Structural Patterns of Tractable Sequentially-Optimal Planning

Michael Katz and **Carmel Domshlak**
Faculty of Industrial Engineering and Management
Technion—Israel Institute of Technology
Haifa, Israel

Abstract

We study the complexity of sequentially-optimal classical planning, and discover new problem classes for whose such optimization is tractable. The results are based on exploiting numerous structural characteristics of planning problems, and a constructive proof technique that connects between certain tools from planning and tractable constraint optimization. In particular, we believe that structure-based tractability results of this kind may help devising new admissible search heuristics. We discuss the prospects of this direction along a principled extension of pattern-database heuristics to "structural patterns" of unlimited dimensionality.

Introduction

General planning is known to be computationally hard (Chapman 1987; Erol, Nau, & Subrahmanian 1995), and even propositional planning is PSPACE-complete (Bylander 1994). However, computational tractability remains a fundamental issue in automated problem solving, and the reason for that is twofold. First, many planning problems in the manufacturing and process industry are believed to be highly structured, allowing for efficient planning if exploiting this structure (Williams & Nayak 1997; Klein, Jonsson, & Bäckström 1998). Second, tractable subclasses of planning underly some of the most influential heuristics that have been suggested for planning via directional search (Bonet & Geffner 2001; Hoffmann & Nebel 2001; Refanidis & Vlahavas 2001; Haslum & Geffner 2000). Unfortunately, this far the palette of tractable planning remains extremely limited, and the situation is even more severe for tractable *optimal* planning. To date, less than a handful of non-trivial fragments of optimal planning problems are known to be tractable (Bäckström & Klein 1991; Bylander 1994; Jonsson & Bäckström 1998; Brafman & Domshlak 2006; Jonsson 2007), and this across numerous notions of optimality that have been considered useful.

We believe that extending the toolbox of tractable optimal planning is critically important as it may allow us, for instance, extending the (still extremely limited) palette of admissible heuristics for planning as search. In this work we study the complexity of *sequentially-optimal planning* within the UB fragment of classical planning problems. Specifically, UB is a fragment of the SAS$^+$ formalism (Bäckström & Nebel 1995) that corresponds to problems with binary-valued state variables and unary-effect actions. The notion of sequential optimality is based on probably the most canonical plan quality measure that sums up the individual costs of the plan's actions. In general, even non-optimal planning for UB is PSPACE-complete, that is, as hard as general classical planning (Bylander 1994). Moreover, only a handful of UB sub-fragments are known to be tractable (Bäckström & Klein 1991; Williams & Nayak 1997; Jonsson & Bäckström 1998; Brafman & Domshlak 2003), and the sequentially-optimal planning is tractable only for two such sub-fragments of UB (Bäckström & Klein 1991; Jonsson & Bäckström 1998).

Here we study the complexity of sequentially-optimal planning for UB as a function of

(1) The *global structure* of the problem's *causal graph*. The special cases we consider here correspond to trees, inverted trees, singly-connected DAGs (polytrees), and directed-path singly-connected DAGs.

(2) The *local connectivity* of the causal graph. Here we study the impact (or absence of such) of the in-degree and/or out-degree of the causal graph's nodes being bounded by a constant.

(3) The k*-dependence* property of the problem's actions. This property has a close connection to the problem's causal graph, but is not expressed by the latter. In some sense, this property refers to a certain *local structure* of the problem's actions with respect to the causal graph.

Considering the interplay between the three parameters above, we discover novel tractable sub-fragments of UB, and in particular, of the sequentially-optimal planning for UB. For some of these fragments, we show that relaxing their specification along any of the three dimensions leads to an NP-hard class of problem. Our tractability results are based (each in a different manner) on a proof technique that connects between certain tools from planning and tractable constraint optimization. This technique appears to be rather robust, suggesting that further investigation in this direction is very promising.

While discovering islands of tractability in the sea of (optimal) planning is of theoretical interest on its own, we believe that *structurally-characterized* tractable fragments of problems (such as presented here), have a potential to help devising new admissible search heuristics. In the last part of the paper we discuss the prospects of this direction along a principled extension of the pattern-database heuristics (Culberson & Schaeffer 1998; Edelkamp 2001) to "structural patterns" that alleviate the requirement for patterns to be of a low (up to poly-logarithmic) dimensionality (Katz & Domshlak 2007).

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 a fragment of the SAS$^+$ formalism (Bäckström & Nebel 1995) that captures problem with binary state variables and unary-effect actions. Following (Bäckström & Klein 1991), in what follows we refer to this subclass of SAS$^+$ as UB. Considering the applicability of actions, in SAS$^+$ it helps to distinguish between *pre*-conditions and *prevail* conditions of the actions. The former are required values of variables that are affected by the action. The latter are required values of variables that are not affected by the action. The *post*-conditions of an action describe the new values taken by its precondition variables after the action execution. For example, having truck T and package P in location L are a prevail and a pre-condition for loading P into T in L, respectively—both conditions are essential, but, after performing the action, the truck is still in L, while the package is no longer there (but inside T).

Definition 1 *A* UB *problem instance is given by a quadruple* $\Pi = \langle \mathcal{V}, A, I, G \rangle$*, where:*

- $\mathcal{V} = \{v_1, \dots, v_n\}$ *is a set of* state variables *over* binary domains $Dom(v_i)$. *The domain* $Dom(v_i)$ *of* v_i *induces an* extended domain $Dom^+(v_i) = Dom(v_i) \cup \{\mathsf{u}\}$, *where* u *denotes the dummy value:* unspecified.
- I *is a fully specified* initial state, *that is,* $I \in \times Dom(v_i)$. *By* $I[i]$ *we denote the value of* v_i *in* I.
- $G \in \times Dom^+(v_i)$ *is a partial assignment to* $\mathcal{V}$, *specifying the set of alternative* goal states. *By* $G[i]$ *we denote the value provided by* G *to* v_i *(with, possibly,* $G[i] = \mathsf{u}$.*)*
- $A = \{a_1, \dots, a_N\}$ *is a finite set of* actions. *Each action* a *is a triplet* $\langle \mathsf{pre}(a), \mathsf{post}(a), \mathsf{prv}(a) \rangle$, *where* $\mathsf{pre}(a), \mathsf{post}(a), \mathsf{prv}(a) \subseteq \times Dom^+(v_i)$ *denote the* pre-, post-, *and* prevail conditions *of* a, *respectively. By* $\mathsf{pre}(a)[i]$, $\mathsf{post}(a)[i]$, *and* $\mathsf{prv}(a)[i]$ *we denote the corresponding values of* v_i.
- *All actions* A *are* unary-effect, *that is, for each* $a \in A$, *there is exactly one variable* $v_i \in \mathcal{V}$ *such that* $\mathsf{post}(a)[i] \neq \mathsf{u}$. *Likewise, for every* $v_i \in \mathcal{V}$, *we have*
  - *either* $\mathsf{pre}(a)[i] = \mathsf{u}$ *or* $\mathsf{prv}(a)[i] = \mathsf{u}$, *and*
  - $\mathsf{post}(a)[i] \neq \mathsf{u}$ *if and only if* $\mathsf{pre}(a)[i] \neq \mathsf{u}$, *in which case* $\mathsf{post}(a)[i] \neq \mathsf{pre}(a)[i]$.

