ECAI 2014
T. Schaub et al. (Eds.)
© 2014 The Authors and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License.
doi:10.3233/978-1-61499-419-0-633

Landmarks in Oversubscription Planning

Vitaly Mirkis and **Carmel Domshlak** [1]

Abstract. In the basic setup of oversubscription planning (OSP), the objective is to achieve an as valuable as possible subset of goals within a fixed allowance of the total action cost [32]. Continuing from the recent successes in exploiting logical goal-reachability landmarks in classical planning, we develop a framework for exploiting such landmarks in heuristic-search OSP. We show how standard landmarks of certain classical planning tasks can be compiled into the OSP task of interest, resulting in an equivalent OSP task with a lower budget, and thus with a smaller search space. We then show how such landmark-based task enrichment can be combined in a mutually stratifying way with the *BFBB* search used for OSP planning. Our empirical evaluation confirms the effectiveness of the proposed landmark-based budget reduction scheme.

1 INTRODUCTION

In most general terms, deterministic planning is a problem of finding paths in large-scale yet concisely represented state-transition systems. In what these days is called classical planning [11], the task is to find an as *cost-effective* path as possible to a *goal-satisfying* state. In contrast, in what Smith [32] baptized as "oversubscription" planning (OSP), the task is to find an as *goal-effective* (or *valuable*) state as possible via a *cost-satisfying* path. In other words, the hard constraint of classical planning translates to only preference in OSP, and the hard constraint of OSP translates to only preference in classical planning. Finally, in "optimal" classical planning and OSP, the tasks are further constrained to finding only *most* cost-effective paths and *most* goal-effective states, respectively.

Classical planning and OSP constitute the most fundamental variants of deterministic planning, with many other variants of deterministic planning being defined in terms of mixing and relaxing the two. For instance, "net-benefit" planning tries to achieve both (classical) cost-effectiveness of the path and (OSP) goal-effectiveness of the end-state by additively combining the two measures, but at the same time, it relaxes the hard constraints of (classical) goal-satisfaction and (OSP) cost-satisfaction [31, 1, 4, 2, 7, 24]. Another popular setup is "cost-bounded" planning, in which both (classical) goal-satisfaction and (OSP) cost-satisfaction are pursued, but both (classical) cost-effectiveness of the path and (OSP) goal-effectiveness of the end-state are relaxed/ignored [33, 34, 13, 15, 19, 12, 27].

While OSP has been advocated over the years on par with classical planning, so far, the theory and practice of the latter have been studied and advanced much more intensively. The remarkable success and continuing progress of heuristic-search solvers for classical planning is one notable example. Primary enablers of this success are the advances in domain-independent approximations, or heuristics, of the cost needed to achieve a goal state from a given state. With our focus here on optimal planning, two classes of approximation techniques have been found especially useful in the context of optimal classical planning: those based on state-space *abstractions* [10, 14, 18, 22] and these based on logical *landmarks* for goal reachability [21, 17, 9, 5, 28].

Considering OSP as heuristic search, a question is then whether some similar-in-spirit (yet possibly mathematically different) approximation techniques can be developed for heuristic-search OSP. Recently, the authors provided the first affirmative answer to this question in the context of abstractions by developing the actual notion of OSP abstractions, investigating the complexity of working with them for the purpose of heuristic approximation, and demonstrating empirically that using OSP abstraction heuristics within a best-first branch-and-bound (*BFBB*) search can be extremely effective in practice [25]. In contrast, the prospects of goal-reachability landmarks in heuristic-search OSP have not been investigated yet.

This is precisely the contribution of this paper: First, we introduce and study ε-landmarks, the logical properties of OSP plans that achieve valuable states. We show that ε-landmarks correspond to regular landmarks of certain classical planning tasks that can be (straightforwardly) derived from the OSP tasks of interest. We then show how such ε-landmarks can be compiled into the OSP task of interest, resulting in an equivalent OSP task, but with a stricter cost satisfaction constraint, and thus with a smaller effective search space. Finally, we show how such landmark-based task enrichment can be combined in a mutually stratifying way with the *BFBB* search used for OSP planning, resulting in an incremental procedure that interleaves search and landmark discovery. The entire framework is independent of the OSP planner specifics, and in particular, of the heuristic functions it employs. Our empirical evaluation on a large set of OSP tasks confirms the effectiveness of the proposed approach.

2 PRELIMINARIES

Since both OSP and classical planning tasks are discussed in the paper, we use a formalism that is based on the standard STRIPS formalism with non-negative operator costs (cf. [17]), extended to OSP in line with the notation of our earlier paper on OSP [25].

Planning Tasks. A *planning task structure* is given by a pair $\langle V, O \rangle$, where V is a finite set of propositional state variables, and O is a finite set of operators. State variables are also called propositions or facts. A state $s \in 2^V$ is a subset of facts, representing the propositions which are currently true. Each operator $o \in O$ is associated with preconditions $\mathsf{pre}(o) \subseteq V$, add effects $\mathsf{add}(o) \subseteq V$, and delete effects $\mathsf{del}(o) \subseteq V$. Applying an operator o in s results in state $(s \setminus \mathsf{del}(o)) \cup \mathsf{add}(o)$, which we denote as $s[\![o]\!]$. The notation is only defined if o is applicable in s, i.e., if $\mathsf{pre}(o) \subseteq s$. Applying a sequence $\langle o_1, \ldots, o_k \rangle$ of operators to a state is defined inductively as $s[\![\epsilon]\!] := s$ and $s[\![\langle o_1, \ldots, o_k \rangle]\!] := (s[\![\langle o_1, \ldots, o_{k-1} \rangle]\!])[\![o_k]\!]$.

[1] Technion, Haifa, Israel, emails: {mirkis@tx}{dcarmel@ie}.technion.ac.il

A *classical planning task* $\Pi = \langle V, O; I, G, cost\rangle$ extends its structure $\langle V, O\rangle$ with an initial state $I \subseteq V$, a goal $G \subseteq V$, and a real-valued, nonnegative operator cost function $cost : O \rightarrow \mathbb{R}^{0+}$. An operator sequence π is called an s-plan if it is applicable in s, and $G \subseteq s[\![\pi]\!]$. The cost of s-plan π is $cost(\pi) := \sum_{o\in\pi} cost(o)$, and π is optimal if its cost is minimal among all s-plans. The objective in classical planning is to find an I-plan of as low cost as possible or prove that no I-plan exists. Optimal classical planning is devoted to searching for optimal I-plans only.

