Operator-Potential Heuristics for Symbolic Search

Daniel Fišer[1,2]**, Álvaro Torralba**[3]**, Jörg Hoffmann**[1]

[1] Saarland University, Saarland Informatics Campus, Saarbrücken, Germany
[2] Czech Technical University in Prague, Faculty of Electrical Engineering, Czech Republic
[3] Aalborg University, Denmark
danfis@danfis.cz, alto@cs.aau.dk, hoffmann@cs.uni-saarland.de

Abstract

Symbolic search, using Binary Decision Diagrams (BDDs) to represent sets of states, is a competitive approach to optimal planning. Yet heuristic search in this context remains challenging. The many advances on admissible planning heuristics are not directly applicable, as they evaluate one state at a time. Indeed, progress using heuristic functions in symbolic search has been limited and even very informed heuristics have been shown to be detrimental. Here we show how this connection can be made stronger for LP-based potential heuristics. Our key observation is that, for this family of heuristic functions, the change of heuristic value induced by each operator can be precomputed. This facilitates their smooth integration into symbolic search. Our experiments show that this can pay off significantly: we establish a new state of the art in optimal symbolic planning.

1 Introduction

A^* search with admissible heuristics and symbolic search are currently the two main contenders for the state of the art in cost-optimal planning. In principle, these are two orthogonal enhancements of a vanilla search algorithm—admissible heuristics aim to reduce the number of explored states, and symbolic search uses Binary Decision Diagrams (BDDs) (Bryant 1986) to efficiently represent and manipulate sets of states, greatly speeding up exhaustive search. A natural idea is to combine the two, and indeed that idea has been presented decades ago in the BDDA* algorithm (Edelkamp and Reffel 1998; Edelkamp 2002).

Yet that combination has not been an unqualified success. For a heuristic to be effective in symbolic search, two properties are required: (1) it must be possible to efficiently evaluate sets of states represented as BDDs, without evaluating the heuristic on each represented state individually; and (2) it must induce a good partitioning, so that sets of states with the same g- and h-value can be efficiently represented as BDDs. Property (1) is fulfilled by some of the strongest heuristics for explicit-state search (e.g. symbolic PDBs (Kissmann and Edelkamp 2011; Franco et al. 2017; Torralba, López, and Borrajo 2018)) so they can be used in BDDA*. However, it has been shown that even very informative heuristics can be detrimental in symbolic

search (Speck, Geißer, and Mattmüller 2020), when they do not fulfill property (2). The main reason is that reducing the amount of expanded states may be detrimental if the underlying BDD representation is less efficient. Due to all this, symbolic bidirectional blind search (without heuristics) is considered the dominant symbolic search approach, and the use of heuristic search in this context has lost traction.

Here we challenge this trend by showing that potential heuristics (Pommerening et al. 2015) yield fresh synergy between heuristic and symbolic search. Such heuristics assign a numeric value (a potential) to each fact of the planning task, in a way so that the sum of the potentials of the facts true in a state is an admissible estimate of the state's goal distance. As we show, potential heuristics are particularly well suited for combination with symbolic search.

Our key observation is that potentials can be computed for each operator rather than for each fact. Such *operator* potentials combine synergically with symbolic search as they have property (1): Under certain conditions, the operator potential of an operator o is equal to the difference in heuristic values $h(s') - h(s)$ for any state transition $s \rightarrow s'$ induced by the operator o. This allows for an efficient encoding of potential heuristics in symbolic search without having to compute the heuristic value of each explored state (Jensen, Veloso, and Bryant 2008). The main difficulty in doing so is that these operator potentials are real (floating-point) numbers, which can lead to rounding and precision issues. Naively rounding these values may lead to a path-dependent and inconsistent heuristic. We show that this can be dealt with by rounding operator potentials within the mixed-integer linear program (MIP) that derives the potential heuristics.

Our empirical analysis shows that potential heuristics also fulfill property (2). That is, they not only reduce the number of explored states, but also lead to improvements on the number of BDD nodes on average. This makes potential heuristics very helpful in symbolic search across a large number of benchmark domains. Overall, symbolic forward search with potential heuristics soundly outperforms symbolic bidirectional blind search, thus establishing a new state of the art in optimal symbolic planning.

2 Preliminaries

We consider the finite domain representation (FDR) of planning tasks (Bäckström and Nebel 1995). An **FDR planning**

task Π is specified by a tuple $\Pi = \langle \mathcal{V}, \mathcal{O}, I, G \rangle$. $\mathcal{V}$ is a finite set of **variables**, each variable $V \in \mathcal{V}$ has a finite **domain** $\mathrm{dom}(V)$. A **fact** $\langle V, v \rangle$ is a pair of a variable $V \in \mathcal{V}$ and one of its values $v \in \mathrm{dom}(V)$. The set of all facts is denoted by $\mathcal{F} = \{\langle V, v \rangle \mid V \in \mathcal{V}, v \in \mathrm{dom}(V)\}$, and the set of facts of variable V is denoted by $\mathcal{F}_V = \{\langle V, v \rangle \mid v \in \mathrm{dom}(V)\}$. A **partial state** p is a variable assignment over some variables $\mathrm{vars}(p) \subseteq \mathcal{V}$. We write $p[V]$ for the value assigned to the variable $V \in \mathrm{vars}(p)$ in the partial state p. We also identify p with the set of facts contained in p, i.e., $p = \{\langle V, p[V] \rangle \mid V \in \mathrm{vars}(p)\}$. A partial state s is a **state** if $\mathrm{vars}(s) = \mathcal{V}$. I is an **initial state**. G is a partial state called **goal**, and a state s is a **goal state** iff $G \subseteq s$. Let p, t be partial states. We say that t **extends** p if $p \subseteq t$.

$\mathcal{O}$ is a finite set of **operators**, each operator $o \in \mathcal{O}$ has a precondition $\mathrm{pre}(o)$ and effect $\mathrm{eff}(o)$, which are partial states over $\mathcal{V}$, and a cost $\mathrm{cost}(o) \in \mathbb{R}_0^+$. An operator o is **applicable** in a state s iff $\mathrm{pre}(o) \subseteq s$. The **resulting state** of applying an applicable operator o in a state s is another state $o[\![s]\!]$ such that $o[\![s]\!][V] = \mathrm{eff}(o)[V]$ for every $V \in \mathrm{vars}(\mathrm{eff}(o))$, and $o[\![s]\!][V] = s[V]$ for every $V \in \mathcal{V} \setminus \mathrm{vars}(\mathrm{eff}(o))$. We also assume that $\mathrm{pre}(o)[V] \neq \mathrm{eff}(o)[V]$ for every $V \in \mathrm{vars}(\mathrm{pre}(o)) \cap \mathrm{vars}(\mathrm{eff}(o))$.

