MBP: a Model Based Planner

Piergiorgio Bertoli[1], **Alessandro Cimatti**[1], **Marco Pistore**[1], **Marco Roveri**[1,2], **Paolo Traverso**[1]

[1] ITC-IRST, Via Sommarive 18, 38055 Povo, Trento, Italy
{bertoli,cimatti,pistore,roveri,traverso}@irst.itc.it
[2] DSI, University of Milano, Via Comelico 39, 20135 Milano, Italy

Abstract

The Model Based Planner (MBP) is a system for planning in non-deterministic domains. It can generate plans automatically to solve various planning problems, like conformant planning, planning under partial observability, and planning for temporally extended goals. MBP is based on Symbolic Model Checking techniques, and Binary Decision Diagrams (BDDs), that provide a practical solution to the problem of dealing with the large size of realistic planning problems, and especially of problems in non-deterministic domains.

1 Introduction

The Model Based Planner (MBP) is a system designed to do planning in non-deterministic domains. Its two main characteristics are the following.

1. It provides a general framework for dealing with different classes of planning problems in non-deterministic domains.
2. It provides planning algorithms that can deal effectively with large state spaces.

MBP assumes a simple but general model of non-determinism, which encompasses uncertainty on the initial situation, on the action effects, and on the state in which the actions will be executed. Given a non-deterministic planning domain, MBP can plan for different kinds of goals. Intuitively, the problems that can be tackled by MBP can be classified according to the following two dimensions. The first is the degree of observability, i.e. what information can be gathered, at run-time, on the state of the domain. We can range from *full observability*, where the state of the world is assumed to be completely observable at run-time, to *null observability*, when no information is available at run-time, to *partial observability*, where only part of the domain information is available at run time. The second dimension is the expressiveness of the goals. Planning problems can range from the case where a set of final states must be reached (with different guarantees of achievement), to the more general case of *temporally extended goals*, i.e., where goals express conditions on whole sequences of states resulting from the execution of a plan.

Currently, MBP can tackle a wide spectrum of problems in this space: planning for temporally extended goals under conditions of total observability, reachability under null observability (the case of conformant planning [Smith and Weld, 1998]), reachability under partial observability. Depending on the kind of problem being tackled, MBP can generate plans of different form: sequential plans, conditional plans, iterative trial-and-error strategies, and plans that take into account the previous execution history.

MBP is based on the *Planning via Symbolic Model Checking* paradigm, first introduced in [Cimatti *et al*., 1997; 1998b; 1998a] (see also [Giunchiglia and Traverso, 1999] for an introduction). According to this idea, planning is done by using Symbolic Model Checking techniques, i.e., by searching through a finite state automaton whose sets of states are represented symbolically as propositional formulae. Search through the state space is performed as a set of logical transformations over propositional formulae. MBP is based on Reduced Ordered Binary Decision Diagrams (in the following simply called BDDs) [Bryant, 1992; 1986], that allow for the compact representation and effective manipulation of propositional formulae. The use of Symbolic Model Checking techniques provides the basis to deal effectively with large state spaces. The MBP planning algorithms are specialized for the different planning problems. For instance, algorithms for planning under the hypothesis of full observability perform a breadth-first search, while the algorithm for planning under partial observability performs a depth-first search through an and-or graph. MBP is available at `http://sra.itc.it/tools/mbp/`.

The paper is structured as follows. In Section 2 we provide a brief overview of the symbolic model checking techniques used in MBP. In Section 3 we discuss briefly the language that is currently used in MBP to describe planning domains, and how it is compiled to a (symbolic representation of a) finite state automaton. Then, we describe the MBP algorithms that can solve: planning under full observability for reachability goals (Section 4) and for temporally extended goals (Section 5), conformant planning (Section 6), and planning under partial observability (Section 7).

2 Planning via Symbolic Model Checking

Model Checking is a formal verification technique based on the exhaustive exploration of finite state automata [Clarke *et al*., 1999]. *Symbolic* Model Checking [McMillan, 1993] is a particular form of model checking, where propositional formulae are used for the compact representation of finite state automata, and transformations over propositional formulae provide a basis for efficient exploration. The use of symbolic techniques allows for the analysis of extremely large systems [Burch *et al.*, 1992]. As a result, symbolic model checking is routinely applied in industrial hardware design, and is taking up in other application domains (see [Clarke and Wing, 1996] for a survey).

