Structural Patterns Heuristics: Basic Idea and Concrete Instance

Michael Katz
Faculty of Industrial Engineering
Technion, Israel
dugi@tx.technion.ac.il

Carmel Domshlak*
Faculty of Industrial Engineering
Technion, Israel
dcarmel@ie.technion.ac.il

Abstract

Considering admissible heuristics for cost-optimal planning, we present a generalization of pattern-database homomorphism abstractions, called "structural patterns". The basic idea of structural patterns boils down to projecting the problem in hand to provably tractable fragments of optimal planning. The key motivation behind this generalization of PDBs is to alleviate the requirement for the projections to be of a low dimensionality.

Introduction

The difference between various algorithms for planning as heuristic search is mainly in the heuristic functions they define and use. Most typically, an (admissible) heuristic function for domain-independent planning is defined as the (optimal) cost of achieving the goals in an *over-approximating abstraction* of the planning problem in hand (Pearl 1984; Bonet & Geffner 2001). Such an abstraction is obtained by relaxing certain constraints that are present in the specification of the real problem, and the desire is to obtain a *tractable* (that is, solvable in polynomial time), and, at the same time, *informative* abstract problem. The main question is thus: What constraints should we relax to obtain such an effective over-approximating abstraction?

Conceptually, one can distinguish between homomorphism and embedding abstractions, and the former are our focus in this paper. A *homomorphism abstraction* systematically contracts several states to create a single abstract state. Most typically, such a state-gluing is obtained by projecting the original problem onto a subset of its parameters, as if ignoring the constraints that fall outside the projection. Homomorphisms have been successfully explored in the scope of domain-independent pattern database (PDB) heuristics (Edelkamp 2001; Haslum, Bonet, & Geffner 2005), inspired by the (similarly named) problem-specific heuristics for search problems such as (k^2-1)-puzzles, Rubik's Cube, etc. (Culberson & Schaeffer 1998). A core property of the PDB heuristics is that the problem is projected onto a space of small (up to logarithmic) dimensionality so that reachability analysis in that space could be done by exhaustive search. Note that this constraint implies an inherent scalability limitation of the PDB heuristics—as the problems of interest grow, limiting patterns to logarithmic dimensionality will unavoidably make them less and less informative with respect to the original problems.

In this paper we suggest a generalization of the PDB abstractions to what we call *"structural patterns"*. In itself, the idea of structural patterns is simple, and it corresponds to projecting the original problem to provably tractable fragments of optimal planning. At least theoretically, moving to structural patterns alleviates the requirement for the projections to be of a low dimensionality. Moreover, we show that the idea is not of a theoretical interest only. We introduce a concrete structural patterns abstraction based on decomposing the problem in hand along its causal graph, and show that the induced admissible heuristic can provide more informative estimates than its state-of-the-art alternatives.

Formalism and Notation

Problems of *classical planning* correspond to reachability analysis in state models with deterministic actions and complete information. In this work we focus on state models corresponding to the SAS$^+$ formalism (Bäckström & Nebel 1995) that captures problems with multi-valued state variables.

Definition 1 *A* SAS$^+$ *problem instance is given by a quadruple* $\Pi = \langle V, A, I, G \rangle$*, where:*

- $V = \{v_1, \ldots, v_n\}$ *is a set of* state variables*, each associated with a finite domain* $dom(v_i)$*.*
- *the* initial state I *is a complete assignment, and the* goal G *is a partial assignment to* V*.*

*The work of both authors is partly supported by Israel Science Foundation and C. Wellner Research Fund.

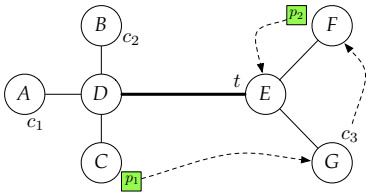


Figure 1: Logistics example adapted from Helmert (2006). Deliver p_1 from C to G, and p_2 from F to E using the cars c_1, c_2, c_3 and truck t, and making sure that c_3 ends up at F. The cars may only use city roads (thin edges), the truck may only use the highway (thick edge).

- $A = \{a_1, \ldots, a_N\}$ *is a finite set of* actions*, where each action a is a pair $\langle \mathsf{pre}(a), \mathsf{eff}(a) \rangle$ of partial assignments to V called* preconditions *and* effects*, respectively. Each action $a \in A$ is associated with a non-negative real-valued cost $\mathcal{C}(a)$.*

An action a is applicable in a state $s \in dom(V)$ iff $s[v] = \mathsf{pre}(a)[v]$ whenever $\mathsf{pre}(a)[v]$ is specified. Applying a changes the value of v to $\mathsf{eff}(a)[v]$ if $\mathsf{eff}(a)[v]$ is specified. In this work we focus on *cost-optimal* (also known as *sequentially optimal*) planning in which the task is to find a plan $\rho \in A^*$ for Π minimizing $\sum_{a \in \rho} \mathcal{C}(a)$.

Across the paper we use a slight variation of a Logistics example of Helmert (2006). This example is depicted in Figure 1, and in SAS$^+$ it has

$$\begin{aligned}
V &= \{p_1, p_2, c_1, c_2, c_3, t\} \\
dom(p_1) &= dom(p_2) = \\
&= \{A, B, C, D, E, F, G, c_1, c_2, c_3, t\} \\
dom(c_1) &= dom(c_2) = \{A, B, C, D\} \\
dom(c_3) &= \{E, F, G\} \\
dom(t) &= \{D, E\} \\
I &= \{p_1 \mapsto C, p_2 \mapsto F, t \mapsto E, \\
&\quad c_1 \mapsto A, c_2 \mapsto B, c_3 \mapsto G\} \\
G &= \{p_1 \mapsto G, p_2 \mapsto E, c_3 \mapsto F\}, \\
A &= \{a_1, \ldots, a_{70}\},
\end{aligned}$$

where the actions correspond to all possible loads and unloads, plus single-segment movements of the vehicles. For instance, if action a captures loading p_1 into c_1 at C, then $\mathsf{pre}(a) = \{p_1 \mapsto C, c_1 \mapsto C\}$, and $\mathsf{eff}(a) = \{p_1 \mapsto c_1\}$.

