Chapter 42
On the Accelerated Convergence of Genetic Algorithm Using GPU Parallel Operations

Cheng-Chieh Li
Tunghai University, Taiwan

Chu-Hsing Lin
Tunghai University, Taiwan

Jung-Chun Liu
Tunghai University, Taiwan

Winston Lo
Tunghai University, Taiwan

ABSTRACT

The genetic algorithm plays a very important role in many areas of applications. In this research, the authors propose to accelerate the evolution speed of the genetic algorithm by parallel computing, and optimize parallel genetic algorithms by methods such as the island model. The authors find that when the amount of population increases, the genetic algorithm tends to converge more rapidly into the global optimal solution; however, it also consumes greater amount of computation resources. To solve this problem, the authors take advantage of the many cores of GPUs to enhance computation efficiency and develop a parallel genetic algorithm for GPUs. Different from the usual genetic algorithm that uses one thread for computation of each chromosome, the parallel genetic algorithm using GPUs evokes large amount of threads simultaneously and allows the population to scale greatly. The large amount of the next generation population of chromosomes can be divided by a block method; and after independently operating in each block for a few generation, selection and crossover operations of chromosomes can be performed among blocks to greatly accelerate the speed to find the global optimal solution. Also, the travelling salesman problem (TSP) is used as the benchmark for performance comparison of the GPU and CPU; however, the authors did not perform algebraic optimization for TSP.

DOI: 10.4018/978-1-5225-0788-8.ch042

INTRODUCTION

The genetic algorithm, which imitates the process of natural selection, is a heuristic approach to produce solutions for global search and optimization problems. Basically, it is a high performance, parallel, global search method consisting of five major phases: initialization, evaluation, selection, crossover, and mutation. From continuous iterations of these five phases, an approximately optimal solution can be reached. However, the genetic algorithm itself has some drawbacks, such as premature convergence at local optimal solutions. To solve these issues, this research proposes to combine the simulated annealing method with the genetic algorithm and take advantage of parallel computing with the GPU. For parallel computing, the GPU is fundamentally different from the CPU, due to apparent difference of the number of cores and the computing architecture. Since GPU architecture is based on SIMD, we need to focus on memory access techniques to realize a high level of parallelization for genetic algorithms using GPUs. We also replace the mutation step in genetic algorithm with the simulated annealing method. In this way, the search rate for global optimal solutions of the genetic algorithm is greatly improved. More importantly, GPUs with many thousands of cores can handle a much larger population for genetic algorithms a general purpose CPU. Hence, we are able to use the island model (Whitley, D., Rana, S., & Heckendorn, R. B,1999; Whitley, D., Rana, S., & Heckendorn, R. B,1997; Gordon, V. S., & Whitley, D,1993) to perform block divisions on the large scale population to easily find the global optimal solution for genetic algorithms.

1. BACKGROUND

1.1. HAS Architecture

Inside conventional computing architecture, only one CPU, or a multi-core CPU handles all operations. To execute massive and high speed operations, lots of CPUs are required, resulting in increasing hardware cost and electric power consumption. The benefit of Heterogeneous System Architecture (HSA) that integrates CPUs and GPUs is that it selects operations of different properties inside an algorithm and sends them to CPUs or GPUs with better hardware architecture for these operations. Thus, we can not only implement optimal hardware architecture for specific algorithms but also execute operations in GPUs and CPUs in parallel to accelerate computations

1.2. Hardware Architecture of GPU

GPUs originally are hardware designed to handle graphics rendering, but in recent years GPU manufacturing companies, such as AMD and NVIDIA, start developing techniques to utilize the large amount of cores inside GPUs for computation. In this research, we adopt Kepler micro architecture developed by NVIDIA, in which the core unit is called as the Streaming Multiprocessor (SMX). Each SMX consists of 192 CUDA cores. As shown in Figure 1, each CUDA core can be treated as a thread processing unit. Each GPU chip has more than one SMX core, which explains the advantage of GPU for parallel computing. But since GPUs are limited by adopting Single instruction, multiple data (SIMD) (Garland, M., Le Grand, S., Nickolls, J., Anderson, J., Hardwick, J., Morton, S., Phillips, E., Zhang, Y., Volkov, V,2008;