From Abstractions to Grounded Languages for Robust Coordination of Task Planning Robots

Extended Abstract

Yu Zhang
SCAI, Arizona State University
Tempe, United States of America
yzhan442@asu.edu

ABSTRACT

Individual robots in distributed systems must often coordinate to optimize the global performance. Where explicit coordination via communication is concerned, it is almost always achieved via a predefined "*language*" designed by human users. Such hand-designed languages tend to be either too rigid or too forgiving, leading to brittle solutions, excess negotiation costs, or unexpected coordination issues (e.g., deadlocks). In this paper, we consider a first step to bridge the gap for task planning robots using symbolic planning. Specifically, we study the automatic construction of languages that are *maximally flexible* while being *sufficiently explicative* for coordination. To this end, we view language as a machinery for specifying temporal-state constraints of plans. Such a view enables us to reverse-engineer a language from the ground up by mapping these *composable* constraints to words. Our language expresses a plan for any given task as a "*plan sketch*" to convey just-enough details while maximizing the flexibility to realize it, leading to robust coordination with optimality guarantees among other benefits. We formulate the problem, analyze it, and provide an approximate solution. We validate the advantages of our approach under various scenarios to shed light on its applications.

KEYWORDS

Planning for Coordination; Robust Coordination; Cooperative Robots

ACM Reference Format:
Yu Zhang. 2023. From Abstractions to Grounded Languages for Robust Coordination of Task Planning Robots: 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

To facilitate explicit coordination via communication between robots in distributed systems, a key consideration is the adoption of a "*language*" that the robots can all speak. Such a language often relies on words with predefined meanings that are designed by human users [3, 8, 20, 27]. However, such languages tend to be either too rigid or too forgiving, leading to brittle solutions, excess negotiation costs, or unexpected coordination issues (e.g., deadlocks). In this paper, as a first step, we consider to bridge the gap for task planning robots using symbolic planning.

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.

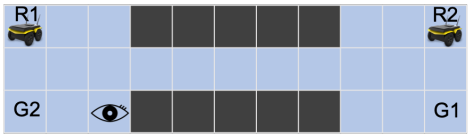

Figure 1: Motivating scenario involving two pathfinding robots, R_1 and R_2, in a gridworld. Each cell can only accommodate a single robot at a time and the darker cells are obstacles. The robots are tasked to reach their goal locations (G_1 and G_2), respectively, in the shortest timespan while avoiding collisions. They have a limited sensing range and communication is costly. During plan execution, there may be locations of interest popping up at random places that require one of the robots to visit (denoted by the eye sign).

Traditional methods for explicit coordination (implicit coordination via observations and actions not considered here) in distributed systems with planning agents can be divided into two classes:

1) *Centralized plan and distributed execution*: provides optimality guarantees except when approximate solutions are considered [16, 18, 21]. Note that the planning process may be centralized or distributed [16]. Explicit coordination in this class involves broadcasting the centralized plan in the planning language and sometimes exchanging messages as stipulated by the plan during plan execution. This approach results in brittle solutions (i.e., a single agent changing its part of plan requires the entire plan to be updated) among other limitations.

2) *Distributed plans and distributed execution*: provides no guarantee of optimality in general [4, 28]. Note that distributed plans imply that the planning process is distributed. Methods in this class are often rule or local-search based [2, 19], making them adaptive to local changes and easy to implement. For explicit coordination, a language is often designed manually on a case-by-case basis, which is prone to unexpected coordination issues (e.g., deadlocks).

Our work serves as a middle ground that bridges the two classes and combines their advantages, contributing a novel perspective for explicit coordination between task planning agents. Consider the scenario in Fig. 1. The problem is difficult for the second class of methods: the robots must coordinate before one of them enters the narrow pathway so methods based on local information only would not work well (i.e., leading to deadlocks). Furthermore, given that the locations of interest are unpredictable, neither can we assign fixed priorities to the robots (e.g., always letting R_1 go through first). While these issues are not present in the first class since the robots coordinate a plan before execution, whenever some location of interest pops up during plan execution, the robots must replan and re-coordinate, significantly increasing the cost.

While similar to the first class, robots in our approach coordinate by communicating "*plan sketches*" that guarantee optimality while maximizing flexibility to reduce the need for replanning and re-coordination (thus differing from work on replanning or plan repair for making replanning more efficient [9]). In the scenario above, the robots can communicate that "R_2 *to wait for* R_1" without specifying the exact plan to be followed, so that robot R_2 later (instead of R_1 even though R_1 would detect the location of interest first, since that would require R_1 to wait for R_2) can adjust locally to visit the location of interest even if its original plan does not pass through it. A crucial property is to enable each robot to make local changes without any negative impact to the (global) makespan as long as the updated plan is still consistent with the plan sketch.

