

Optimal Additive Composition of Abstraction-based Admissible Heuristics

Technical Report IS/IE-2008-04

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

Abstract

We describe a procedure that takes a classical planning task, a forward-search state, and a set of abstraction-based admissible heuristics, and derives an *optimal* additive composition of these heuristics with respect to the given state. Most importantly, we show that this procedure is *polynomial-time* for arbitrary sets of all known to us abstraction-based heuristics such as PDBs, constrained PDBs, merge-and-shrink abstractions, fork-decomposition structural patterns, and structural patterns based on tractable constraint optimization.

1. Introduction

Admissible heuristics are critical for effective planning when either optimal or approximately-optimal solutions are required. As automated planning is known to be NP-hard even for extremely conservative problem formalisms (Bylander, 1994), no heuristic should be expected to work well in all planning tasks. Moreover, even for a fixed planning task, typically no tractable heuristic will home in on all the “combinatorics” of the task in hand. The promise, however, is that (i) different heuristics will target different bolts of the planning complexity, and (ii) composing the individual strengths of numerous heuristics could allow us both solving a larger range of planning tasks, as well as solving each individual task more efficiently.

Since the late 90’s, numerous (though not many) admissible heuristics for domain-independent planning have been suggested and found useful, and research in this direction becomes more and more active. In this paper we focus on the old question of how one should better orchestrate a set of admissible heuristics in the effort of solving a given planning task. One of the well-known and heavily-used properties of admissible heuristics is that taking the maximum of their values maximizes informativeness while preserving admissibility. A more recent, alternative approach to orchestrating a set of admissible heuristics corresponds to carefully separating the information used by the different heuristics in the set so that their values could be summed up instead of maximized over. This direction was first exploited in the works on additive pattern database (APDB) heuristics (Edelkamp, 2001; Felner, Korf, & Hanan, 2004), and more recently it was applied in the scope of constrained PDBs, m -reachability, and structural patterns heuristics (Haslum, Bonet, & Geffner, 2005; Katz & Domshlak, 2007a). The basic idea underlying all these “additive heuristic ensembles” is elegantly simple: for each problem’s action a , if it can possibly be counted by more than one heuristic in the ensemble, then one should ensure that the cumulative accounting for the cost of a does not exceed its true cost in the original problem.

Until very recently, such “action-cost partitioning” was achieved in one certain manner by accounting for the whole cost of each action in computing a single heuristic, while ignoring the cost of that action (by setting it to zero) in computing all the other heuristics in the set (Edelkamp, 2001; Felner et al., 2004; Haslum et al., 2005). Recently, this “all-in-one/nothing-in-rest” action-cost par-

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

tioning has been generalized by Katz and Domshlak (2007a) and Yang *et al.* (2008) to *arbitrary* partitioning of the action cost among the ensembles’ heuristics.

The great flexibility of additive heuristic ensembles, however, is a mixed blessing. For good and for bad, the methodology of taking the maximum over the values provided by an arbitrary set of independently constructed admissible heuristics is entirely non-parametric. In contrast, switching to additive heuristic ensembles requires *selecting an action-cost partitioning scheme*, and this decision problem poses a number of computational challenges. In particular,

1. The space of alternative choices here is verbally infinite as the cost of each action can be partitioned into an arbitrary set of non-negative real numbers, sum of which does not exceed the cost of that action.
2. At least in domain-independent planning, this decision process should be fully unsupervised.
3. The last but not least, the relative quality of each action-cost partition (in terms of the informativeness of the resulting additive heuristic) may vary dramatically between the examined search states. Hence, the choice of the action-cost partitioning scheme should ultimately be a function of the search state in question.

These issues may explain why all previous works on (both domain-dependent and independent) additive heuristic ensembles adopt this or another ad hoc (and search-state independent) choice of action-cost partitioning. As the result, all the reported empirical comparative evaluations of various max-based and additive heuristic ensembles are inconclusive—for some search states along the search process the (pre-selected) additive heuristics’ combination was dominating the max-combination, while for the other states the opposite was the case. In the context of domain-specific APDBs, Yang *et al.* (2008) conclude that “determining which abstractions [here: action-cost partitioning schemes] will produce additives that are better than max over standards is still a big research issue.”

The contribution of this paper is precisely in addressing the problem of choosing the right action-cost partitioning over a given set of heuristics. Specifically, we

- Provide a procedure that, given (i) a classical planning task Π , (ii) a forward-search state s of Π , and (iii) a set of admissible heuristics based on over-approximating abstractions of Π , derives an *optimal action-cost partitioning for s* (that is, a partitioning that maximizes the heuristic estimate of that state). The procedure is *fully unsupervised*, and is based on a linear programming formulation of that optimization problem.
- Show that the time complexity of our procedure is *polynomial* for arbitrary sets of *all* known to us abstraction-based heuristic functions. In particular, such “procedure-friendly” heuristics include PDBs (Edelkamp, 2001; Yang et al., 2008), constrained PDBs (Haslum et al., 2005), merge-and-shrink abstractions (Helmert, Haslum, & Hoffmann, 2007), fork-decomposition structural patterns (Katz & Domshlak, 2007a), and structural patterns based on tractable constraint optimization (Katz & Domshlak, 2007b).

Notice that, in particular, the estimate provided by a max-based ensemble corresponds to the estimate provided by the respective additive ensemble under *some* action-cost partitioning. As such, the max-estimate cannot exceed the one provided by the optimal action-cost partitioning, and thus, at least for the abstraction-based heuristics, we answer the aforementioned question of “to add or not to add”.

2. Background

We consider the standard setting of cost-optimal classical planning for problems described using the SAS⁺ representation language (Bäckström & Nebel, 1995). A **sas⁺ planning task** is a quintuple $\Pi = \langle V, A, I, G, cost \rangle$, where

- $V = \{v_1, \dots, v_n\}$ is a set of *state variables*, each associated with a finite domain $\text{dom}(v_i)$; each complete assignment s to V is called a *state*. I is an *initial state*, and the *goal* G is a partial assignment to V .
- A is a finite set of *actions*, where each action a is a pair $\langle \text{pre}(a), \text{eff}(a) \rangle$ of partial assignments to V called *preconditions* and *effects* of a , respectively, and $\text{cost} : A \rightarrow \mathbb{R}^{0+}$ is a non-negative real-valued *action cost function*.

The semantics of a planning task Π is given by its induced state-transition model, often also called *transition graph*. Searching in this transition graph corresponds to forward state-space search. In what follows we distinguish between the actual edge-weighted transition graph, and its weights-omitted, qualitative skeleton which we call *transition-graph structure*. Informally, transition-graph structures capture the combinatorics of the classical planning problems, while transition graphs annotate this combinatorics with “performance measures”.

- A **transition-graph structure** (or **TG-structure**, for short) is a quintuple $\mathcal{T} = (S, L, Tr, s^I, S^G)$ where S is a finite set of *states*, L is a finite set of *transition labels*, $Tr \subseteq S \times L \times S$ is a set of (labeled) *transitions*, $s^I \in S$ is an *initial state*, and $S^G \subseteq S$ is a set of *goal states*.
- A **transition graph** is a pair $\langle \mathcal{T}, \varpi \rangle$ where \mathcal{T} is a TG-structure with labels L , and $\varpi : L \rightarrow \mathbb{R}^{0+}$ is a *transition cost function*. For a state $s \in S$ and a subset of states $S' \subseteq S$ in \mathcal{T} , the **distance** $\text{dist}(s, S')$ in $\langle \mathcal{T}, \varpi \rangle$ is the cost of a cheapest (with respect to ϖ) path from s to a state in S' along the transitions of \mathcal{T} . Any path from s^I to S^G is a **plan** for $\langle \mathcal{T}, \varpi \rangle$, and cheapest such paths are called **optimal plans**.

The states of the TG-structure $\mathcal{T}(\Pi)$ induced by a SAS⁺ planning task $\Pi = \langle V, A, I, G, \text{cost} \rangle$ are the states of Π , the transition labels of $\mathcal{T}(\Pi)$ are the actions A , and $(s, a, s') \in Tr$ iff (i) $s[v] = \text{pre}(a)[v]$ whenever $\text{pre}(a)[v]$ is specified and (ii) $s'[v] = \text{eff}(a)[v]$ if $\text{eff}(a)[v]$ is specified, and otherwise $s'[v] = s[v]$. The actual transition graph induced by Π is $\langle \mathcal{T}(\Pi), \text{cost} \rangle$.

3. Additive Admissible Heuristics

Our focus here is on additive ensembles of admissible heuristics, or simply, *additive heuristics*. Very often, each individual admissible heuristic for domain-independent planning is defined as the optimal cost of achieving the goals in an over-approximating *abstraction* of the planning problem in hand¹ (Pearl, 1984). Such an abstraction is obtained by relaxing some constraints present in the problem, and the desire is to obtain a tractable (that is, solvable in polynomial time), yet informative abstract problem. In turn, by *additive abstraction* we refer to a set of abstractions, inter-constrained by a requirement to jointly not over-estimate the transition path costs of the original problem.

Definition 1 *An additive abstraction of a transition graph $\langle \mathcal{T}, \varpi \rangle$ is a set of pairs $\mathcal{A} = \{ \langle \langle \mathcal{T}_i, \varpi_i \rangle, \alpha_i \rangle \}_{i=1}^k$ where, for $1 \leq i \leq k$, $\langle \mathcal{T}_i, \varpi_i \rangle$ is a transition graph with structure $\mathcal{T}_i = (S_i, L_i, Tr_i, s_i^I, S_i^G)$, $\alpha_i : S \rightarrow S_i$ is a function called abstraction mapping, $\alpha_i(s^I) = s_i^I$, $\alpha_i(s) \in S_i^G$ for all $s \in S^G$, and, for each pair of states $s, s' \in S$, holds*

$$\sum_{i=1}^k \text{dist}(\alpha_i(s), \alpha_i(s')) \leq \text{dist}(s, s'). \quad (1)$$

For $k = 1$, Definition 1 formalizes standard, non-additive abstractions, while for $k \geq 1$ it poses only a general requirement of not overestimating the distances. For our objective of dealing with

1. Admissible heuristics may also correspond to problem reformulations that are not abstractions; see our discussion later on.

action-cost partitioning, we need a tighter binding between the original and abstract TG-structures. Specifically, we need to (i) associate each abstract transition label with a single original transition label, and (ii) verify that each original transition corresponds to an appropriate set of abstract transition *paths*.

Definition 2 *An ensemble of abstractions (ABS-ensemble) of a TG-structure $\mathcal{T} = (S, L, Tr, s^I, S^G)$ is a set of triplets $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ where, for $1 \leq i \leq k$,*

- $\mathcal{T}_i = (S_i, L_i, Tr_i, s_i^I, S_i^G)$ is a TG-structure,
- $\alpha_i : S \rightarrow S_i$ is an abstraction mapping, and $\beta_i : L_i \rightarrow L$ is an action-associating mapping,
- $\alpha_i(s^I) = s_i^I$, and $\alpha_i(s) \in S_i^G$ for all $s \in S^G$,
- for each transition $\langle s, l, s' \rangle \in Tr$, there is a path ρ from $\alpha_i(s)$ to $\alpha_i(s')$ in \mathcal{T}_i such that (i) all the transitions on ρ have different labels, and (ii) for each label l' along ρ , holds $\beta_i(l') = l$.

The notion of ABS-ensembles allows us generalizing the qualitative skeletons of various additive abstractions that are based on action-cost partitioning. In particular,

- The setting of all β_i being bijective mappings captures additive *homomorphism abstractions* such as (standard and constrained) APDBs (Yang et al., 2008; Haslum et al., 2005), as well as (additive) merge-and-shrink abstractions (Helmert et al., 2007).
- The (possibly confusing at first view) setting of all α_i being injective (with, possibly, $|S| < |S_i|$) captures additive *embedding abstractions*, obtained by expanding the original action set by some new actions derived from the original ones. In such cases, the new actions are constructed with certain desired properties such as positive and/or unary effects only (Bylander, 1994), etc.
- Each individual abstraction in an ABS-ensemble may correspond to a *hybrid (homomorphism/embedding) abstraction* such as these induced by some structural patterns (Katz & Domshlak, 2007a).
- Importantly, nothing in the definition of ABS-ensemble prevents us from using an *arbitrary mixture* of the above three types of abstractions.