Given a non-negative integer $n \in \mathbb{N}_0$, $[n]$ denotes the set $\{1, \ldots, n\}$ with $[0]$ defined as an empty set. A sequence of operators $\pi = \langle o_1, \ldots, o_n \rangle$ is applicable in a state s_0 if there are states $s_1, \ldots, s_n$ such that o_i is applicable in s_{i-1} and $s_i = o_i[\![s_{i-1}]\!]$ for $i \in [n]$. The resulting state of this application is $\pi[\![s_0]\!] = s_n$ and $\mathrm{cost}(\pi) = \sum_{i=1}^{n} \mathrm{cost}(o_i)$ denotes the cost of this sequence of operators. A sequence of operators π is called an s-**plan** iff π is applicable in a state s and $\pi[\![s]\!]$ is a goal state. An s-plan π is called **optimal** if its cost is minimal among all s-plans.

A state s is **reachable** if there exists an operator sequence π applicable in I such that $\pi[\![I]\!] = s$. Otherwise, we say that s is unreachable. The set of all reachable states is denoted by $\mathcal{R}$. An operator o is **reachable** iff it is applicable in some reachable state. A state s is a **dead-end state** iff $G \not\subseteq s$ and there is no s-plan. A set of facts $M \subseteq \mathcal{F}$ is a **mutex** if $M \not\subseteq s$ for every reachable state $s \in \mathcal{R}$.

A **heuristic** $h : \mathcal{R} \mapsto \mathbb{R} \cup \{\infty\}$ estimates the cost of optimal s-plans. The **optimal heuristic** $h^\star(s)$ maps each reachable state s to the cost of the optimal s-plan or to ∞ if s is a dead-end state. A heuristic h is called (a) **admissible** iff $h(s) \leq h^\star(s)$ for every reachable state $s \in \mathcal{R}$; (b) **goal-aware** iff $h(s) \leq 0$ for every reachable goal state s; and (c) **consistent** iff $h(s) \leq h(o[\![s]\!]) + \mathrm{cost}(o)$ for all reachable states $s \in \mathcal{R}$ and operators $o \in \mathcal{O}$ applicable in s. It is well-known that goal-aware and consistent heuristics are also admissible. In the context of heuristic search, h-value of a state node s refers to the heuristic value of s, g-value to the cost of the sequence of operators leading to s, and f-value is the sum of g-value and the maximum of h-value and zero (since we allow negative h-values).

3 Symbolic Search Background

Explicit state-space search operates on individual states, whereas symbolic search (McMillan 1993) works on sets of states represented by their characteristic functions. A characteristic function f_S of a set of states S is a Boolean function assigning 1 to states that belong to S and 0 to states that do not belong to S. Operations like the union ($\cup$), intersection ($\cap$), and complement of sets of states correspond to the disjunction ($\vee$), conjunction ($\wedge$), and negation ($\neg$) of their characteristic functions, respectively. Binary Decision Diagrams (BDDs) (Bryant 1986) are a efficient data-structure to represent Boolean functions in the form of a directed acyclic graph. The size of a BDD is the number of nodes in this representation. The main advantage of using BDDs is that often a BDD is much smaller than the number of states it represents. In fact, BDDs can be exponentially smaller, as certain sets containing exponentially many states can be represented by BDDs of polynomial size (Edelkamp and Kissmann 2008). Most operations on BDDs take only polynomial time in the size of the BDD, which enables the efficient manipulation of large sets of states.

The most prominent implementation of symbolic heuristic search in the context of automated planning is BDDA* (Edelkamp and Reffel 1998) which is a variant of A* (Hart, Nilsson, and Raphael 1968) using BDDs to represent sets of states. In BDDA*, operators of planning tasks are represented as *transition relations* (TRs), also using BDDs. A TR of an operator o is a characteristic function of pairs of states $(s, o[\![s]\!])$ for all states s such that o is applicable in s. Having a TR T_o for every operator $o \in \mathcal{O}$, we can construct a TR of a set of operators with the same cost c as $T_c = \bigvee_{o \in \mathcal{O}, \mathrm{cost}(o) = c} T_o$. As the size of T_c may be exponential in the number of operators with cost c, in practice, it is often a good idea to use *disjunctive partitioning* to keep the size at bay (Jensen, Veloso, and Bryant 2008; Torralba et al. 2017). Moreover, mutexes can be used for a more accurate approximation of reachable states (Torralba et al. 2017).

Like A*, BDDA* expands states by ascending order of their f-value. To take advantage of the symbolic representation, BDDA* represents all states with the same g and h value in a single BDD $S_{g,h}$ (disjunctive partitioning of $S_{g,h}$ can also be used). Given a set of states $S_{g,h}$ and a TR T_c, $\mathrm{image}(S_{g,h}, T_c)$ computes the set of successor states reachable from any state in $S_{g,h}$ by applying any operator represented by T_c.[1] The g-value of the resulting set of successor states is simply $g + c$. These successor states have to be split according to their h value. This can usually be performed efficiently (e.g., with symbolic PDBs (Kissmann and Edelkamp 2011)) by representing the heuristic as a BDD S_h per heuristic value that represents the set of states with that value and performing a conjunction.

GHSETA* and FSETA* (Jensen, Veloso, and Bryant 2008) encode the heuristic function as part of the transition relation, creating multiple TRs depending on the impact of the operators on heuristic value. This is a very efficient way of evaluating the heuristics within symbolic search. However, up to now, all heuristics known to be suitable for this representation were either non-informative, inadmissible, or domain dependent.

---

[1]The details how the function image works are not important here—Torralba et al. (2017) provide a detailed description.

4 Potential Heuristics Background

Potential heuristics (Pommerening et al. 2015) assign a numerical value to each fact, and the heuristic value for a state s is then simply a sum of the potentials of all facts in s.

Definition 1. Let Π denote a planning task with facts $\mathcal{F}$. A **potential function** is a function $\mathsf{P} : \mathcal{F} \mapsto \mathbb{R}$. A **potential heuristic** for P maps each state $s \in \mathcal{R}$ to the sum of potentials of facts in s, i.e., $h^{\mathsf{P}}(s) = \sum_{f \in s} \mathsf{P}(f)$.

We will leverage prior work on so-called disambiguation (Alcázar et al. 2013) to strengthen potential heuristics (Fišer, Horčík, and Komenda 2020). A disambiguation of a variable V for a given set of facts p is simply a set of facts $F \subseteq \mathcal{F}_V$ from variable V such that every reachable state extending p contains one of the facts from F.

Definition 2. Let Π denote a planning task with facts $\mathcal{F}$ and variables $\mathcal{V}$, let $V \in \mathcal{V}$ denote a variable, and let p denote a partial state. A set of facts $F \subseteq \mathcal{F}_V$ is called a **disambiguation of V for p** if for every reachable state $s \in \mathcal{R}$ such that $p \subseteq s$ it holds that $F \cap s \neq \emptyset$ (i.e., $\langle V, s[V] \rangle \in F$).

