LP-based Heuristics for Cost-optimal Planning

Florian Pommerening and **Gabriele Röger** and **Malte Helmert**
Universität Basel
Basel, Switzerland
{florian.pommerening,gabriele.roeger,malte.helmert}@unibas.ch

Blai Bonet
Universidad Simón Bolívar
Caracas, Venezuela
bonet@ldc.usb.ve

Abstract

Many heuristics for cost-optimal planning are based on linear programming. We cover several interesting heuristics of this type by a common framework that fixes the objective function of the linear program. Within the framework, constraints from different heuristics can be combined in one heuristic estimate which dominates the maximum of the component heuristics. Different heuristics of the framework can be compared on the basis of their constraints. With this new method of analysis, we show dominance of the recent LP-based state-equation heuristic over optimal cost partitioning on single-variable abstractions. We also show that the previously suggested extension of the state-equation heuristic to exploit safe variables cannot lead to an improved heuristic estimate. We experimentally evaluate the potential of the proposed framework on an extensive suite of benchmark tasks.

Introduction

Several recent heuristics (van den Briel et al. 2007; Karpas and Domshlak 2009; Bonet 2013; Pommerening, Röger, and Helmert 2013) for cost-optimal planning show that it is feasible and beneficial to obtain heuristic estimates by solving a linear program for every state.

However, they obtain their constraints from very different sources of information. Karpas and Domshlak use landmarks, Pommerening et al. exploit information from abstraction heuristics, and van den Briel et al. and Bonet base their linear program on network flows for the state variables.

We will show that all these approaches can be covered by a single framework that fixes the optimization function of the linear program. This will turn out to be beneficial in two ways: first, we can combine the information from various sources in one heuristic which dominates the individual ones. Second, it provides us with the possibility to compare heuristics on the basis of their constraints, which will lead us to some interesting new theoretical results.

We start by introducing the SAS$^+$ planning formalism and our new framework, which is based on *operator-counting constraints*. Afterwards, we present a wide range of such constraints and explain how they can be used to express existing heuristics. We then prove some theoretical results on interesting connections between the heuristics and end with an experimental study and conclusions.

SAS$^+$ Planning

We consider SAS$^+$ planning tasks with non-negative operator costs. A *task* is a tuple $\Pi = \langle \mathcal{V}, \mathcal{O}, s_I, s_G, \mathit{cost} \rangle$ where $\mathcal{V}$ is a finite set of *variables*. Each variable $V \in \mathcal{V}$ has a finite domain D_V. A (partial) *state* s is a (partial) variable assignment over $\mathcal{V}$. We write $\mathit{vars}(s)$ for the domain of definition of s and $s[V]$ for the value of V in s. The notation $s[V] = \bot$ means that $V \notin \mathit{vars}(s)$. A partial state s is *consistent* with a partial state s' if $s[V] = s'[V]$ for all $V \in \mathit{vars}(s')$. We say that an *atom* $V = v$ is true in state s iff $s[V] = v$.

Each operator o in the finite set of *operators* $\mathcal{O}$ is associated with a precondition $\mathit{pre}(o)$ and an effect $\mathit{eff}(o)$, which are both partial variable assignments over $\mathcal{V}$. We require that $V = v$ cannot be both a precondition and an effect of o. This is not a real restriction because such effects would be redundant. The (complete) state s_I is the *initial state* of the task and the partial state s_G describes its *goal*. The *cost function* $\mathit{cost} : \mathcal{O} \to \mathbb{R}_0^+$ assigns a non-negative cost to each operator.

An operator o is *applicable* in a state s if s is consistent with $\mathit{pre}(o)$. The *resulting state* of applying an applicable operator o in state s is the state $\mathit{res}(o, s)$ with

$$\mathit{res}(o, s)[V] = \begin{cases} \mathit{eff}(o)[V] & \text{if } V \in \mathit{vars}(\mathit{eff}(o)) \\ s[V] & \text{otherwise.} \end{cases}$$

A sequence of operators $\pi = \langle o_1, \ldots, o_n \rangle$ is applicable in state s_0 if there are states $s_1, \ldots, s_n$ with $s_i = \mathit{res}(o_i, s_{i-1})$ for $1 \leq i \leq n$. The resulting state of this application is $\mathit{res}(\pi, s_0) = s_n$. The cost of π is the sum of its operator costs $\mathit{cost}(\pi) = \sum_{i=1}^{n} \mathit{cost}(o_i)$.

For state s, an *s-plan* is an operator sequence π applicable in s such that $\mathit{res}(\pi, s)$ is consistent with s_G. An s_I-plan is just called a *plan*. A plan with minimal cost is called *optimal*.

A function h that maps states to non-negative numbers (or ∞) is called a *heuristic*. A heuristic h is called *admissible* if $h(s) \leq h^*(s)$ for all states s, where $h^*(s)$ is the cost of an optimal s-plan (or ∞ if no s-plan exists).

Operator-counting Constraints

Several recently proposed heuristics (van den Briel et al. 2007; Bonet 2013; Pommerening, Röger, and Helmert 2013)

are based on linear programs of similar form. They formalize constraints that must be satisfied by every plan π and use a variable Y_o for each operator o such that setting Y_o to the number of occurrences of o in π satisfies the constraints.[1]

We will show that these heuristics (and some other ones) can be covered by a common framework based on the notion of *operator-counting constraints*:

Definition 1 (Operator-counting constraints) *Let Π be a planning task with operator set $\mathcal{O}$, and let s be one of its states. Let $\mathcal{Y}$ be a set of non-negative real-valued and integer variables, including an integer variable Y_o for each operator $o \in \mathcal{O}$ along with any number of additional variables. The variables Y_o are called* operator-counting variables.

If π is an s-plan, we denote the number of occurrences of operator $o \in \mathcal{O}$ in π with Y_o^π. A set of linear inequalities over $\mathcal{Y}$ is called an operator-counting constraint *for s if for every s-plan π, there exists a feasible solution with $\mathsf{Y}_o = Y_o^\pi$ for all $o \in \mathcal{O}$.*

A constraint set *for s is a set of operator-counting constraints for s where the only common variables between constraints are the operator-counting variables.*

As an example, the inequality $\mathsf{Y}_{o_1} - 2\mathsf{Y}_{o_2} \geq 0$ is an operator-counting constraint expressing that in every plan, o_1 must occur at least twice as often as o_2. The set of inequalities $\{\mathsf{Y}_{o_1} - \mathsf{Z} \geq 2, \mathsf{Y}_{o_2} + \mathsf{Z} \geq 1\}$ with auxiliary integer variable Z is another operator-counting constraint, which expresses in a roundabout way that the total number of occurrences of o_1 and o_2 in every plan is at least 3. The two operator-counting constraints form a constraint set because they do not share any auxiliary variables.

