Oversubscription Planning as Classical Planning with Multiple Cost Functions

Michael Katz
IBM Research
Yorktown Heights, NY, USA
michael.katz1@ibm.com

Emil Keyder
Invitae
San Francisco, CA, USA
emilkeyder@gmail.com

Florian Pommerening
University of Basel
Basel, Switzerland
florian.pommerening@unibas.ch

Dominik Winterer
ETH Zurich
Zurich, Switzerland*
dominik.winterer@inf.ethz.ch

Abstract

The aim of classical planning is to minimize the summed cost of operators among those plans that achieve a fixed set of goals. Oversubscription planning (OSP), on the other hand, seeks to maximize the utility of the set of facts achieved by a plan, while keeping the cost of the plan at or below some specified bound. Here, we investigate the use of reformulations that yield planning problems with two separate cost functions, but no utilities, for solving OSP tasks. Such reformulations have also been proposed in the context of net-benefit planning, where the planner tries to maximize the difference between the utility achieved and the cost of the plan. One of our reformulations is adapted directly from that setting, while the other is novel. In both cases, they allow for easy adaptation of existing classical planning heuristics to the OSP problem within a simple branch and bound search. We validate our approach using state of the art admissible heuristics in this framework, and report our results.

Introduction

The aim of classical planning is to find a sequence of actions that when applied in the initial state of the problem results in a state in which all goals are true. Partial satisfaction planning, on the other hand, aims to maximize the utility of the goals achieved by the plan, and may choose to ignore some subset of the goals if the cost of achieving them is too high. Two distinct flavors of partial satisfaction planning have been studied in the past: In *net-benefit* planning, action costs and goal utilities are comparable, and solution quality is measured as the net difference between the utility of the obtained end state and the solution cost (van den Briel et al. 2004). In contrast, in *oversubscription planning* the solution cost and state values are assumed to be incomparable (Smith 2004). To take cost into account, a *budget*, or bound, on plan cost is introduced, and the objective is to maximize the utility of the final state under this constraint.

Heuristic search is among the most successful approaches to classical planning, and many different heuristics have been proposed for this purpose. These are typically classified into four families: *abstractions*, (e.g., (Culberson and Schaeffer 1998; Edelkamp 2001; Helmert et al. 2014; Katz and Domshlak 2010a)), *delete relaxations*, (e.g., (Bonet and Geffner 2001; Hoffmann and Nebel 2001; Keyder and Geffner 2008; Domshlak, Hoffmann, and Katz 2015)), *critical paths* (Haslum and Geffner 2000), and *landmarks*, (e.g., (Richter, Helmert, and Westphal 2008; Karpas and Domshlak 2009; Helmert and Domshlak 2009; Keyder, Richter, and Helmert 2010)). The basic principle behind all of these heuristics is the same – the task is relaxed until it satisfies the requirements of some tractable fragment of the planning problem, and a solution to this relaxed task is used as a guide in the original problem space.

Heuristic search has also been very successful in the net-benefit planning setting, when used in combination with a transformation into classical planning (Keyder and Geffner 2009). Taking advantage of the fact that action costs and goal utilities are assumed to be comparable, this transformation encodes the decision to achieve or ignore an individual goal as an explicit action, allowing informed heuristics to reason about the tradeoffs between its cost and utility.

In optimal oversubscription planning, little previous work has focused on heuristic search, and progress has been somewhat slower than in the net-benefit setting. A significant performance improvement was first reported by Mirkis and Domshlak (2013). They exploited *explicit abstractions* (Edelkamp 2001), which were tractable due to their small size. The abstract oversubscription planning problems were additively composed into informative admissible estimates which are then used to prune states in a branch-and-bound search. The approach turned out to work well in practice: in some cases the search space was reduced by three orders of magnitude compared to the baseline algorithm. Later, Mirkis and Domshlak (2014) exploited the notion of *landmarks* for task reformulation, enriching the task with reachability information. Katz and Mirkis (2016) characterized tractable fragments of oversubscription planning tasks according to causal graph structure and variable domain sizes, and derived admissible estimates from these fragments. Unfortunately, even the simplest fragment under this characterization was found to be not solvable in polynomial time. Additional restrictions are therefore required for tractability, similarly to those that were previously exploited in deriving heuristics for classical planning. *Value-driven landmarks* have recently been proposed to extend landmarks to over-

*The contribution to this work was done in a Master thesis at Universities of Basel and Freiburg.

subscription planning (Muller and Karpas 2018).

Our aim in this work is to lay the grounds for adapting heuristics for classical planning to oversubscription planning. Toward this end, we propose a reformulation of the oversubscription planning task as a classical planning task with two cost functions on operators. The first of these corresponds to the original operator costs and ensures that we restrict the set of solutions to those whose cost does not exceed the specified budget, while the second is an encoding of the utilities in the problem. This encoding can be obtained through the compilation used in the net-benefit setting by Keyder and Geffner (2009), but we also propose a new reformulation in which action costs are given by the *change* in utility that results from the application of an action. Using merge-and-shrink (Helmert et al. 2014) and LM-cut (Helmert and Domshlak 2009) heuristics as examples, we show how both reformulations can be solved using existing heuristics. We evaluate both, and discuss directions for possible future improvements.

Another contribution of our work is a first attempt at standardizing the benchmark set for oversubscription planning. For that, we introduce additional constructs to the PDDL language to specify state-additive utility functions and a cost budget. Further, we adapt the Fast Downward translator (Helmert 2006) to parse these problem descriptions, and create a collection of oversubscription planning benchmarks from the classical STRIPS domains used in past International Planning Competitions.

Background

In line with the SAS formalism[1] for deterministic planning (Bäckström and Klein 1991), a *planning task structure* is given by a pair $\langle V, O\rangle$, where V is a set of *state variables*, and O is a finite set of *operators*. Each state variable $v \in V$ has a finite domain $dom(v)$. A pair $\langle v, \vartheta\rangle$ with $v \in V$ and $\vartheta \in dom(v)$ is called a *fact*. A partial assignment to V is called a *partial state*. The subset of variables instantiated by a partial state p is denoted by $\mathcal{V}(p) \subseteq V$. Often it is convenient to view partial state p as a set of facts with $\langle v, \vartheta\rangle \in p$ iff $p[v] = \vartheta$. We say a partial state s is a *state* iff $\mathcal{V}(p) = V$. Partial state p is *consistent* with state s if s and p agree on all variables in $\mathcal{V}(p)$. We denote the set of states of a planning task structure $\langle V, O\rangle$ by S.

Each *operator* o is a pair $\langle \mathsf{pre}(o), \mathsf{eff}(o)\rangle$ of partial states called *preconditions* and *effects*. We assume that all operators are in SAS format i.e. $\mathcal{V}(\mathsf{eff}(o)) \subseteq \mathcal{V}(\mathsf{pre}(o))$ for all $o \in O$. An *operator cost* function is a mapping $\mathcal{C} : O \to \mathbb{R}$. While in classical planning the operator cost functions $\mathcal{C}$ are typically assumed to be non-negative, we emphasize that in general cost functions $\mathcal{C}$ can take negative values as well.

