BLUE WATERS GENERAL ALLOCATION PROJECT REPORT

1 Project Information

Project Title: Parallel algorithms for solving large assignment problems

PI: Rakesh Nagi, Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign.

Co-PI: Ketan Date, Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign.

Corresponding Author: Rakesh Nagi, Email: nagi@illinois.edu.

2 Executive Summary

The goal of our project is to develop fast and scalable algorithms for solving large instances of Linear Assignment Problem (LAP) and Quadratic Assignment Problem (QAP) using Graphics Processing Units (GPU). LAP is polynomial-time solvable with cubic worst case complexity, while the QAP is strongly NP-Hard. To solve a linearized model of the QAP using branch-and-bound, lower bounds must be calculated using the Lagrangian dual technique, in which a large number of LAPs must be solved efficiently. Additionally, in a branch-and-bound scheme, a large number of nodes must be explored in order to find a provable optimal solution. To this end, we have used Blue Waters to develop: (1) A GPU-accelerated Hungarian algorithm for solving large LAPs in an efficient manner; (2) A GPU-accelerated Lagrangian dual ascent heuristic for obtaining lower bounds on the QAP. These algorithms will be used in parallel branch-and-bound scheme to solve large QAPs to optimality.

3 Description of Research Activities and Results

3.1 Key Challenges

Assignment Problems are fundamental to the discovery in diverse branches of science and engineering. Some of their applications include information fusion, protein-protein interaction analysis, facilities design, vehicle routing and resource scheduling. To gain meaningful insights, many applications demand quick solutions to large instances of Assignment Problems containing hundreds of thousands of vertices. This makes it incredibly challenging for the sequential algorithms designed for a single processor. Therefore, designing fast and scalable algorithms suitable for the state-of-the-art parallel programming architectures is essential. In this research, we intend to propose novel parallel algorithms for the Compute Unified Device Architecture (CUDA) enabled NVIDIA Graphics Processing Units (GPUs), to solve the following two Assignment Problems.

Linear Assignment Problem. The objective of the Linear Assignment Problem (LAP) is to assign n resources to n tasks such that the total cost of the assignment is minimized. The mathematical formulation for the LAP is:

$$\min \quad \sum_{i=1}^{n}\sum_{p=1}^{n} b_{ip}x_{ip}; \tag{1}$$

$$\text{s.t.} \sum_{i=1}^{n} x_{ip} = 1, \forall p; \quad \sum_{p=1}^{n} x_{ip} = 1, \forall i; \quad x_{ip} \in \{0,1\}, \forall i, p. \tag{2}$$

The decision variable $x_{ip} = 1$, if resource i is assigned to task p and 0 otherwise. The constraints enforce that each resource should be assigned to exactly one task and each task should be assigned to exactly one resource. b_{ip} is the cost of assigning resource i to task p.

LAP is one of the most well-studied optimization problems that can be solved in polynomial time. Until now, many efficient sequential algorithms have been proposed in the literature, such as the famous Hungarian algorithm (Kuhn, 1955), the Auction algorithm (Bertsekas, 1990), and the shortest path algorithm (Jonker and Volgenant, 1987). The theoretical complexity of the most efficient implementation of the Hungarian or the shortest path algorithm is $O(n^3)$.

Owing to their cubic worst-case complexity, sequential algorithms can prove to be a significant bottleneck, for solving large instances of the LAP. Therefore, a parallel algorithm is required, which can take advantage of a specific architecture and divide the work among multiple processors to alleviate the computational burden. We chose to parallelize the famous Hungarian algorithm (Kuhn, 1955, Munkres, 1957) on a GPU, which has theoretical complexity is $O(n^3)$.