Definition 2 *The* **causal graph** *CG*$(\Pi) = (V, E)$ *of a* SAS$^+$ *problem $\Pi = \langle V, A, I, G \rangle$ is a digraph over the nodes V. An arc (v, v') belongs to CG(Π) iff $v \neq v'$ and there exists an action $a \in A$ such that $\mathsf{eff}(a)[v']$, and either $\mathsf{pre}(a)[v]$ or $\mathsf{eff}(a)[v]$ are specified.*

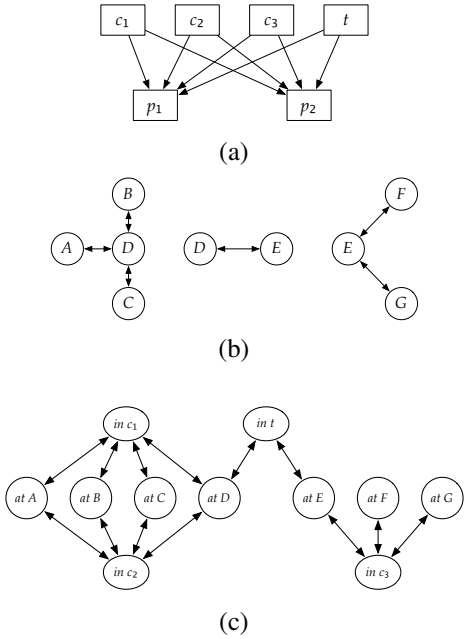


Figure 2: (a) Causal graph; (b) DTGs (labels omitted) of c_1 and c_2 (left), t (centre), and c_3 (right); (c) DTG of p_1 and p_2.

In what follows, for each $v \in V$, by $\mathsf{pred}(v)$ and $\mathsf{succ}(v)$ we refer to the sets of all immediate predecessors and successors of v in *CG*(Π). Likewise, for any partial assignment x to V, and any $V' \subseteq V$, by $x^{[V']}$ we refer to the projection of x onto V'.

Definition 3 *Let $\Pi = \langle V, A, I, G \rangle$ be a* SAS$^+$ *problem, and let $v \in V$. The* **domain transition graph** *DTG*(v, Π) *of v in Π is an arc-labeled digraph over the nodes $dom(v)$ such that an arc (ϑ, ϑ') belongs to DTG(v, Π) iff there is an action $a \in A$ with $\mathsf{eff}(a)[v] = \vartheta'$, and either $\mathsf{pre}(a)[v]$ is unspecified or $\mathsf{pre}(a)[v] = \vartheta$. In that case, the arc (ϑ, ϑ') is labeled with $\mathsf{pre}(a)^{[V \setminus \{v\}]}$ and $\mathcal{C}(a)$.*

Figure 2 depicts the causal and (labels omitted) domain transition graphs for our running example problem Π. Here we note (and later exploit) that Π belongs to the class of *unary-effect*, *1-dependent* SAS$^+$ problems (Katz & Domshlak 2007), that is, for all $a \in A$, we have (i) $|\mathsf{eff}(a)| = 1$, and (ii) if $\mathsf{eff}(a)$ is specified for $V' \subseteq V$, then $|\mathsf{pre}(a)^{[V \setminus V']}| \leq 1$.

From PDBs to Structural Patterns?

Given a problem $\Pi = \langle V, A, I, G \rangle$, each subset of variables $V' \subseteq V$ defines a *pattern abstraction* $\Pi^{[V']} = \langle V', A^{[V']}, I^{[V']}, G^{[V']} \rangle$ by intersecting the initial state, the goal, and all the actions' preconditions and effects

with V' (Edelkamp 2001)[1]. The idea behind the PDB heuristics is elegantly simple. First, we select a (relatively small) set of subsets $V_1, \ldots, V_m$ of V such that, for $1 \leq i \leq m$,

(a) $\Pi^{[V_i]}$ is an over-approximating abstraction of Π,

(b) the size of V_i is sufficiently small to perform reachability analysis in $\Pi^{[V_i]}$ by an (either explicit or symbolic) exhaustive search.

Let $h^{[V_i]}(s)$ be the optimal cost of achieving the abstract goal $G^{[V_i]}$ from the abstract state $s^{[V_i]}$. To obtain an admissible heuristic, if the set of abstract problems $\Pi^{[V_1]}, \ldots, \Pi^{[V_k]}$ satisfy certain requirements of disjointness (Felner, Korf, & Hanan 2004; Edelkamp 2001), the PDB heuristic can be set to $h(s) = \sum_{i=1}^{m} h^{[V_i]}(s)$. Otherwise, one can set $h(s) = \max_{i=1}^{m} h^{[V_i]}(s)$ (Holte *et al.* 2006).

The Achilles heel of the PDB heuristics is that each pattern (that is, each selected subset of variables V_i) is required to be *small* so that reachability analysis in $\Pi^{[V_i]}$ could be done by exhaustive search. In short, computing $h^{[V_i]}(s)$ in polynomial time requires satisfying $|V_i| = O(\log |V|)$. Note that this constraint implies an inherent scalability limitation of the PDB heuristics. As the problems of interest grow, limiting patterns to logarithmic dimensionality will unavoidably make them less and less informative with respect to the original problems, and this unless the domain forces its problem instances to consist of small, loosely-coupled subproblems that can be captured well by individual patterns.

However, pattern databases are not necessarily the only way to proceed. In principle, given a SAS$^+$ problem $\Pi = \langle V, A, I, G \rangle$, one can select a (relatively small) set of subsets $V_1, \ldots, V_m$ of V such that, for $1 \leq i \leq m$,

(a) $\Pi^{[V_i]}$ is an over-approximating abstraction of Π,

(b) *the reachability analysis in* $\Pi^{[V_i]}$ *is tractable (not necessarily due to the size of but) due to the specific structure of* $\Pi^{[V_i]}$.

What is important here is that the second requirement can be satisfied even if the size of each selected pattern V_i is $\Theta(|V|)$.

A priori, this generalization of the PDB idea to structural patterns is appealing as it allows using patterns of unlimited dimensionality. The pitfall, however, is that such structural patterns correspond to tractable fragments of cost-optimal planning, and the palette of such known fragments is extremely limited (Bäckström & Nebel 1995; Bylander 1994; Jonsson & Bäckström 1998; Jonsson 2007). Next, however, we show that this palette can still be extended, and such extensions may allow us to materialize the idea of structural patterns heuristics.

[1] An additional way to define pattern abstractions has been recently suggested by Hoffmann *et al.* (2006). However, the difference between the two approaches is not critical for our discussion here.

Causal Graph Structural Patterns

We begin with providing a syntactically slight, yet semantically very important for us generalization of the mechanism for constructing *disjoint* decompositions of planning problems along subsets of their state variables.

