Structural Patterns Heuristics via Fork Decomposition

Michael Katz and **Carmel Domshlak***
Faculty of Industrial Engineering & Management
Technion, Israel

Abstract

We consider a generalization of the PDB homomorphism abstractions to what is called "structural patterns". The basic idea is in abstracting the problem in hand into provably tractable fragments of optimal planning, alleviating by that the constraint of PDBs to use projections of only low dimensionality. We introduce a general framework for additive structural patterns based on decomposing the problem along its causal graph, suggest a concrete non-parametric instance of this framework called fork-decomposition, and formally show that the admissible heuristics induced by the latter abstractions provide state-of-the-art worst-case informativeness guarantees on several standard domains.

Introduction

The difference between various algorithms for optimal planning as heuristic search is mainly in the admissible heuristic functions they define and use. Very often 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 present in the specification of the problem, and the desire is to obtain a *tractable* (that is, solvable in polynomial time), yet *informative* abstract problem. The main question here is thus: What constraints should we relax to obtain such an effective over-approximating abstraction?

The abstract state space induced by a *homomorphism abstraction* is obtained by a systematic contracting groups of original problem states into abstract states. Most typically, such state-gluing is obtained by projecting the original problem onto a subset of its parameters, eliminating 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 *et al.* 2007), 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 a small (up to logarithmic) dimensionality so that reachability analysis in that space could be done by exhaustive search. This constraint implies a scalability limitation of the PDB heuristics—as the problems of interest grow, limiting patterns to logarithmic dimensionality might make them less and less informative with respect to the original problems.

In this paper we consider a generalization of the PDB abstractions to what is called *structural patterns*. As it was recently suggested by Katz and Domshlak (2007), the basic idea behind structural patterns is simple and intuitive, and it corresponds to abstracting the original problem to provably tractable fragments of optimal planning. The key point is that, at least theoretically, moving to structural patterns alleviates the requirement for the abstractions to be of a low dimensionality. The contribution of this paper is precisely in showing that structural patterns are far from being of theoretical interest only. Specifically, we

- Specify *causal-graph structural patterns (CGSPs)*, a general framework for additive structural patterns that is based on decomposing the problem in hand along its causal graph.
- Introduce a concrete family of CGSPs, called *fork decompositions*, that is based on two novel fragments of tractable cost-optimal planning.
- Following the type of analysis suggested by Helmert and Mattmüller (2008), study the asymptotic performance ratio of the fork-decomposition admissible heuristics, and show that their worst-case informativeness on selected domains more than favorably competes with this of (even parametric) state-of-the-art admissible heuristics.

From PDBs to Structural Patterns

Problems of *classical planning* correspond to reachability analysis in state models with deterministic actions and complete information; here we consider state models captured by the SAS$^+$ formalism (Bäckström & Nebel 1995).

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*

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

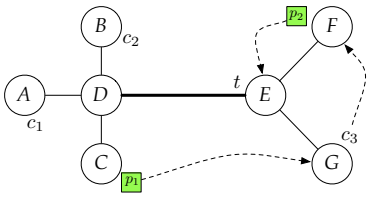


Figure 1: Logistics-style example adapted from Helmert (2006a). Deliver p_1 from C to G, and p_2 from F to E using the cars c_1, c_2, c_3 and truck t, and making sure that c_3 ends up at F. The cars may only use city roads (thin edges), the truck may only use the highway (thick edge).

complete assignment, and the goal *G is a partial assignment to V.*

- *A 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-style example of Helmert (2006a). This example is depicted in Figure 1, and in SAS$^+$ it has

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

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

Pattern Database Heuristics

Given a problem $\Pi = \langle V, A, I, G \rangle$, for any partial assignment x to V, and any $V' \subseteq V$, by $x^{[V']}$ we refer to the projection of x onto V'. Considering homomorphism abstractions of Π, each variable subset $V' \subseteq V$ defines an over-approximating *pattern abstraction* $\Pi^{[V']} = \langle V', A^{[V']}, I^{[V']}, G^{[V']} \rangle$ that is obtained by projecting the initial state, the goal, and all the actions' preconditions and effects onto V' (Edelkamp 2001). 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 $1 \leq i \leq m$, the size of V_i is sufficiently small to perform exhaustive-search reachability analysis in $\Pi^{[V_i]}$. Let $h^{[V_i]}(s)$ be the optimal cost of achieving the abstract goal $G^{[V_i]}$ from the abstract state $s^{[V_i]}$. Since each $\Pi^{[V_i]}$ is an over-approximating abstraction of Π, each $h^{[V_i]}(\cdot)$ is an admissible estimate of the true cost $h^*(\cdot)$. Given that, if the set of abstract problems $\Pi^{[V_1]}, \ldots, \Pi^{[V_m]}$ satisfy certain requirements of additivity (Felner, Korf, & Hanan 2004; Edelkamp 2001), then PDB heuristic can be set to $\sum_{i=1}^m h^{[V_i]}(s)$, and otherwise only to $\max_{i=1}^m h^{[V_i]}(s)$.

We now provide a syntactically slight, yet quite powerful generalization of the standard mechanism for constructing *additive* decompositions of planning problems along subsets of their state variables (Felner, Korf, & Hanan 2004; Edelkamp 2001).

