Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling (ICAPS 2008)

In Search of the Tractability Boundary of Planning Problems

Omer Giménez
Dept. de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Jordi Girona, 1-3
08034 Barcelona, Spain
omer.gimenez@upc.edu

Anders Jonsson
Dept. of Information and Communication Technologies
Universitat Pompeu Fabra
Passeig de Circumval·lació, 8
08003 Barcelona, Spain
anders.jonsson@upf.edu

Abstract

Recently, considerable focus has been given to the problem of determining the boundary between tractable and intractable planning problems. To this end, we present complexity results for two classes of planning problems from the literature. First, we show that approximating a solution to a planning problem in the class 3S to within polynomial factors is NP-hard. We also show that plan existence is NP-hard for planning problems with chain causal graphs and variables with domain size at most 7. In addition to the immediate implications, our results provide some insight into what makes some planning problems intractable.

Introduction

Recent research in planning has made a significant effort to determine the boundary between tractable and intractable planning problems (Brafman and Domshlak 2003; Domshlak and Dinitz 2001; Giménez and Jonsson 2008; Jonsson and Bäckström 1998; Jonsson 2007; Katz and Domshlak 2007a). Typically, researchers have used the causal graph of a planning problem to characterize its structure. As a result, there are classes of planning problems known to be tractable, and classes of problems for which no efficient solution exists, unless some established assumption fails, like $\mathrm{P} \neq \mathrm{NP}$. However, the boundary between tractable and intractable planning problems is still not clearly established. The present paper contributes novel complexity results for two classes of planning problems from the literature, in an effort to reduce this complexity gap.

The problem is not of purely theoretical interest. For instance, complex planning problems can be projected onto tractable fragments of planning problems to generate heuristics to be used during search (Katz and Domshlak 2007b). Also, the causal graph heuristic (Helmert 2006) exploits the hierarchical structure of a planning problem by transforming it into a more tractable form: first, it translates propositional variables into multi-valued variables, a process that simplifies the causal graph of the problem; then, it keeps relaxing the problem until the causal graph becomes acyclic.

Our first contribution relates to the class 3S of planning problems, designed by Jonsson and Bäckström (1998) to show that there are planning problems that are decidable in polynomial time but have exponentially long solutions. The authors devised an algorithm that outputs prefixes of the solution to planning problems in 3S and showed that the algorithm is polynomial in the size of the output. On the other hand, Giménez and Jonsson (2008) developed a polynomial-time algorithm that outputs a complete solution to planning problems in 3S in the form of macros.

Interestingly, both algorithms for plan generation in 3S may generate plans that are exponentially longer than the optimal. It is well known that finding optimal plans is NP-hard for planning problems in 3S, but how good of an approximation can we obtain? In this paper we show that in general, approximating a solution to planning problems in 3S to within polynomial factors is NP-hard. Concretely, we show that the ratio between the lengths of the best solution a tractable planner may obtain and the actual optimal solution grows exponentially with the size of the input, unless $\mathrm{P} = \mathrm{NP}$. The only previous work on the complexity of approximations that we are aware of (Helmert, Mattmüller, and Röger 2006) deals exclusively with planning problems that have polynomial-length solutions.

Our second contribution concerns the class $\mathbb{C}_n$ of planning problems with multi-valued state variables and chain causal graphs, that is, the causal graph is just a directed path. Let $\mathbb{C}_n^k$ be the subclass of $\mathbb{C}_n$ when we restrict to planning problems where the domains of the variables have size k or less. It is known that class $\mathbb{C}_n^2$ is polynomial-time solvable (Brafman and Domshlak 2003), and that deciding class $\mathbb{C}_n$ is NP-hard (Giménez and Jonsson 2008). Hence the complexity of solving and deciding classes $\mathbb{C}_n^2$ and $\mathbb{C}_n$ is well known; our aim is to study the complexity of solving or deciding those classes in between, namely $\mathbb{C}_n^k$ for $k \geq 3$.

Domshlak and Dinitz (2001) showed that there are solvable instances of $\mathbb{C}_n^3$ that require exponentially long plans. This means there is no polynomial-time plan generation algorithm for $\mathbb{C}_n^k$ with $k \geq 3$, as was the case for $\mathbb{C}_n^2$. However, this does not rule out the existence of a polynomial-time algorithm deciding class $\mathbb{C}_n^k$, or even an algorithm that generates plans in some succinct form, like that of Giménez and Jonsson (2008) for the class 3S of planning problems. This is not incompatible with $\mathbb{C}_n$ being NP-hard.

In this contribution we show that deciding class $\mathbb{C}_n^k$ for $k \geq 7$ is a NP-hard problem. This result strengthens that of Giménez and Jonsson (2008). We prove this result in two parts. First we prove that deciding $\mathbb{C}_n^{11}$ is NP-hard, by means of a reduction from the well known CNF-SAT problem. The underlying idea in the reduction admits some improvement, so we show how to obtain another reduction, much more involved, that only requires variable domains of size 7.

Notation

Throughout the paper, we use $[n]$ and $[i, n]$ to denote the sets $\{1, \ldots, n\}$ and $\{i, \ldots, n\}$.

Let V be a set of state variables, and let $D(v)$ be the finite domain of state variable $v \in V$. We define a state s as a function on V that maps each state variable $v \in V$ to a value $s(v) \in D(v)$ in its domain. A partial state p is a function on a subset $V_p \subseteq V$ of state variables that maps each state variable $v \in V_p$ to $p(v) \in D(v)$. For a subset $C \subseteq V$ of state variables, $p \mid C$ is the partial state obtained by restricting the domain of p to $V_p \cap C$. Sometimes we use the notation $(v_1 = x_1, \ldots, v_k = x_k)$ to denote a partial state p defined by $V_p = \{v_1, \ldots, v_k\}$ and $p(v_i) = x_i$ for each $v_i \in V_p$.

A planning problem is a tuple $P = \langle V, \mathsf{init}, \mathsf{goal}, A\rangle$, where V is the set of variables, init is an initial state, goal is a partial goal state, and A is a set of operators. An operator $a = \langle pre(a); post(a)\rangle \in A$ consists of a partial state $pre(a)$ called the *pre-condition* and a partial state $post(a)$ called the *post-condition*. Operator a is applicable in any state s such that $s(v) = pre(a)(v)$ for each $v \in V_{pre(a)}$, and applying operator a in state s results in a new state s' such that $s'(v) = post(a)(v)$ if $v \in V_{post(a)}$ and $s'(v) = s(v)$ otherwise.

