The RoboCup Logistics League as a Benchmark for Planning in Robotics

Tim Niemueller and **Gerhard Lakemeyer**
Knowledge-based Systems Group
RWTH Aachen University, Aachen, Germany
{niemueller, gerhard}@cs.rwth-aachen.de

Alexander Ferrein
MASCOR Institute
FH Aachen, Aachen, Germany
ferrein@fh-aachen.de

Abstract

Planning in robotics as task-level executive is still an exception rather than the norm. Domains are often too dynamic or complex and therefore developers resort to more reactive or deliberative tools. In this paper, we characterize the RoboCup Logistics League (RCLL) as a medium complex robotics planning domain in terms of domain properties, implementation strategies, and planning models. We propose the RCLL in simulation and on real robots as a suitable testbed for a comparison of planning systems.

1 Introduction

Autonomous robots require a task-level executive, a software component that decides on the actions to take to achieve a certain goal. Typical approaches can be roughly divided in three categories: state machine based controllers like SMACH (Bohren and Cousins 2010) or XABSL (Loetzsch, Risler, and Jungel 2006), reasoning systems from Procedural Reasoning Systems (Alami et al. 1998a) or rule-based agents (Niemueller, Lakemeyer, and Ferrein 2013) to more formal approaches like GOLOG (Levesque et al. 1997), and finally planning systems with varying complexity and modeling requirements. There are also hybrid systems integrating aspects of more categories like integrating PDDL-based planning into GOLOG (Claßen et al. 2012).

Planning systems are still the exception rather than the norm in robotics applications, with notable exceptions like (Dornhege and Hertle 2013). Often domains are either too dynamic requiring prohibitively frequent decision points (e.g., robot soccer), or are highly complex imposing tedious modeling requirements to cover (a suitable subset of) the domain (e.g., domestic service robots). In this paper, we propose the *RoboCup Logistics League (RCLL)* as a suitable testbed for a wide variety of planning methodologies. It is a *medium complex domain* inspired by problems from in-factory logistic applications, where a group of robots has to maintain and optimize a material flow among processing machines and eventually deliver to an exit gate. We characterize the domain in terms of classic *domain properties* as a combined cooperative and competitive, dynamic, and continuous environment with partial observability and non-deterministic actions – but the latter two can be relaxed. We describe possible *implementation and modeling strategies* in terms of planning scope and horizon, output type, and distribution, and their implication on suitable *planning models*. The domain contains limited uncertainty in terms of orders posted at times unknown a priori, machines being out-of-order at some times, and due to robots of the other team.

A simulation (Fig. 1) of the competition (Zwilling, Niemueller, and Lakemeyer 2014) is available as open source software.[1] It supports high level abstraction that allows to keep the development and integration effort required to implement a task-level executive and a possible later transition to actual robots manageable even for small teams.

2 The RoboCup Logistics League

RoboCup (Kitano et al. 1997) is an international initiative to foster research in the field of robotics and artificial intelligence. It serves as a common testbed for comparing research results in the robotics field. RoboCup is particularly well-known for its various soccer leagues. In 2012, the new industry-oriented RoboCup Logistics League Sponsored by Festo (LLSF) was founded to tackle the problem of production logistics and renamed to RoboCup Logistics League (RCLL) in 2015. Groups of three robots have to plan, execute, and optimize the material flow in a smart factory scenario and deliver products according to dynamic orders. Therefore, the challenge consists of creating and adjusting a production plan and coordinate the group of robots (Niemueller et al. 2013a).