An *oversubscription planning (OSP) task* $\Pi = \langle V, O; I, cost, u, b\rangle$ extends its structure $\langle V, O\rangle$ with four components: an initial state $I \subseteq V$ and an operator cost function $cost : O \rightarrow \mathbb{R}^{0+}$ as above, plus a succinctly represented and efficiently computable state value function $u : S \rightarrow \mathbb{R}^{0+}$, and a cost budget $b \in \mathbb{R}^{0+}$. In what follows, we assume $u(s) = \sum_{v\in s} u(v)$, i.e., the value of state s is the sum of (mutually independent) values of propositions which are true in s. Conceptually, our results equally apply to general value functions, but the complexity of certain construction steps may vary between different families of value functions.

In OSP, an operator sequence π is called an s-plan if it is applicable in s, and $\sum_{o\in\pi} cost(o) \leq b$. While even an empty operator sequence is an s-plan for any state s, the objective in OSP is to find an I-plan that achieves as valuable a state as possible. By $\widehat{u}(\pi)$ we refer to the value of the end-state of π, that is, $\widehat{u}(\pi) = u(s[\![\pi]\!])$. *Optimal OSP* is devoted to searching for optimal I-plans only: An s-plan π is optimal if $\widehat{u}(\pi)$ is maximal among all the s-plans.

Heuristics. The two major ingredients of any heuristic-search planner are its search algorithm and heuristic function. In classical planning, the heuristic is typically a function $h : 2^V \rightarrow \mathbb{R}^{0+} \cup \{\infty\}$, with $h(s)$ estimating the cost $h^*(s)$ of optimal s-plans. A heuristic h is admissible if it is *lower-bounding*, i.e., $h(s) \leq h^*(s)$ for all states s. All common heuristic search algorithms for optimal classical planning, such as A^*, require admissible heuristics.

In OSP, a heuristic is a function $h : 2^V \times \mathbb{R}^{0+} \rightarrow \mathbb{R}^{0+}$, with $h(s, b)$ estimating the value $h^*(s, b)$ of optimal s-plans under cost budget b. A heuristic h is admissible if it is *upper-bounding*, i.e., $h(s, b) \geq h^*(s, b)$ for all states s and all cost budgets b. Here as well, search algorithms for optimal OSP, such as best-first branch-and-bound (*BFBB*) discussed later on in detail, require admissible heuristics.

Landmarks in Classical Planning. For a state s in a classical planning task Π, a landmark is a property of operator sequences that is satisfied by all s-plans [20]. For instance, a fact landmark for a state s is a fact that is true at some point in every s-plan. Several admissible landmark heuristics have been shown as extremely effective in optimal classical planning [21, 17, 5, 28]. These heuristics use extended notions of landmarks which are subsumed by *disjunctive action landmarks*. Each such landmark is a set of operators such that every s-plan contains at least one action from that set. In what follows we consider this popular notion of landmarks, and simply refer to disjunctive action landmarks for a state s as s-landmarks. For ease of presentation, most of our discussion will take place in the context of landmarks for the initial state of the task, and these will simply be referred to as *landmarks (for* Π*)*.

Deciding whether an operator set $L \subset O$ is a landmark for classical planning task Π is PSPACE-hard [29]. Therefore, all landmark heuristics employ methods for landmark discovery that are polynomial-time, sound, but incomplete. In what follows we assume access to such a procedure; the actual way the landmarks are discovered is tangential to our contribution. For a set $\mathcal{L}$ of s-landmarks, a *landmark cost function* $lcost : \mathcal{L} \rightarrow \mathbb{R}^{0+}$ is admissible if $\sum_{L\in\mathcal{L}} lcost(L) \leq h^*(s)$. For a singleton set $\mathcal{L} = \{L\}$, $lcost(L) := \min_{o\in L} cost(o)$ is a natural admissible landmark cost function, and it extends directly to non-singleton sets of pairwise disjoint landmarks. For more general sets of landmarks, $lcost$ can be devised (in polynomial time) via operator cost partitioning [23], either given $\mathcal{L}$ [21], or within the actual process of generating $\mathcal{L}$ [17].

3 "BRING ME SOMETHING" LANDMARKS

While landmarks play an important role in (both satisficing and optimal) classical planning, so far they have not been exploited in OSP. At first glance, this is probably no surprise: Since landmarks must hold in all plans, and the empty operator sequence is always a plan for any OSP task, the notion of landmark does not seem useful here.

Having said that, consider the anytime "output improvement" property of the forward-search branch-and-bound algorithms used for heuristic-search OSP. The empty plan is not interesting there not only because it is useless, but also because it is "found" by the search algorithm right at the get-go. In general, at all stages of the search, anytime algorithms like *BFBB* maintain the best-so-far solution π, and prune all branches that promise value lower or equal to $\widehat{u}(\pi)$. Hence, in principle, such algorithms may benefit from information about properties that are "satisfied by all plans with value larger than x." Unfortunately, it is not yet clear how the machinery for discovering classical planning landmarks can be adapted to discovery of such "value landmarks" while preserving polynomial-time complexity on general OSPs and arbitrary lower bounds x.

Looking at what is needed and what is available, our goal here is to exploit this machinery as it is. While the value of different s-plans in an OSP task Π varies between zero and the value of the optimal s-plan (which may also be zero), let an *ε-landmark* for state s be any property that is satisfied by any s-plan π that achieves *something valuable*. For instance, with the disjunctive action landmarks we use here, if $L \subseteq O$ is an ε-landmark for s, then every s-plan π with $\widehat{u}(\pi) > 0$ contains an operator from L. In what follows, unless stated otherwise, we focus on ε-landmarks for (the initial state of) Π.