First things first, however, Theorem 1 relates ABS-ensembles and additive abstractions induced by the former via action-cost partitioning (expressed by Eq. 2). Worth noting here that the generality of Definition 2 and Theorem 1 is not for an exercise only—later we exploit it to establish some computational results of an adequate generality.

Theorem 1 (Additive Abstractions) *Let \mathcal{T} be a TG-structure with labels L , and let $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ be an ABS-ensemble of \mathcal{T} . For any function $\varpi : L \rightarrow \mathbb{R}^{0+}$, and any set of functions $\varpi_i : L_i \rightarrow \mathbb{R}^{0+}$, $1 \leq i \leq k$, such that*

$$\forall l \in L : \sum_{i=1}^k \sum_{l' \in \beta_i^{-1}(l)} \varpi_i(l') \leq \varpi(l), \quad (2)$$

we have $\mathcal{A} = \{\langle \mathcal{T}_i, \varpi_i, \alpha_i \rangle\}_{i=1}^k$ being an additive abstraction of the transition graph $\langle \mathcal{T}, \varpi \rangle$.

Proof: Let $\mathcal{T} = (S, L, Tr, s^I, S^G)$ be a TG-structure, $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ be an ABS-ensemble of \mathcal{T} , and $\varpi : L \rightarrow \mathbb{R}^{0+}$, $\varpi_i : L_i \rightarrow \mathbb{R}^{0+}$, $1 \leq i \leq k$ be some functions satisfying Eq. 2. To prove that $\mathcal{A} = \{\langle \mathcal{T}_i, \varpi_i, \alpha_i \rangle\}_{i=1}^k$ is an additive abstraction of the transition graph $\langle \mathcal{T}, \varpi \rangle$ we should show that Eq. 1 holds for each pair of states $s, s' \in S$.

Let $\rho = \langle s_0, l_1, s_1 \rangle, \langle s_1, l_2, s_2 \rangle, \dots, \langle s_{m-1}, l_m, s_m \rangle$ be a cheapest path from $s = s_0$ to $s' = s_m$ in $\langle \mathcal{T}, \varpi \rangle$. Thus, $\text{dist}(s, s') = \sum_{j=1}^m \varpi(l_j)$. For $1 \leq i \leq k$, let $\rho_{i,j}$ be a simple, label-disjoint path from $\alpha_i(s_{j-1})$ to $\alpha_i(s_j)$ in \mathcal{T}_i along transitions labeled only with labels in $\beta_i^{-1}(l_j)$; the existence of such $\rho_{i,j}$ is guaranteed by Definition 2. Thus, for $1 \leq i \leq k$, we have the concatenation $\rho_i = \rho_{i,1} \dots \rho_{i,m}$ being a transition path from $\alpha_i(s)$ to $\alpha_i(s')$ in \mathcal{T}_i .

Given that, we have

$$\text{dist}(\alpha_i(s), \alpha_i(s')) \leq \sum_{l' \in \rho_i} \varpi_i(l') = \sum_{j=1}^m \sum_{l' \in \rho_{i,j}} \varpi_i(l') \leq \sum_{j=1}^m \sum_{l' \in \beta_i^{-1}(l_j)} \varpi_i(l')$$

and, summing over all $1 \leq i \leq k$, we have

$$\sum_{i=1}^k \text{dist}(\alpha_i(s), \alpha_i(s')) \leq \sum_{i=1}^k \sum_{j=1}^m \sum_{l' \in \beta_i^{-1}(l_j)} \varpi_i(l') = \sum_{j=1}^m \sum_{i=1}^k \sum_{l' \in \beta_i^{-1}(l_j)} \varpi_i(l') \stackrel{(*)}{\leq} \sum_{j=1}^m \varpi(l_j) = \text{dist}(s, s'),$$

where inequality (*) follows from Eq. 2 and label-disjointness. \blacksquare

In what follows, if $\Pi = \langle V, A, I, G, \text{cost} \rangle$ is a SAS⁺ planning task, $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ is an ABS-ensemble of $\mathcal{T}(\Pi)$, and $\{\varpi_i : L_i \rightarrow \mathbb{R}^{0+}\}_{i=1}^k$ is a set of transition cost functions, we say that $\mathcal{A} = \{\langle \mathcal{T}_i, \varpi_i \rangle, \alpha_i\}_{i=1}^k$ is an additive abstraction of Π with respect to \mathcal{AE} , denoted by $\mathcal{A} \in_{\Pi} \mathcal{AE}$, if

$$\forall a \in A : \sum_{i=1}^k \sum_{l \in \beta_i^{-1}(a)} \varpi_i(l) \leq \text{cost}(a) \quad (3)$$

In other words, each additive abstraction $\mathcal{A} \in_{\Pi} \mathcal{AE}$ corresponds to a certain action-cost partitioning of Π over \mathcal{AE} and, importantly, vice versa.

Definition 3 Let $\Pi = \langle V, A, I, G, \text{cost} \rangle$ be a SAS⁺ planning task with state set S , and $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ be an ABS-ensemble of $\mathcal{T}(\Pi)$. For any additive abstraction $\mathcal{A} = \{\langle \mathcal{T}_i, \varpi_i \rangle, \alpha_i\}_{i=1}^k \in_{\Pi} \mathcal{AE}$, the **additive heuristic** $h_{\mathcal{A}}$ is the function assigning to each state $s \in S$ the quantity $\sum_{i=1}^k \text{dist}(\alpha_i(s), S_i^G)$. The **optimal additive heuristic** $h_{\mathcal{AE}}$ is the function assigning to each state $s \in S$ the quantity $\max_{\mathcal{A} \in_{\Pi} \mathcal{AE}} h_{\mathcal{A}}(s)$.

Definition 3 specifies the set of *all* additive heuristics for Π obtainable via action-cost partitioning over a given ABS-ensemble. Most importantly, it also specifies an *upper-bound* on the heuristic estimate $h_{\mathcal{AE}}(s)$ that can possibly be obtained for a state s on the basis of that (infinite) set of additive heuristics. The admissibility of each heuristic $h_{\mathcal{A}}$ in that space is immediate from Definition 1, and thus the proof of Theorem 2 is straightforward.

Theorem 2 (Optimal Admissibility) For any SAS⁺ planning task $\Pi = \langle V, A, I, G, \text{cost} \rangle$, any ABS-ensemble \mathcal{AE} of $\mathcal{T}(\Pi)$, and any state s of Π , we have $h_{\mathcal{AE}}(s) \leq \text{dist}(s, S^G)$.

Proof: Let $\Pi = \langle V, A, I, G, \text{cost} \rangle$ be a SAS⁺ planning task, $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ be an ABS-ensemble of $\mathcal{T}(\Pi)$, and s be some state of Π . From Definitions 1 and 3, for any additive abstraction $\mathcal{A} \in_{\Pi} \mathcal{AE}$, we have

$$h_{\mathcal{A}}(s) = \sum_{i=1}^k \text{dist}(\alpha_i(s), S_i^G) \leq \text{dist}(s, S^G) = h^*(s),$$

and thus in particular that holds for \mathcal{A} that maximizes $h_{\mathcal{A}}(s)$ over all $\mathcal{A}' \in_{\Pi} \mathcal{AE}$, that is, for $h_{\mathcal{AE}}(s)$. \blacksquare

4. LP-Optimization

Having specified the notion of optimal additive heuristic $h_{\mathcal{AE}}$, we now proceed with the computational part of the story. Suppose we are given a SAS⁺ planning task $\Pi = \langle V, A, I, G, cost \rangle$ with state set S , and an ABS-ensemble $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ of $\mathcal{T}(\Pi)$. Assuming that individual additive heuristics $h_{\mathcal{A}}$ can be efficiently computed for all $\mathcal{A} \in_{\Pi} \mathcal{AE}$ (as it is the case with the ABS-ensembles of practical interest), the big question now is whether $h_{\mathcal{AE}}$ can be efficiently computed. Here we characterize a family of ABS-ensembles for which the answer to this question is affirmative, and show that this family comprises all the abstraction-based heuristic schemes suggested so far for domain-independent classical planning.

Our characterization is constructive in terms of *computability of $h_{\mathcal{AE}}(s)$ via a compact linear program induced by the triplet of Π , \mathcal{AE} , and s* . (For the sake of readability, we use “compact” as a synonym of “being of size $O(\text{poly}(|\Pi|))$ ”). We begin with introducing a set of linear constraints specifying all possible cost partitions of the actions $a \in A$ over their representatives $\cup_{i=1}^k \beta_i^{-1}(a)$ in the components of \mathcal{AE} . For each (abstract) label $l \in \cup_{i=1}^k L_i$, let w_l be a non-negative real-valued variable uniquely associated with l , and let the set of all these “label-cost” variables being denoted by \vec{w} . The (linear) *additivity constraint $\mathbb{C}^{\text{add}}(\vec{w})$ of Π on \mathcal{AE}* mirrors Eq. 3 as

$$\forall a \in A : \sum_{i=1}^k \sum_{l \in \beta_i^{-1}(a)} w_l \leq cost(a), \quad (4)$$

with \mathcal{H}^{add} being the convex polyhedron specified by $\mathbb{C}^{\text{add}}(\vec{w})$. Note that there is a straightforward bijective correspondence between the points $\mathbf{w} \in \mathcal{H}^{\text{add}}$ and (with some abuse of notation) the additive abstractions $\mathcal{A}_{\mathbf{w}} = \{\langle \langle \mathcal{T}_i, \mathbf{w} \rangle, \alpha_i \rangle\}_{i=1}^k \in_{\Pi} \mathcal{AE}$.

Using the notion of additivity constraint $\mathbb{C}^{\text{add}}(\vec{w})$, we now proceed with characterizing our “ $h_{\mathcal{AE}}$ -friendly” family of *LP-optimizable* ABS-ensembles.

Definition 4 *Let $\Pi = \langle V, A, I, G, cost \rangle$ be a SAS⁺ planning task, $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ be an ABS-ensemble of $\mathcal{T}(\Pi)$, and $\mathbb{C}^{\text{add}}(\vec{w})$ be the additivity constraint of Π on \mathcal{AE} .*

1. *Given a state s of Π , an **LP-encoding** of \mathcal{AE} with respect to s is a triplet $\mathbb{L}(s) = \langle \vec{x}, f(\vec{x}), \mathbb{C}^{\mathcal{AE}}(\vec{x}, \vec{w}) \rangle$ where \vec{x} is a set of non-negative real-valued variables, f is a real-valued affine function over \vec{x} , $\mathbb{C}^{\mathcal{AE}}$ is a set of linear constraints on \vec{x} and \vec{w} , and*

$$\forall \mathbf{w} \in \mathcal{H}^{\text{add}} : \max_{\mathbf{x} \in \mathcal{H}^{\mathcal{AE}}} f(\mathbf{x}) = h_{\mathcal{A}_{\mathbf{w}}}(s), \quad (5)$$

where $\mathcal{H}^{\mathcal{AE}}$ is the convex polyhedron specified by $\mathbb{C}^{\mathcal{AE}} \cup \mathbb{C}^{\text{add}}$.

2. *The ABS-ensemble \mathcal{AE} is called **LP-optimizable** if, for every state s of Π , there exists (and one can generate in poly-time) a compact LP-encoding $\mathbb{L}(s) = \langle \vec{x}, f(\vec{x}), \mathbb{C}^{\mathcal{AE}}(\vec{x}, \vec{w}) \rangle$ of \mathcal{AE} with respect to s .*

The next two theorems provide the two cornerstones of characterizing our “ $h_{\mathcal{AE}}$ -friendly” family of ABS-ensembles on the ground of their LP-optimizability.