In this paper, all operators are unary, which means that $|V_{post(a)}| = 1$. In this case, the causal graph of a planning problem P is a directed graph (V, E) with state variables as nodes. There is an edge $(u, v) \in E$ if and only if $u \neq v$ and there exists an operator $a \in A$ such that $u \in V_{pre(a)}$ and $v \in V_{post(a)}$.

Let v be a variable of a planning problem P. The *domain transition graph of* v is a labelled, directed graph G where the set of vertices is the domain $D(v)$ of v, and there is a directed edge (x, y) with label p, where p is a partial state of P not defined on v, if the planning instance P has an operator $\langle p, v = x; v = y\rangle$. In the particular case where v has a single parent v' in the causal graph of P, the edges of the domain transition graph can simply be labelled with the value or values v' can have for v to move from x to y.

3S

The class 3S (Jonsson and Bäckström 1998) consists of planning problems with binary state variables and acyclic causal graphs. In addition, each state variable is either static, symmetrically reversible or splitting. A variable v is symmetrically reversible if, for each operator $a = \langle p, v = 0; v = 1\rangle$, where p is a partial state not defined on v, there exists a symmetric operator $a' = \langle p, v = 1; v = 0\rangle$, and vice versa. Thus, if p holds, then a and a' can be applied repeatedly to flip the value of v. For a definition of static and splitting, we refer to Jonsson and Bäckström (1998).

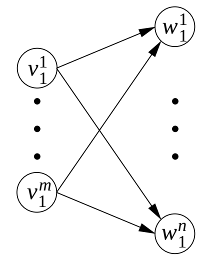

Figure 1: Causal graph of the planning problem P_1

We show that in general, it is not possible to approximate a solution to a 3S planning problem to within a polynomial factor. Our proof is based on a reduction from `MINIMUM SET-COVER`, a problem described by a set $S = \{x_1, \ldots, x_n\}$ of elements and a set $C = \{C_1, \ldots, C_k\}$ of subsets of S, i.e., for each $i \in [k]$, $C_i \subseteq S$. The problem is to determine a minimal subset $C' \subseteq C$ that covers S, i.e., for each $x_j \in S$ there exists $C_i \in C'$ such that $x_j \in C_i$.

First, we reduce `MINIMUM SET-COVER` to a planning problem P_1. This reduction is a modified version of the reduction described by Jonsson and Bäckström (1998). The set of state variables of P_1 is $V = V_1 \cup W_1$, where $V_1 = \{v_1^1, \ldots, v_1^k\}$ contains one state variable v_1^i per subset $C_i \in C$ and $W_1 = \{w_1^1, \ldots, w_1^n\}$ contains one state variable w_1^j per element $x_j \in S$.

For ease of definition we introduce some additional notation. Let $p(X, y)$, $X \in \{V_1, W_1\}$ and $y \in \{0, 1\}$, be the partial state assigning y to all variables in X. Thus $p(V_1, 0) = (v_1^1 = 0, \ldots, v_1^k = 0)$. Let $q(X, m)$, $X \in \{V_1, W_1\}$ and $m \in \{1, \ldots, |X|\}$, be the partial state assigning 0 to all variables in X except x^m, to which it assigns 1. Thus $q(V_1, i) = (v_1^1 = 0, \ldots, v_1^{i-1} = 0, v_1^i = 1, v_1^{i+1} = 0, \ldots, v_1^k = 0)$.

For each state variable v_1^i, P_1 contains an operator $a_1^i = \langle v_1^i = 0; v_1^i = 1\rangle$ and a corresponding symmetric operator $a_1^{i'} = \langle v_1^i = 1; v_1^i = 0\rangle$. For each subset $C_i \in C$ and each element $x_j \in C_i$, there is an operator

$$b_1^{ij} = \langle q(V_1, i), w_1^j = 0; w_1^j = 1\rangle,$$

and a corresponding symmetric operator

$$b_1^{ij'} = \langle q(V_1, i), w_1^j = 1; w_1^j = 0\rangle.$$

In other words, the pre-condition of b_1^{ij} and $b_1^{ij'}$ is that all variables v_1^m for $m \in [k]$ are 0, except v_1^i, which is 1. The initial state is given by $\mathsf{init} = (p(V_1, 0), p(W_1, 0))$ and the goal state by $\mathsf{goal} = (p(V_1, 0), p(W_1, 1))$.

Lemma 1. *P_1 belongs to the class 3S.*

Proof. The causal graph of P_1 is acyclic as shown in Figure 1. Moreover, it follows from the definition of operators that each state variable is symmetrically reversible. $\square$

Lemma 2. *Let C' be a minimal subset that covers S, and let $p = |C'|$. An optimal plan for P_1 has length $2p + n$.*

Proof. We describe an optimal plan for P_1. For each subset $C_i \in C'$, use the operator a_1^i to set v_1^i to 1. For each element $x_j \in C_i$, unless w_1^j has already been set to 1, use b_1^{ij} to set w_1^j to 1. Finally, use $a_1^{i'}$ to reset v_1^i to 0. Since C' is a covering subset, this plan sets the value of each w_1^j to 1 using a total of $2p + n$ operators. Moreover, it is impossible to use less operators to solve P_1 since that would require a smaller covering subset, contradicting that C' is minimal. □

Corollary 3. *An optimal plan for moving from the goal state back to the initial state has length $2p + n$.*

Proof. Follows immediately from the fact that all state variables are symmetrically reversible. All we need to do is replace each b_1^{ij} with $b_1^{ij'}$ in the solution for P_1. □

We generalize P_1 and reduce `MINIMUM SET-COVER` to a planning problem P_t, $t \geq 1$. This time, each clause $C_i \in C$ corresponds to t state variables $v_1^i, \ldots, v_t^i$ and each element $x_j \in S$ corresponds to t state variables $w_1^j, \ldots, w_t^j$, so $V = V_1 \cup W_1 \cup \ldots \cup V_t \cup W_t$. The operators a_1^i and b_1^{ij} remain the same, as well as their symmetric counterparts. For each $u \in [2, t]$ and each clause $C_i \in C$, there is an operator

$$a_u^i = \langle p(V_{u-1}, 0), p(W_{u-1}, 0), v_u^i = 0; v_u^i = 1\rangle,$$

and a corresponding symmetric operator

$$a_u^{i'} = \langle p(V_{u-1}, 0), p(W_{u-1}, 0), v_u^i = 1; v_u^i = 0\rangle.$$

For each $u \in [2, t]$, each clause $C_i \in C$, and each element $x_j \in C_i$, there is an operator

$$b_u^{ij} = \langle p(V_{u-1}, 0), p(W_{u-1}, 1), q(V_u, i), w_u^j = 0; w_u^j = 1\rangle,$$

and a corresponding symmetric operator

$$b_u^{ij'} = \langle p(V_{u-1}, 0), p(W_{u-1}, 1), q(V_u, i), w_u^j = 1; w_u^j = 0\rangle.$$

The initial state is $\mathsf{init} = (p(V_1, 0), \ldots, p(W_t, 0))$ and the goal state is $\mathsf{goal} = (p(V_t, 0), p(W_t, 1))$.

