Neighborhood Evaluation on GPU for the DVRP

Geir Hasle and Christian Schulz

SINTEF ICT, P. O. Box 124 Blindern, NO-0314 Oslo, Norway
{gha,csc}@sintef.no

Keywords: Parallel computing, heterogeneous computing, GPU, local search, VRP.

1 Introduction

For many applications of vehicle routing, there is still a large gap between the requirements and the performance of today's decision support systems. Although there has been a tremendous increase in the ability to solve ever more complex VRPs (partly due to methodological improvements, partly due to the general increase in computing power), the ability to consistently provide better routing plans in shorter time across a variety of instances will give substantial additional savings. In a generic vehicle routing tool, some form of approximate solution method is required [7,8]. Metaheuristics constitute a popular basis for solving rich and large-size VRPs [2,3,6]. For trajectory based metaheuristics, the typical computational bottleneck is neighborhood evaluation. Similarly, the evaluation of individuals is the bottleneck of population based metaheuristics. These tasks are embarrassingly parallel and prime candidates for parallelization [1].

Around year 2000, the architecture of processors for commodity computers started to change. Due to technological limits, the still prevailing Moore's law no longer materialized in the form of a doubling of clock speed every 18 months or so. Hence, the tongue-in-cheek "Beach law"[1] was no longer true. Multi-core processors with an increasing number of cores and higher theoretical performance than their single core predecessors emerged, but each core had lower clock speed. Solution methods for Discrete Optimization Problems (DOPs) need task parallelization to benefit from this development. In addition, there has been a drastic improvement of performance and general programmability of massively parallel stream processing (data parallel) accelerators such as Graphics Processing Units (GPUs). To fully profit from the general recent and future hardware development on modern PC architectures, heterogeneous DOP methods that combine task and data parallelism must be developed. The heterogeneous architecture also invites to a fundamental re-thinking of solution methods.

Although there is a substantial literature on task parallel methods for the VRP [5,14], the literature on data parallel computing for the VRP is almost non-existent [9,13]. However, GPU implementation of local search has been investigated earlier [12]. In the work reported here, we have carefully assessed the potential of GPU computing in metaheuristics for the VRP. We focus on the classical capacitated, Distance-constrained VRP (DVRP), and investigate how GPU-based stream processing may speed up metaheuristics through massively parallel neighborhood exploration.

2 GPU-based Neighborhood Evaluation

The platform for our investigations is a modern NVIDIA Fermi architecture GPU (GTX 480) with 32 streaming processors, each with 16 cores, i.e., a total of 512 cores for parallel computation. The basic and simple idea we investigate is to create one thread for each member that executes feasibility check and delta evaluation of the objective. Conceptually, the GPU evaluates the full neighborhood in parallel in this design. In practice, large neighborhoods will be split automatically and explored in several iterations. For high GPU utilization and maximum speed performance there are a number of implementation issues, the most important ones being related to configuration and use of memory structures, memory latency, and code diversion for configurable thread blocks.

Our DVRP solver uses a giant tour representation with artificial depot nodes between tours. Further, we utilize resource extension functions and investigate segment hierarchies for constant time move

[1] One way of doubling the performance of your computer program is to go to the beach for two years and then buy a new computer.

evaluation [10,11]. A savings algorithm is used to create an initial solution. Neighborhood generation, exploration, and identification of the best move through a stream reduction operator are fully executed on the GPU. The CPU is only used for initializing the problem instance and for keeping track of the incumbent solution. The GPU implementation issues also influence on the optimal choice of datastructures, segment hierarchy depth, etc.. We have identified ten dimensions for a thorough experimental investigation. The optimal choices depend on the local search operator selected. CUDA, a proprietary C++-like development environment was used for the coding.

We investigated local search with 2-opt and 3-opt on the full giant tour representation experimentally using a partial combinatorial setup for the ten dimensions. Sixty well-known DVRP instances of Christofides, Golden et al., and Li et al. with 30-1200 customers were used for our experiments. Naïve GPU implementations of local search typically give a speedup of one order of magnitude relative to a straightforward, sequential CPU implementation. With moderate efforts that focus on memory latency and code diversion, a second order of magnitude can be reached. Our results show that through careful tuning, a third order of magnitude is possible. For the 1200 customer instance the solver investigated 1.2 billion 3-opt moves in 14 seconds, an average of 12 ns per move. Our results also show that GPU implementation of 2-opt only pays off for large size instances.