Theorem 3 (Tractability) *Given a SAS⁺ planning task Π , and an ABS-ensemble \mathcal{AE} of $\mathcal{T}(\Pi)$, if \mathcal{AE} is LP-optimizable, then $h_{\mathcal{AE}}(s)$ is poly-time computable for every state s of Π .*

Proof: Let $\Pi = \langle V, A, I, G, cost \rangle$ be a SAS⁺ planning task, $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ be an LP-optimizable ABS-ensemble of $\mathcal{T}(\Pi)$, and s be a state of Π . Let $\mathbb{L}(s) = \langle \vec{x}, f(\vec{x}), \mathbb{C}^{\mathcal{AE}}(\vec{x}, \vec{w}) \rangle$ be an LP-encoding of \mathcal{AE} with respect to s (as in Definition 4). Consider now the linear program \mathcal{L} defined

by the variables \vec{xw} , constraints $\mathbb{C}^{\mathcal{AE}} \cup \mathbb{C}^{\text{add}}$, and the maximized objective function f . If \mathbf{xw} is a solution for \mathcal{L} , then

$$f(\mathbf{x}) = \max_{\mathbf{w} \in \mathcal{H}^{\text{add}}} \max_{\mathbf{x}' \mathbf{w} \in \mathcal{H}^{\mathcal{AE}}} f(\mathbf{x}') \stackrel{(5)}{=} \max_{\mathbf{w} \in \mathcal{H}^{\text{add}}} h_{\mathcal{A}}(s) = \max_{\mathcal{A} \in \Pi_{\mathcal{AE}}} h_{\mathcal{A}}(s) = h_{\mathcal{AE}}(s) \quad (6)$$

Moreover, by Definition 4, both the number of variables and constraints in \mathcal{L} is $O(\text{poly}(|\Pi|))$. Thus, the claim follows from Eq. 6, and poly-time solvability of linear programming (Schrijver, 1998). ■

Theorem 4 (Composition) *For any SAS⁺ planning task Π , and any set of LP-optimizable ABS-ensembles $\{\mathcal{AE}_1, \dots, \mathcal{AE}_m\}$ of $\mathcal{T}(\Pi)$, if $m = O(\text{poly}(|\Pi|))$, then the “composite” ABS-ensemble $\mathcal{AE} = \cup_{i=1}^m \mathcal{AE}_i$ of $\mathcal{T}(\Pi)$ is also LP-optimizable.*

Proof: Let Π and $\{\mathcal{AE}_1, \dots, \mathcal{AE}_m\}$ be as in the claim, and let s be some state of Π . For $1 \leq j \leq m$, let $\mathbb{L}_j(s) = \langle \vec{x}_j, f_j(\vec{x}_j), \mathbb{C}_j^{\mathcal{AE}}(\vec{x}_j \vec{w}_j) \rangle$ be a (compact) LP-encoding of the ABS-ensemble \mathcal{AE}_j .

First, we specify a composite additivity constraint $\mathbb{C}^{\text{add}}(\vec{w})$ over $\vec{w} = \cup_{j=1}^m \vec{w}_j$ as

$$\forall a \in A : \sum_{j=1}^m \sum_{i=1}^{k_j} \sum_{l \in \beta_{j,i}^{-1}(a)} w_l \leq \text{cost}(a).$$

Given that, the LP-encoding $\mathbb{L}(s) = \langle \vec{x}, f(\vec{x}), \mathbb{C}^{\mathcal{AE}}(\vec{xw}) \rangle$ of $\mathcal{AE} = \cup_{j=1}^m \mathcal{AE}_j$ is set to

- $\vec{x} = \cup_{j=1}^m \vec{x}_j$
- $f(\vec{x}) = \sum_{j=1}^m f_j(\vec{x}_j)$
- $\mathbb{C}^{\mathcal{AE}}(\vec{xw}) = \cup_{j=1}^m \mathbb{C}_j^{\mathcal{AE}}(\vec{x}_j \vec{w}_j)$

For any point $\mathbf{w} = (\mathbf{w}_1, \dots, \mathbf{w}_m) \in \mathcal{H}^{\text{add}}$, and any assignment $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_m)$ to \vec{x} , if $\mathbf{xw} \in \mathcal{H}^{\mathcal{AE}}$, then we have $\mathbf{x}_j \mathbf{w}_j \in \mathcal{H}^{\mathcal{AE}_j}$ for all $1 \leq j \leq m$. Therefore, we have

$$\forall \mathbf{w} \in \mathcal{H}^{\text{add}} : \max_{\mathbf{xw} \in \mathcal{H}^{\mathcal{AE}}} f(\mathbf{x}) = \max_{\mathbf{xw} \in \mathcal{H}^{\mathcal{AE}}} \sum_{j=1}^m f_j(\mathbf{x}_j) \stackrel{(*)}{=} \sum_{j=1}^m \max_{\mathbf{x}'_j \mathbf{w}_j \in \mathcal{H}^{\mathcal{AE}_j}} f_j(\mathbf{x}'_j) = \sum_{j=1}^m h_{\mathcal{AE}_j}(\mathbf{w}_j) = h_{\mathcal{AE}}(\mathbf{w}),$$

where the equality (*) stems from that, having fixed the value of variables \vec{w} , optimizations of the individual objective functions f_1, \dots, f_m are pair-wise independent due to the pair-wise disjointness of the variables $\vec{x}_1, \dots, \vec{x}_m$. In other words, for each point $\mathbf{w} = (\mathbf{w}_1, \dots, \mathbf{w}_m) \in \mathcal{H}^{\text{add}}$ on the convex polyhedron specified by $\mathbb{C}^{\text{add}}(\vec{w})$, we have $\mathbf{xw} \in \mathcal{H}^{\mathcal{AE}}$ if and *only if* $\mathbf{x}_j \mathbf{w}_j \in \mathcal{H}^{\mathcal{AE}_j}$ for all $1 \leq j \leq m$. Finally, it is immediate that both $|\vec{xw}|$ and $|\mathbb{C}^{\mathcal{AE}} \cup \mathbb{C}^{\text{add}}|$ are $O(\text{poly}(|\Pi|))$, and thus \mathcal{AE} is LP-optimizable. ■

With these two gratifying properties of LP-optimizable ABS-ensembles in hand, in what follows we check whether any of the known families of abstraction-based heuristics actually leads to such LP-optimizable ABS-ensembles. The answer to this question turns out to be positive to a surprising extent.

5. Explicit Homomorphism Abstractions

Probably the most well-known family of additive abstractions corresponds to *additive pattern databases (APDBs)*. The idea behind various PDB heuristics is elegantly simple, and goes back to the seminal paper of Culberson and Schaeffer (1998). For any SAS⁺ planning task $\Pi = \langle V, A, I, G, \text{cost} \rangle$, any subset of variables $V' \subseteq V$ defines an over-approximating *projection abstraction* $\Pi^{[V']}$ =

$\langle V', A^{[V']}, I^{[V']}, G^{[V']}, cost' \rangle$ of Π by intersecting the initial state, the goal, and all the actions' pre-conditions and effects with V' (Edelkamp, 2001). In terms of ABS-ensembles, this can be formalized as follows.

Definition 5 Given a SAS⁺ planning task $\Pi = \langle V, A, I, G, cost \rangle$, and a set of variable subsets $\{V_1, \dots, V_k\}$ of V , the **pattern-database ABS-ensemble** of $\mathcal{T}(\Pi)$ is $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ where, for $1 \leq i \leq k$,

- $\mathcal{T}_i = (S_i, L_i, Tr_i, s_i^I, S_i^G)$ is a TG-structure with $S_i = \text{dom}(V_i)$, $L_i = A^{[V_i]}$, $s_i^I = I^{[V_i]}$, $S_i^G = \{s^{[V_i]} \mid s \in S^G\}$, and, overall, $|\mathcal{T}_i| = O(\text{poly}(|\Pi|))$,
- $\alpha_i(s) = s^{[V_i]}$, and $\beta_i(a^{[V_i]}) = a$,
- $\langle \alpha_i(s), l, \alpha_i(s') \rangle \in Tr_i$ iff $\langle s, l, s' \rangle \in Tr$.

While any additive PDB abstraction $\mathcal{A} = \{\langle \langle \mathcal{T}_i, \varpi_i \rangle, \alpha_i \rangle\}_{i=1}^k \in_{\Pi} \mathcal{AE}$ induces an (admissible) additive heuristic $h_{\mathcal{A}}$, so far, the actual choice of abstraction (that is, the actual action-cost partitioning strategy) has remained an open issue (Yang et al., 2008). This is exactly where LP-optimization gradually comes into the picture.

Without loss of generality, let the number of patterns k be $O(\text{poly}(\Pi))$. Computing $h_{\mathcal{A}} = \sum_{i=1}^k \text{dist}(\alpha_i(s), S_i^G)$ for a fixed APDB abstraction $\mathcal{A} = \{\langle \langle \mathcal{T}_i, \varpi_i \rangle, \alpha_i \rangle\}_{i=1}^k \in_{\Pi} \mathcal{AE}$ is usually done by computing each $\text{dist}(\alpha_i(s), S_i^G)$ using the Dijkstra algorithm over the *explicitly constructed* transition graph $\langle \mathcal{T}_i, \varpi_i \rangle$. However, the corresponding single-source shortest path problem also has an elegant LP formulation. Given a directed graph $G = (N, E)$ and a source node $v \in N$, if $d(v')$ is a variable corresponding to the shortest-path length from v to v' , then the solution of the linear program

$$\begin{aligned} \max_{\vec{d}} \quad & \sum_{v'} d(v') \\ \text{s.t.} \quad & d(v) = 0 \\ & d(v'') \leq d(v') + w(v', v''), \quad \forall (v', v'') \in E \end{aligned} \tag{7}$$

induces a solution to the single-source shortest path problem over G and source v .

Given that, a compact LP-encoding of a pattern-database ABS-ensemble $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ is obtained by (i) putting together the linear constraints as in Eq. 7 for TG-structures \mathcal{T}_i , (ii) replacing the edge-weight constants $w(v', v'')$ by variables associated with the corresponding transition labels, and (iii) connecting the latter label-cost variables with the proper additivity constraint. Specifically, given a SAS⁺ planning task $\Pi = \langle V, A, I, G, cost \rangle$, and a pattern-database ABS-ensemble $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ of $\mathcal{T}(\Pi)$, the LP-encoding construction is as follows.

First, let the label-cost variables \vec{w} contain a variable $w_{a'}$ for every abstract action $a' \in \cup_{i=1}^m A^{[V_i]}$; the additivity constraints are defined in terms of these label-cost variables exactly as in Eq. 4. Now, given a state s of Π , we specify the LP-encoding $\mathbb{L}(s) = \langle \vec{x}, f(\vec{x}), \mathbb{C}^{\mathcal{AE}}(\vec{x}, \vec{w}) \rangle$ of \mathcal{AE} with respect to s as

$$\begin{aligned} \vec{x} &= \bigcup_{i=1}^k \{d(s') \mid s' \in S_i\} \cup \{d(G_i)\} \\ \mathbb{C}^{\mathcal{AE}} &= \begin{cases} d(s') \leq d(s'') + w_{a'}, & \forall \langle s'', a', s' \rangle \in Tr_i \\ d(s') = 0, & s' = s^{[V_i]} \\ d(G_i) \leq d(s'), & s' \in S_i^G \end{cases}, \forall i \\ f(\vec{x}) &= \sum_{i=1}^k d(G_i) \end{aligned} \tag{8}$$

Since each TG-structure \mathcal{T}_i in a pattern-database ABS-ensemble \mathcal{AE} is of size $O(\text{poly}(|\Pi|))$, it is immediate that the above LP-encoding of \mathcal{AE} is both compact and poly-time constructible for any state s of Π . Hence, we have $h_{\mathcal{AE}}(s)$ being poly-time computable for any planning task Π , any pattern-database ABS-ensemble \mathcal{AE} , and any state s of Π . Moreover, the same single-source shortest-path problems on *explicit* transition graphs underlie heuristics corresponding to additional powerful homomorphism abstractions such as constrained PDBs (Haslum et al., 2005), and merge-and-shrink abstractions (Helmert et al., 2007). Hence, using the “composition” Theorem 4, we can summarize here with a tractability claim that covers arbitrary combinations of such abstractions.