To this end, we view language as a machinery for specifying plan constraints. A language thus specifies a *plan space abstraction* where a sentence in the language (i.e., a plan sketch) specifies a set of plans. The robots all commit to the same set of plans as a result of coordination (i.e., one robot communicates a plan sketch to the others). To guarantee the feasibility of this approach, given that the robots may be unaware of the local changes made by the others, one of key challenges is for the plans in this set to not introduce *miscoordinations*; to maximize flexibility, the number of plans should be maximized. Since different sets will be specified for different tasks, we instead minimize the number of words in the language, resulting in a *minimal language*. In our approach, we associate words in the language with temporal-state constraints that are composable. We show that searching for a minimal language is NEXP-complete. In light of this result, we develop an approximate solution. We validate the benefits of the languages under various application scenarios. The full paper can be accessed online here [29].

2 RELATED WORK

A language often represents a structured symbolic system mapping symbols to semantic meanings that can be grounded in the environment [10, 11, 22, 24, 25]. In this work, we instead reverse engineer the process by considering the mapping from temporal-state constraints to symbols for language construction. These symbols (i.e., words) are then used to form sentences, which introduce a *plan space abstraction* to resolve miscoordinations. The idea of applying abstraction to planning problems is not new. Most prior work has focused on state abstraction for problem decomposition, which has been well studied in both path and task planning methods [7, 12, 15]. Such decomposition has also been shown to benefit communication and coordination in multi-agent planning [5, 17]. The temporal-state constraints for plan space abstraction used in our approach resemble options in semi-MDP and LTL expressions in temporal logic [6, 23, 26]. However, these prior approaches have mostly focused on applying plan space abstraction to improving planning [1, 23], or learning such abstraction for problem decomposition [13, 14]. We consider plan space abstraction for coordination.

3 EVALUATION

In this evaluation, we demonstrate how the language computed by our method (referred to as a *coordination language*) contributes to robust coordination during plan execution. We consider a warehouse setting (see Fig. 2) where robots are tasked to deliver products between one of the storage zones (located at the corners of the workspace and labeled as $S1$ and $S2$) and one of the dispatch zones (located at the other corners of the workspace and labeled as $D1$ and $D2$). Products must be transported between the corresponding zones (i.e., $S1 - D1$ and $S2 - D2$). For a given task, the robots start randomly from different corners and must deliver, respectively, to the corresponding zones for storage or dispatch. At the same time, a human worker may be present in the workspace at a random location other than the four corners. We assume that the human worker would not change his/her location during the task. Since the robots are from different manufacturers, they would not be able to robustly detect each other but can both detect the human. To guarantee safety, the robots must coordinate to avoid collisions with each other and the human. We assume that robots move at the same speed. The robots can coordinate their plans via a coordination language before execution but can only detect the position of the human *after* the plan execution starts.

Environment Size	Success Rate Exact Plan	Success Rate Our Language	Gain
3×3	18.0%	**24.0%**	33.3%
4×3	28.0%	**42.0%**	50.0%
5×3	27.0%	**34.0%**	25.9%
4×4	43.0%	**58.0%**	34.9%

Table 1: Success rates of 100 tasks with dynamic obstacles.

We tested the success rates of 100 randomly generated tasks when the robots used the exact plan or a sentence in the computed coordination language for expressing the plan to coordinate. When the coordination language is used, the robots can choose other candidate plans (if available) that are expressed by the sentence even when the initial plan would lead to a collision with the human. Table 1 shows the results for environments of different sizes where a language is constructed for each environment. We can see that the use of coordination languages significantly improved the success rate in all environments. As the environment size increases, the success rates also increased as the chance of collision decreases.

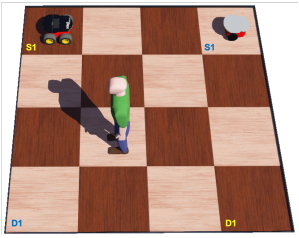

Figure 2: Problem setting for a navigation domain with dynamic obstacles (i.e., a human worker).

4 CONCLUSIONS AND DISCUSSIONS

In this paper, we introduced a novel language formation problem for achieving robust coordination. It bridged a gap in prior methods for coordinating planning agents in distributed systems and combined their advantages. We viewed language as a machinery for resolving miscoordinations and reverse-engineered a language to maximize flexibility during plan execution while guaranteeing optimality. A full version of our work can be accessed online here [29].

ACKNOWLEDGMENTS

This research is supported in part by the NSF grants 2047186 and the AFOSR grant FA9550-18-1-0067.

REFERENCES

[1] David Abel, D Ellis Hershkowitz, and Michael L Littman. 2017. Near optimal behavior via approximate state abstraction. *arXiv preprint arXiv:1701.04113* (2017).

[2] Tamio Arai, Enrico Pagello, Lynne E Parker, et al. 2002. Advances in multi-robot systems. *IEEE Transactions on robotics and automation* 18, 5 (2002), 655–661.