The GPU is a very powerful stream processing accelerator that will significantly speed up metaheuristics for the VRP and other discrete optimization problems. GPUs of today are ideal for performing relatively simple, non-diverging procedures on many pieces of data; they may be used as intensification machines in local search based metaheuristics. We continue our investigations on heterogeneous computing for the VRP. Currently, we investigate how to utilize stream processing to realize metaheuristic mechanisms, and how to achieve a balanced CPU/GPU utilization through self-adaptive search control and careful allocation of subtasks to the available components of a heterogeneous computing system.

References

[1] Alba E. (editor): Parallel Metaheuristics – A New Class of Algorithms. Wiley 2005. ISBN-13 978-0-471-67806-9.

[2] Bräysy O., M. Gendreau, G. Hasle, and A. Løkketangen: A Survey of Heuristics for the Vehicle Routing Problem, Part I: Basic Problems and Supply Side Extensions. SINTEF Report, Oslo, Norway, 2008.

[3] Bräysy O., M. Gendreau, G. Hasle, and A. Løkketangen: A Survey of Heuristics for the Vehicle Routing Problem, Part II: Demand Side Extensions. SINTEF Report, Oslo, Norway, 2008.

[4] Brodtkorb A. R., C. Dyken, T. R. Hagen, J. M. Hjelmervik and O. O. Storaasli: State-of-the-Art in Heterogeneous Computing. Scientific Programming, 18(1) (2010), pp. 1-33.

[5] Crainic T.G.: Parallel Solution Methods for Solving Vehicle Routing Problems. In Golden B., S. Raghavan, E. Wasil (eds): The Vehicle Routing Problem – Latest Advances and New Challenges. Springer 2008. ISBN 978-0-387-77778-8.

[6] Gendreau M., J.-Y. Potvin, O. Bräysy, G. Hasle, A. Løkketangen: Metaheuristics for the Vehicle Routing Problem and extensions: A Categorized Bibliography. Chapter in The Vehicle Routing Problem: Latest Advances and New Challenges, edited by B. Golden, S. Raghavan, and E. Wasil, Springer, 2008.

[7] Hall R.: On the road to integration. ORMS Today 33(3):50-57, 2006.

[8] Hasle G., O. Kloster: Industrial Vehicle Routing Problems. Chapter (pp 397-432) in Hasle G., K-A Lie, E. Quak (eds): Geometric Modelling, Numerical Simulation, and Optimization – Applied Mathematics at SINTEF. ISBN 978-3-540-68782-5, Springer 2007.

[9] Hasle G., Kloster O., Riise A., Schulz C., Smedsrud M.: Using Heterogeneous Computing for Solving Vehicle Routing Problems. TRISTAN VII Book of Extended Abstracts, p354-357 http://www.sintef.no/project/tristan/Tristan%20VII%20-%20Details%20-%20Current.pdf

[10] Irnich S.: A Unified Modeling and Solution Framework for Vehicle Routing and Local Search-Based Metaheuristics. INFORMS Journal on Computing 20(2): 270-287 (2008)

[11] Irnich S., Resource extension functions: properties, inversion, and generalization to segments, OR Spectrum 30 (2008) 113–148.

[12] Luong T.V., N. Melab, E.G. Talbi: "Neighborhood structures for gpu-based local search algorithms". Parallel Processing Letters 20 (2010) 307–324.

[13] Schulz C., Hasle G., Kloster O., Riise A., Smedsrud M.: Parallel local search for the CVRP on the GPU. Talk at The 3rd International Conference on Metaheuristics and Nature Inspired Computing (META'10), Djerba, Tunisia, October 28 2010.

[14] Talbi E-G (ed): Parallel Combinatorial Optimization. Wiley 2006. ISBN-13 978-0-471-72101-7.