Theorem 5 *Given a SAS⁺ planning task Π , and an ABS-ensemble $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ of $\mathcal{T}(\Pi)$, if $\sum_{i=1}^k |\mathcal{T}_i| = O(\text{poly}(|\Pi|))$, then \mathcal{AE} is LP-optimizable, and thus $h_{\mathcal{AE}}(s)$ is poly-time computable for every state s of Π .*

Proof: Let Π be a SAS⁺ planning task, $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$ be an ABS-ensemble of $\mathcal{T}(\Pi)$, s be some state of Π , and $\mathbb{L}(s) = \langle \vec{x}, f(\vec{x}), \mathbb{C}^{\mathcal{AE}}(\vec{x}\vec{w}) \rangle$ be as in Eq. 8. For any $\mathbf{w} \in \mathcal{H}^{\text{add}}$, let $\mathcal{A}_{\mathbf{w}} = \{\langle \langle \mathcal{T}_i, \mathbf{w} \rangle, \alpha_i \rangle\}_{i=1}^k \in_{\Pi} \mathcal{AE}$, and therefore $h_{\mathcal{A}_{\mathbf{w}}}(s) = \sum_{i=1}^k \text{dist}(\alpha_i(s), S_i^G)$. For all i , $\text{dist}(\alpha_i(s), S_i^G) = \max_{\mathbf{x}\mathbf{w} \in \mathcal{H}^{\mathcal{AE}}} d(G_i)$, and therefore $\max_{\mathbf{x}\mathbf{w} \in \mathcal{H}^{\mathcal{AE}}} f(\mathbf{x}) = h_{\mathcal{A}_{\mathbf{w}}}(s)$. From Definition 4 we thus have $\mathbb{L}(s) = \langle \vec{x}, f(\vec{x}), \mathbb{C}^{\mathcal{AE}}(\vec{x}\vec{w}) \rangle$ being an LP-encoding of $\mathcal{AE} = \{\langle \mathcal{T}_i, \alpha_i, \beta_i \rangle\}_{i=1}^k$. In turn, if $\sum_{i=1}^k |\mathcal{T}_i| = O(\text{poly}(|\Pi|))$, then $k = O(\text{poly}(|\Pi|))$, and $|\mathcal{T}_i| = O(\text{poly}(\Pi))$ for all $1 \leq i \leq k$. Therefore the LP-encoding of \mathcal{AE} with respect to s is both compact, poly-time constructible, and thus, $h_{\mathcal{AE}}(s)$ is poly-time computable. ■

6. Hybrid Abstractions: Fork-Decomposition Structural Patterns

PDB heuristics and their enhancements are successfully exploited these days in the planning systems (Haslum et al., 2005; Haslum, Botea, Helmert, Bonet, & Koenig, 2007). However, since the reachability analysis in $\Pi^{[V_i]}$ is done by exhaustive search, each pattern of a PDB heuristic (that is, each selected variable subset V_i) is required to be as small as $O(\log |V|)$ if $|\text{dom}(v)| = O(1)$ for each $v \in V_i$, and as small as $O(1)$, otherwise. For many planning domains this constraint implies an inherent scalability limitation of the PDB heuristics. One attack on this limitation *within* the scope of homomorphism abstractions correspond to the (already covered by Theorem 5) merge-and-shrink abstractions that sophisticatedly compress the abstract transition graphs to keep them within the reach of exhaustive graph searching (Helmert et al., 2007). Another recent proposal generalizes PDB abstractions to what is called *structural patterns* (Katz & Domshlak, 2007a, 2007b).

The basic idea behind structural patterns is in abstracting the problem in hand into instances of provably tractable fragments of optimal planning, alleviating by that the limitation of PDBs to use projections of only low dimensionality. In particular, Katz and Domshlak (2007a) specify a family of *causal-graph structural patterns* (CGSPs), and introduce a concrete instance of CGSPs called *fork-decomposition*. In a one-paragraph summary, the mechanism of fork-decomposition works as follows.

The causal graph $CG(\Pi) = (V, E)$ of a SAS⁺ planning task $\Pi = \langle V, A, I, G, \text{cost} \rangle$ with variables $V = \{v_1, \dots, v_n\}$ is a digraph over nodes V . An arc (v, v') belongs to $CG(\Pi)$ iff $v \neq v'$ and there exists an action $a \in A$ such that $\text{eff}(a)[v']$, and either $\text{pre}(a)[v]$ or $\text{eff}(a)[v]$ are specified. For each variable $v_i \in V$, let $V_i^f \subseteq V$ contain v_i and all its immediate successors in $CG(\Pi)$, and $V_i^i \subseteq V$ contain v_i and all its immediate predecessors in $CG(\Pi)$. The fork-decomposition of Π is obtained by

- (1) schematically constructing a set of projection homomorphism abstractions $\mathbf{\Pi} = \{\Pi^{[V_i^f]}, \Pi^{[V_i^i]}\}_{i=1}^n$ (with, possibly, each $|V_i^f|$ and each $|V_i^i|$ being $\Theta(n)$),

- (2) reformulating the actions of the abstractions to single-effect actions only so that the causal graphs of $\Pi^{[V_i^f]}$ and $\Pi^{[V_i^i]}$ become “forks” and “inverted forks”, respectively, rooted in v_i ; after this action reformulation, the individual abstractions may cease being purely homomorphic,
- (3) within each $\Pi^{[V_i^f]}$, abstracting the domain of v_i to $\{0, 1\}$, and within each $\Pi^{[V_i^i]}$, abstracting the domain of v_i to $\{0, 1, \dots, k\}$ with $k = O(1)$.

This decomposition of Π provides us with the **fork-decomposition ABS-ensemble** $\mathcal{AE} = \{\langle \mathcal{T}(\Pi^{[V_i^f]}), \alpha_i^f, \beta_i^f \rangle, \langle \mathcal{T}(\Pi^{[V_i^i]}), \alpha_i^i, \beta_i^i \rangle\}_{i=1}^n$ of $\mathcal{T}(\Pi)$, with the abstraction mappings α_i^f, α_i^i and the action associations β_i^f, β_i^i being established along the above steps (1-3). The additive abstractions of such an ABS-ensemble \mathcal{AE} are of interest because (i) they can provide quite informative heuristic estimates, and (ii) each abstract problem in $\Pi = \{\Pi^{[V_i^f]}, \Pi^{[V_i^i]}\}_{i=1}^n$ can be solved in polynomial time by special-purpose algorithms for the corresponding fragments of SAS^+ (Katz & Domshlak, 2007a). However, here as well, the choice of the actual abstraction with respect to \mathcal{AE} is important, and optimizing this choice is clearly of interest. Interestingly, LP-optimization can come to the rescue here as well, and this despite the TG-structures of \mathcal{AE} *not* being searchable explicitly in polynomial time.

Theorem 6 *For any SAS^+ planning task Π over n variables, the fork-decomposition ABS-ensemble $\mathcal{AE} = \{\langle \mathcal{T}(\Pi^{[V_i^f]}), \alpha_i^f, \beta_i^f \rangle, \langle \mathcal{T}(\Pi^{[V_i^i]}), \alpha_i^i, \beta_i^i \rangle\}_{i=1}^n$ of $\mathcal{T}(\Pi)$ is LP-optimizable, and thus $h_{\mathcal{AE}}(s)$ is poly-time computable for every state s of Π .*

Proof: From Lemmas 1, 2 we have for each $1 \leq i \leq n$, $\mathcal{AE}_i^f = \{\langle \mathcal{T}(\Pi^{[V_i^f]}), \alpha_i^f, \beta_i^f \rangle\}$ and $\mathcal{AE}_i^i = \{\langle \mathcal{T}(\Pi^{[V_i^i]}), \alpha_i^i, \beta_i^i \rangle\}$ both being LP-optimizable, and thus, using the “composition” Theorem 4, we have \mathcal{AE} being LP-optimizable, and thus $h_{\mathcal{AE}}(s)$ is poly-time computable for every state s of Π . ■

Lemma 1 *For any SAS^+ planning task $\Pi = \langle V, A, I, G, cost \rangle$ and for any variable $r \in V$, the fork-decomposition ABS-ensemble $\mathcal{AE} = \{\langle \mathcal{T}(\Pi^{[V_r^f]}), \alpha_r^f, \beta_r^f \rangle\}$ of $\mathcal{T}(\Pi)$ is LP-optimizable.*

Proof: Our LP-encoding of a fork-decomposition ABS-ensemble \mathcal{AE} corresponds to LP reformulations of the algorithm of Katz and Domshlak (2007a) for fork problem with binary root domain.

Given a SAS^+ planning task $\Pi = \langle V, A, I, G, cost \rangle$ and a variable $r \in V$, let $\mathcal{AE} = \{\langle \mathcal{T}(\Pi_r^f), \alpha^f, \beta^f \rangle\}$ be a fork-decomposition of $\mathcal{T}(\Pi)$ over a single fork rooted in r , and s be some state of Π . Considering that abstract problem, let us denote its variables by V' , its actions by A' , and its goal state $G^{[V']}$ by G' . We can assume that $G'[v]$ is defined for all $v \in V' \setminus \{r\}$; all the goal-less leaves can be simply omitted from the fork. For coherence with the notation of Katz and Domshlak (2007a) we denote the (abstracted to binary-valued) domain of r by $dom(r) = \{0, 1\}$ such that $s[r] = 0$. Let $\sigma(r)$ be a 0/1 sequence of length $1 + \max_{v \in V'} |dom(v)|$, and, for $1 \leq i \leq |\sigma(r)|$, $\sigma(r)[i] = 0$ if i is odd, and $= 1$, if i is even. Let $\succeq^*[\sigma(r)]$ be the set of all non-empty prefixes of $\sigma(r)$ if $G'[r]$ is unspecified, and otherwise, be the set of all prefixes of $\sigma(r)$ ending with $G'[r]$.

First, let the label-cost variables \vec{w} contain a variable $w_{a'}$ for every abstract action $a' \in A'$; the additivity constraints $\mathbb{C}^{\text{add}}(\vec{w})$ are defined in terms of these label-cost variables via β^f as in Eq. 4. Now we specify the LP-encoding $\mathbb{L}(s) = \langle \vec{x}, f(\vec{x}), \mathbb{C}^{\mathcal{AE}}(\vec{x}\vec{w}) \rangle$ of \mathcal{AE} with respect to s as follows. The variable set \vec{x} of $\mathbb{L}(s)$ consists of three types of variables, notably

$$\vec{x} = \{h^f\} \cup \bigcup_{\substack{v \in V' \setminus \{r\}, \\ \vartheta \in dom(v), \\ 1 \leq i \leq |\sigma(r)|}} \{d(v, \vartheta, i)\} \cup \bigcup_{\substack{v \in V' \setminus \{r\}, \\ \vartheta, \vartheta' \in dom(v), \\ \vartheta_r \in \{0, 1\}}} \{p(v, \vartheta, \vartheta', \vartheta_r)\}.$$

The variable h^f stands for the minimal cost of solving our fork-structured problem, and the objective function of $\mathbb{L}(s)$ is simply $f(\vec{x}) = h^f$. Each variable $d(v, \vartheta, i)$ stands for the cost of the cheapest sequence of actions affecting v that changes its value from $s[v]$ to ϑ *given* that the value changes of r induce a 0/1 sequence of length i . Each variable $p(v, \vartheta, \vartheta', \vartheta_r)$ stands for the cost of the cheapest

sequence of actions affecting v that changes its value from ϑ to ϑ' having *fixed* the value of r to ϑ_r . The constraints set $\mathbb{C}^{\mathcal{AE}}$ of $\mathbb{L}(s)$ consists of the following sets of linear constraints.