Definition 2 *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. An* **additive 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) For each $a \in A$ holds

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

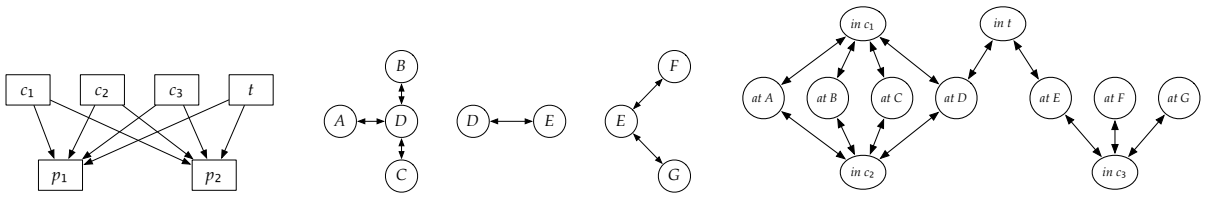
Definition 2 generalizes the idea of "all-or-nothing" action-cost partition from the literature on additive PDBs to arbitrary action-cost partitions—the original cost of each action is partitioned this or another way among the "representatives" of that action in the abstract problems, with Eq. 1 being the only constraint posed on this action-cost partition.

Proposition 1 *For any* SAS$^+$ *problem Π over variables V, any set of V's subsets $\mathcal{V} = \{V_1, \ldots, V_m\}$, and any additive decomposition of Π over $\mathcal{V}$, we have $h^*(s) \geq \sum_{i=1}^m h_i^*(s^{[V_i]})$ for all states s of Π.*[1]

Structural Patterns: Basic Idea

PDB heuristics and their enhancements are successfully exploited these days in the planning research (Edelkamp 2001; Haslum, Bonet, & Geffner 2005; Haslum *et al.* 2007). However, the well-known Achilles heel of the PDB heuristics is that each pattern (that is, each selected variable subset 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 $|V_i| = O(\log |V|)$ if $|dom(v)| = O(1)$ for each $v \in V_i$, and satisfying $|V_i| = O(1)$, otherwise. In both cases, 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

[1] Due to the lack of space, the proofs are given in the full TR.



(a) (b) (c)

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.

instances to consist of small, loosely-coupled subproblems that can be captured well by individual patterns (Helmert & Mattmüller 2008).

However, as recently observed by Katz and Domshlak (2007), pattern databases are not necessarily the only way to proceed with homomorphism abstractions. In principle, given a SAS$^+$ problem $\Pi = \langle V, A, I, G\rangle$, one can select a set of subsets $V_1, \ldots, V_m$ of V such that, for $1 \leq i \leq m$, 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 this requirement can be satisfied even if the size of each selected pattern V_i is $\Theta(|V|)$. In any event, having specified the abstract problems $\Pi^{[V_1]}, \ldots, \Pi^{[V_m]}$ as above, the heuristic estimate is then formulated similarly to the PDB heuristics: If the set of abstract problems $\Pi^{[V_1]}, \ldots, \Pi^{[V_m]}$ satisfy additivity, then we set $h(s) = \sum_{i=1}^{m} h^{[V_i]}(s)$, otherwise we set $h(s) = \max_{i=1}^{m} h^{[V_i]}(s)$.

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; Katz & Domshlak 2007). Next, however, we show that this palette can still be extended, and this in the direction allowing us to materialize the idea of structural patterns heuristics.

Causal Graph Structural Patterns

The key role in what follows plays the causal graph structure that has already been exploited in complexity analysis of classical planning. 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 $CG(\Pi)$ 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. 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 $CG(\Pi)$.

Though less heavily, we also use here the structure of the problem's domain transition graphs (Bäckström & Nebel 1995). The **domain transition graph** $DTG(v, \Pi)$ of v in Π is an arc-labeled digraph over the nodes $dom(v)$ such that an arc (ϑ, ϑ') belongs to $DTG(v, \Pi)$ 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. We now proceed with exploiting this structure of the problems in devising structural pattern heuristics.

Though additive 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 abstract problem untouched. In contrast, targeting tractable fragments of cost-optimal planning, here we aim at receiving abstract problems with causal graphs of *specific structure*. This leads us to introducing what we call *causal graph structural patterns;* the basic idea here is to abstract the given problem Π along a subgraph of its causal graph, obtaining an over-approximating abstraction that preserves the effects of the Π's actions to the largest extent possible.