  By $A_v \subseteq A$ *we denote the actions changing the value of* v, *and from unary-effectness we have* $A_v \cap A_{v'} = \emptyset$ *if* $v \neq v'$.

In this work we focus on *sequentially-optimal* planning (*SO-planning*, for short) in which, for every $\rho \in A^*$, we have $cost(s_0, \rho) = \sum_{a \in \rho} cost(a)$ if ρ is a plan for Π, and $cost(s_0, \rho) = \infty$ otherwise. The (possibly non-uniform) costs of actions in A are assumed to be all non-negative.

Following (Brafman & Domshlak 2003), here we relate between the complexity of UB planning and the *causal graph* structure (Bacchus & Yang 1994; Knoblock 1994; Williams & Nayak 1997; Brafman & Domshlak 2003; Domshlak & Dinitz 2001; Helmert 2006; Brafman & Domshlak 2006; Jonsson 2007).

Definition 2 *The* causal graph CG_Π *of a* UB *problem* $\Pi = \langle \mathcal{V}, A, I, G \rangle$ *is a digraph over the nodes* $\mathcal{V}$. *An edge* $(\overrightarrow{v_i, v_j})$ *appears in* CG_Π *if (and only if) some action* $a \in A$ *changes the value of* v_j *while having a prevail condition involving some value of* v_i, *that is,* $\mathsf{post}(a)[j] \neq \mathsf{u} \wedge \mathsf{prv}(a)[i] \neq \mathsf{u}$.

For each node $v \in CG_\Pi$, by $\mathsf{In}(v)$ and $\mathsf{Out}(v)$ we denote the in- and out-degrees of v, respectively, and $\mathsf{In}(CG_\Pi)/\mathsf{Out}(CG_\Pi)$ stand for the maximal in-degree/out-degree of the CG_Π nodes. Assuming CG_Π is connected, in what follows we give a special attention to the following *acyclic* cases of the causal graph's structure. A causal CG_Π is a

T *tree* if $\mathsf{In}(CG_\Pi) \leq 1$, and there exists $v \in \mathcal{V}$ such that $\mathsf{In}(v) = 0$.

I *inverted tree* if $\mathsf{Out}(CG_\Pi) \leq 1$, and there exists $v \in \mathcal{V}$ such that $\mathsf{Out}(v) = 0$.

P *polytree* if CG_Π contains no undirected cycles.

S *directed-path singly connected* if there is at most one directed path from each node $v \in CG_\Pi$ to any other node $v' \in CG_\Pi$.

In what follows, we use $\mathbf{T}$, $\mathbf{I}$, $\mathbf{P}$, and $\mathbf{S}$ to refer to the corresponding fragments of UB, and we use subscript/superscript b to refer to a fragment induced by the additional constraint of in-degree/out-degree being bounded by a constant. It is not hard to verify that we have $\mathbf{T}, \mathbf{I} \subset \mathbf{P} \subset \mathbf{S}$, with $\mathbf{T} \subset \mathbf{P}_b$ and $\mathbf{I} \subset \mathbf{P}^b$.

Causal Graph and Bounded Connectivity

The complexity of UB planning as a function of the causal graphs structure has been first studied in (Brafman & Domshlak 2003), where it was shown that non-optimal planning is tractable for $\mathbf{P}_b$ and NP-complete[1] for $\mathbf{S}_b$. The gap left in (Brafman & Domshlak 2003) has been recently closed in (Giménez 2006) where it was shown that planning for $\mathbf{P}$ is NP-complete, and the proof of this result actually carries out to the $\mathbf{I}$ fragment as well. In addition, since out-degree of the causal graph in $\mathbf{I}$ is ≤ 1, the result in (Giménez 2006) immediately implies that bounding out-degree of the nodes can possibly lead to tractability only in case of $\mathbf{P}_b^b$, or one of its sub-fragments.

Considering only the structure of the causal graph and its local connectivity, all the above leaves us with $\mathbf{P}_b$ still being a candidate for tractable SO-planning. Our first positive result below affirms this possibility.

Theorem 1 *SO-planning for* $\mathbf{P}_b$ *is tractable.*

The proof of Theorem 1 is constructive, and our algorithm for SO-planning for $\mathbf{P}_b$ is based on compiling a $\mathbf{P}_b$ problem Π into a *constraint optimization problem* $\mathsf{COP}_\Pi = (\mathcal{X}, \mathcal{F})$ over variables $\mathcal{X}$, functional components $\mathcal{F}$, and the global objective $\min \sum_{\varphi \in \mathcal{F}} \varphi(\mathcal{X})$ such that

(i) the description size of COP_Π is polynomial in the description size of Π,

(ii) the tree-width of the cost network of COP_Π is bounded by a constant,

(iii) if Π is unsolvable then all the assignments to $\mathcal{X}$ evaluate the objective function to ∞, and otherwise, the optimum of the global objective is obtained on and only on the assignments to $\mathcal{X}$ that correspond to SO-plans for Π,

(iv) given an optimal solution to COP_Π, the corresponding SO-plan for Π can be reconstructed from the former in polynomial time.

Having such a compilation scheme, we then can solve COP_Π using the standard, poly-time algorithm for constraint optimization over trees (Dechter 2003), and find an optimal solution for Π. Here we reuse some data structures suggested in (Brafman & Domshlak 2003) for non-optimal planning for $\mathbf{P}_b$, but exploit them in a very much different manner. Likewise, useful for us Corollary 1 below follows from Lemma 1 in (Brafman & Domshlak 2003).

Corollary 1 *For any solvable problem* $\Pi \in \mathbf{S}$ *over* n *state variables, any SO-plan* ρ *for* Π*, and any state variable* v *in* Π*, the number of value changes of* v *along* ρ *is* $\leq n$.