Clearly, every $\mathcal{F}_V$ is a disambiguation of V for all possible partial states, and if $\langle V, v \rangle \in p$ and there exists a reachable state extending p, then $\{\langle V, v \rangle\}$ is a disambiguation of V for p. Moreover, if the disambiguation of V for p is an empty set (for any V), then all states extending p are unreachable. Therefore, we can use empty disambiguations to determine unsolvability of planning tasks (if G extends p), or to prune unreachable operators (if a precondition of the operator extends p). So, from now on we will consider only nonempty disambiguations. Fišer, Horčík, and Komenda (2020) showed how to use mutexes to find disambiguations, so here we will assume we already have disambiguations inferred. Furthermore, to simplify the notation, we introduce a disambiguation map.

Definition 3. A mapping $\mathcal{D} : (\mathcal{O} \times \mathcal{V}) \cup \mathcal{V} \mapsto 2^{\mathcal{F}}$ is called a **disambiguation map** if (i) for every operator $o \in \mathcal{O}$ and every variable $V \in \text{vars}(\text{eff}(o))$ it holds that $\mathcal{D}(o, V) \subseteq \mathcal{F}_V$ is a disambiguation of V for $\text{pre}(o)$ such that $|\mathcal{D}(o, V)| \geq 1$; and (ii) for every variable $V \in \mathcal{V}$ it holds that $\mathcal{D}(V) \subseteq \mathcal{F}_V$ is a disambiguation of V for G such that $|\mathcal{D}(V)| \geq 1$.

Now we can state sufficient conditions for the potential heuristic to be admissible, which we will need later on.

Theorem 4. *(Fišer, Horčík, and Komenda 2020) Let $\Pi = \langle \mathcal{V}, \mathcal{O}, I, G \rangle$ denote a planning task with facts $\mathcal{F}$, and let P denote a potential function, and let $\mathcal{D}$ denote a disambiguation map. If*

$$\sum_{V \in \mathcal{V}} \max_{f \in \mathcal{D}(V)} \mathsf{P}(f) \leq 0 \qquad (1)$$

and for every operator $o \in \mathcal{O}$ it holds that

$$\sum_{V \in \text{vars}(\text{eff}(o))} \max_{f \in \mathcal{D}(o, V)} \mathsf{P}(f) - \sum_{f \in \text{eff}(o)} \mathsf{P}(f) \leq \text{cost}(o), \quad (2)$$

then the potential heuristic for P is admissible.

In practice, we can obtain potentials as a solution to a linear program (LP) with constraints corresponding to conditions from Theorem 4: For each $f \in \mathcal{F}$, we create a (real-valued) variable $\mathsf{P}(f)$, add constraints Eq. (1) and Eq. (2),

and then a solution of such LP for any objective function results in a goal-aware and consistent potential function. So far, potential heuristics have been used as described in Definition 1, i.e., each fact gets assigned a potential value and the heuristic value for a state s is the sum of potentials of all facts in s.

5 Operator-Potential Heuristics

Our key observation is that potentials can also be designed in a different way, yielding a new synergy with symbolic search: We can assign a potential to each operator and compute an admissible heuristic value for a state s reached by a sequence of operators π as a sum of operator potentials of all operators in π. We start by introducing an *operator-potential function*.

Definition 5. Given a potential function P, and a disambiguation map $\mathcal{D}$, a function $\mathsf{Q} : \mathcal{O} \mapsto \mathbb{R}$ is called an **operator-potential function** for P and $\mathcal{D}$ if

$$\mathsf{Q}(o) = \sum_{f \in \text{eff}(o)} \mathsf{P}(f) - \sum_{V \in \text{vars}(\text{eff}(o))} \max_{f \in \mathcal{D}(o, V)} \mathsf{P}(f) \quad (3)$$

for every operator $o \in \mathcal{O}$.

Note that the value of $\mathsf{Q}(o)$ is just the value of the left hand side of Eq. (2) with the opposite sign. Or in other words, the operator-potential function for an operator o gives us the lower bound on the change of the heuristic value of the corresponding potential heuristic for the given potential function P and disambiguation map $\mathcal{D}$. This immediately leads to another observation that for every sequence of operators $\pi = \langle o_1, \ldots, o_n \rangle$ applicable in the initial state I such that $s = \pi[\![I]\!]$ it holds that $h^{\mathsf{P}}(I) + \sum_{i \in [n]} \mathsf{Q}(o_i) \leq h^{\mathsf{P}}(s)$. So, we can compute an admissible heuristic estimate for any state s reachable by a sequence of operators π as $h^{\mathsf{P}}(I) + \sum_{i \in [n]} \mathsf{Q}(o_i)$. However, this estimate is path-dependent and therefore not necessarily consistent. Consider a planning task with a binary variable V, and an operator o with empty precondition and $\text{eff}(o) = \{\langle V, 1 \rangle\}$. It is easy to see that the value $\mathsf{Q}(o)$ is fixed for the operator as the smallest change in the heuristic value it can induce, but the actual change of the heuristic value may be different if the operator is applied on the state where $\langle V, 0 \rangle$ or $\langle V, 1 \rangle$ is set.

To avoid this difficulty, we now show that, if the disambiguation map $\mathcal{D}$ maps every operator o and every effect variable $V \in \text{vars}(\text{eff}(o))$ to a singleton, then $h^{\mathsf{P}}(I) + \sum_{i \in [n]} \mathsf{Q}(o_i) = h^{\mathsf{P}}(s)$ for every sequence of operators $\pi = \langle o_1, \ldots, o_n \rangle$ such that $\pi[\![I]\!] = s$. That is, as long as the preconditions on the variables affected by any operator o are known precisely, the potential h-value for any state s can be computed as the potential h-value for the initial state plus the sum of operator potentials of operators from any sequence of operators π leading to s. Lemma 6 shows that equality holds for any two consecutive states, and Lemma 7 shows it holds over any sequence of operators applicable in the initial state.

Lemma 6. *Let P denote a potential function, $\mathcal{D}$ a disambiguation map, Q an operator-potential function for P and $\mathcal{D}$, s a reachable state, and let o denote an operator applicable in s. If $|\mathcal{D}(o, V)| = 1$ for every $V \in \text{vars}(\text{eff}(o))$, then $\sum_{f \in s} \mathsf{P}(f) + \mathsf{Q}(o) = \sum_{f \in o[\![s]\!]} \mathsf{P}(f)$.*

Proof. Let $A = \bigcup_{V \in \text{vars}(\text{eff}(o))} \mathcal{D}(o, V)$. Since $|\mathcal{D}(o, V)| = 1$ for every $V \in \text{vars}(\text{eff}(o))$, Equation (3) can be rewritten as $\mathbb{Q}(o) = \sum_{f \in \text{eff}(o)} \mathbb{P}(f) - \sum_{f \in A} \mathbb{P}(f)$. And since s is reachable and o is applicable in s, it holds that $A \subseteq s$.