Definition 3 *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 $CG(\Pi)$. 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 $(v, v') \in E_\mathcal{G}$.*
   - *(b) for each $(v, v') \in E_\mathcal{G}$, s.t. $\mathsf{eff}(a)[v']$ is specified and either $\mathsf{eff}(a)[v]$ or $\mathsf{pre}(a)[v]$ is specified, 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 $s' \setminus s = \mathsf{eff}(a)^{[V_\mathcal{G}]}$.*
   - *(d) $\mathcal{C}(a) \geq \sum_{i=1}^{l(a)} \mathcal{C}_\mathcal{G}(a^i)$.*

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 $CG(\Pi)$, a CGSP $\Pi_\mathcal{G}$ can always be (efficiently) constructed from Π, and it is ensured that $CG(\Pi_\mathcal{G}) = \mathcal{G}$. The latter is directly enforced by the construction constraints in Definition 3. To see the

former, 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

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

$$\mathsf{pre}(a^i)[v] = \begin{cases} \mathsf{eff}(a)[v], & v = v_j \wedge j < i \wedge (v_j, v_i) \in E_{\mathcal{G}} \\ \mathsf{pre}(a)[v], & v = v_j \wedge j > i \wedge (v_j, v_i) \in E_{\mathcal{G}} \\ \mathsf{pre}(a)[v], & v = v_i \\ \mathsf{pre}(a)[v], & v \notin \{v_1, \ldots, v_k\} \wedge (v, v_i) \in E_{\mathcal{G}} \\ \text{unspecified}, & \text{otherwise} \end{cases}$$

Now, given a SAS$^+$ problem Π and a subgraph $\mathcal{G}$ of $CG(\Pi)$, 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 $CG(\Pi)$, these heuristic estimates induced by the structural patterns $\{\Pi_{\mathcal{G}_1}, \ldots, \Pi_{\mathcal{G}_m}\}$ are additive if holds a certain property given by Definition 4.

Definition 4 *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 $CG(\Pi)$. An* **additive CGSP decomposition of Π over $\mathbf{G}$** *is a set of CGSPs $\mathbf{\Pi} = \{\Pi_{\mathcal{G}_1}, \ldots, \Pi_{\mathcal{G}_m}\}$ such that, for each action $a \in A$, holds*

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

Proposition 2 (Admissibility) *For any SAS$^+$ problem Π, any set of $CG(\Pi)$'s subgraphs $\mathbf{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_m\}$, and any additive CGSP decomposition of Π over $\mathbf{G}$, we have $h^*(s) \geq \sum_{i=1}^{m} h_i^*(s^{[V_{\mathcal{G}_i}]})$ for all states s of Π.*

Relying on Proposition 2, we can now decompose any given problem Π into a set of tractable CGSPs $\mathbf{\Pi} = \{\Pi_{\mathcal{G}_1}, \ldots, \Pi_{\mathcal{G}_m}\}$, solve all these CGSPs in polynomial time, and derive an admissible heuristic for Π. Note that (similarly to Definition 2) Definition 4 leaves the decision about the actual partition of the action costs rather open. In what follows we adopt the most straightforward, *uniform* action-cost partitioning in which the cost of each action a is *equally split* among all the non-redundant abstractions of a in $\mathbf{\Pi}$. The choice of the action-cost partitioning, however, can sometimes be improved or even optimized; for further details see (Katz & Domshlak 2008).

Fork-Decompositions

We now introduce certain concrete additive decompositions of SAS$^+$ planning problems along their causal graphs. In itself, these decompositions do *not* immediately lead to structural patterns abstractions, yet they provide an important building block on our way towards them.

Definition 5 *Let $\Pi = \langle V, A, I, G\rangle$ be a SAS$^+$ problem.*

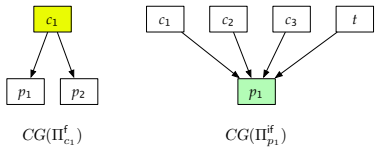


Figure 3: Causal graphs of a fork and an inverted fork structural patterns of the running example.

– **$\mathcal{F}$-decomposition $\mathbf{\Pi}_{\mathcal{F}} = \{\Pi^{\mathsf{f}}_v\}_{v \in V}$,**
– **$\mathcal{I}$-decomposition $\mathbf{\Pi}_{\mathcal{I}} = \{\Pi^{\mathsf{i}}_v\}_{v \in V}$, *and***
– **$\mathcal{FI}$-decomposition $\mathbf{\Pi}_{\mathcal{FI}} = \{\Pi^{\mathsf{f}}_v, \Pi^{\mathsf{i}}_v\}_{v \in V}$**

are additive CGSP decompositions of Π over sets of subgraphs $\mathbf{G}_{\mathcal{F}} = \{\mathcal{G}^{\mathsf{f}}_v\}_{v \in V}$, $\mathbf{G}_{\mathcal{I}} = \{\mathcal{G}^{\mathsf{i}}_v\}_{v \in V}$, and $\mathbf{G}_{\mathcal{FI}} = \mathbf{G}_{\mathcal{F}} \cup \mathbf{G}_{\mathcal{I}}$, respectively, where, for $v \in V$,