An operator o is applicable in a state $s \in S$ iff $s[v] = \mathsf{pre}(o)[v]$ for all $v \in \mathcal{V}(\mathsf{pre}(o))$. Applying o changes the value of each $v \in \mathcal{V}(\mathsf{eff}(o))$ to $\mathsf{eff}(o)[v]$. The resulting state is denoted by $s[\![o]\!]$. An operator sequence $\pi = \langle o_1, \ldots, o_k\rangle$ is applicable in s if there exist states $s_0, \cdots, s_k$ such that (i) $s_0 = s$, and (ii) for each $1 \leq i \leq k$, o_i is applicable in s_{i-1} and $s_i = s_{i-1}[\![o_i]\!]$. We denote the state s_k by $s[\![\pi]\!]$ and call it the end state of π.

[1]Not to be confused with the more commonly used SAS$^+$ formalism (Bäckström and Nebel 1995).

Oversubscription Planning An *oversubscription planning (OSP) task* $\Pi_{\text{OSP}} = \langle V, O, s_I, \mathcal{C}, \mathsf{u}, \mathsf{B}\rangle$ extends a planning task structure $\langle V, O\rangle$ with an *initial state* $s_I \in S$, a non-negative operator cost function $\mathcal{C}$, a *utility* function $\mathsf{u} : S \to \mathbb{R}^{0+}$, and a *cost bound* $\mathsf{B} \in \mathbb{R}^{0+}$.

An operator sequence π is called an s-plan for Π_{OSP} if it is applicable in s_I, and $\sum_{o\in\pi} \mathcal{C}(o) \leq \mathsf{B}$. We call an s_I-plan a *plan* for Π_{OSP}. The utility $\hat{\mathsf{u}}(\pi)$ of a plan is given by the utility of the end-state of π, that is, $\hat{\mathsf{u}}(\pi) = \mathsf{u}(s_I[\![\pi]\!])$. A plan π for Π_{OSP} is *optimal* if $\hat{\mathsf{u}}(\pi)$ is maximal among all plans. While the empty operator sequence is a valid plan for any OSP task, the objective in oversubscription planning is to find a plan with high utility, and *optimal oversubscription planning* aims to find provably optimal plans only. In what follows, we restrict our attention to *additive utility functions*, computed as the total utility of the facts in the final state of the plan: $\mathsf{u}(s) = \sum_{f\in s} \mathsf{u}'(f)$, where u' is a function mapping facts to non-negative real values. Slightly abusing notation, we denote u' by u in the following.

A heuristic for the OSP task $\Pi_{\text{OSP}} = \langle V, O, s_I, \mathcal{C}, \mathsf{u}, \mathsf{B}\rangle$ over states S is a mapping $h : S \times \mathbb{R}^{0+} \to \mathbb{R}^{0+}$, from state-budget pairs to non-negative real values. The *perfect heuristic* h^* maps each state $s \in S$ and bound $\mathsf{b} \in \mathbb{R}^{0+}$ to the utility $\hat{\mathsf{u}}(\pi^*)$ of an optimal plan π^* for the OSP task $\langle V, O, s, \mathcal{C}, \mathsf{u}, \mathsf{b}\rangle$. A heuristic h is admissible if $h \geq h^*$. Note that admissible heuristics *overestimate* the optimal utility instead of underestimating the optimal plan cost as in classical planning.

Multiple Cost Function Planning

We now present an (otherwise classical) planning formalism that limits the set of feasible solutions using *secondary cost functions* and *bounds* over these functions. The primary advantage of this formalism is that while it allows negative values in the primary cost function, it does not allow utilities. We will show below how OSP problems can be naturally expressed in this formalism, and indeed that negative values for the primary cost function are not required, depending on the choice of reformulation.

Definition 1. *A multiple cost function (MCF) planning task is a tuple* $\Pi_{\text{MCF}} = \langle V, O, s_I, G, \mathcal{C}_0, \mathscr{C}\rangle$*, where* $\langle V, O\rangle$ *is a planning task structure,* s_I *is the initial state, and the partial state* G *is the goal. A state consistent with* G *is a goal state. Finally*

- $\mathcal{C}_0$ *is the primary cost function, and*
- $\mathscr{C} = \{\langle \mathcal{C}_i, \mathsf{B}_i\rangle \mid 1 \leq i \leq n\}$ *is a set of* secondary cost functions *over the set of operators* O*, and* bounds*, both non-negative.*

An operator sequence π is a *plan* for Π_{MCF} if G is consistent with $s_I[\![\pi]\!]$ and $\sum_{o\in\pi} \mathcal{C}_i(o) \leq \mathsf{B}_i$ for $1 \leq i \leq n$. A plan is optimal if it has minimal *primary* cost among all plans for Π_{MCF}. A *heuristic* for an MCF planning task with states S is a mapping $h : S \times \mathbb{R}^{|\mathscr{C}|} \to \mathbb{R} \cup \{-\infty, \infty\}$. The *perfect heuristic* h^* maps a state s and a vector of bounds $\mathbf{b}$ to the primary cost $\mathcal{C}_0(\pi^*)$ of an optimal plan π^* for the MCF

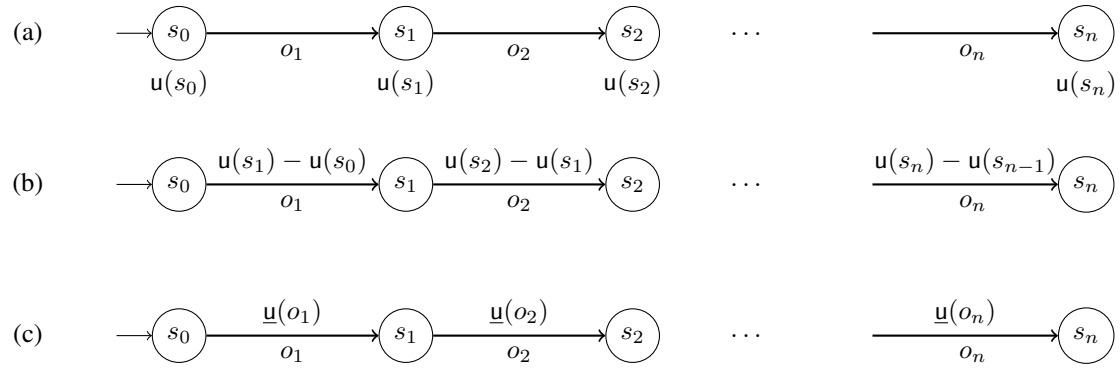


Figure 1: Reformulation of operator sequences with state dependent utility functions (a) into operator sequences where the cost function reflects the utility difference between two successive states (b). The additive utility function allows for a state-independent cost function (c).

planning task $\langle V, O, s, G, \mathcal{C}_0, \mathscr{C}' \rangle$, with $\mathscr{C}' = \{\langle \mathcal{C}_i, \mathbf{b}_i \rangle \mid \langle \mathcal{C}_i, \mathsf{B}_i \rangle \in \mathscr{C}\}$ or to ∞ if no such plan exists. A heuristic h is *admissible* if $h \leq h^*$.