MBP is built on top of the symbolic model checker NUSMV [Cimatti *et al.*, 2000], and relies on the powerful machinery of BDDs. BDDs provide a general interface that allows for a direct map of the symbolic representation mechanisms, i.e., propositional formulae representing sets of states and transformations over propositional formulae. MBP, as NUSMV, uses one of the best BDD packages available, the CUDD [Somenzi, 1997]. A BDD package deals with a single multi-rooted directed acyclic graph (DAG) representing a BDD. Memory efficiency is obtained by using a "unique table", and by sharing common subgraphs between BDDs. The unique table is used to guarantee that at each time there are no isomorphic subgraphs and no redundant nodes in the multi-rooted DAG. Before creating a new node, the unique table is checked to see if the node is already present, and only if this is not the case a new node is created and stored in the unique table. The unique table allows to perform the equivalence check between two BDDs in constant time (since two equivalent functions always share the same subgraph) [Brace *et al.*, 1990; Somenzi, 1997]. Time efficiency is obtained by caching the results of recently computed transformations, thus avoiding the recomputation.

3 MBP Domain Description Language

MBP is a two-stage system. In the first stage, discussed in this section, an internal BDD-based representation of the domain is built. In second stage, discussed in the next sections, different planning algorithms can be applied to the specified planning problems. These algorithms operate solely on the automaton representation, and are completely independent of the particular language used to specify the domain.

MBP planning domains can be described as nondeterministic labelled transition systems, where actions are modeled as transitions from states to sets of states. Planning domains are currently described by means of the high-level action language $\mathcal{AR}$ [Giunchiglia *et al.*, 1997]. $\mathcal{AR}$ allows to specify conditional effects and uncertain effects of actions by means of high level assertions. Non-boolean propositions are allowed. The semantics of $\mathcal{AR}$ yields a serial encoding, i.e. exactly one action is assumed to be executed at each time. The automaton corresponding to an $\mathcal{AR}$ description is obtained by means of the minimization procedure described in [Giunchiglia *et al.*, 1997]. This procedure solves the frame problem and the ramification problem, and is implemented as described in [Cimatti *et al.*, 1997]. Because of the separation between the domain construction and the planning phases, MBP is not bound to $\mathcal{AR}$. Parallel encodings have been investigated in [Cimatti *et al.*, 2001b], where planning techniques are applied to hardware circuits described in SMV language. The use of the $\mathcal{C}$ action language [Giunchiglia and Lifschitz, 1998], which allows to represent planning domains with parallel actions, and the representation of standard deterministic domains specified in PDDL, are also under investigation.

4 Weak, Strong and Strong Cyclic Planning under Full Observability

In nondeterministic domains, a plan is associated with many possible executions. Therefore, a plan can achieve a goal in different ways [Cimatti *et al.*, 2001a]:

- the plan has a chance of success, i.e. some of its exections achieve the goal (we call this case "weak planning");
- the plan is guaranteed to achieve the goal for all possible executions ("strong planning");
- the plan reaches the goal with an iterative trial-and-error strategy ("strong cyclic solutions"), such that all the associated executions always have a possibility of terminating and, when they do, they achieve the goal.

In the case of full observability, MBP can be asked to generate a plan for reaching a goal in one of the above forms. In all cases, MBP exploits the assumption of full observability by generating iterative and conditional plans that repeatedly sense the world, select an appropriate action, execute it, and iterate until the goal is reached. The generated plans are *state-action tables*, that associate to a set of states the action that has to be executed. They resemble universal plans [Schoppers, 1987] and policies [Bonet and Geffner, 2000].