In 2015, the RCLL competition takes place on a field of $12\,\mathrm{m} \times 6\,\mathrm{m}$ partially surrounded by walls (Fig. 1). Compared to previous years, the competition now uses actual

[1] http://www.fawkesrobotics.org/p/llsf-sim/

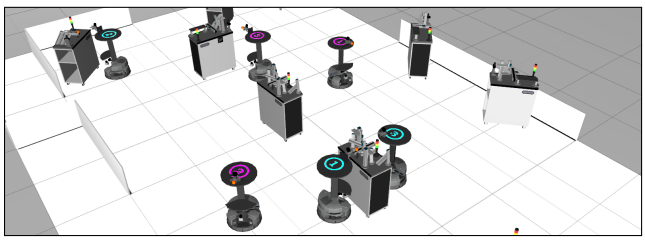

Figure 1: Simulation of the RCLL 2015.

production. Two teams are playing at the same time competing for points, (travel) space and time. Each team has an exclusive base and delivery station which dispense raw material and accept delivery of final products respectively. The goal is to refine the raw material through several processing steps and eventually deliver it, as shown in Fig. 2. The production machines are ring and cap stations. Ring stations each host two types of rings of specified color. A robot then takes an intermediate product to the machine, where it is processed by mounting one of the colored rings. Some rings require additional raw material to pre-fill the machine. Eventually, the cap station is used to mount a cap on the product before delivering it. The machines have a signal light to indicate the current status of a machine, such as "ready", "producing" and "out-of-order".

The game is controlled by the *referee box (refbox)*, a software component which keeps track of puck states, instructs the light signals, and posts orders to the teams (Niemueller et al. 2013b). After the game is started, no manual interference is allowed, robots receive information only from the refbox.

To facilitate rapid development we have created a *simulation* based on Gazebo (Zwilling, Niemueller, and Lakemeyer 2014). The updated simulation for the RCLL 2015 is currently underway and depicted in Fig. 1. It uses the refbox to provide the same environment agency as a real game and can be used to operate on several levels of abstraction, e.g. providing either laser sensor data for self-localization or the position of the robot with noise directly from the simulation. The simulation software for 2014 and 2015 is available as Open Source software.

3 Characterization of the RCLL

In this section we describe some properties of the RCLL domain and characterize potential planning strategies and assess how the RCLL could fit into common planning models.

3.1 Domain Properties

We describe the RCLL following the standard property matrix of (Russell and Norvig 2010) and some additional common properties of interest.

Cooperative vs. Competitive The RCLL is inherently a *multi-robot and multi-agent domain*. While the game can be played with a single robot, meaningful score can only be achieved by efficient cooperation of the robot group. In a cooperative setting, agents have common interest and try to maximize a common performance measure. In a competitive environment, agents have adversarial goals.

The RCLL scenario is *both*, cooperative and competitive at the same time. Robots of the same team directly cooperate on the task itself. Robots of opposing teams indirectly cooperate by avoiding collisions with each other and not blocking machines of the other team. The setting is still competitive, as both teams try to achieve the highest score. However, the robots compete for travel space only, i.e., while indeed avoiding collisions, there is no need to give way to another robot immediately. Other resources, like production machines, are exclusively assigned to one team and robots must not intentionally disturb each other.

Full vs. Partial Observability Describes if agents can observe all relevant aspects of the world through its sensors or only parts of these. In the latter case, a probability distribution over the possible observations outcomes is desirable.

The RCLL is inherently partially observable. Positions of opponent robots are generally unknown and can only be observed if close to a robot of the own team. Additionally, machine signals can only be read when in front of them, therefore a robot must approach it, for example, to check if a production has completed.

Deterministic vs. Non-Deterministic Determines whether the outcome of actions or observations is completely determined by the current state and the executed action. Actions are called *stochastic*, if a probability distribution over the non-deterministic action outcomes is known. A similar statement can be made about observations, i.e. for non-deterministic one must consider that sensor interpretation might be wrong or imprecise. For stochastic observations a probabilistic sensor model can be specified.

The RCLL is a non-deterministic domain. For example, a product might be lost on the way if the grasp was unstable or dropped when feeding it into a machine. A stochastic model is typically not known for the actions. However, for observations, like recognizing the signal light pattern, models might be determined empirically. Typically, sensor data is aggregated, for example, in a Bayesian form, to reach the desired certainty, balancing time and accuracy.