Let $B = s \setminus A$. Clearly, $o[\![s]\!] = B \cup \text{eff}(o)$ and $B \cap \text{eff}(o) = \emptyset$. Therefore, $\sum_{f \in s} \mathbb{P}(f) + \mathbb{Q}(o) = \sum_{f \in o[\![s]\!]} \mathbb{P}(f)$ can be rewritten to $\sum_{f \in B} \mathbb{P}(f) + \sum_{f \in A} \mathbb{P}(f) + \mathbb{Q}(o) = \sum_{f \in B} \mathbb{P}(f) + \sum_{f \in \text{eff}(o)} \mathbb{P}(f)$, and further simplified to $\sum_{f \in A} \mathbb{P}(f) + \mathbb{Q}(o) = \sum_{f \in \text{eff}(o)} \mathbb{P}(f)$. Expanding $\mathbb{Q}(o)$ gives us $\sum_{f \in A} \mathbb{P}(f) + \sum_{f \in \text{eff}(o)} \mathbb{P}(f) - \sum_{f \in A} \mathbb{P}(f) = \sum_{f \in \text{eff}(o)} \mathbb{P}(f)$, which concludes the proof. $\square$

Lemma 7. *Let* P, $\mathcal{D}$, *and* Q *be as before, and let* $\pi = \langle o_1, \dots, o_n \rangle$ *denote a sequence of operators applicable in* I, *and let* $s = \pi[\![I]\!]$. *If* $|\mathcal{D}(o, V)| = 1$ *for every* $o \in \mathcal{O}$ *and every* $V \in \text{vars}(\text{eff}(o))$, *then* $\sum_{f \in I} \mathbb{P}(f) + \sum_{i \in [n]} \mathbb{Q}(o_i) = \sum_{f' \in s} \mathbb{P}(f')$.

Proof. (By induction) It clearly holds for an empty sequence π. Let s' denote a state reachable from I by a sequence $\pi = \langle o_1, \dots, o_{n-1} \rangle$, and let $o_n \in \mathcal{O}$ denote an operator applicable in s', and let $s = o_n[\![s']\!]$. Now, assume that $\sum_{f \in I} \mathbb{P}(f) + \sum_{i \in [n-1]} \mathbb{Q}(o_i) = \sum_{f' \in s'} \mathbb{P}(f')$, and we need to prove that $\sum_{f \in I} \mathbb{P}(f) + \sum_{i \in [n]} \mathbb{Q}(o_i) = \sum_{f' \in s} \mathbb{P}(f')$. From the assumption, it follows that $\sum_{f \in I} \mathbb{P}(f) + \sum_{i \in [n-1]} \mathbb{Q}(o_i) + \mathbb{Q}(o_n) = \sum_{f' \in s'} \mathbb{P}(f') + \mathbb{Q}(o_n)$, so it is enough to show that $\sum_{f' \in s'} \mathbb{P}(f') + \mathbb{Q}(o_n) = \sum_{f \in s} \mathbb{P}(f)$ (Lemma 6). $\square$

Now, getting to the main result of this section, we formulate an *operator-potential heuristic* and we prove that this heuristic is well-defined and equal to the corresponding (fact) potential heuristic.

Definition 8. Let $\mathbb{Q}$ denote an operator-potential function for P and $\mathcal{D}$ such that $|\mathcal{D}(o, V)| = 1$ for every $o \in \mathcal{O}$ and every $V \in \text{vars}(\text{eff}(o))$. An **operator-potential heuristic** $h^{\mathbb{Q}} : \mathcal{R} \mapsto \mathbb{R} \cup \{\infty\}$ for $\mathbb{Q}$ is defined as

$$h^{\mathbb{Q}}(s) = \sum_{f \in I} \mathbb{P}(f) + \sum_{i \in [n]} \mathbb{Q}(o_i) \qquad (4)$$

for any sequence of operators $\pi = \langle o_1, \dots, o_n \rangle$ such that $\pi[\![I]\!] = s$.

Theorem 9. *Let* $\mathcal{D}$ *denote a disambiguation map such that* $|\mathcal{D}(o, V)| = 1$ *for every* $o \in \mathcal{O}$ *and every* $V \in \text{vars}(\text{eff}(o))$, *let* P *denote a potential function, and let* $\mathbb{Q}$ *denote an operator-potential function for* P *and* $\mathcal{D}$. *Then* $h^{\mathbb{Q}}$ *is well-defined, and* $h^{\mathbb{Q}}(s) = h^{\mathbb{P}}(s)$ *for every reachable state* s, *and* $h^{\mathbb{Q}}$ *is admissible (goal-aware, consistent) if* $h^{\mathbb{P}}$ *is admissible (goal-aware, consistent).*

Proof. It follows directly from Lemma 7. $\square$

Note that every planning task can be transformed into another task where $|\mathcal{D}(o, V)| = 1$ holds for every operator o and variable $V \in \text{vars}(\text{eff}(o))$. One possibility is the use of transition normal form (Pommerening and Helmert 2015)
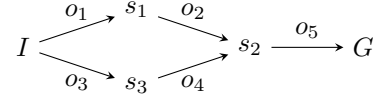
Figure 1: Let $\text{cost}(o_i) = 0$ for all $i \in [4]$, and $\text{cost}(o_5) = 1$, and let $\mathbb{Q}(o_1) = 1$, $\mathbb{Q}(o_2) = 0$, $\mathbb{Q}(o_3) = 0.9$, $\mathbb{Q}(o_4) = 0.1$, and $\mathbb{Q}(o_5) = -1$, and let $h^{\mathbb{Q}}(I) = 0$. A simple example showing inconsistency after rounding operator potentials down to the nearest integers.

which is polynomial, but introduces a set of auxiliary operators, and requires a transformation of the resulting plan back to the original planning task.

Another option is to enforce this property simply by enumerating all possible combinations of facts from all disambiguations $\mathcal{D}(o, V)$ such that $|\mathcal{D}(o, V)| > 1$ for all operators' preconditions. For example, given an operator o with $\text{vars}(\text{eff}(o)) = \{v_1, v_2\}$, and $\mathcal{D}(o, v_1) = \{f_1, f_2\}$ and $\mathcal{D}(o, v_2) = \{f_3, f_4\}$, we replace the operator o with four new operators $o_1, \dots, o_4$ with the same effect $\text{eff}(o_1) = \text{eff}(o_2) = \text{eff}(o_3) = \text{eff}(o_4) = \text{eff}(o)$, but different preconditions $\text{pre}(o_1) = \text{pre}(o) \cup \{f_1, f_3\}$, $\text{pre}(o_2) = \text{pre}(o) \cup \{f_1, f_4\}$, $\text{pre}(o_3) = \text{pre}(o) \cup \{f_2, f_3\}$, and $\text{pre}(o_4) = \text{pre}(o) \cup \{f_2, f_4\}$. This way, the transformed planning task can grow exponentially in the number of effects not appearing in the preconditions of an operator. However, we can prune operators whose preconditions form mutex, and in our experiments, we ran out of memory in only one task.

