RBF training problem.

3. CONCLUSIONS AND FUTURE WORK

MRPSO is a robust but simple implementation of PSO which scales well with a large number of processors. In future work, we will experiment with larger problems, analyze communication overhead and test other swarm topologies.

4. REFERENCES

[1] James Kennedy and Russell C. Eberhart. Particle swarm optimization. In *International Conference on Neural Networks IV*, pages 1942–1948, Piscataway, NJ, 1995. IEEE Service Center.

[2] Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified data processing on large clusters. *Sixth Symposium on Operating System Design and Implementation*, November 2004.