AI Magazine Volume 22 Number 3 (2001) (© AAAI)

TOKENPLAN

A Planner for Both Satisfaction and Optimization Problems

Yannick Meiller and Patrick Fabiani

■ TOKENPLAN is a planner based on the use of Petri nets. Its main feature is the flexibility it offers in the way it builds the planning graph. The Fifth International Conference on Artificial Intelligence Planning and Scheduling planning competition validated its behavior with a GRAPHPLAN-like behavior. The next step is to demonstrate the benefits we expect from our planner in planning problems involving optimization and uncertainty handling.

Action planning is generally done in two stages: (1) building the search space and (2) searching it for a solution. The way work load is shared between these stages depends on the particular approach in use. Similarly to GRAPHPLAN (Blum and Furst 1997), TOKENPLAN alternates explicit stages of search-space expansion, with explicit stages of plan search and extraction. The particularity of our planner lays in the flexibility it offers in the way it builds the search space, which, in turn, leads to valuable consequences over the search itself. This flexibility is implemented using Petri nets as a basis for our representation.

Insights of TOKENPLAN cannot be detailed here, but the reader will find its description in full length in Fabiani and Meiller (2000). Instead, this article sketches out the approach of TOKENPLAN, along with the benefits we expect from it in planning problems involving optimization and uncertainty handling.

TOKENPLAN and Petri Nets

The reader unfamiliar with Petri nets needs only know the following: They are made of places, transitions, and tokens. A place can be seen as a token holder. When it contains one or more tokens, it is said to be marked. Transitions allow tokens to circulate from place to place. When all the places connected as input to a transition are marked, this transition can be triggered. When it is triggered, tokens flow through it toward places connected as output. The parallel with a STRIPS-like description of a planning domain appears clearly: Transitions correspond to operators, input places to preconditions, and output places to effects. TOKENPLAN starts by completing this exact conversion. All subsequent work considers the obtained Petri net representation.

For a state to be represented, relevant places are marked with tokens. These tokens hold on a label of the specific bindings of the variables of the predicate associated with the place. Starting from the initial marking and propagating the tokens through the net, we can list the markings that can be reached in one step, two steps, and so on. In fact, TOKENPLAN approximates this listing. Indeed, all markings reachable in one step are unified in one sole marking: They are superposed, and copies of identical tokens in identical places are deleted. By keeping track of the evolution of this marking from one step to the next, we obtain a leveled plan graph similar to GRAPHPLAN's. This plan graph is a compact representation of the search space. To extract a plan if any, a backward search is performed.

Of course, this search space is bigger than the actual space of reachable states because the set of states represented by each marking contains other states than the ones that are actually reachable. Indeed, some tokens, which appear in a same-level marking after the union of individual markings was completed, could not actually be where they are simultaneously otherwise. To make the approximation more accurate, pairwise mutual-exclusion relations

Copyright © 2001, American Association for Artificial Intelligence. All rights reserved. 0738-4602-2000 / $2.00

are computed. The same rules as the ones for computing GRAPHPLAN's mutex relations are applied (after being formulated in terms of tokens). However, part of the mutex relations, which we call *permanent mutex*, can be deduced with no costly computation using colored tokens. The idea is to use some "physical" properties of tokens to embed some structural properties of the planning domain. In some cases, TOKENPLAN can avoid the explicit computation of nearly half the mutex relations (nonubiquity of robots and balls in the gripper domain, for example).

Control over the Level of Splitting of the Search Space

In a particular level of the plan graph, when TOKENPLAN triggers a transition, it means it considers triggering it in all markings where its preconditions are marked. Similarly to GRAPHPLAN, TOKENPLAN always considers the largest sets of states that can be described in the domain language.

The particularity of TOKENPLAN is that it can achieve both levels of splitting, and also all the intermediate ones, using different classes of tokens.

An FSS-like approach, however, would consider systematically every single state independently, checking whether a particular action can be applied to it. In this case, the search space is said to be totally split, whereas in a GRAPHPLAN-like approach, there is no splitting at all.

The particularity of TOKENPLAN is that it can achieve both levels of splitting, and also all the intermediate ones, using different classes of tokens. Two tokens of different classes are different tokens, even though they might hold identical labels, and they might mark a same place. As different tokens, both of them will be kept in the planning graph, which introduces splitting. Moreover, to be triggered, a transition needs to have all its precondition places marked with tokens of the same class, ensuring different pieces of search space will not be mixed together.

If there is only one class, then TOKENPLAN has a GRAPHPLAN-like behavior: As soon as its precondition places are marked, a transition can be triggered, considering all the possible states where these preconditions hold. Suppose now that instead of a single class of tokens, a new one is created each time a transition is triggered. Then each class corresponds to a single state, and TOKENPLAN builds a search space in the same way FSS would, introducing full splitting. Of course, lots of different class designs are possible, leading to lots of intermediate splitting strategies. Classes can be designed according to the presence of such or such a feature but also according to some heuristics, or even some thresholds of uncertainty or utility for example.