Given an OSP task $\Pi = \langle V, O; I, cost, u, b\rangle$, let a classical planning task $\Pi_\varepsilon = \langle V_\varepsilon, O_\varepsilon; I_\varepsilon, cost_\varepsilon, G_\varepsilon\rangle$ be constructed as $V_\varepsilon = V \cup \{g\}$, $I_\varepsilon = I$, $G_\varepsilon = \{g\}$, and $O_\varepsilon = O \cup O_g$, where, for each proposition v with $u(v) > 0$, O_g contains an operator o_v with $\mathsf{pre}(o_v) = \{v\}$, $\mathsf{add}(o_v) = \{g\}$, $\mathsf{del}(o_v) = \emptyset$, and $cost_\varepsilon(o_v) = 0$. For all the original operators $o \in O$, $cost_\varepsilon(o) = cost(o)$. In other words, Π_ε extends the structure of Π with a set of zero-cost actions such that applying any of them indicates achieving a positive value in Π. In what follows, we refer to Π_ε as the *ε-compilation* of Π. Constructing Π_ε from Π is trivially polynomial time, and

Theorem 1 *For any OSP task* Π*, any landmark* L *for* Π_ε *such that* $L \subseteq O$ *is an* ε*-landmark for* Π*.*

With Theorem 1 in hand,[2] we can now derive ε-landmarks for Π using any method for classical planning landmark extraction, such as that employed by the LAMA planner [30] or the LM-Cut family of techniques [17, 5]. However, at first glance, the discriminative power of knowing "what is needed to achieve *something* valuable" seems to be negligible when it comes to deriving effective heuristic estimates for OSP. The good news is that, in OSP, such information can be effectively exploited in a slightly different way.

[2] Due to space limitations, all proofs are delegated to a full technical report [26].

```
BFBB ($\Pi = \langle V, O; I, cost, u, b \rangle$)
  open := new max-heap ordered by $f(n) = h(s[n], b - g(n))$
  open.insert(make-root-node($I$))
  closed:= $\emptyset$; best-cost:= $\emptyset$
  initialize best solution $n^* := I$
  while not open.empty()
      $n$ := open.pop-max()
      if $f(n) \leq u(s[n^*])$: break
      if $s[n] \notin$ closed or $g(n) <$ best-cost($s[n]$):
          closed:= closed $\cup \{s[n]\}$
          best-cost($s[n]$) := $g(n)$
          foreach $o \in O(s[n])$:
              $n'$ := make-node($s[n][\![o]\!]$)
              if $g(n') > b$ or $f(n') \leq u(s[n^*])$: continue
              if $u(s[n']) > u(s[n^*])$: update $n^* := n'$
              open.insert($n'$)
  return $n^*$
```

Figure 1. Best-first branch-and-bound (*BFBB*) search for OSP

3.1 ε-Landmarks and Budget Reduction

In the same way that A^* constitutes a canonical heuristic-search algorithm for optimal classical planning, anytime *best-first branch-and-bound (BFBB)* probably constitutes such an algorithm for optimal OSP.[3] Figure 1 depicts a pseudo-code description of *BFBB*. $s[n]$ there denotes the state associated with search node n. In *BFBB* for OSP, a node n with maximum evaluation function $h(s[n], b - g(n))$ is selected from the OPEN list. The duplicate detection and reopening mechanisms in *BFBB* are similar to those in A^*. In addition, *BFBB* maintains the best solution n^* found so far and uses it to prune all generated nodes evaluated no higher than $u(s[n^*])$. Likewise, complying with the semantics of OSP, all generated nodes n with cost-so-far $g(n)$ higher than the problem's budget b are also immediately pruned. When the OPEN list becomes empty or the node n selected from the list promises less than the lower bound, *BFBB* returns (the plan associated with) the best solution n^*, and if h is admissible, i.e., the h-based pruning of the generated nodes is sound, then the returned plan is guaranteed to be optimal.

Now, consider a schematic example of searching for an optimal plan for an OPS task Π with budget b, using *BFBB* with an admissible heuristic h. Suppose that there is only one sequence of (all unit-cost) operators, $\pi = \langle o_1, o_2, \dots, o_{b+1} \rangle$, applicable in the initial state of Π, and that the only positive value state along π is its end-state. While clearly no value higher than zero can be achieved in Π under the given budget of b, the search will continue beyond the initial state, unless $h(I, \cdot)$ counts the cost of all the $b+1$ actions of π. Now, suppose that $h(I, \cdot)$ counts only the cost of $\{o_i, \dots, o_{b+1}\}$ for some $i > 0$, but $\{o_1\}, \{o_2\}, \dots, \{o_{i-1}\}$ are all discovered to be ε-landmarks for Π. Given that, suppose that we modify Π by (a) setting the cost of operators $o_1, o_2, \dots, o_{i-1}$ to zero, and (b) *reducing the budget* to $b - i + 1$. This modification seems to preserve the semantics of Π, while on the modified task, *BFBB* with the same heuristic h will prune the initial state and thus establish without any search that the empty plan is an optimal plan for Π. Of course, the way Π is modified in this example is as simplistic as the example itself. Yet, this example does motivate the idea of *landmark-based budget reduction* for OSP, as well as illustrates the basic idea behind the *generically sound* task modifications that we discuss next.

Let $\Pi = \langle V, O; I, cost, u, b \rangle$ be an OSP task, $\mathcal{L} = \{L_1, \dots, L_n\}$ be a set of pairwise disjoint ε-landmarks for Π, and $lcost$ be an admissible landmark cost function from $\mathcal{L}$. Given that, a new OSP task $\Pi_{\mathcal{L}} = \langle V_{\mathcal{L}}, O_{\mathcal{L}}; I_{\mathcal{L}}, cost_{\mathcal{L}}, u_{\mathcal{L}}, b_{\mathcal{L}} \rangle$ with budget $b_{\mathcal{L}} =$

```
compile-and-BFBB ($\Pi = \langle V, O; I, cost, u, b \rangle$)
  $\Pi_\varepsilon$ = $\varepsilon$-compilation of $\Pi$
  $\mathcal{L}$ := a set of landmarks for $\Pi_\varepsilon$
  $lcost$ := admissible landmark cost function for $\mathcal{L}$
  $\Pi_{\mathcal{L}^*}$ := budget reducing compilation of $(\mathcal{L}, lcost)$ into $\Pi$
  $n^*$ := BFBB($\Pi_{\mathcal{L}^*}$)
  return plan for $\Pi$ associated with $n^*$
```