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

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

Note that all three fork-decompositions in Definition 5 are entirely *non-parametric* in the sense of flexibility left by the general definition of CGSPs. Illustrating Definition 5, let us consider the (uniform) $\mathcal{FI}$-decomposition of the problem Π from our running example, assuming all the actions in Π have the same unit cost. After eliminating from $\mathbf{G}_{\mathcal{FI}}$ all the singletons[2], we get $\mathbf{G}_{\mathcal{FI}} = \{\mathcal{G}^{\mathsf{f}}_{c_1}, \mathcal{G}^{\mathsf{f}}_{c_2}, \mathcal{G}^{\mathsf{f}}_{c_3}, \mathcal{G}^{\mathsf{f}}_{t}, \mathcal{G}^{\mathsf{i}}_{p_1}, \mathcal{G}^{\mathsf{i}}_{p_2}\}$. Considering the action sets of the problems in $\mathbf{\Pi}_{\mathcal{FI}}$, each original driving action is present in some three problems in $\mathbf{\Pi}_{\mathcal{FI}}$, while each load/unload action is present in some five such problems. For instance, the action "drive-c_1-from-A-to-D" is present in $\{\Pi^{\mathsf{f}}_{c_1}, \Pi^{\mathsf{i}}_{p_1}, \Pi^{\mathsf{i}}_{p_2}\}$, and the action "load-$p_1$-into-$c_1$-at-A" is present in $\{\Pi^{\mathsf{f}}_{c_1}, \Pi^{\mathsf{f}}_{c_2}, \Pi^{\mathsf{f}}_{c_3}, \Pi^{\mathsf{f}}_{t}, \Pi^{\mathsf{i}}_{p_1}\}$. Since we assume a uniform partitioning of the action costs, the cost of each driving and load/unload action in each corresponding abstract problem is set to $1/3$ and $1/5$, respectively.

From Proposition 2 we have the sum of costs of solving the problems $\mathbf{\Pi}_{\mathcal{FI}}$,

$$h^{\mathcal{FI}} = \sum_{v \in V} \left(h^*_{\Pi^{\mathsf{f}}_v} + h^*_{\Pi^{\mathsf{i}}_v} \right), \tag{3}$$

being an admissible estimate of h^*. The question now is how good this estimate is. The optimal cost of solving our problem is 19. Taking as a basis for comparison the seminal (non-parametric) $h_{\max}$ and h^2 heuristics (Bonet & Geffner 2001; Haslum & Geffner 2000), we have $h_{\max} = 8$ and $h^2 = 13$. At the same time, we have $h^{\mathcal{FI}} = 15$, and hence it appears that using $h^{\mathcal{FI}}$ is at least promising.

Unfortunately, despite the seeming simplicity of the problems in $\mathbf{\Pi}$, turns out that fork-decompositions as they are do not fit the requirements of the structural patterns framework. On the one hand, the causal graphs of $\{\Pi^{\mathsf{f}}_{c_1}, \Pi^{\mathsf{f}}_{c_2}, \Pi^{\mathsf{f}}_{c_3}, \Pi^{\mathsf{f}}_{t}\}$ and $\{\Pi^{\mathsf{i}}_{p_1}, \Pi^{\mathsf{i}}_{p_2}\}$ form directed forks and inverted forks, respectively (see Figure 3), and, in general, the number of

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

variables in each such problem is $\Theta(n)$. On the other hand, 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 state variables are strongly connected, optimal planning for forks and inverted forks remain NP-hard (see Helmert (2003) and (2004) for the respective results). In the next section, however, we show that this is not the end of the story for fork-decompositions.

Meeting CGSPs 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 the corresponding 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 is not incidental, and Propositions 3 and 4 below characterize some substantial islands of tractability within these structural fragments of SAS$^+$.

Proposition 3 (Tractable Forks) *Given a* SAS$^+$ *problem* $\Pi = \langle V, A, I, G\rangle$ *with a fork causal graph rooted at* r*, if (i)* $|dom(r)| = 2$*, or (ii) for all* $v \in V$*,* $|dom(v)| = O(1)$*, then cost-optimal planning for* Π *is poly-time.*

For the next proposition we use the notion of k-dependent actions—an action a is called k-dependent if it is preconditioned by $\leq k$ variables that are not affected by a (Katz & Domshlak 2007).

Proposition 4 (Tractable Inverted Forks) *Given a* SAS$^+$ *problem* $\Pi = \langle V, A, I, G\rangle$ *with an inverted fork causal graph rooted at* $r \in V$*, if* $|dom(r)| = O(1)$ *and all the actions* A *are 1-dependent, then cost-optimal planning for* Π *is poly-time.*

Propositions 3 and 4 allow us to meet between the fork-decompositions and structural patterns. 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. Such domain abstractions have been suggested for domain-independent planning by Hoffmann *et al.* (2006), and recently successfully exploited in planning as heuristic search by Helmert *et al.* (2007).

Definition 6 *Let* $\Pi = \langle V, A, I, G\rangle$ *be a* SAS$^+$ *problem,* $v \in V$ *be a variable of* Π*, and* $\Phi = \{\phi_1, \ldots, \phi_k\}$ *be a set of mappings from* $dom(v)$ *to some sets* $\Gamma_1, \ldots, \Gamma_k$*. An* **additive domain decomposition of** Π **over** Φ *is a set of* SAS$^+$ *problems* $\mathbf{\Pi}_\Phi = \{\Pi_1, \ldots, \Pi_k\}$ *such that*

(1) For each $\Pi_i = \langle V_i, A_i, I_i, G_i\rangle$*, we have*[3]

(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) For each $a \in A$ *holds*

$$\mathcal{C}(a) \geq \sum_{i=1}^{k} \mathcal{C}_i\left(\phi_i(a)\right). \tag{4}$$