The MBP algorithms for weak and strong planning share a similar control structure, based on a breadth-first search proceeding backwards from the goal, towards the initial states. At each step, they incrementally build the BDD corresponding to a state-action table that either have a chance (in the case of weak planning) or are guaranteed (in the case of strong planning) to lead to the current goal. At each iteration step, loops are eliminated and the set of states for which a solution has been already found is used as a target for the next iteration step. MBP provides two different algorithms for strong cyclic planning, which can be applied in different situations. The first algorithm is based on a global approach, which requires the analysis of the whole state space in one shot. The second, based on a local approach, tries to construct the state-action table incrementally from the goal, building on the strong planning algorithm. The work on weak, strong, and strong cyclic planning is described in [Cimatti *et al.*, 1998b; 1998a; Daniele *et al.*, 1999; Cimatti *et al.*, 2001a].

5 Planning for Temporally Extended Goals under Full Observability

MBP takes in input temporally extended goals, that express conditions on the whole executions associated to the solution plan (rather than conditions on the final states). Extended goals are formulated as CTL formulae [Emerson, 1990]. CTL can express temporal conditions that take into account the fact that an action may non-deterministically result in different outcomes. Hence, extended goals allow us to distinguish between temporal requirements on "all the possible executions" and on "some executions" of a plan.

In the general case of extended goals, MBP generates plans that are strictly more expressive than memory-less plans. Besides expressing conditional and iterative behaviors, the generated plans can execute different actions is a state, depending on the previous execution history. This expressiveness is required to deal with extended goals.

The planning algorithm builds an automaton corresponding to the CTL formula, that is then used to guide the search for a solution. Weak, strong, and strong cyclic planning can be expressed in terms of CTL formulae, and the algorithms described in previous section can be seen as specialized versions of the general case. The work on planning for temporally extended goals is described in [Pistore and Traverso, 2001].

6 Conformant Planning

Conformant planning is based on the hypothesis of null observability, i.e. no information can be gathered at run time. A solution to a conformant planning problem is a sequence of actions that is guaranteed to achieve the goal regardless of the uncertainty in the initial condition and in the nondeterministic effects of actions. Since it is not possible to acquire information at run-time, search must be carried out in the space of belief states, i.e. the powerset of the space of states of the planning domain.

In MBP, conformant planning can be carried out with two different approaches, both based on the representation of a belief state by means of a BDD. In the first approach [Cimatti and Roveri, 1999; 2000], a breadth-first search is carried out in the belief space, where the frontier is represented symbolically with the introduction of additional BDD variables. These are used to represent symbolically the plan associated with each belief state. In the second approach [Bertoli *et al.*, 2001a], search is performed in the belief space, integrating the symbolic model checking techniques of the first approach with a heuristic-style search. The first approach is also called *fully-symbolic*, while the second is called *semi-symbolic*, in order to emphasize the combination of explicit-state and symbolic search techniques. Conformant planning is also closely related to the problem of finding synchronization sequences in hardware circuits [Cimatti *et al.*, 2001b].

7 Planning under Partial Observability

In the case of partially observable domains, MBP allows for an extended version of $\mathcal{AR}$, that provides the user with the ability to specify the observation variables, i.e., the variables whose values can be observed at run-time, during execution. MBP allows the user to specify under which conditions a variable is observable. Observation variables can be either *action-independent* or *action-dependent*. The former ones are observable in every state after the execution of any action of the domain. Intuitively, they provide useful information automatically. Action-dependent observation variables can be observed only after the execution of some actions. As a result, MBP allows for both observations resulting from the execution of sensing actions [Pryor and Collins, 1996; Weld *et al.*, 1998] and automatic sensing that depends on the current state of the world [Tovey and Koenig, 2000].

Compared to conformant planning, the structure of the plan generated by MBP is no longer sequential, but tree-shaped, in order to represent a conditional course of actions. MBP generates conditional plans that branch on conditions on the value of observable variables. The planning algorithm implemented in MBP explores an and-or graph induced by the domain. The nodes of the graph are sets of states (the so-called "belief states" [Bonet and Geffner, 2000]). Or-nodes correspond to alternative actions that can be executed, and And-nodes correspond to different branches resulting from observations. The algorithm is basically a postorder traversal of the search space, proceeding forward from the initial belief state, and ruling out cyclic plans. The work on planning via symbolic model checking under partial observability is described in [Bertoli *et al.*, 2001b].