6 Handling Floating-Point Potentials

Although Theorem 9 identifies conditions under which operator-potential heuristics are consistent and equal to the corresponding potential heuristics, in practice there is an additional complication resulting from the fact that the $\mathbb{Q}(o)$ values are typically represented as floating-point numbers. So, we should not compare $\mathbb{Q}(o)$ values on equality. Moreover, having floating-point h-values is an even larger issue in symbolic search, as states are aggregated based on their h-value. If floating-point numbers are used, one could get different BDDs for every state in the search, greatly reducing the efficiency of symbolic search. So, it is desirable to represent in a single BDD all states whose h-values are rounded to the same integer value. Rounding operator potentials down to the nearest integers would resolve this problem and it would keep the heuristic function admissible. Unfortunately, this kind of rounding could make the heuristic inconsistent.

Consider a planning task depicted in Figure 1. Clearly, the operator-potential heuristic $h^{\mathbb{Q}}$ is both admissible and consistent. Now, let $\hat{\mathbb{Q}}$ denote an operator potential function obtained by rounding down $\mathbb{Q}$ to the nearest integers, i.e., $\hat{\mathbb{Q}}(o_1) = 1$, $\hat{\mathbb{Q}}(o_2) = 0$, $\hat{\mathbb{Q}}(o_3) = 0$, $\hat{\mathbb{Q}}(o_4) = 0$, and $\hat{\mathbb{Q}}(o_5) = -1$. The sum $\sum_{f \in I} \mathbb{P}(f) + \sum_{i \in [n]} \mathbb{Q}(o_i)$ (cf. Definition 8) provides an admissible estimate, because rounding down can only make the sum smaller. However, rounding down can also make this estimate path-dependent, i.e., we can obtain different values for a state depending on the path by which we reached the state, and inconsistent.

Consider the states s_1 and s_2, and operator sequences $\pi = \langle o_1 \rangle$ and $\pi' = \langle o_3, o_4 \rangle$ from Figure 1. Since the heuristic value for the initial state is zero, the inequality $h^{\mathtt{Q}}(s_1) - h^{\mathtt{Q}}(s_2) \leq \text{cost}(o_2)$ holds, because $h^{\mathtt{Q}}(s_1) = 1$, $h^{\mathtt{Q}}(s_2) = 1$, and $\text{cost}(o_2) = 0$. But after rounding, we get $\hat{\mathtt{Q}}(o_1) = 1$ and $\hat{\mathtt{Q}}(o_3) + \hat{\mathtt{Q}}(o_4) = 0$ resulting in a higher estimate for s_1 than for s_2 using $\hat{\mathtt{Q}}$. We resolve this issue by encoding the rounding directly into the mixed-integer linear program (MIP) expressing the potentials. Since the operator potential is just a left hand side of Eq. (2) with the opposite sign, we can extend the original LP for the inference of potentials with new integer variables, each corresponding to operator-potential $\mathtt{Q}(o)$, and add a new constraint Eq. (3) for each $\mathtt{Q}(o)$. That is, the new MIP has a real-valued variable $\mathtt{P}(f)$ for each $f \in \mathcal{F}$, an integer variable $\mathtt{Q}(o)$ for each operator $o \in \mathcal{O}$, and a set of constraints Eq. (1), Eq. (2), and Eq. (3). Similarly to before, a solution for such MIP for any objective function yields a goal-aware and consistent operator-potential function. And since the variables corresponding to $\mathtt{Q}(o)$ are integer-valued, the resulting operator-potentials are also integer-valued while the fact potentials can remain real-valued. Therefore, the MIP solver finds operator-potentials that are properly rounded, so that we can compare the operator-potentials on equality without running into problems with floating-point numbers.

7 Symbolic Search with Operator-Potentials

Using potential heuristics in BDDA* is not straightforward, as the standard way of evaluating the heuristics by constructing a BDD S_h representing all states with a heuristic value equal to h may not always be feasible. The naive approach would be to enumerate all possible sub-sets of features whose potentials add up-to h. However, this requires enumerating exponentially many sub-sets and it may result in an exponentially large intermediate BDD. We overcome this difficulty by using operator-potential heuristics instead.

To do so, we use GHSETA* (Jensen, Veloso, and Bryant 2008), a symbolic heuristic search algorithm which partitions the TRs not only by the cost of the corresponding operators, but also by the change of the heuristic value they induce.[2] That is, instead of creating a TR T_c for all operators o having $\text{cost}(o) = c$, we create a TR $T_{c,q}$ representing all operators o such that $\text{cost}(o) = c$ and $\mathtt{Q}(o) = q$. For the initial state, the g-value is set to zero, and the h-value is set to $\sum_{f \in I} \mathtt{P}(f)$. For all subsequent states $S_{g,h}$ expanded by the TR $T_{c,q}$, the g-value and h-value of the resulting state $S'_{g',h'} = \text{image}(S_{g,h}, T_{c,q})$ is set to $g' = g + c$ and $h' = h + q$, respectively.

The pseudocode for the algorithm using a *consistent* operator-potential heuristic is encapsulated in Algorithm 1. On lines 1 and 2, the TRs corresponding to all unique pairs of operator costs and $\mathtt{Q}(o)$ values are constructed. The heuristic value for the initial state is computed on line 3. The open list of sets of states (represented by BDDs) ordered by $f = g + \max(0, h)$ values is initialized with the initial state

---

[2]The alternative FSETA* does not directly support negative heuristic values (Jensen, Veloso, and Bryant 2008).

Algorithm 1: Symbolic forward A* with a consistent operator-potential heuristic.

Input: A planning task Π, a consistent operator potential function $\mathtt{Q}$.
Output: An optimal plan or "unsolvable".