Figure 2. *BFBB* search with landmark-based budget reduction

$b - \sum_{i=1}^{n} lcost(L_i)$ is constructed as follows. The set of variables $V_{\mathcal{L}} = V \cup \{v_{L_1}, \dots, v_{L_n}\}$ extends V with a new proposition per ε-landmark in $\mathcal{L}$. These new propositions are all initially true, and $I_{\mathcal{L}} = I \cup \{v_{L_1}, \dots, v_{L_n}\}$. The value function $u_{\mathcal{L}} = u$ remains unchanged—the new propositions do not affect the value of the states. Finally, the operator set is extended as $O_{\mathcal{L}} = O \cup \bigcup_{i=1}^{n} O_{L_i}$, with O_{L_i} containing an operator $\overline{o}$ for each $o \in L_i$, with $\mathsf{pre}(\overline{o}) = \mathsf{pre}(o) \cup \{v_{L_i}\}$, $\mathsf{add}(\overline{o}) = \mathsf{add}(o)$, $\mathsf{del}(\overline{o}) = \mathsf{del}(o) \cup \{v_{L_i}\}$, and, importantly, $cost_{\mathcal{L}}(\overline{o}) = cost(o) - lcost(L_i)$. In other words, $\Pi_{\mathcal{L}}$ extends the structure of Π by mirroring the operators of each ε-landmark L_i with their "$lcost(L_i)$ cheaper" versions, while ensuring that these cheaper operators can be applied no more than once along an operator sequence from the initial state. At the same time, introduction of these discounted operators for L_i is compensated for by reducing the budget by precisely $lcost(L_i)$, leading to effective equivalence between Π and $\Pi_{\mathcal{L}}$.

Theorem 2 *Let $\Pi = \langle V, O; I, cost, u, b \rangle$ be an OSP task, $\mathcal{L}$ be a set of pairwise disjoint ε-landmarks for Π, $lcost$ be an admissible landmark cost function from $\mathcal{L}$, and $\Pi_{\mathcal{L}}$ be the respective budget reducing compilation of Π. For every π for Π with $\widehat{u}(\pi) > 0$, there is a plan $\pi_{\mathcal{L}}$ for $\Pi_{\mathcal{L}}$ with $\widehat{u}(\pi_{\mathcal{L}}) = \widehat{u}(\pi)$, and vice versa.*

The above budget reducing compilation of Π to $\Pi_{\mathcal{L}}$ is clearly polynomial time. Putting things together, we can see that the compile-and-BFBB procedure depicted in Figure 2 (1) generates an ε-compilation Π_ε of Π, (2) uses off-the-shelf tools for classical planning to generate a set of landmarks $\mathcal{L}$ for Π_ε and an admissible landmark cost function $lcost$, and (3) compiles $(\mathcal{L}, lcost)$ into Π, obtaining an OSP task $\Pi_{\mathcal{L}}$. The optimal solution for $\Pi_{\mathcal{L}}$ (and thus for Π) is then searched for using a search algorithm for optimal OSP such as *BFBB*.

Before we proceed to consider more general sets of landmarks, a few comments concerning the setup of Theorem 2 are now probably in place. First, if the reduced budget $b_{\mathcal{L}}$ turns out to be lower than the cost of the cheapest action applicable in the initial state, then no search is needed, and the empty plan can be reported as optimal right away. Second, zero-cost landmarks are useless in our compilation as much as they are useless in deriving landmark heuristics for optimal planning. Hence, $lcost$ in what follows is assumed to be strictly positive. Third, having both o and $\overline{o}$ applicable at a state of Π_ε brings no benefits yet adds branching to the search. Hence, in our implementation, for each landmark $L_i \in \mathcal{L}$ and each operator $o \in L_i$, the precondition of o in $O_{\mathcal{L}}$ is extended with $\{\neg v_{L_i}\}$. It is not hard to verify that this extension[4] preserves the correctness of $\Pi_{\mathcal{L}}$ in terms of Theorem 2. Finally, if the value of the initial state is not zero, that is, the empty plan has some positive value, then ε-compilation Π_ε of Π will have no positive cost landmarks at all. However, this can easily be fixed by considering as "valuable" only facts v such that both $u(v) > 0$ and $v \notin I$. For now we put this difficulty aside and assume that $\widehat{u}(\epsilon) = 0$. Later, however, we come back to consider it more systematically.

[3] *BFBB* is also extensively used for net-benefit planning [3, 7, 8], as well as some other variants of deterministic planning [4, 6].

[4] This modification requires augmenting our STRIPS-like formalism with negative preconditions, but this augmentation is straightforward.

3.2 Non-Disjoint ε-Landmarks

While the compilation $\Pi_{\mathcal{L}}$ above is sound for pairwise disjoint landmarks, this is not so for more general sets of ε-landmarks. For example, consider a planning task Π in which, for some operator o, we have $cost(o) = b$, $\widehat{u}(\langle o \rangle) > 0$, and $\widehat{u}(\pi) = 0$ for all other operator sequences $\pi \neq \langle o \rangle$. That is, a value greater than zero is achievable in Π, but only via the operator o. Suppose now that our set of ε-landmarks for Π is $\mathcal{L} = \{L_1, \dots, L_n\}$, $n > 1$, and that all of these ε-landmarks contain o. In this case, while the budget in $\Pi_{\mathcal{L}}$ is $b_{\mathcal{L}} = b - \sum_{i=1}^{n} lcost(L_i)$, the cost of the cheapest replica $\overline{o}$ of o, that is, the cost of the cheapest operator sequence achieving a non-zero value in Π, is $cost(o) - \min_{i=1}^{n} lcost(L_i) > b_{\mathcal{L}}$. Hence, no state with positive value will be reachable from $I_{\mathcal{L}}$ in $\Pi_{\mathcal{L}}$, and thus Π and $\Pi_{\mathcal{L}}$ are not "value equivalent" in the sense of Theorem 2.

Since non-disjoint landmarks can bring more information, and they are typical to outputs of standard techniques for landmark extraction in classical planning, we now present a different, slightly more involved, compilation that is both polynomial and sound for arbitrary sets of ε-landmarks. Let $\Pi = \langle V, O; I, cost, u, b \rangle$ be an OSP task, $\mathcal{L} = \{L_1, \dots, L_n\}$ be a set of ε-landmarks for Π, and $lcost$ be an admissible landmark cost function from $\mathcal{L}$. For each operator o, let $\mathcal{L}(o)$ denote the set of all landmarks in $\mathcal{L}$ that contain o. Given that, a new OSP task $\Pi_{\mathcal{L}^*} = \langle V_{\mathcal{L}^*}, O_{\mathcal{L}^*}; I_{\mathcal{L}^*}, cost_{\mathcal{L}^*}, u_{\mathcal{L}^*}, b_{\mathcal{L}^*} \rangle$ is constructed as follows. Similarly to $\Pi_{\mathcal{L}}$, we have $b_{\mathcal{L}^*} = b - \sum_{i=1}^{n} lcost(L_i)$, $V_{\mathcal{L}^*} = V \cup \{v_{L_1}, \dots, v_{L_n}\}$, $I_{\mathcal{L}^*} = I \cup \{v_{L_1}, \dots, v_{L_n}\}$, and $u_{\mathcal{L}^*} = u$. The operator set $O_{\mathcal{L}^*}$ extends O with two sets of operators:

- For each operator $o \in O$ that participates in some landmark from $\mathcal{L}$, $O_{\mathcal{L}^*}$ contains an action $\overline{o}$ with $\mathsf{pre}(\overline{o}) = \mathsf{pre}(o) \cup \{v_L \mid L \in \mathcal{L}(o)\}$, $\mathsf{add}(\overline{o}) = \mathsf{add}(o)$, $\mathsf{del}(\overline{o}) = \mathsf{del}(o) \cup \{v_L \mid L \in \mathcal{L}(o)\}$, $cost_{\mathcal{L}^*}(\overline{o}) = cost(o) - \sum_{L \in \mathcal{L}(o)} lcost(L)$.
- For each $L \in \mathcal{L}$, $O_{\mathcal{L}^*}$ contains an action $get(L)$ with $\mathsf{pre}(get(L)) = \{\neg v_L\}$, $\mathsf{add}(get(L)) = \{v_L\}$, $\mathsf{del}(get(L)) = \emptyset$, $cost_{\mathcal{L}^*}(get(L)) = lcost(L)$.

For example, let $\mathcal{L} = \{L_1, L_2, L_3\}$, $L_1 = \{a, b\}$, $L_2 = \{b, c\}$, $L_3 = \{a, c\}$, with all operators having the cost of 2, and let $lcost(L_1) = lcost(L_2) = lcost(L_3) = 1$. In $\Pi_{\mathcal{L}^*}$, we have $V_{\mathcal{L}^*} = V \cup \{v_{L_1}, v_{L_2}, v_{L_3}\}$ and $O_{\mathcal{L}^*} = O \cup \{\overline{a}, \overline{b}, \overline{c}, get(L_1), get(L_2), get(L_3)\}$, with, e.g., $\mathsf{pre}(\overline{a}) = \mathsf{pre}(a) \cup \{v_{L_1}, v_{L_3}\}$, $\mathsf{add}(\overline{a}) = \mathsf{add}(a)$, $\mathsf{del}(\overline{a}) = \mathsf{del}(a) \cup \{v_{L_1}, v_{L_3}\}$, and $cost_{\mathcal{L}^*}(\overline{a}) = 0$, and, for $get(L_1)$, $\mathsf{pre}(get(L_1)) = \mathsf{del}(get(L_1)) = \emptyset$, $\mathsf{add}(get(L_1)) = \{v_{L_1}\}$, and $cost_{\mathcal{L}^*}(get(L_1)) = 1$.

Theorem 3 *Let $\Pi = \langle V, O; I, cost, u, b \rangle$ be an OSP task and $\Pi_{\mathcal{L}^*}$ a budget reducing compilation of Π. For every π for Π with $\widehat{u}(\pi) > 0$, there is a plan $\pi_{\mathcal{L}^*}$ for $\Pi_{\mathcal{L}^*}$ with $\widehat{u}(\pi_{\mathcal{L}^*}) = \widehat{u}(\pi)$, and vice versa.*

4 ε-LANDMARKS & INCREMENTAL *BFBB*

As we discussed earlier, if the value of the initial state is not zero, i.e., the empty plan has some positive value, then the basic ε-compilation Π_ε of Π will have no positive cost landmarks at all. In passing we noted that this small problem can be remedied by considering as "valuable" only facts v such that both $u(v) > 0$ and $v \notin I$. We now consider this aspect of OSP more closely, and show how ε-landmarks discovery and incremental revelation of plans by *BFBB* can be combined in a mutually stratifying way.

Let $\Pi = \langle V, O; I, cost, u, b \rangle$ be the OSP task of our interest, and suppose we are *given* a set of plans $\pi_1, \dots, \pi_n$ for Π. If so, then we are no longer interested in searching for plans that "achieve something," but in searching for plans that achieve *something beyond what $\pi_1, \dots, \pi_n$ already achieve*. For $1 \leq i \leq n$, let $s_i = I[\![\pi_i]\!]$ be the end-state of π_i, and for any set of propositions $s \subseteq V$, let $\mathsf{goods}(s) \subseteq s$ be the set of all facts $v \in s$ such that $u(v) > 0$. If a new plan π with end-state s achieves something beyond what $\pi_1, \dots, \pi_n$ already achieve, then $\mathsf{goods}(s) \setminus \mathsf{goods}(s_i) \neq \emptyset$ for *all* $1 \leq i \leq n$.

```
inc-compile-and-BFBB ($\Pi = \langle V, O; I, cost, u, b \rangle$)
  initialize global variables:
      $n^* := I$            // best solution so far
      $S_{\text{ref}} := \{I\}$     // current reference states
  loop:
    $\Pi_{(\varepsilon, S_{\text{ref}})}$ = ($\varepsilon, S_{\text{ref}}$)-compilation of $\Pi$
    $\mathcal{L}$ := a set of landmarks for $\Pi_{(\varepsilon, S_{\text{ref}})}$
    $lcost$ := admissible landmark cost function from $\mathcal{L}$
    $\Pi_{\mathcal{L}^*}$ := budget reducing compilation of $(\mathcal{L}, lcost)$ into $\Pi$
    if inc-BFBB($\Pi_{\mathcal{L}^*}, S_{\text{ref}}, n^*$) = $done$:
      return plan for $\Pi$ associated with $n^*$

inc-BFBB ($\Pi, S_{\text{ref}}, n^*$)
  open := new max-heap ordered by $f(n) = h(s[n], b - g(n))$
  open.insert(make-root-node($I$))
  closed:= $\emptyset$  best-cost:= $\emptyset$;
  while not open.empty()
    $n$ := open.pop-max()
    if goods($s[n]$) $\not\subseteq$ goods($s'$) for all $s' \in S_{\text{ref}}$:
      $S_{\text{ref}} := S_{\text{ref}} \cup \{s[n]\}$
      if termination criterion: return $updated$
    if $f(n) \leq u(s[n^*])$: break
      // . . .
      // similar to BFBB in Figure 1
  return $done$
```