Definition 4 *Let* $\Pi = \langle V, A, I, G \rangle$ *be a* SAS$^+$ *problem, and let* $\mathcal{V} = \{V_1, \ldots, V_m\}$ *be a set of some subsets of* V. *A* **disjoint decomposition of** Π **over** $\mathcal{V}$ *is a set of* SAS$^+$ *problems* $\mathbf{\Pi} = \{\Pi_1, \ldots, \Pi_m\}$, *such that*

(1) For each $\Pi_i = \langle V_i, A_i, I_i, G_i \rangle$, *we have*

(a) $I_i = I^{[V_i]}$, $G_i = G^{[V_i]}$, *and*

(b) if $a^{[V_i]} \stackrel{\mathsf{def}}{=} \langle \mathsf{pre}(a)^{[V_i]}, \mathsf{eff}(a)^{[V_i]} \rangle$, *then*

$$A_i = \{a^{[V_i]} \mid a \in A \wedge \mathsf{eff}(a)^{[V_i]} \neq \emptyset\}.$$

(2) Each $a \in A$ *satisfies*

$$\mathcal{C}(a) \geq \sum_{i=1}^{m} \mathcal{C}_i(a^{[V_i]}). \qquad (1)$$

Definition 4 generalizes the idea of "all-or-nothing" action cost partitioning from the literature on PDBs disjoining to arbitrary action cost partitioning. In short, the original cost of each action is distributed this or another way among the "representatives" of that action in the subproblems, with Eq. 1 being the only constraint posed on this cost distribution.

Proposition 1 *For any* SAS$^+$ *problem* $\Pi = \langle V, A, I, G \rangle$, *any set of* V*'s subsets* $\mathcal{V} = \{V_1, \ldots, V_m\}$, *and any disjoint decomposition of* Π *over* $\mathcal{V}$, *we have* $h^*(I) \geq \sum_{i=1}^{m} h_i^*(I^{[V_i]})$.

Proof sketch: If $\rho = a_1 \cdot a_2 \cdot \ldots \cdot a_s$ is a cost-optimal plan for Π, then $h^*(I) = \mathcal{C}(\rho) = \sum_{i=1}^{s} \mathcal{C}(a_i)$. For $1 \leq i \leq m$, let $\rho^{[V_i]} = a_1^{[V_i]} \cdot a_2^{[V_i]} \cdot \ldots \cdot a_s^{[V_i]}$. From (1) in Definition 4, we have $\rho^{[V_i]}$ being a (not necessary optimal) plan for Π_i, and thus $h_i^*(I^{[V_i]}) \leq \mathcal{C}_i(\rho^{[V_i]}) = \sum_{j=1}^{s} \mathcal{C}_i(a_j^{[V_i]})$. From (2) in Definition 4 we then have $\sum_{i=1}^{m} h_i^*(I^{[V_i]}) \leq \sum_{i=1}^{m} \sum_{j=1}^{s} \mathcal{C}_i(a_j^{[V_i]}) = \sum_{j=1}^{s} \sum_{i=1}^{m} \mathcal{C}_i(a_j^{[V_i]}) \leq \sum_{j=1}^{s} \mathcal{C}(a_j) = h^*(I)$. ∎

Although disjoint decomposition over subsets of variables is rather powerful, it is too general for our purposes because it does not account for any structural requirements one may have for the abstract problems. For instance, focusing on the causal graph, when we project the problem onto subsets of its variables,

we leave all the causal-graph connections between the variables in each projected problem untouched. However, as here we aim at receiving abstract problems with causal graphs of *specific structure*, we should somehow project the original problem onto a subgraph (or a set of subgraphs) of the causal graph respecting such structural requirements. This leads us to introducing what we call *causal graph structural patterns.*

Definition 5 *Let $\Pi = \langle V, A, I, G\rangle$ be a* SAS$^+$ *problem, and $\mathcal{G} = (V_\mathcal{G}, E_\mathcal{G})$ be a subgraph of the causal graph CG(Π). A* **causal-graph structural pattern (CGSP)** *$\Pi_\mathcal{G} = \langle V_\mathcal{G}, A_\mathcal{G}, I_\mathcal{G}, G_\mathcal{G}\rangle$ is a* SAS$^+$ *problem defined as follows.*

1. *$I_\mathcal{G} = I^{[V_\mathcal{G}]}$, $G_\mathcal{G} = G^{[V_\mathcal{G}]}$,*
2. *$A_\mathcal{G} = \bigcup_{a\in A} A_\mathcal{G}(a)$, where each $A_\mathcal{G}(a) = \{a^1, \ldots, a^{l(a)}\}$, $l(a) \leq |\mathsf{eff}(a)|$, is a set of actions over $V_\mathcal{G}$ such that*
   (a) *for each $a^i \in A_\mathcal{G}(a)$, if $\mathsf{eff}(a^i)[v']$, and either $\mathsf{eff}(a^i)[v]$ or $\mathsf{pre}(a^i)[v]$ are specified, then $(v, v') \in \mathcal{G}$.*
   (b) *for each $(v, v') \in \mathcal{G}$, and each $a^i \in A_\mathcal{G}(a)$, if $\mathsf{eff}(a^i)[v']$ is specified, then either $\mathsf{eff}(a^i)[v]$ or $\mathsf{pre}(a^i)[v]$ is specified as well.*
   (c) *for each $s \in dom(V_\mathcal{G})$, if $\mathsf{pre}(a)^{[V_\mathcal{G}]} \subseteq s$, then the action sequence $\rho = a^1 \cdot a^2 \cdot \ldots \cdot a^{l(a)}$ is applicable in s, and if applying ρ in s results in $s' \in dom(V_\mathcal{G})$, then $s' \setminus s = \mathsf{eff}(a)$.*
   (d) *$\mathcal{C}(a) \geq \sum_{i=1}^{l(a)} \mathcal{C}_\mathcal{G}(a^i)$.*

Corollary 1 *For any* SAS$^+$ *problem $\Pi = \langle V, A, I, G\rangle$, and any subgraph $\mathcal{G} = (V_\mathcal{G}, E_\mathcal{G})$ of the causal graph CG(Π), we have CG($\Pi_\mathcal{G}$) = $\mathcal{G}$.*

Proposition 2 *For any* SAS$^+$ *problem $\Pi = \langle V, A, I, G\rangle$, and any subgraph $\mathcal{G} = (V_\mathcal{G}, E_\mathcal{G})$ of the causal graph CG(Π), at least one CGSP $\Pi_\mathcal{G}$ can be efficiently constructed from Π.*