We begin with providing some helpful notation that refers to the components of a given UB problem $\Pi = \langle \mathcal{V}, A, I, G \rangle$. First, for each variable $v \in \mathcal{V}$, we denote the initial value $I[v]$ of v by b_v, and the opposite value by w_v. Using this notation and Corollary 1, by

$$\sigma(v) = \begin{cases} \mathsf{b}_v^1 \cdot \mathsf{w}_v^1 \cdot \mathsf{b}_v^2 \cdots \mathsf{b}_v^{j+1}, & n = 2j, \\ \mathsf{b}_v^1 \cdot \mathsf{w}_v^1 \cdot \mathsf{b}_v^2 \cdots \mathsf{w}_v^{j}, & n = 2j-1, \end{cases}, j \in \mathbb{N}$$

we denote the maximally possible, locally time-stamped sequence of values of v along an optimal plan ρ. Next, for each variable v, by $\trianglerighteq[v]$ we denote the set of all non-empty *prefixes* of $\sigma(v)$. A prefix $\sigma' \in \trianglerighteq[v]$ is called *valid* if the last element of σ' is compatible with $G[v]$, that is, equals $G[v]$ if $G[v] \neq \mathsf{u}$. The subset of all valid prefixes of $\sigma(v)$ is denoted by $\trianglerighteq^*[v] \subseteq \trianglerighteq[v]$.

We now proceed with specifying the constraint optimization problem COP_Π.

1. The variable set $\mathcal{X}$ contains a variable x_v for each planning variable $v \in \mathcal{V}$, and the domain $Dom(x_v)$ consists of all valid prefixes of $\sigma(v)$. That is,
$$\mathcal{X} = \{x_v \mid v \in \mathcal{V}\}, \quad Dom(x_v) = \trianglerighteq^*[v] \qquad (1)$$
2. For each planning variable v with parents $\mathsf{pred}(v) = \{w_1, \ldots, w_k\}$, the set $\mathcal{F}$ contains a non-negative, real-valued function φ_v with the scope
$$Q_v = \{x_v, x_{w_1}, \ldots, x_{w_k}\} \qquad (2)$$

The more involved part is specifying the function set $\mathcal{F}$. In what follows, we assume that, for each $v \in \mathcal{V}$, each $a \in A_v$, and each $w \in \mathsf{pred}(v)$, we have $\mathsf{prv}(a)[w] \neq \mathsf{u}$. (In case of $\mathbf{P}_b$, this assumption causes neither loss of generality nor a complexity blow up.) First, for each planning variable v with $\mathsf{pred}(v) = \emptyset$, and for each $\sigma' \in \trianglerighteq^*[v]$, we set

$$\varphi_v(\sigma') = \left\lfloor \frac{|\sigma'|}{2} \right\rfloor \cdot C(a_{\mathsf{w}_v}) + \left\lfloor \frac{|\sigma'| - 1}{2} \right\rfloor \cdot C(a_{\mathsf{b}_v}) \quad (3)$$

where $\mathsf{post}(a_{\mathsf{w}_v}) = \{\mathsf{w}_v\}$, $\mathsf{post}(a_{\mathsf{b}_v}) = \mathsf{b}_v$, and $C(a) = cost(a)$ if $a \in A$, and ∞, otherwise. It is easy to verify that $\varphi_v(\sigma')$ corresponds to the optimal cost of performing $|\sigma'| - 1$ value changes of v in Π.

Next we proceed with specifying the function φ_v for a variable v with $\mathsf{pred}(v) = \{w_1, \ldots, w_k\}, k \geq 1$. For

[1]The result in (Brafman & Domshlak 2003) is stated for $\mathbf{S}$, but its proof can easily be rephrased to hold for $\mathbf{S}_b$.

$$\begin{tikzcd}
v_1 \arrow[dr] & v_2 \arrow[d] & v_3 \arrow[dl] & & x_{v_1} \arrow[r, dash] \arrow[rr, dash, bend left] \arrow[dr, dash] & x_{v_2} \arrow[r, dash] \arrow[d, dash] & x_{v_3} \arrow[dl, dash] \\
 & v_4 \arrow[dl] \arrow[dr] & v_5 \arrow[d] & & & x_{v_4} \arrow[r, dash] \arrow[dl, dash] \arrow[dr, dash] & x_{v_5} \arrow[d, dash] \\
v_7 & & v_6 & & x_{v_7} & & x_{v_6}
\end{tikzcd}$$

Figure 1: Causal graph of a planning problem $\Pi \in \mathbf{P}_b$ (left), and the cost network of COP_Π (right).

each valid prefix $\sigma' \in \trianglerighteq^*[v]$, and each set of valid prefixes $\sigma'_1 \in \trianglerighteq^*[w_1], \ldots, \sigma'_k \in \trianglerighteq^*[w_k]$, we want to set $\varphi_v(\sigma', \sigma'_1, \ldots, \sigma'_k)$ to the *optimal cost of performing* $|\sigma'| - 1$ *value changes of* v *in* Π, *given that* $w_1, \ldots, w_k$ *change their values* $|\sigma'_1| - 1, \ldots, |\sigma'_k| - 1$ *times*, respectively. In what follows, we reduce determining $\varphi_v(\sigma', \sigma'_1, \ldots, \sigma'_k)$ to solving a single-source shortest path problem on a weighted directed graph $G'_e(v)$ that extends a similarly-named graphical structure suggested in (Brafman & Domshlak 2003).

Given $\sigma'_1, \ldots, \sigma'_k$ as above, the digraph $G'_e(v)$ is created in three steps. First, we construct a digraph $G(v) = (V, E)$, nodes of which stand for the elements of $\sigma(v)$, and, for each action $a \in A_v$, if $\mathsf{post}(a)[v] = \mathsf{w}_v$, then, for each pair of nodes $\mathsf{b}^i_v, \mathsf{w}^i_v$, the edge set E contains an edge e from b^i_v to w^i_v, labeled with a *pair* label $l(e) = \|\mathsf{prv}(a), cost(a)\|$. Otherwise, if $\mathsf{post}(a)[v] = \mathsf{b}_v$, then E contains similar edges for each pair of nodes $\mathsf{w}^i_v, \mathsf{b}^{i+1}_v$, *mutatis mutandis*. The prv and cost parts of $l(e)$ are denoted by $\mathsf{prv}(e)$ and $cost(e)$, respectively.

