cf2sat and cf2cs+cf2sat: Two Conformant Planners

Héctor Palacios
Departamento de Tecnología
Universitat Pompeu Fabra
Pg Circunvalación, 8
08003 Barcelona, SPAIN
hector.palacios@upf.edu

Abstract

Even under polynomial restrictions on plan length, conformant planning remains a very hard computational problem as plan verification itself can take exponential time. We present two planners for the IPC-5. The first is an optimal complete conformant planner called cf2sat, which transform the PDDL into a propositional theory, that is later compiled into normal form called d–DNNF to obtain a new formula that encodes all the possible plans. This planner gives good results on *pure* conformant problems as emptyroom and sorting-nets, but fails to scale on problems more close to classical planning as bomb-in-the-toilet. Although the heavy price of conformant planning cannot be avoided in general, in many cases conformant plans are verifiable efficiently by means of simple forms of disjunctive inference. We present a second planner cf2cs+cf2sat which is a suboptimal conformant planner that try first to solve a problem by translating it (cf2cs) into an equivalent classical problem, that is then solved by an off-the-shelf classical planner. This translation leads to an efficient but incomplete planner capable of solving non-trivial problems quickly. The translation accommodates simple 'reasoning by cases' by means of an 'split-and-merge' strategy. If cf2cs is not able to solve the problem, cf2cs+cf2sat switch to cf2sat to ensure completeness. Even thought cf2cs is incomplete, it deals successfully with simple problems as bomb-in-the-toilet, and other non-trivial problems as emptyroom.

Introduction

Conformant planning is a form of planning where a goal is to be achieved when the initial situation is not fully known and actions may have non-deterministic effects (Goldman & Boddy 1996)[1]. Conformant planning is computationally harder than classical planning, as even under polynomial restrictions on plan length, plan verification remains hard (Turner 2002).

We present two conformant planners based on two strategies combined properly. The first, cf2sat, is an optimal complete conformant planner that translates the problem into a logic theory, as in SATPLAN (Kautz & Selman 1996). A SAT solver call over this theory would result in one of the possible executions (actions and fluents), which assume a particular initial state, but we want to obtain a plan conformant to all the initial states. From this theory we generate a new one encoding all the possible conformant plans, and call a SAT solver once to obtain a plan. We obtained good results running cf2sat on some very complex domains but failed to scale in more simple problems (Palacios & Geffner 2005).

For this reason we have proposed an incomplete method for mapping from conformant planning to classical planning, cf2cs (Palacios & Geffner 2006). This works by doing limited disjunctive reasoning and allow to solve popular benchmarks like bomb-in-the-toilet, trying to fill the gap left by cf2sat between pure conformant planning and classical planning. The second planner, cf2cs+cf2sat, is a suboptimal complete conformant planning that starts by trying to solve the problem using cf2cs. If it is not possible, the algorithm switch to cf2sat, which is complete.

In the rest of the report we present with more detail the conformant2sat planner and the conformant2classical transformation. Later, we comment about their performance and integration to obtain the presented planners.

Mapping Conformant Planning to SAT

For a conformant planning problem, if the number m of possible initial states $s_0 \in Init$ is bounded (e.g., bounded number of disjunctions of bounded size in the initial situation) and actions are deterministic, the conformant planning problem P with a fixed horizon n can be mapped in the SAT problem over the formula (Palacios & Geffner 2005)

$$\bigwedge_{s_0 \in Init} T^{s0}(P, n) \quad (1)$$

where $T(P, n)$ is the propositional theory that encodes the problem P with horizon n. $T^{s0}(P, n)$ is $T(P, n)$ with two modifications: first, fluent literals L_0 (L at time 0) are replaced by true/false iff L is true/false in the (complete) state s_0, and second, fluent literals L_i, $i > 0$, are replaced by 'fresh' literals L_i^{s0}, one for each $s_0 \in Init$.

Eq. 1 can be thought as expressing m 'classical planning problems', one for each possible initial state $s_0 \in Init$, that are *coupled* in the sense that they all share the same set of actions; namely, the action variables are the only variables shared across the different subtheories $T^{s0}(P, n)$ for $s_0 \in Init$.

[1] We assume that actions are deterministic and all the uncertainty is on the initial state. This assumption does not lead to loss of expressivity.

However, a planner using Eq. 1 naively will not scale. We have already proposed two approaches to optimal classical conformant planning based on logical formulations (Palacios *et al.* 2005; Palacios & Geffner 2005). Both of them translate the problem into CNF, and obtain a plan by doing logical operations and search. In cf2sat (Palacios & Geffner 2005) (for *conformant2sat*) we construct a new propositional formula:

$$T_{\text{cf}}(P) \quad = \quad \bigwedge_{s_0 \in Init} \text{project}[\, T(P) \,|\, s_0 \;;\, Actions \,] \qquad (2)$$

by doing logical operations as projection (dual of forgetting) and conditioning. The project operation allows to safely *And* over each theory depending on each initial state. The models of the formula project$[\, T(P) \,|\, s_0 \;;\, Actions \,]$ are the models of $T(P) \,|\, s_0$ but looking only at the action variables.