- (i) For all goal-achieving sequences $\sigma \in \succeq^*[\sigma(r)]$ of value changes of r , and each pair of r -changing actions $a, a' \in A'$ such that $\text{eff}(a)[r] = 1$ and $\text{eff}(a')[r] = 0$,

$$h^f \leq \lceil \frac{|\sigma| - 1}{2} \rceil \cdot w_a + \lfloor \frac{|\sigma| - 1}{2} \rfloor \cdot w_{a'} + \sum_{v \in V' \setminus \{r\}} d(v, G'[v], |\sigma|)$$

Semantics: The cost of solving the problem is not greater than the sum of achieving the goal values for all the leafs given a value sequence of the root, plus the cost of providing that value sequence.

- (ii) For each leaf $v \in V' \setminus \{r\}$ and for each $\vartheta' \in \text{dom}(v)$

$$d(v, \vartheta', 1) \leq p(v, s[v], \vartheta', \sigma(r)[1])$$

and, for each $\vartheta \in \text{dom}(v)$, and $1 < i \leq |\sigma(r)|$,

$$d(v, \vartheta', i) \leq d(v, \vartheta, i - 1) + p(v, \vartheta, \vartheta', \sigma(r)[i])$$

Semantics: The cost of achieving ϑ' from $s[v]$ given $|\sigma(r)| = i$ is bounded by the cost of achieving ϑ given $|\sigma(r)| = i - 1$, and achieving ϑ' from ϑ given $\sigma(r)[i]$.

- (iii) For each $v \in V' \setminus \{r\}$, $\vartheta \in \text{dom}(v)$,

$$p(v, \vartheta, \vartheta, 0) = 0, \quad p(v, \vartheta, \vartheta, 1) = 0$$

Likewise, for each v -changing action $a \in A'$, if $\text{pre}(a)[r]$ is unspecified, then, for $\vartheta_r \in \{0, 1\}$,

$$p(v, \vartheta, \text{eff}(a)[v], \vartheta_r) \leq p(v, \vartheta, \text{pre}(a)[v], \vartheta_r) + w_a$$

and otherwise,

$$p(v, \vartheta, \text{eff}(a)[v], \text{pre}(a)[r]) \leq p(v, \vartheta, \text{pre}(a)[v], \text{pre}(a)[r]) + w_a$$

Semantics: Shortest-path constraints as in Eq. 7.

This finalizes our LP-encoding for an ABS-ensemble consisting of a single fork-structured abstraction with a binary root domain. Now, let $D = \max_{v \in V'} |\text{dom}(v)|$. The variables set size $|\vec{x}|$ is $O(|V| \cdot D^2)$ and the constraints set size $|\mathbb{C}^{\mathcal{AE}}|$ is $O(D \cdot |A'|^2 + |V'| \cdot D^3 + |A'| \cdot |V'| \cdot D)$, therefore the LP-encoding is compact.

The correctness stems from the correctness of the algorithm in (Katz & Domshlak, 2007a). First, if Π is solvable, then the algorithm finds an optimal plan $\rho = \rho_\sigma$ for some $\sigma \in \succeq^*[\sigma(r)]$. Furthermore, if $\rho \downarrow_r = \langle a_2 \dots a_{|\sigma|} \rangle$,² then for each $v \in V' \setminus \{r\}$, $\rho \downarrow_v$ can be divided into sequences of actions changing the value of v , where each pair of consequent sequences is separated by an action changing the value of r , i.e., $\rho \downarrow_v = \langle \rho_v^1 \cdot \rho_v^2 \cdot \dots \cdot \rho_v^{|\sigma|} \rangle$. Note that, for each $1 \leq i \leq |\sigma|$, all actions of ρ_v^i are prevailed either by $\sigma[i]$ or not prevailed at all. Let $\rho_v^i = \langle a_1^i \cdot \dots \cdot a_{k_i}^i \rangle$ and $\bigcup_{i=1}^{|\sigma|} \bigcup_{j=1}^{k_i} \{\vartheta_j^i\} \subseteq \text{dom}(v)$ be such that $\vartheta_{j-1}^i = \text{pre}(a_j^i)[v]$ and $\vartheta_j^i = \text{post}(a_j^i)[v]$. Given that, let $\mathbf{x} \in \text{dom}(\vec{x})$ be specified as follows.

$$\mathbf{x}[p(v, \vartheta, \vartheta', 0)] = \begin{cases} 0, & \vartheta = \vartheta' \\ \sum_{j \leq i \leq j'} \text{Cost}(a_i^i), & \vartheta = \vartheta_j^i, \vartheta' = \vartheta_{j'}^i, 1 \leq j < j' \leq k_i, i \text{ is odd} \\ \infty, & \text{otherwise} \end{cases}$$

2. For ease of notation we start with a_2 .

$$\mathbf{x}[p(v, \vartheta, \vartheta', 1)] = \begin{cases} 0, & \vartheta = \vartheta' \\ \sum_{j \leq l \leq j'} \text{Cost}(a_l^i), & \vartheta = \vartheta_j^i, \vartheta' = \vartheta_{j'}^i, 1 \leq j < j' \leq k_i, i \text{ is even}, \\ \infty, & \text{otherwise} \end{cases}$$

$$\mathbf{x}[d(v, \vartheta, i)] = \begin{cases} \sum_{m=1}^{i'-1} \sum_{l=1}^{k_m} \text{Cost}(a_l^m) + \sum_{l=1}^j \text{Cost}(a_l^{i'}), & \vartheta = \vartheta_j^{i'}, 1 \leq i' \leq i \leq |\sigma|, \\ \infty, & \text{otherwise} \end{cases}$$

$$\mathbf{x}[h^f] = \text{Cost}(\rho).$$

If $\mathbf{w} \in \text{dom}(\vec{w})$ is such that for each $a \in A'$, $\mathbf{w}[w_a] = \text{Cost}(a)$, then \mathbf{xw} agree with the constraints sets (i), (ii), and (iii), and thus $\mathbf{xw} \in \mathcal{H}^{\mathcal{AE}}$, resulting in

$$\max_{\mathbf{x}'\mathbf{w}' \in \mathcal{H}^{\mathcal{AE}}} h^f \geq \mathbf{x}[h^f] = \text{Cost}(\rho) = h^*(s).$$

Now, let ρ_σ be the plan constructed by the algorithm in (Katz & Domshlak, 2007a) for each $\sigma \in \sqsupset^*[\sigma(r)]$, and let \mathbf{xw} be some point in $\mathcal{H}^{\mathcal{AE}}$. Let $\rho_{\sigma \downarrow v} = \langle \rho_v^1 \cdot \rho_v^2 \cdot \dots \cdot \rho_v^{|\sigma|} \rangle$, such that all actions in ρ_v^i are prevailed either by $\sigma[i]$ or not prevailed at all, and let $\rho_v^i = \langle a_1^i \cdot \dots \cdot a_{k_i}^i \rangle$. Finally, let $\vartheta_0^1, \vartheta_1^1, \dots, \vartheta_{k_1}^1, \vartheta_1^2, \dots, \vartheta_{k_2}^2, \dots, \vartheta_1^{|\sigma|}, \dots, \vartheta_{k_{|\sigma|}}^{|\sigma|}$ be the simple path in domain transition graph of v such that $\vartheta_0^1 = s[v]$ and for each $1 \leq i \leq |\sigma|$ and each $1 \leq j \leq k_i$, $\vartheta_j^i = \text{post}(a_j^i)[v]$. For ease of presentation, denote $\vartheta_0^{i+1} = \vartheta_{k_i}^i$ for each $1 \leq i \leq |\sigma|$. Then

$$\mathbf{x}[d(v, G'[v], |\sigma|)] \stackrel{(ii)}{\leq} \sum_{i=1}^{|\sigma|} \sum_{j=1}^{k_i} \mathbf{x}[p(v, \vartheta_{j-1}^i, \vartheta_j^i, \sigma[i])] \stackrel{(iii)}{\leq} \sum_{i=1}^{|\sigma|} \sum_{j=1}^{k_i} \mathbf{w}[w_{a_j^i}] \stackrel{\text{Eq. 4}}{\leq} \text{Cost}(\rho_{\sigma \downarrow v})$$

and therefore $\mathbf{x}[h^f] \leq \text{Cost}(\rho_\sigma)$ for each $\sigma \in \sqsupset^*[\sigma(r)]$ and each $\mathbf{xw} \in \mathcal{H}^{\mathcal{AE}}$, and thus

$$\max_{\mathbf{xw} \in \mathcal{H}^{\mathcal{AE}}} h^f \leq h^*(s). \quad \blacksquare$$

Lemma 2 For any SAS⁺ planning task $\Pi = \langle V, A, I, G, \text{cost} \rangle$ and for any variable $r \in V$, the inverted fork-decomposition ABS-ensemble $\mathcal{AE} = \{ \langle \mathcal{T}(\Pi^{[V^r]}), \alpha_r^i, \beta_r^i \rangle \}$ of $\mathcal{T}(\Pi)$ is LP-optimizable.

Proof: Our LP-encoding of a fork-decomposition ABS-ensemble \mathcal{AE} corresponds to LP reformulations of the algorithm of Katz and Domshlak (2007a) for inverted-fork problem with a constant bounded root domain.

Given a SAS⁺ planning task $\Pi = \langle V, A, I, G, \text{cost} \rangle$ and a variable $r \in V$, let $\mathcal{AE} = \{ \langle \mathcal{T}(\Pi_r^i), \alpha^i, \beta^i \rangle \}$ be an inverted fork-decomposition of $\mathcal{T}(\Pi)$ over a single inverted fork rooted in r . Considering that abstract problem, let us denote its variables by V' , its actions by A' , and its goal state $G^{[V']}$ by G' .

First, let the label-cost variables \vec{w} contain a variable $w_{a'}$ for every abstract action $a' \in A'$; the additivity constraints $\mathbb{C}^{\text{add}}(\vec{w})$ are defined in terms of these label-cost variables via β^i as in Eq. 4. Now, given a state s of Π , we specify the LP-encoding $\mathbb{L}(s) = \langle \vec{x}, f(\vec{x}), \mathbb{C}^{\mathcal{AE}}(\vec{x}\vec{w}) \rangle$ of \mathcal{AE} with respect to s as follows. The variable set \vec{x} of $\mathbb{L}(s)$ consists of two types of variables, notably

$$\vec{x} = \{h^i\} \cup \bigcup_{\substack{v \in V' \setminus \{r\}, \\ \vartheta, \vartheta' \in \text{dom}(v)}} \{d(v, \vartheta, \vartheta')\}.$$

The variable h^i stands for the minimal cost of solving our inverted fork-structured problem, and the objective function of $\mathbb{L}(s)$ is simply $f(\vec{x}) = h^i$. Each variable $d(v, \vartheta, \vartheta')$ stands for the cost of the cheapest sequence of actions affecting v that changes its value from ϑ to ϑ' . The constraint $\mathbb{C}^{\mathcal{AE}}$ of $\mathbb{L}(s)$ consists of the following sets of linear constraints.

- (i) For each cycle-free path $\rho_r = a_1 \cdot \dots \cdot a_m$ from $s[r]$ to $G'[r]$ in the domain transition graph of r in $\Pi_{G_r^i}$,

$$h^i \leq \sum_{v \in V' \setminus \{r\}} d(v, s_0[v], s_1[v]) + \sum_{i=1}^m \left(w_{a_i} + \sum_{v \in V' \setminus \{r\}} d(v, s_i[v], s_{i+1}[v]) \right)$$

where for each $v \in V' \setminus \{r\}$, and for each $0 \leq i \leq m+1$,

$$s_i[v] = \begin{cases} s[v], & i = 0 \\ G'[v], & i = m+1, \text{ and } G'[v] \text{ is specified} \\ \text{pre}(a_i)[v], & 1 \leq i \leq m, \text{ and } \text{pre}(a_i)[v] \text{ is specified} \\ s_{i-1}[v] & \text{otherwise} \end{cases} \quad (9)$$

Semantics: The cost of solving the problem is not greater than the cost of any cycle-free path of r plus sums of costs of reaching the prevail conditions of actions on this path and reaching the goal afterwards.