Expected Benefits of This Flexibility

We are interested in planning under uncertainty in dynamic environments for autonomous systems. Our approach aims at making game-theoretic approaches more efficient by taking advantage of the capabilities of classical AI planning to handle sets of states instead of single states. The challenge is then to be able to handle sets of states relevant to the criteria that are to be optimized by the game-theoretic approach, that is, to be able to control the level of splitting of the search space.

No-splitting approaches are rarely adapted to optimization because the sets of states they handle are not necessarily coherent in regard to the criteria that are to be optimized. For example, two identical states will be unified, although they might not have the same utility value because of the way they have been reached.

Total splitting approaches, based on a systematic discretization of the state space, such as in Markov decision processes, can lead to an excessive fineness of description and an excessively large state space, which, in turn, makes any search in them untractable.

The idea behind controlling the level of splitting is to be able to design at planning time sets of states that are tailored for the planning task at hand. For example, classes might correspond to utility regions in an optimization problem. Then the search for a plan could be conducted at the level of the classes instead of at the state level. It would be equivalent to say that the initial systematic discretization is abstracted up to the proper level to solve the particular problem at hand. Another way of using classes could be to limit the search space to some of them. For example, each class could correspond to a different possible initial state. Then, a plan involving only tokens pertaining to all classes would be a *conformant plan*. A plan containing actions that depend on the class of tokens would be a *contingent plan*.

Settings for the Competition

During the competition, we used settings corresponding to a GRAPHPLAN-like behavior (that is, no splitting at all) because of TOKENPLAN's state of development at the time. Therefore, automatic color encoding of part of the mutex relations was used. The result of the competition validated TOKENPLAN's "good" behavior

please join us!

■ member's electronic library ■ national conference on artificial intelligence ■ discounts on ai books ■ innovative applications of artificial intelligence ■ discounts on journals ■ robot exhibition ■ spring symposium series ■ aaai press ■ fall symposium series ■ botball tournament ■ ai magazine ■ classic paper award ■ aaai fellows ■ allen newell award ■ distinguished service award ■ mobile robot competition ■ ai topics website ■ technical reports ■ grants for workshops, conferences, and symposia ■ electronic directory ■ online ai journal

american association for artificial intelligence
445 burgess drive ■ menlo park, california 94025 usa
www.aaai.org ■ membership@aaai.org
650-321-4457 ■ 650-321-4457 fax

compared to existing state-of-the-art approaches before we designed new splitting strategies. This version was implemented in Lisp, with a backward search adapted from the code kindly provided by S. Kambhampati.

Conclusion and Future Works

This planning competition showed that recent advances in AI planning such as GRAPHPLAN-like approaches can be handled properly by TOKENPLAN. With these settings, its performances position it in the main stream of current planning systems. However, the main benefits of TOKENPLAN are expected with other types of strategy for splitting the search space. We sketched out how these could be designed to be more suitable to uncertainty handling and optimization in planning.

Our current work focuses on this last point. We want to describe more formally the notion of splitting of the search space. On the experimental side, diverse splitting strategies have to be put through the test, first on deterministic classical planning benchmarks and then on planning problems dealing with uncertainty or optimization. These tasks will aim at getting a better feeling of the influence of the level of splitting over planning performances. Eventually, we aim at getting some characterization of splitting strategies and the problem classes they should be applied to.

References

Blum, A., and Furst, M. 1997. Fast Planning through Planning Graph Analysis. *Artificial Intelligence* 90:281–300.

Fabiani, P., and Meiller, Y. 2000. Planning with Tokens. Paper presented at the Recent Advances in Planning Workshop at the Fourteenth European Conference on Artificial Intelligence, 20–25 August, Berlin, Germany.

Yannick Meiller has started his Ph.D. program at SupAéro and works at ONERA on motion and information gathering planning under uncertainty under the supervision of Patrick Fabiani. His master's thesis dealt with modular neural command for behavioral mobile robotics (under the supervision of B. Amy, IMAG Institute, 1998). Finally, he had the opportunity to work on focusing GRAPHPLAN with S. Kambhampati (Arizona State University) in 1997. His research interests include planning, reasoning under uncertainty, optimization, knowledge representation, and autonomous agents. His e-mail address is meiller@cert.fr.

Patrick Fabiani graduated as Ingénieur from Ecole Polytechnique in 1990 and then from SupAéro in 1992. He obtained his Ph.D. from SupAéro in 1996 on the design of observation strategies for a surveillance system in a dynamically changing uncertain world. He is a senior research scientist at ONERA in Toulouse and has been working mostly on autonomous systems (helicopters, UAVs, UUVs, and so on) and more recently on airport ground traffic control. In 1997 to 1998, he was in the Robotics Lab in the Computer Science Department at Stanford University as a visiting scholar in J.-C. Latombe's research group. His research interests include decision, planning, reasoning under uncertainty, situation assessment, and autonomous agents. His e-mail address is fabiani@cert.fr.