Planning challenges in a cooperative tire changing task

Timothy Daly DALY@AXIOM-DEVELOPER.ORG
Laleh Roostapour LROOSTAP@CS.CMU.EDU
Scott E. Fahlman SEF@CS.CMU.EDU
Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA 15217 USA

Abstract

Our focus in the Tires project is to address the higher-level cognitive issues that arise in the course of human-robot interaction, so we are currently working with a simulated robot and a simulated environment. We ultimately want to couple our work to real robotic systems.

We examine some of the challenges of planning a robot task to change a tire in cooperation with a human. The robot has to form a plan which is robust in the face of external changes caused by human interactions. We need to know the details of the task as well as model the actions of the human.

Due to the large amount of information we need a way to store and efficiently search what we know. Scone, a Knowledge Representation System, has features which support many of the project goals. It is designed to efficiently scale to millions of entries, handle non-monotonic reasoning, and efficiently create virtual copies.

1. Introduction

The Tires project is investigating the problem of changing a tire with cooperation between human and a robot. The problem is complex and has many challenges. The robot has to form a plan which is robust in the face of external changes caused by human interactions. There has to be a way to model human beliefs so the robot can track, and possibly predict, the human actions.

Scone (Fahlman 2006,2011) has a general purpose multiple-context mechanism that will allow us to build and isolate a belief model. This context mechanism can be used to support a history of events that actually happen, which we call an episode.

We plan to develop a general purpose episodic memory mechanism that can be used for a range of other tasks. In particular, we would like to use the episodic memory to allow learning from actual events and to support dialogues.

Our focus in the Tires project is to address the higher-level cognitive issues that arise in the course of human-robot interaction, so we are currently working with a simulated robot and a simulated environment. We ultimately want to couple our work to real robotic systems.

We assume that we have a fully mobile, two-handed robot with a range of available sensors capable of recognizing any needed piece of information. We assume that the robot is capable of the required manipulations such as grasping or supplying sufficient force to lift a tire.

We assume that we are working in a well-modeled task environment such as that presented by the DARPA Robotics Challenge simulator. (Gazebo 2013) This will be used to constrain the range of planning challenges and to provide a testbed for validating generated plans. It has the further advantage of leveraging work by other groups that have solved issues such as path planning for the mobile robot. Simulation of the plan allows us to avoid the frame problem as we can test what might be changed by plan actions.

The Tires project uses the Scone Knowledge Base to manage the information required for the task. We build structures such as scripts and rules within Scone. We use Scone's hierarchical inheritance mechanism to place information at the proper level of generality.

In particular, we depend heavily on Scone's virtual copy mechanism to create an episodic memory mechanism. Episodes are created during execution and record a **course of action**, which is the actual set of steps executed from the general plan graph. Scone maintains a current **context** which allows us to reproduce the world state at any point in the path. This supports learning and a natural language dialogue mechanism.

There are many issues that arise in planning to change a tire, especially in managing the information that needs to be brought to bear on the plan. We need to know high level scripts such as the main steps in changing a tire as well as low level details about torque requirements for lug nuts. These ought to be arranged in a hierarchy of concerns but there are many level-crossing considerations, like the flashlight problem, which we mention below.

For purposes of explanation we break the system into stages of planning, execution, and learning. If there is sufficient time and resources on the actual robot platform these could all happen at the same time.

2. Planning

At plan time we use a blackboard to make goals, subgoals, facts, and the current plan graph globally available. Because there are many different processes searching for goals and modifying the plan graph it is important that the information is visible to all.

Constructing a plan to change a tire has many challenges. There is a lot of information that needs to be available to make decisions. For instance, which tool is chosen depends on the kind of lug nut which depends on the kind of wheel which depends on the vehicle. This information is added by different parts of the system and used by other, unconnected parts.

Goals are stated on the blackboard. A control process, the **hostess**, dynamically constructs a planner by selection **experts** which have specific information about how to achieve the goals in their area of expertise. The blackboard structure allow us to naturally expose level-crossing information without explicit function calls.