Definition 2 (Operator-counting integer/linear program) *The* operator-counting integer program IP_C *for constraint set C is:*

$$\textit{Minimize} \sum_{o \in \mathcal{O}} \textit{cost}(o)\mathsf{Y}_o \;\; \textit{subject to} \;\; C.$$

The operator-counting linear program LP_C *is the LP-relaxation of* IP_C.

From Definition 1, if π is a plan, then there exists a solution to both $\mathrm{IP}_\mathcal{C}$ and $\mathrm{LP}_\mathcal{C}$ where $\mathsf{Y}_o = Y_o^\pi$ for all $o \in \mathcal{O}$. The cost of the plan is $\textit{cost}(\pi) = \sum_{o \in \mathcal{O}} \textit{cost}(o) \cdot Y_o^\pi$, and hence the optimal plan cost is an upper bound for the objective value of the IP/LP. This allows us to define the following admissible heuristics:

Definition 3 (IP and LP heuristic) *Let Π be a planning task, and let $\mathcal{C}$ be a function that maps states s of Π to constraint sets for s.*

The IP heuristic $h^{\mathrm{IP}}_\mathcal{C}(s)$ *is the objective value of the integer program* $\mathrm{IP}_{\mathcal{C}(s)}$. *The* LP heuristic $h^{\mathrm{LP}}_\mathcal{C}(s)$ *is the objective value of the linear program* $\mathrm{LP}_{\mathcal{C}(s)}$. *Infeasible IPs/LPs are treated as having an objective value of ∞.*

Note that the requirement that an operator-counting constraint must have a feasible solution with $\mathsf{Y}_o = Y_o^\pi$ for *every* plan π is stricter than necessary for admissibility. It is sufficient that whenever a solution exists, there is one optimal plan π^* such that *all* operator-counting constraints have a feasible solution with $\mathsf{Y}_o = Y_o^{\pi^*}$.

If all operator costs of a planning task are integer, we can obviously improve the LP heuristic estimate without losing admissibility by rounding up to the nearest integer.

Since adding operator-counting constraints can only reduce the set of feasible solutions for an operator-counting integer/linear program, the resulting heuristic estimates cannot decrease by the use of additional constraints.

Proposition 1 (Dominance) *Let $\mathcal{C}, \mathcal{C}'$ be functions that map states s of Π to constraint sets for s such that $\mathcal{C}(s) \subseteq \mathcal{C}'(s)$ for all states s. Then the IP/LP heuristic for $\mathcal{C}'$ dominates the respective heuristic for $\mathcal{C}$: $h^{\mathrm{IP}}_\mathcal{C} \leq h^{\mathrm{IP}}_{\mathcal{C}'}$ and $h^{\mathrm{LP}}_\mathcal{C} \leq h^{\mathrm{LP}}_{\mathcal{C}'}$.*

Types of Operator-counting Constraints

In this section we describe four types of operator-counting constraints that capture different state-of-the-art heuristics for optimal planning.

Landmark Constraints

A *disjunctive action landmark* (Zhu and Givan 2003; Helmert and Domshlak 2009) for a state s is a set of operators of which at least one must be part of any s-plan.

Using linear programming to derive heuristic estimates from landmarks was introduced by Karpas and Domshlak (2009) as cost partitioning for landmarks. The LP formulation was improved by Keyder, Richter, and Helmert (2010). Bonet and Helmert (2010) introduced an alternative LP representation that directly fits into the operator-counting constraint framework and showed that it is the dual of the representation by Keyder et al.

Strengthening other heuristics with landmarks is not a new idea: Domshlak, Katz, and Lefler (2012) propose it for abstraction heuristics and Bonet (2013) for the LP-based state-equation heuristic. He uses the same constraints as Bonet and Helmert (2010):

Definition 4 *Let $L \subseteq \mathcal{O}$ be a disjunctive action landmark for state s of task Π. The* landmark constraint $c^{\mathrm{lm}}_{s,L}$ *for L is*

$$\sum_{o \in L} \mathsf{Y}_o \geq 1.$$

Since at least one operator of the landmark occurs in every s-plan, landmark constraints clearly meet the requirements of operator-counting constraints.

Net Change Constraints

Bonet (2013) introduces the state-equation heuristic h^{SEQ} by relating planning tasks to Petri nets and deriving constraints based on the net change of the number of tokens in the Petri net locations caused by the firing of transitions. Here, we present the general ideas behind h^{SEQ} again by working on the planning task directly, without the translation to Petri nets. This will not only lead us to a deeper understanding but also to a wider class of constraints.

[1] For the post-hoc optimization heuristic (Pommerening, Röger, and Helmert 2013) Y_o instead is the cost incurred by o in π, which is the number of occurrences of o in π multiplied by $\textit{cost}(o)$.

We define the *net change* of an atom $V = v$ from a state s to a state s' as the change of its truth value, where 1 denotes that the atom becomes true, 0 that it is unchanged, and -1 that it becomes false.

Definition 5 *The* net change of an atom $V = v$ from a state s to a state s' *is*

$$\mathit{netchange}_{V=v}^{s \to s'} = \begin{cases} 1 & \text{if } s[V] \neq v \text{ and } s'[V] = v \\ -1 & \text{if } s[V] = v \text{ and } s'[V] \neq v \\ 0 & \text{otherwise.} \end{cases}$$

We say that an operator o applied in state s *produces* an atom $V = v$ if $s[V] \neq v$ and $\mathit{res}(o, s)[V] = v$ and that it *consumes* the atom if $s[V] = v$ and $\mathit{res}(o, s)[V] \neq v$.

Obviously, the net change of the atom from state s to the successor state $\mathit{res}(o, s)$ is 1 if o produces the atom, -1 if it consumes it and 0 otherwise. We would like to retain this more operator-centric view:

Definition 6 *Let o be an operator and $\pi = \langle o_1, \ldots, o_n \rangle$ be applicable in state s. The* net change that o induces for atom $V = v$ in s *is*

$$\begin{aligned} \mathit{netchange}(o)_{V=v}^{s} &= \mathit{netchange}_{V=v}^{s \to \mathit{res}(o,s)} \\ &= \begin{cases} 1 & \text{if } o \text{ applied in } s \text{ produces } V = v \\ -1 & \text{if } o \text{ applied in } s \text{ consumes } V = v \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

The accumulated net change *induced by sequence π is*

$$\mathit{netchange}(\pi)_{V=v}^{s} = \sum_{i=1}^{n} \mathit{netchange}(o_i)_{V=v}^{\mathit{res}(\langle o_1, \ldots, o_{i-1} \rangle, s)}.$$

It is obvious that we do not need to consider the intermediate states of the application of an operator sequence π but can directly compare the initial and the resulting state:

Proposition 2 *The accumulated net change induced by the application of an operator sequence π in state s is the net change from s to the resulting state $\mathit{res}(\pi, s)$:*

$$\mathit{netchange}(\pi)_{V=v}^{s} = \mathit{netchange}_{V=v}^{s \to \mathit{res}(\pi,s)}.$$

We would like to use this information to derive operator-counting constraints.

