Theoretical and Experimental Results on the Goal-Plan Tree Problem

(Short Paper)

Patricia H. Shaw, Berndt Farwer, and Rafael H. Bordini
Department of Computer Science, University of Durham, Durham DH1 3LE, U.K.
{p.h.shaw,berndt.farwer,r.bordini}@durham.ac.uk

ABSTRACT

Agents programmed in BDI-inspired languages have goals to achieve and a library of plans that can be used to achieve them, typically requiring further goals to be adopted. This is most naturally represented by a structure that has been called a Goal-Plan Tree. One of the uses of such structure is in agent deliberation (in particular, deciding whether to commit to achieving a certain goal or not). This paper presents new experimental results combining various types of goal-plan tree reasoning from the literature.

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]: Intelligent Agents; D.2.2 [**Design Tools and Techniques**]: Petri nets

General Terms

Experimentation

Keywords

Autonomous Agents, Reasoning, Petri nets

1. INTRODUCTION

Agents programmed in BDI-inspired languages have goals to achieve and a library of plans that can be used to achieve them, typically requiring further goals to be adopted. This is most naturally represented by a structure that has been called a Goal-Plan Tree. Whilst no planning takes place in such agents, a certain type of reasoning – done over such representation of agents' commitments towards goals to be achieved and courses of actions to achieve them – can significantly impact the agent's performance by judicious scheduling of the plan execution. More importantly, it can significantly improve *deliberation*, in the sense that an agent can make reasoned choices on whether to commit to achieving a new goal or not.

In all the work by Thangarajah *et al.* [7, 8, 9, 6], a *goal-plan tree* is used to represent the structure of the various plans and subgoals related to each goal for an individual agent. At each node of the tree, *summary information* is used to represent the various constraints under consideration. This is similar to previous work by Clement and Durfee [1, 2, 3], using summary information with HTN planning to co-ordinate the actions of multiple agents. A different approach was introduced by Shaw and Bordini [5], where a goal-plan tree is mapped into a Petri net, thus avoiding the need for summary information.

To our knowledge, while Thangarajah *et al.* have reported on experimental results for the individual types of reasoning, no results appear in the literature showing what is the performance obtained when an agent is doing all those forms of reasoning simultaneously. It also remains unknown if their approach is still equally efficient when the various types of reasoning are combined and how it scales up, as the amount of summary information to handle could potentially grow exponentially with the size of the goal-plan tree [3], which could have a significant impact on the performance of the agent for larger problems.

The work in [5] considered reasoning about both positive and negative effects of a plan on other plans using a Petri-net based technique. In this paper we focus on reasoning about resources *combined* into a coherent reasoning process that also encompasses reasoning about positive and negative interactions, using Petri nets to do so. Whilst we here only use a Petri net approach to solving the goal-plan tree problem, we are currently investigating a number of alternative techniques.

The remainder of the paper is organised as follows. Section 2 summarises the main ideas of our formalisation of the goal-plan tree problem and its complexity (the actual formalisation was omitted due to lack of space), Section 3 shows the use of Petri nets to produce a combination of the various types of reasoning on goal-plan trees, Section 4 shows the experimental results and analysis of this approach to the goal-plan tree problem, and Section 5 concludes the paper.

2. THEORETICAL RESULTS (OUTLINE)

A goal-plan tree is a bipartite directed graph, connecting (sub)goals with plans, and plans with subgoals. An example of a goal-plan tree is shown in Figure 1, reproduced from [8][1]. In order for a plan within the tree to be completed, all of its subgoals must first be completed. However, to achieve a goal or a subgoal, only one of its alternative plans needs to be executed. The *goal-plan tree problem (GPT)* is the question of whether a schedule of execution exists that satisfies the pre- and post-conditions of each plan, achieving all top-level goals without running out of resources.

We have proved that GPT is NP complete by showing that $\mathrm{GPT} \in \mathrm{NP}$ and $\mathrm{3SAT} \leq_{\mathrm{p}} \mathrm{GPT}$.

Due to lack of space, we can only give a brief example showing the idea behind the reduction. The principle is that we have one resource for each variable. For instance, if there were n variables,

[1]Goals and subgoals are represented by rectangles, while plans are represented by ovals.

Cite as: Theoretical and Experimental Results on the Goal-Plan Tree Problem (Short Paper), Patricia H. Shaw, Berndt Farwer and Rafael H. Bordini, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. XXX-XXX.

