Learning robots to rescue

Arnoud Visser

20 September 2017

1 INTRODUCTION

The RoboCup Rescue Simulation League aims to develop agents and robots that demonstrate intelligent behavior in a disaster response scenario [1]. The importance of simulation is not only demonstrated at the RoboCup, but also at the DARPA Robotics Challenge Field Trials, where the Virtual Robotics Challenge [2] was a prequel.

Several overview articles were written on the coordination and task allocation research performed with the RoboCup Rescue Agent simulator, the most influential [3]. The multi-agent strategy planning and team coordination problem was approached with a variety of solution techniques, such as decentralized communicating POMDPs [4] and distributed constraint optimization [5].

Coordination inside a robot team is already difficult if the map of the environment is known; it is even more difficult if the map is unknown and have to be explored first [6]. The shared map to be generated by the robots during the Virtual Robot competition [7] has a central role in the coordination of such large robot teams. The shared map is where the distributed sensor information is collected and registered, by each robot independently. The information has to be sent via often unreliable communication links [8], so the robot has selected which information is to be broadcasted (the robots have a need to know what could be of interest for its teammates and the operator). The operator is human, but has to monitor such large robot teams [9] that most of the decisions have to be made independently by the robots. The operator can only give high-level commands (such as the areas to be searched, routes to be followed, etc.) which not even guaranteed to reach the robots (on time).

2 LEARNING COORDINATION POLICIES

The theoretical background of planning inside the RoboCup Rescue Simulation League is well described in [10], including problem descriptions from an objective and subjective perspective and including (approximate) solving approaches. In this paper the coordination problem is for instance first described as decentralized partially observable Markov decision process, which is interpreted as a series of Bayesian games. In this Bayesian game each robot has some private information (not communicated with the team). Two of solution approaches to the problem were worked out in more detail, which both will be summarized here.

2.1 Lossless clustering of multi-agent beliefs

The first attempt to battle the size of the problem was to cluster set of observations into joint types [11]. In this work the Dec-POMDP planning method is not approached with offline planning, where the planning and execution phase are strictly separate. Instead, [11] applies an online planning approach, inspired by [12]. Online planning methods attempt to utilize the received runtime data to create and update a belief about the true state of the environment. They interleave planning and execution phase. Therefore the planning phase has to be performed quickly enough to enable for a smooth execution. Each agent i uses its action-observation

history (IAOH) to infer a belief about the true state of the environment, and this belief is updated upon each individual action performed and individual observation received. As the environment changes depending on the joint action a, each agent has to deduce the individual action (and the resulting individual observation) of all the other agents, relying on a common knowledge assumption. Although the online planning algorithms employ coordination mechanisms, the beliefs of the agents may be different nonetheless, which poses a threat to an effective coordination. Yet, this situation closest resembles the situation in a rescue scenario.

The trick is to map a joint type to a set of joint action-observation histories (JAOHs), which all lead to the decision to execute the same joint action a. This joint type Θ specifies the type of private information each robot has. Each robot knows its own type Θ_i with certainty but not those of other robots. For the other robots it only knows a probability distribution $P \in \Delta(\Theta)$ over the joint type space. When a mechanism exists to ensure that each robot finds the same set of best-response policies π_t, then each robot will maintain the same joint-type space Θ for the team as well as the same probability distribution $P(\Theta)$ over that joint-type space without having to communicate. Each robot i matches its true history of observations and actions, h_i^t, to one of the types in its type space Θ_i^t and then selects an action to execute based on its policy π^t and type Θ_i^t.

2.2 Hierarchical decomposition of decision processes

The second attempt battles the size of the problem with a hierarchical decomposition [13]. Also here the problem is modeled with a Bayesian Game Approximation, but only at a high conceptual level. On a lower conceptual level the problem is modeled as a much simpler Markov Decision Process (MDP), due to the strict computation limits.

The most important aspect of the agents' micro-level behavior is the exploration. Following the commonly employed practice among participating teams, the map is partitioned using K-Means into a number of sectors prior to the beginning of the simulation. Subsequently, agents get assigned to clusters based on their proximity and such that all clusters have an equal number of assigned agents. Inside the assigned cluster the agent starts to patrol and take (greedy) appropriate action when encountered.