Lemma 4. *P_t belongs to the class 3S.*

Proof. If we place state variables in the causal graph in the order $V_1, W_1, \ldots, V_t, W_t$, all edges go from left to right, ensuring acyclicity. As an example, the causal graph of P_2 appears in Figure 2. From the definition of operators it follows that each state variable is symmetrically reversible. □

Lemma 5. *Let L_t denote the length of an optimal plan solving P_t. For each $t > 1$, $L_t = 2pL_{t-1} + 2p + n$.*

Proof. To solve P_t it is necessary to use a_t^i to set v_t^i to 1 for each $C_i \in C'$, use operators b_t^{ij} to set each w_t^j to 1, and use $a_t^{i'}$ to reset v_t^i to 0. Each time we do this we have to solve P_{t-1} to satisfy the pre-condition of b_t^{ij}, and then again (returning P_{t-1} to its initial state) to satisfy the pre-condition of $a_t^{i'}$. In total, this uses $2pL_{t-1} + 2p + n$ operators. □

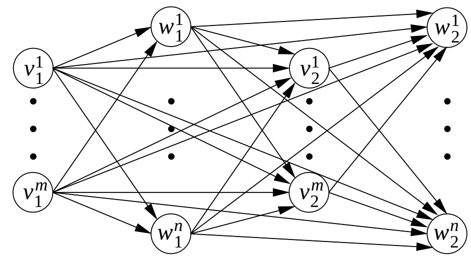

Figure 2: Causal graph of the planning problem P_2

Lemma 6. *An optimal plan for solving P_t contains a total of $(2p + n)\sum_{u=0}^{t-1}(2p)^u$ operators.*

Proof. By induction. For P_1, we know that the optimal plan has length $2p + n = (2p + n)\sum_{u=0}^{0}(2p)^u$. For $t > 1$, assume $L_{t-1} = (2p + n)\sum_{u=0}^{t-2}(2p)^u$. From Lemma 5 we have

$$\begin{aligned} L_t &= 2pL_{t-1} + 2p + n = \\ &= 2p(2p + n)\sum_{u=0}^{t-2}(2p)^u + 2p + n = \\ &= (2p + n)\sum_{u=0}^{t-2}(2p)^{u+1} + 2p + n = \\ &= (2p + n)\left[\sum_{u=1}^{t-1}(2p)^u + 1\right] = \\ &= (2p + n)\sum_{u=0}^{t-1}(2p)^u. \quad \square \end{aligned}$$

Theorem 7. *Unless $\mathrm{P} = \mathrm{NP}$, there exists no polynomial-time algorithm for approximating the optimal solution to P_{k+n} to within polynomial factors.*

Proof. Raz and Safra (1997) showed that the problem of approximating the solution to `MINIMUM SET-COVER` to within logarithmic factors is NP-hard. Specifically, there exists $c > 0$ such that unless $\mathrm{P} = \mathrm{NP}$, there exists no polynomial-time algorithm for approximating the optimal solution using less than $(c \log n)p$ clauses.

From Lemma 6, the optimal solution to P_t contains $(2p+n)\sum_{u=0}^{t-1}(2p)^u = \Theta((2p)^{t-1}(2p+n))$ operators. The best approximation of p we can hope for using a polynomial-time algorithm is $(c \log n)p$, so any approximation of the optimal plan contains $\Omega((c \log n)^{t-1}(2p)^{t-1}(2(c \log n)p+n))$ operators, which is at least $(c \log n)^{t-1}$ longer than optimal. If we select $t = k + n$, P_{k+n} contains $|V| = (k + n)^2$ state variables and $(c \log n)^{k+n-1} = (c \log n)^{\sqrt{|V|}-1}$ is exponential in the size of the input. □

$\mathbb{C}_n^{11}$ is NP-hard

Domshlak and Dinitz (2001) introduced the class $\mathbb{C}_n$ of planning problems with chain causal graph (that is, the

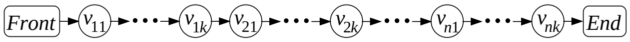

Figure 3: Causal graph of the planning problem $P(F)$

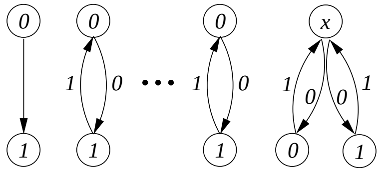

Figure 4: The domain transition graph of the variables $s_1, s_2, \ldots, s_{2n-1}, v_s$.

causal graph is a directed path) and multi-valued variables. In this paper we study the subclass $\mathbb{C}_n^k$ of $\mathbb{C}_n$, $k \geq 2$, that contains all planning problems of $\mathbb{C}_n$ whose variables have domain size at most k. Domshlak and Dinitz showed that there are solvable instances of $\mathbb{C}_n^3$ that require exponentially long plans. Thus, there is no polynomial-time plan generation algorithm for $\mathbb{C}_n^k$ with $k \geq 3$. Giménez and Jonsson (2008) showed that plan existence for $\mathbb{C}_n$ is NP-hard. However, their reduction requires variables with arbitrarily large domains. We present a new reduction that only requires variables with domains of bounded size. More precisely, in this section we prove that plan existence for $\mathbb{C}_n^{11}$ is NP-hard, while in the following section we extend this result to $\mathbb{C}_n^7$.

We show that $\mathbb{C}_n^{11}$ is NP-hard by reduction from CNF-SAT. That is, to every CNF formula F we associate a planning instance $P(F)$ of $\mathbb{C}_n^{11}$ such that $P(F)$ is solvable if and only if F is satisfiable. In this section we describe the reduction, explain the intuitive idea behind it, and finally provide formal proof of its correctness.

