Parallel Algorithms for Hard Combinatorial Optimisation Problems in Multi-Agent Systems

(Doctoral Consortium)

Filippo Bistaffa
University of Verona
filippo.bistaffa@univr.it

Categories and Subject Descriptors

D.1.3 [**Software**]: Programming Techniques—*Concurrent Programming (Parallel programming)*

Keywords

Multi-Core, GPGPU, Coalition Formation

ABSTRACT

Multi-agent systems represent a powerful tool to model several interesting real-world problems. Unfortunately, the limited scalability of many state-of-the-art algorithms hinders their applicability in practical situations: in fact, complex dynamics and interactions among a large number of agents often make the search for an optimal solution an unfeasible task. Against this background, the study and design of new highly parallel computational models could greatly improve solution techniques in the above mentioned fields. In particular, I will introduce two parallel approaches to the *coalition formation* problem in the context of multi-agent systems, detailing how their performances can benefit from the use of modern parallel architectures.

1. INTRODUCTION

Multi-agent systems represent a powerful approach to the description and solution of many practical problems, with several applications such as coordinated defence systems, smart grid electricity networks, transportation, logistics and sensor networks. Typical multi-agent systems are characterised by complex dynamics and interactions among a large number of agents, which usually translate into hard combinatorial problems, posing significant challenges from the computational point of view. Hence, the study of innovative techniques which can deal with such computational burden is fundamental and would give a strong contribution to the research in this scenarios. In particular, my research focuses on highly parallel computational models (e.g., GPGPUs or *many-core* CPUs), in order to improve current solution techniques by means of new algorithmic approaches that take advantage of such novel hardware architectures.

Appears in: *Alessio Lomuscio, Paul Scerri, Ana Bazzan, and Michael Huhns (eds.), Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), May 5-9, 2014, Paris, France.*

It is important to note that several interesting problems in this field (e.g., coalition formation, winner determination problem in combinatorial auctions, etc.) exhibit a common structure and many similar features, hence the above mentioned innovations could be applied to many different approaches in multi-agent systems. In particular, coalition formation is one of the fundamental methods for establishing collaborations among agents, each with individual objectives and properties, which, unfortunately, requires an exponential amount of time to be solved to optimality. Coalition formation is typically used to model the cooperation among fully cooperative agents (e.g., rescue teams or sensor networks) or among selfish agents that are only interested in their own utility, to help customers in the context of group buying or energy users in the collective energy purchasing scenario. In particular, in the latter scenario each agent is characterised by an energy consumption profile that represents its energy consumption throughout a day [3]. More precisely, each coalition of agents is associated to the total cost that the group would incur if they buy energy as a collective on two different markets: the spot market, a short term market intended for small amounts of energy, and the forward market, a long term one in which bigger portions of energy can be bought at cheaper prices. The above mentioned scenario, together with several other real-world applications, is characterised by sparse synergies between the agents which may constrain the formation of some coalitions [2, 5]. These constraints may be due to communication infrastructures (e.g., non-overlapping communication loci or energy limitations for sending messages across a network), social or trust relationships (e.g., energy consumers who prefer to group with their friends and relatives in forming energy cooperatives), or physical constraints (e.g., emergency responders that have enough fuel to join only specific teams).

In what follows, I will describe the work of a recent paper [2], in which a new algorithm to solve coalition structure generation (CSG) on synergy graphs (namely, the CFSS algorithm) has been proposed, detailing how this new solution technique could benefit from the computational power provided by the above mentioned multi-core architectures.

2. PARALLEL CFSS

The CFSS algorithm [2] is based on a novel representation of the CSG problem on synergy graphs, which, by means of a series of edge contractions on the constraint graph, allows us to build a search tree where each node corresponds to a feasible coalition structure, while avoiding redundancy.