- (ii) For each $v \in V' \setminus \{r\}$, $\vartheta \in \text{dom}(v)$,
- $$d(v, \vartheta, \vartheta) = 0$$

Likewise, for each v -changing action $a \in A'$,

$$d(v, \vartheta, \text{post}(a)[v]) \leq d(v, \vartheta, \text{pre}(a)[v]) + w_a$$

Semantics: Shortest-path constraints as in Eq. 7.

This finalizes our LP-encoding for an ABS-ensemble consisting of a single inverted fork-structured abstraction with a constant bounded root domain. Now, let $D = \max_{v \in V'} |\text{dom}(v)|$ and $d = |\text{dom}(r)|$. The variables set size $|\vec{x}|$ is $O(|V| \cdot D^2)$ and the constraints set size $|\mathbb{C}^{\mathcal{AE}}|$ is $O(d^d + |V'| \cdot |A'| \cdot D)$, therefore the LP-encoding is compact.

The correctness stems from the correctness of the algorithm in (Katz & Domshlak, 2007a). First, if Π is solvable, then the algorithm finds an optimal plan ρ for some cycle-free path $\rho_r = a_1 \cdot \dots \cdot a_m$ in the domain transition graph of r from $s[r]$ to $G'[r]$. Furthermore, $\rho_r = \rho_r$, and for each $v \in V' \setminus \{r\}$, ρ_v can be divided into sequences of actions changing the value of v , where each pair of consequent sequences is separated by an action changing the value of r , i.e., $\rho_v = \langle \rho_v^0 \cdot \rho_v^1 \cdot \dots \cdot \rho_v^m \rangle$. Note that, for each $0 \leq i \leq m$, all actions of ρ_v^i are not prevailed by anything.

Let $\rho_v^i = \langle a_1^i \cdot \dots \cdot a_{k_i}^i \rangle$, and let $\vartheta_0^0, \vartheta_1^0, \dots, \vartheta_{k_0}^0, \vartheta_1^1, \dots, \vartheta_{k_1}^1, \dots, \vartheta_1^m, \dots, \vartheta_{k_m}^m$ be the path in domain transition graph of v such that $\vartheta_0^0 = s[v]$ and for each $0 \leq i \leq m$ and each $1 \leq j \leq k_i$, $\vartheta_j^i = \text{post}(a_j^i)[v]$. For ease of presentation, denote $\vartheta_0^{i+1} = \vartheta_{k_i}^i$ for each $0 \leq i < m$. For each $0 \leq i \leq m+1$, let s_i be defined as in Eq. 9. Then, $s_0[v] = \vartheta_0^0$, and for each $0 \leq i \leq m$, $s_{i+1}[v] = \vartheta_{k_i}^i$. Finally, let $\text{dist}_v(\vartheta, \vartheta')$ denote the cost of the shortest path from ϑ to ϑ' in the domain transition graph of v , or ∞ if no such path exists. We construct $\mathbf{x} \in \text{dom}(\vec{x})$ as follows.

$$\mathbf{x}[d(v, \vartheta, \vartheta')] = \text{dist}_v(\vartheta, \vartheta'),$$

$$\mathbf{x}[h^i] = \text{Cost}(\rho).$$

If $\mathbf{w} \in \text{dom}(\vec{w})$ is such that for each $a \in A'$, $\mathbf{w}[w_a] = \text{Cost}(a)$, then

$$\mathbf{x}[d(v, s_i[v], s_{i+1}[v])] = \text{dist}_v(s_i[v], s_{i+1}[v]) = \text{Cost}(\rho_v^i)$$

for each $0 \leq i \leq m$, and therefore \mathbf{xw} agree with the constraints sets (i) and (ii), and thus $\mathbf{xw} \in \mathcal{H}^{\mathcal{AE}}$, resulting in

$$\max_{\mathbf{x}' \mathbf{w}' \in \mathcal{H}^{\mathcal{AE}}} h^i \geq \mathbf{x}[h^i] = \text{Cost}(\rho) = h^*(s).$$

On the other hand, if \mathbf{xw} is some point in $\mathcal{H}^{\mathcal{AE}}$, then

$$\sum_{i=0}^m \mathbf{x}[d(v, s_i[v], s_{i+1}[v])] \stackrel{(ii)}{\leq} \sum_{i=0}^m \sum_{j=1}^{k_i} \mathbf{w}[w_{a_j^i}] \stackrel{Eq. 4}{\leq} Cost(\rho_v)$$

and therefore $\mathbf{x}[h^i] \leq Cost(\rho) = h^*(s)$ for each $\mathbf{xw} \in \mathcal{H}^{\mathcal{AE}}$, and thus

$$\max_{\mathbf{xw} \in \mathcal{H}^{\mathcal{AE}}} h^f \leq h^*(s).$$

■

7. Structural Patterns and Tree-structured COPs

Fork-decomposition structural patterns are grounded in two specific fragments of tractable cost-optimal planning. In principle, it is possible that structural patterns based on some other such fragments also lead to LP-optimizable ABS-ensembles. Here we consider two such fragments that have been recently characterized by Katz and Domshlak (2007b). Both these fragments correspond to problems over binary-valued state variables and actions inducing a polytree³ causal graph. In the first fragment, \mathbf{P}_b , all the causal graph’s nodes have $O(1)$ -bounded in-degree. In the second fragment, $\mathbf{P}(1)$, all actions are 1-dependent.

While the poly-time solution schemes provided by Katz and Domshlak for \mathbf{P}_b and $\mathbf{P}(1)$ substantially differ one from another, they both correspond to *reductions of the planning problems to compact and tree-structured constraint optimization problems (COPs)*. This joint property of \mathbf{P}_b and $\mathbf{P}(1)$ (that might also hold for some other problem fragments as well) turns out to be very helpful to our objective of joining \mathbf{P}_b - and $\mathbf{P}(1)$ -based structural patterns to the “ $h_{\mathcal{AE}}$ -friendly” family of LP-optimizable ABS-ensembles.

Let us start by considering a general, tree-structured constraint optimization problem $\text{COP} = (\mathcal{X}, \mathcal{F})$ over finite-domain variables \mathcal{X} , functional components \mathcal{F} , and the objective $\min \sum_{\varphi \in \mathcal{F}} \varphi(\mathcal{X})$. Fixing an arbitrary rooting of the COP’s constraint network at $r \in \mathcal{X}$, in what follows we refer to that rooted tree of COP via its set of *directed* edges $E = \{(x, x')\}$. In these terms, we have $\mathcal{F} = \{\varphi_x : \text{dom}(y) \times \text{dom}(x) \rightarrow \mathbb{R}^{0+} \mid (y, x) \in E\}$.

It is well-known that tree-structured COPs as above can be solved in low polynomial time by a dynamic-programming-style, message-passing algorithm (Dechter, 2003). The bad news is that (similarly to what we had with the Dijkstra algorithm for solving PDBs) we cannot use this message-passing algorithm for our needs. The good news, however, is that such tree-structured COPs can also be solved via linear programming⁴. Specifically, given such a problem $\text{COP} = (\mathcal{X}, \mathcal{F})$, let

$$\vec{c} = \{h^{\text{cop}}\} \cup \bigcup_{\substack{(x', x) \in E, \\ \bar{x}' \in \text{dom}(x')}} \{c(x|\bar{x}')\}$$

be a set of non-negative, real-valued variables, with the semantics of each $c(x|\bar{x}')$ being “optimal solution for the sub-tree rooted at x given $x' = \bar{x}'$ ”. Then, the solution of LP

$$\begin{aligned} & \max_{\vec{c}} h^{\text{cop}} \\ & \text{s.t. } \forall \bar{r} \in \text{dom}(r) : \quad h^{\text{cop}} \leq \sum_{(r, x) \in E} c(x|\bar{r}), \\ & \quad \forall (x, y) \in E, \bar{x} \in \text{dom}(x), \bar{y} \in \text{dom}(y) : \\ & \quad \quad c(y|\bar{x}) \leq \sum_{(y, z) \in E} c(z|\bar{y}) + \varphi_y(\bar{x}, \bar{y}). \end{aligned} \tag{10}$$

3. Polytree is a DAG with an acyclic induced undirected graph.

4. We suspect that this LP formulation is not new, but we found no previous mention of it.

induces a solution for COP.

Lemma 3 *Given a tree-structured constraint optimization problem $\text{COP} = (\mathcal{X}, \mathcal{F})$ over finite-domain variables \mathcal{X} and functional components \mathcal{F} , we have*

$$\min_{\bar{x} \in \text{dom}(\mathcal{X})} \sum_{\varphi \in \mathcal{F}} \varphi(\bar{x}) = \max_{\mathbf{c} \in \mathcal{H}^{\text{COP}}} h^{\text{COP}}$$

where \mathcal{H}^{COP} is the convex polyhedron specified by the linear constraints as in Eq. 10.

Proof: First, note that

$$\forall \bar{x} \in \text{dom}(\mathcal{X}) \text{ and } \forall \mathbf{c} \in \mathcal{H}^{\text{COP}}, \quad h^{\text{COP}} \leq \sum_{(y,z)} \varphi_z(\bar{x}) = \sum_{\varphi \in \mathcal{F}} \varphi(\bar{x}).$$

Therefore

$$\max_{\mathbf{c} \in \mathcal{H}^{\text{COP}}} h^{\text{COP}} \leq \min_{\bar{x} \in \text{dom}(\mathcal{X})} \sum_{\varphi \in \mathcal{F}} \varphi(\bar{x}).$$

Let $\mathbf{c} \in \mathcal{H}^{\text{COP}}$ be an assignment constructed by going over the edges (y, z) of the tree bottom up, and for each $\bar{y} \in \text{dom}(y)$ setting

$$c(z|\bar{y}) = \min_{\bar{z} \in \text{dom}(z)} \sum_{(z,z')} c(z'|\bar{z}) + \varphi_z(\bar{y}, \bar{z}) \quad (11)$$

and then setting

$$h^{\text{COP}} = \min_{\bar{r} \in \text{dom}(r)} \sum_{(r,x)} c(x|\bar{r}). \quad (12)$$

Let $\bar{x} \in \text{dom}(\mathcal{X})$ be the assignments to COP variables that obtain the minimum in Eq. 11-12 above, going over the edges (y, z) of the tree top down, starting with \bar{r} that obtains the minimum in Eq. 12. Then

$$h^{\text{COP}} = \min_{\bar{x} \in \text{dom}(\mathcal{X})} \sum_{\varphi \in \mathcal{F}} \varphi(\bar{x}),$$

and therefore

$$\max_{\mathbf{c} \in \mathcal{H}^{\text{COP}}} h^{\text{COP}} \geq \min_{\bar{x} \in \text{dom}(\mathcal{X})} \sum_{\varphi \in \mathcal{F}} \varphi(\bar{x}).$$

■

With Lemma 3 in hand, we now make two additional steps towards an LP-encoding of ABS-ensembles \mathcal{AE} containing structural patterns reducible to tree-structured COPs. In each such *individual* structural pattern, each value $\varphi_y(\bar{x}, \bar{y})$ of each functional component φ_y is somehow pre-computed from the costs of the actions. In our case, however, the costs of the actions in the abstract problems are not fixed in advanced, but should be determined by the process of LP-optimization. This is where our next steps come into the picture.