Next, the digraph $G(v)$ is expanded into a digraph $G'(v) = (V', E')$ by substituting each edge $e \in E$ with a set of edges (between the same nodes), labeled with all possible assignments of the *time-stamped* elements of $\sigma'_1, \ldots, \sigma'_k$ to $\mathsf{prv}(e)$. For example, an edge $e \in E$ labeled with $\|\mathsf{b}_{w_1}\mathsf{b}_{w_2}, 10\|$ might be substituted in E' with edges labeled with $\{\|\mathsf{b}^1_{w_1}\mathsf{b}^1_{w_2}, 10\|, \|\mathsf{b}^1_{w_1}\mathsf{b}^2_{w_2}, 10\|, \|\mathsf{b}^2_{w_1}\mathsf{b}^1_{w_2}, 10\|, \ldots\}$. Finally, we set $V' = V \cup \{s_v, t_v\}$, and add a single edge labeled with the first elements of $\sigma'_1, \ldots, \sigma'_k$ and zero cost (i.e., $\|\mathsf{b}^1_{w_1} \cdots \mathsf{b}^1_{w_k}, 0\|$) from s_v to the (original source) node b^1_v, plus a single edge labeled with the last elements of $\sigma'_1, \ldots, \sigma'_k$ and zero cost from the (original sink) node of $G(v)$ to t_v.

Informally, the digraph $G'(v)$ can be viewed as a projection of the prefixes $\sigma'_1, \ldots, \sigma'_k$ on the base digraph $G(v)$. The digraph $G'_e(v) = (V'_e, E'_e)$ is then constructed from $G'(v)$ as follows. The nodes V'_e correspond to the *edges* of $G'(v)$. The edges $(\overrightarrow{v_e, v_{e'}}) \in E'_e$ correspond to all pairs of immediately consecutive edges $e, e' \in E'$ such that, for $1 \leq i \leq k$, either $\mathsf{prv}(e)[w_i] = \mathsf{prv}(e')[w_i]$, or $\mathsf{prv}(e')[w_i]$ appears after $\mathsf{prv}(e)[w_i]$ on σ'_i. Finally, each edge $(\overrightarrow{v_e, v_{e'}}) \in E'_e$ is weighted with $cost(e')$.

The graph $G'_e(v)$ provides the last building block for the algorithm depicted in Figure 2. Given a problem $\Pi \in \mathbf{P}_b$, the algorithm compiles it into the constraint optimization problem COP_Π. It is not hard to verify from the definition of $\mathbf{P}_b$ and Eqs. 1-2 that (i) for each variable $x \in \mathcal{X}$, $|Dom(x)| = n + 1$, and (ii) the tree-width of the cost network of COP_Π is bounded by a constant. (Figure 1 illustrates the polytree causal graph of a $\Pi \in \mathbf{P}_b$, and the cost network of the corresponding COP_Π; the top-most variables and the maximal cliques in the latter correspond to the functional components of COP_Π.) Given that, COP_Π can be solved in poly-time using the well-known algorithm for constraint optimization over trees (Dechter 2003). Lemma 1 (proof omitted) addresses the correctness and complexity of this construction, and thus accomplishes the proof of Theorem 1.

```
**procedure** polytree-k-indegree($\Pi = (\mathcal{V}, A, I, G)$)
create variables $\mathcal{X}$ as in Eq. 1
create functions $\mathcal{F} = \{\varphi_v \mid v \in \mathcal{V}\}$ with scopes as in Eq. 2
for each $v \in \mathcal{V}$ do
    if $\mathsf{pred}(v) = \emptyset$ then specify $\varphi_v$ according to Eq. 3
    elseif $\mathsf{pred}(v) = \{w_1, \ldots, w_k\}$ then
        construct $G(v)$ (based on $I[v]$, $G[v]$, and $A_v$)
        for each $\{\sigma'_1 \in \trianglerighteq^*[w_1], \ldots, \sigma'_k \in \trianglerighteq^*[w_k]\}$ do
            construct $G'(v)$ (from on $G(v)$, and $\sigma'_1, \ldots, \sigma'_k$)
            construct $G'_e(v)$ (from $G'(v)$)
            for each $\sigma' \in \trianglerighteq^*[v]$ do
                $\pi$ := minimal-cost path of $|\sigma'| - 1$ edges
                    from the source node $\langle \mathsf{b}_{w_1} \cdots \mathsf{b}_{w_k} \rangle$ of $G'_e(v)$
                if returned $\pi$
                    then $\varphi_v(\sigma', \sigma'_1, \ldots, \sigma'_k)$ := $cost(\pi)$
                    else $\varphi_v(\sigma', \sigma'_1, \ldots, \sigma'_k)$ := $\infty$
set $\mathsf{COP}_\Pi$ := $(\mathcal{X}, \mathcal{F})$ with global objective $\min \sum_{\varphi \in \mathcal{F}} \varphi(\mathcal{X})$
$\overline{x}$ := solve-tree-cop($\mathsf{COP}_\Pi$)
if $\sum_{\varphi \in \mathcal{F}} \varphi(\overline{x}) = \infty$ then return failure
return extract-plan-polytree-k-indegree($\overline{x}$)
```

Figure 2: Sequentially optimal planning for $\mathbf{P}_b$.

Lemma 1 *Given a planning problem* $\Pi \in \mathbf{P}_b$, *(i)* COP_Π *can be constructed in poly-time, and (ii) given a solution* $\overline{x}$ *for* COP_Π *with* $\sum_{\varphi \in \mathcal{F}} \varphi(\overline{x}) = \alpha$, *if* $\alpha = \infty$, *then* Π *is unsolvable, and otherwise, a plan of cost* α *for* Π *can be reconstructed from* $\overline{x}$ *in poly-time.*

The algorithm for $\mathbf{P}_b$ in Figure 2 is polynomial, but is rather involved and its complexity is exponential in $poly(\mathsf{In}(CG_\Pi))$. It is quite possible that more efficient algorithms for $\mathbf{P}_b$, or, definitely, for some of its fragments can be devised. For instance, it can already be shown that, for $\mathbf{T}$ problems with uniform-cost actions, there exists a simple and efficient SO-planning algorithm, and its complexity is given in Theorem 2.

Theorem 2 *SO-planning for* $\mathbf{T}$ *with uniform-cost actions can be done in time* $\Theta(|\mathcal{V}|^2)$.

Causal Graph and Local Structure

The causal graphs provide important information about the structure of the planning problems, but obviously not all the important such information. In fact, a closer look at Definition 2 reveals that *some information used for defining causal graphs then gets hidden by this structure*. To start with an example, let us consider the multi-valued encoding of the Logistics-style problems (Helmert 2006). In these problems, each variable representing the location of a package has as its parents in *CG all* the variables representing alternative transportation means (i.e., tracks, planes, etc.), and yet, each individual action affecting the location of a package is prevailed by at most *one* such parent variable. (You cannot load/unload a package into/from more than one vehicle.) In short, even when $\mathsf{In}(CG)$ is $\Theta(n)$, the number of v's parents that actively determine applicability of an action from A_v may still be bounded by a constant. This stresses the fact that causal graph provides an *aggregative view* on variable independence, and *local structure* information is suppressed by this view.