Informally speaking, the accumulated net change of an operator sequence sums up all operator applications that produce the atom and subtracts the number of operator applications that consume the atom. Without knowing the state in which an operator is applied, we cannot in general decide whether an operator application consumes or produces an atom. However, we can give upper and lower bounds on the induced net change for arbitrary plans depending on their operator counts. To do so, we distinguish four disjoint classes of operators for each atom:

Operators that *always produce* atom $V = v$:

$$\begin{aligned} \mathit{AP}_{V=v} = \{o \in \mathcal{O} \mid \mathit{eff}(o)[V] = v \text{ and} \\ \mathit{pre}(o)[V] = v' \text{ with } v' \neq v\} \end{aligned}$$

Operators that *sometimes produce* atom $V = v$:

$$\begin{aligned} \mathit{SP}_{V=v} = \{o \in \mathcal{O} \mid \mathit{eff}(o)[V] = v \text{ and} \\ \mathit{pre}(o)[V] \text{ is undefined}\} \end{aligned}$$

Operators that *always consume* atom $V = v$:

$$\begin{aligned} \mathit{AC}_{V=v} = \{o \in \mathcal{O} \mid \mathit{eff}(o)[V] = v' \text{ with } v \neq v' \\ \text{and } \mathit{pre}(o)[V] = v\} \end{aligned}$$

Operators that *sometimes consume* atom $V = v$:

$$\begin{aligned} \mathit{SC}_{V=v} = \{o \in \mathcal{O} \mid \mathit{eff}(o)[V] = v' \text{ with } v \neq v' \\ \text{and } \mathit{pre}(o)[V] \text{ is undefined}\} \end{aligned}$$

Operators that do not fall into one of these classes never change the truth value of the atom.

We can use these definitions to give bounds on the net change induced by an operator which do not depend on the state s:

$$\mathit{netchange}(o)_{V=v}^{s} \in \begin{cases} \{1\} & \text{if } o \in \mathit{AP}_{V=v} \\ \{0, 1\} & \text{if } o \in \mathit{SP}_{V=v} \\ \{-1\} & \text{if } o \in \mathit{AC}_{V=v} \\ \{-1, 0\} & \text{if } o \in \mathit{SC}_{V=v} \\ \{0\} & \text{otherwise} \end{cases}$$

This justifies the following proposition:

Proposition 3 *The accumulated net change induced by the application of operator sequence π in s can be bounded from above and below as follows:*

$$\begin{aligned} \sum_{o \in \mathit{AP}_{V=v}} Y_o^\pi + \sum_{o \in \mathit{SP}_{V=v}} Y_o^\pi - \sum_{o \in \mathit{AC}_{V=v}} Y_o^\pi &\geq \mathit{netchange}(\pi)_{V=v}^{s} \\ &\geq \sum_{o \in \mathit{AP}_{V=v}} Y_o^\pi - \sum_{o \in \mathit{AC}_{V=v}} Y_o^\pi - \sum_{o \in \mathit{SC}_{V=v}} Y_o^\pi. \end{aligned}$$

Note that if both classes $\mathit{SP}_{V=v}$ and $\mathit{SC}_{V=v}$ are empty, the inequalities become an equality.

We can easily specify the possible net changes that a variable can accumulate on its way to an arbitrary goal state:

$$\mathit{pnc}_{V=v}^{s \to \star} = \begin{cases} \{0, 1\} & \text{if } s_G[V] = \bot \text{ and } s[V] \neq v \\ \{-1, 0\} & \text{if } s_G[V] = \bot \text{ and } s[V] = v \\ \{1\} & \text{if } s_G[V] = v \text{ and } s[V] \neq v \\ \{-1\} & \text{if } s_G[V] = v' \text{ and } s[V] = v \neq v' \\ \{0\} & \text{otherwise} \end{cases}$$

Together with Propositions 2 and 3 this yields constraints that have a feasible solution for every s-plan:

Definition 7 (Net change constraint) *For atom $V = v$ and state s, let $L = \min \mathit{pnc}_{V=v}^{s \to \star}$, and let $U = \max \mathit{pnc}_{V=v}^{s \to \star}$. The*

lower-bound net change constraint $c^{\text{ncl}}_{s,V=v}$ *for atom* $V = v$ *and state* s *is the constraint*

$$\sum_{o \in AP_{V=v}} \mathsf{Y}_o \; + \sum_{o \in SP_{V=v}} \mathsf{Y}_o \; - \sum_{o \in AC_{V=v}} \mathsf{Y}_o \; \geq \; L$$

and the upper-bound net change constraint $c^{\text{ncu}}_{s,V=v}$ *is the constraint*

$$U \; \geq \sum_{o \in AP_{V=v}} \mathsf{Y}_o \; - \sum_{o \in AC_{V=v}} \mathsf{Y}_o \; - \sum_{o \in SC_{V=v}} \mathsf{Y}_o \, .$$

Post-Hoc Optimization Constraints

Post-hoc optimization heuristics (Pommerening, Röger, and Helmert 2013) exploit the fact that sometimes we know that certain operators do not contribute to a heuristic estimate. They give rise to another type of operator-counting constraints:

Definition 8 (Post-hoc optimization constraint) *Let* Π *be a planning task with operator set* $\mathcal{O}$*, let* h *be an admissible heuristic for* Π*, and let* $N \subseteq \mathcal{O}$ *be a set of operators that are* noncontributing *in the sense that* h *is still admissible in a modified planning task where the cost of all operators in* N *is set to* 0*.*

Then the post-hoc optimization constraint $c^{\text{PH}}_{s,h,N}$ *for* h*,* N *and state* s *of* Π *consists of the inequality*

$$\sum_{o \in \mathcal{O} \setminus N} \mathit{cost}(o) \mathsf{Y}_o \; \geq \; h(s) \, .$$

An example of suitable heuristics are pattern database (PDB) heuristics (Culberson and Schaeffer 1998; Edelkamp 2001), which are based on projections of a planning task to a subset (called a *pattern*) of the state variables. Operators that do not modify any variable in the pattern are noncontributing. Due to the importance of PDB heuristics, we give a separate definition for the special case of post-hoc constraints for PDB heuristics:

Definition 9 *For a planning task* Π*, a state* s *and PDB heuristic* h^P *for pattern* P*, the* PDB constraint $c^{\text{pdb}}_{s,P}$ *is the constraint*

$$\sum_{\substack{o \in \mathcal{O} \\ o \text{ affects } P}} \mathit{cost}(o) \mathsf{Y}_o \; \geq \; h^P(s) \, .$$

Cost Partitioning Constraints for Abstractions

The final type of constraints we introduce are more complex than the previous ones because they require auxiliary IP/LP variables, which means that they consist of several inequalities that cannot be considered in isolation. To develop these constraints, we must discuss the notion of cost partitioning for abstraction heuristics.

