Paving the way for Large-Scale Combinatorial Auctions

(Extended Abstract)

Francisco Cruz-Mencia
IIIA-CSIC, CAOS-UAB
Campus de la UAB, s/n
Bellaterra, Barcelona
fcruz@iiia.csic.es

Jesus Cerquides
IIIA-CSIC
Campus de la UAB, s/n
Bellaterra, Barcelona
cerquide@iiia.csic.es

Antonio Espinosa
CAOS-UAB
Campus de la UAB, s/n
Bellaterra, Barcelona
antoniomiguel.espinosa@uab.cat

Juan Carlos Moure
CAOS-UAB
Campus de la UAB, s/n
Bellaterra, Barcelona
juancarlos.moure@uab.es

Juan Antonio Rodriguez-Aguilar
IIIA-CSIC. Campus UAB, s/n
Bellaterra, Barcelona
jar@iiia.csic.es

ABSTRACT

The Winner Determination Problem (WDP) in Combinatorial Auctions comes up in a wide range of applications. Linear Programming (LP) relaxations are a standard method for approximating combinatorial optimisation problems. In this paper we propose how to encode the WDP so that it can be approximated with AD^3. Moreover, we contribute with $PAR\text{-}AD^3$, the first parallel implementation of AD^3. We show that while AD^3 is up to 4.6 times faster than CPLEX in a single-thread execution, $PAR\text{-}AD^3$ is up to 23 times faster than parallel CPLEX in an 8-core architecture.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Algorithms, Performance

Keywords

Combinatorial auctions; Large-scale optimization; Linear programming

1. INTRODUCTION

Combinatorial auctions (CAs) have been largely studied because of its wide applicability to many real-life scenarios. However, although such application domains claim to be large-scale, namely involving thousands of bids, current results indicate that the scale of the CAs that can be optimally solved is small [3].

Linear Programming (LP) relaxations are a standard method for approximating combinatorial optimisation problems in computer science. [6] report that realistic problems with a large number of variables cannot be solved by off-the-shelf, commercial LP solvers.

In order to solve LP relaxations, there has been a recent upsurge of interest in the Alternating Direction Method of Multipliers (ADMM) which was invented in the 1970s [2, 1]. ADMM is specially well suited for application in a wide variety of large-scale distributed modern problems. Along this line, [5] proposes AD^3, a novel algorithm based on ADMM, which proves to outperform off-the-shelf, commercial LP solvers for problems including declarative constraints.

Against this background, we make the following contributions: a) We show how to encode the WDP for CAs so that it can be approximated by AD^3. b) We propose an optimised, parallel implementation of AD^3, the so-called $PAR\text{-}AD^3$. c) We show that while AD^3 is up to 13 times faster than CPLEX in a single-thread execution, $PAR\text{-}AD^3$ is up to 23 times faster than parallel CPLEX in an 8-core architecture in hard instances.

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May 4–8, 2015, Istanbul, Turkey.*

2. SOLVING COMBINATORIAL AUCTIONS WITH AD^3

Graphical models are widely used in computer vision, natural language processing and computational biology, where a fundamental problem is to find the maximum a posteriori probability (MAP) given a factor graph. Linear Programming (LP) relaxations have been extensively applied to approximate the MAP for graphical models.

A Combinatorial Auction (CA) is an auction in which bidders can place bids for a combination of items instead of individual ones. In this scenario, one of the fundamental problems is the Winner Determination Problem (WDP), which consists in finding the set of bids that maximise the auctioneer's benefit. Notice that the WDP is an $\mathcal{NP}$-complete problem.

Although special-purpose algorithms have addressed the WDP, the state-of-the-art method for solving a WDP is to encode it as an integer linear program (ILP) and solve it using an off-the-shelf commercial solver (such as CPLEX or Gurobi).

Solutions to an LP relaxation can provide a very effective start to finding a good feasible solution to the non-relaxed optimisation problem. Hereafter we focus on solving the

LP relaxation of the WDP by means of AD^3. Since AD^3 requires a factor graph to operate, we first need to encode the WDP as a factor graph.

3. PARALLEL REALISATION OF AD^3

We propose a rearrangement of the algorithm targeting multicore computer architectures, *PAR-AD*^{3}, that eases the exploitation of the exposed parallelism at two dimensions: thread-level and data-level. For that, we reorganize both the data structures layout and the order of operations to promote parallel scaling (thread parallelism) and vectorisation (data parallelism). This approach is generalisable to other similar graph processing algorithms.

We present an efficient realisation of the message-passing algorithmic pattern using shared variables. The key contributions of our design are: a) An edge-centric representation of the shared variables that improves memory access performance. b) A reorganisation of the operations that promotes parallel scaling and vectorising.