In what follows, we consider the impact of such local structure on the complexity of SO-planning for our structural fragments UB. Given a UB problem $\Pi = (\mathcal{V}, A, I, G)$, let the *dependence bound* of Π be given by $db_\Pi = \max_{a \in A} |\{v \in \mathcal{V} \mid \mathsf{prv}(a)[v] \neq \mathsf{u}\}|$. For any problem Π, and any k, if $db_\Pi \leq k$, we say that Π satisfies the property of k*-dependence*. For any fragment $\mathbf{F}$ of UB and any $k \in \mathbb{N}$, by $\mathbf{F}(k)$ we denote the subfragment of $\mathbf{F}$ satisfying k-dependence.

Recall that the fragment $\mathbf{P}$ of UB is NP-hard even for non-optimal planning (Giménez 2006). Our main result here is *positive*—at least for the most extreme (yet, says the example above, practically useful) setting of $k = 1$, satisfying k-dependence does bring us to an island of tractability $\mathbf{P}(1)$.

Theorem 3 *SO-planning for* $\mathbf{P}(1)$ *with uniform-cost actions (and thus, planning for general* $\mathbf{P}(1)$*) are tractable.*

Similarly to the polytree-k-indegree algorithm for $\mathbf{P}_b$, our algorithm for $\mathbf{P}(1)$ exploits the idea of compiling a planning problem Π into a tractable constraint optimization problem COP_Π. However, the planning-to-COP compilation here is very much different from this for $\mathbf{P}_b$. In fact, this difference is unavoidable since the construction in polytree-k-indegree heavily relies on the fact that $\mathsf{In}(CG_\Pi) = O(1)$, and we do not have this property in $\mathbf{P}(1)$.

Here as well, we begin with providing some notation. Given a $\mathbf{P}(1)$ problem $\Pi = (\mathcal{V}, A, I, G)$, for each $v \in \mathcal{V}$, each $w \in \mathsf{pred}(v)$, and each $\alpha \in \{\mathsf{b}_v, \mathsf{w}_v\}$, $\beta \in \{\mathsf{b}_w, \mathsf{w}_w\}$, by $a_{\alpha|\beta}$ we denote the action a with $\mathsf{post}(a)[v] = \alpha$ and $\mathsf{prv}(a)[w] = \beta$. Since Π is 1-dependent, this implies that a is prevailed by the value of w only. (Note that $a_{\alpha|\beta}$ may *not* belong to the action set A of Π.)

The proof of Theorem 3 is based on a certain property of the $\mathbf{P}(1)$ problems with respect to the notion of *conservative* action sequences given by Definition 3.

Definition 3 *Let* $\Pi = (\mathcal{V}, A, I, G)$ *be an* $\mathrm{UB}(1)$ *problem instance. An action sequence* ϱ *from* A *is called* conservative *if, for each pair of actions* $a_{\alpha|\beta}, a_{\alpha|\gamma} \in \varrho$, *we have* $\beta = \gamma$, *and thus* $a_{\alpha|\beta} = a_{\alpha|\gamma}$. *That is, all the changes of each variable to a certain value are performed by the same action.*

Given that, the (possibly empty) set of all conservative plans Π *is denoted by* $\mathcal{P}^c(\Pi)$.

The property of conservatism is clearly strong, and thus, in general, solvable problems in $\mathrm{UB}(1)$ may not have conservative plans at all. Lemma 2, however, states that this is very much not the case for $\mathbf{P}(1)$.

Lemma 2 *For every solvable* $\mathbf{P}(1)$ *problem* $\Pi = (\mathcal{V}, A, I, G)$, *we have* $\mathcal{P}^c(\Pi) \neq \emptyset$. *Moreover, if* Π *is uniform-cost, then* $\mathcal{P}^c(\Pi)$ *contains at least one SO-plan.*

The impact of Lemma 2 on our construction for uniform-cost $\mathbf{P}(1)$ below is that we can now restrict our attention to conservative plans only. Given that, the constraint optimization problem $\mathsf{COP}_\Pi = (\mathcal{X}, \mathcal{F})$ for a uniform-cost problem $\Pi = (\mathcal{V}, A, I, G) \in \mathbf{P}(1)$ is specified as follows.

The variable set $\mathcal{X}$ contains a variable x_v for each planning variable $v \in \mathcal{V}$, and a variable x_v^w for each edge $(\overrightarrow{w, v}) \in CG_\Pi$. That is,

$$\begin{aligned} \mathcal{X} &= \mathcal{X}^{\mathcal{V}} \cup \mathcal{X}^{\mathcal{E}}, \quad \mathcal{X}^{\mathcal{V}} = \{x_v \mid v \in \mathcal{V}\} \\ \mathcal{X}^{\mathcal{E}} &= \{x_v^w \mid (\overrightarrow{w, v}) \in CG_\Pi\} \end{aligned} \tag{5}$$

For each variable $x_v \in \mathcal{X}^{\mathcal{V}}$, the domain $Dom(x_v)$ consists of all *valid* prefixes of $\sigma(v)$. For each variable $x_v^w \in \mathcal{X}^{\mathcal{E}}$, the domain $Dom(x_v^w)$ consists of all triples of integers $[\![\delta^w_{\mathsf{w}_v}, \delta^w_{\mathsf{b}_v}, n^w_v]\!]$ satisfying Eq. 6.

$$\begin{aligned} Dom(x_v) &= \trianglerighteq^*[v] \\ Dom(x_v^w) &= \left\{ [\![\delta^w_{\mathsf{w}_v}, \delta^w_{\mathsf{b}_v}, n^w_v]\!] \;\middle|\; \begin{array}{l} 0 \leq n^w_v \leq n \\ \delta^w_{\mathsf{w}_v}, \delta^w_{\mathsf{b}_v} \in \{0,1\} \end{array} \right\} \end{aligned} \tag{6}$$

The semantics of Eq. 6 is as follows. Let $\{w_1, \dots, w_k\}$ be an *arbitrary* fixed ordering of $\mathsf{pred}(v)$. If x_v takes the value $\sigma_v \in Dom(x_v)$, then v is forced to provide σ_v sequence of values. In turn, if $x_v^{w_i}$ takes the value $[\![\delta^{w_i}_{\mathsf{w}_v}, \delta^{w_i}_{\mathsf{b}_v}, n^{w_i}_v]\!]$, then $n^{w_i}_v$ corresponds to the number of value changes of v, $\delta^{w_i}_{\mathsf{w}_v} = 1$ ($\delta^{w_i}_{\mathsf{b}_v} = 1$) forces the parents $\{w_1, \dots, w_i\} \subseteq \mathsf{pred}(v)$ to support all the changes of v to w_v (respectively, to b_v), and $\delta^{w_i}_{\mathsf{w}_v} = 0$ ($\delta^{w_i}_{\mathsf{b}_v} = 0$) relieves the parents $\{w_1, \dots, w_i\}$ from this responsibility.