Let $F = C_1 \wedge \cdots \wedge C_k$ be a CNF formula on k clauses and n variables $x_1, \ldots, x_n$. We define the planning problem $P(F) = (V, \mathsf{init}, \mathsf{goal}, A)$ as follows. The variable set V is $\{s_i | i \in [2n-1]\} \cup \{v_s\} \cup \{v_{ij} | i \in [k], j \in [n]\} \cup \{v_e\} \cup \{e_i | i \in [2n-1]\}$, with domains $D(s_i) = D(e_i) = D(v_e) = \{0, 1\}$ for $i \in [2n-1]$, $D(v_s) = \{0, 1, x\}$, and $D(v_{ij}) = \{g_x, g_0, g_1, a_x, a_0, a_1, b_0, b_1, c_x, c_0, c_1\}$ for $i \in [k], j \in [n]$. The initial state is defined by $\mathsf{init}(s_i) = \mathsf{init}(e_i) = \mathsf{init}(v_e) = 0$, $\mathsf{init}(v_s) = x$, and $\mathsf{init}(v_{ij}) = a_x$ for $i \in [k], j \in [n]$, and the goal state is a partial state defined by $\mathsf{goal}(v_{in}) = g_x$ for each $i \in [k]$ and $\mathsf{goal}(e_i) = i \bmod 2$ for each $i \in [2n-1]$.

The set of operators A is described by the domain transition graphs in Figures 4, 5 and 6. Dashed edges correspond to operators that depend on the formula F. The formal description of operators for variables s_i, e_i, v_s, and v_e, $i \in [2n-1]$, is given by

- $\langle s_1 = 0; s_1 = 1 \rangle$.
- for each $i \in [2, 2n-1]$, $\langle s_{i-1} = 0, s_i = 0; s_i = 1 \rangle$ and $\langle s_{i-1} = 1, s_i = 1; s_i = 0 \rangle$.
- for each $m \in \{0, 1\}$, $\langle s_{2n-1} = 0, v_s = x; v_s = m \rangle$ and $\langle s_{2n-1} = 1, v_s = m; v_s = x \rangle$.
- $\langle v_{kn} = p, v_e = 0; v_e = 1 \rangle$ for $p \in \{a_0, a_1, g_0, g_1\}$, and $\langle v_{kn} = p, v_e = 1; v_e = 0 \rangle$ for $p \in \{a_x, g_x\}$,
- for each $i \in [2, 2n-1]$, $\langle e_{i-1} = 1, e_i = 0; e_i = 1 \rangle$ and $\langle e_{i-1} = 0, e_i = 1; e_i = 0 \rangle$.

A similar formal description of the operators for the variables v_{ij} would require too much space, so we refer to the corresponding figures. Operators corresponding to dashed edges depend on F in the following way. For a variable v_{ij}, if the literal x_i appears in C_j, the dashed edge from a_x to b_1 instead points to g_1; if the literal $\overline{x_i}$ appears in C_j, the dashed edge from a_x to b_0 instead points to g_0. Note that no two consecutive edges have the same label; to change the value of a variable twice it is necessary to change the value of its predecessor in between.

Intuition

The intuition behind the planning problem $P(F)$ is as follows. We can view the problem as having three parts: one containing variables $\{s_i\}_{i \in [2n-1]}$ and v_s, one containing variables $\{v_{ij}\}_{i \in [k], j \in [n]}$, and one containing variables v_e and $\{e_i\}_{i \in [2n-1]}$ (cf. the causal graph in Figure 3, where $Front$ corresponds to the chain $s_1, \ldots, s_{2n-1}, v_s$, and End to $v_e, e_1, \ldots, e_{2n-1}$). The purpose of the first part is to generate a message of n bits representing a formula assignment σ. The purpose of the middle part is to check whether the assignment σ satisfies the formula F. Finally, the purpose of the last part is to ensure that the message is passed all the way through to the end of the chain.

The first part is designed so that the value of the variable v_s can change exactly $2n$ times. The only way to change the value of v_s is to move from x to either 0 or 1 and back to x. Thus, changing the value of v_s $2n$ times corresponds to n decisions of whether to move to 0 or 1. We use the value changes of v_s to represent a formula assignment σ by letting $\sigma(x_i)$ be the result of the i-th decision of moving to 0 or 1.

The second part has two purposes: passing on the message generated by v_s, and checking whether the corresponding assignment satisfies the formula F. Note that each value in the domain of v_{ij} has subscript x, 0, or 1. The operators are defined in a way that forces v_{ij} to move to a value with subscript $m \in \{x, 0, 1\}$ if the predecessor of v_{ij} is in a value with subscript m (or, in the case of v_{11}, the value of v_s is m). Thus, v_{ij} is forced to propagate the sequence of 0's and 1's separated by x's.

For each clause C_i and each variable x_j of the formula F, the variable v_{ij} checks whether C_i is satisfied by the assignment $\sigma(x_j)$ to x_j. To do this, v_{ij} has to be able to identify the j-th bit of the message. Unless clause C_i is satisfied by

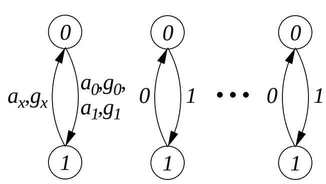

Figure 5: The domain transition graph of the variables $v_e, e_1, \ldots, e_{2n-1}$.

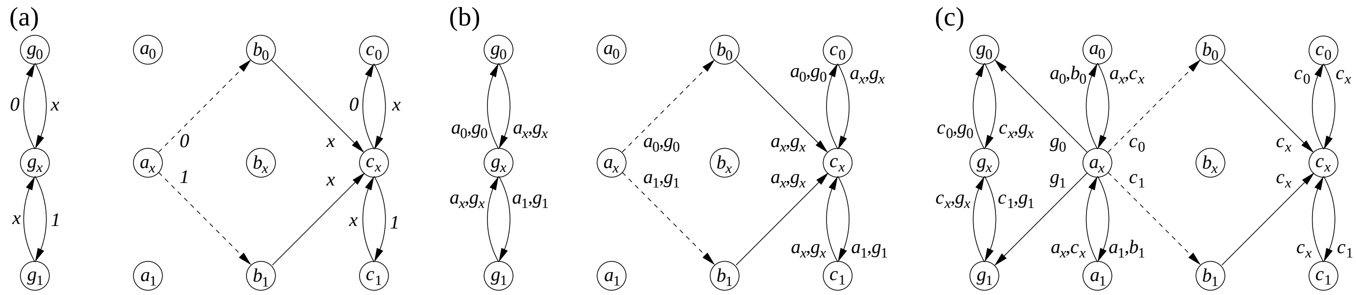

Figure 6: Domain transition graph of (a) v_{11}, (b) v_{i1} for $i \neq 1$, and (c) v_{ij} for $j \neq 1$.

