Adaptive Multi-agent System for Situated Task Allocation

Quentin Baert, Anne-Cécile Caron, Maxime Morge, Jean-Christophe Routier, Kostas Stathis

Adaptive Multi-agent System for Situated Task Allocation

Extended Abstract

Quentin Baert, Anne-Cécile Caron, Maxime Morge, Jean-Christophe Routier

Univ. Lille, CNRS, Centrale Lille, UMR 9189 - CRIStAL - Centre de Recherche en Informatique Signal et Automatique de Lille

{quentin.baert,anne-cecile.caron,maxime.morge, jean-christophe.routier}@univ-lille.fr

Kostas Stathis

Department of Computer Science, Royal Holloway University of London

kostas.stathis@rhul.ac.uk

KEYWORDS

Distributed problem solving; Bargaining and negotiation

1 INTRODUCTION

Multi-agent scheduling [1, 2, 4, 8, 10–17, 20, 22–25] has received significant attention in tackling the problem of load balancing and task allocation in distributed systems. Apart from dividing the work through decentralization, we consider dynamicity because allocation of tasks must be concurrent with their execution, and adaptation because tasks must be reallocated when a disruptive event is performed. We will assume that agents are fully distributed and cooperative in order to optimize the global runtime, i.e a system-centric metric rather than user-centric metrics [10–13, 20, 24]. We will also assume that a task can be performed by any single agent without preemption and precedence order. Moreover, tasks have no deadlines, are indivisible and not shareable.

We follow a market-based approach [10–13, 20, 24] to tackle the multi-agent situated task allocation problem. In order to improve load balancing, agents adopt a locality-based strategy in concurrent one-to-many negotiations for task delegations. The task reallocation is dynamic since the negotiation process is iterated and concurrent with the tasks processing. Moreover, the system is adaptive to disruptive events, e.g. task consumptions. As a practical application, we consider the distributed deployment of the MapReduce design pattern for processing large datasets [9]. Our preliminary empirical results show that, for such an application, the locality-based strategy improves the runtime.

2 NEGOTIATION PROCESS

In a multi-agent situated task allocation problem, tasks have different costs (runtime) for different agents due to the resource locality. A task allocation is evaluated by considering the maximum completion time (makespan). In order to locally improve the task allocation, agents consider socially rational task delegations that strictly decrease the local makespan until a stable allocation is reached.

Decentralized task delegation process. Agents operate in concurrent, one-to-many and single-round negotiations for task delegations. Each negotiation, which is based on the Contract Net Protocol [21], is divided in three steps: 1) the choice of the task to negotiate by the strategy of the initiator; 2) the refusals/bids from the peers based on the social rationality of the task delegation; 3) the selection of the winning bid by the initiator, e.g. the bidder with the smallest workload. Several negotiations may concurrently occur as in [3] and the responsiveness of the task reallocation can be improved [5]. Even if the computation of the local makespan by the initiator of a negotiation is based on its beliefs, that can be inaccurate, a successful negotiation can only reach a socially rational task delegation, and so it tends to improve the makespan.

Concurrent consumptions and delegations. Task delegations and task consumptions are concurrent. Fig. 1 represents their impact on the allocation until all of the tasks are performed ($\mathcal{T}_{\text{final}} = \emptyset$). Starting from the initial allocation P_0, agents perform socially rational task delegations to improve the makespan (e.g. the path from P_0 to P_k) until a task consumption (e.g. the edge from P_k to P'_0), which eventually interrupts the path toward a stable allocation (e.g. the path from P_k to P_{stable} shown in gray). A task consumption may occur when the agents have reached a stable allocation (e.g. P'_{stable}) or not (e.g. P_k). Even if task delegations and task consumptions are concurrent and the two of them tend to decrease the makespan, these processes are complementary since a task removal may allow new socially rational task delegations.

Locality-based strategy. According to a trivial strategy, an agent can perform the largest task in its bundle and negotiate the smallest one in the set of potential socially rational delegations. By contrast, we build the strategy upon the locality since resource fetching consumes time. Intuitively, an agent should perform first the tasks which may cost more for its peers and it should negotiate first the tasks which may cost less for its peers. For this purpose, we define the local availability ratio of an agent for a task as the ratio between the number of local resources for this task with respect to the agent and the total number of resources for the task. According to our strategy, an agent performs first the large local tasks and it negotiates first the large distant ones based on its local beliefs and knowledge, i.e. the local availability ratios and the task costs. This

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

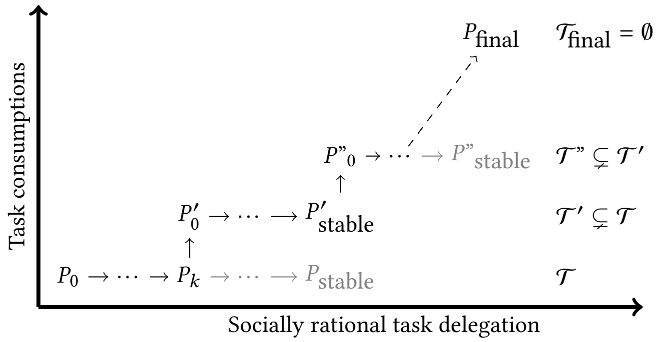

Figure 1: Concurrent task consumptions (vertical edges) and task delegations (horizontal edges).

strategy is built on a composite task bundle, called locality-based bundle which is divided in three subbundles as follows.

