Combinatorial optimization of NP-hard problems using dynamic GPU programming

Project brief by Kim Svensson
Supervised by Dr. Sarvapali D. Ramchurn

April 28, 2013

Problem

With NP-hard problems and algorithms with a complexity class of P or greater, a doubling in CPU speed allows only for a small change in sample size due to the nature of such problems, as execution speed and sample size of an algorithm is mutually exclusive. As such, computer scientists have to find other ways of computing difficult problems on large sets of data. The traditional approach has been to increase the CPU speed and the number of CPUs. However, such implementation is often costly and suffers from latency and architectural difficulties. As the GPU function like a single instruction multiple data (SIMD) architecture with up to thousands of programmable cores, researchers have used GPUs to run concurrent algorithms faster than the equivalent sequential algorithms designed for CPUs. Especially since the development of NVIDIA CUDA and OpenCL implementation in both hardware and software. This advance in computer science can allow for large computations that were previously unfeasible by leveraging the parallel nature of GPUs. In combinatorial optimization the problem consist of finding the optimal solution for a problem, which for a large set of data can be unfeasable if the problem is NP-hard.

Goals

The goal of this project is to implement a dynamic programming algorithm for an NP-Hard problem running on a GPU. In order to measure any performance gain from using a GPU as an execution unit, a comparable CPU-bound algorithm has to be implemented as a baseline. Further, a sequential algorithm is not suited for GPUs so the code has to be altered to leverage the parallel execution within the bounds of the GPU. For this purpose, knowledge of concurrent programming and the GPU limitations will be explored and evaluated to determine the best suited methods. Last, the end goal is to develop a concurrent algorithm for the GPU which substantially lowers the time complexity of the problem evaluated. The Combinatorial Auction Problem is a well-suited use case because of the large set of possible combinations, which can be equal to the power set of all available items. Which is where this project can show the major advantage using GPU programming to solve for large sets of data.

Scope

The project should give an understanding and implementation of said problem running on a GPU, where you can draw conclusions on whether the implementation allows for a significant shorter run-time. If possible, the code for the GPU will be further optimized and refined to allow for an even shorter execution time. If time allows, a graphical representation application will be developed where it is applicable to enable a visualisation of problems and its paths depending on the input parameters and internal restrictions.