A *classical planning task* is an MCF planning task $\Pi = \langle V, O, s_I, G, \mathcal{C}_0, \emptyset \rangle$ with $\mathcal{C}_0$ being non-negative. As the set of secondary cost functions only constrains the set of plans, every plan for an MCF task $\Pi_{\text{MCF}} = \langle V, O, s_I, G, \mathcal{C}_0, \mathscr{C} \rangle$ is also a plan for the classical planning task $\Pi = \langle V, O, s_I, G, \mathcal{C}_0, \emptyset \rangle$.

For this paper, we assume MCF tasks with at most one secondary cost function, i.e., having $|\mathscr{C}| \leq 1$.

Reformulation of OSP into MCF

We now show how to reformulate an OSP task into an MCF task. The key idea here is to compile the (additive) utility function into the primary cost function of an MCF planning task. This is achieved either through the straightforward application of the reformulation previously used in the net-benefit setting (Keyder and Geffner 2009), or through the novel *state-delta* reformulation that we propose here.

In the following, we use $\mathsf{u}_{max}(v)$ to denote the maximum utility over the values of the variable v, $\max_{\vartheta \in dom(v)} \mathsf{u}(\langle v, \vartheta \rangle)$, and M to denote the finite upper bound on the utility that can be obtained by a plan, given by

$$M := \sum_{v \in V} \mathsf{u}_{max}(v).$$

We note that the upper bound M allows us to switch from maximizing the original utility value u to *minimizing* the value $\underline{\mathsf{u}} : S \to \mathbb{R}^{0+}$ given by $\underline{\mathsf{u}}(s) = M - \mathsf{u}(s)$, and to restate the objective of the task as finding a plan π *minimizing* the value $M - \hat{\mathsf{u}}(\pi)$. The following two reformulations both implicitly make use of this observation.

Soft Goals Reformulation

The *soft goals* reformulation (Keyder and Geffner 2009) allows classical planning algorithms to be used to solve partial satisfaction problems by requiring valid plans to make explicit whether or not they achieve each soft goal in the problem. In the STRIPS setting, this is done by augmenting the goal with fluents that represent the fact that this choice has been made, and by introducing FORGO and COLLECT actions that achieve them. The FORGO actions have no preconditions but cost equal to the utility of the associated fluent, while the COLLECT actions have cost 0 but require the associated fact to have been achieved as a precondition. This reformulation has previously been adapted to the SAS$^+$ setting for net-benefit planning tasks (Katz and Mirkis 2016). Here, we present a slight variant for the oversubscription planning case.

Definition 2. *Let* $\Pi_{\text{OSP}} = \langle V, O, s_I, \mathcal{C}, \mathsf{u}, \mathsf{B} \rangle$ *be an oversubscription planning task. The* soft goals reformulation $\Pi^{sg}_{\text{MCF}} = \langle V', O', s_I, G', \mathcal{C}_0, \{\langle \mathcal{C}', \mathsf{B} \rangle\} \rangle$ *of* Π_{OSP} *is an* MCF planning task*, where*

- $V' = \{v' | v \in V\}$*, with* $dom(v') = dom(v) \cup \{v_g\}$*,*
- $O' = O \cup \{$*sg-action*$^{v,\vartheta} = \langle \{v = \vartheta\}, \{v = v_g\} \rangle \mid \vartheta \in dom(v), v \in V, \mathsf{u}_{max}(v) > 0\}$
- $G' = \{v_g | v \in V, \mathsf{u}_{max}(v) > 0\}$*,*
- $\mathcal{C}_0(o) = \begin{cases} 0 & \text{if } o \in O, \\ \mathsf{u}_{max}(v) - \mathsf{u}(\vartheta) & \text{if } o = \text{sg-action}^{v,\vartheta}, \end{cases}$
- $\mathcal{C}'(o) = \begin{cases} \mathcal{C}(o) & \text{if } o \in O, \\ 0 & \text{otherwise}. \end{cases}$

Note that since the additional actions have both precondition and effect defined on the same variable, this reformulation is also valid for the SAS formalism.

For a plan π for Π_{OSP}, by π^{sg} we denote a plan for its soft goals reformulation can be obtained by appending the operators *sg-action*$^{v,\vartheta}$ for $\langle v, \vartheta \rangle \in s_I[\![\pi]\!]$, in some predefined order.

Theorem 1. *Let* Π_{OSP} *be an oversubscription planning task and* Π^{sg}_{MCF} *its soft goals reformulation. If* π *is a plan for* Π_{OSP} *with utility* $\hat{\mathsf{u}}(\pi)$*, then* π^{sg} *is a plan for* Π^{sg}_{MCF} *with cost* $\mathcal{C}_0(\pi^{sg}) = M - \hat{\mathsf{u}}(\pi)$*. Conversely, if* π' *is a plan for* Π^{sg}_{MCF}*, removing the operators sg-action*$^{v,\vartheta}$ *results in a plan* π *for* Π_{OSP} *with utility* $\hat{\mathsf{u}}(\pi) = M - \mathcal{C}_0(\pi')$*.*

Proof. The proof follows from the proof of the equivalent theorem in the net-benefit planning setting (Keyder and

Geffner 2009), and the fact that both planning formalisms respect the bound specified by B. We omit the details due to lack of space. □

State-Delta Reformulation

The idea behind the state-delta reformulation, illustrated in Figure 1, is to compute by how much each operator changes the utility of a state when applied. In other words, for a state s and an operator o applicable in s, we compute the value $\underline{\mathsf{u}}(s, o) := \underline{\mathsf{u}}(s[\![o]\!]) - \underline{\mathsf{u}}(s)$.

Theorem 2. *The value $\underline{\mathsf{u}}(s, o)$ is independent of the state s.*

Proof. By definition of SAS, $\mathcal{V}(\mathsf{eff}(o)) \subseteq \mathcal{V}(\mathsf{pre}(o))$ for every operator $o \in O$. For a variable $v \in V \setminus \mathcal{V}(\mathsf{eff}(o))$, we have $s[v] = s[\![o]\!][v]$ and hence $\mathsf{u}(\langle v, s[v]\rangle) - \mathsf{u}(\langle v, s[\![o]\!][v]\rangle) = 0$. Therefore, it suffices to consider variables $v \in \mathcal{V}(\mathsf{eff}(o))$:

$$\begin{aligned}\underline{\mathsf{u}}(s, o) &= (M - \mathsf{u}(s[\![o]\!])) - (M - \mathsf{u}(s)) \\ &= \sum_{v \in V} \mathsf{u}(\langle v, s[v]\rangle) - \mathsf{u}(\langle v, s[\![o]\!][v]\rangle) \\ &= \sum_{v \in \mathcal{V}(\mathsf{eff}(o))} \mathsf{u}(\langle v, s[v]\rangle) - \mathsf{u}(\langle v, s[\![o]\!][v]\rangle) \\ &= \sum_{v \in \mathcal{V}(\mathsf{eff}(o))} \mathsf{u}(\langle v, \mathsf{pre}(o)[v]\rangle) - \mathsf{u}(\langle v, \mathsf{eff}(o)[v]\rangle).\end{aligned}$$

□

Thus, we can define a (state-independent) cost function over operators $\underline{\mathsf{u}} : O \to \mathbb{R}$ as

$$\underline{\mathsf{u}}(o) = \sum_{v \in \mathcal{V}(\mathsf{eff}(o))} \mathsf{u}(\langle v, \mathsf{pre}(o)[v]\rangle) - \mathsf{u}(\langle v, \mathsf{eff}(o)[v]\rangle).$$

When the operator changes the value of a variable to one with lower utility, $\underline{\mathsf{u}}(o)$ will be negative. Note that Theorem 2 requires that the task be in SAS rather than SAS$^+$ form, as this ensures that the value of each variable changed by the operator is uniquely specified in the precondition. This property is required in order to compute the utility change resulting from the operator independently of the state in which it is applied.