Proposition 5 *For any* SAS$^+$ *problem* Π *over variables* V*, any variable* $v \in V$*, any domain abstractions* $\Phi = \{\phi_i\}_{i=1}^k$ *of* v*, and any additive domain decomposition of* Π *over* Φ*, we have* $h^*(s) \geq \sum_{i=1}^k h^*_i(\phi_i(s))$ *for all states* s *of* Π*.*

Targeting tractability of the causal graph structural patterns, we now connect between fork-decompositions and domain decompositions as in Definition 6. Given a $\mathcal{FJ}$-decomposition $\mathbf{\Pi}_{\mathcal{FJ}} = \{\Pi^{\mathsf{f}}_v, \Pi^{\mathsf{i}}_v\}_{v\in V}$ of Π, we

- For each $\Pi^{\mathsf{f}}_v \in \mathbf{\Pi}$, 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 $\phi_{v,i} : dom(r) \to \{0, 1\}$, and then additively decompose Π^{f}_v into $\mathbf{\Pi}^{\mathsf{f}}_v = \{\Pi^{\mathsf{f}}_{v,i}\}_{i=1}^{k_v}$ over Φ_v.
- For each $\Pi^{\mathsf{i}}_v \in \mathbf{\Pi}$, first, reformulate it in terms of 1-dependent actions only; such a reformulation can easily be done in time/space $O(poly(|\Pi^{\mathsf{i}}_v|))$. Then, associate the root r of $CG(\Pi^{\mathsf{i}}_v)$ with mappings $\Phi'_v = \{\phi'_{v,1}, \ldots, \phi'_{v,k'_v}\}$, $k'_v = O(poly(|\Pi|))$, and all $\phi'_{v,i} : dom(r) \to \{0, 1, \ldots, b_{v,i}\}$, $b_{v,i} = O(1)$, and then additively decompose Π^{i}_v into $\mathbf{\Pi}^{\mathsf{i}}_v = \{\Pi^{\mathsf{i}}_{v,i}\}_{i=1}^{k'_v}$ over Φ'_v.

From Propositions 2 and 5 we then have

$$h^{\mathcal{FJ}} = \sum_{v\in V}\left(\sum_{i=1}^{k_v} h^*_{\Pi^{\mathsf{f}}_{v,i}} + \sum_{i=1}^{k'_v} h^*_{\Pi^{\mathsf{i}}_{v,i}}\right), \tag{5}$$

being an admissible estimate of h^* for Π, and, from Propositions 3 and 4, $h^{\mathcal{FJ}}$ is also poly-time computable. The question is, however, how further abstracting our fork decompositions using domain abstractions as above affects the informativeness of the heuristic estimate. Below we show that the answer to this question can be somewhat surprising.

To illustrate a mixture of structural and domain abstractions as above, here as well we use our running Logistics-style example. 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 $\vartheta \in dom(v)$,

$$\phi_{v,i}(\vartheta) = \phi'_{v,i}(\vartheta) = \begin{cases} 0, & d(I[v], \vartheta) < i \\ 1, & \text{otherwise} \end{cases} \tag{6}$$

For example, 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 binary abstract domains of c_1 corresponding to the partitions

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

$\{A\}/\{B,C,D\}$ and $\{A,D\}/\{B,C\}$ of $dom(c_1)$, respectively. Now, given the decomposition of Π over forks and $\{\Phi_v, \Phi'_v\}_{v\in V}$ as above, consider the problem $\Pi^{\mathsf{i}}_{p_1,1}$, obtained from abstracting Π along 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, we are left in $\Pi^{\mathsf{i}}_{p_1,1}$ only with actions conditioned by c_1 and c_2. If so, then no information is lost[4] if we

1. remove from $\Pi^{\mathsf{i}}_{p_1,1}$ both 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 can be obtained directly by replacing the cost-partitioning steps corresponding to Eqs. 2 and 4 by a single, joint action cost partitioning applied over the final abstractions $\bigcup_{v\in V}(\mathbf{\Pi}^{\mathsf{f}}_v \cup \mathbf{\Pi}^{\mathsf{i}}_v)$ and satisfying

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

Overall, computing $h^{\mathcal{FJ}}$ as in Eq. 5 under "all binary-range domain abstractions" provides us with $h^{\mathcal{FJ}} = 12\frac{7}{15}$, and knowing that the original costs are all integers we can safely adjust it to $h^{\mathcal{FJ}} = 13$. Hence, even under the most severe domain abstractions as above, $h^{\mathcal{FJ}}$ does not fall from h^2 in our example problem.

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, $\{\Phi'_v\}$ are set to

$$\forall \vartheta \in dom(v): \quad \phi'_{v,i} = \begin{cases} 0, d(I[v],\vartheta) < 2i-1 \\ 1, d(I[v],\vartheta) = 2i-1 \\ 2, d(I[v],\vartheta) > 2i-1 \end{cases} \tag{8}$$