Quadratic Assignment Problem. The Quadratic Assignment Problem (QAP) is one of the oldest mathematical problems in the literature and it has received substantial attention from the researchers around the world. QAP was originally introduced by Koopmans and Beckmann (1957) as a mathematical model to locate indivisible economical activities (such as facilities) on a set of locations and the cost of the assignment is a function of both distance and flow. The objective is to assign each facility to a location so as to minimize a quadratic cost function. The generalized mathematical formulation for the QAP, given by Lawler (1963), can be written as follows:

$$\min \quad \sum_{i=1}^{n}\sum_{p=1}^{n} b_{ip}x_{ip} + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{p=1}^{n}\sum_{q=1}^{n} f_{ij}d_{pq}x_{ip}x_{jq}; \tag{3}$$

$$\text{s.t.} \sum_{i=1}^{n} x_{ip} = 1, \forall p; \quad \sum_{p=1}^{n} x_{ip} = 1, \forall i; \quad x_{ip} \in \{0,1\}, \forall i, p. \tag{4}$$

Here, $x_{ip} = 1$ if facility i is assigned to location p and 0 otherwise. The constraints enforce that each facility should go to exactly one location and each location should have exactly one facility.

Despite having the same constraint set as the LAP, the QAP is a strongly NP-hard problem (Sahni and Gonzalez, 1976), i.e., it cannot be solved efficiently within a guaranteed time limit. Additionally, it is difficult to find a provable ϵ-optimal solution to QAP. The quadratic nature of the objective function also adds to the solution complexity. One of the ways of solving the QAP is to convert it into a Mixed Integer Linear Program (MILP) by introducing additional variables and constraints. Different linearizations were proposed by Lawler (1963), Kaufman and Broeckx (1978), Frieze and Yadegar (1983), Adams and Johnson (1994), Adams et al. (2007), etc. We chose to parallelize the Lagrangian dual ascent algorithm for Level-2 Refactorization-Linearization Technique (RLT2) proposed by Adams et al. (2007), in which we need to solve $O(n^4)$ LAPs and adjust $O(n^6)$ Lagrange multipliers to obtain a strong lower bound on the QAP.

3.2 Why It Matters

Both LAP and QAP have many theoretical and practical applications. LAP appears as an inner problem in many of the NP-hard problems, such as Traveling Salesman Problem (TSP), Vehicle Routing Problem (VRP), Data Association (DA), etc. Therefore, having a fast, scalable, and cost effective LAP solver is extremely important for many theoretical and practical problems. QAP may serve as a specialization to many "harder" optimization problems, such as the Graph Association (GA), Vehicle Routing Problem (VRP), etc., in alternative formulations. Therefore, to solve these problems efficiently, we need to solve the QAP subproblems efficiently. As a result, a fast and scalable QAP solver coupled with a fast LAP solver can be a powerful tool for researchers working on such NP-hard problems.

Since we are addressing the two fundamental Assignment Problems, the theory and algorithms developed in this research can be potentially extended to other problems from this class. Therefore, this research can be seen as foundational work in parallel/accelerated algorithms for a broad class of Assignment Problems. Since the Assignment Problems are pervasive in science and engineering applications, these efficient algorithms will be extremely valuable to the researchers dealing with massive instances of these problems, which will pave the way to new breakthroughs in science and engineering through the analysis of "big data."

3.3 Why Blue Waters

In a typical branch-and-bound tree, we need to explore a large number of nodes in order to find an optimal solution. Also, as the problem size grows, the number of nodes that need to be explored grows exponentially. Therefore, we need a large number of processors which can explore the solution space in parallel. Additionally, the GPU-accelerated dual ascent procedure benefits from the large number of powerful GPU-enabled processors available at the Blue Waters facility. Coupling the parallel branch-and-bound with the fast GPU-based lower bounding techniques will enable us to solve large-sized problems from the QAPLIB (Burkard et al., 1997), which still remain unsolved.