Theorem 3. *For a sequence of operators π applicable in state s, $\underline{\mathsf{u}}(s) + \sum_{o \in \pi} \underline{\mathsf{u}}(o) = \underline{\mathsf{u}}(s[\![\pi]\!])$.*

The proof is straightforward from the definition of $\underline{\mathsf{u}}$ on operators. Thus, finding a sequence of operators leading to a state with the minimal value $\underline{\mathsf{u}}$ corresponds exactly to finding a sequence of operators of a minimal summed cost $\underline{\mathsf{u}}$. We can thus solve the OSP task as a classical planning problem with multiple cost functions and an empty goal:

Definition 3. *Let $\Pi_{\text{OSP}} = \langle V, O, s_I, \mathcal{C}, \mathsf{u}, \mathsf{B}\rangle$ be an oversubscription planning task. The* state-delta reformulation *$\Pi^{sd}_{\text{MCF}} = \langle V, O, s_I, G, \mathcal{C}_0, \{\langle \mathcal{C}, \mathsf{B}\rangle\}\rangle$ of Π_{OSP} is the* MCF planning task*, where*

- *$G = \emptyset$, and*
- *$\mathcal{C}_0(o) = \sum_{v \in \mathcal{V}(\mathsf{eff}(o))} \mathsf{u}(\mathsf{pre}(o)[v]) - \mathsf{u}(\mathsf{eff}(o)[v])$, for $o \in O$.*

Theorem 4. *Let Π_{OSP} be an oversubscription planning task and Π^{sd}_{MCF} its state-delta reformulation. If π is a plan of Π_{OSP} with utility $\hat{\mathsf{u}}(\pi)$ then π is a plan of Π^{sd}_{MCF} with cost $\mathcal{C}_0(\pi) = \mathsf{u}(s_I) - \hat{\mathsf{u}}(\pi)$ and vice versa.*

Proof. Operator applicability is defined in the same way for Π_{OSP} and Π^{sd}_{MCF}, so if π is a plan in one task, it is applicable in the other and results in the same state.

The operator sequence π respects the bounds of Π_{OSP} iff $\sum_{o \in \pi} \mathcal{C}(o) \leq \mathsf{B}$ iff π respects the bounds of the (only) secondary cost function of Π^{sd}_{MCF}. Therefore, and because all states are goal states in Π^{sd}_{MCF}, π is a plan in Π_{OSP} iff it is a plan in Π^{sd}_{MCF}.

The primary cost of π is $\mathcal{C}_0(\pi) = \sum_{o \in \pi} \underline{\mathsf{u}}(o)$, which is equal to $\underline{\mathsf{u}}(s_I[\![\pi]\!]) - \underline{\mathsf{u}}(s_I) = \mathsf{u}(s_I) - \hat{\mathsf{u}}(\pi)$ according to Theorem 3. □

As $\mathsf{u}(s_I)$ is constant, a plan π maximizes $\hat{\mathsf{u}}(\pi)$ iff it minimizes $\mathcal{C}_0(\pi)$.

Discussion

The next corollary follows directly from Theorems 1 and 4:

Corollary 1. *An oversubscription planning task, and its state-delta and soft goals reformulations have the same optimal plans (modulo the removal of sg-action$^{v,\vartheta}$ operators in the case of the soft goals reformulation).*

Corollary 1 implies that the original oversubscription problem can be solved optimally by finding an optimal solution to either of the MCF planning tasks presented above.

The use of each the two reformulations we have presented has potential advantages compared to the other. Though both reformulations require a planning algorithm that is able to take into account the bounds specified on the secondary cost functions, the state-delta reformulation also requires planning algorithms that can handle negative action costs, which may be computationally difficult. However, it does not add any additional operators, variables, or values to variable domains.

On the other hand, the soft goals reformulation does not require support for negative costs, but the addition of new operators and variables results in more information to reason about and a larger state space. While this effect can be curtailed by requiring that the goal-achieving actions are applied in a fixed order, this growth can still prove difficult to handle for some approaches.

Heuristics for OSP via Reformulation

Having proposed two different reformulations of OSP as MCF planning, we now show how to devise heuristics for OSP from heuristics for MCF planning.

Soft Goals Reformulation

For the *soft goals* reformulation, admissible classical planning heuristics compute an underestimate of the difference between the utility upper bound M and the utility obtained by an optimal plan from a state s. An overestimate of the utility that can be obtained from the state is therefore given by $M - h_{\text{MCF}}(s, \mathsf{b}_s)$:

Definition 4. *Let* Π_{OSP} *be an OSP task and* Π_{MCF} *its* soft goals *reformulation. Let* $h_{\text{MCF}} : S \times \mathbb{R} \to \mathbb{R} \cup \{-\infty, \infty\}$ *be some heuristic for* Π_{MCF}*. The* soft goals reformulation heuristic *of* h_{MCF}*, is denoted by* h^{sg}_{MCF} *and has the value* $h^{sg}_{\text{MCF}}(s, \mathsf{b}_s) = M - h_{\text{MCF}}(s, \mathsf{b}_s)$ *on every state* s *of* Π_{OSP}*.*

The following lemma establishes the connection between the informativeness of heuristics for OSP tasks and heuristics for their soft goals reformulations.

Lemma 1. *For an OSP task* Π_{OSP} *and its soft goals reformulation* $\Pi_{\text{MCF}} = \Pi^{sg}_{\text{MCF}}$*,* $h^*_{\Pi_{\text{OSP}}} = (h^*_{\Pi_{\text{MCF}}})^{sg}$*.*

The lemma is a direct outcome from Theorem 1. We use it in order to show the following main result.

Theorem 5. *Let* $\Pi_{\text{OSP}} = \langle V, O, s_I, \mathcal{C}, \mathsf{u}, \mathsf{B}\rangle$ *be an OSP task,* Π_{MCF} *its soft goals multiple cost function reformulation, and* h_{MCF} *an admissible heuristic for* Π_{MCF}*. Then* h^{sg}_{MCF} *is an admissible heuristic for* Π_{OSP}*.*

Proof. Let h^*_{MCF} be the perfect heuristic for Π_{MCF} and h^*_{OSP} the perfect heuristic for Π_{OSP}. With Definition 4, we can rewrite $h^*_{\text{MCF}}(s, \mathsf{b}_s)$ as $M - (h^*_{\text{MCF}})^{sg}(s, \mathsf{b}_s)$, which is $M - h^*_{\text{OSP}}(s, \mathsf{b}_s)$ according to Lemma 1.