The detailed proof of Proposition 2 is omitted here, but one possible construction of the action sets $A_\mathcal{G}(a)$ is as follows—if $\{v_1, \ldots, v_k\}$ is the subset of $V_\mathcal{G}$ affected by a, then $A_\mathcal{G}(a) = \{a^1, \ldots, a^k\}$ with

$$\mathsf{eff}(a^i)[v] = \begin{cases} \mathsf{eff}(a)[v], & v = v_i \\ \text{unspecified}, & \text{otherwise} \end{cases}$$

$$\mathsf{pre}(a^i)[v] = \begin{cases} \mathsf{eff}(a)[v], & v = v_j \wedge j < i \wedge (v_j, v_i) \in E_\mathcal{G} \\ \mathsf{pre}(a)[v], & v = v_j \wedge j > i \wedge (v_j, v_i) \in E_\mathcal{G} \\ \mathsf{pre}(a)[v], & v = v_i \\ \text{unspecified}, & \text{otherwise} \end{cases}$$

Now, given a SAS$^+$ problem Π and a subgraph $\mathcal{G}$ of CG(Π), if the structural pattern $\Pi_\mathcal{G}$ can be solved cost-optimally in polynomial time, we can use its solution as an admissible heuristic for Π. Moreover, given a set $\mathbf{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_m\}$ of subgraphs of the causal graph CG(Π), these heuristic estimates for structural patterns $\{\Pi_{\mathcal{G}_1}, \ldots, \Pi_{\mathcal{G}_m}\}$ are additive if holds a certain property given by Definition 6.

Definition 6 *Let $\Pi = \langle V, A, I, G\rangle$ be a* SAS$^+$ *problem, and $\mathbf{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_m\}$ be a set of subgraphs of the causal graph CG(Π). A* **disjoint CGSP decomposition of Π over $\mathbf{G}$** *is a set of CGSPs $\boldsymbol{\Pi} = \{\Pi_{\mathcal{G}_1}, \ldots, \Pi_{\mathcal{G}_m}\}$ such that each action $a \in A$ satisfies*

$$\mathcal{C}(a) \;\geq\; \sum_{i=1}^{m} \sum_{a' \in A_{\mathcal{G}_i}(a)} \mathcal{C}_{\mathcal{G}_i}(a'), \qquad (2)$$

Proposition 3 *For any* SAS$^+$ *problem $\Pi = \langle V, A, I, G\rangle$, any set of CG($\Pi$)'s subgraphs $\mathbf{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_m\}$, and any disjoint CGSP decomposition of Π over $\mathbf{G}$, we have $h^*(I) \geq \sum_{i=1}^m h^*_i(I_{\mathcal{G}_i})$.*

Proof sketch: If $\rho = a_1 \cdot a_2 \cdot \ldots \cdot a_s$ is a cost-optimal plan for Π, then $h^*(I) = \mathcal{C}(\rho) = \sum_{j=1}^s \mathcal{C}(a_j)$. By Definition 5, for $1 \leq i \leq m$, we have

$$\rho_{\mathcal{G}_i} = a_1^1 \cdot \ldots \cdot a_1^{l(a_1)} \cdot \ldots \cdot a_s^1 \cdot \ldots \cdot a_s^{l(a_s)}$$

being a (not necessary cost-optimal) plan for the CGSP $\Pi_{\mathcal{G}_i}$. Given that, we have $h^*_i(I_{\mathcal{G}_i}) \leq \mathcal{C}_{\mathcal{G}_i}(\rho_{\mathcal{G}_i}) = \sum_{j=1}^s \sum_{a' \in A_{\mathcal{G}_i}(a_j)} \mathcal{C}_{\mathcal{G}_i}(a')$. From Eq. 2, for $1 \leq j \leq s$, we have $\sum_{i=1}^m \sum_{a' \in A_{\mathcal{G}_i}(a_j)} \mathcal{C}_{\mathcal{G}_i}(a') \leq \mathcal{C}(a_j)$, and thus $\sum_{i=1}^m h^*_i(I_{\mathcal{G}_i}) \leq \sum_{i=1}^m \sum_{j=1}^s \sum_{a' \in A_{\mathcal{G}_i}(a_j)} \mathcal{C}_{\mathcal{G}_i}(a') = \sum_{j=1}^s \sum_{i=1}^m \sum_{a' \in A_{\mathcal{G}_i}(a_j)} \mathcal{C}_{\mathcal{G}_i}(a') \leq \sum_{j=1}^s \mathcal{C}(a_j) = h^*(I)$. ∎

Relying on Proposition 3, we can now decompose any given problem Π into a set of tractable CGSPs $\boldsymbol{\Pi} = \{\Pi_{\mathcal{G}_1}, \ldots, \Pi_{\mathcal{G}_m}\}$, solve all these CGSPs in polynomial time, and derive an admissible heuristic for Π. Note that (similarly to Def. 4) Def. 6 leaves the decision about the actual partition of the action costs rather open. In our discussion henceforth, we consider the (kind of "least-committing") *uniform* action-cost partitioning in which the action cost is *equally split* among its *non-redundant* projections in $\boldsymbol{\Pi}$.

Disjoint Fork-Decompositions

We now introduce certain decomposition of SAS$^+$ planning problems along their causal graphs. In itself, this decomposition does *not* lead to structural patterns abstractions, yet it provides an important building block on our way towards them.

Definition 7 *Let $\Pi = \langle V, A, I, G\rangle$ be a* SAS$^+$ *problem. The* **fork-decomposition**

$$\boldsymbol{\Pi} = \{\Pi_{\mathcal{G}_v^\mathsf{f}}, \Pi_{\mathcal{G}_v^\mathsf{if}}\}_{v \in V}$$

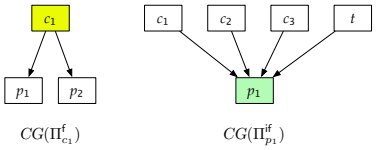


Figure 3: Causal graphs of a fork and an inverted fork structural patterns of the running example.

is a disjoint CGSP decomposition of Π *over subgraphs* $\mathbf{G} = \{\mathcal{G}^{\mathsf{f}}_v, \mathcal{G}^{\mathsf{if}}_v\}_{v \in V}$ *where, for* $v \in V$*,*

$$V_{\mathcal{G}^{\mathsf{f}}_v} = \{v\} \cup \mathsf{succ}(v), \quad E_{\mathcal{G}^{\mathsf{f}}_v} = \bigcup_{u \in \mathsf{succ}(v)} \{(v,u)\}$$

$$V_{\mathcal{G}^{\mathsf{if}}_v} = \{v\} \cup \mathsf{pred}(v), \quad E_{\mathcal{G}^{\mathsf{if}}_v} = \bigcup_{u \in \mathsf{pred}(v)} \{(u,v)\}$$