Yet, the coordination between agents takes place at macro level. The Bayesian game is defined as a tuple $< I, \Theta, A, P(\Theta), U >$. The joint space Θ and the probability distribution $P(\Theta)$ are already introduced in section 2.1. In addition, there is the set of agents I and the joint actions A. Most design freedom for the Bayesian game is in the utility function U. Several choices for the utility function U were tried inspired by the utilities used by RoboCup Rescue competition [13].

The results of this approach were quite successful, with coordination on par with state-of-the-art distributed constraint optimization [5], while having less memory requirements. Yet, it also showed that selecting the right planning challenges is important to see the difference of cooperation between the team mates. Some problems are just too hard and all approaches will fail. Some problems are too easy and will be solved with each strategy. Only problems with the right amount of challenge should be used to evaluate the strength and weaknesses of algorithms. In the RoboCup Rescue this is taken into account by incorporating the variance in team performance into the score.

3 CONCLUSION

This two examples show the power of an online-planning approach such as Bayesian Game approximations. This way of modeling makes it possible to bridge the gap to coordination problems encountered in benchmarks as the RoboCup Rescue, such as distributed decision making based on incomplete information, with limits on the information that can be exchanged, including time lag to distribute this information over the team. Modeling that each robot has a certain amount of some private information, next to a certain amount of common knowledge, is a natural assumption.

REFERENCES

[1] Raymond Sheh, Sören Schwertfeger, and Arnoud Visser. 16 years of robocup rescue. *KI - Künstliche Intelligenz*, 30(3):267–277, Oct 2016.

[2] Carlos E Aguero, Nate Koenig, Ian Chen, Hugo Boyer, Steven Peters, John Hsu, Brian Gerkey, Steffi Paepcke, Jose L Rivero, Justin Manzo, et al. Inside the virtual robotics challenge: Simulating real-time robotic disaster response. *IEEE Transactions on Automation Science and Engineering*, 12(2):494–506, 2015.

[3] Sarvapali D. Ramchurn, Alessandro Farinelli, Kathryn S. Macarthur, and Nicholas R. Jennings. Decentralized coordination in robocup rescue. *The Computer Journal*, 53(9):1447–1461, 2010.

[4] Ranjit Nair, Milind Tambe, and Stacy Marsella. Team formation for reformation in multiagent domains like robocuprescue. In *RoboCup 2002: Robot Soccer World Cup VI*, pages 150–161. Springer, 2002.

[5] Paul Scerri, Alessandro Farinelli, Steven Okamoto, and Milind Tambe. Allocating tasks in extreme teams. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pages 727–734. ACM, 2005.

[6] Lucian Busoniu, Robert Babuska, and Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, And Cybernetics-Part C: Applications and Reviews, 38 (2), 2008*, 2008.

[7] Steven Balakirsky, Chris Scrapper, Stefano Carpin, and Michael Lewis. Usarsim: providing a framework for multi-robot performance evaluation. In *Proceedings of PerMIS*, 2006.

[8] Jacopo Banfi, Alberto Quattrini Li, Nicola Basilico, and Francesco Amigoni. Communication-constrained multirobot exploration: Short taxonomy and comparative results. In *Proceedings of the IROS Workshop on On-Line Decision-Making in Multi-Robot Coordination (DEMUR2015)*, pages 1–8, October 2015.

[9] Alain Caltieri and Francesco Amigoni. High-level commands in human-robot interaction for search and rescue. In *RoboCup 2013: Robot World Cup XVII*, pages 480–491. Springer, 2013.

[10] Frans A. Oliehoek and Arnoud Visser. *A Decision-Theoretic Approach to Collaboration: Principal Description Methods and Efficient Heuristic Approximations*, pages 87–124. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[11] Štefan Konecnỳ. Lossless clustering of multi-agent beliefs to approximate large horizon dec-pomdps. Master's thesis, Universiteit van Amsterdam, 2011.

[12] Rosemary Emery-Montemerlo. *Game-Theoretic Control for Robot Teams*. PhD thesis, The Robotics Institute, Carnegie Mellon University, 2005.

[13] Mircea Trăichioiu. Hierarchical decision theoretic planning for robcup rescue agent simulation. Master's thesis, Universiteit van Amsterdam, 2014.