$$\varphi(\llbracket \delta^w_{\mathsf{w}_v}, \delta^w_{\mathsf{b}_v}, n^w_v \rrbracket, \sigma_w) = \begin{cases} 0, & \delta^w_{\mathsf{w}_v} = 0, \delta^w_{\mathsf{b}_v} = 0, \\ 0, & \delta^w_{\mathsf{w}_v} = 1, \delta^w_{\mathsf{b}_v} = 0, (a_{\mathsf{w}_v|\mathsf{b}_w} \in A_v) \vee ((|\sigma_w| > 1) \wedge (a_{\mathsf{w}_v|\mathsf{w}_w} \in A_v)), \\ 0, & \delta^w_{\mathsf{w}_v} = 0, \delta^w_{\mathsf{b}_v} = 1, (a_{\mathsf{b}_v|\mathsf{b}_w} \in A_v) \vee ((|\sigma_w| > 1) \wedge (a_{\mathsf{b}_v|\mathsf{w}_w} \in A_v)), \\ 0, & \delta^w_{\mathsf{w}_v} = 1, \delta^w_{\mathsf{b}_v} = 1, (a_{\mathsf{w}_v|\mathsf{b}_w}, a_{\mathsf{b}_v|\mathsf{b}_w} \in A_v) \vee ((|\sigma_w| > 1) \wedge (a_{\mathsf{w}_v|\mathsf{w}_w}, a_{\mathsf{b}_v|\mathsf{w}_w} \in A_v)), \\ 0, & \delta^w_{\mathsf{w}_v} = 1, \delta^w_{\mathsf{b}_v} = 1, |\sigma_w| \geq n^w_v, a_{\mathsf{w}_v|\mathsf{b}_w}, a_{\mathsf{b}_v|\mathsf{w}_w} \in A_v, \\ 0, & \delta^w_{\mathsf{w}_v} = 1, \delta^w_{\mathsf{b}_v} = 1, |\sigma_w| > n^w_v, a_{\mathsf{w}_v|\mathsf{w}_w}, a_{\mathsf{b}_v|\mathsf{b}_w} \in A_v, \\ \infty, & \text{otherwise} \end{cases} \tag{4}$$

For each variable $x \in \mathcal{X}$, the set $\mathcal{F}$ contains a non-negative, real-valued function φ_x with the scope

$$Q_x = \begin{cases} \{x_v\}, & x = x_v, k = 0 \\ \{x_v, x_v^{w_k}\}, & x = x_v, k > 0 \\ \{x_v^{w_1}, x_{w_1}\}, & x = x_v^{w_1}, k > 0 \\ \{x_v^{w_j}, x_v^{w_{j-1}}, x_{w_j}\}, & x = x_v^{w_j}, 1 < j \leq k \end{cases} \tag{7}$$

where $\mathsf{pred}(v) = \{w_1, \ldots, w_k\}$ (and $k = 0$ means $\mathsf{pred}(v) = \emptyset$). Proceeding now with specifying the functional components $\mathcal{F}$ of COP_Π, first, for each x_v with $\mathsf{pred}(v) = \emptyset$, and for each $\sigma_v \in \trianglerighteq^*[v]$, we set $\varphi_{x_v}(\sigma_v)$ to

$$\varphi_{x_v}(\sigma_v) = \begin{cases} 0, & |\sigma_v| = 1, \\ 1, & (|\sigma_v| = 2) \wedge (a_{\mathsf{w}_v} \in A_v), \\ |\sigma_v| - 1, & (|\sigma_v| > 2) \wedge (a_{\mathsf{w}_v}, a_{\mathsf{b}_v} \in A_v), \\ \infty, & \text{otherwise} \end{cases} \tag{8}$$

In turn, for each planning variable $v \in \mathcal{V}$ with $\mathsf{pred}(v) = \{w_1, \ldots, w_k\}$, $k > 0$, the function φ_{x_v} is set to

$$\varphi_{x_v}(\sigma_v, \llbracket \delta^{w_k}_{\mathsf{w}_v}, \delta^{w_k}_{\mathsf{b}_v}, n^{w_k}_v \rrbracket) = \begin{cases} 0, & (|\sigma_v| = 1) \wedge (\llbracket \delta^{w_k}_{\mathsf{w}_v}, \delta^{w_k}_{\mathsf{b}_v}, n^{w_k}_v \rrbracket = \llbracket 0, 0, 0 \rrbracket), \\ 1, & (|\sigma_v| = 2) \wedge (\llbracket \delta^{w_k}_{\mathsf{w}_v}, \delta^{w_k}_{\mathsf{b}_v}, n^{w_k}_v \rrbracket = \llbracket 1, 0, 1 \rrbracket), \\ |\sigma_v| - 1, & (|\sigma_v| > 2) \wedge (\llbracket \delta^{w_k}_{\mathsf{w}_v}, \delta^{w_k}_{\mathsf{b}_v}, n^{w_k}_v \rrbracket = \llbracket 1, 1, |\sigma_v| - 1 \rrbracket), \\ \infty, & \text{otherwise} \end{cases} \tag{9}$$

The functions φ_{x_v} capture the, *marginal over the actions* A_v, cost of providing a sequence σ_v of value changes of v in Π, given that (in case of Eq. 9) the parents of v are "ready to support these value changes".

In specifying the remaining functional components we use an "indicator" function φ specified in Eq. 4. The semantics of φ is that, for each $v \in \mathcal{V}$, each $w \in \mathsf{pred}(v)$, and each $(\llbracket \delta^w_{\mathsf{w}_v}, \delta^w_{\mathsf{b}_v}, n^w_v \rrbracket, \sigma_w) \in Dom(x^w_v) \times Dom(x_w)$, we have $\varphi(\llbracket \delta^w_{\mathsf{w}_v}, \delta^w_{\mathsf{b}_v}, n^w_v \rrbracket, \sigma_w) = 0$ if the value sequence σ_w of w can support *all* the changes of v to w_v (if $\delta^w_{\mathsf{w}_v}$ = 1) and *all* the changes of v to b_v (if $\delta^w_{\mathsf{b}_v}$ = 1), out of n^w_v value changes of v in Π.

