On the Impact of Belief State Representation in Conformant Planning

Son Thanh To
New Mexico State University
Dept. of Computer Science
sto@cs.nmsu.edu

Abstract

My doctoral thesis investigates how a belief state representation impacts on the implementation of a best-first search and progression based conformant planner and its performance. On one hand, we want the size of formulae representing belief states to be compact for scalability. On the other hand, the representation should allow us to develop a complete and efficient transition function for computing successor belief states. This computation is hard due to incomplete information in the conformant planning problem. I propose an abstract algorithm for defining a transition function given an arbitrary representation. My goal is to find a class of belief state representations, to define the transition function for each representation by instantiating the abstract algorithm, and to develop complete competitive conformant planners using those representations. I also investigate problem transformation techniques which improve the performance of planners using a certain representation and identify domains that highlight the strength or weakness of each representation. Finally, I investigate the development of a system which uses alternative representations and is able to automatically select a suitable representation for an arbitrary problem.

Introduction and Motivation

Conformant planning is the problem of finding a sequence of actions that achieves a goal from every possible initial state of the world. Typically, conformant planning arises in presence of incomplete knowledge about the initial state of the world. Since its introduction in (Smith and Weld 1998), conformant planning has attracted the attention of several researchers. A number of efficient and sophisticated conformant planners have been developed, including Conformant-FF (CFF) (Brafman and Hoffmann 2004), POND (Bryce, Kambhampati, and Smith 2006), `t0` (Palacios and Geffner 2007), and CPA (Tran et al. 2009). They are all best-first search and progression based conformant planning. The investigation of advantages and disadvantages of the representation used in each of these conformant planners are presented in (To, Pontelli, and Son 2009).

The selection of a representation for development of a planner is critical due to the following reasons: the size of formulae representing belief states influences directly the scalability of the planner, the representation method affects the difficulty of defining a complete transition function for computing successor belief states, and how efficient the transition function will be. Thus, a well-chosen representation should be compact and allow to build a complete and efficient transition function. For example, given a problem such that the domain contains n propositions and the completely unknown initial information. Clearly, the initial belief states consists of 2^n states though it is equivalent to an empty logical formula such as DNF, CNF, NNF,... On one hand, we want to represent the initial belief state in a formula of size 0 instead of an exponential size as of the belief state. On the other hand, we also want that the computation of the successor belief states is fast but the search is still complete. This is hard since the computation of successor belief states is challenging and costly due to incomplete knowledge. This motivates me to develop an abstract algorithm for computing successor belief states using an arbitrary representation. Moreover, given a specific representation, the abstract algorithm helps us answer the question whether we can develop a competitive planner using that representation and how complicated the implementation can be. Next, we need to find a class of potential belief state representations, to elaborate the abstract algorithm for defining a transition function associated with each representation, and to implement complete and competitive planners using those representations. So far I have developed three such systems: DNF (To, Pontelli, and Son 2009), CNF (To, Son, and Pontelli ICAPS-2010), and PIP (To, Son, and Pontelli AAAI-2010) which perform impressively in a class of benchmarks. I am working on other representations for developing more conformant planners following this way.

Another potential approach to conformant planning is transformation of problems aimed at reducing the size of the formula representing the initial belief state. This direction is inspired by the `one-of` *combination* technique (Tran et al. 2009) which helps reduce the size of disjunctive normal form formulae encoding initial belief states. I also investigate problem transformation techniques which enhance the performance of planners using a certain representation by reducing the size of formulae encoding belief states, e.g. `one-of` *relaxation* (To, Son, and Pontelli ICAPS-2010) for CNF-representation based planners. On the other hand, the size of formulae representing belief states in a problem may vary drastically depending on the representation used. These

two observations lead me to tackle another problem, that is to identify domains which highlight the strength or weakness of each representation. Finally, I investigate how to develop a conformant planning system which uses alternative representations and is able to automatically select a suitable representation for an arbitrary problem based on the characterization of the problem.