the assignment to an earlier variable x_q, $q < j$, v_{ij} remains within the subdomain $\{a_x, a_0, a_1\}$ prior to arrival of the j-th bit. If $j = 1$, v_{i1} should react to the first bit. Otherwise, v_{ij} identifies the j-th bit by the fact that $v_{i(j-1)}$ is in a value labeled c_m, $m \in \{0, 1\}$. As a result of receiving the j-th bit, v_{ij} either moves to g_m (if the assignment $\sigma(x_j) = m$ satisfies C_i) or b_m (if $\sigma(x_j) = m$ does not satisfy C_i).

If v_{ij} moves to g_m, all subsequent variables v_{it}, $t \in [j + 1, n]$, also have to move to g_m. Regardless of what the rest of the message is, they remain within the subdomain $\{g_x, g_0, g_1\}$. Since the subscript at the end of the message is x, this ensures that the goal state $\mathsf{goal}(v_{in}) = g_x$ is satisfied (representing that the clause C_i has been satisfied). If v_{ij} moves to b_m, it is forced to move to c_x next, and from there to either c_0 or c_1, correctly indicating to variable $v_{i(j+1)}$ that the $(j + 1)$-th bit of the message has arrived.

The last part is designed so that the goal state of variables $\{e_i\}_{i\in[2n-1]}$ can only be satisfied if the value of v_e changes at least $2n$ times. This is only possible if the value of v_s as well as each variable in the set $\{v_{ij}\}_{i\in[n],j\in[k]}$ also changes at least $2n$ times. The only purpose of this part is to ensure that the whole message generated by v_s is passed through to the end of the middle part. Otherwise, variables in $\{v_{ij}\}_{i\in[n],j\in[k]}$ can "cheat" by not propagating a bit and instead waiting for the next bit to arrive before moving.

Proof of correctness

Let F be a CNF formula. We prove the correctness of the reduction $P(F)$, that is, we show that F is satisfiable if and only if $P(F)$ is solvable. To this end, we introduce some notation and several easy lemmas to characterize the plans solving $P(F)$.

In what follows, when we refer to a (partial) plan we mean any valid sequence of operators, not necessarily solving $P(F)$. Let π be a (partial) plan of $P(F)$, and let v be a variable of $P(F)$. We denote by $T(v)$ the number of times that v changes value during the execution of π.

Lemma 8. *Let π be a (partial) plan of planning problem $P(F) = (V, \mathsf{init}, \mathsf{goal}, A)$. Let $v \in V \setminus \{s_1\}$ be a variable of $P(F)$, and let v' be its causal graph predecessor. Then,*

a) $T(v) \leq T(v') + 1$, *and*

b) $T(v) \leq T(v')$ *if* $v \in \{v_{11}, \ldots, v_{kn}, v_e, e_1, \ldots, e_{2n-1}\}$.

Proof. None of the domain transition graphs of $P(F)$ has two consecutive edges with the same label. This means that, for a fixed value of v', variable v cannot change twice in π without v' changing its value. Thus variable v can change at most $T(v') + 1$ times: once for every one of the $T(v')$ changes of v', and an additional change if variable v can change value before v' does it, that is, if the domain transition graph of v has some edge starting at $\mathsf{init}(v)$ with value $\mathsf{init}(v')$. Since this additional change may only happen if v is s_i for $i \in [2n - 1]$ or v_s, it follows that $T(v) \leq T(v')$ if v is one of the remaining variables. □

Lemma 9. *Let π be a (partial) plan of planning problem $P(F)$. Then,*

- $T(s_i) \leq i$ *for* $i \in [2n - 1]$, *and*
- $T(v_s) \leq 2n$.

Proof. Variable s_1 can only change once, so $T(s_1) \leq 1$. Variable s_{i-1} is the causal graph predecessor of s_i for $i \in [2, 2n - 1]$, and s_{2n-1} is the predecessor of v_s. The claim follows by induction due to case (a) of Lemma 8. □

Lemma 10. *Let π be a plan solving the planning problem $P(F)$. Then,*

- $T(e_i) \geq 2n - i$ *for* $i \in [2n - 1]$, *and*
- $T(v_e) \geq 2n$.

Proof. We use a descending induction on i to show that $T(e_i) \geq 2n - i$ for $i \in [2n - 1]$. The base case $T(e_{2n-1}) \geq 1$ holds trivially, since the initial and goal values of e_{2n-1} are different. Now assume the induction hypothesis $T(e_{i+1}) \geq 2n - i - 1$ true for some i. By case (b) of Lemma 8, $T(e_{i+1}) \leq T(e_i)$. On the other hand, since $\mathsf{goal}(e_i) \neq \mathsf{goal}(e_{i+1})$ but π solves $P(F)$, it follows that $T(e_i) \neq T(e_{i+1})$. Hence $T(e_{i+1}) < T(e_i)$, that is, $T(e_i) \geq 2n - i$, as claimed.

Variable v_e precedes e_1 in the causal graph, so the previous argument applied verbatim implies $T(v_e) \geq 2n$. □

Corollary 11. *Let π be a plan solving the planning problem $P(F)$. Then, $T(v) = 2n$ for the variables $v \in \{v_s, v_{11}, \ldots, v_{kn}, v_e\}$.*

Proof. By case (b) of Lemma 8 we have that $T(v_s) \geq T(v_{11}) \geq \cdots \geq T(v_{kn}) \geq T(v_e)$. But, by Lemmas 9 and 10, all these values are equal to $2n$, since $2n \geq T(v_s)$ and $T(v_e) \geq 2n$. □

Until now, we have shown that plans solving $P(F)$, if any, are of a very specific form: variables v_s, v_{ij} for $i \in [k], j \in [n]$, and v_e in the central region of the causal graph change values the maximal number of times possible. Consider the sequence of $2n+1$ values that variable v_s takes in a plan solving $P(F)$, that is, $x, m_1, x, m_2, \ldots, m_n, x$, where $m_j \in \{0,1\}$ for all $j \in [n]$. We denote by m_π the message $m_1, m_2, \ldots, m_n$ induced by π, and we denote by σ_π the formula assignment defined by $\sigma_\pi(x_j) = m_j$ for all $j \in [n]$.