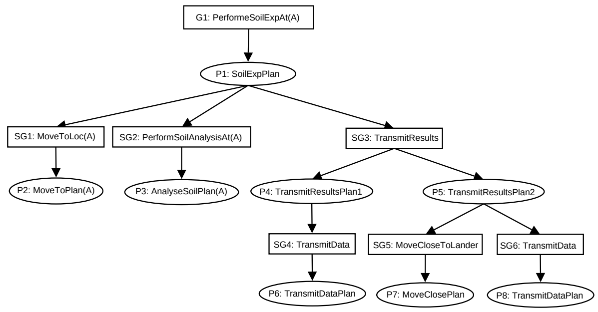
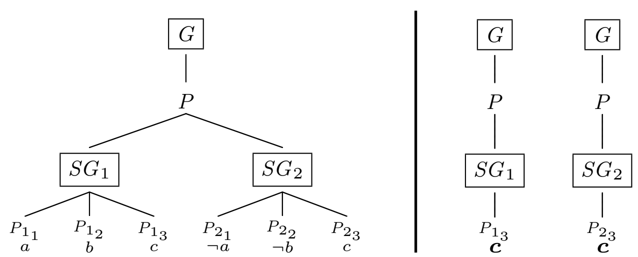


Figure 1: Goal-plan tree for a Mars rover [8].

then there would have to be n different types of resources, and one instance of each is available at the beginning. Plans correspond to the literals of the 3CNF formula. Plans for both α and $\neg\alpha$ use the same resource (named α), but plans for a different α' or $\neg\alpha'$ use a different resource (α'), etc. All plans have empty preconditions.

EXAMPLE 1. *Consider the formula $(a \vee b \vee c) \wedge (\neg a \vee \neg b \vee c)$. We construct the following goal-plan tree (left-hand side below):*



The right-hand side of the figure shows the two possible ESX-trees[2] associated with the consumption of resource c. Only one branch remains as the other one is truncated due to positive interaction. The leaves represent the literals whose resources have been used; that is, starting from the multiset of resources containing one instance each of a, b, and c, we need only use the resources for c (corresponding to c being assigned true*) in constructing a schedule of execution P_{1_3} or P_{2_3}. The total number of ESX-trees for the goal-plan tree on the left is six, leading to other schedules (e.g., P_{1_1}, P_{2_2} with the consumption of the resources a and b). Not using a specific resource means that any truth assignment for the propositional variable corresponding to that resource satisfies the formula.*

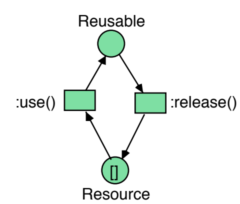
To show the NP-hardness of GPT, we have formally shown the polynomial-time reducibility of 3SAT to GPT. Furthermore, we have shown that GPT is actually in NP (again omitted due to space). The idea is to (non-deterministically) guess an ESX-tree for the input goal-plan tree and a sequence of the plans in it, with their individual pre-conditions, effects, and resource allocations, then check (in polynomial time) that this actually solves the goal-plan tree problem.

3. REASONING ABOUT RESOURCES

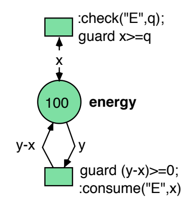
While in [5] we showed that it is possible to avoid the use of summary information when reasoning about positive and negative interactions using a Petri-net approach to the goal-plan tree problem, this is not possible when reasoning about resources. However, we have here minimised the use of summary information compared to the levels used in [9]. We only use (a compact form of) summary information where it is absolutely required and can give a significant improvement on the resource usage.

Summary information is used in two ways. First, a summary of all the resource requirements is produced and used to decide if a goal can be taken on, based on existing resource availability. Second, where a goal or subgoal has a choice of plans, summary information just for the subtrees is provided so as to select a preferred plan (i.e., the one with the lowest resource requirements).

There are two main classes of resources: reusable resources and consumable resources. An instance of a reusable resource can only be used by one plan at a given time, but when that plan has finished executing, the resource is available again for another plan to use it. A typical example of such a type of resource is a communication channel. On the other hand, consumable resources can only be used once, and then no longer exist, for example (units of) energy or time. Reusable resources can be represented as shown in Figure 2(a). However, the results in this paper refer to reasoning about *consumable* resources only; Figure 2(b) shows the basic representation of consumable resources. It uses a check function that is only able to fire (i.e., return "true") if there is at least a quantity q of that resource currently available.