An *abstraction heuristic* maps each state s of a planning task Π through a homomorphic mapping α to an *abstract state* $\alpha(s)$. The heuristic estimate for s is the cost of the cheapest path from $\alpha(s)$ to an abstract goal state in the transition system induced by α on the transition system of Π.

In general the estimates of several abstraction heuristics can only be combined admissibly by taking their maximum. *Action cost partitioning* allows more informative estimates by modifying the cost function of the abstract systems so that the estimates can be admissibly summed up.

Optimal cost partitioning (Katz and Domshlak 2010) computes the best estimate obtainable by cost partitioning for a given set of abstraction mappings. Unfortunately, the computation is state-dependent and the evaluation in every state is often prohibitively expensive in practice (Pommerening, Röger, and Helmert 2013) although it only takes polynomial time. However, it is still interesting from a theoretical perspective.

For a set $\mathcal{A}$ of abstraction mappings for task Π, let $\mathcal{T}^\alpha = \langle S^\alpha, s^\alpha_I, G^\alpha, T^\alpha \rangle$ denote the transition system induced by $\alpha \in \mathcal{A}$, where S^α is the set of abstract states, s^α_I is the abstract initial state and G^α is the set of abstract goal states. Each transition $\langle s, o, s' \rangle \in T^\alpha$ from state s to state s' is labeled with the operator o that induces it. The subset $\mathit{SCT}^\alpha \subseteq T^\alpha$ contains all *state-changing* transitions $\langle s, o, s' \rangle$ with $s \neq s'$.

The estimate of the optimal cost partitioning heuristic $h^{\text{OCP}}_{\mathcal{A}}$ for $\mathcal{A}$ in state $s \in S$ is the objective value of the following LP or ∞ if the LP is not bounded feasible:

$$\text{Maximize} \sum_{\alpha \in \mathcal{A}} \mathsf{H}^\alpha \text{ subject to}$$

$$\begin{aligned}
\mathsf{D}^\alpha_{s'} &= 0 && \text{for all } \alpha \in \mathcal{A} \text{ and } s' = \alpha(s) \\
\mathsf{D}^\alpha_{s''} &\leq \mathsf{D}^\alpha_{s'} + \mathsf{C}^\alpha_o && \text{for all } \alpha \in \mathcal{A} \text{ and } \langle s', o, s'' \rangle \in \mathit{SCT}^\alpha \\
\mathsf{H}^\alpha &\leq \mathsf{D}^\alpha_{s'} && \text{for all } \alpha \in \mathcal{A} \text{ and } s' \in G^\alpha \\
\sum_{\alpha \in \mathcal{A}} \mathsf{C}^\alpha_o &\leq \mathit{cost}(o) && \text{for all } o \in \mathcal{O}
\end{aligned}$$

with all variables restricted to be non-negative.

Intuitively, variable C^α_o describes the cost of operator o in $\mathcal{T}^\alpha$. Given this cost partitioning, variable $\mathsf{D}^\alpha_{s'}$ measures the cheapest cost to reach s' from $\alpha(s)$ in the abstract system and H^α is the (additive) heuristic estimate for abstraction α.

At first glance, the cost partitioning LP seems unrelated to the operator-counting constraint framework. It is concerned with maximization, not minimization, and it has no operator-counting variables. It is thus not apparent how and if optimal cost partitioning can be integrated with, say, landmark constraints. However, it turns out that the *dual* of the LP (which must have the same objective value) is a direct fit for our framework, offering a new perspective on optimal cost partitioning. This dual LP is of the following form: Minimize $\sum_{o \in \mathcal{O}} \mathit{cost}(o) \mathsf{Y}_o$ subject to the operator-counting constraints $c^{\text{OCP}}_{s,\alpha}$ (defined next) for all abstraction mappings $\alpha \in \mathcal{A}$. For an intuitive interpretation of the auxiliary variables used in $c^{\text{OCP}}_{s,\alpha}$, see the following proof.

Definition 10 *Let* α *be an abstraction mapping for task* Π *and let* $\mathcal{T}^\alpha = \langle S^\alpha, s^\alpha_I, G^\alpha, T^\alpha \rangle$ *be the induced abstract transition system with* $\mathit{SCT}^\alpha \subseteq T^\alpha$ *being the set of state-changing transitions.*

For a state s *of* Π*, the* optimal cost partitioning constraint $c^{\text{OCP}}_{s,\alpha}$ *consists of:*

1. *a* transition count inequality *for each operator $o \in \mathcal{O}$:*

$$\mathsf{Y}_o \geq \sum_{\substack{t \in SCT^\alpha \\ t \text{ labeled with } o}} \mathsf{T}^\alpha_t ,$$

2. *a* goal inequality $\sum_{s' \in G^\alpha} \mathsf{G}^\alpha_{s'} \geq 1$,
3. *a* transition flow inequality *for all abstract states $s' \neq \alpha(s)$ in S^α:*

$$\sum_{\substack{t \in SCT^\alpha \\ t \text{ ends in } s'}} \mathsf{T}^\alpha_t - \sum_{\substack{t \in SCT^\alpha \\ t \text{ starts in } s'}} \mathsf{T}^\alpha_t \geq \begin{cases} \mathsf{G}^\alpha_{s'} & \text{if } s' \in G^\alpha \\ 0 & \text{if } s \notin G^\alpha \end{cases} .$$

Proposition 4 *Optimal cost partitioning constraints as specified in Definition 10 are operator-counting constraints.*

Proof sketch: Let π be an s-plan. To satisfy the inequalities we set Y_o to Y^π_o. $\mathsf{G}^\alpha_{s'}$ is 1 if executing π in $\mathcal{T}^\alpha$ ends in s', and 0 otherwise. The variables T^α_t are set to the number of times that transition t is used when executing π in $\mathcal{T}^\alpha$. ∎

A closer look at the proof shows that the instantiation of the LP variables for a given plan only uses integer values. This means that the constraints remain sound when interpreting $c^{\text{OCP}}_{s,\alpha}$ as an *integer program*. This results in a heuristic that *dominates* Katz and Domshlak's optimal cost partitioning. (Of course, the new heuristic is not known to be polynomial-time computable.) It is not hard to find examples that show that the dominance is strict. This relationship parallels the relationship between IP-based landmark heuristics based on *hitting sets* and LP-based landmark heuristics based on cost partitioning (Bonet and Helmert 2010).