From Definition 4 we have

$$h^{sg}_{\text{MCF}}(s, \mathsf{b}_s) = M - h_{\text{MCF}}(s, \mathsf{b}_s),$$

and from admissibility of h_{MCF} we have

$$h_{\text{MCF}}(s, \mathsf{b}_s) \leq h^*_{\text{MCF}}(s, \mathsf{b}_s),$$

so

$$\begin{aligned} h^{sg}_{\text{MCF}}(s, \mathsf{b}_s) &\geq M - h^*_{\text{MCF}}(s, \mathsf{b}_s) \\ &= M - (M - h^*_{\text{OSP}}(s, \mathsf{b}_s)) \\ &= h^*_{\text{OSP}}(s, \mathsf{b}_s). \end{aligned}$$

□

State-Delta Reformulation

For the *state-delta* reformulation, admissible heuristics underapproximate the *difference* between the utility of the state $u(s)$ and the best utility that can be obtained $\hat{\mathsf{u}}(\pi^*)$. Thus, the difference between the utility of a state and the heuristic value is an overapproximation for the best utility that can be obtained:

Definition 5. *Let* Π_{OSP} *be an OSP task and* Π_{MCF} *its* state-delta *reformulation. Let* $h_{\text{MCF}} : S \times \mathbb{R} \to \mathbb{R} \cup \{-\infty, \infty\}$ *be some heuristic for* Π_{MCF}*. The* state-delta reformulation heuristic *of* h_{MCF}*, is denoted by* h^{sd}_{MCF} *and has the value* $h^{sd}_{\text{MCF}}(s, \mathsf{b}_s) = \mathsf{u}(s) - h_{\text{MCF}}(s, \mathsf{b}_s)$ *on every state* s *of* Π_{OSP}*.*

The following lemma establishes the connection between the informativeness of heuristics for OSP tasks and heuristics for their state-delta reformulations.

Lemma 2. *For an OSP task* Π_{OSP} *and its state-delta reformulation* $\Pi_{\text{MCF}} = \Pi^{sd}_{\text{MCF}}$*,* $h^*_{\Pi_{\text{OSP}}} = (h^*_{\Pi_{\text{MCF}}})^{sd}$*.*

The lemma is a direct outcome from Theorem 4. We use it in order to show the following main result.

Theorem 6. *Let* $\Pi_{\text{OSP}} = \langle V, O, s_I, \mathcal{C}, \mathsf{u}, \mathsf{B}\rangle$ *be an OSP task,* Π_{MCF} *its state-delta multiple cost function reformulation, and* h_{MCF} *an admissible heuristic for* Π_{MCF}*. Then* h^{sd}_{MCF} *is an admissible heuristic for* Π_{OSP}*.*

The proof is almost identical to the proof of Theorem 5.

Heuristics for MCF Planning

Having established how to exploit admissible heuristics for MCF planning tasks to derive admissible estimates for OSP tasks, we now turn our attention to obtaining heuristic estimates for MCF planning tasks.

Admissible Heuristics for MCF Planning

Let Π_{MCF} be an MCF planning task, π be some plan of Π_{MCF} and s be some state along π. Let π_s denote the suffix of π that starts with s. Observe that since the secondary cost functions are non-negative, B_i is an upper bound on the cost of π_s according to the cost function $\mathcal{C}_i$. Thus, the cost of an optimal plan for $\Pi_{\text{MCF}}(s) = \langle V, O, s, G, \mathcal{C}_0, \mathscr{C}\rangle$, is an admissible estimate for the state s. Further, we can increase the bounds arbitrarily without forfeiting admissibility. Thus, in particular, an admissible estimate for the classical planning task $\Pi = \langle V, O, s, G, \mathcal{C}_0\rangle$ is an admissible estimate for the state s of the MCF planning task as well.

Abstraction Heuristics for MCF Planning

In classical planning, abstractions can be obtained by e.g. projecting the problem on to a subset of its variables (Edelkamp 2001), or through a merge-and-shrink process (Helmert, Haslum, and Hoffmann 2007; Helmert et al. 2014). One of the strengths of abstraction heuristics in classical planning is their low per-node computation time during search. For explicit abstractions, such as projections and merge-and-shrink, the computation is basically a linear-time lookup. For implicit abstractions (Katz and Domshlak 2010a), the computation is more complicated, but is still of low polynomial time.

Abstractions for MCF planning generalize the definition for classical planning (Helmert, Haslum, and Hoffmann 2007) by additionally requiring reachable abstract state distances under the secondary cost functions to be below their respective bounds. Formally, a *(labeled) transition system* (with multiple cost functions) is a tuple $\Theta = \langle S, L, \mathbf{c}, T, s_0, S_*\rangle$ where S is a finite set of states, L is a finite set of labels, $\mathbf{c} = \langle c_0, \cdots, c_n\rangle$ are functions $c_i : L \to \mathbb{R}$ $(1 \leq i \leq n)$, $T \subseteq S \times L \times S$ a set of labeled transitions, s_0 the initial state and S_* the goal states.

The induced transition system of an MCF task $\Pi_{\text{MCF}} = \langle V, O, s_I, G, \mathcal{C}_0, \mathscr{C}\rangle$ is the transition system $\Theta_{\Pi_{\text{MCF}}} = \langle S', L', \mathbf{c}', T', s'_0, S'_*\rangle$ where S' are the states of Π_{MCF}, $L' = O$, $\mathbf{c}_i(o) = \mathcal{C}_i(o)$, $(s, o, t) \in T'$ iff s is consistent with $\mathsf{pre}(o)$ and $t = s[\![o]\!]$, s'_0 is the initial state of the planning task and S'_* are the goal states of the planning task. An abstraction is a mapping $\alpha : S' \to S^\alpha$ where S^α are the states of the transition system $\Theta^\alpha = \langle S^\alpha, L, \mathbf{c}, T^\alpha, s^\alpha_0, S^\alpha_*\rangle$ with $T^\alpha = \{\langle \alpha(s), o, \alpha(t)\rangle \mid (s, o, t) \in T\}$, $s^\alpha_0 = \alpha(s_0)$ and $S^\alpha_* = \{\alpha(s) \mid s \in S\}$. Θ^α is called the abstract transition system.

We proceed now with an example of how to derive merge-and-shrink heuristics for MCF planning tasks with (possibly) negative primary cost functions. We start by introducing a generic scheme for abstraction heuristics.

Definition 6. *Let* Π_{MCF} *be an MCF task,* α *be an abstraction and* Θ^α *its abstract transition system. The heuristic*

$h^\alpha_\Theta : (S \times \mathbb{R}^n) \to \mathbb{R} \cup \{-\infty, \infty\}$ *is the* MCF planning abstraction heuristic *of* Π_{MCF} *if it maps a state* $s \in S$ *and bounds* $\mathsf{b}_1, \ldots \mathsf{b}_n$ *to the cost of a path* ρ *in the abstract transition system* Θ^α*, such that*

- *for all* $1 \leq i \leq n$, $\mathcal{C}_i(\rho) \leq \mathsf{b}_i$, *and*
- ρ *is cost-minimal among such paths according to the primary cost* $\mathcal{C}_0$.