(a) Reusable resource.



(b) Consumable resource.

Figure 2: Petri nets for the two main resource types.

When multiple different consumable resources are used, there can be two types of summary information, depending on the level of detail required. The first provides the detail splitting up the summaries based on the different resources, while the second gives the overall summary as a sum of the summaries for each resource.

The summary information can either be pre-processed (i.e., done off-line), or produced dynamically by generating Petri nets on-the-fly. Either way, the result is the same, and the summary information produced gives the *best case* and *worst case* resource requirements. These are the minimum and maximum resource requirements when taking into account goals or subgoals that have a choice of plans with different summary resource requirements.

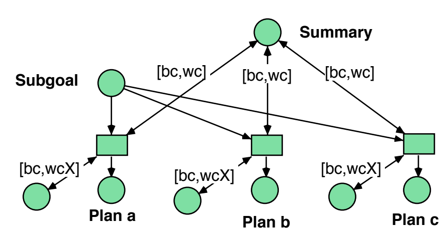


Figure 3: Selecting the best plan based on required resources.

The summary information is generated using the tree structure, summing up the requirements starting at the leaves. Where there is a choice of plans, the summaries for those plans are stored with the

[2]ESX-trees are obtained from goal-plan trees; only one plan to achieve each (sub)goal is selected; reasoning techniques for positive reasoning can further trim the resulting tree.

subgoal to aid the selection between the plans; see Figure 3. Here, the summaries for the different resources are accumulated together when calculating the summary information so that only a single number is stored for each branch, and the break down is passed on up the tree listing the best case and worst case depending on which branch is chosen. If some resources are required to be conserved more than others, weights could be added here to indicate an additional cost of using a particular resource, thus favouring the alternatives.

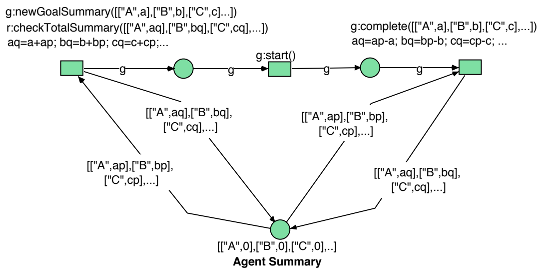


Figure 4: Checking the summary information.

After all goals have their summary information, the summary information at the root of the tree can then be used by the agent to decide whether it is safe to start acting towards achieving the goal in relation to the amount or resources it currently has available, and any other goals which the agent may be already committed to achieve. Figure 4 shows the Petri-net module used by the agent to check the summary information before starting a goal. It keeps a sum of the summary information for the goals that the agent is committed to achieving, so before starting a course of action to achieve a new goal, it checks that there are sufficient resources for the sum of existing goals and the summary from the new goal. If there is, then the goal is adopted and when the goal has been achieved, its summary information is removed from the summary for currently executing goals.

4. EXPERIMENTAL RESULTS

In [5], it was shown how Petri nets can be used for goal-plan tree reasoning, and results were given for reasoning about negative and positive interactions between goals within an agent, along with the combination of the two forms of reasoning. In this paper, our experiments are aimed at studying the effects of a third type of reasoning, that of reasoning about consumable resources, and we here show the results from combining this reasoning with the first two types. To our knowledge, no experimental results combining all three forms of reasoning have previously appeared in the literature.

The results on reasoning about negative and positive interactions used variations on the conditions between high and normal levels of interaction, along with the duration or impact of the interaction. This was tested in two scenarios: an abstract scenario and a more concrete example using a Mars Rover. The performance of the reasoning agent was compared against a “dummy” agent where no such reasoning was included.

In the Petri nets, negative interaction was simulated by having the goals using a common set of variables to store different values that were later required. If after a goal had set a variable, and before the goal had read the value, another goal changed the value stored in the variable, then the first goal failed. The duration that the value needed to be protected for was varied between normal and long, to stress test the reasoning.

Positive interaction was simulated by multiple goals having common effects, with the effects being represented by values assigned to variables. For multiple goals to be achieved, only one plan execution was then sufficient to produce the required effects. By setting the plans higher up the goal-plan tree (i.e., nearer the root goal) to interact with other goals, greater impact on the number of plans executed to achieve the same goals is obtained.