Actions that achieve the goals are organized into a rule-based format. Each rule has a set of preconditions required for the rule to be applicable, a list of actions which update the plan, and postconditions update the blackboard with the expected state after applying the rule.

2.1 Experts

The rules are organized into groups called **experts**. For instance, the rules about tires are in a tire expert. Rules about tools are in a tool expert. We use the Scone hierarchical inheritance mechansim to collect general tool rules with tool-specific rules (such as how to handle a socket set) in a subordinate rule set.

Rules about sensing inherit from a general sensor expert with specific information about what a particular sensor can provide. For instance, a vision sensor can add relative position while a kinect sensor can add motion and depth. Force sensors can handle low-level torque requirements.

The hostess process constructs a planner by searching for rules that can satisfy a goal posted on the blackboard. We use the Scone marker-passing based searches to find postconditions that fulfill all or part of the goal. That match causes not only the particular rule to be selected but the whole ruleset under the associated expert.

Selected rules are compiled into a RETE (Forgy 1982) so that applicable rules can be activated when their preconditions match blackboard facts. The RETE gives an efficient search to choose among hundreds of possible applicable rules, each contributed by different experts as part of their rule collections. These rule collections (experts) can all be compiled into a single efficient RETE structure to make the pattern matching efficient.

An example would be a computer-aided design (CAD) expert. The CAD expert would be able to answer questions about the number of lug nuts, the dimensions of the lug nuts, the opening size of the required tool. The rule set allows it to answer questions needed by other rules. A tire expert that knows about locking lug nuts, left-handed threads on some models of vehicles, wheel coverings such as hub caps and tire sizes and weights for planning grip points.

A global path plan expert can handle issues of moving from the trunk to the tire, avoiding obstacles such as a human, and ensuring that the vehicle is on level ground. This information is already available in some of the other DARPA Robotics Challenge work. A local path planner, as in Lozano-Perez (Lozano-Perez 1983), would consist of rules for placing lug nuts on studs, possibly using force compliance. Rules in this expert know about alignment requirements between studs, lug nuts, and lug wrenches.

In every case, these experts would have their set of rules added to the planner to deal with issues in their area of expertise. Thus, the planner is constructed as needed using only those rules which are required by the task at hand.

2.2 The plan graph

An expanding plan, called the plan graph, is maintained on the blackboard. The plan is a lattice of world states forming a partial order of actions that might occur. There is an implicit path through the plan graph if nothing goes wrong. However, since we are working in cooperation with a human it is likely that parts of the plan need to be skipped since the human achieved the goal. Alternatively, a different path has to be taken because the human changed the world state preconditions required for the next natural step.

In any case, once a step in the plan is chosen a rule will fire. As each rule is fired, the action portion of the rule is applied to the plan graph to modify one or more portions of the plan graph.

The postconditions need to be checked by sensors to ensure that the expected result of performing the action actually occurred.

Each state of the plan has a set of values that need to be sensed in order to apply the actions of the plan. This is necessary since the project assumes that the robot will be interacting with a human. The human could perform actions independently so we need to make sure that we remain aware of the changing world state and choose actions accordingly.

There are both local and global challenges to planning to change a tire. A local challenge might be that we do not have the proper tool to remove the tire. A global challenge might be that the task is being performed after dark so we need to use a flashlight. Unfortunately, the **flashlight problem** requires us to use at least one hand to handle the flashlight which might cause parts of the plan, such as lifting the tire with both hands, to require replanning.

3. Execution

During execution we maintain a **whiteboard** which contains the plan graph, information from sensors constantly updating the world state, and a **belief structure** of what we think the human has done or plans to do.

Each action that occurs is recorded as an event in an **episode**. Episodes are written as a series of Scone contexts. The context mechanism causes a virtual copy of the world to which we apply additions or deletions that record the actions. The total record in an episode is called a **course of action**.