We say that a (partial) plan π is *admissible* if, even if not actually solving $P(F)$, it behaves in the same way valid plans do. A formal definition follows.

Definition 12. *Let π be a (partial) plan of planning problem $P(F) = (V, \mathsf{init}, \mathsf{goal}, A)$, and let $v \in V$ be a variable of $P(F)$. For $t \in [0, T(v)]$, we define $V(v,t)$ as the value that v has after plan π changes its value for the t-th time. In particular, $V(v,0) = \mathsf{init}(v)$. For $t \in [T(v)]$, we define $P(v,t) \in [|\pi|]$ as the position in plan π of the operator that changes the value of v for the t-th time. Obviously, $P(v,t) < P(v,t+1)$ for any t.*

We say that π is admissible *if, for any variable v in $\{v_{11}, \ldots, v_{kn}, v_s\}$ and its causal graph predecessor v', it holds that $P(v',t) < P(v,t)$ for $t \in [T(v)]$, and $P(v,t) < P(v',t+1)$ for $t \in [T(v')-1]$. In other words, variable v changes exactly once between two changes of v'.*

Lemma 13. *If plan π solves the planning problem $P(F)$, then π is admissible.*

Proof. By Corollary 11, any pair of consecutive variables v', v in $\{v_s, v_{11}, \ldots, v_{kn}, v_e\}$ change values $2n$ times. That is, plan π contains $2n$ operators changing v at positions $P(v,1) < P(v,2) < \cdots < P(v,2n)$, and $2n$ operators changing v' at positions $P(v',1) < P(v',2) < \cdots < P(v',2n)$.

Recall that, as shown when proving case (a) of Lemma 8, no variable v can change value twice without its causal graph predecessor v' changing in between. Similarly, as shown when proving case (b) of the same lemma, $P(v',1) < P(v,1)$ if $v \in \{v_{11}, \ldots, v_{kn}, v_s\}$. Then, we have $2n$ operators changing v, and $2n$ positions to place them, namely, $2n-1$ positions between $P(v',t')$ and $P(v',t'+1)$ for each $t' \in [2n-1]$, and an additional position after $P(v',2n)$. Clearly, the only possibility is to interleave them, $P(v',1) < P(v,1) < P(v',2) < P(v,2) < \cdots < P(v,2n-1) < P(v',2n) < P(v,2n)$. Hence π is admissible. □

In what follows, S_i^t denotes the values of variables for a clause C_i produced by a plan π. In Lemma 16 we show that, if an admissible plan π induces an assignment σ_π, then these S_i^t can only have a single form, namely Q_i^t, which we introduce in Definition 15.

Definition 14. *Let π be a (partial) plan of the planning problem $P(F)$, and let C_i be a clause of F. We define S_i^t, the* partial state *of clause C_i after the t-th change, as the partial state $S_i^t(v) = V(v,t)$ defined on variables $v \in \{v_{i1}, \ldots, v_{in}\}$.*

For instance, for any plan π, S_i^0 is the partial state $\langle v_{i1} = a_x, \ldots, v_{in} = a_x \rangle$ and, if π is valid, then $S_i^{2n}(v_{in})$ must be g_x to satisfy the goal state. To simplify the notation, we may write a partial state S_i^t as if it were a word of n symbols, like in $S_i^0 = a_x \cdots a_x$.

Definition 15. *Let σ be a satisfying assignment of F. For the planning problem $P(F)$ we define the* partial state Q_i^t *induced by σ for clause C_i at time t, where $t \in [0, 2n]$, as follows. Let $q \in [n]$ be the smallest index such that $\sigma(x_q) = 1$ and C_i contains x_q, or $\sigma(x_q) = 0$ and C_i contains $\overline{x_q}$. Finally, let $j \in [n]$, and let $m = \sigma(x_j) \in \{0,1\}$.*

a) If $q > j$, then

$$\begin{array}{rlll} Q_i^{2j-2} & = \overbrace{c_x \cdots c_x}^{j-1} & a_x & a_x \cdots a_x \\ Q_i^{2j-1} & = c_m \cdots c_m & b_m & a_m \cdots a_m \\ Q_i^{2j} & = c_x \cdots c_x & c_x & a_x \cdots a_x \end{array}$$

b) If $q = j$, then

$$\begin{array}{rlll} Q_i^{2j-2} & = \overbrace{c_x \cdots c_x}^{j-1} & a_x & a_x \cdots a_x \\ Q_i^{2j-1} & = c_m \cdots c_m & g_m & g_m \cdots g_m \\ Q_i^{2j} & = c_x \cdots c_x & g_x & g_x \cdots g_x \end{array}$$

c) If $q < j$, then

$$\begin{array}{rllll} Q_i^{2j-2} & = \overbrace{c_x \cdots c_x}^{q-1} & \overbrace{g_x \cdots g_x}^{j-q} & g_x & g_x \cdots g_x \\ Q_i^{2j-1} & = c_m \cdots c_m & g_m \cdots g_m & g_m & g_m \cdots g_m \\ Q_i^{2j} & = c_x \cdots c_x & g_x \cdots g_x & g_x & g_x \cdots g_x \end{array}$$

Partial states of the form Q_i^{2p} for $p \in [n-1]$ are defined twice, but it is easy to check that the definitions coincide.

Lemma 16. *Let π be an admissible plan of the planning problem $P(F)$. Then, for any partial state S_i^t induced by π, it holds that $S_i^t = Q_i^t$, where Q_i^t is the corresponding partial state induced by σ_π.*

Proof. The domain transition graphs of variables $v = v_{ij}$ for $i \in [k]$, $j \in [n]$ have the following characteristic in common: all edges starting at the same vertex in the domain transition graph of v have different labels. That is, if v' is the causal graph predecessor of v, and (y, y') is a pair of values $y \in D(v)$ and $y' \in D(v')$, then there is at most one operator of the form $\langle v = y, v' = y'; v = z \rangle$ for some $z \in D(v)$. Hence the values that variable v_s takes during the execution of π determine uniquely all partial states S_i^t.

Hence we only need to check that any partial state S_i^t is indeed Q_i^t. This follows from a double induction on t and i. The base case $t = 0$ follows trivially, since S_i^0 is the initial state init restricted to variables $\{v_{i1}, \ldots, v_{in}\}$, which coincides with Q_i^0, that is, Q^{2j-2} with $j = 1$ in cases (a) and (b) of Definition 15.