1 **for each** $c, q \in \{\text{cost}(o), \mathtt{Q}(o) \mid o \in \mathcal{O}\}$ **do**
2   Construct $T_{c,q}$ from $\{o \in \mathcal{O} \mid \text{cost}(o) = c, \mathtt{Q}(o) = q\}$;
3 $h_I \leftarrow \sum_{f \in I} \mathtt{P}(f)$;
4 $S_{0,h_I} \leftarrow \{I\}$;
5 $\text{open} \leftarrow \{\langle \max(0, h_I), S_{0,h_I} \rangle\}$;
6 $\text{closed} \leftarrow \emptyset$;
7 **while** $\text{open} \neq \emptyset$ **do**
8     $S_{g,h} \leftarrow \texttt{PopMin}(\text{open}) \setminus \text{closed}$;
9   **if** $S_{g,h}$ *contains a goal state* **then**
10   **return** $\texttt{ExtractPlan}(S_{g,h})$;
11     $\text{closed} \leftarrow \text{closed} \cup S_{g,h}$;
12   **for each** $T_{c,q}$ **do**
13        $S_{g+c,h+q} \leftarrow \text{image}(S_{g,h}, T_{c,q}) \setminus \text{closed}$;
14   **if** $S_{g+c,h+q} \neq \emptyset$ **then**
15           $f \leftarrow g + c + \max(0, h + q)$;
16           $\texttt{InsertOrUpdate}(\text{open}, \langle f, S_{g+c,h+q} \rangle)$;
17 **return** "unsolvable";

---

on lines 4 and 5. On line 6, a BDD representing all closed states is constructed. The while-cycle on lines 7–16 is an A* algorithm adapted to the symbolic search. On line 8, we extract the set of states with the lowest f-value from the open list and remove all closed states from this set. If a goal state is reached (line 9 and 10), an optimal plan is extracted and returned (for details see (Torralba et al. 2017)). If the current set of states $S_{g,h}$ does not contain a goal state, then all these states are added to the set of closed states (line 11). On lines 12–16, all operators are applied and the resulting states that were not closed yet are assigned the correct g and h values and either inserted into the open list (if there is no $S_{g+c,h+q}$ in the open list), or the set of states in the open list is extended with the new set of states.
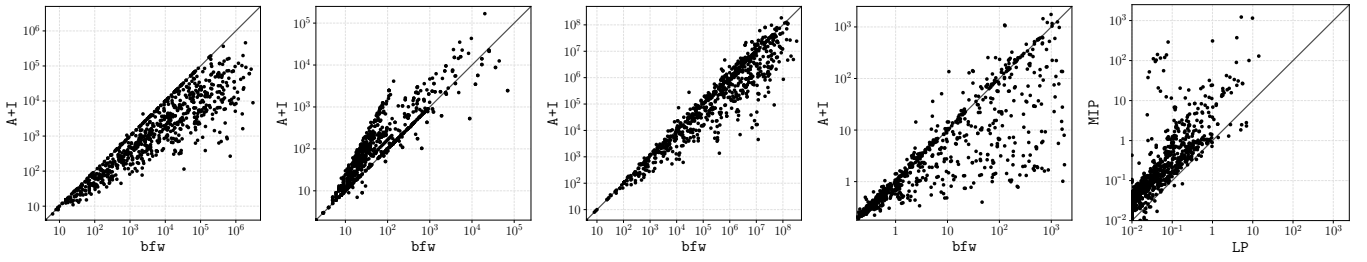
Note that we need the operator-potential heuristic to be consistent in order to avoid re-opening states (cf. lines 8 and 13). This also means that if the computation of consistent operator-potentials require the transformation of the planning task described in Section 5, then the same transformed task must be used also for the symbolic search.

GHSETA* can also be adapted to work with the path-dependent (and thus possibly inconsistent) variant of the operator-potential heuristic that does not require transforming the planning task. We need to allow re-opening states that were previously closed with a higher g-value. We can achieve that by maintaining not one BDD representing closed states, but one BDD for each g-value (cf. lines 6 and 11). And when removing closed states from the set of states, we remove only the closed states with the same or lower g-value (cf. lines 8 and 13).

8 Experimental Evaluation

We implemented our search algorithm in C.[3] Operators and facts are pruned with the h^2 heuristic in forward and back-

---

[3]https://gitlab.com/danfis/cpddl, branch aaai22-symba-op-pot

(a) Mean number of BDD nodes per expanded BDD. (b) Number of expanded BDDs. (c) Sum of expanded BDD nodes. (d) Runtime in seconds. (e) Inference time of potentials.

Figure 2: (a)–(d): Comparison of symbolic forward uniform-cost search (bfw) against GHSETA* with the best-performing variant of the consistent operator-potential heuristic (A+I) on commonly solved tasks. (e): Time (s) spent computing potentials for the original formulation (LP) and with added constraints on integer operator potentials (MIP) on transformed tasks.

ward direction (Alcázar and Torralba 2015), and the translation from PDDL to FDR uses the inference of mutex groups proposed by Fišer (2020). We used all planning domains from the optimal track of International Planning Competitions (IPCs) from 1998 to 2018 excluding the ones containing conditional effects after translation. We merged, for each domain, all benchmark suites across different IPCs. This leaves 48 domains overall.

We used a cluster of computing nodes with Intel Xeon Scalable Gold 6146 processors and CPLEX (I)LP solver v12.10. The time and memory limits were set to 30 minutes and 8 GB, respectively. We used a time limit of 30 seconds for applying mutexes on the goal BDD and 10 seconds for merging transition relation BDDs (Torralba et al. 2017).

We evaluated GHSETA* with the following variants of the *consistent* operator-potential heuristics obtained on the transformed planning tasks:

- I: maximize the $h^\mathbb{Q}$-value of the initial state (Pommerening et al. 2015).

- A+I: maximize the $h^\mathbb{Q}$-value for the average (syntactic) state while enforcing the maximum $h^\mathbb{Q}(I)$ (Seipp, Pommerening, and Helmert 2015; Fišer, Horčík, and Komenda 2020).

- S$_{1k}$+I: maximize the $h^\mathbb{Q}$-value for 1 000 states sampled using random walks, while enforcing the maximum $h^\mathbb{Q}(I)$ (Seipp, Pommerening, and Helmert 2015; Fišer, Horčík, and Komenda 2020).

- M$_2$+I: maximize the $h^\mathbb{Q}$-value for all reachable states approximated with mutex pairs while enforcing the maximum $h^\mathbb{Q}(I)$ (Fišer, Horčík, and Komenda 2020).

We also evaluated the same consistent operator-potential heuristics with the tasks transformed to the transition normal form (Pommerening and Helmert 2015) (prefixed with tnf-); and the *path-dependent* operator-potential heuristics on the original planning task (prefixed with pd-).

We compare these to symbolic uniform-cost search using forward (bfw) and bidirectional search (bbi)[4]. Furthermore, we compare to other state-of-the-art planners. We ran

---

[4]Our implementation does not lack in performance behind the IPC 2018 SymbA* planner—its overall coverage is 942 and 852 for blind bidirectional and blind forward search, respectively.

	scrp	A+I	M$_2$+I	comp2	S$_{1k}$+I	pd-A+I	bbi	pot$_{A+I}$	I	bfw	lmc	ms	tnf-A+I	tot
scrp	–	**23**	**23**	19	**26**	24	27	26	30	31	36	37	39	1 112
A+I	13	–	7	**17**	**17**	9	24	26	26	30	33	37	36	1 109
M$_2$+I	13	2	–	17	**16**	10	23	25	23	31	33	36	35	1 103
comp2	16	16	**18**	–	23	18	22	27	25	32	33	38	37	1 096
S$_{1k}$+I	13	3	6	16	–	9	22	22	22	29	29	35	34	1 081
pd-A+I	12	2	5	16	**15**	–	22	22	23	27	30	32	34	1 080
bbi	14	14	14	12	16	15	–	23	20	23	24	29	31	1 004
pot$_{A+I}$	5	9	9	13	14	11	18	–	19	23	24	22	29	1 000
I	10	2	3	9	5	5	16	15	–	20	21	24	29	996
bfw	10	7	6	7	9	7	7	17	10	–	17	22	28	936
lmc	2	4	5	6	8	8	13	14	15	20	–	21	28	901
ms	1	1	2	4	5	4	11	4	10	16	15	–	24	859
tnf-A+I	4	0	1	6	3	2	7	7	6	12	11	12	–	803

Table 1: Summary of domain coverage. A value in row x and column y is the number of domains where x solved more tasks than y, it is bold if higher than the value in row y and column x. "tot" shows overall number of solved tasks. Highlighted rows correspond to GHSETA* with operator-potential heuristics.