Given the indicator function φ, for each $v \in \mathcal{V}$, the functional component $\varphi_{x_v^{w_1}}$ is specified as

$$\varphi_{x_v^{w_1}}(\llbracket \delta^{w_1}_{\mathsf{w}_v}, \delta^{w_1}_{\mathsf{b}_v}, n^{w_1}_v \rrbracket, \sigma_{w_1}) = \varphi(\llbracket \delta^{w_1}_{\mathsf{w}_v}, \delta^{w_1}_{\mathsf{b}_v}, n^{w_1}_v \rrbracket, \sigma_{w_1}), \tag{10}$$

and the rest of the functional components $\varphi_{x_v^{w_2}}, \ldots, \varphi_{x_v^{w_k}}$ are specified as

$$\varphi_{x_v^{w_j}}(\llbracket \delta^{w_j}_{\mathsf{w}_v}, \delta^{w_j}_{\mathsf{b}_v}, n^{w_j}_v \rrbracket, \llbracket \delta^{w_{j-1}}_{\mathsf{w}_v}, \delta^{w_{j-1}}_{\mathsf{b}_v}, n^{w_{j-1}}_v \rrbracket, \sigma_{w_j}) = \begin{cases} \varphi(\llbracket \delta^{w_j}_{\mathsf{w}_v} - \delta^{w_{j-1}}_{\mathsf{w}_v}, \delta^{w_j}_{\mathsf{b}_v} - \delta^{w_{j-1}}_{\mathsf{b}_v}, n^{w_j}_v \rrbracket, \sigma_{w_j}), & \begin{array}{l} n^{w_j}_v = n^{w_{j-1}}_v, \\ \delta^{w_j}_{\mathsf{w}_v} \geq \delta^{w_{j-1}}_{\mathsf{w}_v}, \\ \delta^{w_j}_{\mathsf{b}_v} \geq \delta^{w_{j-1}}_{\mathsf{b}_v}, \end{array} \\ \infty & \text{otherwise} \end{cases} \tag{11}$$

This finalized the construction of COP_Π, and this construction constitutes the first two steps of the algorithm $\mathsf{polytree\text{-}1\text{-}dep}$ in Figure 3(a). The subsequent steps of this algorithm are conceptually similar to these of the $\mathsf{polytree\text{-}k\text{-}indegree}$ algorithm, with the major difference being in the plan reconstruction routines (omitted here). It is not hard to verify from Eqs. 5-7, and the fact that the causal graph of $\Pi \in \mathbf{P}(1)$ forms a polytree that (i) for each variable $x \in \mathcal{X}$, $|Dom(x)| = poly(n)$, and (ii) the tree-width of the cost network of $\mathcal{F}$ is ≤ 3. Figure 3(b-c) depicts the causal graph of a problem $\Pi \in \mathbf{P}(1)$, and the cost network of the corresponding COP_Π; here as well, the top-most variables and the cliques in the cost network correspond to the functional components of COP_Π. Lemma 3 (proof omitted) provides the central part of the proof of Theorem 3.

Lemma 3 *Given a planning problem* $\Pi \in \mathbf{P}(1)$ *with uniform-cost actions, (i)* COP_Π *can be constructed in poly-time, and (ii) given a solution* $\overline{x}$ *for* COP_Π *with* $\sum_{\varphi \in \mathcal{F}} \varphi(\overline{x}) = \alpha$*, if* $\alpha = \infty$*, then* Π *is unsolvable, and otherwise, a plan of cost* α *for* Π *can be reconstructed from* $\overline{x}$ *in poly-time.*

Finally, one may ask whether 1-dependence is not a strong enough property to make the SO-planning tractable even for some more complex than polytree forms of the causal graph. Theorem 4, however, provides a negative answer to this question.

Theorem 4 *Sequentially optimal planning for* $\mathbf{S}^b_b(1)$ *with uniform-cost actions is NP-complete.*

```
procedure polytree-1-dep($\Pi = (\mathcal{V}, A, I, G)$)
create variables $\mathcal{X}$ as in Eqs. 5-6
create functions $\mathcal{F} = \{\varphi_x \mid x \in \mathcal{X}\}$ with scopes as in Eq. 7
for each $x \in \mathcal{X}$ do set $\varphi_x$ according to Eqs. 8-11
set $\mathsf{COP}_\Pi := (\mathcal{X}, \mathcal{F})$ with global objective $\min \sum_{\varphi \in \mathcal{F}} \varphi(\mathcal{X})$
$\overline{x}$ := solve-tree-cop($\mathsf{COP}_\Pi$)
if $\sum_{\varphi \in \mathcal{F}} \varphi(\overline{x}) = \infty$ then return failure
return extract-plan-polytree-1-dep($\overline{x}$)
```

(a)

$$\begin{tikzcd}
v_1 \arrow[dr] & v_2 \arrow[d] & v_3 \arrow[dl] \\
 & v_4 \arrow[dl] \arrow[dr] & v_5 \arrow[d] \\
v_7 & & v_6
\end{tikzcd}$$

(b)

$$\begin{tikzcd}
x_{v_1} \arrow[d, dash] & x_{v_2} \arrow[d, dash] \arrow[dl, dash] & x_{v_3} \arrow[d, dash] \arrow[dl, dash] & \\
x^{v_1}_{v_4} \arrow[r, dash] & x^{v_2}_{v_4} \arrow[r, dash] & x^{v_3}_{v_4} \arrow[d, dash] & \\
 & & x_{v_4} \arrow[dl, dash] \arrow[d, dash] & x_{v_5} \arrow[dl, dash] \arrow[d, dash] \\
 & x^{v_4}_{v_7} \arrow[d, dash] & x^{v_4}_{v_6} \arrow[r, dash] & x^{v_5}_{v_6} \arrow[d, dash] \\
 & x_{v_7} & & x_{v_6}
\end{tikzcd}$$

(c)

Figure 3: (a) SO-planning for $\mathbf{P}(1)$; (b) Causal graph of a planning problem $\Pi \in \mathbf{P}(1)$, and (c) the cost network of COP_Π (right).

The proof of Theorem 4 is by a polynomial reduction from the well-known Minimal Vertex Cover problem.

From PDBs to Structural Patterns?

The tractability results in general are of theoretical interest on their own because they shed light on what makes and what does not make problems hard for planning and for plan optimization. However, we also believe that structurally-characterized tractable fragments of optimal planning can be operationalized in devising new admissible search heuristics. Here we discuss the prospects of this direction in the scope of extending the pattern-database heuristics (Culberson & Schaeffer 1998; Edelkamp 2001) to "structural patterns" of unlimited dimensionality.