If no such path to an abstract goal state exists, the heuristic value is ∞*. Otherwise, if there exists such a path that contains a cycle of a negative total cost under* $\mathcal{C}_0$*, then the heuristic value is* $-\infty$.

For an MCF planning task with one secondary cost function, an abstraction heuristic $h^\alpha_\Theta(s, \mathsf{b})$ can be computed using the following scheme:

(I) Construct abstract transition system Θ^α,

(II) Compute shortest path distances from $\alpha(s_I)$ to all abstract states in Θ^α according to the secondary cost function $\mathcal{C}_1$ and discard abstract states with abstract distances strictly larger than b, and

(III) Compute shortest path distances from all remaining abstract states to some abstract goal state, according to the primary cost function $\mathcal{C}_0$.

There are essentially two challenges in turning this scheme into an abstraction heuristic. First, since the primary cost function is potentially negative, there might be reachable cycles of total negative cost in Θ^α resulting in an uninformative heuristic. The concrete choice of methods for constructing Θ^α in step (I) should aim to prevent, or at least alleviate, this problem. In this work, we use existing methods for constructing merge-and-shrink abstractions (Sievers, Wehrle, and Helmert 2014), leaving techniques for constructing abstractions that avoid negative cost cycles as future work.

The second challenge lies in the runtime complexity of heuristic computation. The reachable abstract states in step (II) depend on the budget b, and for maximal informativeness, step (III) should be performed at every evaluated state, taking into account the reachability of abstract states considering the remaining budget b at that search node. However, this opens up the possibility that a state may be rediscovered during the search with a higher remaining budget, which must be taken into account when considering whether the behavior of the search algorithm remains optimal or not. Additionally, the possibly negative cost function mandates the use of a shortest path algorithm that supports negative weights. Such algorithms are computationally more expensive than the typically used shortest path algorithms for non-negative weights. We alleviate this problem by performing the computation in step (III) only once, for reachability defined under the initial budget b_0.

Experimental Evaluation

We first discuss the creation of a set of reference OSP problems to evaluate the performance of our proposed techniques.

Creating a Benchmark Set for OSP

Since there is currently no publicly available benchmark set for oversubscription planning, we have created one. We followed a similar procedure to that described by Domshlak and Mirkis (2015), based on the collection of classical International Planning Competition (IPC) domains. In contrast to the previous approach, we consider all planning tasks for which any solution is known, not only a provably optimal one. Such upper bounds on solution costs can be obtained from planning.domains (Muise 2016), a repository of planning benchmarks to which researchers are contributing meta-data on solved planning problems. We set the bounds for oversubscription planning tasks to be either 25, 50, 75, or 100% of the best known solution cost for the classical planning task, resulting in four versions of each classical planning domain, which we refer to as *suites*. The problems are generated by assigning each goal in the original problem a utility of 10, and assigning to a randomly chosen 5% of the facts in the problem a uniformly distributed integer utility in the range $[1, 5]$. All other facts are assigned a utility of 0.

To accommodate cost bounds and utilities, we extended PDDL with two additional sections in the problem file. The first section (:BOUND contains the bound on the solution cost, while the second section (:UTILITY contains a collection of function assignments of numeric values to grounded predicates, e.g., *(= (ON C B) 1)*. To translate the PDDL instances to a multi-valued formalism, we adapted the translator of the Fast Downward planning system to handle this syntax. The PDDL domain collection and the adapted translator are available for download (Katz et al. 2019; Winterer 2019).

Transforming SAS$^+$ to SAS

Fast Downward translates PDDL into SAS$^+$ representation, which is more compact than SAS. While the *soft goals* reformulation can work directly on the SAS$^+$ representation, the *state-delta* reformulation requires tasks to be in the SAS format. To achieve that, we need to modify operators with preconditions not specified for some effect variables. We used a procedure similar to the transition normalization (Pommerening and Helmert 2015) for this purpose. As the transition normalization increases the state space exponentially, we propose an optimization to moderate that increase. Note that our reformulation restricts the preconditions to be specified on effect variables only for variables with specified utility on at least one value. Thus, we do not modify the variables whose values do not have utilities specified.

Comparison to a Baseline

In our experiments, we compare different heuristics within a best-first branch-and-bound (BFBB) search, which we implemented in Fast Downward planning system. BFBB uses two heuristic functions. One is for choosing the next node to expand (guidance heuristic), and another one for pruning the nodes (pruning heuristic). We compare the following configurations differing in their pruning heuristic:

Bl Blind heuristic $h_{Bl}(s, \mathsf{b}) = M$

	25%				50%				75%				100%			
Coverage	Bl	M&S^{sd}	M&S^{sg}	LMC	Bl	M&S^{sd}	M&S^{sg}	LMC	Bl	M&S^{sd}	M&S^{sg}	LMC	Bl	M&S^{sd}	M&S^{sg}	LMC
airport	**27**	18	18	25	**22**	18	18	21	**21**	18	18	19	**21**	18	18	18
depot	**16**	**16**	**16**	15	**11**	**11**	**11**	**11**	**7**	**7**	**7**	**7**	**4**	**4**	**4**	**4**
elevators08	**30**	**30**	**30**	29	**25**	**25**	**25**	24	**23**	22	22	22	**17**	**17**	**17**	**17**
elevators11	**20**	**20**	**20**	**20**	**19**	**19**	**19**	**19**	**18**	17	17	17	**14**	**14**	**14**	**14**
freecell	**77**	**77**	**77**	58	**30**	**30**	**30**	23	**21**	**21**	**21**	15	**20**	**20**	**20**	14
grid	**5**	**5**	**5**	4	**3**	**3**	**3**	**3**	**2**	**2**	**2**	**2**	**1**	**1**	**1**	**1**
hiking14	**19**	**19**	**19**	14	**14**	**14**	**14**	12	**13**	**13**	**13**	11	**11**	**11**	**11**	10
mprime	**35**	**35**	**35**	**35**	**28**	**28**	**28**	27	**24**	**24**	**24**	**24**	**19**	**19**	**19**	**19**
mystery	**29**	**29**	27	**29**	**27**	**27**	26	26	**21**	**21**	20	18	**18**	**18**	17	**18**
nomystery11	**20**	**20**	**20**	**20**	**14**	**14**	**14**	**14**	**10**	**10**	**10**	9	**8**	**8**	**8**	**8**
openstacks14	**20**	**20**	**20**	19	**15**	**15**	**15**	13	**7**	**7**	**7**	5	**3**	**3**	**3**	**3**
parcprinter08	**17**	**17**	**17**	15	**13**	**13**	**13**	12	**11**	**11**	**11**	**11**	**11**	10	**11**	**11**
parcprinter11	**13**	**13**	**13**	11	**9**	**9**	**9**	**9**	**7**	**7**	**7**	**7**	6	6	**7**	**7**
parking11	**11**	**11**	**11**	10	**1**	**1**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
parking14	**14**	**14**	**14**	11	**4**	**4**	**4**	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
pipes-notank	**45**	18	18	**45**	**30**	18	18	29	**22**	17	17	20	**15**	14	14	14
pipes-tank	**35**	28	28	31	**20**	19	19	17	**16**	15	15	14	**11**	**11**	**11**	10
rovers	**15**	14	14	14	**8**	**8**	**8**	**8**	**6**	**6**	**6**	**6**	**5**	**5**	**5**	**5**
scanalyzer08	**13**	**13**	**13**	**13**	**12**	**12**	**12**	**12**	**12**	**12**	**12**	9	**12**	**12**	**12**	9
scanalyzer11	**10**	**10**	**10**	**10**	**9**	**9**	**9**	**9**	**9**	**9**	**9**	6	**9**	**9**	**9**	6
sokoban08	**30**	**30**	**30**	**30**	**29**	**29**	**29**	28	**24**	**24**	**24**	**24**	**22**	**22**	**22**	21
sokoban11	**20**	**20**	**20**	**20**	**20**	**20**	**20**	**20**	**20**	**20**	**20**	**20**	**19**	**19**	**19**	18
tetris14	**17**	2	2	**17**	**14**	2	1	12	**11**	2	2	9	**9**	1	2	6
tidybot11	**20**	1	1	**20**	**20**	1	1	19	**18**	1	1	13	**13**	1	1	7
tidybot14	**20**	0	0	**20**	**18**	0	0	16	**14**	0	0	7	**6**	0	0	0
transport08	**17**	**17**	**17**	15	**15**	**15**	**15**	14	**12**	11	11	11	**11**	**11**	**11**	**11**
transport11	**15**	**15**	**15**	**15**	**11**	**11**	**11**	**11**	**8**	7	7	7	**6**	**6**	**6**	**6**
transport14	**13**	**13**	**13**	12	**9**	**9**	**9**	**9**	**9**	**9**	**9**	8	**7**	**7**	**7**	6
trucks	**13**	**13**	**13**	11	**8**	**8**	**8**	6	**6**	**6**	**6**	5	**5**	**5**	**5**	4
woodwork08	**25**	**25**	**25**	24	**15**	**15**	**15**	14	**10**	**10**	**10**	**10**	7	7	**9**	7
woodwork11	**18**	**18**	**18**	16	**10**	**10**	**10**	8	**5**	**5**	**5**	4	**2**	**2**	**2**	**2**
Sum equal	511	511	511	511	414	414	414	414	361	361	361	361	339	339	339	339
Sum all	1190	1092	1090	1139	897	831	829	862	748	695	694	701	651	620	624	615

