Asynchronous Communication Aware Multi-Agent Task Allocation

Extended Abstract

Ben Rachmut
Ben Gurion University of the Negev
Beer Sheva, Israel
rachmut@post.bgu.ac.il

Sofia Amador Nelke
Holon Institute of Technology
Holon, Israel
sofiaa@hit.ac.il

Roie Zivan
Ben Gurion University of the Negev
Beer Sheva, Israel
zivanr@bgu.ac.il

ABSTRACT

Multi-agent task allocation in physical environments with spatial and temporal constraints are hard problems relevant to many realistic applications. A task allocation algorithm based on Fisher market clearing (FMC_TA), which can be performed centrally or distributively, has been shown to produce high quality allocations compared to the centralized and distributed state of the art incomplete optimization algorithms. However, the algorithm is synchronous and thus depends on perfect communication between agents. We propose FMC_ATA, an asynchronous version of FMC_TA, which is robust to message latency and message loss. In contrast to the former version of the algorithm, FMC_ATA allows agents to identify events and initiate the generation of an updated allocation. Thus, it is more compatible with dynamic environments.

KEYWORDS

Communication Aware; Multi-Agent Optimization; Task Allocation

ACM Reference Format:
Ben Rachmut, Sofia Amador Nelke, and Roie Zivan. 2023. Asynchronous Communication Aware Multi-Agent Task Allocation: Extended Abstract. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 3 pages.

1 INTRODUCTION

Task allocation is a major challenge in realistic scenarios, e.g., disaster response, where rescue units need to coordinate in order to save as many lives as possible [5, 11, 15]. Such coordination is extremely challenging since, in such scenarios, the communication among agents is expected to be severely degraded and unreliable [1, 6, 12]. Moreover, such scenarios are highly dynamic due to the appearance of new events or the change of the status of handled events [16]. Identification of dynamic events would most likely be by the agents performing in the environment. Thus, we expect the agents to be able to reinitialize the solving process when necessary [3, 4].

Fisher Market Clearing Task Allocation (FMC_TA) [8, 9] is an algorithm that was proposed for solving problems where a team of heterogeneous agents needs to cooperate in an environment that includes multiple tasks, which require ad-hoc coalitions of agents with different skills in order to properly handle them. The algorithm is composed of two phases. In the first, the problem is reduced to a Fisher market, having task performing agents as buyers in the market and tasks as goods. Then, the corresponding Fisher market clearing allocation is found. When the allocation resulted in tasks that are shared among a number of agents, an ad-hoc coalition was generated, which included the agents that received a share of the task. In the second phase, a distributed ordering heuristic is performed for agents to decide on the schedule in which they perform tasks. The Fisher market clearing outcome is guaranteed to be envy free and Pareto optimal [2, 14]. This unique combination results in an allocation where agents share important and complex tasks efficiently. FMC_TA was shown to dominate the state of the art centralized and distributed task allocation algorithms.

However, FMC_TA is a synchronous algorithm in which, in each iteration, agents perform calculations only after they receive all messages that they expect to be sent to them by their neighbors [7, 18, 20]. Unfortunately, such a synchronous algorithmic design incurs a number of drawbacks. If a message is delayed, the iteration starts late. If a message is lost, the agents are in a deadlock.

Motivated by the need to adjust this algorithm, which has shown high quality performance in multi-agent task allocation scenarios, to realistic dynamic settings in which communication is imperfect, we propose FMC_ATA, an asynchronous version of FMC_TA. It allows agents to perform computation and generate messages whenever they receive a message. FMC_ATA performs *a single phase* in which the allocation of tasks to agents, the ad-hoc coalitions that share tasks, and the schedule for each agent is produced. We compared the performance of FMC_ATA and FMC_TA. Our results indicate that the solution quality of FMC_ATA is similar to the solution quality of FMC_TA, even in the presence of extreme communication disturbances.

2 FISHER MARKET CLEARING ASYNCHRONOUS TASK ALLOCATION

The asynchronous version of FMC_TA (FMC_ATA) was designed to be (and evidently is) robust both to message latency and message loss. The three major aspects that differentiate it from FMC_TA are: **1)** In FMC_ATA, agents do not wait for messages to arrive from all their neighbors in order for a computation step to begin. On the contrary, each message received triggers such a computation step. **2)** In FMC_ATA the two phases of FMC_TA are merged into a single step and performed simultaneously. **3)** In FMC_TA (both in the centralized and in the distributed version), it is assumed that there is a central entity that is aware of the location and importance of all tasks and that it propagates this information to the agents. In FMC_ATA, agents dynamically discover tasks and propagate the relevant information to their peers. FMC_ATA is a distributed

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

asynchronous algorithm that includes two types of entities: active agents and task agents. In practice, the role of a task agent is performed by one of the active agents, e.g., the first to identify the task. The communication graph is bipartite, i.e., the neighbors of each active agent are only task agents and vice versa.