Research Problems

My dissertation includes the following problems

- Investigation of Representations: Investigate how to select a representation and, given a representation, how to define a transition function for computing successor belief states. This helps answer the question whether it is possible to develop a competitive conformant planner using an arbitrary representation.
- Development of Conformant Planners Using Different Representations: Introduced three systems: DNF with *minimal DNF* (To, Pontelli, and Son 2009), CNF of *reduced CNF* (To, Son, and Pontelli ICAPS-2010), and PIP with *prime implicates* (To, Son, and Pontelli AAAI-2010). Have been working on other representations, e.g. DNNF and its variants, for development of new conformant planners using those representations.
- Problem Transformation: Focusing on reducing the size of formulae encoding belief states using a specific representation. The result obtained so far is `one-of` relaxation (To, Son, and Pontelli ICAPS-2010).
- Classification of Conformant Problems: Identify classes of problems for which a certain representation is suitable.
- A Planner with Adaptive Representations: Investigate developing a conformant planner that is able to automatically select a suitable representation for each problem.

Background: Conformant Planning

A *conformant planning problem* is a tuple $P = \langle F, O, I, G\rangle$, where F is a set of propositions, O is a set of actions, I describes the initial state, and G describes the goal. A *literal* is either a proposition $p \in F$ or its negation $\neg p$. $\bar{\ell}$ denotes the complement of a literal ℓ—i.e., $\bar{\ell} = \neg\ell$, where $\neg\neg p{=}p$ for $p{\in}F$. For a set of literals L, $\overline{L} = \{\bar{\ell} \mid \ell \in L\}$. A conjunction of literals is often represented as the set of its literals.

A set of literals X is *consistent* if there exists no $p \in F$ such that $\{p, \neg p\} \subseteq X$. A set of literals X is *complete* if for each $p \in F$, either $p \in X$ or $\neg p \in F$. A *state* s is a consistent and complete set of literals. A *belief state* is a set of states. We will often use lowercase (resp. uppercase) letter, possibly with indices, to represent a state (resp. a belief state).

Each action a in O is associated with a precondition ϕ (denoted by $pre(a)$) and a set of conditional effects C_a of the form $\psi \rightarrow \ell$ (also written as $a : \psi \rightarrow \ell$), where ϕ and ψ are sets of literals and ℓ is a literal.

A state s satisfies a literal ℓ, denoted by $s \models \ell$, if $\ell \in s$. s satisfies a conjunction of literals X, denoted by $s \models X$, if it satisfies every literal belonging to X. The satisfaction of a formula in a state is defined in the usual way. Likewise, a belief state S satisfies a literal ℓ, denoted by $S \models \ell$, if $s \models \ell$ for every $s \in S$. S satisfies a conjunction of literals X, denoted by $S \models X$, if $s \models X$ for every $s \in S$.

Given a state s, an action a is *executable* in s if $s \models pre(a)$. The effect of executing a in s is

$$e(a, s) = \{\ell \mid \psi \rightarrow \ell \in C_a.\ s \models \psi\}$$

The transition function, denoted by Φ, in the conformant planning domain of P is defined by

$$\Phi(a, S) = \begin{cases} \{s \setminus \overline{e(a,s)} \cup e(a,s) | s \in S\} & S \models pre(a) \\ \mathit{undefined} & \text{otherwise} \end{cases} \quad (1)$$

We can extend the function Φ to define $\widehat{\Phi}$, a transition function which maps sequences of actions and belief states to belief states. $\widehat{\Phi}$ is used to reason about the effects of plans. Let S be a belief state. We say that an action a is executable in a belief state S if it is executable in every state belonging to S. Let $\alpha_n = [a_1, \ldots, a_n]$ be a sequence of actions:

- If $n = 0$ then $\widehat{\Phi}([\,], S) = S$;
- If $n > 0$ then
  - if $\widehat{\Phi}(\alpha_{n-1}, S)$ is undefined or a_n is not executable in $\widehat{\Phi}(\alpha_{n-1}, S)$, then $\widehat{\Phi}(\alpha_n, S)$ is undefined;
  - if $\widehat{\Phi}(\alpha_{n-1}, S)$ is defined and a_n is executable in $\widehat{\Phi}(\alpha_{n-1}, S)$ then $\widehat{\Phi}(\alpha_n, S){=}\{\Phi(a_n, s') | s' {\in} \widehat{\Phi}(\alpha_{n-1}, S)\}$ where $\alpha_{n-1} = [a_1, \ldots, a_{n-1}]$.

