Parallelization of Enhanced Firework Algorithm using MapReduce

Simone A. Ludwig, Department of Computer Science, North Dakota State University, Fargo, ND, USA

Deepak Dawar, Department of Computer Science, North Dakota State University, Fargo, ND, USA

ABSTRACT
Swarm intelligence algorithms are inherently parallel since different individuals in the swarm perform independent computations at different positions simultaneously. Hence, these algorithms lend themselves well to parallel implementations thereby speeding up the optimization process. FireWorks Algorithm (FWA) is a recently proposed swarm intelligence algorithm for optimization. This work investigates the scalability of the parallelization of the Enhanced FireWorks Algorithm (EFWA), which is an improved version of FWA. The authors use the MapReduce platform for parallelizing EFWA, investigate its ability to scale, and report on the speedup obtained on different benchmark functions for increasing problem dimensions.

Keywords: Fireworks, Map, Reduce, Scalability, Swarm Intelligence

1. INTRODUCTION
Optimization is the process for searching for the best solution from a set of feasible solutions for a given problem with a set of constraints. Real world optimization problems can be challenging, apart from being highly complex and time consuming to solve, a given problem’s objective function may also be non-continuous and non-differentiable, which adds another layer of complexity. These real world optimization problems are ubiquitously found in various scientific and engineering domains.

For solving complex real world problems, researchers have been looking into optimization techniques inspired by natural processes such as Darwinian evolution, social group behavior and foraging strategies. Over the past few decades a significant growth in the field of nature-inspired optimization algorithms has been seen. There are two main categories of algorithms: evolutionary computing methods and swarm intelligence algorithms.

DOI: 10.4018/IJSIR.2015040102
The common idea underlying evolutionary algorithms is that given a population of individuals, natural selection (biologically referred to as survival of the fittest) is used to improve the fitness of the overall population. Given a function to be maximized, a set of candidate solutions is randomly created and the fitness function used as a fitness measure (the higher the better) is applied. Based on this fitness measure, some of the better candidates are chosen to undergo recombination and mutation. Recombination is applied to two candidates and results in one or more new candidates, whereas mutation is only applied to one candidate and results in one new candidate. After recombination and mutation are applied a set of new candidates replace the old ones and the next generation begins. This process is iterated until a candidate with sufficient quality is found or a predefined number of iterations is reached (Eiben, 2003).

Swarm intelligence is characterized by the collective behavior of decentralized, self-organized systems that are typically made up of a population of simple agents interacting locally with one another and with their environment (Beni & Wang, 1989). The behavior of swarm optimization can be envisioned by comparing it to swarms searching for optimal food sources, where the direction in which each individual is influenced is its current movement, the best food source it ever experienced, and the best food source any individual in the swarm ever experienced.

There are several different algorithms that fall into the swarm intelligence category. The most famous algorithm is the Particle Swarm Optimization (PSO) algorithm (Eberhart & Kennedy, 1995). The PSO algorithm searches for the best solution by keeping track of the best solution of each particle and weighing this with the best solution of the swarm. Another well-known algorithm is the Ant Colony Optimization (ACO) algorithm (Dorigo, Maniezzo, & Colorni, 1996), which uses the interaction of ants. Similarly, the cuckoo search (Gandomi, Yang, & Alavi, 2013) uses the brooding parasitism of cuckoos, and the bat algorithm (Yang & Gandomi, 2012) uses the echolocation of foraging bats, whereas the bee algorithm (Yang, 2005) uses the foraging behavior of honeybees. Another algorithm that belongs to the swarm intelligence family is the Firefly algorithm (Yang, 2009). The algorithm is inspired by the flashing behavior of fireflies. The primary purpose for a firefly’s flash is to act as a signal system to attract other fireflies.

FireWorks Algorithm (FWA) (Tan & Zhu, 2010) is a recently developed swarm intelligence algorithm. It tries to simulate the firework explosion process to find the optimal solution in the search space. A firework is an initial location, which produces sparks (adjacent locations) through a series of explosions. It was reported in (Tan & Zhu, 2010) that FWA significantly outperforms Standard Particle Swarm Optimization (SPSO) and Clonal PSO (Tan & Xiao, 2007). FWA and its variants (Zheng, Xu, & Ling, 2012) have been studied and evaluated on both single and multi-objective optimization problems (Zheng, Song, & Chen, 2013). FWA has also been successfully applied to many real life optimization problems (Janecek & Tan, 2011).

Its initial success notwithstanding, FWA has its own drawbacks. It tends to do poorly on shifted functions, and has a high computational cost relative to other swarm intelligence algorithms. Enhanced FireWorks Algorithm (EFWA) is an improved version of FWA proposed in (Zheng, Janecek, & Tang, 2013) that tries to overcome its shortcomings, and has shown promising results. EFWA brings down the computational cost of FWA and at the same time outperforms FWA significantly.

In this paper, we introduce a parallel version of the EFWA algorithm and evaluation the implementation with a scalability analysis using different benchmark functions and multiple dimensions. The implementation makes use of the MapReduce paradigm that allows the algorithm to scale to thousands of computers. We will evaluate the MapReduce implementation using an available infrastructure that provides us with the necessary computing power. Furthermore, a comparison with a MapReduce PSO implementation is conducted.
Adaptive Neuro-Fuzzy Control Approach Based on Particle Swarm Optimization
Gomaa Zaki El-Far (2012). Innovations and Developments of Swarm Intelligence Applications (pp. 81-98).
www.igi-global.com/chapter/adaptive-neuro-fuzzy-control-approach/65807?camid=4v1a

Optimum Design of Hybrid EDFA/FRA by Particle Swarm Optimization
www.igi-global.com/chapter/optimum-design-hybrid-edfa-fra/72827?camid=4v1a