3.4 Accomplishments

Results for LAP. In our recent paper (Date and Nagi, 2016), we developed parallel versions of the Hungarian algorithm, specifically for CUDA-enabled NVIDIA GPUs. In this paper, we tested two versions of the Hungarian algorithm: (1) The classical Kuhn-Munkres (Munkres, 1957) version (CUDA-CL), which has a complexity of $O(n^4)$; and (2) The alternating tree version (Lawler, 1976, Papadimitriou and Steiglitz, 1998) (CUDA-TR), which has a complexity of $O(n^3)$. The main contribution of our work is an efficient parallelization of the augmenting path search phase of the Hungarian algorithm, which is the most time consuming phase. In our accelerated algorithm, multiple CUDA threads jointly search for augmenting paths from all the unassigned rows, using parallel breadth-first-search (P-BFS). Our algorithm finds more than one assignments in each iteration, and therefore, converges to the optimal solution in fewer number of iterations (as seen in Table 1). We tested both our accelerated algorithms on randomly generated problem instances and compared the execution times with those of the sequential algorithm and an OpenMP version executed on the CPU. The computational results for small-scale problems are shown in Fig. 1. The computational results for large-scale problems are shown in Fig. 2. These results clearly show that as the problem size grows, the parallel algorithms become superior to the sequential algorithm. Also, the alternating tree version is the most efficient version for dense cost matrices which have fewer zero-cost elements per row/column. Therefore, this version is best suited for LAPs with non-integer cost matrices.

Table 1: Number of assignments found during different stages for CU-TREE

n	**Initial Assignments**	**Assignments in Iterations**						
		[1-5]	**[6-10]**	**[11-15]**	**[16-20]**	**[21-25]**	**[26-30]**	**[31-35]**
1000	866	96	26	6	6			
2000	1745	177	47	18	10	3		
3000	2608	203	143	24	10	6	6	
4000	3473	254	184	52	21	11	4	1
5000	4316	354	231	61	10	22	4	2

