Scalable Multiple Robot Task Planning with Plan Merging and Conflict Resolution

Demonstration Track

Gilberto Marcon dos Santos
Collaborative Robotics and Intelligent Systems (CoRIS) Institute, Oregon State University
marcondg@oregonstate.edu

Julie A. Adams
Collaborative Robotics and Intelligent Systems (CoRIS) Institute, Oregon State University
julie.a.adams@oregonstate.edu

ABSTRACT

Agents can individually devise plans and coordinate to achieve common goals. Methods exist to factor planning problems into separate tasks and distribute the plan synthesis process, while reducing the overall planning complexity. Merging distributedly generated plans becomes computationally costly when task plans are tightly coupled, and conflicts arise due to dependencies between plan actions. New plan merging algorithms allow factoring and solving large problems with a growing number of agents and tasks, but are yet to be demonstrated in physical real-world systems. This Demo presents an architecture that deploys plan merging algorithms in a physical multi-robot setting and emulates a First Response Domain.

KEYWORDS

multiagent planning; plan merging; multi-robot planning

ACM Reference Format:
Gilberto Marcon dos Santos and Julie A. Adams. 2021. Scalable Multiple Robot Task Planning with Plan Merging and Conflict Resolution: Demonstration Track. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 3 pages.

1 INTRODUCTION

The growing number of robot capabilities enables robots to accomplish complex tasks and assist in first response to major disasters. Integrating those individual capabilities into a coherent effort to accomplish specific goals requires complex decision making and planning. Real-world applications predominantly rely on a human operator for reasoning, because automated methods fail to scale and solve complex real-world problems [1]. Multiple robot systems can factor and resolve large problems distributedly, but must coordinate to resolve conflicts. New plan merging algorithms, such as the Temporal Optimal Conflict Resolution Algorithm (TCRA*) [7], resolve conflicts across individually-generated plans and allow robots to simultaneously execute their actions and accomplish tasks faster. Task allocation combined with plan merging algorithms enable factored planning and allow complex problems to be solved using a large number of robots and tasks [7], but remain to be demonstrated in real-world settings. This demo introduces a domain-independent approach to deploy a factored planning method and coordinate heterogeneous multiple robot systems, while executing complex plans. The system deploys the Coalition Formation then Planning framework [4] in a physical real-world multiple robot system and evaluates multiple plan merging algorithms.

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

2 ARCHITECTURE

The multiple robot system's architecture is domain independent. The available robots' capabilities and initial states are gathered before planning in order to generate a planning problem, which is solved using the Coalition Formation then Planning framework [4]. Each robot executes an independent and distributed instance of the execution system, which is composed of the Robot Operating System (ROS) [8] Communication Node and the Plan Execution Node. The robots follow the plan, and coordinate using their internal Robot Execution System, which controls each robot's behavior at a high level during plan execution.

Robots use the Communication Node to request information from other robots, as ROS does not support multiple robots natively. The Communication Node contains a knowledge database with all the information relevant for plan execution, including the global plan and messages from other robots. A robot can query another robot, which in turn reads its internal knowledge database and sends a reply. Robots are allowed a high level of autonomy during plan execution and the communication between robots is limited to disseminating plans and announcing action completions.

The Plan Execution Node commands a robot's actions and interfaces with the Communication Node to obtain information about other robots and standard ROS nodes for robot localization, navigation, etc. Plan execution begins as soon as a plan is received.

The Plan Execution Node implements the Plan Execution Algorithm, Algorithm 1, in order to coordinate plan execution. The pending action set, $A_{pending}$, represents the actions that remain to be executed, and is initialized with all the plan actions (line 1). The set of plan actions involving the robot, A_r, are identified (line 2). The set intersection between the pending actions and the robot actions, $A_{pending} \cap A_r$, represents the actions involving the robot that remain to be executed (i.e., the robot pending actions). While there are robot pending actions (line 3), each pending action is evaluated (line 4). Actions that have no pending predecessors (line 5) are executed and removed from the pending action set $A_{pending}$ (line 6). When the action is completed, it is announced (line 7) to all robots whose actions are successors of the completed action. The pending action set, $A_{pending}$, is updated by accounting for the action completion announcements issued by other robots (line 8). The predecessors of an action a are the actions a_0 that are ordered before action a, $a_0 \prec a$. The successors of an action a are the actions a_0 that are ordered after action a, $a \prec a_0$.

Data: A multiagent plan π;
Initialize the pending action set $A_{pending}$ with all the actions in plan π;
Identify the robots' actions, A_r, in plan π;
while $A_{pending} \cap A_r$ *is not empty* **do**
  foreach *action* $a \in A_{pending} \cap A_r$ **do**
    if *action a does not have predecessors in* $A_{pending}$ **then**
      Execute and remove action a from $A_{pending}$;
      Announce that action a is completed;
  Update the pending action set $A_{pending}$;

Algorithm 1: The Plan Execution Algorithm.

The Plan Execution Algorithm allows parallel action execution, as multiple robots execute their plan actions independently, and global coordination only occurs during the planning process, as the robots execute their actions according to each action's predecessors.

3 EXPERIMENTS AND RESULTS

The experimental methodology was designed to evaluate the plan merging algorithms [7] using a simulated real-world mission and multiple robots with specific capabilities. The experiments were performed in an indoor environment that offers a combination of large open spaces, narrow hallways, doorways, and an office space with three separate, but variable sized rooms.