For example, the problem $\Pi^{\mathsf{i}}_{p_1}$ is now decomposed into $\Pi^{\mathsf{i}}_{p_1,1},\ldots,\Pi^{\mathsf{i}}_{p_1,3}$ along the abstractions of $dom(p_1)$. Applying now the same computation of $h^{\mathcal{FJ}}$ as in Eq. 5 over the new set of domain abstractions gives $h^{\mathcal{FJ}} = 15\frac{1}{2}$, which, again, can be safely adjusted to $h^{\mathcal{FJ}} = 16$. Note that this value is *higher* than $h^{\mathcal{FJ}} = 15$ obtained using the (generally intractable) "pure" fork-decomposition as in Eq. 3. At first view, this outcome may seem counterintuitive as the domain abstractions are applied *over* the fork-decomposition, and one would expect a stronger abstraction to provide less precise estimates. This, however, is not necessarily the case. For instance, domain abstraction for the root of an inverted fork may create independence between the root and its preconditioning parent variables, and exploiting such domain-abstraction-specific independence relations leads to more targeted action cost partitioning in Eq. 7. To illustrate such a surprising "estimate improvement", notice that, before applying the domain abstraction as in Eq. 8 on our example, the truck-moving actions $move\text{-}D\text{-}E$ and $move\text{-}E\text{-}D$ appear in *three* patterns Π^{f}_t, $\Pi^{\mathsf{i}}_{p_1}$ and $\Pi^{\mathsf{i}}_{p_2}$, while after domain abstraction they appear in *five* patterns $\Pi^{\mathsf{f}}_{t,1}$, $\Pi^{\mathsf{i}}_{p_1,1}$, $\Pi^{\mathsf{i}}_{p_1,2}$, $\Pi^{\mathsf{i}}_{p_1,3}$ and $\Pi^{\mathsf{i}}_{p_2,1}$. However, a closer look at the action sets of these five patterns reveals that the dependencies of p_1 in $CG(\Pi^{\mathsf{i}}_{p_1,1})$ and $CG(\Pi^{\mathsf{i}}_{p_1,3})$, and of p_2 in $CG(\Pi^{\mathsf{i}}_{p_2,1})$ on t are redundant, and thus there is no need to keep the representatives of $move\text{-}D\text{-}E$ and $move\text{-}E\text{-}D$ in the corresponding patterns. Hence, after all, the two truck-moving actions appear only in *two* post-domain-abstraction patterns. Moreover, in both these patterns the truck-moving actions are fully counted, and this in contrast to the pre-domain-abstraction patterns where the portion of the cost of these actions allocated to $\Pi^{\mathsf{i}}_{p_2}$ simply gets lost.

[4] No information is lost here because we still keep either fork or inverted fork for each variable of Π.

Accuracy of Fork-Decomposition Heuristics

Going beyond our running example, obviously we would like to assess the effectiveness of fork-decomposition heuristics on a wider set of domains and problem instances. A standard method for that is to implement admissible heuristics within some optimality-preserving search algorithm, run it against a number of benchmark problems, and count the number of node expansions performed by the search algorithm. The fewer nodes the algorithm expands, the better. While such experiments are certainly useful and important, as noted by Helmert and Mattmüller (2008), their results almost never lead to absolute statements of the type "Heuristic h is well-suited for solving problems from benchmark suite X", but only to relative statements of the type "Heuristic h expands fewer nodes than heuristic h' on a benchmark suite X". Moreover, one would probably like to get a formal certificate for the effectiveness of her heuristic before proceeding with its implementation.

In what follows, we formally analyze the effectiveness of the fork-decomposition heuristics similarly to the way Helmert and Mattmüller (2008) study some state-of-the-art admissible heuristics. Given domain $\mathcal{D}$ and heuristic h, Helmert and Mattmüller consider the *asymptotic performance ratio* of h in $\mathcal{D}$. The goal is to find a value $\alpha(h,\mathcal{D}) \in [0,1]$ such that (i) for all states s in all problems $\Pi \in \mathcal{D}$, $h(s) \geq \alpha(h,\mathcal{D}) \cdot h^*(s) + o(h^*(s))$, and (ii) there is a family of problems $\{\Pi_n\}_{n\in\mathbb{N}} \subseteq \mathcal{D}$ and solvable, non-goal states $\{s_n\}_{n\in\mathbb{N}}$ such that $s_n \in \Pi_n$, $\lim_{n\to\infty} h^*(s_n) = \infty$, and $h(s_n) \leq \alpha(h,\mathcal{D}) \cdot h^*(s_n) + o(h^*(s_n))$. In other words, h is *never* worse than $\alpha(h,\mathcal{D}) \cdot h^*$ (plus a sublinear term), and it can become as bad as $\alpha(h,\mathcal{D}) \cdot h^*$ (plus a sublinear term) for arbitrary large inputs; note that the existence and uniqueness of $\alpha(h,\mathcal{D})$ are guaranteed for any h and $\mathcal{D}$.

Helmert and Mattmüller (2008) study the asymptotic performance ratio of the admissible heuristics h^+, h^k, h^{PDB}, and $h^{\mathrm{PDB}}_{\mathrm{add}}$ on some benchmark domains from the first four International Planning Competitions, namely GRIPPER,

Domain	h^+	h^k	h^{PDB}	$h^{\text{PDB}}_{\text{add}}$	$h^{\mathcal{F}}$	$h^{\mathcal{J}}$	$h^{\mathcal{FJ}}$
GRIPPER	2/3	0	0	2/3	2/3	0	1/3
LOGISTICS	3/4	0	0	1/2	1/2	1/2	1/2
BLOCKSWORLD	1/4	0	0	0	0	0	0
MICONIC	6/7	0	0	1/2	5/6	1/2	1/2
SATELLITE	1/2	0	0	1/6	1/6	1/6	1/6

Table 1: Performance ratios of multiple heuristics in selected planning domains; ratios for h^+, h^k, h^{PDB}, $h^{\text{PDB}}_{\text{add}}$ are by Helmert and Mattmüller (2008).