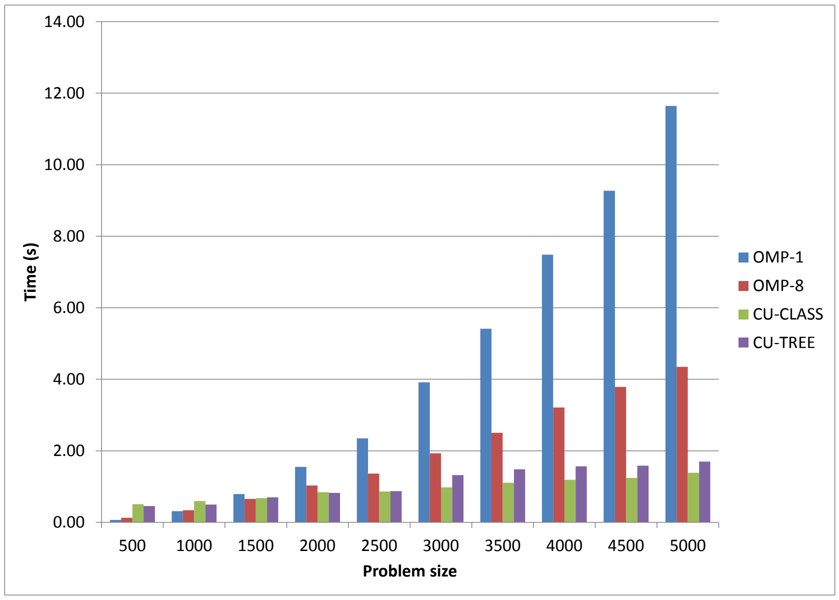

Figure 1: Execution time for small problems with cost range $[0, n]$

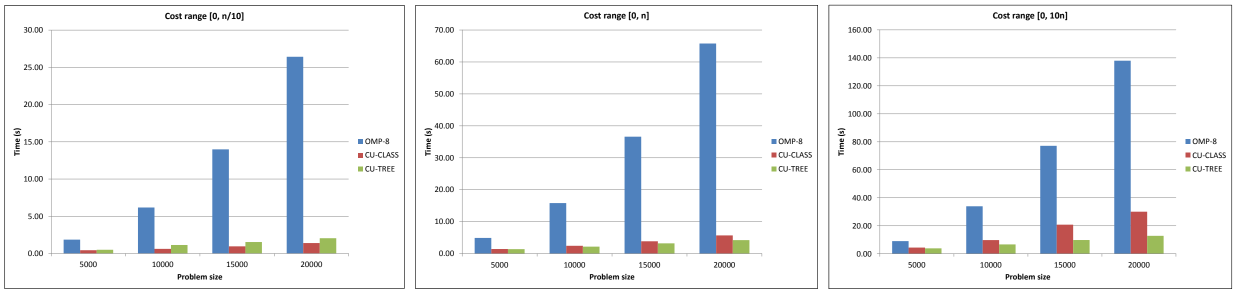

Figure 2: Execution time (s) for large problems with different cost ranges

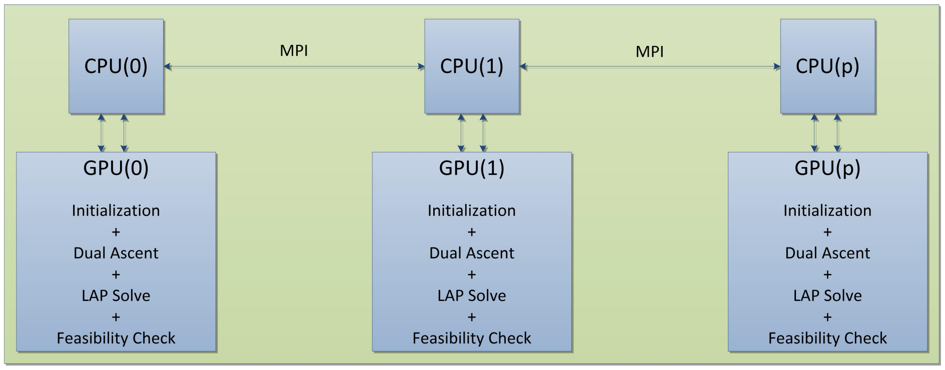
CPU(0)
MPI
CPU(1)
MPI
CPU(p)
GPU(0)
Initialization + Dual Ascent + LAP Solve + Feasibility Check
GPU(1)
Initialization + Dual Ascent + LAP Solve + Feasibility Check
GPU(p)
Initialization + Dual Ascent + LAP Solve + Feasibility Check

Figure 3: Architecture for parallel/accelerated dual ascent

Table 2: Lower bound strength for RLT2 dual ascent – 2000 iterations

Problem	**LAP Counts (X, Y, Z)**	**# of GPUs**	**LB**	**OPT**	**% GAP**	**Itn time (s)**
Nug18	(1, 324, 93636)	1	1909.29	1930	1.08	5.07
Nug20	(1, 400, 144400)	1	2511.79	2570	2.32	10.12
Nug22	(1, 484, 213444)	2	2603.84	2650	1.77	10.64
Nug25	(1, 625, 360000)	4	3582.83	3744	4.50	12.23
Nug30	(1, 900, 518400)	15	5755.35	6124	6.41	12.86

Results for QAP. We designed a parallel Lagrangian dual ascent heuristic for solving RLT2 using hybrid MPI+CUDA architecture (as seen in Fig. 3). The $O(n^4)$ LAPs are split across these GPUs and solved using our GPU-accelerated Hungarian algorithm, while the $O(n^6)$ Lagrange multipliers are updated by multiple CUDA threads in parallel.

We tested the GPU-accelerated Dual Ascent for RLT2, coupled with the GPU-accelerated Hungarian algorithm on the various Nugent problem sets (Nug18, Nug20, Nug22, Nug25, and Nug30) from the QAPLIB. As shown in Table 2, our parallel Dual Ascent implementation provides strong lower bounds when tested on up to 15 GPUs from Blue Waters. We expect to see similar scaling behavior for large problems with $30 < n \leq 40$.

In the branch-and-bound scheme, the lower bounds mentioned above proved to be extremely valuable. We tested the branch-and-bound algorithm on the "medium-sized" instances (Nug18, Nug20, Nug22, and Nug25). These tests looked very promising (see Table 3), since all these problems were solved within two hours. The larger instances Nug25 and Nug30 have been previously solved to optimality by other researchers, however, they are extremely challenging and require intense computational effort. We will be attempting to solve these (and other larger) problems and we believe that our current approach will prove to be extremely efficient.