The initial state of the world I is a belief state and is represented by a formula. In all benchmarks, I is a conjunction of a set of literals, a set of `one-of`-clauses—representing an exclusive-or of its components—and a set of `or`-clauses—representing the logical or of its components. By S_I we denote the set of all states satisfying I. Typically, the goal description G can contain literals and `or`-clauses.

A sequence of actions $[a_1, \ldots, a_n]$ is a solution of P if $\widehat{\Phi}([a_1, \ldots, a_n], S_I)$ satisfies the goal G.

Investigation of Representations

A formula φ is said to be in *R-form* if φ is an instance of a belief state representation R. Given a representation R, several aspects need to be considered: (*i*) the size of the formula representing belief states affects directly memory consumption, which impact on the scalability of the candidate planner; (*ii*) the possibility to define a complete transition function for computing successor belief states and the difficulty of implementing this function; and (*iii*) the cost of the transition function which influences directly the raw performance of the planner. I propose an abstract algorithm for defining a complete transition function associated with an arbitrary representation. The algorithm also helps answer the third question. The abstract algorithm is developed based on the steps of defining function Φ, which is described in the section Background, as follows

1. Check whether the precondition $pre(a)$ is satisfied in φ or not. If so, continue the following steps.
2. Compute the effect of executing a in φ, like $e(a, s)$ in the definition of Φ. Observe that φ represents the belief state

$S = \{s \mid s \models \varphi\}$. If for every effect $\psi_i \rightarrow l_i$ in C_a, either $\varphi \models \psi_i$ or $\varphi \models \neg\psi_i$ holds, then we have either $l_i \in e(a,s)$ or $l_i \notin e(a,s)$ for every s in S. That means that every state s in S has the same set $e(a,s)$. In this case, therefore, we can compute the effect of executing a in φ as $e(a,\varphi) = \{l_i \mid l_i \rightarrow \psi_i \in C_a \wedge \varphi \models \psi_i\}$ and we say that φ is *enabling for action* a. What if φ is not enabling for a, i.e. there exists an effect $l_i \rightarrow \psi_i$ in C_a such that neither $\varphi \models \psi_i$ nor $\varphi \models \neg\psi_i$ holds. This means that S consists of two distinct groups of states S_1 and S_2 such that $\forall s \in S_1.s \models \psi_i$ and $\forall s \in S_2.s \models \neg\psi_i$. Observe that, $\varphi_1 = \varphi \wedge \psi_i$ representing S_1, $\varphi_2 = \varphi \wedge \neg\psi_i$ representing S_2, and $\varphi \equiv \varphi_1 \vee \varphi_2$. Hence, φ can be replaced with the set $\{\varphi_1, \varphi_2\}$ [1]. Note that ψ_i is known in every element in this set. If we keep doing this for every other effect $\psi_j \rightarrow l_j$, i.e. every formula φ_k in the set will be replaced with the set $\{\varphi_{k_1}, \varphi_{k_2}\}$ if ψ_j is unknown in φ_k ($\{\varphi_{k_1}, \varphi_{k_2}\}$ is computed in the same manner as $\{\varphi_1, \varphi_2\}$ is). Then we will obtain a set of formulae, denoted by $enb_a(\varphi)$, whose disjunction is equivalent to φ and every formula in this set is enabling for a. Thus, we now can compute the effect of executing a in each of these formulae.

3. For each formula φ_i in $enb_a(\varphi) = \{\varphi_1, \ldots, \varphi_n\}$, we need to compute the result of executing a in φ_i, denoted by φ'_i. Intuitively, φ'_i encodes the belief state $\{s \setminus \overline{e(a,s)} \cup e(a,s) \mid s \models \varphi_i\}$. Observe that, $e(a,s) = e(a,\varphi_i)$ for every s satisfying φ_i due to the fact that φ_i is enabling for a. We take account of this to compute φ'_i. In some representations, we may need to convert φ_i to another (type of) formula, e.g. a disjunction of formulae, for ease of computation. One may refer to (To, Son, and Pontelli ICAPS-2010) as an example.