Episodic vs. Sequential In an episodic environment a robot's experience is divided into atomic episodes each not depending on the actions taken previously. In the sequential case previous action potentially influence future decisions.

The RCLL is clearly sequential, as every decision regarding the task structure, traveled routes, or resource assignment influences the remainder of the game and other robots.

Static vs. Dynamic Classifies whether the world does not change while the agent reasons about it, or if it does in the case of dynamic domains.

The RCLL is dynamic, as robots of the other team make their own independent choices. Even robots of the same team may make decisions while reasoning and the refbox triggers events of the environment, like finishing a production step or setting a machine out-of-order.

Discrete vs. Continuous Determines whether the state of the environment can be described strictly in discrete terms, or if continuous elements exist.

The RCLL can be modeled either way, but leans towards a continuous representation. The game has the concept of a continuous game time and robot motions are continuous. However, the time can be discretized, say, to seconds, as can the positions, for instance, as being at a certain machine, spot on the field, or on the move.

Known vs. Unknown If the environment is known, the outcomes for all actions are given.

The RCLL's environment is known, in particular because its agency is provided for by the referee box. Filling a product into a certain machine will produce a specified outcome depending on whether a valid production step was triggered.

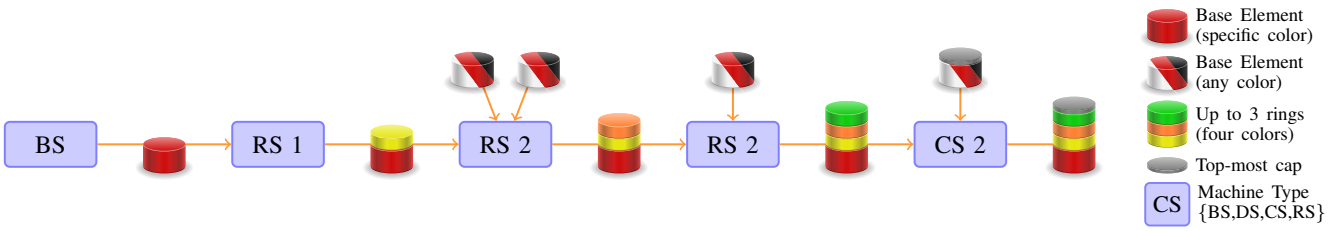

Figure 2: Refinement steps for production of highest complexity product in RCLL 2015 (legend on the right).

Implicit vs. Explicit Time Distinguishes durative actions with an explicit time span from actions where the duration is only given implicitly.

The times in the RCLL are mostly explicit. For example, a production step at a certain machine requires a certain amount of time. While this time is randomized, it is consistent within a single game and is specified to be in a certain range. It is generally impossible to correctly estimate the duration of actions, as for example, a moving robot might need to slow down to avoid a collision.

Finite vs. Infinite Domain Determines whether the domain of the planning problem is finite or infinite.

That depends on the modeling granularity chosen. It is possible to represent the domain with a finite domain for the general game. But some aspects require infinite domains, for example, if the time is modeled in a continuous fashion or actual robot positions are needed.

3.2 Strategy Characterization

The implementation of a planning and execution system for the RCLL might follow different strategies. They determine the choice of modeling, applicable tools and planners, and system requirements. The criteria are derived from (Alami et al. 1998b).

Global vs. Local Scope Determines whether to consider the overall fleet of robots or to limit reasoning and planning to a single robot.

The decision for the planning scope has a major influence in many regards. First, the granularity of planned actions typically varies. While on a global scope, planning would more likely be about task planning, that is, when is which robot to handle certain machines, but on a local scope individual actions like moving, or grasping a product will probably play a role. Additionally, it determines how much information must be communicated. For a global scope, all information must be available for the planner, either on a robot or on a central station. Then, the tasks must be assigned and communicated to the individual robots. On a local scope, robots need only to communicate the important information, and even if communication breaks down for a limited time (not unlikely in a RoboCup competition where wifi is used for communication), the robots can still continue operations, even though maybe not optimally. Also, a combined strategy is possible where a global task planner creates an optimal plan given the known information and generates jobs. These could be assigned to the individual robots, for example, by using an auction scheme, which in turn plan individually or employ an approach like plan merging (Alami et al. 1998b).