Then, assuming the inductive hypothesis holds for Q_i^{2j-2} for a fixed $j \geq 1$ and all $i \in [k]$, we prove that it also holds for partial states Q_i^{2j-1} and Q_i^{2j}. To show this, one also proceeds by induction, this time on i. This is necessary because the induction hypothesis must hold for Q_{i-1}^t before proving it for Q_i^t, since the causal graph predecessor of v_{i1} is $v_{(i-1)n}$.

To complete the proof, the following checking remains. For any $i \in [k]$ and $j, t, q \in [n]$, let $v = v_{ij}$ and let v' be its causal graph predecessor. Then there exists an edge in the domain transition graph of v from value $Q_i^{t-1}(v)$ to value $Q_i^t(v)$ having label $Q_i^t(v')$ if $j > 1$, label $Q_{i-1}^t(v')$ if $j = 1$ and $i > 1$, and label $V(v_s, t)$ if $j = i = 1$. □

Proposition 17. *Let F be a CNF formula. If π is a plan solving the planning problem $P(F)$, then the formula assignment σ_π satisfies F.*

Proof. This is just a direct consequence of Lemma 16. A valid plan π is also admissible, so the partial states S_i^t are equal to the partial states Q_i^t induced by σ_π, for all $t \in [0, 2n]$ and $i \in [k]$. On the other hand, since π is a valid plan, it must hold that $S_i^{2n}(v_{in}) = \mathsf{goal}(v_{in})$ for all $i \in [k]$. This implies that Q_i^{2n} necessarily has to follow either case (b) or (c) of Definition 15. Thus $q \leq n$, which means that the assignment σ_π satisfies clause C_i. □

Proposition 18. *Let F be a CNF formula. If σ is an assignment satisfying F, then there exists a valid plan π solving $P(F)$ such that $\sigma_\pi = \sigma$.*

(Sketch). The plan π contains $2n$ operators that change the value of variable v_s, to form the message $m_\pi = \sigma(x_1)\sigma(x_2)\cdots\sigma(x_n)$. While doing so, it propagates this message up to variable v_e; by the proof of Lemma 16, variables v_{ij} can change values in a unique way so as to retain admissibility. Note that, although Lemma 16 completely determines the values that variables v_{ij} get, many different operator orderings may achieve this result; any one shall do.

Finally, it is easy to complement this plan π with the necessary operators to allow variable v_s to change values $2n$ times, and the operators to allow variables e_i to reach their goal states. Since σ is a satisfying assignment, for any clause C_i, $i \in [k]$ there is a $q \in [n]$ such that C_i is satisfyed by x_q, so $S_i^{2n}(v_{in}) = g_x$, as required by the goal state. □

Theorem 19. *The problem* CNF-SAT *is polynomial-time reducible to $\mathbb{C}_n^{11}$.*

Proof. A direct consequence of Propositions 17 and 18, and the fact that we can produce the planning problem $P(F)$ in polynomial time. □

$\mathbb{C}_n^7$ is NP-hard

We describe how the reduction introduced in the previous section can be improved in a way in which the planning problem $P(F)$ only needs domains of size 7. The new reduction we obtain follows the same idea, but we use a much more involved construction to check if the assignment σ_π satisfies clause C_i. Previously, we had n variables $\{v_{ij}\}_{j\in[n]}$, and the individual role of one variable v_{ij} was, essentially, to check whether the j-th bit $\sigma(x_j)$ of the message being transmitted makes clause C_i become true. Now, we replace each variable v_{ij} with three different variables v_{ij}^1, v_{ij}^2 and v_{ij}^3, that will collectively play the same role.

Figure 8 shows the domain transition graphs of variables v_{ij}^1, for $j \neq 1$, and v_{ij}^2 and v_{ij}^3, for $i \in [k], j \in [n]$. The

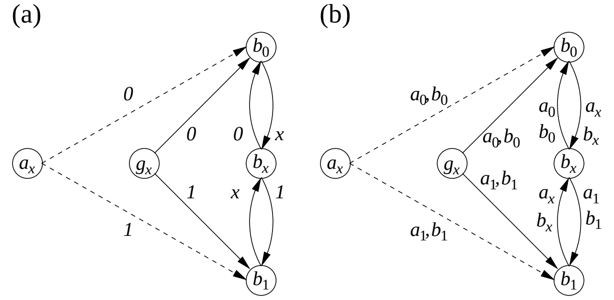

Figure 7: Domain transition graph of (a) v_{11}^1, and (b) v_{i1}^1 for $i \in [2, k]$.

dashed edge in the domain transition graphs of v_{ij}^1 from a_x to b_0, and the edge from a_x to b_1, will point instead to g_x if, respectively, clause C_i contains literal $\overline{x_j}$ or literal x_j. Figure 7 shows the domain transition graphs of variables v_{i1}^1 with $i \in [2, k]$ and variable v_{11}^1. Again, the dashed edges point to g_x if clause C_i contains a literal $\overline{x_1}$, or a literal x_1. The initial value for all variables of type v_{ij}^k is a_x; the goal state goal is $\mathsf{goal}(v_{in}^2) = g_x$ for $i \in [k]$, and undefined for all other variables of this type.

We briefly explain how this new construction achieves the same goal than the previous one. The reader is advised to attach the same interpretation as in the previous reduction to values $\{a_0, a_1, a_x\}$ and $\{g_0, g_1, g_x\}$, that is, respectively, that the j-th bit of the message has not been received yet, and that the clause has been made true. Values $\{b_0, b_1, b_x\}$ in v_{ij}^1 and v_{ij}^3 convey the meaning that the j-th bit has been already processed; in variable v_{ij}^2, value b_x means that the j-th bit of the message has not satisfied the clause C_i. In what remains of the section, we provide a high-level description of the workings of these variables.

Consider a triplet v_{ij}^1, v_{ij}^2 and v_{ij}^3. If causal graph predecessor v' of variable v_{ij}^1 holds a value in $\{a_0, a_1, a_x\}$, then v_{ij}^1 replicates its behaviour, passing along the message. If, however, v' belongs to $\{b_0, b_1, g_x\}$, variable v_{ij}^1 will move to the right hand side of the transition graph. Depending on the dashed edges, v_{ij}^1 will pass through g_x. When this happens, all variables of the form $v_{ij'}^q$ for $j' \in [j, n]$ and $q \in [3]$ can propagate this fact since, indeed, all such variables can jump from a_x to g_x when its predecessor has value g_x.

