Planning as an Architectural Control Mechanism

Nick Hawes
School of Computer Science
University of Birmingham, UK
n.a.hawes@cs.bham.ac.uk

Michael Brenner
Institut für Informatik
Albert-Ludwigs-Universität
Freiburg, Germany
brenner@informatik.uni-freiburg.de

Kristoffer Sjöö
Centre for Autonomous Systems
Royal Institute of Technology (KTH), Stockholm, Sweden
krsj@csc.kth.se

ABSTRACT

We describe recent work on PECAS, an architecture for intelligent robotics that supports multi-modal interaction.

Categories and Subject Descriptors
I.2.8 [Computing Methodologies]: ARTIFICIAL INTELLIGENCE: *Problem Solving, Control Methods, and Search*

General Terms: Design, Theory

Keywords: architecture, planning, robotics, integration

1. INTRODUCTION

An information-processing (IP) architecture designed to enable autonomous robots to interact intelligently with humans in multiple contexts must solve a number of problems. One of these is to unify the processing of multiple, concurrently active, heterogeneous subsystems into a single stream of intelligent behaviour. To do this the architecture must mediate between both the different *representations* used throughout the system and the *processes* in its multiple subsystems. Over the last couple of years we have been exploring designs for IP architectures for intelligent robots. In this work we have already addressed mediation between representations [1, 3]. In this paper we present recent work on the problem of mediating between processes.

2. BACKGROUND AND MOTIVATION

The following summarises some of the assumptions that underlie our work. We have been developing the **PECAS** architecture to fulfill the requirements of scenarios featuring situated dialogue coupled with table-top manipulation (our **P**layMate scenario) or semantic mapping (our **E**xplorer scenario). Our architecture is based on the CoSy Architecture Schema (**CAS**), which structures systems into *subarchitectures* (SAs) which cluster *processing components* around *working memories* [2]. From this schema we have created a number of SAs (vision, communication, navigation, manipulation etc.) which can be selectively grouped into a single architecture for a particular scenario. All these SAs are active in parallel, and all operate on SA-specific representations (as is necessary for robust and efficient task-specific processing). These disparate representations are unified by a *binding SA*, which performs abstraction and cross-modal information fusion on the information from the other SAs [3]. This gives us a way of mediating between heterogeneous content in our systems, but it does not say anything about how we can use this content in the generation of behaviour.

We have built a number of systems using PECAS, including interactive robots for table-top manipulation and for semantic mapping of indoor environments. These robots have multiple capabilities that can be used to perform many different user-specified tasks. In order to provide the robots with a generic and extensible way to deal with such tasks, we treat the computation and coordination of overall system behaviour as a *planning* problem. The use of planning gives the robot a high degree of autonomy: complex goal-driven behaviours need not be hard-coded into the system, but are planned by the robot itself. Likewise, the robot can autonomously adapt its plans to changing situations using *continual planning* and is therefore well suited to dynamic environments. However, relying on automated planning means that all tasks for the robot need to be posed as goals for a planner, and all behaviour to achieve these goals must be encoded as actions that the planner can process.

3. BEHAVIOUR AS PLANNING

Planning systems are given goals as logical formulae. In our architecture, such goals are typically generated when intentional content is processed by the communication SA (i. e. given by a human in natural language), then made available to the motivation SA via binding. For example, if the user wants the robot to bring them a certain book, this might lead the robot to form a goal such as `(holds human1 book1)`. Note that while goals are most commonly provided by a human, they can also arise from internal processes.

While the traditional use of planning is achieving goals in the physical world using physical actions, such direct interpretations of behaviour are the exception rather than the rule in human-robot interaction. Here, where information is incomplete, uncertain, and distributed over several agents and throughout subsystems, much of the actions to be performed by the system are to do with processing *information*. Whilst some IP may be performed continuously by the system (e. g. listening for sounds to recognise, SLAM) much IP is too costly to be performed routinely and should instead be performed only when relevant to the task at hand, i. e. it should be planned based on context.

4. PLANNING FOR IP

Underlying our approach to IP is the functionally decomposed, concurrently active, structure of PECAS. As each SA is effectively a self-contained processing unit, our design

HRI'09, March 11–13, 2009, La Jolla, California, USA.
ACM 978-1-60558-404-1/09/03.

leads naturally to an integration strategy: each SA is treated as a separate agent in a multi-agent planning problem. Although this separation has many features which we do not have space to discuss, a crucial one is that each SA's knowledge is separate within the planning state, and can only be reasoned about using epistemic operators (e. g. `(K vision.sa colour(obj1))`, meaning that the vision SA knows the colour of an object). Likewise, goals are often epistemic in nature, e. g. when a human or a SA wants to query the vision SA for the colour of an object.