4. Convert the disjunction $\varphi'_1 \vee \ldots \vee \varphi'_k$ to an equivalent formula in R-form. This is the formula representing the successor belief state of φ after executing a. Note that there may exist more than one formula representing the same belief state in R-form. Therefore, we may need to consider simplifying the resulting formula. This is a trade-off between gaining a smaller size of the resulting formula and spending more time for the simplification. A good algorithm should be a good balance between the two extremes.

The key idea of the above algorithm is that instead of working directly with a formula, we translate it to another equivalent formula (set of formulae) in which the computation is easier. The correctness of the abstract algorithm is intuitive due to equivalence of the conversions. It is worth to mention that the computational cost of checking entailment, which is necessary in the first two steps, depends largely on the representation used. The cost for checking satisfaction is one of the most important factor that need to be considered in the selection of representations. In several representations; e.g. OBDD, DNF, DNNF, prime implicates,...; checking entailment is tractable.

[1] For ease of computation depending on the representation used, φ_1 or φ_2 can be converted to a disjunctive set of formulae

Proposed Representations

The representations proposed in this section have been (being) investigated in my dissertation. Some of them have already been implemented in a planner such as *minimal DNF* in DNF (To, Pontelli, and Son 2009), *reduced-CNF* in CNF (To, Son, and Pontelli ICAPS-2010), and *prime implicates* in PIP (To, Son, and Pontelli AAAI-2010).

Disjunctive Representations—DNF

First, we consider the *minimal DNF-state* representation which has been implemented in the DNF system (To, Pontelli, and Son 2009). A detailed description of minimal DNF-state and the accompanying transition function can be found in the same paper. minimal DNF-state is compact for many problems. For example, a belief state $\{\{f,g\},\{f,\neg g\}\}$ can be encoded in the minimal DNF-state $\{\{f\}\}$. The development of the transition function is straightforward since minimal DNF-state is a special form of DNF, the computation for successor belief states is polynomial if the number of effects of each action is bounded, and checking entailment is easy. Moreover, the overhead of the last step in the abstract algorithm is not much since the resulting disjunction is already in DNF-form. This is validated by the competitive performance of DNF. The main disadvantage of this representation is that its size can explode in a certain class of domains.

We also consider *prime implicants* as another representation, which is a special type of DNF. Obviously, prime implicants form has many properties, advantages, and weakness as minimal-DNF does. This representation has more strengths as it is unique for each belief state so that checking for repetitions of belief states in a search tree is tractable. In many cases, it has the minimum size among the DNF-formulae representing the same belief state. A main disadvantage of this representation is that maintaining prime implicants form of belief states is very expensive. Identifying a class of problems on which this cost of computation is acceptable is the key in this problem.

Conjunctive Representations—CNF

The disadvantage of DNF-representations makes us think about this type of representations. Using a CNF representation, we can make use of the advance of SAT-solvers and the preprocessing techniques for simplifying CNF-formulae, i.e. *unit-propagation*, *subsumption*, and *self-subsumption*. Therefore, the CNF-formulae representing belief states can be maintained in a small size easier and checking satisfaction is not that expensive as it should be (NP-hard).

The first CNF-representation we investigated is *reduced-CNF* which have been implemented in CNF (To, Son, and Pontelli ICAPS-2010). The definition, full technical description, advantages, and disadvantages of reduced-CNF representation are presented in the same paper.

The second CNF-form we studied is *prime implicates*. The technical concerns, properties, advantages and disadvantages of prime implicates is presented in (To, Son, and Pontelli AAAI-2010).

DNNF Representation

A DNNF (decomposable negation normal form) is a special form of NNF (negation normal form) which satisfies the decomposable property, i.e. no two conjuncts of the same conjunction in the formula share an atom. A detailed description of DNNF and its properties can be found in (Darwiche 2001). DNNF has several desire properties, e.g. checking satisfaction of a literal is linear in the size of the formula. Moreover, there are belief states that have linear DNNF representations, yet exponential DNF representations. In addition, a disjunction of DNNF formulae is a DNNF formula. Currently, I am dealing with the problem of simplifying DNNF formulae.