Relationship to Existing Heuristics

We now show how several existing heuristics can be expressed within the operator-counting constraint framework. Full proofs would require a detailed presentation of these previous heuristics, so due to space limitations we only sketch the basic ideas.

State-equation heuristic The state-equation heuristic (Bonet 2013) is defined via an LP that directly fits our framework. Bonet also suggests a safety-based improvement and a landmark-based improvement of h^{SEQ}. A close look at the constraints reveals the following connections to our framework:

Proposition 5 *For state s, let $\mathcal{C}(s)$ denote the set of lower-bound net change constraints for s and all atoms. Then the state-equation heuristic h^{SEQ} equals the LP heuristic $h^{\text{LP}}_{\mathcal{C}}$.*

The safety-based improvement of h^{SEQ} corresponds to extending each set $\mathcal{C}(s)$ with the upper-bound net change constraints for s and all atoms $V = v$ of variables V with $SP_{V=v'} = \emptyset$ for all $v' \in D_V$.

The landmark-based improvement of h^{SEQ} corresponds to extending $\mathcal{C}(s)$ with the landmark constraints for the given landmarks.

Post-hoc optimization heuristic The definition of the post-hoc optimization heuristic (Pommerening, Röger, and Helmert 2013) involves a variable merging step which speeds up its computation but has no influence on the heuristic estimate. If we omit this merging step, it is defined by an LP which minimizes the objective function $\sum_{o \in \mathcal{O}} \mathsf{C}_o$, where intuitively C_o represents the total cost incurred by operator o in a plan. We can easily transform this LP into the required form by replacing each occurrence of C_o with $cost(o)\mathsf{Y}_o$. Then the following proposition is obvious:

Proposition 6 *Let $h^{\text{PhO}}_{\mathcal{H},\mathcal{N}}$ be the post-hoc optimization heuristic for a set of heuristics $\mathcal{H}$ where $\mathcal{N}(h)$ denotes the noncontributing operators of $h \in \mathcal{H}$. For state s, let $\mathcal{C}_{\mathcal{H},\mathcal{N}}(s) = \{c^{\text{PH}}_{s,h,\mathcal{N}(h)} \mid h \in \mathcal{H}\}$. Then $h^{\text{PhO}}_{\mathcal{H},\mathcal{N}} = h^{\text{LP}}_{\mathcal{C}_{\mathcal{H},\mathcal{N}}}$.*

Landmark heuristic with optimal cost partitioning Optimal cost partitioning for landmarks (Karpas and Domshlak 2009) can be expressed in our framework. As already indicated in the motivation of the landmark constraints, this follows from the related work by Keyder, Richter, and Helmert (2010) and Bonet and Helmert (2010).

Proposition 7 *For state s, let $\mathcal{C}(s)$ be a set of landmark constraints for s. Then $h^{\text{LP}}_{\mathcal{C}}(s) = h^{\text{LM}}_{\text{opt}}(s)$, where $h^{\text{LM}}_{\text{opt}}$ is the landmark heuristic with optimal cost partitioning using the same landmarks as in $\mathcal{C}$.*

The LM-cut heuristic (Helmert and Domshlak 2009) computes a cost partitioning for a set of action landmarks it finds. Bonet (2013) proposed to use these landmarks for specifying landmark constraints in the sense of this paper. From the previous proposition and Proposition 1, it follows that LP heuristics using these constraints dominate LM-cut.

Optimal cost partitioning for abstractions We derived the optimal cost partitioning constraints from the dual of the LP which defines the optimal cost partitioning heuristic. As a bounded feasible LP and its dual have the same optimal value, the heuristic fits our framework.

Proposition 8 *Let $\mathcal{A}$ be a set of abstractions. For state s, let $\mathcal{C}(s)$ be the constraints $\{c^{\text{OCP}}_{s,\alpha} \mid \alpha \in \mathcal{A}\}$. Then $h^{\text{OCP}}_{\mathcal{A}} = h^{\text{LP}}_{\mathcal{C}_{\mathcal{A}}}$.*

In all cases above where $h^{\text{LP}}_{\mathcal{C}} = h'$ for a constraint function $\mathcal{C}$ and some existing heuristic h', it of course follows that the corresponding IP heuristic $h^{\text{IP}}_{\mathcal{C}}$ dominates h'.

Net Change vs. Atomic Projections

One of the interesting features of the state-equation heuristic h^{SEQ} is that it appears to fall outside the common concepts for planning heuristics like abstraction or delete relaxation (Bonet 2013). Here, we present a first theoretical result that compares it to an established heuristic. Specifically, we prove that the LP heuristic induced by lower-bound net change constraints (shown in the previous section to be equal to h^{SEQ}) dominates optimal cost partitioning on single-variable projections.

In detail, let the set of abstractions Sys^1 denote all projections on single goal variables. (Projections to non-goal variables are not interesting because all abstract states are goal states.) A projection on a variable V maps state s to the

abstract state $\{V \mapsto s[V]\}$, which we identify with $s[V]$ in the following for simplicity.

We will show that the state equation heuristic dominates the optimal cost partitioning for Sys^1.

Lemma 1 *Let $\Pi = \langle \mathcal{V}, \mathcal{O}, s_I, s_G, cost \rangle$ be a planning task, let $V \in vars(s_G)$ be one of its goal variables, and let α be the projection to V. Let $C = \{c^{\text{ncl}}_{s,V=v} \mid v \in D_V\}$ be the constraint set consisting of all lower-bound net change constraints for V. If there is a feasible (real or integer) solution for C (where Y_o denotes the value of variable Y_o), then the following assignment is a feasible (real or integer) solution for the optimal cost partitioning constraint $c^{\text{OCP}}_{s,\alpha}$:*

$$\mathsf{Y}_o = Y_o \quad \textit{for all } o \in \mathcal{O}$$
$$\mathsf{G}^\alpha_v = 1 \quad \textit{for } v = s_G[V]$$
$$\mathsf{T}^\alpha_{\langle v,o,v' \rangle} = \begin{cases} Y_o & \textit{if } \textit{pre}(o)[V] \textit{ is defined or } v = s[V] \\ 0 & \textit{otherwise} \end{cases}$$
$$\textit{for all } \langle v, o, v' \rangle \in \textit{SCT}^\alpha.$$

Proof sketch: The transition count inequality for operator o is satisfied because there is at most one transition $t = \langle v, o, v' \rangle \in \textit{SCT}^\alpha$ with $\mathsf{T}^\alpha_t \neq 0$: If o has no effect on variable V it does not induce a state-changing transition. Consider the case that o has an effect on V. If it has a precondition on V then $\langle \textit{pre}(o)[V], o, \textit{eff}(o)[V] \rangle$ is the only transition in $\textit{SCT}^\alpha$ labeled with o. Otherwise, the only transition labeled with o that is assigned a positive value is $\langle s[V], o, \textit{eff}(o)[V] \rangle$.