A* with the LM-Cut (lmc) heuristic (Helmert and Domshlak 2009), with the merge-and-shrink (ms) heuristic with SCC-DFP merge strategy and non-greedy bisimulation shrink strategy (Helmert et al. 2014; Sievers, Wehrle, and Helmert 2016), and with the potential heuristic (pot$_{A+I}$), i.e., a variant of A+I for A*. We further compare to two of the best-performing non-portfolio planners from IPC 2018: Complementary2 (comp2) (Franco et al. 2017; Franco, Lelis, and Barley 2018), and Scorpion (scrp) (Seipp 2018; Seipp and Helmert 2018). Table 1 shows the comparison across all planners in terms of total coverage, and in terms of the number of domains in which each planner is superior to others.

Operator Potentials in Symbolic Search

The comparison of symbolic potential variants against the baseline forward search without heuristics (bfw) clearly demonstrates that potential heuristics are beneficial for the performance of symbolic search over a wide range of dif-

ferent domains. The best variant of GHSETA* with an operator-potential heuristic (A+I) solves 173 more tasks than the baseline (bfw). It increases coverage on 30 different domains, and it is detrimental in only 7 domains. Among the different variants of potential heuristics, we observe that the optimization criteria can have a significant impact on performance. The best variant is A+I, closely followed by S_{1k}+I and M_2+I, and significantly better than I. These results are well in line with the results of the same potential heuristics in explicit search (Fišer, Horčík, and Komenda 2020).

For a heuristic to be beneficial in symbolic search, it is required that sets of states with the same g and h value are efficiently represented with BDDs. The positive coverage results from Table 1 suggest that this is indeed the case for the operator-potential heuristics. To confirm this, Figure 2 compares the performance of the baseline, symbolic search without any heuristics (bfw), against the best configuration of our symbolic search with potential heuristics (A+I).

First, we observe that the average number of BDD nodes per expanded BDD was almost always lower (Figure 2a). This means that, indeed, the partitioning induced by operator-potential heuristics is often beneficial, resulting on sets of states during the search that have a concise BDD representation. This property, however, is not guaranteed by the method. In particular, the average number of BDD nodes per expanded BDD was higher for bfw in only ten tasks, though only by a small margin. On the other hand, the number of expanded BDDs is almost always increased (Figure 2b), which is not surprising, as sets of states during the search are not only partitioned by g-value, but also by h-value.

Most remarkably, the number of BDD nodes from all expanded BDDs often decreased with potential heuristics (Figure 2c). This confirms that these heuristics are not only informative for explicit-state search, avoiding expansion of certain states, but also beneficial in symbolic search by inducing a good BDD partitioning. This contrasts with previous results on very informative heuristics ($\frac{1}{2}h^*$, $\frac{3}{4}h^*$), that despite their accuracy are often detrimental for the performance of symbolic search (Speck, Geißer, and Mattmüller 2020).

Furthermore, the runtime of the planner is often decreased (Figure 2d), confirming that partitioning the TRs using operator potentials is an effective way of evaluating the heuristic in symbolic search. So, not only potential heuristics can be informative for symbolic search, but they can also be efficiently evaluated. The increase in runtime for some of the tasks is often due to increased computational effort in the inference of integer operator potentials.

Compared against symbolic bidirectional uniform cost search (bbi), which has state-of-the-art performance in symbolic search planning (Torralba et al. 2014; Torralba, Linares López, and Borrajo 2016), A+I solves 105 more tasks. The two algorithms are still quite complementary though, with A+I being superior in 24 domains and bbi in 14. This suggests that there is a potential to further improve bidirectional heuristic search by integrating operator-potential heuristics.

Explicit-State Search with Potential Heuristics

Compared to the potential heuristics in explicit-state search, A+I solves 109 instances more than pot_{A+I}. Moreover, there

are only 9 domains where using symbolic search with the same heuristics is detrimental, compared to 26 domains where it is beneficial.

Note that these two configurations are using the same optimization criteria to compute the potentials. However, as explained in Section 6, in order to obtain consistent operator-potential heuristics, we must (1) split the operators so that all variables mentioned in the effect appear in the preconditions; and (2) use MIP to ensure that operator potentials have an integer value. In most domains this is not an issue, and there are only a few domains where the number of operators increases significantly (e.g., only in agricola and maintenance the task size at least doubled). Nevertheless, this sometimes has an overhead on the computation of the potential heuristics, as illustrated by Figure 2e, which makes the improvements in coverage even more remarkable.

To analyze if this overhead could be easily avoided, we also compared A+I against pd-A+I and tnf-A+I. The path-dependent variant pd-A+I partially avoids the overhead shown in Figure 2e, because it does not require the transformation of tasks as per (1). Yet, in terms of coverage, results are inferior in general, reducing coverage in 9 domains and improving only in agricola, and caldera. We also tried a variant of pd-A+I with floating-point potentials (i.e., only LP is solved) and rounding to nearest smaller integer (to deal with partitioning of states), but it solved 107 fewer tasks from 20 domains. This shows that it pays off to force the heuristic to be consistent, unless the task size increases significantly.

Finally, using the transition normal form (tnf-A+I) turns out to be detrimental for symbolic search and never pays-off, because the symbolic search must also use the task in transition normal form with a lot of auxiliary zero-cost operators.

Comparison Against State of the Art

Compare finally the performance of our best-performing variants (A+I, M_2+I, and S_{1k}+I) against the unrelated approaches lmc, ms, comp2, and scrp. Our planners clearly beat lmc and ms in terms of overall coverage and frequency of per-domain superiority. The state-of-the-art planners comp2 and scrp are roughly on par in overall coverage. In terms of individual domains, the clear conclusion is that our new techniques are highly complementary to the previous state of the art: A+I, M_2+I, and S_{1k}+I outperform comp2 and scrp in 12 to 16 domains.