A heuristic is admissible if it never overestimates the true cost of reaching the nearest goal state. Most often, an admissible heuristic for planning is derived from the optimal cost of achieving the goals in an over-approximating abstraction of the planning problem in hand. One type of abstractions, *homomorphisms*, simplify the original problem by systematically contracting groups of states to single states. Most typically, a such 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.

In planning, homomorphisms have been successfully explored in the scope of domain-independent pattern database (PDB) heuristics (Culberson & Schaeffer 1998; Edelkamp 2001; 2002). In short, given a SAS$^+$ problem $\Pi = \langle \mathcal{V}, A, I, G \rangle$, each subset of variables $\mathcal{V}' \subseteq \mathcal{V}$ defines a *pattern abstraction* $\Pi^{[\mathcal{V}']} = \langle \mathcal{V}', A^{[\mathcal{V}']}, I^{[\mathcal{V}']}, G^{[\mathcal{V}']} \rangle$ by intersecting the initial state, the goal, and all the actions' pre, prevail and post condition lists with $\mathcal{V}'$ (Edelkamp 2001). The idea behind the PDB heuristics is elegantly simple. First, we select a (relatively small) set of subsets $\mathcal{V}_1, \ldots, \mathcal{V}_k$ of $\mathcal{V}$ such that, for $1 \leq i \leq k$,

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

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

Let $h^{[\mathcal{V}_i]}(s)$ be the optimal cost of achieving the abstract goal $G^{[\mathcal{V}_i]}$ from the abstract state $s^{[\mathcal{V}_i]}$. To obtain an admissible heuristic, if the set of abstract problems $\Pi^{[\mathcal{V}_1]}, \ldots, \Pi^{[\mathcal{V}_k]}$ satisfy certain requirements of disjointness (Felner, Korf, & Hanan 2004; Edelkamp 2001), the PDB heuristic can be set to $h(s) = \sum_{i=1}^{k} h^{[\mathcal{V}_i]}(s)$. Otherwise, one can set $h(s) = \max_{i=1}^{k} h^{[\mathcal{V}_i]}(s)$ (Holte *et al.* 2006).

The Achilles heel of the PDB heuristics is that each pattern (that is, each selected subset of variables $\mathcal{V}_i$) is required to be *small* so that reachability analysis in $\Pi^{[\mathcal{V}_i]}$ could be done by exhaustive search. In short, computing $h^{[\mathcal{V}_i]}(s)$ in polynomial time requires satisfying $|\mathcal{V}_i| = O(poly(\log |\mathcal{V}|))$. Note that this constraint implies an inherent scalability limitation of the PDB heuristics—as the problems of interest grow, limiting patterns to poly-logarithmic dimensionality will unavoidably make them less and less informative with respect to the original problems.

One can notice, however, that this is not necessarily the only way to proceed. In principle, given a SAS$^+$ problem $\Pi = \langle \mathcal{V}, A, I, G \rangle$, one can select a (relatively small) set of subsets $\mathcal{V}_1, \ldots, \mathcal{V}_k$ of $\mathcal{V}$ such that, for $1 \leq i \leq k$,

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

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

What is important here is that the second requirement can in principle be satisfied even if the size of each selected pattern $\mathcal{V}_i$ is $\Theta(|\mathcal{V}|)$.

A priori, this generalization of the PDB heuristics idea to structural patterns is appealing as it allows using patterns of unlimited dimensionality. The pitfall, however, is that such structural patterns correspond exactly to tractable fragments of optimal planning, and, as we already mentioned, the palette of such known fragments is extremely limited. This obstacle on the way to structural pattern heuristics has been the principal original motivation for our work, and we believe that further research on the optimal planning tractability is required.

On the other hand, some preliminary positive evidence for realizability of the structural patterns heuristics idea is already abound (Katz & Domshlak 2007).

Summary and Future Work

We have studied the complexity of sequentially-optimal classical planning with respect to the interplay between the topology of the problem's causal graph, and certain types of local structure of the problems induced by their action sets. For some problem classes, we showed that such optimal planning is tractable, and that relaxing the specification of these classes takes us out to NP-hard classes of problem. We believe that there is a room for further extending the palette of tractable optimal planning, and we plan to continue our research in this direction. In particular, we conjecture that sequentially-optimal planning for $\mathbf{P}(1)$ problems with non-uniform action costs is tractable as well. Proving that, however, will require some additional insights into the problem.

Finally, we discussed the prospects of using structure-based tractability results in the scope of homomorphism abstractions for new admissible heuristics for general planning. We believe that this direction is very promising, and, obviously, completely open for future investigation. In particular, in addition to extending the palette of tractable optimal planning as much as possible, materializing this idea of structural pattern heuristics will require addressing numerous additional issues inherited from the PDB framework (e.g., optimizing the pattern set selection).

Acknowledgments

The authors were partially supported by the C. Wellner Research Fund.

References

Bacchus, F., and Yang, Q. 1994. Downward refinement and the efficiency of hierarchical problem solving. *AIJ* 71(1):43–100.

Bäckström, C., and Klein, I. 1991. Planning in polynomial time: The SAS-PUBS class. *Comp. Intell.* 7(3):181–197.

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.

Brafman, R. I., and Domshlak, C. 2003. Structure and complexity of planning with unary operators. *JAIR* 18:315–349.

Brafman, R. I., and Domshlak, C. 2006. Factored planning: How, when, and when not. In *AAAI*, 809–814.

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

Chapman, D. 1987. Planning for conjunctive goals. *AIJ* 32(3):333–377.

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

Dechter, R. 2003. *Constraint Processing*. Morgan Kaufmann.

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.

Edelkamp, S. 2002. Symbolic pattern databases in heuristic search planning. In *AIPS*, 274–293.

Erol, K.; Nau, D. S.; and Subrahmanian, V. S. 1995. Complexity, decidability and undecidability results for domain-independent planning. *AIJ* 76(1–2):75–88.

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

Giménez, O. 2006. Solving planning domains with polytree causal graphs is NP-complete. arXiv:cs.AI/0610095.

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

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

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

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 heuristics. In *ICAPS-07 Workshop on Heuristics for Domain-independent Planning: Progress, Ideas, Limitations, Challenges*.

Klein, I.; Jonsson, P.; and Bäckström, C. 1998. Efficient planning for a miniature assembly line. *Artificial Intelligence in Engineering* 13(1):69–81.

Knoblock, C. 1994. Automatically generating abstractions for planning. *AIJ* 68(2):243–302.

Refanidis, I., and Vlahavas, I. P. 2001. The GRT planning system: Backward heuristic construction in forward state-space planning. *JAIR* 15:115–161.

Williams, B., and Nayak, P. 1997. A reactive planner for a model-based executive. In *IJCAI*, 1178–1185.