Table 1: Per-domain coverage comparison of the blind heuristic (Bl), two merge-and-shrink configurations (M&S^{sd} and M&S^{sg}), and the LM-cut heuristic (LMC) for the four domain suites.

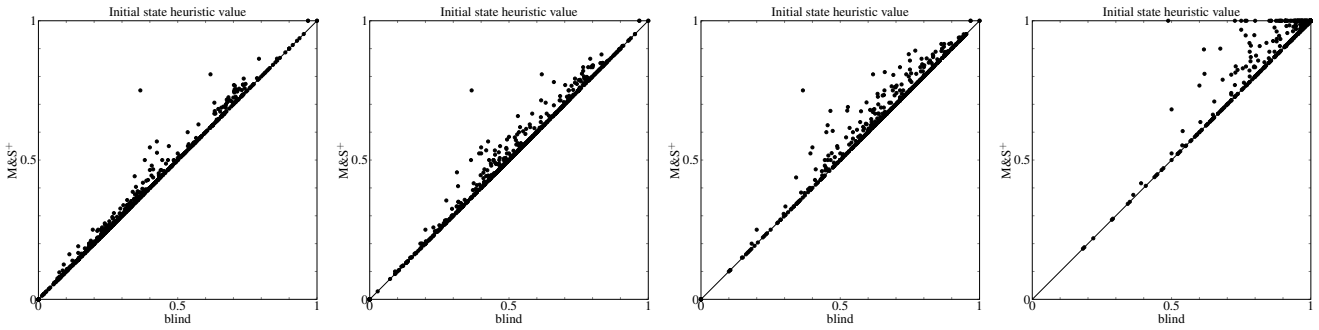
M&S^{sd} A merge and shrink approach to compute $h^{\alpha}_{\Theta}(s, \mathbf{b})$ for the *state-delta* reformulation. For step (I) we used the bisimulation based shrinking, and as merge strategy SCC-DFP (Sievers, Wehrle, and Helmert 2014) according to secondary cost function $\mathcal{C}_1$. For step (III) we used the Bellman-Ford algorithm (Shimbel 1954) to compute (possibly negative) shortest path distances. For better runtime complexity, we do step (III) only once, with fixed budget B_0. The heuristic $h^{\alpha}_{\Theta}(s, \mathbf{b})$ is reformulated into an OSP heuristic according to Definition 5.

M&S^{sg} A merge and shrink approach to compute $h^{\alpha}_{\Theta}(s, \mathbf{b})$ for the *soft goals* reformulation. Here as well, for step (I) we used the bisimulation based shrinking, and as merge strategy SCC-DFP (Sievers, Wehrle, and Helmert 2014) according to secondary cost function $\mathcal{C}_1$. For step (III) we used the Dijkstra algorithm. For better runtime complexity, we do step (III) only once, with fixed budget B_0.

LM-cut An LM-cut heuristic for the *soft goals* reformulation, ignoring the secondary cost function.

For a fair comparison, we set the guidance heuristic in all our approaches to the blind heuristic. Unfortunately, we could not compare to the recent work of Muller and Karpas (2018), as the code was not available from the authors at the time of this paper submission. To compare to previous state-of-the-art approaches to OSP, much work is still needed to adapt these techniques to work in an out-of-the-box fashion. For instance, the planner of Mirkis and Domshlak (2013) requires a specification of variable patterns to be used in their PDB heuristic. Similarly, the approach described in Mirkis and Domshlak (2014) also did not work out-of-the-box, since it is based on the previous one. However, the performance of these approaches is not too far from the simple blind heuristic, always returning the maximal utility, and therefore we use the blind heuristic as our baseline.

The experiments were performed on Intel(R) Xeon(R) CPU E7-8837 @ 2.67GHz machines, with the time and memory limit of 30min and 2GB, respectively.



(a) (b) (c) (d)

Figure 2: Comparison of the heuristic value in the initial state with the blind and the merge-and-shrink heuristics for different problem suites, (a) 25%, (b) 50%, (c) 75%, and (d) 100%.

Results

Table 1 shows the number of problems solved optimally for each domain, comparing our approach to the baseline. On many domains, the performance of both approaches in terms of coverage is the same, for all suites. These rows are not shown in the table and are instead summarized in the "Sum equal" row. Overall, the baseline almost always achieves higher coverage, with the difference being smaller for problem suites with higher cost bounds. Compared to the best-performing non-baseline planner, blind BNB solves 51, 35, 47, and 27 more problems on the 25, 50, 75, and 100% suites respectively. Heuristics beat blind search in only two cases: in the PARC-PRINTER11 and WOODWORKING08 instances of the 100% suite, by at most 1 and 2 problems, respectively. We note that in the domains AIRPORT, TETRIS, TIDYBOT11, TIDYBOT14, PIPESWORLD-NOTANKAGE, and PIPESWORLD-TANKAGE the construction of the merge-and-shrink abstraction often does not terminate within the time bound.