The goal inequality is trivially satisfied.

For the transition flow inequalities an argument similar to the one for the transition count inequalities leads to the following equations for abstract state v:

$$1) \quad \sum_{\substack{t \in \textit{SCT}^\alpha \\ t \text{ ends in } v}} \mathsf{T}^\alpha_t = \sum_{o \in \textit{AP}_{V=v} \cup \textit{SP}_{V=v}} \sum_{\substack{t \in \textit{SCT}^\alpha \\ t \text{ labeled with } o \\ t \text{ ends in } v}} \mathsf{T}^\alpha_t = \sum_{o \in \textit{AP}_{V=v}} \mathsf{Y}_o + \sum_{o \in \textit{SP}_{V=v}} \mathsf{Y}_o$$

$$2) \quad \sum_{\substack{t \in \textit{SCT}^\alpha \\ t \text{ starts in } v}} \mathsf{T}^\alpha_t = \sum_{o \in \textit{AC}_{V=v} \cup \textit{SC}_{V=v}} \sum_{\substack{t \in \textit{SCT}^\alpha \\ t \text{ labeled with } o \\ t \text{ starts in } v}} \mathsf{T}^\alpha_t = \sum_{o \in \textit{AC}_{V=v}} \mathsf{Y}_o + \sum_{o \in \textit{SC}_{V=v}} 0 = \sum_{o \in \textit{AC}_{V=v}} \mathsf{Y}_o .$$

Subtracting the second equation from the first shows that the left-hand side of the transition flow inequality for abstract state v matches the left-hand side of $c^{\text{ncl}}_{s,V=v}$.

It is easy to check that for $v \neq s[V]$ the right-hand side of $c^{\text{ncl}}_{s,V=v}$ is 1 if $s_G[V] = v$ and otherwise 0. This is exactly the right-hand side of the transition flow inequality under the given valuation for G^α_v. As $c^{\text{ncl}}_{s,V=v}$ is satisfied, the transition flow inequality must be satisfied as well. ∎

Theorem 1 *The state equation heuristic dominates the optimal cost partitioning heuristic for* Sys^1*: $h^{\text{OCP}}_{\text{Sys}^1} \leq h^{\text{SEQ}}$.*

Proof: By Lemma 1, every solution of the LP solved by h^{SEQ} can be extended to a solution for the LP for $h^{\text{OCP}}_{\text{Sys}^1}$ (using the new formulation based on operator-counting constraints). Moreover, both solutions have the same objective value. Therefore the estimate of the optimal cost partitioning heuristic cannot exceed the estimate of the state-equation heuristic. ∎

Role of Upper-bound Net Change Constraints

In the paper that introduced the state-equation heuristic, Bonet (2013) also suggested the previously-mentioned extension exploiting safe state variables and reported a (modest) improvement in performance when using it.

From Proposition 5 we know that the safety-based extension corresponds to adding upper-bound net change constraints to the LP. We now show that these constraints cannot improve the heuristic beyond the basic h^{SEQ} estimate.

Lemma 2 *For every state variable V it holds that*

$$\sum_{v \in D_V} \sum_{o \in \textit{AP}_{V=v}} \mathsf{Y}_o = \sum_{v \in D_V} \sum_{o \in \textit{AC}_{V=v}} \mathsf{Y}_o .$$

Proof sketch: In a directed graph the sum of in-degrees over all vertices is equal to the sum of out-degrees over all vertices. For the lemma, consider a directed graph whose vertices are the values of the variable V and there is an arc $v \xrightarrow{o} v'$ for each operator o that has precondition $V = v$ and effect $V = v'$. ∎

Lemma 3 *For state variable V and $v \in D_V$ it holds that*

$$\sum_{o \in \textit{SC}_{V=v}} \mathsf{Y}_o = \sum_{v' \in D_V \setminus \{v\}} \sum_{o \in \textit{SP}_{V=v'}} \mathsf{Y}_o .$$

Proof: From the definitions of *SP* and *SC* it is clear that $o \in \textit{SP}_{V=v'}$ with $v' \neq v$ implies that $o \in \textit{SC}_{V=v''}$ for all $v'' \neq v'$ and in particular $o \in \textit{SC}_{V=v}$. Conversely, $o \in \textit{SC}_{V=v}$ implies that there is a $v' \neq v$ with $o \in \textit{SP}_{V=v'}$. It also follows from the definition of *SP* that $\textit{SP}_{V=v'}$ and $\textit{SP}_{V=v''}$ are disjoint for $v' \neq v''$. Therefore the sum on the right side counts no Y_o more than once and the equation holds. ∎

Theorem 2 *Every feasible solution of the set of all lower-bound net change constraints for a task is also feasible for the set of all upper-bound net change constraints.*

Proof sketch: Consider an arbitrary value v for variable V.

If we sum up the left sides of the lower-bound net change constraints for all other values of V we get the sum

$$S = \sum_{v' \in D_V \setminus \{v\}} \sum_{o \in \textit{AP}_{V=v'}} \mathsf{Y}_o + \sum_{v' \in D_V \setminus \{v\}} \sum_{o \in \textit{SP}_{V=v'}} \mathsf{Y}_o - \sum_{v' \in D_V \setminus \{v\}} \sum_{o \in \textit{AC}_{V=v'}} \mathsf{Y}_o .$$

We can reformulate this as

$$S = \sum_{v' \in D_V} \sum_{o \in AP_{V=v'}} Y_o - \sum_{o \in AP_{V=v}} Y_o + \sum_{v' \in D_V \setminus \{v\}} \sum_{o \in SP_{V=v'}} Y_o - \sum_{v' \in D_V} \sum_{o \in AC_{V=v'}} Y_o + \sum_{o \in AC_{V=v}} Y_o .$$

Using Lemmas 2 and 3, S can be simplified to

$$S = \sum_{o \in AC_{V=v}} Y_o - \sum_{o \in AP_{V=v}} Y_o + \sum_{o \in SC_{V=v}} Y_o .$$

If we also sum up the right-hand side of the lower-bound net change constraints for all $v' \neq v$, we get the overall inequality $S \geq \sum_{v' \in D_V \setminus \{v\}} \min pnc^{s \to \star}_{V=v'}$.

The upper-bound net change constraint for $V = v$ is $-S \leq \max pnc^{s \to \star}_{V=v}$. Therefore we can prove that every feasible solution of the set of all lower-bound net change constraints for variable V is also feasible for $c^{\text{ncu}}_{V=v}$ by showing that $\sum_{v' \in D_V \setminus \{v\}} \min pnc^{s \to \star}_{V=v'} \geq -\max pnc^{s \to \star}_{V=v}$.