3.1 Human Interaction

Since the human is an independent agent they might accomplish portions of the tire changing task. For instance, the human might use the lug wrench to loosen the lug nuts. We need to react to human actions in an appropriate and flexible manner. Koppula and Saxena (Koppula 2013) show modifying responses based on human actions.

If the human loosened the lug nuts we need to sense that situation and post it to the whiteboard. The preconditions of each step of the plan graph are in a RETE which drives off the facts posted on the whiteboard. A runtime sensor would post the fact that the lug nuts were loosened. This would cause the "next step" of the plan to skip ahead to a point in the plan that has a matching precondition.

There will be portions of the plan graph that reach a situation where the only possible "next step" action is to "ask the human." Here we need to have a way of constructing a natural language sentence about the goal we are trying to achieve. Assuming the human achieves the goal we need to sense the new world state and "rejoin the plan already in progress."

3.2 Human teaching

Another issue might be that the tool experts we know only have rules for handling a lug wrench. If the lug wrench is unavailable we can ask the human for help. The human could perform the step or could try to teach the robot how to cope with the problem.

Suppose the human shows us how to use a socket set. That involves choosing the appropriate socket, adding in any necessary extension segment, choosing a proper handle, and combining the results into a tool that fulfills the role of a lug wrench.

This **teaching event** is part of the episodic course of action. We will later try to extract rules which can be added to Scone, causing a permanent change of behavior. This might involve more human interaction, as detailed below.

4. Episodes and Learning

Episodes record what actually happened as a sequence of individual events. They have two uses, learning and dialogue.

The episodic course of action can be compared against the plan graph to provide feedback useful for learning. In the socket wrench example we can "clip out" the portion of the plan graph starting with the need for the lug wrench and ending with the action to loosen the lug nuts. We can compare that with the course of action actually taken which includes teaching by the human about how to use a socket wrench.

Learning involves modifying what we know in order to affect future behavior. We need to learn at several levels. We need to create Scone elements representing new concepts, such as a new kind of tool expert, as a subtype of the general tool expert. It involves learning preconditions, actions, and postconditions which are the fundamental parts of a rule. These are combined into rules, and collected into an expert.

Alternatively, a particular expert might already exist and we add a rule to augment what it knows. For instance, we already have a rule to "fix a tire" by "remove the old tire" and "attach a new one". However, we might add a rule to "inflate the tire".

In the socket wrench example, learning involves constructing a new socket wrench tool expert, a subtype of the general tool expert, which knows how to manipulate a socket set as an alternative to a lug wrench.

Learning occurs because subsequent plans would include rules for substituting a socket set for a lug wrench if the sensors indicate that the lug wrench is not available. Both of these plans would be part of the full plan graph, each one added by a tool expert.

4.1 Episodes and Dialogue

Episodes can be used to support a dialogue about the task. Every event in the course of action involves a new context which is a virtual copy of the prior world state with additions and deletions caused by an action.

By restoring a context we restore the world state at that point in the course of action. This makes it possible to ask the usual who, what, when, where, why, and how questions.

This implies that we have carefully constructed goals, actions, sensing information, and other world state with an eye toward making it possible to construct meaningful sentences.

We have done work to put a very constrained natural language front end on Scone. It uses the knowledge base for parsing by looking up Scone concepts and asking for their parts of speech. These

are matched against grammar rules for input. The grammar is highly constrained to essentially perform a prefix rewrite of the input, similar to an LL(1) parser. For instance,

```
A lug nut is a type of fastener.
```

translates to the Scone function call

```
(type-of {lug nut} {fastener})
```

(Note: the curly brace notation is used to indicate a Scone element.)