4. EMPIRICAL EVALUATION

In this section, we assess *PAR-AD*^{3} performance against the state-of-the-art optimisation software CPLEX

Experiment setup. In order to generate CA WDP instances, we employ CATS [4]. Each instance is generated out of the following list of distributions thoroughly described in [3]: arbitrary, matching, paths, regions, scheduling, L1, L3, L4, L5, L6, L7. We discarded to employ the L2 distribution because the CATS generator is not capable of generating large instances. While the first five distributions were designed to generate realistic CA WDP instances, the latter ones generate artificial instances. Both AD^3 and *PAR-AD*^{3} are well suited for large-scale hard problems. We considered a number of goods within $[10^3, 10^4]$ in steps of 10^3 goods. ,The number of bids are ranged within $[10^4, 4{\cdot}10^4]$ in steps of 10^4 bids. Each problem scenario is characterised by a combination of distribution, number of goods, and number of bids. Our experiments consider 5 different instances for each problem scenario. Experiments are executed in a computer with two four-core Intel Xeon Processors L5520 @2.27GHz with 32 GB RAM with the hyper-threading mechanism off.

Distributions hardness. We empirically determine the hardness of the relaxation for our experimental data by solving the LP using CPLEX simplex (simplex henceforth), CPLEX barrier (barrier henceforth). Scheduling, matching, L1 and L4 are very well addressed by simplex, where solving time is, in general, less than one second. Both AD^3 and *PAR-AD*^{3} are not competitive in this scenario.

Single-thread analysis. After comparing the publicly-available (sequential) version of AD^3 against the single-thread execution of *PAR-AD*^{3}, we observed that *PAR-AD*^{3} outperformed AD^3, reaching a peak speedup of 12.4X. Next, we compared the single-thread average performance of *PAR-AD*^{3} against simplex and barrier. In general, the larger the WDP instances, the larger the *PAR-AD*^{3} benefits. Single-threaded *PAR-AD*^{3} reaches a peak speedup of 13X for the hardest distribution when compared to barrier, the best of the two state-of-the-art solvers.

Multi-thread analysis. We have run *PAR-AD*^{3}, simplex and barrier with 8 parallel threads each. We observe that *PAR-AD*^{3} outperforms simplex and barrier in many more scenarios than in the single thread execution. We infer that *PAR-AD*^{3} better benefits from parallelisation than simplex and barrier. Our peak speedup is 23X.

5. CONCLUSIONS AND FUTURE WORK

We have proposed a novel approach to solve the LP relaxation for the WDP in CAs. Our approach encodes the optimisation problem as a factor graph and uses AD^3 to efficiently find the solution.

In order to achieve efficiency, we have identified some of the bottlenecks in message-passing graph based algorithms and proposed some techniques to achieve good performance and scalability, in particular when executing in parallel. We propose *PAR-AD*^{3}: an optimised and parallel version of AD^3.

Our experimental results validate *PAR-AD*^{3} efficiency gains in large scale scenarios. We have shown that *PAR-AD*^{3} performs better than CPLEX for large-scale CAs in the computationally hardest distributions, both in single- and multi-threaded scenarios, with a peak speedup of 23X. Furthermore, the speedup is larger in multi-threaded scenarios, showing that *PAR-AD*^{3} scales better with hardware than CPLEX. Presented results pave the way for facing the WDP in large scale CAs, either optimally by integrating *PAR-AD*^{3} in a MIP solver or approximately by tightening the LP relaxation towards an exact approximated solution. As future work, we note that *PAR-AD*^{3} characteristics are well suited for accelerators, such as GPUs.

6. ACKNOWLEDGMENTS

This research has been supported by MICINN-Spain under contracts TIN2011-28689-C02-01, TIN2013-45732-C4-4-P and TIN2012-38876-C02-01.

REFERENCES

[1] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics with Applications*, 2(1):17–40, 1976.

[2] R. Glowinski and A. Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires. *ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique*, 9(R2):41–76, 1975.

[3] K. Leyton-Brown, E. Nudelman, and Y. Shoham. Empirical hardness models: Methodology and a case study on combinatorial auctions. *Journal of the ACM (JACM)*, 56(4):22, 2009.

[4] K. Leyton-Brown, M. Pearson, and Y. Shoham. Towards a universal test suite for combinatorial auction algorithms. In *Proceedings of the 2nd ACM conference on Electronic commerce*, pages 66–76. ACM, 2000.

[5] A. F. T. Martins, M. A. T. Figueiredo, P. M. Q. Aguiar, N. A. Smith, and E. P. Xing. Ad^3: Alternating directions dual decomposition for map inference in graphical models. *Journal of Machine Learning Research*, 46, 2014. to appear.

[6] C. Yanover, T. Meltzer, and Y. Weiss. Linear programming relaxations and belief propagation – an empirical study. *J. Mach. Learn. Res.*, 7:1887–1907, Dec. 2006.