(a) Before contraction (b) After contraction

Figure 1: Example of an edge contraction in a triangle graph (the dashed edge is contracted to give the graph on the right)

Figure 1 shows the contraction of the edge $(\{A\}, \{C\})$, which results in a new vertex $\{A, C\}$ connected to vertex $\{B\}$. Intuitively, a contraction of an edge represents the merging of the coalitions associated to its incident vertices. This new model improves the state-of-the-art with several contributions: in fact, its scalability benefits from its polynomial memory requirements, which allow to provide approximate solutions with quality guarantees for systems with thousands of agents. Furthermore, the search space can be partitioned avoiding redundancy, hence enabling a parallel solving algorithm. In [2] we proposed a simple multi-core implementation of the CFSS algorithm, obtained by having different threads to search different branches of the search tree. The only required synchronisation point is the computation of the current best solution that must be read and updated by every thread. In particular, the distribution of the computational burden is done by assigning each subtree rooted in every node of the first generation to an arbitrary number of threads. This approach allows to obtain a speedup higher than 7× on a 12 cores machine, but its effectiveness relies on the determination of some parameters governing the partition of the search tree among the threads, which, in our work, has been done by a manual tuning process.[1] More advanced techniques found in literature, such as estimating the number of nodes in the search tree as suggested in [4], could provide a more robust implementation able to adapt and perform well on all possible instances. Moreover, the scalability of CFSS could benefit by the adoption of modern highly parallel computational architectures (i.e., GPGPUs with dynamic parallelism), allowing us to optimally solve instances one order of magnitude bigger than the current limit (60 agents).

In general, GPGPUs represent a great opportunity to improve state-of-the-art algorithms in multi-agent system: in fact, many interesting scenarios are characterised by computationally hard problems, whose search space is exponential in the number of agents. The next section will describe an example of application in which this concept can be applied, taking advantage of structure of the problem to exploit the massive amount of computational power provided by CUDA architectures.

3. FUTURE WORK: GPGPU

One of the most promising open research lines in the above mentioned field includes the CUDA implementation of the joint sum operator, used by many GDL-based algorithms [1], which would allow a speedup in various solution techniques.

[1]After an empirical analysis, we verified that the distribution of the nodes over the search tree does not vary significantly among different instances.

Unfortunately, the copious amount of memory and time needed for the joint sum operation limits the instances that can be solved to systems of tens of agents. Thus, the reduced scalability of such approach hinders its applicability to real-world situations, especially due to the significant memory requirements of algorithms like dynamic programming of belief propagation.

Some previous works in the scientific literature focus on the parallel implementation of such operation: in particular, Zheng and Mengshoel [6] propose a multi-threaded algorithm to compute messages required by belief propagation in Bayesian networks. Authors devise a parallelisation model in which every row of the separator table is assigned to one thread, whose task is to retrieve the corresponding input rows and compute their outputs.

More in general, one of the fundamental challenges in the design of an efficient massively parallel solution technique is to structure the entire algorithmic approach in order to take advantage of the specific features of GPGPUs in terms of hardware architecture: achieving optimal memory accesses and obtaining an adequate computational throughput are not trivial tasks, but if done correctly, the performance and scalability improvements can be overwhelming.

In this sense, GPGPUs can be seen as an "enabling technology" for multi-agent systems: the above mentioned advantages, together with new computational models such as dynamic parallelism[2] (as an implementation of the decentralised nature of agents), could represent a significant contribution to several real-world applications of multi-agent systems.

4. REFERENCES

[1] S. Aji and R. McEliece. The generalized distributive law. *Information Theory, IEEE Transactions on*, 46(2):325–343, 2000.

[2] F. Bistaffa, A. Farinelli, J. Cerquides, J. A. Rodríguez-Aguilar, and S. D. Ramchurn. Anytime coalition structure generation on synergy graphs. *Accepted at: International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2014.

[3] F. Bistaffa, A. Farinelli, M. Vinyals, and A. Rogers. Decentralised stable coalition formation among energy consumers in the smart grid (demonstration). In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 1461–1462, 2012.

[4] L. H. S. Lelis, L. Otten, and R. Dechter. Predicting the size of depth-first branch and bound search trees. In *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2013.

[5] T. Voice, S. Ramchurn, and N. Jennings. On coalition formation with sparse synergies. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 223–230, 2012.

[6] L. Zheng, O. J. Mengshoel, and J. Chong. Belief propagation by message passing in junction trees: Computing each message faster using gpu parallelization. *CoRR*, abs/1202.3777, 2012.

[2]Dynamic parallelism enables a CUDA thread to create and synchronise new nested work, using the CUDA runtime API to launch other kernels, optionally synchronise on thread completion, perform device memory management, and create and use streams and events, all without CPU involvement.