Similarly, the natural language input for questions is highly constrained to requires forms beginning with the who, what, when, where, why, and how questions. Particular functions are associated with each question to perform the required searches and formulate the reply. For instance,

```
When was the wheel removed from the axle?
```

would search the episodic course of action, find the Scone context where the goal

```
(remove {wheel} {axle})
```

was satisifed, and return the context

```
The wheel was removed from the axle in context {T74}.
```

5. Learning by Human Guidance

In the socket wrench example we construct a dialogue with a human to help us create rules with proper conditions. There are interesting questions about classifying knowledge. In the simple case presented we walk the knowledge network asking questions. Ideally we would already know the concept of a socket wrench from FrameNet (FrameNet 2013) or WordNet (WordNet 2013).

```
Tell the tire expert that
 when a lug nut is tight and a socket set is available
 do loosen lug nut with a socket wrench
 then lugnut is loose.

  Is lug nut a new concept? Yes
  Is {lug nut} a type of {thing}? Yes
  Is {lug nut} a type of {number}? No
  Is {lug nut} a type of {part}? Yes
  Is {lug nut} a type of {nut}? Yes

==>   (type-of {lug nut} {nut})

  Is {socket wrench} a type of {thing}? Yes
```

```
  Is {socket wrench} a type of {number}? No
  Is {socket wrench} a type of {part}? Yes
  Is {socket wrench} a type of {nut}? No
  Is {socket wrench} a type of {tool}? Yes
  Is {socket wrench} a type of {wrench}? Yes

==>   (type-of {socket wrench} {wrench})

==>   (new-rule :precondition ({fact} {lug nut} {tight})
                             (goal {loosen} {lug nut})
                             (tool {socket wrench} {loosen})
              :action (use-tool {socket-wrench} {lug nut})
              :postcondition ({fact} {lug nut} {loose})
              :expert {tire expert})
```

6. Summary

The Tires project demonstrates how various features of Scone can be usefully applied to solve parts of the tire changing task. For example, the hierarchical nature of Scone allows us to place information at the appropriate level of generality, so that an expert in using a specific tool can inherit rules from the more general tool expert.

Marker-passing provides an efficent search mechanism for finding experts (collections of rules) by searching for postconditions that match all or part of existing goals. This allows selection of rules and sets of rules (experts) which apply to the problem.

The virtual copy mechanism allows us to create whole copies of the world where we can apply additions and deletions, a form of non-monotonic reasoning. This allows us to make models of human behavior and conjecture many "what-if" scenerios without changing the rest of the knowledge base.

The context mechanism allows us to recreate particular situations in the course of action so we can learn new rules from differences. It also allows us to recreate a world state in order to support a dialogue answering who, what, when, where, why, and how questions.

Contexts also allow us to construct and maintain a belief network to capture what we know about human interactions, keeping them separate from the rest of the information perceived by the robot.

Acknowledgements

The work reported here was supported in part by the U.S. Office of Naval Research, under grant N000141310224. Any opinions, findings, conclusions, or recommentations expressed here are those of the author, and do not necessarily reflect the views of ONR or the U.S. government.

References

Fahlman, Scott E. (2006) *Marker-Passing Inference in the Scone Knowledge-Base System* KSEM 2006, LNAI3092, pp114-126.

Fahlman, Scott E. (2011) *Using Scone's multiple-context mechanism to emulate human-like reasoning* AAAI Fall Symposium on Advances in Cognitive Systems.

Forgy, Charles (1982) *Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem* Artificial Intelligence Vol 19, pp17-37.

FrameNet (2013) `http://framenet.icsi.berkeley.edu`

Gazebo (2013) *Gazeo Robot Simulator* `http://gazebosim.org`

Koppula, Hema and Saxena, Ashutosh *Anticipating Human Activities for Reactive Robotic Response* Cornell Personals Robotics. `http://pr.cs.cornell.edu/videos.php`

Lozano-Pérez, Tomás; Mason, Matthew T.; Taylor, Russell H. (1983) *Automatic Synthesis of Fine-Motion Strategies for Robots* MIT AI Lab, AI Memo 759.

WordNet (2013) `http://wordnet.princeton.edu`