LOGISTICS, BLOCKSWORLD, MICONIC-STRIPS, SATELLITE, MICONIC-SIMPLE-ADL, and SCHEDULE. (Familiarity with the domains is assumed; for an overview consult Helmert, 2006b.) The h^+ estimate corresponds to the optimal cost of solving the well-known "ignore-deletes" abstraction of the original problem, and it is generally NP-hard to compute (Bylander 1994). The h^k, $k \in \mathbb{N}^+$, family of heuristics is based on a relaxation where the cost of reaching a set of n satisfied atoms is approximated by the highest cost of reaching a subset of k satisfied atoms (Bonet & Geffner 2001); computing h^k is exponential only in k. The h^{PDB} and $h^{\text{PDB}}_{\text{add}}$ heuristics are regular (maximized over) and additive (summed-up) pattern database heuristics where the size of each pattern is assumed to be $O(\log(n))$, and (importantly) the choice of the patterns is assumed to be optimal.

The results of Helmert and Mattmüller provide us with a baseline for evaluating our admissible heuristics $h^{\mathcal{F}}$, $h^{\mathcal{J}}$, and $h^{\mathcal{FJ}}$ corresponding to $\mathcal{F}$-, $\mathcal{J}$-, and $\mathcal{FJ}$-decompositions, respectively. At this point, the reader may rightfully wonder whether some of these three heuristics are not dominated by the others, and thus are redundant for our analysis. Proposition 6, however, shows that this is not the case—each of these three heuristics can be strictly more informative than the other two, depending on the problem instance and/or the state being evaluated.

Proposition 6 (Undominance) *None of the heuristic functions $h^{\mathcal{F}}$, $h^{\mathcal{J}}$, and $h^{\mathcal{FJ}}$ (with and/or without domain abstractions) dominate each other.*

Table 1 presents now the asymptotic performance ratios of $h^{\mathcal{F}}$, $h^{\mathcal{J}}$, and $h^{\mathcal{FJ}}$ in GRIPPER, LOGISTICS, BLOCKSWORLD, MICONIC-STRIPS, and SATELLITE,[5] putting them in line with the corresponding results of Helmert and Mattmüller for h^+, h^k, h^{PDB}, and $h^{\text{PDB}}_{\text{add}}$. We have also studied the ratios of $\max\{h^{\mathcal{F}}, h^{\mathcal{J}}, h^{\mathcal{FJ}}\}$, and in these five domains they appear to be identical to these of $h^{\mathcal{F}}$.[6] Taking a conservative position, the performance ratios for the fork-decomposition heuristics in Table 1 are "worst-case" in the sense that

(i) here we neither optimize the action-cost partitioning (setting it to "uniform"), nor eliminate clearly redundant patterns, and

(ii) use domain abstractions to (up to) ternary-valued abstract domains only.

Specifically, the domains of the inverted-fork roots are all abstracted using the "distance-from-initial-value" ternary-valued domain decompositions as in Eq. 8, while the domains of the fork roots are all abstracted using the "leave-one-out" binary-valued domain decompositions as in Eq. 9.

$$\forall \vartheta_i \in dom(v): \quad \phi_{v,i}(\vartheta) = \begin{cases} 0, & \vartheta = \vartheta_i \\ 1, & \text{otherwise} \end{cases} \tag{9}$$

In a sketch, the results in Table 1 are obtained as follows.

GRIPPER Assuming $n > 0$ balls should be moved from one room to another, all the three heuristics $h^{\mathcal{F}}, h^{\mathcal{J}}, h^{\mathcal{FJ}}$ account for all the required pickup and drop actions, and only for $O(1)$-portion of move actions. On the other hand, the former actions are responsible for $2/3$ of the optimal-plan length (= cost). Now, with the basic uniform action-cost partition, $h^{\mathcal{F}}$, $h^{\mathcal{J}}$, and $h^{\mathcal{FJ}}$ account for whole, $O(1/n)$, and $1/2$ of the total pickup/drop actions' cost, respectively, providing the ratios as in Table 1.[7]

LOGISTICS Optimal plan contains at least as much loads/unloads as move actions, and all the three heuristics $h^{\mathcal{F}}, h^{\mathcal{J}}, h^{\mathcal{FJ}}$ fully account for the former, providing a lower bound of $1/2$. An instance on which all three heuristics achieve exactly $1/2$ consists of two trucks t_1, t_2, no airplanes, one city, and n packages such that the initial and goal locations of all the packages and trucks are all pair-wise different.

BLOCKSWORLD Arguments similar to these of Helmert and Mattmüller (2008) for $h^{\text{PDB}}_{\text{add}}$.