Problem Transformation

There have been several approaches following this direction, e.g translating conformant problems to classical problems (Palacios and Geffner 2007), reducing the size of DNF formulae representing initial belief states using `one-of` combination technique (Tran et al. 2009). I have been focusing on reducing the size of formulae encoding belief states in a specific representation. Note that a transformation technique which works well for a representation is not necessary good for another representation. For example, `one-of` combination makes the CNF-formula encoding belief states of the problem larger.

The result obtained so far is `one-of` relaxation technique (To, Son, and Pontelli ICAPS-2010) which helps reduce the size of CNF-forms of belief states and hence enhance the performance of any conformant planner which use a CNF-representation, e.g. CNF and PIP, impressively on domains that satisfy the condition of relaxation.

Classification of Conformant Problems

A problem, which highlights the strength of a certain representation, may emphasize the weakness of others. The reasons are as follows. First, the size of belief state formulae can vary greatly depending on the representations used. Second, the problem may satisfy the condition of a transformation technique which does work well for some representation(s) but does not for the others. Examples for both cases are provided in (To, Son, and Pontelli ICAPS-2010).

Adaptive Representation Conformant Planner

My goal is to develop a conformant planner which is able to select a suitable representation for each problem based on the characterization of the problem. This idea is partially applied in (To, Son, and Pontelli AAAI-2010), which appears to be a valid approach to combine strengths of different representations and to reduce their disadvantages.

Conclusion

Having committed with this topic after spending several years on different areas, I strongly believe that this is a right direction of my dissertation. The reason I have chosen these problems is not only because they are very interesting to me. I also expect that, by solving these problems, I can contribute to the development of the planning research area in several ways as follows

- Help understand better the impact of belief state representation in conformant planning.
- Propose a general approach to developing conformant planning systems. More specifically, help answer the questions: how to select a good representation, how to develop a planner given a representation, what technical aspects needs to be considered in the development of a planner to improve/enhance its performance,...
- Given a problem, help to select a good planner for the problem. On the other hand, given a planner, identify a class of problems which highlight the performance of the planner.
- Provide several highly competitive conformant planners, e.g. DNF, CNF, and PIP. Each of these planner outperforms all other state-of-the-art planners on a certain class of problems impressively. For example, CNF and PIP are the only two conformant planners which can solve the problem instances $coins - 21$ and higher.
- Propose transformation techniques which help enhance performance of planners, e.g `one-of` relaxation which, if applied in a planner using a conjunctive representation, will boost the performance of the planner on a certain class of problems
- Propose new benchmarks, e.g. new challenging benchmarks proposed in (To, Son, and Pontelli ICAPS-2010)

References

Brafman, R., and Hoffmann, J. 2004. Conformant planning via heuristic forward search: A new approach. In *ICAPS*, 355–364.

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

Bryce, D.; Kambhampati, S.; and Smith, D. 2006. Planning Graph Heuristics for Belief Space Search. *JAIR* 26:35–99.

Darwiche, A. 2001. Decomposable Negation Normal Form. *Journal of the ACM* 48(4):608–647.

Palacios, H., and Geffner, H. 2007. From Conformant into Classical Planning: Efficient Translations that may be Complete Too. In *ICAPS*.

Smith, D., and Weld, D. 1998. Conformant graphplan. In *AAAI*, 889–896.

To, S. T.; Pontelli, E.; and Son, T. C. 2009. A Conformant Planner with Explicit Disjunctive Representation of Belief States. In *Proceedings of ICAPS-2009*.

To, S. T.; Son, T. C.; and Pontelli, E. 2010. A New Approach to Conformant Planning Using CNF. To Appear in *Proceedings of ICAPS-2010*.

To, S. T.; Son, T. C.; and Pontelli, E. 2010. On the Use of Prime Implicates in Conformant Planning. To Appear in *Proceedings of AAAI-2010*.

Tran, D.-V.; Nguyen, H.-K.; Pontelli, E.; and Son, T. C. 2009. Improving performance of conformant planners: Static analysis of declarative planning domain specifications. In *PADL*, LNCS 5418, 239–253. Springer.