Consider a single COP-reducible cost-optimal planning problem. First, *suppose* that, for any *fixed* action-costs vector \mathbf{w}^\dagger , each functional-component value $\bar{\varphi} \equiv \varphi_y(\bar{x}, \bar{y})$ corresponds to a solution value of some compact (canonical form) linear program

$$\begin{aligned} & \max f_{\bar{\varphi}}(\vec{z}_{\bar{\varphi}} \vec{w}) \\ \text{s.t. } & \mathbf{A}_{\bar{\varphi}} \cdot \vec{z}_{\bar{\varphi}} \vec{w} \leq \mathbf{b}_{\bar{\varphi}} \\ & \vec{w} \leq \mathbf{w}^\dagger \end{aligned} \quad (13)$$

where $\mathbf{A}_{\bar{\varphi}}$ and $\mathbf{b}_{\bar{\varphi}}$ are a matrix and a vector of coefficients, respectively. If so, then, given \mathbf{w}^\dagger , we can reformulate the linear program in Eq. 10 by (i) replacing the *constants* $\varphi_y(\bar{x}, \bar{y})$ by the corresponding affine functions $f_{\bar{\varphi}}(\bar{z}_{\bar{\varphi}} \cup \bar{w})$, and (ii) for each $\bar{\varphi}$, adding its linear constraints as in Eq. 13. The extended program is still linear, and we still have

$$\min_{\bar{x} \in \text{dom}(\mathcal{X})} \sum_{\varphi \in \mathcal{F}} \varphi(\bar{x}) = \max_{\mathbf{c}, \mathbf{z}, \mathbf{w} \in \mathcal{H}^{\text{COP}}} h^{\text{COP}}$$

where \mathbf{z} and \mathbf{w} are assignments to $\bar{z} = \bigcup_{\bar{\varphi}} \bar{z}_{\bar{\varphi}}$ and action-cost variables \bar{w} , respectively, and \mathcal{H}^{COP} is the convex polyhedron specified by these *extended* linear constraints.

Now, given a SAS⁺ planning task Π , let $\mathcal{AE} = \{\langle \mathcal{T}(\Pi'), \alpha, \beta \rangle\}$ be a single-*abstraction* ABS-ensemble of $\mathcal{T}(\Pi)$ such that cost-optimal planning for Π' is reducible to a tree-structured constraint optimization problem $\text{COP}_{\Pi'}$ satisfying Eq. 13. The extended linear program specified above provides the basis for the LP-encoding of such ABS-ensembles. First, as before, let the label-cost variables \bar{w} contain a variable $w_{a'}$ for every abstract action $a' \in A'$; the additivity constraints $\mathbb{C}^{\text{add}}(\bar{w})$ are defined in terms of these label-cost variables via β as in Eq. 4. Now, given a state s of Π , we specify an LP-encoding $\mathbb{L}(s) = \langle \bar{x}, f(\bar{x}), \mathbb{C}^{\mathcal{AE}}(\bar{x}\bar{w}) \rangle$ of \mathcal{AE} with respect to s as follows.

- The variable set $\bar{x} = \bar{c}\bar{z}$ consists of the variables of Eqs. 10 and 13, and the objective of $\mathbb{L}(s)$ is $f(\bar{x}) = h^{\text{COP}}$.
- The constraint $\mathbb{C}^{\mathcal{AE}}(\bar{x}\bar{w})$ of $\mathbb{L}(s)$ consists of all the linear constraints from Eq. 10, as well as the constraint $\mathbf{A}_{\bar{\varphi}} \cdot \bar{z}_{\bar{\varphi}}\bar{w} \leq \mathbf{b}_{\bar{\varphi}}$ from Eq. 13 for all functional-component values $\bar{\varphi}$ of $\text{COP}_{\Pi'}$.

This finalizes the desired LP-encoding; extending it to such multiple-*abstraction* ABS-ensembles $\mathcal{AE} = \{\langle \mathcal{T}(\Pi'_i), \alpha_i, \beta_i \rangle\}_{i=1}^k$ (and, again, possibly some other LP-optimizable abstractions) is, again, guaranteed by the “composition” Theorem 4.

Theorem 7 *Given a SAS⁺ planning task Π , and an ABS-ensemble $\mathcal{AE} = \{\langle \mathcal{T}(\Pi'_i), \alpha_i, \beta_i \rangle\}_{i=1}^k$ of $\mathcal{T}(\Pi)$, if $k = O(\text{poly}(|\Pi|))$, and cost-optimal planning for each Π'_i is poly-time reducible to a compact and tree-structured constraint optimization problem satisfying Eq. 13, then \mathcal{AE} is LP-optimizable, and thus $h_{\mathcal{AE}}(s)$ is poly-time computable for every state $s \in S$.*

Proof: Let Π be a SAS⁺ planning task, $\mathcal{AE} = \{\langle \mathcal{T}(\Pi'_i), \alpha_i, \beta_i \rangle\}_{i=1}^k$ be an ABS-ensemble of $\mathcal{T}(\Pi)$, such that cost-optimal planning for each Π'_i is poly-time reducible to a compact and tree-structured constraint optimization problem satisfying Eq. 13, s be some state in S , and $\mathbb{L}(s) = \langle \bar{x}, f(\bar{x}), \mathbb{C}^{\mathcal{AE}}(\bar{x}\bar{w}) \rangle$ be as described above. For any $\mathbf{w} \in \mathcal{H}^{\text{add}}$, let $\mathcal{A}_{\mathbf{w}} = \{\langle \langle \mathcal{T}_i, \mathbf{w} \rangle, \alpha_i \rangle\}_{i=1}^k \in_{\Pi} \mathcal{AE}$, and therefore $h_{\mathcal{A}_{\mathbf{w}}}(s) = \sum_{i=1}^k \text{dist}(\alpha_i(s), S_i^G)$. Note that if Π'_i with initial state $\alpha_i(s)$ is poly-time reducible to a compact and tree-structured constraint optimization problem $\text{COP}_i = (\mathcal{X}_i, \mathcal{F}_i)$, satisfying Eq. 13, then $\text{dist}(\alpha_i(s), S_i^G) = \min_{\bar{x} \in \text{dom}(\mathcal{X}_i)} \sum_{\varphi \in \mathcal{F}_i} \varphi(\bar{x})$. From Lemma 3 we have then $\text{dist}(\alpha_i(s), S_i^G) = \max_{\mathbf{x}, \mathbf{w} \in \mathcal{H}^{\text{add}}} h_i^{\text{COP}}$, and therefore

$$\max_{\mathbf{x}, \mathbf{w} \in \mathcal{H}^{\text{add}}} f(\mathbf{x}) = \sum_{i=1}^k \text{dist}(\alpha_i(s), S_i^G) = h_{\mathcal{A}_{\mathbf{w}}}(s).$$

From Def. 4 we then have $\mathbb{L}(s) = \langle \bar{x}, f(\bar{x}), \mathbb{C}^{\mathcal{AE}}(\bar{x}\bar{w}) \rangle$ being an LP-encoding of \mathcal{AE} . In turn, if $k = O(\text{poly}(|\Pi|))$, and for each $1 \leq i \leq k$ cost-optimal planning for each Π'_i is poly-time reducible to a compact and tree-structured constraint optimization problem satisfying Eq. 13, then the LP-encoding of \mathcal{AE} is both compact and poly-time constructible for any state s of Π . Hence, $h_{\mathcal{AE}}(s)$ is poly-time computable for any state s of Π . ■

The last but not least is, of course, the question of whether the requirement posed by Eq. 13 is any realistic with respect to the COPs induced by the abstract planning problems. Fortunately, Theorems 8 and 9 close the story with some very good news on that matter.

Theorem 8 *Cost-optimal planning for any task Π in \mathbf{P}_b is poly-time reducible to a compact and tree-structured constraint optimization problem satisfying Eq. 13.*

Proof: We present here a proof which is based on the construction of the constraint optimization problem in (Katz & Domshlak, 2007b) (Section 3.1, pages 214 – 219). First, for each $v \in V$, and each $\sigma(v)$, by $\tau(v)$ we denote a per-value time-stamping of $\sigma(v)$. Now, for each non-root variable v with $\text{pred}(v) = \{u_1, \dots, u_k\}, k \geq 1$, for each goal-valid value-changing sequence $\tau' \in \subseteq^*[\tau(v)]$ of v , and each set of such goal-valid value-changing sequences $\{\tau'_1 \in \subseteq^*[\tau(u_1)], \dots, \tau'_k \in \subseteq^*[\tau(u_k)]\}$ of v 's parents, the digraph $G'_e(v)$ is created in three steps. First, we construct a labeled directed graph $G(v)$ capturing information about all sequences of assignments on $\text{pred}(v)$ that can enable n or less value flips of v . The graph $G(v)$ is defined as follows:

1. $G(v)$ consist of $\eta = \max_{\tau' \in \subseteq^*[\tau(v)]} |\tau'|$ nodes.
2. $G(v)$ forms a *0-1 multichain*, i.e., (i) the nodes of the graph are labeled with 0 and 1, starting with 0; (ii) there are no two subsequent nodes with the same label; (iii) for $1 \leq i \leq \eta - 1$, edges from the node i are only to the node $i + 1$.

Observe that such a construction of $G(v)$ promises that the label of the last node will be consistent with the goal value $G[v]$ if such is specified.

3. The nodes of $G(v)$ are denoted precisely by the elements of the longest goal-valid value-changing sequence $\tau' \in \subseteq^*[\tau(v)]$, that is, 0_v^i stands for the i th 0-labeled node in $G(v)$.
4. Suppose that there are m actions in A_v that, under different preconditions, change the value of v from 0_v to 1_v . In this case, for each i , there are m edges from 0_v^i to 1_v^i , and $|A_v| - m$ edges from 1_v^i to 0_v^{i+1} . Each such edge e is labeled with the cost of the corresponding action, as well as with the prevail conditions of that action, which is a k -tuple of the values of u_1, \dots, u_k . This compound label of e is denoted by $l(e)$, and the prevail condition and cost parts of $l(e)$ are henceforth denoted by $\text{prv}(e)$ and $\text{cost}(e)$, respectively.

Informally, the graph $G(v)$ captures information about all *potentially possible* executions of the actions in A_v along a cost-optimal plan for Π . Each path from the source node of $G(v)$ uniquely corresponds to one such an execution. Although the number of these alternative executions may be exponential in n , their graphical representation via $G(v)$ is compact—the number of edges in $G(v)$ is $O(n \cdot |A_v|)$. Note that the information about the number of times each action in A_v can be executed is not captured by $G(v)$. The following two steps add this essential information into the graphical structure.

At the second step, the digraph $G(v) = (V, E)$ 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), but with the labels corresponding to all possible assignments of the elements of τ'_1, \dots, τ'_k to $\text{prv}(e)$. Finally, we set $V' = V \cup \{s_v, t_v\}$, and add a single edge labeled with the first elements of τ'_1, \dots, τ'_k and zero cost (that is, $\| \langle 0_{u_1}^1 \dots 0_{u_k}^1 \rangle, 0 \|$) from s_v to the original source node 0_v^1 , plus a single edge labeled with the last elements of τ'_1, \dots, τ'_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 value-changing sequences τ'_1, \dots, τ'_k on the base digraph $G(v)$.

At the third step, a digraph $G'_e(v) = (V'_e, E'_e)$ is constructed from $G'(v)$ as follows.

- (i) The nodes V'_e correspond to the *edges* of $G'(v)$.
- (ii) The edges $(v_e, v_{e'}) \in E'_e$ correspond to all pairs of immediately consecutive edges $e, e' \in E'$ such that, for $1 \leq i \leq k$, either $\text{prv}(e)[u_i] = \text{prv}(e')[u_i]$, or $\text{prv}(e')[u_i]$ appears after $\text{prv}(e)[u_i]$ along τ'_i .
- (iii) Each edge $(v_e, v_{e'}) \in E'_e$ is weighted with $\text{cost}(e')$.

```

procedure polytree-k-indegree( $\Pi = (V, A, I, G)$ )
  takes a problem  $\Pi \in \mathbf{P}_b$ 
  returns a problem  $\text{COP}_\Pi$  over tree-structured constraint network
create a set of variables  $\mathcal{X}$  and set their domains as in Eq. 1
create a set of functions  $\mathcal{F} = \{\varphi_v \mid v \in V\}$  with scopes as in Eq. 2
for each  $v \in V$  do
  if  $\text{pred}(v) = \emptyset$  then
    specify  $\varphi_v$  according to Eq. 3
  elseif  $\text{pred}(v) = \{u_1, \dots, u_k\}$  then
    construct graph  $G(v)$ 
    for each  $k$ -tuple  $\tau'_1 \in \succeq^*[\tau(u_1)], \dots, \tau'_k \in \succeq^*[\tau(u_k)]$  do
      construct graph  $G'(v)$  from graph  $G(v)$  and sequences  $\tau'_1, \dots, \tau'_k$ 
      construct graph  $G'_e(v)$  from graph  $G'(v)$ 
      for each goal-valid sequence  $\tau' \in \succeq^*[\tau(v)]$  do
         $\pi :=$  minimal-cost path of  $|\tau'| - 1$  edges
          from the source node  $\langle 0_{u_1} \dots 0_{u_k} \rangle$  of  $G'_e(v)$ 
        if returned  $\pi$  then
           $\varphi_v(\tau', \tau'_1, \dots, \tau'_k) := \text{cost}(\pi)$ 
        else
           $\varphi_v(\tau', \tau'_1, \dots, \tau'_k) := \infty$ 
        endif
      endfor
    endfor
  endfor
endif
return  $\text{COP}_\Pi := (\mathcal{X}, \mathcal{F})$  with global objective  $\min \sum_{\varphi \in \mathcal{F}} \varphi(\mathcal{X})$ 

```

Figure 1: Algorithm for constructing a constraint optimization problem COP_Π over tree-structured constraint network for \mathbf{P}_b . The equation numbers refer to the equations in (Katz & Domshlak, 2007b).