Considering all cases of the definition of *pnc*, it is not complicated to show that both sides of this inequality are in fact equal (omitted here for lack of space). ∎

This result challenges the safety-based improvement of the state-equation heuristic.

Corollary 1 *The safety-based improvement of the state-equation heuristic cannot improve the heuristic estimates.*

The experimental benefit reported for the safety-based improvement must hence be due to other factors, such as faster heuristic computation or noise. We will get back to this point in our experimental evaluation, which comes next.

Experimental Evaluation

We implemented a general framework for LP heuristics in the Fast Downward planning system (Helmert 2006), supporting net change, landmark, PDB and optimal cost partitioning constraints. The underlying LP solver is CPLEX v12.5.1. For our evaluation, we use all tasks for optimal planning from the IPC benchmark suite. All experiments were conducted on Intel Xeon E5-2660 processors (2.2 GHz) with a time limit of 30 minutes and a memory limit of 2 GB for each planner run.

For the different types of operator-counting constraints, we consider the following constraint groups:

SEQ All lower-bound net change constraints, corresponding to the state-equation heuristic h^{SEQ}.

PhO-Sys^1 All post-hoc optimization constraints for projections on goal variables.

PhO-Sys^2 All post-hoc optimization constraints for systematically generated projections on up to two variables. This corresponds to the most successful configuration of h^{PhO} reported by Pommerening, Röger, and Helmert (2013), but omits the variable merging optimization.

	SEQ	PhO-Sys^1	PhO-Sys^2	LMC	OPT-Sys^1	LMC+ PhO-Sys^2	LMC+ SEQ	PhO-Sys^2+ SEQ	LMC+ PhO-Sys^2+ SEQ	$h^{\text{LM-cut}}$
barman (20)	4	4	4	4	4	4	4	4	4	4
elevators (20)	7	9	16	16	4	17	16	15	16	**18**
floortile (20)	4	2	2	6	2	6	6	4	6	**7**
nomystery (20)	10	11	**16**	14	8	**16**	12	14	14	14
openstacks (20)	11	**14**	**14**	**14**	5	**14**	11	11	11	**14**
parcprinter (20)	**20**	11	13	13	7	14	**20**	**20**	**20**	13
parking (20)	3	**5**	1	2	1	1	2	1	1	3
pegsol (20)	**18**	17	17	17	10	17	**18**	17	16	17
scanalyzer (20)	11	9	4	11	7	10	10	10	8	**12**
sokoban (20)	16	19	**20**	**20**	13	**20**	**20**	**20**	19	**20**
tidybot (20)	7	13	**14**	**14**	4	**14**	10	8	10	**14**
transport (20)	**6**	**6**	**6**	**6**	4	**6**	**6**	5	**6**	**6**
visitall (20)	17	16	16	10	15	17	**19**	17	18	11
woodworking (20)	9	5	10	11	2	13	**16**	10	**16**	12
Sum IPC 2011 (280)	143	141	153	158	86	169	**170**	156	165	165
IPC 1998–2008 (1116)	487	446	478	586	357	589	**618**	516	598	598
Sum (1396)	630	587	631	744	443	758	**788**	672	763	763

Table 1: Coverage on the IPC benchmark suite. Best results are highlighted in bold.

LMC All landmark constraints for the landmarks found by the LM-cut heuristic.

OPT-Sys^1 All optimal cost partitioning constraints for projections on goal variables. This corresponds to the constraint set used in Theorem 1.

Individual constraint groups To get an idea of the quality of the different constraint groups, we ran experiments with A*, using one of the above configurations at a time. The resulting coverage is reported in the first block of Table 1.

Optimal cost partitioning on LM-cut landmarks leads to the highest coverage with a clear lead over the state-equation heuristic and the post-hoc optimization heuristic for Sys^2 patterns, which are almost on par. Using only Sys^1 patterns, the post-hoc optimization heuristic solves 44 fewer tasks and the optimal cost partitioning lags far behind with only 443 solved tasks, compared to 744 by the best LP configuration.

Comparing the coverage of the standard LM-cut heuristic and of the LMC configuration reveals that the additional effort of computing the *optimal* cost partitioning for the same landmarks does not pay off in terms of providing sufficiently better guidance. A possible reason for this is that the LM-cut heuristic already approximates h^+ very closely, and the corresponding LP heuristic is also bounded by h^+.

To measure the impact of the safety-based improvement of the state-equation heuristic, we conducted an additional experiment (not reported in the table) where we extended SEQ with the corresponding upper bound net change constraints. As expected from Corollary 1, this has no effect on the number of expanded nodes. However, with the additional constraints we solve six tasks less, which can be attributed to slightly slower evaluations of the LP solver.

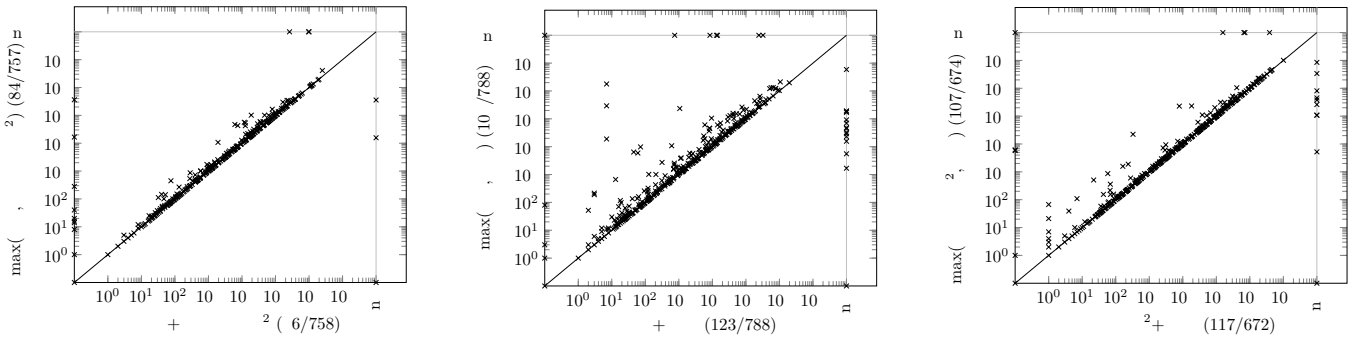

Figure 1: Number of expansions (excluding nodes on the final f-layer). The numbers (x/y) behind the configurations express that among the y solved tasks, x have been solved with perfect heuristic estimates.

We do not have a theoretical result that the upper bound net change constraints are strictly weaker than the lower bound net change constraints. However, if we solely use all upper bound net change constraints, the LP heuristic returns very poor estimates, resulting in a coverage of only 569 tasks. So these constraints are indeed strictly weaker.