9   Conclusion

While heuristic search and symbolic search are both contenders for the throne in optimal planning, and their combination is a natural and promising avenue, the results with that combination have thus far been disappointing. As we show, this picture changes dramatically when leveraging the fact that potential heuristics can be viewed as potentials over operators, which enables their smooth integration into symbolic search. We have shown that and how this can be done, in particular while retaining consistency. Our empirical results show that this boosts the performance of optimal symbolic planning, which is now on par with the best heuristic search based optimal planners.

Acknowledgements

References

Alcázar, V.; Borrajo, D.; Fernández, S.; and Fuentetaja, R. 2013. Revisiting Regression in Planning. In Rossi, F., ed., *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI'13)*, 2254–2260. AAAI Press/IJCAI.

Alcázar, V.; and Torralba, Á. 2015. A Reminder about the Importance of Computing and Exploiting Invariants in Planning. In Brafman, R.; Domshlak, C.; Haslum, P.; and Zilberstein, S., eds., *Proceedings of the 25th International Conference on Automated Planning and Scheduling (ICAPS'15)*, 2–6. AAAI Press.

Bäckström, C.; and Nebel, B. 1995. Complexity Results for SAS$^+$ Planning. *Computational Intelligence*, 11(4): 625–655.

Bryant, R. E. 1986. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, 35(8): 677–691.

Edelkamp, S. 2002. Symbolic Pattern Databases in Heuristic Search Planning. In Ghallab, M.; Hertzberg, J.; and Traverso, P., eds., *Proceedings of the 6th International Conference on Artificial Intelligence Planning and Scheduling (AIPS'02)*, 274–283. AAAI Press, Menlo Park.

Edelkamp, S.; and Kissmann, P. 2008. Limits and Possibilities of BDDs in State Space Search. In Fox, D.; and Gomes, C., eds., *Proceedings of the 23rd National Conference of the American Association for Artificial Intelligence (AAAI'08)*, 1452–1453. AAAI Press.

Edelkamp, S.; and Reffel, F. 1998. OBDDs in Heuristic Search. In Herzog, O.; and Günter, A., eds., *Proceedings of the 22nd Annual German Conference on Advances in Artificial Intelligence (KI'98)*, volume 1504 of *Lecture Notes in Computer Science*, 81–92. Springer.

Fišer, D. 2020. Lifted Fact-Alternating Mutex Groups and Pruned Grounding of Classical Planning Problems. In Conitzer, V.; and Sha, F., eds., *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI'20)*. AAAI Press.

Fišer, D.; Horčík, R.; and Komenda, A. 2020. Strengthening Potential Heuristics with Mutexes and Disambiguations. In *Proceedings of the 30th International Conference on Automated Planning and Scheduling (ICAPS'20)*, 124–133. AAAI Press.

Franco, S.; Lelis, L. H. S.; and Barley, M. 2018. The Complementary2 Planner in IPC 2018. In *IPC 2018 planner abstracts*, 32–36.

Franco, S.; Torralba, A.; Lelis, L. H.; and Barley, M. 2017. On Creating Complementary Pattern Databases. In Sierra, C., ed., *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, 4302–4309. AAAI Press/IJCAI.

Hart, P. E.; Nilsson, N. J.; and Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2): 100–107.

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 19th International Conference on Automated Planning and Scheduling (ICAPS'09)*, 162–169. AAAI Press.

Helmert, M.; Haslum, P.; Hoffmann, J.; and Nissim, R. 2014. Merge & Shrink Abstraction: A Method for Generating Lower Bounds in Factored State Spaces. *Journal of the Association for Computing Machinery*, 61(3): 16:1–16:63.

Jensen, R. M.; Veloso, M. M.; and Bryant, R. E. 2008. State-set branching: Leveraging BDDs for heuristic search. *Artificial Intelligence*, 172(2-3): 103–139.

Kissmann, P.; and Edelkamp, S. 2011. Improving Cost-Optimal Domain-Independent Symbolic Planning. In Burgard, W.; and Roth, D., eds., *Proceedings of the 25th National Conference of the American Association for Artificial Intelligence (AAAI'11)*, 992–997. AAAI Press.

McMillan, K. L. 1993. *Symbolic Model Checking*. Kluwer Academic Publishers.

Pommerening, F.; and Helmert, M. 2015. A Normal Form for Classical Planning Tasks. In Brafman, R.; Domshlak, C.; Haslum, P.; and Zilberstein, S., eds., *Proceedings of the 25th International Conference on Automated Planning and Scheduling (ICAPS'15)*, 188–192. AAAI Press.

Pommerening, F.; Helmert, M.; Röger, G.; and Seipp, J. 2015. From Non-Negative to General Operator Cost Partitioning. In Bonet, B.; and Koenig, S., eds., *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*, 3335–3341. AAAI Press.

Seipp, J. 2018. Fast Downward Scorpion. In *IPC 2018 planner abstracts*, 77–79.

Seipp, J.; and Helmert, M. 2018. Counterexample-Guided Cartesian Abstraction Refinement for Classical Planning. *Journal of Artificial Intelligence Research*, 62: 535–577.

Seipp, J.; Pommerening, F.; and Helmert, M. 2015. New Optimization Functions for Potential Heuristics. In Brafman, R.; Domshlak, C.; Haslum, P.; and Zilberstein, S., eds., *Proceedings of the 25th International Conference on Automated Planning and Scheduling (ICAPS'15)*, 193–201. AAAI Press.

Sievers, S.; Wehrle, M.; and Helmert, M. 2016. An Analysis of Merge Strategies for Merge-and-Shrink Heuristics. In Coles, A.; Coles, A.; Edelkamp, S.; Magazzeni, D.; and Sanner, S., eds., *Proceedings of the 26th International Conference on Automated Planning and Scheduling (ICAPS'16)*, 294–298. AAAI Press.

Speck, D.; Geißer, F.; and Mattmüller, R. 2020. When Perfect Is Not Good Enough: On the Search Behaviour of Symbolic Heuristic Search. In *Proceedings of the 30th International Conference on Automated Planning and Scheduling (ICAPS'20)*, 263–271. AAAI Press.

Torralba, Á.; Alcázar, V.; Borrajo, D.; Kissmann, P.; and Edelkamp, S. 2014. SymBA*: A Symbolic Bidirectional A* Planner. In *IPC 2014 planner abstracts*, 105–109.

Torralba, Á.; Alcázar, V.; Kissmann, P.; and Edelkamp, S. 2017. Efficient symbolic search for cost-optimal planning. *Artificial Intelligence*, 242: 52–79.

Torralba, Á.; Linares López, C.; and Borrajo, D. 2016. Abstraction Heuristics for Symbolic Bidirectional Search. In Kambhampati, S., ed., *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*, 3272–3278. AAAI Press/IJCAI.

Torralba, Á.; López, C. L.; and Borrajo, D. 2018. Symbolic perimeter abstraction heuristics for cost-optimal planning. *Artificial Intelligence*, 259: 1–31.