MICONIC-STRIPS All the three heuristics fully account for all the loads/unloads. In addition, $h^{\mathcal{F}}$ accounts for the full cost of all the moves to the passengers' initial locations, and for half of the cost of all other moves. This provides us with the lower bounds of $1/2$ and $5/6$, respectively. Tightness of $1/2$ for $h^{\mathcal{J}}$ and $h^{\mathcal{FJ}}$ is, e.g., on the instance consisting of n passengers, $2n+1$ floors, and all the initial and goal locations being pair-wise different. Tightness of $5/6$ for $h^{\mathcal{F}}$ is, e.g., on the instance consisting of n passengers, $n+1$ floors, the elevator and all the passengers are initially at floor $n+1$, and each passenger i wishes to get to floor i.

SATELLITE The length of an optimal plan for a problem with n images to be taken and k satellites to be moved to some end-positions is $\leq 6n + k$. All the three heuristics fully account for all the image-taking actions, and one satellite-moving action per satellite as above, providing a lower bound of $\frac{1}{6}$. Tightness of $1/6$ for all three heuristics is on the following instance: Two satellites with instruments $\{i\}_{i=1}^{l}$ and $\{i\}_{i=l+1}^{2l}$, respectively, where $l = n - \sqrt{n}$. Each pair of instruments $\{i, l+i\}$ can take images in modes $\{m_0, m_i\}$. There is a set of directions $\{d_j\}_{j=0}^{n}$ and a set of image objectives $\{o_i\}_{i=1}^{n}$ such

[5] We have not accomplished yet the analysis of MICONIC-SIMPLE-ADL and SCHEDULE; the action sets in these domains are much more heterogeneous, and thus more involved for analytic analysis of fork-decompositions.

[6] Note that "ratio of max" should not necessarily be identical to "max of ratios".

[7] We note that a very slight modification of the uniform action-cost partition results in ratio of $2/3$ for all our three heuristics. Such optimizations, however, are outside of our scope here.

that, for $1 \leq i \leq l$, $o_i = (d_0, m_i)$, and, for $l < i \leq n$, $o_i = (d_i, m_0)$. Finally, the calibration direction for each pair of instruments $\{i, l+i\}$ is d_i.

Overall, the results for fork-decomposition heuristics in Table 1 are very gratifying. First, note that the performance ratios for h^k and h^{PDB} are all 0. This is because every k-elementary (for h^k) and $\log(n)$-elementary (for h^{PDB}) subgoal set can be reached in the number of steps that only depends on k (respectively, $\log(n)$), and not n, while $h^*(s_n)$ grows linearly in n in all the five domains. This leaves us with $h^{\text{PDB}}_{\text{add}}$ being the only state-of-the-art (tractable and) admissible heuristic to compare with. Table 1 shows that the asymptotic performance ratio of $\max\{h^{\mathcal{F}}, h^{\mathcal{J}}, h^{\mathcal{FJ}}\}$ is at least as good as this of $h^{\text{PDB}}_{\text{add}}$ in *all* five domains, and it is superior to $h^{\text{PDB}}_{\text{add}}$ in MICONIC-STRIPS, getting here quite close to h^+. Comparing between $h^{\text{PDB}}_{\text{add}}$ and fork-decomposition heuristics, it is crucial to recall that the ratios devised by Helmert and Mattmüller for $h^{\text{PDB}}_{\text{add}}$ are with respect to *optimal, manually-selected* set of patterns. In contrast, fork-decomposition heuristics are completely *non-parametric*, and thus require no tuning of the pattern-selection process.

Summary

We presented a generalization of the pattern-database projections, called structural patterns, that is based on abstracting 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 patterns to be of a low dimensionality. We defined the notion of (additive) causal graph structural patterns (CGSPs), and studied their potential on a concrete CGSP framework 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. We showed that the asymptotic performance ratios of the resulting heuristics on selected planning domains are at least as good, and sometimes transcends, these of the state-of-the-art admissible heuristics.

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. Likewise, currently we explore combining structural patterns (and, in particular, CGSPs) with PDB-style projection patterns, as well as with more flexible "merge-and-shrink" abstractions suggested in (Helmert, Haslum, & Hoffmann 2007). Very roughly, the idea here is to "glue" and/or "duplicate" certain variables prior to projecting the problem in hand onto its tractable structural components. We believe that a successful such combination of techniques has a potential to improve the effectiveness of the heuristics, and in particular their domain-dependent asymptotic performance ratios.

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.; Botea, A.; Helmert, M.; Bonet, B.; and Koenig, S. 2007. Domain-independent construction of pattern database heuristics for cost-optimal planning. In *AAAI*, 1007–1012.

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

Helmert, M., and Mattmüller, R. 2008. Accuracy of admissible heuristic functions in selected planning domains. In *AAAI*. (Extended abstract in the ICAPS'07 workshops).

Helmert, M.; Haslum, P.; and Hoffmann, J. 2007. Flexible abstraction heuristics for optimal sequential planning. In *ICAPS*, 176–183.

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. 2006a. The Fast Downward planning system. *JAIR* 26:191–246.

Helmert, M. 2006b. *Solving Planning Tasks in Theory and Practice*. Ph.D. Dissertation, Albert-Ludwigs University, Freiburg.

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

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*, 200–207.

Katz, M., and Domshlak, C. 2008. Optimal additive composition of abstraction-based admissible heuristics. In *ICAPS (this volume)*.

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