The overall algorithm for constructing a constraint optimization problem COP_Π over tree-structured constraint network is depicted in Figure 1. Now, for each $\varphi \in \mathcal{F}$ and for each assignment to the scope of φ , we define a linear program satisfying Eq. 13 as follows. For each planning variable v with $\text{pred}(v) = \emptyset$, and each of its goal-valid (time-stamped) value-changing sequences $\tau' \in \succeq^*[\tau(v)]$, the constraints set as in Eq. 13 corresponding to $\bar{\varphi} = \varphi_v(\tau')$ is

$$z_{\bar{\varphi}} \leq \left\lfloor \frac{|\tau'|}{2} \right\rfloor \cdot w_a + \left\lfloor \frac{|\tau'| - 1}{2} \right\rfloor \cdot w_{a'},$$

for each $a, a' \in A_v$ such that $\text{post}(a)[v] = 1$ and $\text{post}(a')[v] = 0$. Next, for each planning variable v with $\text{pred}(v) = \{u_1, \dots, u_k\}, k \geq 1$, for each goal-valid value-changing sequence $\tau' \in \succeq^*[\tau(v)]$ of v , and each set of such goal-valid value-changing sequences $\{\tau'_1 \in \succeq^*[\tau(u_1)], \dots, \tau'_k \in \succeq^*[\tau(u_k)]\}$ of v 's parents, the constraints set corresponding to $\bar{\varphi} = \varphi_v(\tau', \tau'_1, \dots, \tau'_k)$ is the set of shortest-path constraints of the digraph $G'_e(v)$ as in Eq. 7. The union of all the constraint sets above together with the LP formulation as in Eq. 10 results in LP satisfying Eq. 13. ■

```

procedure polytree-1-dep( $\Pi = (V, A, I, G)$ )
  takes a problem  $\Pi \in \mathbf{P}(1)$ 
  returns a problem  $\text{COP}_\Pi$  over tree-structured constraint network
create a set of variables  $\mathcal{X}$  as in Eqs. 19-20
create a set of functions  $\mathcal{F} = \{\varphi_x \mid x \in \mathcal{X}\}$  with scopes as in Eq. 21
for each  $x \in \mathcal{X}$  do
  specify  $\varphi_x$  according to Eqs. 22-36
endfor
return  $\text{COP}_\Pi := (\mathcal{X}, \mathcal{F})$  with global objective  $\min \sum_{\varphi \in \mathcal{F}} \varphi(\mathcal{X})$ 

```

Figure 2: Algorithm for constructing a constraint optimization problem COP_Π over tree-structured constraint network for $\mathbf{P}(1)$. The equation numbers refer to the equations in (Katz & Domshlak, 2007b).

Theorem 9 *Cost-optimal planning for any task Π in $\mathbf{P}(1)$ is poly-time reducible to a compact and tree-structured constraint optimization problem satisfying Eq. 13.*

Proof: We present here a proof which is based on the construction of the constraint optimization problem in (Katz & Domshlak, 2007b) (Section 5.2, pages 233 – 242). The overall algorithm for constructing a constraint optimization problem COP_Π over tree-structured constraint network is depicted in Figure 2. Now, for each $\varphi \in \mathcal{F}$ and for each assignment to the scope of φ , we define a linear program satisfying Eq. 13 exploiting the fact that

- (i) each of the constraints on $\bar{\varphi}$ given by Equations 22-36 in (Katz & Domshlak, 2007b) is of the form $\bar{\varphi} = \min \{\psi \mid \psi \in \Psi\}$, and
- (ii) each such constraint can be replaced by maximizing $\bar{\varphi}$ under the (now linear) constraints $\{\bar{\varphi} \leq \psi \mid \psi \in \Psi\}$.

For example, for each planning variable v with $\text{pred}(v) = \emptyset$, and each of its goal-valid value-changing sequences $\sigma \in \succeq^*[\sigma(v)]$, if $\bar{\varphi} = \varphi_v(\sigma)$, then we create the constraints set according to Eq. 22 in (Katz & Domshlak, 2007b) as

$$z_{\bar{\varphi}} \leq \left\lceil \frac{|\sigma| - 1}{2} \right\rceil \cdot w_a + \left\lfloor \frac{|\sigma| - 1}{2} \right\rfloor \cdot w_{a'},$$

for each $a, a' \in A_v$ such that $\text{post}(a)[v] = 1$ and $\text{post}(a')[v] = 0$. The rest of the constraint sets for $z_{\bar{\varphi}} \in \vec{z}$ are constructed in the same manner, according to Eq. 23-36 in (Katz & Domshlak, 2007b). The union of all these constraint sets together with LP formulation as in Eq. 10 results in LP satisfying Eq. 13. ■

8. Discussion

Numerous recent works have suggested that additive ensembles of admissible heuristics is a powerful tool for heuristic-search systems. However, the action-cost partitioning parameter of such ensembles kept the “how to add (if at all)” question totally open. Here we described a procedure that closes this question for arbitrary ensembles of all known to us abstraction-based heuristics such as PDBs, constrained PDBs, merge-and-shrink abstractions, fork-decomposition structural patterns, and structural patterns based on tractable constraint optimization. The procedure is based on a linear-programming formulation of the optimization problem: given a classical planning task, a

forward-search state, and a set of abstraction-based admissible heuristics, construct an optimal additive composition of these heuristics with respect to the given state. Most importantly, the time complexity of our procedure is polynomial for arbitrary ensembles of all the above abstraction-based heuristics. We now outline some of the (in our opinion) more important challenges for future work.

- **Structure optimization.** Probably the most important issue that remains almost entirely open is this of “structure optimization”. While our framework optimizes the composition of a *given* set of TG-structures, ultimately we would like to move to even more parametric such ensembles, allowing flexibility in the actual choice of TG-structures. For instance, it would clearly help to know what PDBs should (optimally) be added to the ensemble, what domain abstractions should (optimally) be performed on the roots of the inverted forks and forks, what polytrees should (optimally) span the causal graph of the problem, etc. Note that the first step in this direction has already been made between the lines of this paper—an immediate corollary of Theorem 5 is that, for any forward-search state s , and any *fixed upper bound on the size of the PDBs*, one can construct in polynomial time the actual *optimal (for s) pattern-database ABS-ensemble*, and this simply by running LP-optimization over the ensemble of all possible such PDBs.
- **Additive m -reachability.** As we mentioned, the seminal m -reachability heuristics h^m are not covered by our framework. While computing a single h^m heuristic for a fixed m is poly-time, h^m is not based on a problem abstraction (even in the very permissive sense of Definition 1)—the state-graph over which h^m is computed is an AND/OR-graph (and not an OR-graph such as transition graphs), each original problem state is mapped to a *set* of abstract states (and not to a concrete such state), and the actual computation of h^m corresponds to computing a critical tree (and not a shortest path) to the goal. Tangentially, the problem of computing a critical-tree in an AND/OR-graph does not appear to have an LP reformulation. Hence, the complexity of computing optimal additive h^m heuristics is still open and very much interesting.
- **LP-encodings for “double relaxations”.** The basic idea of LP-optimizing heuristic composition naturally extends also to *intractable* planning relaxations that admit “second-order” LP-relaxations. For instance, some intractable planning relaxations formalizable via sounds and complete integer-valued LPs (such as the deletes-ignoring relaxation underlying h^+ , or more recent action-ordering relaxation of van den Briel (2007)) appear to be quite natural such candidates. Things, however, are more complicated than that because, very roughly, (i) Definition 4 requires a very specific type of LP-encodings (satisfying Eq. 5), and (ii) none of the known to us ILP-to-LP “second-order” relaxations appear to be of that type. Hence, developing “Eq. 5 friendly” LP-relaxations for h^+ , action-ordering relaxation, and other informative yet intractable planning relaxations is now definitely of interest.
- Finally, we would like to address an almost immediate source of skepticism with respect to the practicality of our optimization procedure—using it requires solving a large LP at every search node, while typically such per-node computations are expected to be of *low* polynomial time. We believe, however, that the superior informativeness of the optimal additive heuristics has a clear potential to eventually outweigh the cost of heuristic computation due to substantial reductions in the number of expanded search nodes. In other words, while the informal notion of “low polynomial time” changes with the progress of hardware technology, it is widely believed these days that $P \neq NP$, and thus reducing the amount of expanded nodes is still the most important objective of the heuristic-search research in the long run.

References

- Bäckström, C., & Nebel, B. (1995). Complexity results for SAS⁺ planning. *Computational Intelligence*, 11(4), 625–655.

- Bylander, T. (1994). The computational complexity of propositional STRIPS planning. *Artificial Intelligence*, 69(1-2), 165–204.
- Culberson, J., & Schaeffer, J. (1998). Pattern databases. *Computational Intelligence*, 14(4), 318–334.
- Dechter, R. (2003). *Constraint Processing*. Morgan Kaufmann.
- Edelkamp, S. (2001). Planning with pattern databases. In *Proceedings of the European Conference on Planning (ECP)*, pp. 13–34.
- Felner, A., Korf, R. E., & Hanan, S. (2004). Additive pattern database heuristics. *Journal of Artificial Intelligence Research*, 22, 279–318.
- Haslum, P., Bonet, B., & Geffner, H. (2005). New admissible heuristics for domain-independent planning. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI)*, pp. 1163–1168.
- Haslum, P., Botea, A., Helmert, M., Bonet, B., & Koenig, S. (2007). Domain-independent construction of pattern database heuristics for cost-optimal planning. In *Proceedings of the 19th National Conference on Artificial Intelligence (AAAI)*, pp. 1007–1012.
- Helmert, M., Haslum, P., & Hoffmann, J. (2007). Flexible abstraction heuristics for optimal sequential planning. In *Proceedings of the 17th International Conference on Automated Planning and Scheduling (ICAPS)*.
- Katz, M., & Domshlak, C. (2007a). Structural patterns heuristics. In *ICAPS-07 Workshop on Heuristics for Domain-independent Planning: Progress, Ideas, Limitations, Challenges*.
- Katz, M., & Domshlak, C. (2007b). Structural patterns of tractable sequentially-optimal planning. In *Proceedings of the 17th International Conference on Automated Planning and Scheduling (ICAPS)*.
- Pearl, J. (1984). *Heuristics - Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley.
- Schrijver, A. (1998). *Theory of Linear and Integer Programming*. John Wiley & Sons.
- van den Briel, M., Benton, J., Kambhampati, S., & Vossen, T. (2007). An LP-based heuristic for optimal planning. In *Proceedings of the 13th International Conference on Principles and Practice of Constraint Programming (CP)*, pp. 651–665, Providence, RI.
- Yang, F., Culberson, J., Holte, R., Zahavi, U., & Felner, A. (2008). A general theory of additive state space abstractions. *Journal of Artificial Intelligence Research*, 32, 631–662.