Complete vs. Incremental Planning Specifies whether to generate a full plan towards a final goal, or whether to plan for some specified horizon and incrementally extend the plan while executing it.

Current typical systems in the RCLL follow an incremental strategy where only the next best option is determined. If a planning system is used, longer planning horizons or even complete plans could be considered.

Centralized vs. Distributed Differentiates whether a centralized instance plans for all robots and communicates plans or if the robots plan for themselves and then coordinate.

In the RCLL, a central station is allowed to provide additional computing power for coordination and planning. But since communication is unreliable, this comes at the risk of not being able to communicate information or task assignments. At this time, all teams follow a distributed scheme. If more capable planning is introduced into the league, a centralized station might provide the required processing power.

Centralized planning could mean that a central station runs individual planners for each robot and then coordinates and distributes these plans. A centralized scheme does therefore not necessarily imply a global planning scope.

Sequential vs. Conditional Plans Determines whether the resulting plan is a sequence of actions or a policy.

For the RCLL, it seems more likely to generate a sequential plan and adapt it or replan if the plan is no longer executable, for example, because new information rendered it invalid. On the other hand, the number of decision points in the RCLL should be manageable when generating a policy.

3.3 Planning Models

A planning model (Geffner and Bonet 2013) depends on the domain properties (Sec. 3.1) and its application is influenced by the chosen planning strategy (Sec. 3.2). We will describe the major planning models and our expectation of the applicability for the RCLL. However, the domain has yet to be formally described and therefore some assumptions may not hold in the end.

Classical Planning Classical planning assumes full information about all parameters relevant at planning time and deterministic actions.

Obviously, in a robotics scenario like the RCLL simplifying assumptions are necessary for modeling. Actions do fail

occasionally and some information is simply not known, like machines being out-of-order, or processing and order times.

Approaches like Continual Planning – cf. (desJardins et al. 1999) or (Brenner and Nebel 2009) – which interleave planning and execution, can remedy these problems. Then, classical planners can be used to plan for certain run-time assumptions (e.g., all machines are available) and monitor plan execution to recover from failed actions or an unexpected world state by replanning.

Conformant Conformant planning allows to account for sensor feedback and uncertainty in the environment.

The RCLL should be a suitable domain for conformant planning, as the sensor feedback relevant to planning can be minimized to machine status feedback, e.g. signal light patterns. A conformant plan could thus plan for both possible situations at a machine, it can be working or out-of-order. In the former case the production will be performed, in the latter the planner might decide to move on an perform another production step first.

MDP or FOND If the environment but the actions are non-deterministic, the problem can be modeled as Markov Decision Processes (MDP) in a probabilistic setting, i.e., if expected action outcomes can be described by a stochastic model, a logical setting is described as Fully Observable Non-Deterministic (FOND).

As we described earlier, the RCLL is not per-se fully observable. But the domain description can be relaxed and augmented with assumptions that MDP or FOND planners should be applicable. For example, the planner can simply make assumptions about machine sensor feedback. A controller must then recognize if the machine is indeed out of order and adapt or replan.

POMDP or Contingent If an environment is only partially observable, actions are non-deterministic, and there is uncertainty ascribed to sensors, more expressive planning methods are required. Contingent planning extends the classical model with sensing, while partially observable MDPs (POMDP) require a stochastic sensor model.

This class of planners subsumes the previous models, but planners have to cope with a much increased complexity and thus typically require much longer planning times. The RCLL should provide enough complexity to justify the use of these models, but also not overwhelm the domain designer with details that have to be modeled.

