Poster: Towards Multi-Robot Applications Planning Under Uncertainty

Claudio Menghi, Sergio García, and Patrizio Pelliccione
University of Gothenburg | Chalmers, Gothenburg, Sweden
[claudio.menghi,sergio.garcia,patrizio.pelliccione]@gu.se

Jana Tumova
KTH Royal Institute of Technology, Stockholm, Sweden
tumova@kth.se

ABSTRACT

Novel robotic applications are no longer based on single robots. They rather require *teams of robots* that collaborate and interact to perform a desired mission. They must also be used in contexts in which only *partial knowledge* about the robots and their environment is present. To ensure mission achievement, robotic applications require the usage of planners that compute the set of actions the robots must perform. Current planning techniques are often based on centralized solutions and hence they do not scale when teams of robots are considered, they consider rather simple missions, and they do not work in partially known environments. To address these challenges, we present a planning solution that decomposes the team of robots into subclasses, considers complex high-level missions given in temporal logic, and at the same time works when only partial knowledge of the environment is available.

ACM Reference Format:
Claudio Menghi, Sergio García, and Patrizio Pelliccione and Jana Tumova. 2018. Poster: Towards Multi-Robot Applications Planning Under Uncertainty. In *ICSE '18 Companion: 40th International Conference on Software Engineering Companion, May 27-June 3, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3183440.3195046

Robots are used in many contexts to help humans in performing repetitive and dangerous tasks. They are usually assigned to a mission that they must perform. When teams of robots are considered, the mission can be *global*, i.e., the high-level mission that must be accomplished by the whole team, or *local*, i.e., the mission that must be achieved by a single robot, possibly by collaborating with other robots [7]. Consider, as an example, a team of three robots (r_1, r_2, and r_3) deployed in the environment graphically described in Fig. 1. The global mission the team of robots has to achieve is to check whether toxic chemicals have been released by the container located in room l_4. This global mission is decomposed in a set of local missions (one for each robot) specified in Table 1.

These missions together with a model of the robotic application are typically used by planners to automatically compute *plans*, i.e., sets of actions the robots must perform to achieve their missions. A model of the robotic application is usually provided through some form of transition system. These transition systems can be used

ICSE '18 Companion, May 27-June 3, 2018, Gothenburg, Sweden

ACM ISBN 978-1-4503-5663-3/18/05.
https://doi.org/10.1145/3183440.3195046

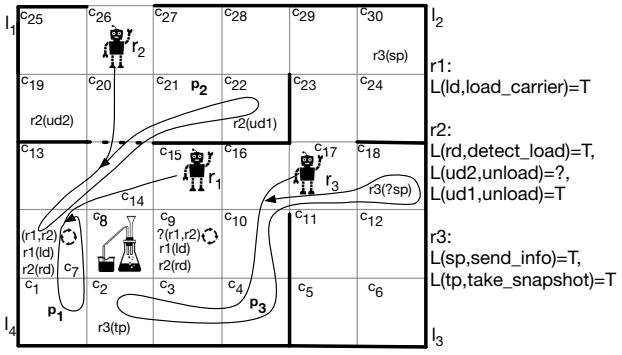

Figure 1: An example showing the model of the robots and their environment and the plans computed by MAPmAKER.

to encode the example shown in Fig. 1. In this example, actions concern the robot movements between cells and tasks the robots can perform. The robots can move between cells separated by gray lines and they cannot cross black bold lines. The cells in which a robot r can perform a task α are marked with the label $r(\alpha)$. For example, in cell c_2 robot r_3 can perform action tp, which specifies that the robot can take a snapshot of r_1 and r_2 removing debris. Table 2 explains the actions the robots can perform. When actions are performed, services are provided by the robots. Services encode high-level functionalities that can be associated with different actions. In our example, robot r_1 provides the service *load_carrier* when action *ld* is executed. This is represented in Fig. 1 through the label $L(ld, load\_carrier) = \top$. Synchronization capabilities model situations in which two robots must simultaneously reach the same cell (meet) and perform the actions that label this cell. This is represented in Fig. 1 by indicating within a cell the name of the robots that must synchronize followed by a rotating arrow. For example, robots r_1 and r_2 can synchronously execute actions *ld* and *rd* in cell c_7.

Recent planners for robotic applications must tackle different challenges: (1) the planning problem should be solved by using algorithms that make the problem tractable; (2) the planning algorithm should work when partial knowledge about the system—the robots and their environment—is present; (3) the planning should consider complex missions specified in temporal logics.

Tractability is a baseline for developing automatic solutions. It concerns the possibility of a machine to solve a problem. Most of the current planners are based on centralized techniques that consider the entire set of robots within the team. This makes planning computationally expensive, especially when the number of robots

Table 1: Local missions of the robots in Figure 1.

LTL formulae	Plain english text
r_1 : G(F(*load_carrier*))	Load debris on top of a *carrier*
r_2 : G(F(*detect_load* ∧F(*unload*)))	Wait until the load is received, then proceed with the unload.
r_3 : G(F(*take_snapshot* ∧F(*send_info*)))	Take pictures of the toxic chemicals, then send them.

Table 2: Description of the actions of Figure 1.