To realise internal and external information exchange each SA can use two special actions: *tell-value* and *ask-value*. These provide and request information, respectively, and have epistemic effects. Interaction with humans or other external agents can also use (but is not limited to) these actions. As a result, planning of IP becomes a matter of planning for epistemic goals in a multiagent system. For example, if a human teacher tells our robot that "the ball is blue", this gives rise to the motivation that all SAs dealing with colours (e. g. vision) should know the colour of the ball in question. This may lead to a plan in which the communication SA uses a tell-value action to give the vision SA this information. Note that the factual information provided by the teacher is not directly entered into the robot's knowledge base. Instead it gives rise to a more complex motivation which enables the planner to initate more complex IP as necessary, e.g. triggering a colour learning process.

This design gives the robot more autonomy in deciding on the task-specific information flow through its subsystems. But there is also another assumption underlying this design: whilst the binding SA is used to share information throughout the architecture, not all information in the system can or should be shared this way. Some information is unavailable because it is modality specific, and even cross-modal knowledge is often irrelevant to the task at hand. If all information was shared this would overwhelm the system with (currently) irrelevant information (e.g. lists of all the people, rooms, objects, object categories etc. that parts of the system know about). Thus, in order to restrict the knowledge the planner gives "attention" to without losing important information, it needs to be able to *extend* its planning state on-the-fly, i. e. during the continual planning process. We call this process *task-driven state generation*.

To support this the planner makes use of meta-level information, so-called *produce* and *consume* facts. They describe which SAs can produce which predicates (i. e. where certain types of information can come from) and which SAs can consume which predicates (i. e. where certain types of information should go). This enables more general formalisations for, e. g., teaching goals (all SAs which consume a particular predicate should be told the value of any new instances of that predicate), and requesting information (if a SA needs the value of a state variable, then it should ask a SA that can produce it). Produce and consume facts provide the planner with enough information to use tell-value and ask-value in its IP planning. We assume that the SA-specific details for asking and telling can be left opaque to the planner and will be filled in at execution time by the executing SA. It is not obvious whether this assumption will be valid in all cases, but it provides a useful starting point.

5. EXAMPLES

In this section we will provide examples of our approach in action. In our mobile robotic scenario, where a robot interactively explores an office environment and runs simple errands for human users, task-driven state generation is used in several ways to help the robot deal with its necessarily incomplete knowledge. When the robot is given a command such as "Bring me the Borland book" the planner realises that in order to achieve this task it first needs to satisfy the epistemic subgoal of knowing where the book is. Thus, in the initial phases of the continual planning process, it will query SAs who (as specified by appropriate *produce* facts) can provide information about the location of the book. In this case, it is the conceptual mapping SA which can provide default knowledge about the locations of objects, e. g. the library in the case of books. Having updated the state with the information from conceptual mapping, more detailed planning becomes possible, allowing the robot to plan to move to the library to search for the book.

Essentially the same process is used for planning human-robot interaction. If the agent who is believed to be able to "produce" facts about the book location is not an internal SA but a human, the robot plans to *ask-value* the human. However, the preconditions for ask-value may vary for different addressees. In particular, external agents must first be approached and engaged in a conversation. Since the same planning approach is used for both physical actions as well as internal IP, the planner can directly initiate a situated dialogue including the physical movement of the robot.

In our table-top scenario the robot is capable of learning and recognising visual features such as colour and shape. If the confidence of a recognition result falls within a particular window of uncertainty (e.g likely but not certain), the robot can generate clarification behaviour. Clarification is represented as a goal in which the requesting SA should know the value of a particular predicate (e.g. the colour of an object). The plan created for this goal consists of the requesting SA (e.g vision) asking SAs which produce this predicate for its value (e.g. communication). If the communication SA is asked, this may result in the robot asking a nearby human for the information, and if an answer is provided to the robot, the information is made available via the binder. This highlights how our design allows the planner to operate without knowing the details of how each SA implements its responses to ask-value and tell-value actions.

6. CONCLUSION

We have presented work on using planning to allow a robot to coordinate multiple processes in PECAS, an architecture for intelligent robots. Our approach has been applied to different HRI scenarios, demonstrating its generality.

Acknowledgments

Supported by the EU integrated projects CoSy and CogX.

7. REFERENCES

[1] M. Brenner, N. Hawes, J. Kelleher, and J. Wyatt. Mediating between qualitative and quantitative representations for task-orientated human-robot interaction. In *IJCAI '07*, 2007.

[2] N. Hawes, A. Sloman, J. Wyatt, M. Zillich, H. Jacobsson, G.-J. Kruijff, M. Brenner, G. Berginc, and D. Skočaj. Towards an integrated robot with multiple cognitive functions. In *AAAI '07*, 2007.

[3] H. Jacobsson, N. Hawes, G.-J. Kruijff, and J. Wyatt. Crossmodal content binding in information-processing architectures. In *HRI '08*, 2008.