Pioneer P3-DX robots, fitted with laser range finders, were used for the demo. A metric map of the environment was created using a Rao-Blackwellized particle filter simultaneous localization and mapping package [5]. The map was created in real-time by a single robot traversing the environment, while collecting range sensor and wheel odometry data. The map was partitioned using a Voronoi segmentation [3], resulting in a topological map of the environment that allows for high-level planning.

Tasks were allocated to robots according to the robots' capabilities and the tasks' requirements using a dynamic programming coalition formation algorithm [9]. Each task was solved individually using the Actions Concurrency and Time Uncertainty Planner (ActuPlan) [2], which was selected because it supports concurrent durative actions with uncertain action durations.

A First Response Domain [4], was used for the experiments. Each robot was assigned a rescuer, ambulance, or hazard collector role, which did not change. The robots are unable to physically load and unload the simulated victims or hazardous material objects, but they waited (5 seconds) and vocalized the completion of these and all tasks. The six robots, including two rescuers, two ambulances, and two hazard collectors, began each plan execution from the rescue base. Two victim rescue tasks and two hazard collection tasks were solved. The coalition formation algorithm allocated each ambulance to each victim rescue task and each hazard collector to each hazard collection task. Rescuers were allocated two tasks each, as all tasks require a rescuer.

The coupling between tasks requires algorithms that systematically address conflicts while merging task plans [7]. Conflicts arise between plans' actions, as the effects of one action can make other actions infeasible. Plan merging algorithms are also responsible for minimizing the total plan execution duration, or makespan, while addressing plans' conflicts. The Serial Algorithm, the Solution Test Algorithm (STA), and TCRA* ($\epsilon = 1$) [7] were evaluated.

The robots were timed while executing their plans. The plan execution outcomes are: Success, all robots have completed their tasks and returned to the base successfully; and Failure, robots failed to finish within a one-hour time limit or collided with one another and the experiment is terminated to prevent physical damage. All failures were due to collisions, as all trials finished with success or collision before the time limit expired. The rate of success and the makespan are the dependent variables.

The plan merging algorithm used to merge the various task plans are the independent variables. The Serial, STA, and TCRA* merging algorithms were evaluated with the direct and transitive conflict models, with and without the transitive closure [6].

The TCRA* plan, which was identical for all TCRA* instances and for STA using the direct model, with and without closure, had the overall best makespan, as shown in Table 1. STA using the transitive model without transitive closure resulted in the second best makespan and the STA using the transitive model with transitive closure resulted in the third best. The Serial plan was the worst. Success was generally higher for plans with longer makespan due to a more serialized action execution, which minimizes the chances of collisions between robots.

Table 1: Multiple Robot Makespan Descriptive Statistics.

Alg.	Model	Trans. Clos.	Succ.	Mean	Std. Dev.	Med.
TCRA*	Direct	No	5/10	12m46s	00m51s	12m49s
		Yes				
	Trans.	No				
		Yes				
STA	Direct	No				
		Yes				
	Trans.	No	6/10	15m56s	00m28s	16m01s
		Yes	8/10	18m26s	01m02s	18m10s
Serial	N/A	N/A	7/10	25m40s	01m22s	26m00s

4 CONCLUSION

A multiple robot coordination system was introduced to deploy and demonstrate plan merging algorithms in a physical real-world setting. The demo showcases different plan merging algorithms [7]. The multiple robot results were consistent with the simulated evaluation results [7]. The TCRA* algorithm maximized simultaneous action execution and resulted in the shortest plan execution duration. STA and the Serial Algorithm resulted in longer plan execution duration. The algorithms support temporal uncertainty, but do not address uncertain action outcomes. New algorithms will be necessary due to the exponential growth in the action outcomes.

ACKNOWLEDGMENTS

The authors thank Miguel Augusto Ruiz for his technical assistance in the preparation of this demo. This work was partially supported by NSF grant #1723924.

REFERENCES

[1] Ron Alterovitz, Sven Koenig, and Maxim Likhachev. 2016. Robot planning in the real world: Research challenges and opportunities. *AI Magazine*, 37, 2, 76–84.

[2] Eric Beaudry, Froduald Kabanza, and Francois Michaud. 2012. Using a Classical Forward Search to Solve Temporal Planning Problems under Uncertainty. In *Workshops at the AAAI Conference on Artificial Intelligence*, 2–8.

[3] Richard Bormann, Florian Jordan, Wenzhe Li, Joshua Hampp, and Martin Hägele. 2016. Room segmentation: survey, implementation, and analysis. In *International Conference on Robotics and Automation*. Danica Kragic, Antonio Bicchi, and Alessandro De Luca, (Eds.), 1019–1026.

[4] Anton Dukeman and Julie A. Adams. 2017. Hybrid mission planning with coalition formation. *Journal of Autonomous Agents and Multi-Agent Systems*, 31, 6, (November 2017), 1424–1466.

[5] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard. 2007. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE Transactions on Robotics*, 23, 1, 34–46.

[6] Gilberto Marcon dos Santos. 2020. *Coordination for Scalable Multiple Robot Planning Under Temporal Uncertainty*. Ph.D. Dissertation. Oregon State University.

[7] Gilberto Marcon dos Santos and Julie A. Adams. 2020. Optimal temporal plan merging. In *International Conference on Autonomous Agents and Multiagent Systems*. (May 2020), 851–859.

[8] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. 2009. ROS: an open-source robot operating system. In *Workshops at the IEEE International Conference on Robotics and Automation*.

[9] Lovekesh Vig and Julie A. Adams. 2006. Multi-robot coalition formation. *IEEE Transactions on Robotics*, 22, 4, 637–649.