Figure 3. Iterative *BFBB* with landmark enhancement

We now put this observation to work. Given an OSP task $\Pi = \langle V, O; I, cost, u, b \rangle$ and a set of reference states $S_{\text{ref}} = \{s_1, \dots, s_n\}$ of Π, let a classical planning task $\Pi_{(\varepsilon, S_{\text{ref}})} = \langle V_\varepsilon, O_\varepsilon; I_\varepsilon, G_\varepsilon, cost_\varepsilon \rangle$ be constructed as follows. The variable set $V_\varepsilon = V \cup \{x_1, \dots, x_n, search, collect\}$ extends V with a new proposition per state in S_{ref}, plus two auxiliary control variables. In the initial state, all the new variables but *search* are false, i.e., $I_\varepsilon = I \cup \{search\}$, and the goal is $G_\varepsilon = \{x_1, \dots, x_n\}$. The operator set O_ε contains three sets of operators: First, each operator $o \in O$ is represented in O_ε by an operator $\overline{o}$, with the only difference between o and $\overline{o}$ (including cost) being that $\mathsf{pre}(\overline{o}) = \mathsf{pre}(o) \cup \{search\}$. We denote this set of new operators $\overline{o}$ by $\overline{O}$. Second, for each $s_i \in S_{\text{ref}}$ and each value-carrying fact g that is *not* in s_i, i.e., for each $g \in \mathsf{goods}(V) \setminus s_i$, O_ε contains a zero-cost action $o_{i,g}$ with $\mathsf{pre}(o_{i,g}) = \{g, collect\}$, $\mathsf{add}(o_{i,g}) = \{x_i\}$, $\mathsf{del}(o_{i,g}) = \emptyset$. Finally, O_ε contains a zero-cost action *finish* with $\mathsf{pre}(finish) = \emptyset$, $\mathsf{del}(finish) = \{search\}$, and $\mathsf{add}(finish) = \{collect\}$.

It is easy to verify that (1) the goal G_ε cannot be achieved without applying the *finish* operator, (2) the operators $\overline{o}$ can be applied only before *finish*, and (3) the subgoal achieving operators $o_{i,g}$ can be applied only after *finish*. Hence, the first part of any plan for $\Pi_{(\varepsilon, S_{\text{ref}})}$ determines a plan for Π, and the second part "verifies" that the end-state of that plan achieves a subset of value-carrying propositions $\mathsf{goods}(V)$ that is included in no state from S_{ref}.[5]

Theorem 4 *Let $\Pi = \langle V, O; I, cost, u, b \rangle$ be an OSP task, $S_{ref} = \{s_1, \dots, s_n\} \subseteq 2^V$ be a subset of Π's states, and L be a landmark for $\Pi_{(\varepsilon, S_{ref})}$ such that $L \subseteq \overline{O}$. For any plan π for Π such that $\mathsf{goods}(I[\![\pi]\!]) \setminus \mathsf{goods}(s_i) \neq \emptyset$ for all $s_i \in S_{ref}$, π contains an instance of at least one operator from $L' = \{o \mid \overline{o} \in L\}$.*

Theorem 4 allows us to define an iterative version of *BFBB*, successive iterations of which correspond to running the regular *BFBB*

[5] This "plan in two parts" technique appears to be helpful in many planning formalism compilations; see, e.g., [24].

on successively more informed $(\varepsilon, S_{\text{ref}})$-compilations of Π, with the states discovered at iteration i making the $(\varepsilon, S_{\text{ref}})$-compilation used at iteration $i+1$ more informed. The respective procedure inc-compile-and-BFBB is depicted in Figure 3. This procedure maintains a set of reference states S_{ref} and the best solution so far n^*, and loops over calls to inc-BFBB, a modified version of *BFBB*. At each iteration of the loop, inc-BFBB is called with an $(\varepsilon, S_{\text{ref}})$-compilation of Π, created on the basis of the *current* S_{ref} and n^*, and it is provided with access to both S_{ref} and n^*. The reference set S_{ref} is then extended by inc-BFBB with all the non-redundant value-carrying states discovered during the search, and n^* is updated if the search discovers nodes of higher value.

If and when the OPEN list becomes empty or the node n selected from the list promises less than the lower bound, inc-BFBB returns an indicator, *done*, that the best solution n^* found so far, across the iterations of inc-compile-and-BFBB, is optimal. In that case, inc-compile-and-BFBB leaves its loop and extracts that optimal plan from n^*. However, inc-BFBB may also terminate in a different way, if a certain complementary termination criterion is satisfied. The latter criterion comes to assess whether the updates to S_{ref} performed in the current session of *BFBB* warrant updating the $(\varepsilon, S_{\text{ref}})$-compilation and restarting the search.[6] If terminated this way, inc-BFBB returns a respective indicator, and inc-compile-and-BFBB goes into another iteration of its loop, with the *updated* S_{ref} and n^*.

5 EMPIRICAL EVALUATION

We have implemented a prototype heuristic-search OSP solver on top of the Fast Downward planner [16]. The implementation included[7]:

- $(\varepsilon, S_{\text{ref}})$-compilation of OSP tasks Π;
- Generation of disjunctive action landmarks for $(\varepsilon, S_{\text{ref}})$-compilations using the LM-Cut procedure [17] of Fast Downward;
- The incremental *BFBB* procedure inc-compile-and-BFBB from the previous section, with the search termination criterion being satisfied (only) if the examined node n improves over current value lower bound; and
- An additive abstraction heuristic from the framework of Mirkis and Domshlak [25], incorporating (i) an ad hoc action cost partition over k projections of the planning task onto connected subsets of ancestors of the respective k goal variables in the causal graph, and (ii) a value partition that associates the value of each goal (only) with the respective projection. The size of each projection was limited to 1000 abstract states.

After some preliminary evaluation, we also added two (optimality preserving) enhancements to the search. First, the auxiliary variables of our compilations increase the dimensionality of the problem, and this is well known to negatively affect the quality of the projection abstractions. Hence, we devised the projections with respect to the *original* OSP problem Π, and the open list was ordered as if the search is done on the original problem, that is, by $h(s[n]^{\downarrow V},\ b - g(n) + \sum_{v_L \not\in s[n]} lcost(L))$, where $s[n]^{\downarrow V}$ is the projection of the state $s[n]$ on the variables of the original OSP task Π. This change in heuristic evaluation is sound, as Theorem 3 in particular implies that any admissible heuristic for Π is also an admissible heuristic for $\Pi_{\mathcal{L}^*}$, and vice versa. Second, when a new node n is generated, we check whether $g(n) + \sum_{v_L \not\in s[n]} lcost(L) \geq g(n') + \sum_{v_L \not\in s[n']} lcost(L)$, for some previously generated node n' that corresponds to the same state of the original problem Π, i.e., $s[n']^{\downarrow V} = s[n]^{\downarrow V}$. If so, then n is pruned right away. Optimality preservation of this enhancement is established in [26].