8 Conclusions and Future Development

MBP deals effectively with several kinds of planning problems in non-deterministic domains by using Symbolic Model Checking techniques. The experimental evaluations reported in [Cimatti *et al.*, 2001a; Bertoli *et al.*, 2001b; Pistore and Traverso, 2001; Cimatti and Roveri, 2000; Bertoli *et al.*, 2001a] show that MBP deals in practice with problems of significant complexity, and most often performs dramatically better than state-of-the art planners for non-deterministic domains.

Future work includes an extension of MBP to the general case of planning for temporally extended goals under partial observability, and to the case of planning with non-deterministic/noisy sensing. We also plan to consider the use of techniques based on heuristic search and POMDP planning.

References

[Bertoli *et al.*, 2001a] P. Bertoli, A. Cimatti, and M. Roveri. Heuristic Search + Symbolic Model Checking = Efficient Conformant Planning. In *Proc.* 7^{th} *International Joint Conference on Artificial Intelligence (IJCAI-01)*. AAAI Press, August 2001.

[Bertoli *et al.*, 2001b] P. Bertoli, A. Cimatti, M. Roveri, and P. Traverso. Planning in Nondeterministic Domains under Partial Observability via Symbolic Model Checking. In *Proc.* 7^{th} *International Joint Conference on Artificial Intelligence (IJCAI-01)*. AAAI Press, August 2001.

[Bonet and Geffner, 2000] B. Bonet and H. Geffner. Planning with Incomplete Information as Heuristic Se arch in Belief Space. In S. Chien, S. Kambhampati, and C.A. Knoblock, editors, 5^{th} *International Conference on Artificial Intelligence Planning and Scheduling*, pages 52–61. AAAI-Press, April 2000.

[Brace *et al.*, 1990] K.S. Brace, R.L. Rudell, and R.E. Bryant. Efficient Implementation of a BDD Package. In *27th ACM/IEEE Design Automation Conference*, pages

40–45, Orlando, Florida, June 1990. ACM/IEEE, IEEE Computer Society Press.

[Bryant, 1986] R. E. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, August 1986.

[Bryant, 1992] R. E. Bryant. Symbolic Boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys*, 24(3):293–318, September 1992.

[Burch *et al.*, 1992] J. R. Burch, E. M. Clarke, K. L. McMillan, D. L. Dill, and L. J. Hwang. Symbolic Model Checking: 10^{20} States and Beyond. *Information and Computation*, 98(2):142–170, June 1992.

[Cimatti and Roveri, 1999] A. Cimatti and M. Roveri. Conformant Planning via Model Checking. In Susanne Biundo, editor, *Proceeding of the Fifth European Conference on Planning*, Lecture Notes in Artificial Intelligence, Durham, United Kingdom, September 1999. Springer-Verlag. Also ITC-IRST Technical Report 9908-01, ITC-IRST Trento, Italy.

[Cimatti and Roveri, 2000] A. Cimatti and M. Roveri. Conformant Planning via Symbolic Model Checking. *Journal of Artificial Intelligence Research (JAIR)*, 13:305–338, 2000.

[Cimatti *et al.*, 1997] A. Cimatti, E. Giunchiglia, F. Giunchiglia, and P. Traverso. Planning via Model Checking: A Decision Procedure for $\mathcal{AR}$. In S. Steel and R. Alami, editors, *Proceeding of the Fourth European Conference on Planning*, number 1348 in Lecture Notes in Artificial Intelligence, pages 130–142, Toulouse, France, September 1997. Springer-Verlag. Also ITC-IRST Technical Report 9705-02, ITC-IRST Trento, Italy.

[Cimatti *et al.*, 1998a] A. Cimatti, M. Roveri, and P. Traverso. Automatic OBDD-based Generation of Universal Plans in Non-Deterministic Domains. In *Proceeding of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, Madison, Wisconsin, 1998. AAAI-Press. Also IRST-Technical Report 9801-10, Trento, Italy.

[Cimatti *et al.*, 1998b] A. Cimatti, M. Roveri, and P. Traverso. Strong Planning in Non-Deterministic Domains via Model Checking. In *Proceeding of the Fourth International Conference on Artificial Intelligence Planning Systems (AIPS-98)*, Carnegie Mellon University, Pittsburgh, USA, June 1998. AAAI-Press.