Note that, in contrast with the previous reduction, variable v_{ij}^1 is allowed to change twice without v' having changed: once from a_x to g_x to provoke the shortcut for all remaining variables, and then move again from g_x to b_m for $m \in \{0, 1\}$ to actually propagate the j-th bit of the message. However, to reach the goal state for each v_{in}^2, where $i \in [k]$, one of the v_{ij}^1 has to move to g_x for the correct bit of the message.

As we have seen, variable v_{ij}^1 checks whether $\sigma(x_j)$ satisfies the clause C_i. However, it does not have enough values to remember if it visited value g_x or not during the course of the plan execution. Variable v_{ij}^2, on the contrary, is able to remember it, by keeping separate values $\{g_0, g_1, g_x\}$ to keep memory of this fact and, at the same time, allow the

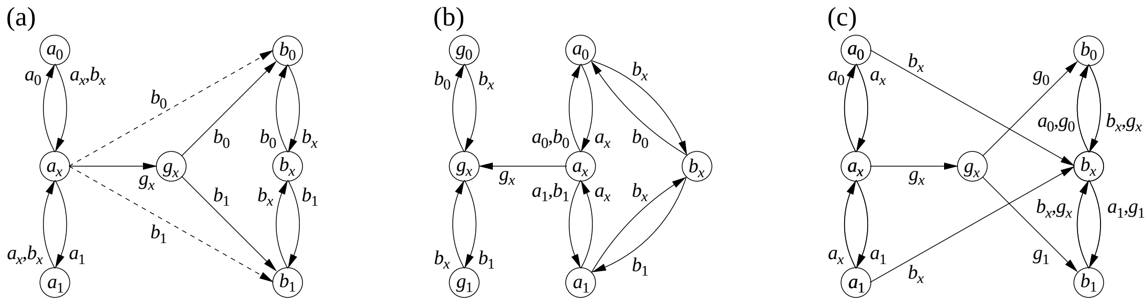

Figure 8: Domain transition graphs of (a) v^1_{ij} for $i \in [k], j \in [2, n]$; (b) v^2_{ij} and (c) v^3_{ij} for $i \in [k], j \in [n]$.

propagation of messages. Note also that, as a side effect, the domain transition graph of variable v^2_{ij} also provides a convenient *delay* when v^1_{ij} moves to b_0 or b_1, since, as can be seen, v^2_{ij} must first move to a_0 or a_1; only at the next round will v^2_{ij} follow v^1_{ij} into b_x. This delay is the one that allows the variables to *count* the number of bits being transmitted, and only act on the j-th one.

Finally, variable v^3_{ij} is necessary because the domain of variable v^2_{ij} does not contain values b_0 and b_1, so the causal graph succesor of v^2_{ij} cannot distinguish between situations a and b when v^2_{ij} is passing along a bit 0 or 1. Variable v^3_{ij}, as can be seen, essentially replicates the values of v^2_{ij}, but correctly tracks situations a and b at all times.

Conclusion

In this paper we have presented novel complexity results for two classes of planning problems. Although presented in terms of the specific classes 3S and $\mathbb{C}^7_n$, it is worth mentioning that the implications are of a more general nature.

First, we showed that finding an approximation polynomially close to optimal is NP-hard for planning problems in the class 3S. Consequently, the best we can hope for is an approximation that is exponentially longer than optimal. Note that although the class 3S allows for three types of operators, we have only used one type of operator in our reduction, namely symmetrically reversible. It follows that the very same reduction automatically extends to any other class of planning problems allowing symmetrically reversible operators and graphs of the form we use in the reduction. We conjecture that it is possible to drop the requirement that the reversible operators be symmetrical, which would make the proof even more general.

We also showed that deciding whether or not a planning problem has a solution is NP-hard for the class $\mathbb{C}_n$, even if the variables of the problem have domain size at most 7. Clearly, this result generalizes to any class of planning problems that is defined in terms of causal graphs whose depth is not bounded. Here, the depth is the maximum length of shortest paths between pairs of variables of the causal graph. Some of the causal graph may be connected, but if part of the causal graph has the form of a chain or other sparsely connected structure, even small domain sizes are sufficient to make the problem intractable.

Could this result be extended to smaller domains, like $\mathbb{C}^6_n$? We believe it could, but have been unable to prove it. What about $\mathbb{C}^3_n$? We really doubt it: domain transition graphs of size 3 seem too small to construct a reduction like the one presented. However, so far we have failed to derive a polynomial-time plan existence algorithm for $\mathbb{C}^3_n$.

Acknowledgements

This work was partially funded by APIDIS and MEC grant TIN2006-15387-C03-03.

References

Brafman, R., and Domshlak, C. 2003. Structure and Complexity in Planning with Unary Operators. *Journal of Artificial Intelligence Research* 18:315–349.

Domshlak, C., and Dinitz, Y. 2001. Multi-Agent Off-line Coordination: Structure and Complexity. In *Proceedings of the 6th European Conference on Planning*, 277–288.

Giménez, O., and Jonsson, A. 2008. The Complexity of Planning Problems with Simple Causal Graphs. *Journal of Artificial Intelligence Research* 31:319–351.

Helmert, M.; Mattmüller, R.; and Röger, G. 2006. Approximation Properties of Planning Benchmarks. In *Proceedings of the 17th European Conference on Artificial Intelligence*, 585–589.

Helmert, M. 2006. The Fast Downward Planning System. *Journal of Artificial Intelligence Research* 26:191–246.

Jonsson, P., and Bäckström, C. 1998. Tractable plan existence does not imply tractable plan generation. *Annals of Mathematics and Artificial Intelligence* 22(3-4):281–296.

Jonsson, A. 2007. The Role of Macros in Tractable Planning Over Causal Graphs. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 1936–1941.

Katz, M., and Domshlak, C. 2007a. Structural patterns of tractable sequentially-optimal planning. In *Proceedings of the 17th International Conference on Automated Planning and Scheduling*.

Katz, M., and Domshlak, C. 2007b. Structural Patterns Heuristics: Basic Idea and Concrete Instance. In *Proceedings of the Workshop on Heuristics for Domain-independent Planning at ICAPS-07*.

Raz, R., and Safra, S. 1997. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the 29th ACM Symposium on Theory of Computing*, 475–484.