Theorem 1 (Palacios & Geffner, 2005) *The models of $T_{cf}(P)$ in (2) are one-to-one correspondence with the conformant plans for the problem P.*

We feed $T_{\text{cf}}(P)$ into a SAT solver to obtain a plan. Logical operations became feasible by compiling the propositional theory into d–DNNF (Darwiche 2002), a formal norm akin to OBDD. The result of compiling a propositional theory ϕ to d–DNNF is a logical circuit that encodes all the possible models of ϕ. Summarizing, the cf2sat algorithm is:

- The following operations are repeated starting from a planning horizon $N = 0$ which is increased by 1 until a solution is found[2].
  1. The theory $T(P)$ is **compiled** into the d–DNNF theory $T_{\text{c}}(P)$
  2. From $T_{\text{c}}(P)$, the transformed theory

     $$T_{\text{cf}}(P) \quad = \quad \bigwedge_{s_0 \in Init} \text{project}[\, T_{\text{c}}(P) \,|\, s_0 \;;\, Actions \,]$$

     is obtained by operations that are linear in time and space in the size of the DAG representing $T_{\text{c}}(P)$.
  3. The theory $T_{\text{cf}}(P)$ is converted into CNF and the **SAT solver** is called upon it.

The plan obtained can be optimal in terms of the number of actions if we forbidden the concurrent execution of every pair of actions. This is known as the *sequential* setting. If we allow non-interfering actions to be executed simultaneously, *parallel* setting, the total executing time or makespan will be minimized.

Actually, it is not necessary to do projection and conditioning for every initial state. By compiling the theory $T(P)$ doing case analysis first on the variables of initial state, we can obtain each project$[\, T(P) \,|\, s_0 \;;\, Actions \,]$ as a sub-circuit of $T_{\text{cf}}(P)$. Therefore, Eq. 2 can be obtained in linear time over the compiled theory $T_{\text{cf}}(P)$. Translation from this new circuit into CNF is done by introducing propositional variables for each gate and adding clauses to encode the relation between them.

[2]A better lower bound can be the length of an optimal classical plan for one initial state

Compiling uncertainty away: Conformant to Classical Planning (sometimes)

The main motivation of cf2cs is to narrow the gap between conformant planning and classical planning by developing an approach that targets 'simple' conformant problems effectively. The approach is not complete but it provides solutions to non-trivial problems. For instance, simple rules suffice to show that a robot that moves n times to the right in an empty grid of size n, will necessarily end up in the rightmost column.

We have proposed to solve *some non-trivial conformant planning problems* by translating them intro *classical planning problems* (Palacios & Geffner 2006). New problems are fed into a off-the-shelf classical planner. The translation is sound as the classical plans are all conformant, but it is incomplete as the converse does not always hold. The translation scheme accommodates 'reasoning by cases' by means of an 'split-and-merge' strategy; namely, atoms L/X_i that represent conditional beliefs 'if X_i then L' are introduced in the classical encoding that are then combined by suitable actions when certain invariants in the plan are verified.

The translation scheme maps a conformant planning problem P into a classical planning problem $K(P)$. For each atom a in P we add to $K(P)$ new atoms Ka and $K\neg a$. At time t if $Ka \wedge \neg K\neg a$ (resp. $\neg Ka \wedge K\neg a$) holds, then a is true (resp. false) in all the states of the belief state. The initial state of $K(P)$ indicates the atoms that are known to be true or false in the initial belief state of P. Otherwise it states that the value of those atoms is unknown: $\neg Ka \wedge \neg K\neg a$. The goal of P is assumed to be a list of atoms $\{g_1, \ldots, g_n\}$. Therefore, the goal of $K(P)$ requires all those atoms to be known: $\{Kg_1, \ldots, Kg_n\}$. This encoding is related to 0-approximation (Baral & Son 1997). In general, it allow to capture that after doing some actions, the effect can be unsure if the real value of the conditions is not known.

This encoding, so far, does not allow any kind of disjunctive reasoning. We extend the translation further so that the disjunctions in P are taken into account in a form that is similar to the Disjunction Elimination inference rule used in Logic

$$\text{If } X_1 \vee \cdots \vee X_n,\; X_1 \supset L, \ldots, X_n \supset L \text{ then } L \qquad (3)$$

For doing this, we add to $K(P)$ atoms L/X_i to encode that $L \supset X_i$ holds. For example, if for problem P we have the disjunction $X_1 \vee \cdots \vee X_n$ in the initial state, and actions $a_1, \ldots, a_n$ with conditional effect $A \wedge X_i \rightarrow L$; In $K(P)$ those actions will have also conditional effect $A \rightarrow L/X_i$. Informally, $A \rightarrow L/X_i$ can be read as: "If we apply a_i when A is true, we conclude that L is true if X_i is true"[3]. After applying every action a_i, if some invariants were preserved, we can conclude L because $L/X_1 \wedge \cdots \wedge L/X_n$ holds. To allow this conclusion, we add to $K(P)$ a new action $merge_{X,L}$ with conditional effect $L/X_1 \wedge \cdots \wedge L/X_n \rightarrow KL$.