4 Conclusion

As of this time, teams participating in the RCLL typically employ a local, incremental, and distributed strategy without planning. That is, robots collect information and classify the current situation and commit to the next action. An example is a CLIPS-based agent system (Niemueller, Lakemeyer, and Ferrein 2013) that collects information in a knowledge base and employs a rule-based reasoning system that models a hierarchical task structure to decide on the next action.

As a medium complex domain the RCLL provides a proper balance between required modeling effort, necessary planning model features, horizon, and complexity, and runtime. It is therefore an interesting planning domain allowing to compare planning systems in the context of multi-robot system in a semi-standardized environment providing limited amounts of uncertainty, partial observability, and nondeterminism in action execution. Especially, the availability of a simulation system reduces the initial development and integration effort.

References

Alami, R.; Chatila, R.; Fleury, S.; Ghallab, M.; and Ingrand, F. 1998a. An architecture for autonomy. *The International Journal of Robotics Research* 17(4).

Alami, R.; Fleury, S.; Herrb, M.; Ingrand, F.; and Robert, F. 1998b. Multi-Robot cooperation in the MARTHA project. *Robotics & Automation Magazine, IEEE* 5(1).

Bohren, J., and Cousins, S. 2010. The SMACH High-Level Executive. *Robotics Automation Magazine, IEEE* 17(4).

Brenner, M., and Nebel, B. 2009. Continual planning and acting in dynamic multiagent environments. *Autonomous Agents and Multi-Agent Systems* 19(3).

Claßen, J.; Röger, G.; Lakemeyer, G.; and Nebel, B. 2012. PLATAS – Integrating Planning and the Action Language Golog. *KI - Künstliche Intelligenz* 26(1).

desJardins, M. E.; Durfee, E. H.; Charles L. Ortiz, J.; and Wolverton, M. J. 1999. A Survey of Research in Distributed, Continual Planning. *AI Magazine* 20(4).

Dornhege, C., and Hertle, A. 2013. Integrated Symbolic Planning in the Tidyup-Robot Project. In *AAAI Spring Symposium - Designing Intelligent Robots: Reintegrating AI II*.

Geffner, H., and Bonet, B. 2013. *A Concise Introduction to Models and Methods for Automated Planning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool.

Kitano, H.; Asada, M.; Kuniyoshi, Y.; Noda, I.; and Osawa, E. 1997. RoboCup: The Robot World Cup Initiative. In *Proc. 1st Int. Conf. on Autonomous Agents*.

Levesque, H. J.; Reiter, R.; Lespérance, Y.; Lin, F.; and Scherl, R. B. 1997. Golog: A logic programming language for dynamic domains. *Journal of Logic Programming* 31(1-3).

Loetzsch, M.; Risler, M.; and Jungel, M. 2006. XABSL - A Pragmatic Approach to Behavior Engineering. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*.

Niemueller, T.; Ewert, D.; Reuter, S.; Ferrein, A.; Jeschke, S.; and Lakemeyer, G. 2013a. RoboCup Logistics League Sponsored by Festo: A Competitive Factory Automation Testbed. In *RoboCup Symposium 2013*.

Niemueller, T.; Lakemeyer, G.; Ferrein, A.; Reuter, S.; Ewert, D.; Jeschke, S.; Pensky, D.; and Karras, U. 2013b. Proposal for Advancements to the LLSF in 2014 and beyond. In *Proc. of 1st Workshop on Developments in RoboCup Leagues at IEEE ICAR*.

Niemueller, T.; Lakemeyer, G.; and Ferrein, A. 2013. Incremental Task-level Reasoning in a Competitive Factory Automation Scenario. In *AAAI Spring Symposium - Designing Intelligent Robots: Reintegrating AI*.

Russell, S. J., and Norvig, P. 2010. *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education.

Zwilling, F.; Niemueller, T.; and Lakemeyer, G. 2014. Simulation for the RoboCup Logistics League with Real-World Environment Agency and Multi-level Abstraction. In *RoboCup Symposium*.