[Cimatti *et al.*, 2000] A. Cimatti, E.M. Clarke, F. Giunchiglia, and M. Roveri. NuSMV : a new symbolic model checker. *International Journal on Software Tools for Technology Transfer (STTT)*, 2(4), March 2000.

[Cimatti *et al.*, 2001a] A. Cimatti, M. Pistore, M. Roveri, and P. Traverso. Weak, Strong, and Strong Cyclic Planning via Symbolic Model Checking. Technical report, IRST, Trento, Italy, 2001.

[Cimatti *et al.*, 2001b] A. Cimatti, M. Roveri, and P. Bertoli. Searching Powerset Automata by Combining Explicit-State and Symbolic Model Checking. In *Proceedings of the 7th International Conference on Tools and Algorithms for the Construction of Systems*, pages 313–327. Springer, April 2001.

[Clarke and Wing, 1996] E. M. Clarke and J. M. Wing. Formal methods: State of the art and future directions. *ACM Computing Surveys*, 28(4):626–643, December 1996.

[Clarke *et al.*, 1999] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model Checking*. The MIT Press, Cambridge, Massachusetts, 1999.

[Daniele *et al.*, 1999] M. Daniele, P. Traverso, and M. Y. Vardi. Strong Cyclic Planning Revisited. In Susanne Biundo, editor, *Proceeding of the Fifth European Conference on Planning*, Lecture Notes in Artificial Intelligence, Durham, United Kingdom, September 1999. Springer-Verlag.

[Emerson, 1990] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics*, chapter 16, pages 995–1072. Elsevier, 1990.

[Giunchiglia and Lifschitz, 1998] Enrico Giunchiglia and Vladimir Lifschitz. An action language based on causal explanation: Preliminary report. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98) and of the 10th Conference on Innovative Applications of Artificial Intelligence (IAAI-98)*, pages 623–630, Menlo Park, July 26–30 1998. AAAI Press.

[Giunchiglia and Traverso, 1999] F. Giunchiglia and P. Traverso. Planning as Model Checking. In Susanne Biundo, editor, *Proceeding of the Fifth European Conference on Planning*, Lecture Notes in Artificial Intelligence, Durham, United Kingdom, September 1999. Springer-Verlag.

[Giunchiglia *et al.*, 1997] E. Giunchiglia, G. N. Kartha, and V. Lifschitz. Representing action: Indeterminacy and ramifications. *Artificial Intelligence*, 95(2):409–438, 1997.

[McMillan, 1993] K.L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publ., 1993.

[Pistore and Traverso, 2001] M. Pistore and P. Traverso. Planning as Model Checking for Extended Goals in Nondeterministic Domains. In *Proc.* 7^{th} *International Joint Conference on Artificial Intelligence (IJCAI-01)*. AAAI Press, August 2001.

[Pryor and Collins, 1996] L. Pryor and G. Collins. Planning for Contingency: a Decision Based Approach. *J. of Artificial Intelligence Research*, 4:81–120, 1996.

[Schoppers, 1987] M. J. Schoppers. Universal plans for Reactive Robots in Unpredictable Environments. In *Proc. of the 10th International Joint Conference on Artificial Intelligence*, pages 1039–1046, 1987.

[Smith and Weld, 1998] David E. Smith and Daniel S. Weld. Conformant graphplan. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98) and of the 10th Conference on Innovative Applications of Artificial Intelligence (IAAI-98)*, pages 889–896, Menlo Park, July 26–30 1998. AAAI Press.

[Somenzi, 1997] F. Somenzi. CUDD: CU Decision Diagram package — release 2.1.2. Department of Electrical and Computer Engineering — University of Colorado at Boulder, April 1997.

[Tovey and Koenig, 2000] C. Tovey and S. Koenig. Gridworlds as testbeds for planning with incomplete information. In *Proceedings AAAI*, 2000.

[Weld *et al*., 1998] Daniel S. Weld, Corin R. Anderson, and David E. Smith. Extending graphplan to handle uncertainty and sensing actions. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98) and of the 10th Conference on Innovative Applications of Artificial Intelligence (IAAI-98)*, pages 897–904, Menlo Park, July 26–30 1998. AAAI Press.