These rules more detailed and other rules can be read in (Palacios & Geffner 2006). They yield expressivity without sacrificing efficiency, as they manage to *accommodate*

[3]It is true if a_i does not modify X_i. In general it is more subtle. More details on (Palacios & Geffner 2006)

non-trivial forms of disjunctive inference in a classical theory without having to carry disjunctions explicitly in the belief state: some disjunctions are represented by atoms like L/X_i, and others are maintained as *invariants* enforced by the resulting encoding.

Theorem 2 (Soundness $K(P)$**)** *(Palacios & Geffner, 2006) Any plan that achieves the literal KL in $K(P)$ is a plan that achieves L in the conformant problem P.*

Results

We ran the optimal planner cf2sat with the Darwiche's d–DNNF compiler c2d v2.18 (Darwiche 2004) and the SAT solver `siege_v4`, obtaining very good results on problems as Emptyroom, Cube-Center, Ring And Sorting-Nets. In general, the compiling step was not the bottleneck. It was not the case in domains like Bomb-in-the-Toilet, where the big number of objects lead to huge theories impossible to be compiled. A middle case was the Ring domain, which lead to big d–DNNFs but later they were very easy for the SAT solver.

We also ran the translator cf2cs from conformant to classical planning on domains where it was able to work, as Emptyroom, Cube-Center, Bomb-in-the-Toilet, Safe, Grid, Logistics. Then we solve those new classical instances by calling the FF (Hoffmann & Nebel 2001) classical planner. Among the popular benchmarks, there are three domains, Sorting-Nets, (Incomplete) Blocks, and Ring, which cannot be handled by this translation scheme. The results were excellent. We were surprised to obtain in general optimal plans even though FF is a suboptimal planner. An interesting point is that the instances resulting of cf2cs have actions with many conditional effects, and many planners available were not able to deal with these instances.

All the relevant programs were written in C++:

- For cf2sat
  - Translator from PDDL to CNF, cconf. It was written by Blai Bonet in joint work (Palacios *et al.* 2005).
  - Translator from $T_{\text{cf}}(P)$, in d–DNNF, to CNF.
- For cf2cs
  - Translator from a PDDL of conformant problem to a PDDL of the equivalent classical problem. The parser was taken from cconf.

Planners for the IPC-5

For the IPC-5, we present two complete planners.

- cf2sat: An optimal parallel conformant planning, using the d–DNNF compiler c2d v2.20 (Darwiche 2004) and the SAT solver `siege_v4`, by Lawrence Ryan, or *zChaff* (Moskewicz *et al.* 2001)[4].
- cf2cs(FF)+cf2sat: A suboptimal conformant planning. It starts trying to solve the problem with cf2cs(FF). If not possible to try with cf2cs(FF) or not solution is found, cf2cs(FF)+cf2sat switch to cf2sat.

[4] siege_v4 was reported to be fast on planning theories (Kautz 2004). Our experiments confirmed that affirmation. Sometimes the CNFs are too big for siege_v4. On these case we try with solve the instances with *zChaff* which is slower in general for our theories.

References

Baral, C., and Son, T. C. 1997. Approximate reasoning about actions in presence of sensing and incomplete information. In *Proc. ILPS 1997*, 387–401.

Darwiche, A. 2002. On the tractable counting of theory models and its applications to belief revision and truth maintenance. *J. of Applied Non-Classical Logics*.

Darwiche, A. 2004. New advances in compiling cnf into decomposable negation normal form. In *Proc. ECAI 2004*, 328–332.

Goldman, R. P., and Boddy, M. S. 1996. Expressive planning and explicit knowledge. In *Proc. AIPS-1996*.

Hoffmann, J., and Nebel, B. 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research* 14:253–302.

Kautz, H., and Selman, B. 1996. Pushing the envelope: Planning, propositional logic, and stochastic search. In *Proceedings of AAAI-96*, 1194–1201. AAAI Press / MIT Press.

Kautz, H. 2004. Satplan04: Planning and satisfiability. In *Proceedings of IPC-4*.

Moskewicz, M. W.; Madigan, C. F.; Zhao, Y.; Zhang, L.; and Malik, S. 2001. Chaff: Engineering an Efficient SAT Solver. In *Proc. of the 38th Design Automation Conference (DAC'01)*.

Palacios, H., and Geffner, H. 2005. Mapping conformant planning to sat through compilation and projection. In *Spanish Conference on Artificial Inteligence (CAEPIA)*.

Palacios, H., and Geffner, H. 2006. Compiling uncertainty away: Solving conformant planning problems using a classical planner (sometimes). Technical report. Submitted to AAAI-06.

Palacios, H.; Bonet, B.; Darwiche, A.; and Geffner, H. 2005. Pruning conformant plans by counting models on compiled d-DNNF representations. In *Proc. of the 15th Int. Conf. on Planning and Scheduling (ICAPS-05)*. AAAI Press.

Turner, H. 2002. Polynomial-length planning spans the polynomial hierarchy. In *JELIA '02: Proc. of the European Conference on Logics in AI*, 111–124. Springer-Verlag.