An active agent aa_i and a task agent ta_j perform steps of computation asynchronously as a reaction to messages they receive. In a such step of the algorithm, task agent ta_j updates its local view according to the message received (i.e, bids and arrival times). The price $p_k \in \vec{p}$ for a skill (sub-task) k is calculated as follows $p_k = \sum_{i \in A} b_{ik}$, where b_{ik} is the bid received from aa_i corresponding to sub-task v_{jk}. The resulting allocations of the sub-task related to skill k are determined by $x_{ik} = \frac{b_{ik}}{p_k}$. Then, ta_j proceeds to the scheduling process, which is integrated into the same step of computation (in contrast to FMC_TA, where the scheduling phase was performed in a separate step). According to the arrival time reported by the active agents, the earliest time in which all active agents that were allocated to perform the sub-task can arrive and start performing the sub-task concurrently is calculated. Finally, messages are sent to each of $ta'_j s$ neighbors with the up-to-date allocations and the calculated starting times.

The active agent discovers a task she starts representing it, and it sends a "handshake" message to all relevant active agents containing the required information regarding the new task. This type of message allows tasks to be detected synchronously and added to the tasks included in the algorithm.

An active agent aa_i in FMC_ATA distinguishes between the two types of messages and reacts accordingly. If the message type is a "handshake", a new task v_j is added to the set tasks aa_i is aware of. Next, its personal utility is calculated for each of the skills that aa_i has and v_j requires. If the message type is a standard message (i.e., aa_i is familiar with the corresponding task v_j), aa_i updates its allocation and the earliest shared execution time.

Following the update of the active agent's local view, it proceeds to re-calculate its bids based on the allocation it received (following [17]). For a new sub-task k (i.e., received by a handshake message), the agent initiates its current allocation as $x_{jk} = 1$, i.e., it assumes it performs the sub-task alone.

The following steps of the active agent's algorithm are equivalent to the second phase in FMC_TA. It generates an initial schedule of the tasks allocated to it according to their Bang per Buck [8] and calculates the arrival time for each of the tasks accordingly. Then, the agent tries to promote tasks that are not shared, without changing the arrival time of shared tasks. Each such message includes a bid and an initial arrival time for every relevant sub-task.

3 EXPERIMENTAL EVALUATION

To analyze the performance of *FMC_ATA*, we used a distributed asynchronous simulator, where agents were implemented as Python threads. That allows for examining imperfect communication by enabling patterns of message delays and probability for message loss. Imperfect communication was simulated according to the method suggested in [13, 19]. The delay was selected in terms of the number of *Non-Concurrent Logic Operations* (NCLO), an independent measure for evaluating the performance of algorithms in asynchronous distributed settings [10, 19]. The simulator's code is public and

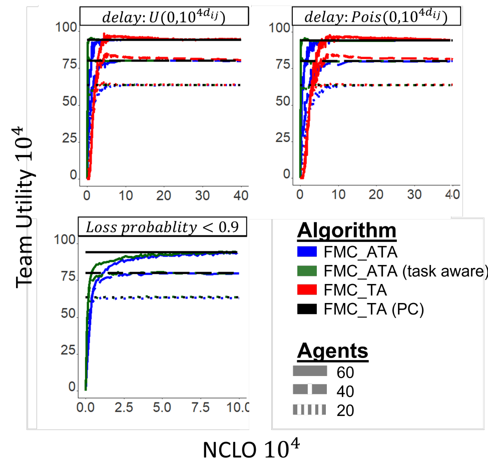

Figure 1: Team utility as a function of NCLOs. Message delays sampled from a Uniform and Poisson distributions and probability of 90% for message loss.

available[1]. In each experiment, we randomly generated 50 in different instances. The results presented in the graphs are an average of those 50 runs. Each scenario in an experiment included two types of agents, active agents, and task agents. A random geographic location (coordinates x and y) was selected uniformly between 0 and 10^6. Each problem instance included active agents with a set of three unique skills.

Figure 1 presents the results of the convergence process of the algorithms in the presence of imperfect communication. The average team utility as a result of the schedule that would have been produced by each algorithm every 1000 NCLOs, is presented. Each instance has a set of 25 random tasks. We examined how the algorithms scale by evaluating their performance on problems with different amounts of active agents (e.g., 20, 40, and 60). The curves correspond to different versions of the algorithm. We considered two versions of the proposed FMC_ATA algorithm in our experiments. In FMC_ATA_task_aware, active agents are initially informed of all tasks included in the problem. In contrast, in FMC_ATA, agents discover tasks during execution. We compared those algorithms with synchronous FMC_TA [8]In addition, we examined the performance of FMC_TA with perfect communication (PC).

Each sub-graph presents a different communication pattern. Delays were taken from a uniform and a Poisson distribution dependent on a normalized value of the distance between entities. For Message loss, we present a scenario where the probability of a message being lost is constant and is set to 0.9. Notice that FMC_TA was not included in the experiments that included message loss since it deadlocks in such scenarios.

In all the experiments, the algorithm converges to solutions with similar quality. This indicates that FMC_ATA converges to the same market-clearing solution, and it preserves the solution properties of FMC_TA. Moreover, FMC_ATA has an improved convergence rate in scenarios with 60 active agents. This happens despite the fact that FMC_TA agents are informed of active tasks by a central entity. This demonstrates the vulnerability of the synchronous algorithm to message latency.