[3] Mihai Barbuceanu and Mark S Fox. 1995. COOL: A Language for Describing Coordination in Multi Agent Systems.. In *ICMAS*. 17–24.

[4] Zahy Bnaya and Ariel Felner. 2014. Conflict-oriented windowed hierarchical cooperative A*. In *ICRA*. IEEE.

[5] Bradley J Clement, Edmund H Durfee, and Anthony C Barrett. 2007. Abstract reasoning for planning and coordination. *Journal of Artificial Intelligence Research* 28 (2007), 453–515.

[6] E Allen Emerson. 1990. Temporal and modal logic. In *Formal Models and Semantics*. Elsevier, 995–1072.

[7] Kutluhan Erol, James Hendler, and Dana S Nau. 1994. HTN planning: Complexity and expressivity. In *AAAI*. 1123–1128.

[8] Tim Finin, Richard Fritzson, Don McKay, and Robin McEntire. 1994. KQML as an agent communication language. In *CIKM*. ACM, 456–463.

[9] Maria Fox, Alfonso Gerevini, Derek Long, and Ivan Serina. 2006. Plan Stability: Replanning versus Plan Repair.. In *ICAPs*, Vol. 6. 212–221.

[10] Ze Gong and Yu Zhang. 2018. Temporal spatial inverse semantics for robots communicating with humans. In *ICRA*. IEEE.

[11] Stevan Harnad. 1990. The symbol grounding problem. *Physica D: Nonlinear Phenomena* 42, 1-3 (1990), 335–346.

[12] Subbarao Kambhampati, Craig A Knoblock, and Qiang Yang. 1995. Planning as refinement search: A unified framework for evaluating design tradeoffs in partial-order planning. *Artificial Intelligence* 76, 1-2 (1995), 167–238.

[13] George Konidaris, Leslie Kaelbling, and Tomas Lozano-Perez. 2014. Constructing symbolic representations for high-level planning. In *AAAI*.

[14] George Konidaris, Scott Kuindersma, Roderic Grupen, and Andrew Barto. 2012. Robot learning from demonstration by constructing skill trees. *IJRR* 31, 3 (2012), 360–375.

[15] Jonas Kvarnström and Patrick Doherty. 2000. TALplanner: A temporal logic based forward chaining planner. *Annals of mathematics and Artificial Intelligence* 30, 1-4 (2000), 119–169.

[16] Raz Nissim, Ronen I Brafman, and Carmel Domshlak. 2010. A general, fully distributed multi-agent planning algorithm. In *AAMAS*. 1323–1330.

[17] Frans Oliehoek, Stefan Witwicki, and Leslie Kaelbling. 2021. A sufficient statistic for influence in structured multiagent environments. *JAIR* 70 (2021), 789–870.

[18] Frans A Oliehoek and Christopher Amato. 2016. *A concise introduction to decentralized POMDPs*. Springer.

[19] Lynne E Parker. 1998. ALLIANCE: An architecture for fault tolerant multirobot cooperation. *IEEE transactions on robotics and automation* 14, 2 (1998), 220–240.

[20] Stefan Poslad. 2007. Specifying protocols for multi-agent systems interaction. *TAAS* 2, 4 (2007), 15.

[21] Guni Sharon, Roni Stern, Ariel Felner, and Nathan R Sturtevant. 2015. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence* 219 (2015), 40–66.

[22] Luc Steels. 2012. Grounding language through evolutionary language games. In *Language Grounding in Robots*. Springer, 1–22.

[23] Richard S. Sutton, Doina Precup, and Satinder Singh. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *AIJ* 112, 1 (1999), 181 – 211.

[24] Stefanie Tellex, Ross Knepper, Adrian Li, Daniela Rus, and Nicholas Roy. 2014. Asking for Help Using Inverse Semantics. In *RSS*. Berkeley, USA.

[25] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. 2011. Approaching the symbol grounding problem with probabilistic graphical models. *AI magazine* 32, 4 (2011), 64–76.

[26] Moshe Y Vardi. 1996. An automata-theoretic approach to linear temporal logic. In *Logics for concurrency*. Springer, 238–266.

[27] Ping Xuan, Victor Lesser, and Shlomo Zilberstein. 2001. Communication decisions in multi-agent cooperation: Model and experiments. In *AAMAS*. ACM, 616–623.

[28] Yu Zhang, Kangjin Kim, and Georgios Fainekos. 2016. DISCOF: Cooperative pathfinding in distributed systems with limited sensing and communication range. In *DARS*.

[29] Yu Zhang and Li Wang. 2019. From Abstractions to "Natural Languages" for Planning Agents. *CoRR* abs/1905.00517 (2019). arXiv:1905.00517 http://arxiv.org/abs/1905.00517