In both scenarios, and under all conditions, the reasoning agent significantly outperformed the dummy agent. The reasoning ensured that with reasoning about negative interaction, the agent was consistently able to achieve all the goals, and used significantly fewer plans when positive interaction was present. The two forms of reasoning were combined without any side effects to each other, and produced a reasoning agent that performed well under combined positive and negative interaction, achieving all goals and giving a high reduction in the number of plans executed.

4.1 Resources

To analyse the performance of the reasoning about resources, we have used the goal-plan tree shown in Figure 5. There are 30 goals using 5 different types of resources, with each goal using varying quantities of each resource. The level of resource availability was varied between *high*, *medium*, and *low* availability to test the performance of the reasoning under highly constrained conditions, and to compare the cost of reasoning against a dummy agent when both are able to achieve all goals. The number of goals achieved, plans executed, and the time taken were measured, with the timings being measured using the system time on an Apple MacBook 2.0 GHz dual core processor, 1GB memory, OS-X 10.4 running Java version 1.5 and Renew version 2.1 [4]. The actions within the plans were given an artificial duration of 50ms to simulate their execution.

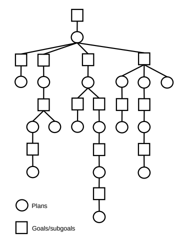
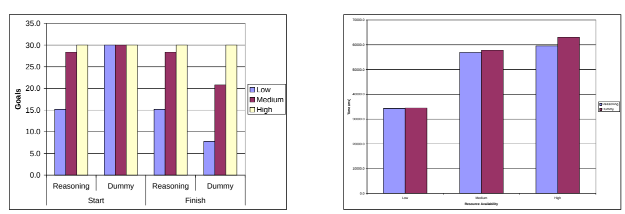


Figure 5: Goal-plan tree used to test reasoning about resources.

The results given below show the averages over 50 repeats of each experimental setup. This setup is equivalent to that used by Thangarajah *et al.* and was used to allow comparison to their approach [9].



(a) Number of goals started and finished under differing levels of resource availability.

(b) Time to achieve the goals under differing levels of resource availability.

Figure 6: Results of reasoning about resources.

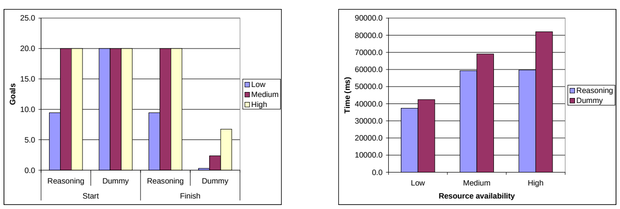
Figure 6 shows the results of reasoning about resources. While the dummy agent automatically started all the goals, the reasoning agent only started the goals it had sufficient resources available to achieve. As a result, the reasoning agent was able to achieve all the goals it started, while the dummy agent wasted a lot of resources on multiple goals, so it was generally only able to achieve a few goals, until there were sufficient resources available to achieve all goals (i.e., where there was high availability of resources).

The timings for the reasoning agent and the dummy agent in this scenario are very similar; however, it should be remembered that the reasoning agent achieves 36–96% more goals in the low and medium resource setting. In the high resource availability, it shows that the reasoning agent often takes the plan option with the fewer subgoals and plans, as these often also have lower resource requirements. Therefore, the dummy agent appears to take slightly longer, which also emphasises the minimal extra cost from the reasoning.

4.2 Combined reasoning

In this section, we show the results from the reasoning about resources being combined with the reasoning for positive and negative interactions. While it is possible with some approaches that this could cause side effects where the resource reasoning suggests one plan option to reduce the amount of resources used and the positive or negative reasoning suggests a different plan option, this was not the case with this approach, and the different reasoners were combined seamlessly.

The setup for the negative and positive interaction used here was the one for normal levels of interaction, with random duration and impacts. We assigned 20 top-level goals and the number of goals achieved, plans executed, and time taken were again measured.



(a) Number of goals started and finished under differing levels of resource availability.

(b) Time taken to achieve the goals under differing levels of resource availability.

Figure 7: Results from the combined reasoning.

Figure 7 shows the results from the combined reasoning. The effects of the positive interaction are highlighted in the reasoning with medium resource availability. While there would not normally be sufficient resources available to achieve all the goals, with less plans being required to achieve the goals, this has saved resources allowing all the goals to be achieved.