In order to look beyond coverage and measure heuristic informativeness, we also considered the heuristic estimates computed in the initial states of our problems. Figure 2 depicts the results of this comparison for the soft goals reformulation: each point corresponds to a single instance, with the x-axis given by h^*/M, or the utility of the optimal plan divided by the value of the blind heuristic, and the y-axis given by $h^*/\text{M\&S}^{sg}$. We observe that as the bound increases (moving to the right), points move toward the upper right of the graph, as both $\text{M\&S}^{sg}$ and the blind estimate become closer to the actually achievable utility. As expected, $\text{M\&S}^{sg}$ dominates M as a heuristic estimate, and all points lie either on or above the diagonal, but there are no obvious patterns regarding how the relative informativeness of $\text{M\&S}^{sg}$ and the blind heuristic varies with the bound.

Conclusions and Future Work

We have introduced two reformulations of the oversubscription planning task into equivalent classical planning tasks with two cost functions, allowing existing heuristics for classical planning to be adapted to the oversubscription planning setting. We have shown how to adapt the merge-and-shrink and LM-cut heuristics to this problem. Our experimental evaluation shows the feasibility of such an approach. In the absence of a standard benchmark set and PDDL constructs for describing oversubscription planning tasks, we have introduced the required syntax and created a benchmark set for our evaluation, making it available to the planning community. By adapting the Fast Downward planning framework, in which many classical planning heuristics are already implemented, to oversubscription planning, we have simplified the future use of classical planning heuristics for these tasks via the suggested reformulations.

In future work we intend to investigate further heuristic adaptations, including bound-aware heuristics that are able to make use of the remaining budget at each search node to obtain more informed heuristic estimates, as well as the use of these heuristics in other search schemes that are closer in spirit to the A* algorithm. One promising direction is further investigation of the connections between the two reformulations. Another possible subject of interest is the interaction between the reformulation and heuristic additivity criteria, such as action cost partitioning (Katz and Domshlak 2008; 2010b) or disjointness for pattern databases (Haslum et al. 2007). We would also like to integrate and automate the approach of Mirkis and Domshlak (2013; 2014) and explore connections between their approach and our reformulations. In addition, we would like to explore techniques for node ordering in branch-and-bound search. Finally, we would like to adapt existing search pruning techniques for classical planning (Domshlak, Katz, and Shleyfman 2012; Alkhazraji et al. 2012) to branch-and-bound search for oversubscription planning tasks.

Acknowledgments

We thank Vitaly Mirkis for inspiration and his extensive contributions during the early stages of this research. Florian Pommerening received funding for this work from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no. 817639).

References

Alkhazraji, Y.; Wehrle, M.; Mattmüller, R.; and Helmert, M. 2012. A stubborn set algorithm for optimal planning. In *Proc. ECAI 2012*, 891–892.

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

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

Bonet, B., and Geffner, H. 2001. Planning as heuristic search. *AIJ* 129(1):5–33.

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

Domshlak, C., and Mirkis, V. 2015. Deterministic oversubscription planning as heuristic search: Abstractions and reformulations. *JAIR* 52:97–169.

Domshlak, C.; Hoffmann, J.; and Katz, M. 2015. Red-black planning: A new systematic approach to partial delete relaxation. *AIJ* 221:73–114.

Domshlak, C.; Katz, M.; and Shleyfman, A. 2012. Enhanced symmetry breaking in cost-optimal planning as forward search. In *Proc. ICAPS 2012*, 343–347.

Edelkamp, S. 2001. Planning with pattern databases. In *Proc. ECP 2001*, 84–90.

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

Haslum, P.; Botea, A.; Helmert, M.; Bonet, B.; and Koenig, S. 2007. Domain-independent construction of pattern database heuristics for cost-optimal planning. In *Proc. AAAI 2007*, 1007–1012.

Helmert, M., and Domshlak, C. 2009. Landmarks, critical paths and abstractions: What's the difference anyway? In *Proc. ICAPS 2009*, 162–169.

Helmert, M.; Haslum, P.; Hoffmann, J.; and Nissim, R. 2014. Merge-and-shrink abstraction: A method for generating lower bounds in factored state spaces. *JACM* 61(3):16:1–63.

Helmert, M.; Haslum, P.; and Hoffmann, J. 2007. Flexible abstraction heuristics for optimal sequential planning. In *Proc. ICAPS 2007*, 176–183.

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

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

Karpas, E., and Domshlak, C. 2009. Cost-optimal planning with landmarks. In *Proc. IJCAI 2009*, 1728–1733.

Katz, M., and Domshlak, C. 2008. Optimal additive composition of abstraction-based admissible heuristics. In *Proc. ICAPS 2008*, 174–181.

Katz, M., and Domshlak, C. 2010a. Implicit abstraction heuristics. *JAIR* 39:51–126.

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

Katz, M., and Mirkis, V. 2016. In search of tractability for partial satisfaction planning. In *Proc. IJCAI 2016*, 3154–3160.

Katz, M.; Keyder, E.; Pommerening, F.; and Winterer, D. 2019. PDDL benchmarks for oversubscription planning. https://doi.org/10.5281/zenodo.2576024.

Keyder, E., and Geffner, H. 2008. Heuristics for planning with action costs revisited. In *Proc. ECAI 2008*, 588–592.

Keyder, E., and Geffner, H. 2009. Soft goals can be compiled away. *JAIR* 36:547–556.

Keyder, E.; Richter, S.; and Helmert, M. 2010. Sound and complete landmarks for and/or graphs. In *Proc. ECAI 2010*, 335–340.

Mirkis, V., and Domshlak, C. 2013. Abstractions for oversubscription planning. In *Proc. ICAPS 2013*, 153–161.

Mirkis, V., and Domshlak, C. 2014. Landmarks in oversubscription planning. In *Proc. ECAI 2014*, 633–638.

Muise, C. 2016. Planning.Domains. In *26th International Conference on Automated Planning and Scheduling, System Demonstrations and Exhibits*.

Muller, D., and Karpas, E. 2018. Value driven landmarks for oversubscription planning. In *Proc. ICAPS 2018*, 171–179.

Pommerening, F., and Helmert, M. 2015. A normal form for classical planning tasks. In *Proc. ICAPS 2015*, 188–192.

Richter, S.; Helmert, M.; and Westphal, M. 2008. Landmarks revisited. In *Proc. AAAI 2008*, 975–982.

Shimbel, A. 1954. Structure in communication nets. *Proceedings of the symposium on information networks* 4.

Sievers, S.; Wehrle, M.; and Helmert, M. 2014. Generalized label reduction for merge-and-shrink heuristics. In *Proc. AAAI 2014*, 2358–2366.

Smith, D. E. 2004. Choosing objectives in over-subscription planning. In *Proc. ICAPS 2004*, 393–401.

van den Briel, M.; Sanchez, R.; Do, M. B.; and Kambhampati, S. 2004. Effective approaches for partial satisfaction (over-subscription) planning. In *Proc. AAAI 2004*, 562–569.

Winterer, D. 2019. Fast downward translator component for oversubscription planning. https://doi.org/10.5281/zenodo.2628252.