- *The maximum locality bundle* contains the tasks s.t. agent owns at least one resource and there is no other agent which owns more resources for this task. The tasks are sorted in decreasing order of cost.
- *The intermediate locality bundle* contains the tasks which are partially local. The tasks are sorted in decreasing order of local availability ratio and the tasks with the same local availability ratio are sorted in decreasing order of cost.
- *The distant bundle* contains the tasks which are distant. The tasks are sorted in increasing order of cost.

When an agent looks for a task to perform, it starts from the top of the maximum locality bundle, i.e. the largest local task. When an agent looks for a task to negotiate, it starts from the bottom of the distant bundle (i.e. the largest distant task) and it selects the first one which is a potential socially rational delegation according to its beliefs.

3 EMPIRICAL VALIDATION

We have implemented a multi-agent system which deploys the MapReduce design pattern in a distributed system setting [6]. Our agents negotiate the reduce tasks during the job in order to improve the poor load balancing due to data skews [7, 18, 19], and so the job execution time.

Our experiments[1] are based on a 8 Gio dataset (82, 283 keys, i.e. tasks) which has been generated such that the initial task allocation (cf. Fig. 2) is poorly load balanced in order to check that the locality has an impact on the makespan and thus on the job runtime.

We compare the median job execution times when the agents adopt the trivial strategy or the locality-based one with 10 runs for each configuration. We observe that the locality-based strategy significantly improves the runtime, around −7.6%. When there is no negotiation, it corresponds to the Hadoop behaviour where the initial task allocation is never challenged. For comparison, in this case the job runtime is 853 seconds (around +100%). We deduce that the price of the negotiation can be neglected regarding the impact of the load balancing. Moreover, the negotiation allows to decrease the job execution time and even more with the locality-based strategy.

Fig. 2 compares the task allocation when all the tasks have been performed whatever they have been negotiated (or not) between the 16 agents in accordance with the trivial strategy or the locality-based one. We observe that the makespan of the initial allocation is approximately 3.3 10^8, around 2.5 10^8 (−24%) for the negotiation with the trivial strategy and 2 10^8 (−30.7%) for the locality-based one. We deduce that the negotiation, in particular with the locality-based strategy, allows to improve the load balancing.

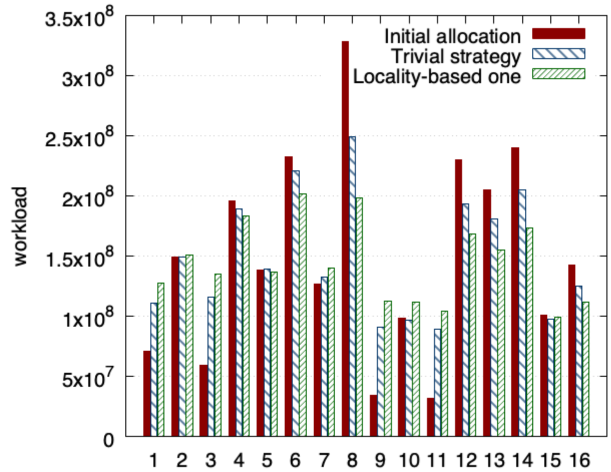

Figure 2: Initial and *ex-post* task allocation with the trivial strategy and the locality-based one.

4 CONCLUSION

We have introduced a task reallocation mechanism which is dynamic, since it is concurrent with the tasks processing, and adaptive to disruptive events, i.e. task consumptions. In this context, we have proposed a locality-based strategy adopted by the agents operating in concurrent one-to-many negotiations for task delegations. According to their local beliefs and knowledge, the agents perform first large local tasks and they negotiate first large distant ones. In order to validate our approach, we have developed a proof-of-concept prototype where agents negotiate reduce tasks of the MapReduce design pattern in a distributed setting. Our preliminary empirical results show that, for such an application, the locality-based strategy can improve the load balancing and so the job execution time.

As part of our future work we are currently evaluating our prototype with numerous and representative experiments with real-word datasets in various settings.

[1]Our experiments have been performed on 16 PCs with 4 cores Intel(R) i7 and 16GB RAM each.

REFERENCES

[1] Sherief Abdallah and Victor Lesser. 2005. Modeling task allocation using a decision theoretic model. In *Proc. of AAMAS*. ACM, 719–726.

[2] Alessandro Agnetis, Jean-Charles Billaut, Stanislaw Gawiejnowicz, Dario Pacciarelli, and Ameur Soukhal. 2014. *Multiagent Scheduling - Models and Algorithms*. Springer.

[3] Bedour Alrayes, Özgür Kafalı, and Kostas Stathis. 2017. Concurrent bilateral negotiation for open e-markets: the CONAN strategy. *Knowledge and Information Systems* 56, 2 (2017), 463–501.

[4] Bo An, Victor Lesser, David Irwin, and Michael Zink. 2010. Automated Negotiation with Decommitment for Dynamic Resource Allocation in Cloud Computing. In *Proc. of AAMAS*. 981–988.

[5] Quentin Baert, Anne-Cécile Caron, Maxime Morge, and Jean-Christophe Routier. 2017. Negotiation Strategy of Divisible Tasks for Large Dataset Processing. In *Multi-Agent Systems and Agreement Technologies - 15th European Conference, EUMAS 2017, and 5th International Conference, AT 2017, Évry, France, December 14-15, 2017, Revised Selected Papers*. 370–384.

[6] Quentin Baert, Anne-Cécile Caron, Maxime Morge, and Jean-Christophe Routier. 2018. Fair multi-agent task allocation for large datasets analysis. *Knowledge and Information Systems* 54, 3 (Mar 2018), 591–615.

[7] Quan Chen, Daqiang Zhang, Minyi Guo, Qianni Deng, and Song Guo. 2010. SAMR: A self-adaptive MapReduce scheduling algorithm in heterogeneous environment. In *International Conference on Computer and Information Technology*. IEEE, 2736–2743.

[8] Han-Lim Choi, Luc Brunet, and Jonathan P How. 2009. Consensus-based decentralized auctions for robust task allocation. *IEEE transactions on robotics* 25, 4 (2009), 912–926.

[9] J. Dean and S. Ghemawat. 2008. MapReduce: simplified data processing on large clusters. *Commun. of ACM* 51, 1 (2008), 107–113.

[10] Saurabh Kumar Garg, Srikumar Venugopal, James Broberg, and Rajkumar Buyya. 2013. Double auction-inspired meta-scheduling of parallel applications on global grids. *J. Parallel and Distrib. Comput.* 73, 4 (2013), 450–464.

[11] Preetam Ghosh, Nirmalya Roy, Sajal K Das, and Kalyan Basu. 2004. A game theory based pricing strategy for job allocation in mobile grids. In *Proc. of 18th International Symposium on Parallel and Distributed Processing*. IEEE, 82.

[12] Daniel Grosu and Anthony T Chronopoulos. 2005. Noncooperative load balancing in distributed systems. *Journal of parallel and distributed computing* 65, 9 (2005), 1022–1034.

[13] Hesam Izakian, Ajith Abraham, and Behrouz Tork Ladani. 2010. An auction method for resource allocation in computational grids. *Future Generation Computer Systems* 26, 2 (2010), 228–235.

[14] Yichuan Jiang. 2016. A survey of task allocation and load balancing in distributed systems. *IEEE Transactions on Parallel and Distributed Systems* 27, 2 (2016), 585–599.

[15] Yichuan Jiang and Zhichuan Huang. 2012. The rich get richer: Preferential attachment in the task allocation of cooperative networked multiagent systems with resource caching. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 42, 5 (2012), 1040–1052.

[16] Qin-Ma Kang, Hong He, Hui-Min Song, and Rong Deng. 2010. Task allocation for maximizing reliability of distributed computing systems using honeybee mating optimization. *Journal of Systems and Software* 83, 11 (2010), 2165–2174.

[17] Sarit Kraus, Onn Shehory, and Gilad Taase. 2003. Coalition formation with uncertain heterogeneous information. In *Proc. of AAMAS*. ACM, 1–8.

[18] YongChul Kwon, Magdalena Balazinska, Bill Howe, and Jerome Rolia. 2012. Skewtune: mitigating skew in mapreduce applications. In *Proc. of the SIGMOD International Conference on Management of Data*. ACM, 25–36.

[19] Miguel Liroz-Gistau, Reza Akbarinia, and Patrick Valduriez. 2015. FP-Hadoop: efficient execution of parallel jobs over skewed data. *VLDB Endowment* 8, 12 (2015), 1856–1859.

[20] Satish Penmatsa and Anthony T Chronopoulos. 2011. Game-theoretic static load balancing for distributed systems. *J. Parallel and Distrib. Comput.* 71, 4 (2011), 537–555.

[21] Reid G. Smith. 1980. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. *IEEE Transactions on computers* 29, 12 (December 1980), 1104–1113.

[22] Joanna Turner, Qinggang Meng, Gerald Schaefer, and Andrea Soltoggio. 2018. Distributed Strategy Adaptation with a Prediction Function in Multi-Agent Task Allocation. In *Proc. of AAMAS*. 739–747.

[23] Joanna Turner, Qinggang Meng, Gerald Schaefer, and Andrea Soltoggio. 2018. Fast consensus for fully distributed multi-agent task allocation. In *Proc. of the 33rd Annual ACM Symposium on Applied Computing*. ACM, 832–839.

[24] William E Walsh and Michael P Wellman. 1998. A market protocol for decentralized task allocation. In *ICMAS*. 325–332.

[25] Peng-Yeng Yin, Shiuh-Sheng Yu, Pei-Pei Wang, and Yi-Te Wang. 2007. Task allocation for maximizing reliability of a distributed system using hybrid particle swarm optimization. *Journal of Systems and Software* 80, 5 (2007), 724–735.