Since, unlike classical and net-benefit planning, OSP lacks standard benchmarks for comparative evaluation, we have cast in this role the STRIPS classical planning domains from the International Planning Competitions (IPC) 1998-2006. This "translation" to OSP was done by associating a separate unit-value with each sub-goal. The evaluation included the regular *BFBB* planning for Π, solving Π using landmark-based compilation via compile-and-BFBB, and the straightforward setting of inc-compile-and-BFBB described above. All three approaches were evaluated under the blind heuristic and the additive abstraction heuristic as above.

Figure 4 depicts the results of our evaluation in terms of expanded nodes on all the aforementioned IPC tasks for which we could determine offline the minimal cost needed to achieve all the goals in the task. Each task was approached under four different budgets, corresponding to 25%, 50%, 75%, and 100% of the minimal cost needed to achieve all the goals in the task, and each run was restricted to 10 minutes. Figures 4(a) and 4(b) compare the performance of *BFBB* and compile-and-BFBB with blind (a) and abstraction (b) heuristics. Figures 4(c) and 4(d) provide a similar comparison between *BFBB* and inc-compile-and-BFBB. [8]

As Figure 4 shows, the results are satisfactory. With no informative heuristic guidance at all, the number of nodes expanded by compile-and-BFBB was typically much lower than the number of nodes expanded by *BFBB*, with the difference reaching three orders of magnitude more than once. Of the 760 task/budget pairs behind Figure 4a, 81 pairs were solved by compile-and-BFBB with no search at all (by proving that no plan can achieve value higher than that of the initial state), while, unsurprisingly, only 4 of these tasks were solved with no search by *BFBB*.

As expected, the value of landmark-based budget reduction is lower when the search is equipped with a meaningful heuristic (Figure 4b). Yet, even with our abstraction heuristic in hand, the number of nodes expanded by compile-and-BFBB was often substantially lower than the number of nodes expanded by *BFBB*. Here, *BFBB* and compile-and-BFBB solved with no search 39 and 85 task/budget pairs, respectively. Finally, despite the rather ad hoc setting of our incremental inc-compile-and-BFBB procedure, switching from compile-and-BFBB to inc-compile-and-BFBB was typically beneficial, though much deeper investigation and development of inc-compile-and-BFBB is obviously still required.

ACKNOWLEDGMENTS This work was partially supported by the ISF grant 1045/12, and the EOARD grant FA8655-12-1-2096.

REFERENCES

[1] J. A. Baier, F. Bacchus, and S. A. McIlraith, 'A heuristic search approach to planning with temporally extended preferences', in *IJCAI*, pp. 1808–1815, (2007).

[2] J. Benton, M. Do, and S. Kambhampati, 'Anytime heuristic search for partial satisfaction planning', *AIJ*, **173**(5-6), 562–592, (2009).

[3] J. Benton, M. van den Briel, and S. Kambhampati, 'A hybrid linear programming and relaxed plan heuristic for partial satisfaction planning problems', in *ICAPS*, pp. 34–41, (2007).

[4] B. Bonet and H. Geffner, 'Heuristics for planning with penalties and rewards formulated in logic and computed through circuits', *AIJ*, **172**(12-13), 1579–1604, (2008).

[6] While the optimality of the algorithm holds for *any* such termination condition, the latter should greatly affect the runtime efficiency of the algorithm.

[7] We are not aware of any other domain-independent planner for optimal OSP.

[8] We do not compare here between the running times, but the per-node CPU time overhead due to landmark-based budget reduction was $\leq 10\%$.