Illustrating Definition 7, let us consider the (uniform) fork-decomposition of the problem Π from our running example, assuming all the actions in Π have the same unit cost. After eliminating from $\mathbf{G}$ all the singletons[2], we get $\mathbf{G} = \{\mathcal{G}^{\mathsf{f}}_{c_1}, \mathcal{G}^{\mathsf{f}}_{c_2}, \mathcal{G}^{\mathsf{f}}_{c_3}, \mathcal{G}^{\mathsf{f}}_{t}, \mathcal{G}^{\mathsf{if}}_{p_1}, \mathcal{G}^{\mathsf{if}}_{p_2}\}$. Considering the action sets of the problems in $\mathbf{\Pi}$, each original driving action is present (by its projections) in some three problems in $\mathbf{\Pi}$, while each load/unload action is present in some five such problems. For instance, the projections of the action "drive-c_1-from-A-to-D" are present in $\{\Pi^{\mathsf{f}}_{c_1}, \Pi^{\mathsf{if}}_{p_1}, \Pi^{\mathsf{if}}_{p_2}\}$, and the projections of the action "load-p_1-into-c_1-at-A" are present in $\{\Pi^{\mathsf{f}}_{c_1}, \Pi^{\mathsf{f}}_{c_2}, \Pi^{\mathsf{f}}_{c_3}, \Pi^{\mathsf{f}}_{t}, \Pi^{\mathsf{if}}_{p_1}\}$. Since our fork-decomposition is uniform, the cost of each driving (load/unload) action projection is set to $1/3$ (respectively, to $1/5$).

From Proposition 3 we have that the sum of costs of solving the problems $\mathbf{\Pi}$, that is,

$$h_{\text{⩚}} = \sum_{v \in V} \left(h^*_{\Pi^{\mathsf{f}}_v} + h^*_{\Pi^{\mathsf{if}}_v} \right), \tag{3}$$

is an admissible estimate of h^*. The question now is how good this estimate is. The optimal cost of solving our problem is 19, and

$$\begin{aligned} h_{\text{⩚}} &= h^*_{\Pi^{\mathsf{f}}_{c_1}} + h^*_{\Pi^{\mathsf{f}}_{c_2}} + h^*_{\Pi^{\mathsf{f}}_{c_3}} + h^*_{\Pi^{\mathsf{f}}_{t}} + h^*_{\Pi^{\mathsf{if}}_{p_1}} + h^*_{\Pi^{\mathsf{f}}_{p_2}} = \\ &= \tfrac{8}{5} + \tfrac{8}{5} + (\tfrac{8}{5} + \tfrac{6}{3}) + (\tfrac{8}{5} + \tfrac{2}{3}) + (\tfrac{6}{5} + \tfrac{9}{3}) + (\tfrac{2}{5} + \tfrac{4}{3}) = \\ &= 15 \end{aligned} \tag{4}$$

Taking as a basis for comparison the seminal $h_{\max}$ and h^2 heuristics (Bonet & Geffner 2001; Haslum & Geffner 2000), we have $h_{\max} = 8$ and $h^2 = 13$. Hence, it appears that using the additive CGSP heuristic $h_{\text{⩚}}$ is at least promising.

[2]If the causal graph $CG(\Pi)$ is connected and $n > 1$, then this elimination is not lossy.

Unfortunately, despite the seeming simplicity of the problems in $\mathbf{\Pi}$, turns out that fork-decompositions by themselves do not fit the requirements of the structural patterns framework. The causal graphs of $\{\Pi^{\mathsf{f}}_{c_1}, \Pi^{\mathsf{f}}_{c_2}, \Pi^{\mathsf{f}}_{c_3}, \Pi^{\mathsf{f}}_{t}\}$ and $\{\Pi^{\mathsf{if}}_{p_1}, \Pi^{\mathsf{if}}_{p_2}\}$ form directed forks and inverted forks, respectively (see Figure 3), and, in general, the number of variables in each such problem is $\Theta(n)$. Unfortunately for us, Domshlak and Dinitz (2001) show that even non-optimal planning for SAS$^+$ problems with fork and inverted fork causal graphs is NP-complete. Moreover, even if the domain-transition graphs of all the projections are strongly connected, optimal planning for forks and inverted forks remain NP-hard (see Helmert (2003) and (2004) for the respective results). However, in the next section we show that this is not the end of the story on fork-decompositions.

Meeting Structural and Domain Abstractions

While hardness of optimal planning for problems with fork and inverted fork causal graphs put a shadow on relevance of fork-decompositions, closer look at the proofs of these hardness results of Domshlak and Dinitz (2001) and Helmert (2003; 2004) reveals that these proofs in particular rely on root variables having large domains. It turns out that this dependence is not incidental, and Propositions 4 and 5 below present some significant islands of tractability within these structural fragments of SAS$^+$.

Proposition 4 *Given a* SAS$^+$ *problem* $\Pi = \langle V, A, I, G\rangle$ *inducing a fork causal graph with a root* $r \in V$*, if (i)* $|dom(r)| = 2$*, or (ii) for all* $v \in V$*, we have* $|dom(v)| = O(1)$*, then finding a cost-optimal plan for* Π *is poly-time.*

Proof sketch: First, if $|dom(r)| = 2$, let $dom(r) = \{0, 1\}$, where $I[r] = 0$. Let $\sigma(r)$ be a 0/1 sequence of length $1 + d$, where $d = \max_{u \in \mathsf{succ}(r)} |dom(u)|$, and, for $1 \le i \le |\sigma(r)|$,

$$\sigma(r)[i] = \begin{cases} 0, & \text{i is odd,} \\ 1, & \text{i is even} \end{cases}$$

Finally, let $\trianglerighteq^*[\sigma(r)]$ be the set of all non-empty prefixes of $\sigma(r)$ if $G[r]$ is unspecified, and the set of all non-empty prefixes of $\sigma(r)$ ending with $G[r]$ otherwise.

(1) For each $u \in \mathsf{succ}(r)$, let DTG^0_u and DTG^1_u be the subgraphs of $DTG(u, \Pi)$ obtained by removing from the latter all the arcs labeled with 1 and 0, respectively. For each $u \in \mathsf{succ}(r)$, and each $x, y \in dom(u)$, compute the shortest (that is, cost-minimal) paths from x to y in DTG^0_u and DTG^1_u.