Combinations of constraint groups In the following, we examine combinations of different types of operator-counting constraints. However, not all combinations make sense.

From Theorem 1 we know that the state-equation configuration dominates OPT-Sys^1 in terms of heuristic guidance. Since OPT-Sys^1 also requires much more time to compute the heuristic, we will not consider it in our combinations. Moreover, SEQ as well as PhO-Sys^2 give better coverage results and provably better guidance than PhO-Sys^1. Therefore, we will also omit PhO-Sys^1 from the combinations.

This leaves us with all combinations of SEQ, PhO-Sys^2, and LMC. The coverage results are included in Table 1.

A combination of SEQ and PhO-Sys^2 looks promising because they have their strengths and weaknesses in different domains. For example, using PhO-Sys^2 solves 14 tasks in the tidybot domain, while only 7 can be solved with SEQ. In the parcprinter domain the picture is reversed: using SEQ, we solve 20 tasks in contrast to only 13 with PhO-Sys^2. Indeed, the combination solves 672 instances, a clear improvement on each individual heuristic solving 630 and 631 tasks, respectively.

The combination of PhO-Sys^2 and LMC also pays off, solving 758 task instead of 631 and 744, respectively.

The best combination of two of our constraint groups consists of SEQ and LMC: with 788 task, it solves 44 more tasks than its best component, LMC, alone. This combination also outperforms the standard LM-cut heuristic (with 763 tasks), which was previously the best performer among the heuristics discussed in this paper.

However, adding more constraints does not always have a positive effect. While the combination of all three components is still better than the combination of SEQ and PhO-Sys^2 and of PhO-Sys^2 and LMC, it leads to 25 fewer solving instances than the combination of SEQ and LMC.

Constraint interactions Can we explain the better performance of the combinations with the better guidance of more individual components, or is there an additional positive effect through interactions of the different constraints in the LP? The plots in Figure 1 show the number of expansions using one LP heuristic with two constraint groups against the expansions using the maximum of the two individual LP heuristics.

In all three cases, we see clear synergy effects: combining two sets of constraints in a single LP indeed leads to stronger heuristic estimates than maximizing the heuristic estimates from two separate LPs. These synergy effects are much more pronounced in the combinations of SEQ and PhO-Sys^2 and of SEQ and LMC than in the combination of PhO-Sys^2 and LMC. In all three cases, there is a solid number of tasks (10–14) that are solved with perfect heuristic estimates by the combination into one LP, but not by the maximum of two LP heuristics.

Considering coverage, however, the picture is somewhat more mixed: some tasks can only be solved by the approaches using a single large LP, others only by the maximum over two LP heuristics, and both approaches end up too close to tell apart in terms of overall coverage.

Conclusion

We introduced a class of IP/LP heuristics based on operator-counting constraints that subsumes many existing heuristics, including the state-equation, post-hoc optimization and admissible landmark heuristic as well as optimal cost partitioning of (explicit-state) abstraction heuristics. Our new LP for optimal cost partitioning of abstraction heuristics is based on a dualization of the originally suggested LP and suggests new ways to combine abstraction heuristics with other sources of knowledge, such as landmarks.

Our two other main theoretical results are that the state-equation heuristic dominates optimal cost partitioning on single-variable abstractions (and therefore also the post-hoc optimization heuristic for these abstractions) and that the safety-based extension of the state-equation heuristic cannot improve heuristic accuracy.

Within our framework, heuristics can be arbitrarily combined. In our experiments, the best configuration combines

constraints from the state-equation heuristic and from optimal cost partitioning for LM-cut landmarks. This configuration solves 25 more tasks on the IPC benchmark set than the state-of-the-art LM-cut heuristic.

Another heuristic that fits into our framework and that we have not discussed is the IP/LP heuristic by van den Briel et al. (2007). While their base IP is very similar to the LP of the state-equation heuristic, they propose two extensions that further exploit the domain structure, which appear to be an interesting starting point for future work. Bonet and van den Briel (2014) further extend these ideas, defining new heuristics which would also be very interesting to explore in the context of our framework.

Acknowledgments

This work was supported by the Swiss National Science Foundation (SNSF) as part of the project "Abstraction Heuristics for Planning and Combinatorial Search" (AHPACS).

References

Bonet, B., and Helmert, M. 2010. Strengthening landmark heuristics via hitting sets. In Coelho, H.; Studer, R.; and Wooldridge, M., eds., *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010)*, 329–334. IOS Press.

Bonet, B., and van den Briel, M. 2014. Flow-based heuristics for optimal planning: Landmarks and merges. In *Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling (ICAPS 2014)*. AAAI Press.

Bonet, B. 2013. An admissible heuristic for SAS^+ planning obtained from the state equation. In Rossi, F., ed., *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, 2268–2274.

Culberson, J. C., and Schaeffer, J. 1998. Pattern databases. *Computational Intelligence* 14(3):318–334.

Domshlak, C.; Katz, M.; and Lefler, S. 2012. Landmark-enhanced abstraction heuristics. *Artificial Intelligence* 189:48–68.

Edelkamp, S. 2001. Planning with pattern databases. In Cesta, A., and Borrajo, D., eds., *Pre-proceedings of the Sixth European Conference on Planning (ECP 2001)*, 13–24.

Helmert, M., and Domshlak, C. 2009. Landmarks, critical paths and abstractions: What's the difference anyway? In Gerevini, A.; Howe, A.; Cesta, A.; and Refanidis, I., eds., *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS 2009)*, 162–169. AAAI Press.

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

Karpas, E., and Domshlak, C. 2009. Cost-optimal planning with landmarks. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*, 1728–1733.

Katz, M., and Domshlak, C. 2010. Optimal admissible composition of abstraction heuristics. *Artificial Intelligence* 174(12–13):767–798.

Keyder, E.; Richter, S.; and Helmert, M. 2010. Sound and complete landmarks for and/or graphs. In Coelho, H.; Studer, R.; and Wooldridge, M., eds., *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010)*, 335–340. IOS Press.

Pommerening, F.; Röger, G.; and Helmert, M. 2013. Getting the most out of pattern databases for classical planning. In Rossi, F., ed., *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, 2357–2364.

van den Briel, M.; Benton, J.; Kambhampati, S.; and Vossen, T. 2007. An LP-based heuristic for optimal planning. In Bessiere, C., ed., *Proceedings of the Thirteenth International Conference on Principles and Practice of Constraint Programming (CP 2007)*, volume 4741 of *Lecture Notes in Computer Science*, 651–665. Springer-Verlag.

Zhu, L., and Givan, R. 2003. Landmark extraction via planning graph propagation. In *ICAPS 2003 Doctoral Consortium*, 156–160.