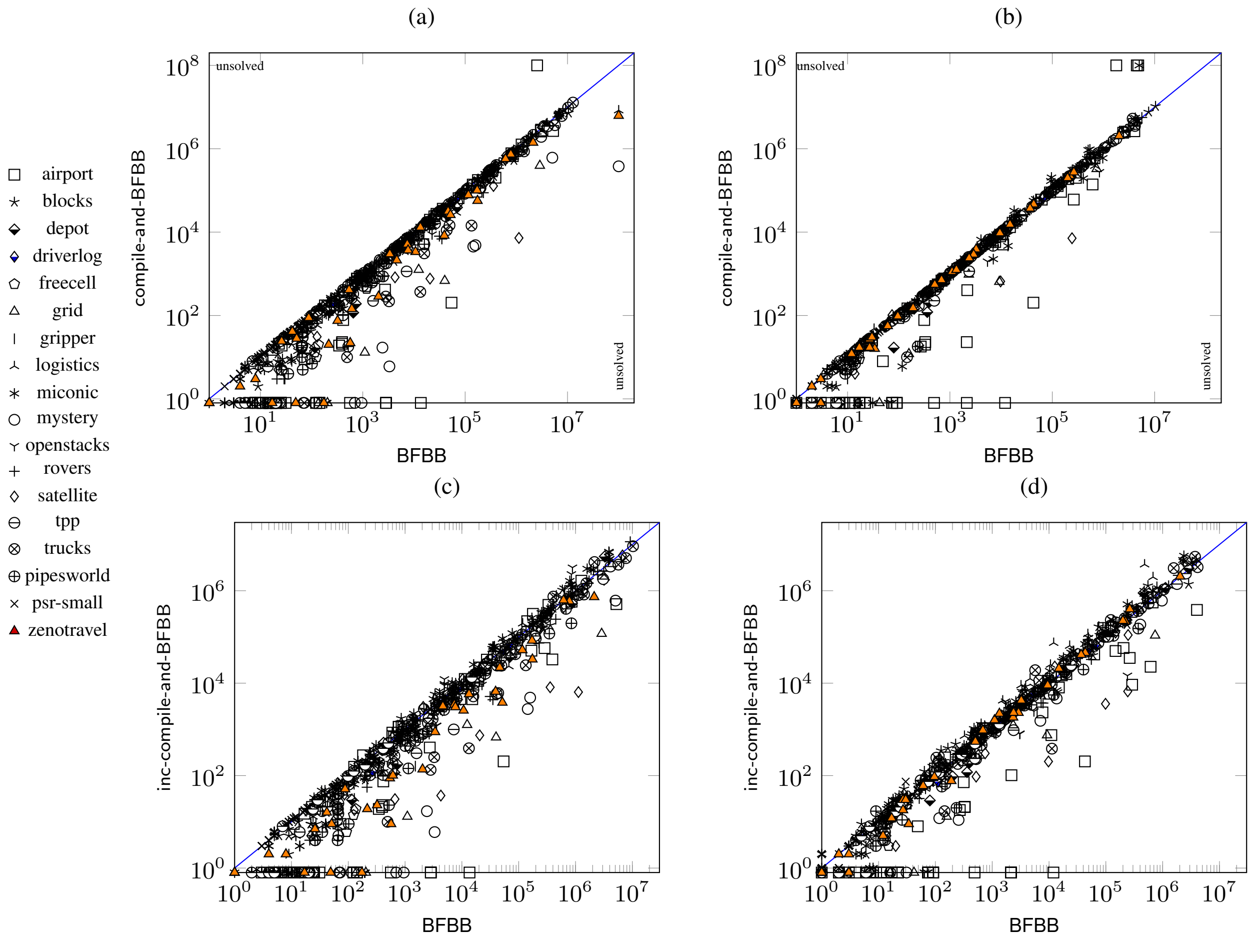

Figure 4. Comparative view of empirical results in terms of expanded nodes: (a) & (b) *BFBB* vs. compile-and-BFBB, (c) & (d) *BFBB* vs. inc-compile-and-BFBB, with blind ((a) & (c)) and additive projections ((b) & (d)) heuristics

[5] B. Bonet and M. Helmert, 'Strengthening landmark heuristics via hitting sets', in *ECAI*, pp. 329–334, (2010).
[6] R. Brafman and Y. Chernyavsky, 'Planning with goal preferences and constraints', in *ICAPS*, pp. 182–191, Monterey, CA, (2005).
[7] A. J. Coles and A. Coles, 'LPRPG-P: Relaxed plan heuristics for planning with preferences', in *ICAPS*, (2011).
[8] M. B. Do, J. Benton, M. van den Briel, and S. Kambhampati, 'Planning with goal utility dependencies', in *IJCAI*, pp. 1872–1878, (2007).
[9] C. Domshlak, M. Katz, and S. Lefler, 'Landmark-enhanced abstraction heuristics', *AIJ*, **189**, 48–68, (2012).
[10] S. Edelkamp, 'Planning with pattern databases', in *ECP*, pp. 13–24, (2001).
[11] R. E. Fikes and N. Nilsson, 'STRIPS: A new approach to the application of theorem proving to problem solving', *AIJ*, **2**, 189–208, (1971).
[12] A. Gerevini, A. Saetti, and I. Serina, 'An approach to efficient planning with numerical fluents and multi-criteria plan quality', *AIJ*, **172**(8-9), 899–944, (2008).
[13] P. Haslum, 'Heuristics for bounded-cost search', in *ICAPS*, (2013).
[14] P. Haslum, A. Botea, M. Helmert, B. Bonet, and S. Koenig, 'Domain-independent construction of pattern database heuristics for cost-optimal planning', in *AAAI*, pp. 1007–1012, (2007).
[15] P. Haslum and H. Geffner, 'Heuristic planning with time and resources', in *ECP*, (2001).
[16] M. Helmert, 'The Fast Downward planning system', *JAIR*, **26**, 191–246, (2006).
[17] M. Helmert and C. Domshlak, 'Landmarks, critical paths and abstractions: What's the difference anyway?', in *ICAPS*, pp. 162–169, (2009).
[18] M. Helmert, P. Haslum, and J. Hoffmann, 'Flexible abstraction heuristics for optimal sequential planning', in *ICAPS*, pp. 200–207, (2007).
[19] J. Hoffmann, C. P. Gomes, B. Selman, and H. A. Kautz, 'SAT encodings of state-space reachability problems in numeric domains', in *IJCAI*, pp. 1918–1923, (2007).
[20] J. Hoffmann, J. Porteous, and L. Sebastia, 'Ordered landmarks in planning', *JAIR*, **22**, 215–278, (2004).
[21] E. Karpas and C. Domshlak, 'Cost-optimal planning with landmarks', in *IJCAI*, pp. 1728–1733, (2009).
[22] M. Katz and C. Domshlak, 'Implicit abstraction heuristics', *JAIR*, **39**, 51–126, (2010).
[23] M. Katz and C. Domshlak, 'Optimal admissible composition of abstraction heuristics', *AIJ*, **174**, 767–798, (2010).
[24] E. Keyder and H. Geffner, 'Soft goals can be compiled away', *JAIR*, **36**, 547–556, (2009).
[25] V. Mirkis and C. Domshlak, 'Abstractions for oversubscription planning', in *ICAPS*, (2013).
[26] V. Mirkis and C. Domshlak, 'Landmarks in oversubscription planning', Technical Report IE/IS-2014-01, Technion, (2014).
[27] H. Nakhost, J. Hoffmann, and M. Müller, 'Resource-constrained planning: A Monte Carlo random walk approach', in *ICAPS*, (2012).
[28] F. Pommerening and M. Helmert, 'Incremental LM-Cut', in *ICAPS*, (2013).
[29] J. Porteous, L. Sebastia, and J. Hoffmann, 'On the extraction, ordering, and usage of landmarks in planning', in *ECP*, (2001).
[30] S. Richter, M. Helmert, and M. Westphal, 'Landmarks revisited', in *AAAI*, pp. 975–982, (2008).
[31] R. Sanchez and S. Kambhampati, 'Planning graph heuristics for selecting objectives in over-subscription planning problems', in *ICAPS*, pp. 192–201, (2005).
[32] D. Smith, 'Choosing objectives in over-subscription planning', in *ICAPS*, pp. 393–401, (2004).
[33] J. T. Thayer and W. Ruml, 'Bounded suboptimal search: A direct approach using inadmissible estimates', in *IJCAI*, pp. 674–679, (2011).
[34] J. T. Thayer, R. T. Stern, A. Felner, and W. Ruml, 'Faster bounded-cost search using inadmissible estimates', in *ICAPS*, (2012).