(2) For each $\sigma \in \trianglerighteq^*[\sigma(r)]$, and each $u \in \mathsf{succ}(r)$, build a layered digraph $\mathcal{L}_u(\sigma)$ with $|\sigma| + 1$ layers

$L_0, \ldots, L_{|\sigma|}$, where L_0 consists of only $I[u]$, and for $1 \leq i \leq |\sigma|$, L_i consists of all nodes reachable from the nodes L_{i-1} in DTG^0_u if i is odd, and in DTG^1_u if i is even. For each $x \in L_{i-1}, y \in L_i$, $\mathcal{L}_u(\sigma)$ contains an arc (x, y) weighted with the cost of the cost-minimal path from x to y in DTG^0_u if i is odd, and in DTG^1_u if i is even.

(3) For each $\sigma \in \trianglerighteq^*[\sigma(r)]$, let $|\sigma| = s$. A candidate plan ρ_σ for Π is constructed as follows.

  (a) For each $u \in \mathsf{succ}(r)$, find a cost-minimal path from $I[u]$ to $G[u]$ in $\mathcal{L}_u(\sigma)$. Note that the i-th edge on this path (taking us from $x \in L_{i-1}$ to $y \in L_i$) corresponds to the cost-minimal path from x to y in either DTG^0_u or DTG^1_u. Let us denote this path from x to y by S^i_u.

  (b) Set $\rho_\sigma = S^1 \cdot a_{\sigma[2]} \cdot S^2 \cdot \ldots \cdot a_{\sigma[s]} \cdot S^s$, where sequence S^i is obtained by an arbitrary merge of the sequences $\{S^i_u\}_{u \in \mathsf{succ}(r)}$, and a_α is the action that changes the value of r to α.

(4) Set and return $\rho = \mathrm{argmin}_{\sigma \in \trianglerighteq^*[\sigma(r)]} \mathcal{C}(\rho_\sigma)$.

It is not hard to verify that the complexity of the above procedure is polynomial in the description size of Π, and that the constructed plan ρ is cost-optimal.

We omit here the proof of the second case, and only note that a rather simple analysis upper-bounds the time complexity for this case by $\Theta(d^{d^{d+2}+2d+2}) = O(1)$. While for practice this "constant" bound is rather sarcastic, here we have not tried to be complexity-optimal either. Finding more realistic bounds for this concrete problem is definitely of interest.

■

Proposition 5 *Given a 1-dependent* SAS$^+$ *problem* $\Pi = \langle V, A, I, G\rangle$ *inducing an inverted fork causal graph with a root* $r \in V$*, if* $|dom(r)| = O(1)$*, then finding a cost-optimal plan for* Π *is poly-time.*

Proof sketch: Let $|dom(r)| = d$. A naive algorithm that finds a cost-optimal plan for Π in time $\Theta(d^{d+1} + |\Pi|^3) = \Theta(|\Pi|^3)$ is as follows.

(1) Create all $\Theta(d^d)$ cycle-free paths from $I[r]$ to $G[r]$ in $DTG(r, \Pi)$.

(2) For each $u \in \mathsf{pred}(r)$, and each $x, y \in dom(u)$, compute the cost-minimal path from x to y in $DTG(u, \Pi)$.

(3) For each path in $DTG(r, \Pi)$ generated in step (1), construct a plan for Π based on that path for r, and the shortest paths computed in (2).

(4) Take minimal cost plan from (3).

The time complexity of this algorithm is $\Theta(|\Pi|^3)$, and it finds a optimal plan, if such exist. The latter can be shown as follows. For each cost-optimal plan ρ, it is easy to verify that $\rho{\downarrow_r}$ is one of the paths generated in step (1). For each $u \in \mathsf{pred}(r)$, let S_u denote the sequence of values from $dom(u)$ that is required by the prevail conditions of the actions along $\rho{\downarrow_r}$. If so, for each $u \in \mathsf{pred}(r)$, $\rho{\downarrow_u}$ corresponds to a path from $I[u]$ to $G[u]$ in $DTG(u, \Pi)$, traversing the values (= nodes) in S_u in the order required by S_u. And a plan for Π generated in (3) consists of minimal such paths for all $u \in \mathsf{pred}(r)$. Therefore, at least one of the plans generated in (3) will be cost-optimal for Π. ■

Propositions 4 and 5 allow us to meet between the fork-decompositions and tractable structural patterns at least for 1-dependent planning domains such as Logistics[3]. The basic idea is to further abstract each CGSP in fork-decomposition of Π by abstracting domains of its variables to meet the requirements of the tractable fragments.

Definition 8 *Let* $\Pi = \langle V, A, I, G\rangle$ *be a* SAS$^+$ *problem,* $v \in V$*, and let* $\Phi = \{\phi_1, \ldots, \phi_k\}$ *be a set of mappings from* $dom(v)$ *to some sets* $\Gamma_1, \ldots, \Gamma_k$*. A* **disjoint domain decomposition of** Π **over** Φ *is a set of* SAS$^+$ *problems* $\mathbf{\Pi} = \{\Pi_1, \ldots, \Pi_k\}$*, such that*

(1) For each $\Pi_i = \langle V_i, A_i, I_i, G_i\rangle$*, we have*[4]

  (a) $I_i = \phi_i(I)$*,* $G_i = \phi_i(G)$*, and*

  (b) if $\phi_i(a) \stackrel{\mathsf{def}}{=} \langle \phi_i(\mathsf{pre}(a)), \phi_i(\mathsf{eff}(a))\rangle$*, then*

$$A_i = \{\phi_i(a) \mid a \in A \wedge \phi_i(\mathsf{eff}(a)) \not\subseteq \phi_i(\mathsf{pre}(a))\}.$$

(2) Each $a \in A$ *satisfies*

$$\mathcal{C}(a) \geq \sum_{i=1}^{k} \mathcal{C}_i\left(\phi_i(a)\right). \tag{5}$$

Proposition 6 *For any* SAS$^+$ *problem* $\Pi = \langle V, A, I, G\rangle$*, any* $v \in V$*, any set of domain abstractions* $\Phi = \{\phi_1, \ldots, \phi_k\}$*, and any disjoint domain decomposition of* Π *over* Φ*, we have* $h^*(I) \geq \sum_{i=1}^{k} h^*_i(\phi_i(I))$.