3.5 Next Generation Work

Our ultimate objective through this research is to provide efficient solution methods for a class of Assignment Problems. These problems arise in diverse branches of science and engineering and they are part of many cutting edge projects with high socio-economic impact. Using our methods, we hope that scientists and engineers will be able to solve Assignment Problems containing hundreds of thousands of vertices within a matter of minutes, leading to transformative discoveries. We intend to package the sophisticated algorithms stemming from this research into a programming

Table 3: Branch-and-bound results on Nugent problems

Problem	OPT	PE Banks	PEs per bank	Nodes	Total Time (min)	PE Bank Utilization		
						Min	Avg	Max
Nug18	1930	18	1	306	9.26	0.524	0.805	0.986
Nug20[†]	2570	20	1	216	56.51	0.155	0.549	0.994
Nug22	2650	22	2	462	105.05	0.651	0.729	0.998
Nug25[†]	3744	200	4	19419	436.12	0.688	0.864	0.975

[†]Symmetry elimination rules were used for these problem instances.

library, which can be used by scientists and engineers across the world, to not only solve large-scale Assignment Problems, but to analyze and compare the performance of different algorithms, thereby ensuring continued advancement of science and engineering.

4 List of Publications, Data Sets Associated with This Work

1. Date, K. and Nagi, R. (2016). GPU-accelerated Hungarian algorithms for the Linear Assignment Problem. *Parallel Computing*, 57:52–72. `http://dx.doi.org/10.1016/j.parco.2016.05.012`.

2. Date, K. and Nagi, R. (2017). Exact algorithms for large Quadratic Assignment Problems on Graphics Processing Unit clusters. *To be submitted to INFORMS Journal on Computing.*

References

Adams, W. P., Guignard, M., Hahn, P. M., and Hightower, W. L. (2007). A level-2 reformulation–linearization technique bound for the quadratic assignment problem. *European Journal of Operational Research*, 180(3):983–996.

Adams, W. P. and Johnson, T. A. (1994). Improved linear programming-based lower bounds for the quadratic assignment problem. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 16:43–77.

Bertsekas, D. P. (1990). The Auction algorithm for assignment and other network flow problems: A tutorial. *Interfaces*, 20(4):133–149.

Burkard, R. E., Karisch, S. E., and Rendl, F. (1997). Qaplib–a quadratic assignment problem library. *Journal of Global Optimization*, 10(4):391–403.

Date, K. and Nagi, R. (2016). GPU-accelerated Hungarian algorithms for the Linear Assignment Problem. *Parallel Computing*, 57:52–72.

Date, K. and Nagi, R. (2017). Exact algorithms for large Quadratic Assignment Problems on Graphics Processing Unit clusters. *To be submitted to INFORMS Journal on Computing.*

Frieze, A. and Yadegar, J. (1983). On the quadratic assignment problem. *Discrete applied mathematics*, 5(1):89–98.

Jonker, R. and Volgenant, A. (1987). A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340.

Kaufman, L. and Broeckx, F. (1978). An algorithm for the quadratic assignment problem using bender's decomposition. *European Journal of Operational Research*, 2(3):207–211.

Koopmans, T. C. and Beckmann, M. (1957). Assignment problems and the location of economic activities. *Econometrica: Journal of the Econometric Society*, pages 53–76.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Lawler, E. L. (1963). The quadratic assignment problem. *Management science*, 9(4):586–599.

Lawler, E. L. (1976). *Combinatorial optimization: networks and matroids.* Courier Corporation.

Munkres, J. (1957). Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial & Applied Mathematics*, 5(1):32–38.

Papadimitriou, C. H. and Steiglitz, K. (1998). *Combinatorial optimization: algorithms and complexity.* Courier Corporation.

Sahni, S. and Gonzalez, T. (1976). P-complete approximation problems. *Journal of the ACM (JACM)*, 23(3):555–565.