[1] https://github.com/benrachmut/Simulation_For_Research

ACKNOWLEDGMENTS

This research is partially supported by US-Israel Binational Science Foundation (BSF) grant #2018081 and US National Science Foundation (NSF) grant #1838364.

REFERENCES

[1] Estefany Carrillo, Suyash Yeotikar, Sharan Nayak, Mohamed Khalid M Jaffar, Shapour Azarm, Jeffrey W Herrmann, Michael Otte, and Huan Xu. 2021. Communication-aware multi-agent metareasoning for decentralized task allocation. *IEEE Access* 9 (2021), 98712–98730.

[2] Nikhil R Devanur, Christos H Papadimitriou, Amin Saberi, and Vijay V Vazirani. 2002. Market equilibrium via a primal-dual-type algorithm. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.* IEEE, 389–395.

[3] Alessandro Farinelli, Luca Iocchi, and Daniele Nardi. 2017. Distributed on-line dynamic task assignment for multi-robot patrolling. *Autonomous Robots* 41, 6 (2017), 1321–1345.

[4] Ferdinando Fioretto, Enrico Pontelli, and William Yeoh. 2018. Distributed constraint optimization problems and applications: A survey. *Journal of Artificial Intelligence Research* 61 (2018), 623–698.

[5] E Gil Jones, M Bernardine Dias, and Anthony Stentz. 2007. Learning-enhanced market-based task allocation for oversubscribed domains. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2308–2313.

[6] Kathryn Macarthur, Ruben Stranders, Sarvapali Ramchurn, and Nicholas Jennings. 2011. A distributed anytime algorithm for dynamic task allocation in multi-agent systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 25. 701–706.

[7] R. Maheswaran, J. Pearce, and M. Tambe. 2004. Distributed Algorithms for DCOP: A Graphical Game-Based Approach. In *Proceedings of PDCS.* 432–439.

[8] Sofia Amador Nelke, Steven Okamoto, and Roie Zivan. 2020. Market Clearing-based Dynamic Multi-agent Task Allocation. *ACM Transactions of Intelligent Systems Technology.* 11, 1 (2020), 4:1–4:25.

[9] Sofia Amador Nelke and Roie Zivan. 2017. Incentivizing Cooperation between Heterogeneous Agents in Dynamic Task Allocation. In *Proceedings of the 16th International Conference on Autonomous Agents and Multi-agent Systems, (AAMAS).* 1082–1090.

[10] Arnon Netzer, Alon Grubshtein, and Amnon Meisels. 2012. Concurrent forward bounding for distributed constraint optimization problems. *Artificial Intelligence* 193 (2012), 186–216.

[11] Ernesto Nunes, Marie Manner, Hakim Mitiche, and Maria Gini. 2017. A taxonomy for task allocation problems with temporal and ordering constraints. *Robotics and Autonomous Systems* 90 (2017), 55–70.

[12] Michael Otte, Michael J Kuhlman, and Donald Sofge. 2020. Auctions for multi-robot task allocation in communication limited environments. *Autonomous Robots* 44, 3 (2020), 547–584.

[13] Ben Rachmut, Roie Zivan, and William Yeoh. 2021. Latency-Aware Local Search for Distributed Constraint Optimization. In *20th 2nd International Conference on Autonomous Agents and Multi-agent Systems (AAMAS).* 1019–1027.

[14] J Hans Reijnierse and Jos AM Potters. 1998. On finding an envy-free Pareto-optimal division. *Mathematical Programming* 83, 1 (1998), 291–311.

[15] Satoshi Tadokoro, Hiroaki Kitano, Tomoichi Takahashi, Itsuki Noda, Hitoshi Matsubara, Atsushi Shinjoh, Tetsuhiko Koto, Ikuo Takeuchi, Hironao Takahashi, Fumitoshi Matsuno, et al. 2000. The robocup-rescue project: A robotic approach to the disaster mitigation problem. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, Vol. 4. IEEE, 4089–4094.

[16] Changyun Wei, Koen V Hindriks, and Catholijn M Jonker. 2016. Dynamic task allocation for multi-robot search and retrieval tasks. *Applied Intelligence* 45, 2 (2016), 383–401.

[17] Li Zhang. 2011. Proportional response dynamics in the Fisher market. *Theoretical Computer Science* 412, 24 (2011), 2691–2698.

[18] W. Zhang, G. Wang, Z. Xing, and L. Wittenberg. 2005. Distributed Stochastic Search and Distributed Breakout: Properties, Comparison and Applications to Constraint Optimization Problems in Sensor Networks. *Artificial Intelligence* 161, 1–2 (2005), 55–87.

[19] Roie Zivan and Amnon Meisels. 2006. Message delay and DisCSP search algorithms. *Annals of Mathematics and Artificial Intelligence(AMAI)* 46 (2006), 415–439.

[20] Roie Zivan, Steven Okamoto, and Hilla Peled. 2014. Explorative anytime local search for distributed constraint optimization. *Artificial Intelligence* 212 (2014), 1–26.