Targeting tractability of the causal graph structural patterns, we connect between fork-decompositions and domain decompositions as in Definition 8. Given a fork-decomposition $\mathbf{\Pi} = \{\Pi^{\mathsf{f}}_v, \Pi^{\mathsf{if}}_v\}_{v \in V}$ of Π,

- For each $\Pi^{\mathsf{f}}_v \in \mathbf{\Pi}$,

(a) Associate the root r of $CG(\Pi^{\mathsf{f}}_v)$ with mappings $\Phi_v = \{\phi_{v,1}, \ldots, \phi_{v,k_v}\}$, $k_v = O(poly(|\Pi|))$, and all $\phi_{v,i} : dom(r) \to \{0, 1\}$.

[3] If the problem of interest falls outside this problem class, then it should (and could) be first abstracted to a problem within that class. The palette of concrete choices for such an abstraction is rather wide, and currently we investigate their relative pros and cons.

[4] For a partial assignment S on V, $\phi_i(S)$ denotes the abstracted partial assignment obtained from S by replacing $S[v]$ (if any) with $\phi_i(S[v])$.

(b) Disjointly decompose Π_v^{f} into $\mathbf{\Pi}_v^{\mathsf{f}} = \{\Pi_{v,i}^{\mathsf{f}}\}_{i=1}^{k_v}$ over Φ_v.

- For each $\Pi_v^{\mathsf{if}} \in \mathbf{\Pi}$,

(a) Associate the root r of $CG(\Pi_v^{\mathsf{if}})$ with mappings $\Phi'_v = \{\phi'_{v,1}, \ldots, \phi'_{v,k'_v}\}$, $k'_v = O(poly(|\Pi|))$, all $\phi'_{v,i} : dom(r) \to \{0, 1, \ldots, b_{v,i}\}$, $b_{v,i} = O(1)$.

(b) Disjointly decompose Π_v^{if} into $\mathbf{\Pi}_v^{\mathsf{if}} = \{\Pi_{v,i}^{\mathsf{if}}\}_{i=1}^{k'_v}$ over Φ'_v.

For 1-dependent problems Π, from Proposition 3 and 6 we then have

$$h_{\mathbin{/\!/\!/}} = \sum_{v \in V} \left(\sum_{i=1}^{k_v} h^*_{\Pi^{\mathsf{f}}_{v,i}} + \sum_{i=1}^{k'_v} h^*_{\Pi^{\mathsf{if}}_{v,i}} \right), \qquad (6)$$

being an admissible estimate of h^* for Π, and from Propositions 4-5 we have that $h_{\mathbin{/\!/\!/}}$ is also computable in polynomial time. The question is, however, how further abstracting our fork-projections affects the informativeness of the heuristic estimate. As we show later, the answer is somewhat surprising.

Let us again use our running example to illustrate the mixture of structural and domain projections as outlined above. To begin with an extreme setting of domain abstractions, first, let the domain abstractions for roots of both forks and inverted forks be to binary domains. Among multiple options for choosing the mapping sets $\{\Phi_v\}$ and $\{\Phi'_v\}$, here we use a simple choice of distinguishing between different values of each variable v on the basis of their distance from $I[v]$ in $DTG(v, \Pi)$. Specifically, for each $v \in V$, we set $\Phi_v = \Phi'_v$, and, for each value $\vartheta \in dom(v)$,

$$\phi_{v,i}(\vartheta) = \phi'_{v,i}(\vartheta) = \begin{cases} 0, & d(I[v], \vartheta) < i \\ 1, & \text{otherwise} \end{cases}$$

For instance, the problem $\Pi^{\mathsf{f}}_{c_1}$ is decomposed (see Figure 2b) into two problems, $\Pi^{\mathsf{f}}_{c_1,1}$ and $\Pi^{\mathsf{f}}_{c_1,2}$, with the 0/1 abstract domain of c_1 corresponding to the partitions $\{A\}/\{B, C, D\}$ and $\{A, D\}/\{B, C\}$ of $dom(c_1)$, respectively. The (interesting for certain reasons below) problem $\Pi^{\mathsf{if}}_{p_1}$ is decomposed (see Figure 2c) into six problems $\Pi^{\mathsf{f}}_{p_1,1}, \ldots, \Pi^{\mathsf{f}}_{p_1,6}$ along the abstractions of $dom(p_1)$ depicted in Figure 4.

Now, given the decomposition of Π over forks and $\{\Phi_v, \Phi'_v\}_{v \in V}$ as above, consider the problem $\Pi_{p_1,1}$, obtained from projecting Π onto the inverted fork of p_1 and then abstracting $dom(p_1)$ using

$$\phi_{p_1,1}(\vartheta) = \begin{cases} 0, & \vartheta \in \{C\} \\ 1, & \vartheta \in \{A, B, D, E, F, G, c_1, c_2, c_3, t\} \end{cases}$$

It is not hard to verify that, from the original actions affecting p_1, in $\Pi_{p_1,1}$ we are left only with actions conditioned by c_1 and c_2. If so, then no information is lost[5] if we

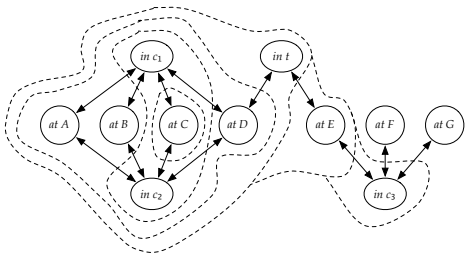


Figure 4: Binary-range domain abstractions for Π_{p_1}. The values within and outside each dashed curve are mapped to 0 and 1, respectively.

1. remove from $\Pi_{p_1,1}$ both the variables c_3 and t, and the actions changing (only) these variables, and
2. redistribute the (fractioned) cost of the removed actions between all other representatives of their originals in Π.