While even the dummy agent would normally be able to achieve all the goals with high resource availability, when the negative and positive interactions among the plans were added, it simply failed due to the negative interaction, even though there were some resources left at the end.

Due to the reduction in the number of plans that the reasoning agent was executing compared to the dummy agent, the time taken for the reasoning agent to achieve more goals is actually less than the dummy agent. This is emphasised where resource availability is higher, allowing the dummy agent to execute more plans in (failed) attempts to achieve goals. Remember, the dummy agent was attempting to execute plans to achieve goals even where the goals did not succeed due to running out of resources or negative interference.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented an alternative practical approach to reasoning about resources when an agent takes on multiple long-term goals to achieve. The GPT is of fundamental importance for any agent developed with a (BDI-like) agent-oriented programming language, yet so far the complexity of the problem was not known.

Our experimental results clearly show a significant improvement in the number of goals being achieved due to reasoning on goal-plan trees, with little impact in the time taken to perform the reasoning. While [5] showed results for combining positive and negative interactions, to the best of our knowledge, this is the first time that results for reasoning about positive and negative interactions between goals as well as reasoning about consumable resources (all performed in one combined process), have been obtained. The results also suggest a tractable subclass of instances, although the general problem is NP-hard. We aim at a formal characterisation of this class in future work.

The Petri nets used in our experiments have so far been produced manually, but their modular design provides scope for automating this process, so that it can be incorporated into an agent architecture for on-the-fly reasoning about new goals to be potentially adopted (note that this is important where agents can change their plan libraries at run time). Our long-term objective is to incorporate such reasoning into the interpreters of agent-oriented programming languages, and to experiment with various different techniques so as to find, experimentally, the conditions on the structure of the goal-plan tree where each works best.

6. REFERENCES

[1] B. J. Clement and E. H. Durfee. Identifying and resolving conflicts among agents with hierarchical plans. In *Proc. AAAI Workshop on Negotiation: Settling Conflicts and Identifying Opportunities, Technical Report WS-99-12*, pages 6–11. AAAI Press, 1999.

[2] B. J. Clement and E. H. Durfee. Theory for coordinating concurrent hierarchical planning agents using summary information. In *AAAI'99/IAAI'99: Proc. 16th Nat. Conf. on Artificial Intelligence and the 11th Innovative Applications of AI Conf.*, pages 495–502. Am. Assoc. for AI, 1999.

[3] B. J. Clement and E. H. Durfee. Performance of coordinating concurrent hierarchical planning agents using summary information. In *Proc. 4th Int. Conf. on Multi-Agent Systems (ICMAS'00)*, pages 373–374. IEEE Comp. Soc., 2000.

[4] O. Kummer, F. Wienberg, and M. Duvigneau. Renew – the Reference Net Workshop. Release 2.1. Available at: http://www.renew.de/, May 2006.

[5] P. Shaw and R. Bordini. Towards alternative approaches to reasoning about goals. In *Proc. 5th Int. Workshop on Declarative Agent Languages and Technologies (DALT'07), held with AAMAS'07*, pages 104–121. LNCS 4897, Springer, 2008.

[6] J. Thangarajah and L. Padgham. An empirical evaluation of reasoning about resource conflicts in intelligent agents. In *Proc. 3rd Int. Joint Conf. on Autonomous Agents and Multiagent Systems (AAMAS'04)*, pages 1298–1299, 2004.

[7] J. Thangarajah, L. Padgham, and M. Winikoff. Detecting and avoiding interference between goals in intelligent agents. In *Proc. 18th Int. Joint Conf. on Artificial Intelligence (IJCAI'03)*, pages 721–726. Morgan Kaufmann, 2003.

[8] J. Thangarajah, L. Padgham, and M. Winikoff. Detecting and exploiting positive goal interaction in intelligent agents. In *Proc. 2nd Int. Joint Conf. on Autonomous Agents and Multiagent Systems (AAMAS'03)*, pages 401–408. ACM Press, 2003.

[9] J. Thangarajah, M. Winikoff, and L. Padgham. Avoiding resource conflicts in intelligent agents. In F. van Harmelen, editor, *Proc. 15th European Conf. on Artifical Intelligence 2002 (ECAI'02)*. IOS Press, 2002.