Action	Textual description
ld	The robot can load debris
rd	The robot can receive debris
$ud1$, $ud2$	The robot can unload debris using two different actuators
tp	The robot can take pictures
sp	The robot can send pictures to a communication network

within the team increases [7]. For this reason, recent research interest had focused on decomposing a global mission into a set of local missions which can be exploited by algorithms that analyze the satisfaction of local missions inside subsets of the team of robots reducing computational effort (e.g., [7]). However, the applicability of these algorithms has never been studied when only partial knowledge about the system is available.

Partial knowledge has been considered in the software engineering, formal methods, and robotic literature. One of the ways of representing partial knowledge is through partial models, which have been used to support requirement analysis and elicitation (e.g., [4, 5]), to help designers in producing a model of the system that satisfies a set of desired properties (e.g., [1, 2]), and to verify whether already designed models possess some properties of interest (e.g., [1, 3]). However, partial models have never been used in the development of planners in which partial information is mainly represented through probabilistic models (e.g., [6]). In our work, the presence of partial knowledge about the robots and their environment is modeled by considering uncertainty about actions execution, service provisioning and the robots synchronization (meet). Uncertainty about action execution models the fact that it is unknown whether an action can be executed in a specific location. In Fig 1, when cells are separated by a dashed black bold line, it is unknown if it is possible to move between these cells. When it is unknown whether an action representing a task can be executed, it is preceeded by symbol ?. For example, it is unknown whether robot r_3 can perform action sp in cell c_{18}. Uncertainty about service provisioning concerns cases in which actions can be executed, but it is not known whether by executing an action a service is provided. In Fig. 1, the label $L(ud2, unload) =?$ indicates that there is partial knowledge about the provision of the *unload* service when action $ud2$ is performed. Finally, unknown synchronization capabilities specify that it is unknown whether robot synchronization is possible. This is represented in Fig. 1 by using the symbol ? before the name of the robots that must sync and a rotating arrow. For example, it is unknown whether robots r_1 and r_2 can simultaneously reach cell c_9 and perform actions ld and rd.

Planners able to consider complex missions specified in temporal logics are limited and only recent (e.g., [7]). We would like our planner to be able to consider local missions specified as Linear Temporal Logic (LTL) formulae defined over the services provided by the robots. The LTL formalization of the local missions of the robots of the example in Fig. 1 is presented in Table 1.

To overcome these challenges, we present MAPmAKER: a Multi-robot plAnner for PArtially Known EnviRonments. MAPmAKER is a planning solution that works in partially known environments, supports properties specified in LTL, and is decentralized. Partial knowledge is handled by calling a classical planning algorithm twice. These calls allow the computation of two types of plans: definitive and possible plans. *Definitive plans* are sequences of actions that guarantees the achievement of local missions. *Possible plans* are a sequence of actions that may allow the achievement of local missions. For example, plan p_1 in Fig. 1 is a definitive plan, while plans p_2 and p_3 are possible plans since it is unknown whether robot r_2 can move between cells c_{20} and c_{14} and whether r_3 can perform action sp in c_{18}. Decentralization is supported by decomposing the robotic team into sub-teams based on their missions. The algorithm computes subsets of the team of robots that contains robots depending upon each other for the achievement of their missions. Then, each subset of robots is evaluated in isolation. In the example, two subsets of robots are created: one containing robots r_1 and r_2 (since these robots must meet in cell c_7), and one containing robot r_3. To enable the planner to consider missions specified in LTL our algorithm is developed on the top of an existing solution [7] able to consider LTL missions in a decentralized manner.

Evaluation is performed by checking (1) how MAPmAKER helps planning when partial information is available, by comparing its behavior with the one obtained by using a classical planner; (2) how the employed decentralized algorithm helps in plan computation, by comparing its performance with a non decentralized algorithm. The results show that MAPmAKER is effective when the computed possible plans are actually executable in the real model of the robotic application, and it supports mission achievement in cases that the classical planners can not solve. They also show that the decentralised algorithm succeeds in improving the planner performances.

ACKNOWLEDGMENTS

Research partly supported from the EU H2020 Research and Innovation Programme under GA No. 731869 (Co4Robots).

REFERENCES

[1] Anna Bernasconi, Claudio Menghi, Paola Spoletini, Lenore D. Zuck, and Carlo Ghezzi. 2017. From Model Checking to a Temporal Proof for Partial Models. In *Software Engineering and Formal Methods (SEFM)*. Springer, 54–69.
[2] Claudio Menghi, Paola Spoletini, Marsha Chechik, and Carlo Ghezzi. 2018. Supporting Verification-Driven Incremental Distributed Design of Components. In *Fundamental Approaches to Software Engineering (FASE)*.
[3] Claudio Menghi, Paola Spoletini, and Carlo Ghezzi. 2016. Dealing with Incompleteness in Automata-Based Model Checking. In *Formal Methods*, Vol. 9995. Springer, 531–550.
[4] Claudio Menghi, Paola Spoletini, and Carlo Ghezzi. 2017. COVER: Change-based Goal Verifier and Reasoner.. In *REFSQ Workshops*. Springer.
[5] Claudio Menghi, Paola Spoletini, and Carlo Ghezzi. 2017. Integrating Goal Model Analysis with Iterative Design. In *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer, 112–128.
[6] Nicholas Roy, Geoffrey Gordon, and Sebastian Thrun. 2006. Planning under uncertainty for reliable health care robotics. In *Field and Service Robotics*. Springer, 417–426.
[7] Jana Tumova and Dimos V Dimarogonas. 2016. Multi-agent planning under local LTL specifications and event-based synchronization. *Automatica* (2016).