The latter revision of the action cost partitioning is obtained directly by replacing the cost-partitioning steps corresponding to Eqs. 2 and 5 by a single, joint action cost partitioning applied over the final projections $\bigcup_{v \in V} (\mathbf{\Pi}_v^{\mathsf{f}} \cup \mathbf{\Pi}_v^{\mathsf{if}})$ and satisfying

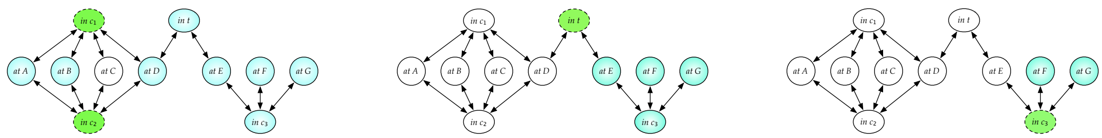
$$\mathcal{C}(a) \geq \sum_{v \in V} \left(\sum_{i=1}^{k_v} \sum_{a' \in A_{\mathcal{G}^{\mathsf{f}}_v}(a)} \mathcal{C}^{\mathsf{f}}_{v,i}(\phi_{v,i}(a')) \; + \; \sum_{i=1}^{k'_v} \sum_{a' \in A_{\mathcal{G}^{\mathsf{if}}_v}(a)} \mathcal{C}^{\mathsf{if}}_{v,i}(\phi'_{v,i}(a')) \right) \qquad (7)$$

Overall, computing $h_{\mathbin{/\!/\!/}}$ as in Eq. 6 under these "all binary range domain abstractions" provides us with $h_{\mathbin{/\!/\!/}} = 12\frac{7}{15}$, and knowing that the original costs are all integers we can safely adjust it to $h_{\mathbin{/\!/\!/}} = 13$. Hence, even under most severe domain abstractions as above, $h_{\mathbin{/\!/\!/}}$ on our example problem does not fall from h^2.

Let us now slightly relax our domain abstractions for the roots of the inverted forks to be to the ternary range $\{0, 1, 2\}$. While mappings $\{\Phi_v\}$ stay as before, $\{\Phi'_v\}$ is set to

$$\forall \vartheta \in dom(v): \quad \phi'_{v,i} = \begin{cases} 0, d(I[v], \vartheta) < 2i - 1 \\ 1, d(I[v], \vartheta) = 2i - 1 \\ 2, d(I[v], \vartheta) > 2i - 1 \end{cases}$$

[5] One of the reasons why no information is lost is the fact that we keep either fork or inverted fork for each variable of Π. In any event, here we omit further formal justifications of this optimization step.



$dom(p_1)$ in $\Pi^{\mathsf{f}}_{p_1,1}$ $dom(p_1)$ in $\Pi^{\mathsf{f}}_{p_1,2}$ $dom(p_1)$ in $\Pi^{\mathsf{f}}_{p_1,3}$

Figure 5: Ternary-range domain abstractions for Π_{p_1}; values that mapped to the same abstract value are shown as nodes with the same color and borderline.

For instance, the problem $\Pi^{\mathsf{if}}_{p_1}$ is decomposed now into three problems $\Pi^{\mathsf{f}}_{p_1,1}, \ldots, \Pi^{\mathsf{f}}_{p_1,3}$ along the abstractions of $dom(p_1)$ depicted in Figure 5.

Applying now the same computation of $h_{\Lsh}$ as in Eq. 6 over our new set of domain abstractions gives $h_{\Lsh} = 15\frac{1}{2}$, which, again, can be safely adjusted to $h_{\Lsh} = 16$. Note that this value is *higher* than $h_{\Lsh} = 15$ obtained using the fork-decomposition alone (as in Eq. 3). At first view, this outcome may seem counter-intuitive as the domain abstractions are applied *over* the fork-decomposition. The explanation, however, is that (as shown above) the domain abstractions for the roots of inverted forks may create independence between the roots and their preconditioning variables. And exploiting such domain-abstraction specific independence relations leads to more targeted action cost partitioning as in Eq. 7.

Discussion

We presented a structural-patterns generalization of PDB abstractions that is based on projecting the problem in hand to provably tractable fragments of optimal planning. The key motivation behind this generalization is to alleviate the requirement for the projections to be of a low dimensionality. To show the practical potential of the basic idea, we introduced and looked into a concrete structural patterns abstraction based on decomposing the problem into a set of fork and inverted fork components of its causal graph, combined with abstracting the domains of certain variables within these individual components.

The basic principles of the structural patterns framework motivate further research in numerous directions, and in particular, in (1) discovering new islands of tractability of optimal planning, and (2) translating and/or abstracting the general planning problems into such islands. In our ongoing work we aim at pursuing both these directions by "mining" the tractable fragments of optimal SAS^+ planning, and, in particular, of planning with unary-effect actions (Katz & Domshlak 2007), and by performing formal and empirical analysis of alternative schemes for abstracting general SAS^+ problems to meet the specification of such islands of tractability.

References

Bäckström, C., and Nebel, B. 1995. Complexity results for SAS^+ planning. *Comp. Intell.* 11(4):625–655.

Bonet, B., and Geffner, H. 2001. Planning as heuristic search. *AIJ* 129(1–2):5–33.

Bylander, T. 1994. The computational complexity of propositional STRIPS planning. *AIJ* 69(1-2):165–204.

Culberson, J., and Schaeffer, J. 1998. Pattern databases. *Comp. Intell.* 14(4):318–334.

Domshlak, C., and Dinitz, Y. 2001. Multi-agent off-line coordination: Structure and complexity. In *ECP*, 277–288.

Edelkamp, S. 2001. Planning with pattern databases. In *ECP*, 13–34.

Felner, A.; Korf, R. E.; and Hanan, S. 2004. Additive pattern database heuristics. *JAIR* 22:279–318.

Haslum, P., and Geffner, H. 2000. Admissible heuristics for optimal planning. In *ICAPS*, 140–149.

Haslum, P.; Bonet, B.; and Geffner, H. 2005. New admissible heuristics for domain-independent planning. In *AAAI*, 1163–1168.

Helmert, M. 2003. Complexity results for standard benchmark domains in planning. *AIJ* 146(2):219–262.

Helmert, M. 2004. A planning heuristic based on causal graph analysis. In *ICAPS*, 161–170.

Helmert, M. 2006. The Fast Downward planning system. *JAIR* 26:191–246.

Hoffmann, J.; Sabharwal, A.; and Domshlak, C. 2006. Friends or foes? An AI planning perspective on abstraction and search. In *ICAPS*, 294–303.

Holte, R. C.; Felner, A.; Newton, J.; Meshulam, R.; and Furcy, D. 2006. Maximizing over multiple pattern databases speeds up heuristic search. *AIJ* 170(16-17):1123–1136.

Jonsson, P., and Bäckström, C. 1998. State-variable planning under structural restrictions: Algorithms and complexity. *AIJ* 100(1–2):125–176.

Jonsson, A. 2007. The role of macros in tractable planning over causal graphs. In *IJCAI*, 1936–1941.

Katz, M., and Domshlak, C. 2007. Structural patterns of tractable sequentially-optimal planning. In *ICAPS*.

Pearl, J. 1984. *Heuristics — Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley.