Reformulating Oversubscription Planning Tasks

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

Vitaly Mirkis
Amazon Research
Haifa, Israel*
vitamin@amazon.com

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

Dominik Winterer
Unaffiliated
Germany†
dominik_winterer@gmx.de

Abstract

Most modern heuristics for classical planning are specified in terms of minimizing the summed operator costs. Heuristics for oversubscription planning (OSP), on the other hand, maximize the utility on states. In this work we aim to provide the grounds for the adaptation of existing heuristics for classical planning to the OSP setting. To this end, we reformulate the OSP task to a classical planning task extended with an additional operator costs function, reflecting the utility information fully. We exemplify how existing heuristics from classical planning can be adapted to such a setting with a merge-and-shrink heuristic and empirically validate the feasibility of our approach.

Introduction

The field of automated planning is concerned with the problem of finding a course of action satisfying certain predefined goals. While the classical planning problem requires achieving all goals, partial satisfaction planning relaxes this restriction, allowing to achieve a subset of the goals. As a result, even an empty plan is a trivial valid solution, and therefore the aim of partial satisfaction planning is to obtain solutions of best possible quality. In *net-benefit* planning (van den Briel *et al.* 2004), a subfield of partial satisfaction planning, the assumption is that the solution cost and state values are comparable. As a consequence, the solution quality is measured as the net difference between the value of the obtained end state and the solution cost. In *oversubscription planning* (Smith 2004), on the other hand, the solution cost and state values are assumed to be incomparable. Thus, to take the cost into account, a bound on the cost or a *budget* is introduced (Smith 2004), and the objective is to maximize the value of the obtained end state, while constraining the solution cost.

Heuristic search is among the best performing approaches to both classical and net-benefit planning, with many search guiding heuristics developed over the years. These heuristics 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 *et al.* 2015), *critical paths* (Haslum and Geffner 2000), and *landmarks*, (e.g., Richter *et al.* 2008; Karpas and Domshlak 2009; Helmert and Domshlak 2009; Keyder *et al.* 2010). The basic principle behind all these heuristics is the same – relaxing the task at hand to fit some tractable fragment of the planning problem. In net-benefit planning, these heuristics are often applied not directly to the net-benefit task, but to a reformulation into classical planning (Keyder and Geffner 2009).

In optimal oversubscription planning, however, not much work was focused on heuristic search, and the progress was somewhat slower. A significant performance improvement was first reported by Mirkis and Domshlak (2013). They exploited *explicit abstractions* (Edelkamp 2001), which are 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) exploit the notion of *landmarks* for task reformulation, enriching the task with reachability information. Katz and Mirkis (2016) characterize tractable fragments of oversubscription planning tasks according to causal graph structure and variable domain sizes, and derive admissible estimates from these fragments. Unfortunately, even the simplest fragment under this characterization was found to be not solvable in polynomial time. Thus, additional restrictions are required to achieve tractability, similarly to the ones that were previously exploited in deriving heuristics for classical planning.

Our aim in this work is to lay grounds for adapting many existing and future heuristics for classical planning to oversubscription planning. In order to do that, similarly in spirit to what was done for net-benefit planning by Keyder and Geffner (2009), we suggest a reformulation of an oversubscription planning task to a classical planning task with two cost functions on operators. The first one corresponds to the original operator costs and is intended for restricting the set of feasible solutions. The second one corresponds to the net difference in state values. We then search for an optimal

*The participation in this work was done prior current position.

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

feasible solution of the reformulated planning task according to the second cost function. Using merge-and-shrink abstraction heuristics (Helmert *et al.* 2014) as an example, we show how this reformulation can exploit existing heuristics. Another contribution of our work is the first attempt at standardizing the benchmark set for oversubscription planning. For that, we introduce additional sections to PDDL intended to specify state-additive utility functions and a cost budget. Further, we adapt the Fast Downward translator (Helmert 2006) to parse these sections, and we create a collection of oversubscription planning benchmarks from the classical STRIPS domains used in 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 $1 \leqslant i \leqslant k$, 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.

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 $\mathcal{C}$ and 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 $\sum_{o\in\pi} \mathcal{C}(o) \leqslant \mathsf{B}$. We call an s_I-plan a *plan* for Π_{OSP}. By the value $\hat{\mathsf{u}}(\pi)$ of a plan we refer to the value 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 the plans. While an empty operator sequence is a plan for every OSP task, the objective in oversubscription planning is to find a plan achieving a state of high utility and *optimal oversubscription planning* is devoted to searching for optimal plans only. In what follows, we restrict our attention to *additive utility function*, computed as a sum over the state facts. Such value functions have the form $\mathsf{u}(s) = \sum_{f\in s} \mathsf{u}'(f)$, where u' is a function mapping facts to non-negative real values. Slightly abusing the 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 $h : S \times \mathbb{R}^{0+} \mapsto \mathbb{R}^{0+} \cup \{\infty\}$ from state-budget pairs to a non-negative real value or infinity. 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 $\langle V, O, s, \mathcal{C}, \mathsf{u}, \mathsf{b}\rangle$ or to $-\infty$ if no such plan exists. A heuristic h is admissible if $h \geqslant h^*$. 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 extended classical planning formalism that limits the set of feasible solutions with secondary cost functions and can have negative values in the primary cost function.

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 and*

- s_I *is a state, called* initial state
- G *is a partial state, called* goal state
- $\mathcal{C}_0$ *is a cost function*
- $\mathscr{C} = \{\langle \mathcal{C}_i, \mathsf{B}_i\rangle \mid 1 \leqslant i \leqslant n\}$ *where* $\mathcal{C}_i$ *is a non-negative cost function and* $\mathsf{B}_i \in \mathbb{R} \cup \{\infty\}$.

We call the cost function $\mathcal{C}_0$ the primary cost function and each cost function $\mathcal{C}_i$ with $1 \leqslant i \leqslant n$ a secondary cost function. An operator sequence π is a *plan* for Π_{MCF} if G is consistent with $s[\![\pi]\!]$ and $\sum_{o\in\pi} \mathcal{C}_i(o) \leqslant \mathsf{B}_i$ for $1 \leqslant i \leqslant n$. A plan is optimal if it has minimal primary cost among all plans of Π_{MCF}. A *heuristic* for MCF planning task $\Pi_{\text{MCF}} = \langle V, O, s_I, G, \mathcal{C}_0, \mathscr{C}\rangle$ with states S is a mapping $h : S \times \mathbb{R}^{|\mathscr{C}|} \mapsto \mathbb{R} \cup \{-\infty, \infty\}$. 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 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 $h \leqslant h^*$.

A *classical planning task* is an MCF planning task $\Pi = \langle V, O, s_I, G, \mathcal{C}_0, \varnothing\rangle$ 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 $\Pi = \langle V, O, s_I, G, \mathcal{C}_0, \varnothing\rangle$.

In classical planning, abstractions can be obtained by e.g. projecting the problem on a subset of its variables (Edelkamp 2001), or through a merge-and-shrink process (Helmert *et al.* 2007; 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

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

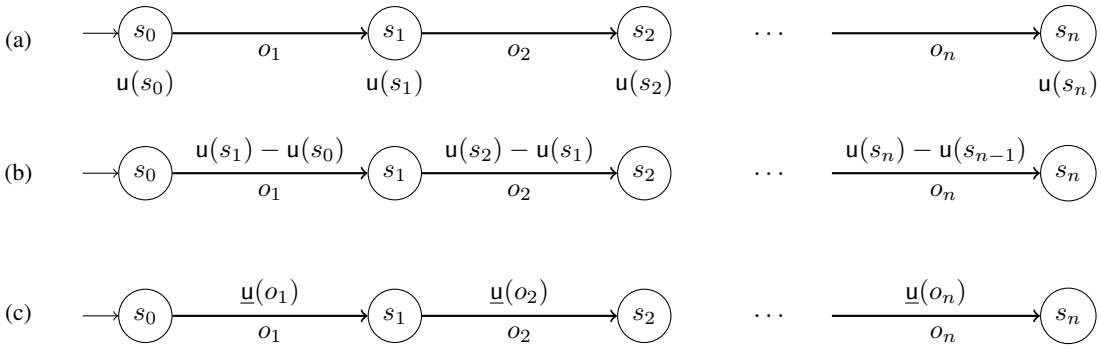

Figure 1: The figures show the idea behind reformulating an operator sequences with a state dependent utility function (a) into an operator sequences where a cost function reflects the utility difference between two successive states (b). The additive utility function allows for a state-independent cost function (c).

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 *et al.* 2007) by additionally requiring reachable abstract state distances under the secondary cost functions to be below their respecive 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 $c_i : L \mapsto \mathbb{R}$ $(1 \leqslant i \leqslant n)$, $T \subseteq S \times L \times S$ a set of labeled transitions, s_0 the initial state and S_* the goal states.

The induced transition 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 is consistent with $\mathsf{eff}(o)$, 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 $\alpha : S' \mapsto S^\alpha$ where S^α are the states of the transition system $\Theta^\alpha = \langle S^\alpha, L, \mathbf{c}, T^\alpha, s_0^\alpha, S_*^\alpha \rangle$ with $T^\alpha = \{\langle \alpha(s), o, \alpha(t) \rangle \mid (s, o, t) \in T\}$, $s_0^\alpha = \alpha(s_0)$ and $S_*^\alpha = \{\alpha(s) \mid s \in S\}$. Θ^α is called the abstract transition system.

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

Reformulation

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. We start by noting that for an additive state value function u, there is an easily computable finite upper bound

$$M := \sum_{v \in V} \max_{\vartheta \in dom(v)} \mathsf{u}(\langle v, \vartheta \rangle).$$

This upper bound allows us to switch from maximization to minimization of the utility value. Thus, our first step in the formulation is to switch to a new state value function $\underline{\mathsf{u}} : S \to \mathbb{R}^{0+}$ defined by $\underline{\mathsf{u}}(s) = M - \mathsf{u}(s)$, and the objective of the new task is to find a plan π *minimizing* the value $\hat{\mathsf{u}}(\pi)$. The idea behind our reformulation, illustrated in Figure 1, is to compute by how much each operator changes the utility of a state, if 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 1. *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 $v \in V \backslash \mathcal{V}(\mathsf{eff}(o))$, 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

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

Note that the cost function $\underline{\mathsf{u}}$ may have negative values. We say that an operator o *achieves utility* if $\underline{\mathsf{u}}(o) < 0$ and o *destroys utility* if $\underline{\mathsf{u}}(o) > 0$.

Theorem 2. *For a sequence of operators π applicable in state s, we have $\underline{\mathsf{u}}(s) + \sum_{o \in \pi} \underline{\mathsf{u}}(o) = \underline{\mathsf{u}}(s[\![\pi]\!])$.*

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 with multiple cost functions and an empty goal.

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

- $G = \varnothing$*, 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 3. *Let* Π_{OSP} *be an oversubscription planning task and* Π_{MCF} *its multiple cost function reformulation. If* π *is a plan of* Π_{OSP} *with utility* $\hat{\mathsf{u}}(\pi)$ *then* π *is a plan of* Π_{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_{MCF}, so if π is a plan in one task, it is certainly applicable in the other task and ends in the same state, i.e. $s_I[\![\pi]\!]$ is well-defined and the same state in both tasks.

The operator sequence π respects the bounds of Π_{OSP} iff $\sum_{o\in\pi} \mathcal{C}(o) \leqslant \mathsf{B}$ iff π respects the bounds of the (only) secondary cost function of Π_{MCF}. Therefore, and because all states are goal states in Π_{MCF}, π is a plan in Π_{OSP} iff it is a plan in Π_{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 2. □

As $\mathsf{u}(s_I)$ is constant, a plan π maximizes $\hat{\mathsf{u}}(\pi)$ iff it minimizes $\mathcal{C}_0(\pi)$ and the following result directly follows:

Corollary 1. *An oversubscription planning task and its multiple cost function reformulation have the same optimal plans.*

Heuristics for OSP via Reformulation

Having proposed the OSP reformulation, we now turn our attention to devising heuristics for MCF planning. We start by clarifying how heuristics from MCF planning can be integrated into an OSP approach.

Definition 3. *Let* Π_{OSP} *be an OSP task,* Π_{MCF} *its multiple cost function reformulation, and* S *the states of* Π_{OSP}*. Let* $h_{\text{MCF}} : S \times \mathbb{R} \mapsto \mathbb{R} \cup \{-\infty, \infty\}$ *be a heuristic for* Π_{MCF}*. The* multiple cost function reformulation heuristic *of* h_{MCF}*, denoted by* $h^{\mathscr{R}}_{\text{MCF}}$ *is defined by* $h^{\mathscr{R}}_{\text{MCF}}(s, \mathsf{b}_s) = \mathsf{u}(s) - h_{\text{MCF}}(s, \mathsf{b}_s)$*.*

Multiple cost function reformulation heuristics are heuristics for OSP tasks. The following lemma establishes the connection between the informativeness of heuristics for MCF planning tasks and their multiple cost function reformulation heuristics.

Lemma 1. *For an OSP task* Π_{OSP} *and* $\Pi_{\text{MCF}} = \Pi^{\mathscr{R}}_{\text{MCF}}$*, we have* $h^*_{\Pi_{\text{OSP}}} = (h^*_{\Pi_{\text{MCF}}})^{\mathscr{R}}$*.*

The lemma is a direct outcome from Theorem 3. We use it in order to show the following main result.

Theorem 4. *Let* $\Pi_{\text{OSP}} = \langle V, O, s_I, \mathcal{C}, \mathsf{u}, \mathsf{B}\rangle$ *be an OSP task,* Π_{MCF} *its multiple cost function reformulation, and* h_{MCF} *an admissible heuristic for* Π_{MCF}*. Then* $h^{\mathscr{R}}_{\text{MCF}}$ *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 3, we can rewrite $h^*_{\text{MCF}}(s, \mathsf{b}_s)$ as $\mathsf{u}(s) - (h^*_{\text{MCF}})^{\mathscr{R}}(s, \mathsf{b}_s)$, which is $\mathsf{u}(s) - h^*_{\text{OSP}}(s, \mathsf{b}_s)$ according to Lemma 1.

From Definition 3 we have

$$h^{\mathscr{R}}_{\text{MCF}}(s, \mathsf{b}_s) = \mathsf{u}(s) - h_{\text{MCF}}(s, \mathsf{b}_s),$$

and from admissibility of h_{MCF} we have

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

so

$$\begin{aligned} h^{\mathscr{R}}_{\text{MCF}}(s, \mathsf{b}_s) &\geqslant \mathsf{u}(s) - h^*_{\text{MCF}}(s, \mathsf{b}_s) \\ &= \mathsf{u}(s) - (\mathsf{u}(s) - h^*_{\text{OSP}}(s, \mathsf{b}_s)) \\ &= h^*_{\text{OSP}}(s, \mathsf{b}_s). \end{aligned}$$

□

Abstraction Heuristics for MCF Planning

Having established how admissible heuristics of MCF planning tasks can be exploited for deriving admissible heuristics of OSP tasks, we now show a concrete example of this by deriving a merge-and-shrink heuristic for OSP. We start by introducing a generic scheme for abstraction heuristics.

Definition 4. *Let* Π_{MCF} *be an MCF task,* α *be an abstraction and* Θ^α *its abstract transition system. The heuristic* $h^\alpha_\Theta : (S \times \mathbb{R}^n) \mapsto \mathbb{R} \cup \{-\infty, \infty\}$ *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* $1 \leqslant i \leqslant n$*,* $\mathcal{C}_i(\rho) \leqslant \mathsf{b}_i$*, 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 a uninformative heuristic. Concrete choice of methods for constructing Θ^α in step (I) should aim at preventing or at least

alleviate this problem. In this work, we use existing methods for constructing merge-and-shrink abstractions (Sievers *et al.* 2014), leaving the methods for constructing abstractions that avoid negative cost cycles for 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 maximizing the informativeness of the heuristic, step (III) should be performed for every evaluated state, given the reachability of abstract states under the budget b for that concrete state. Additionally, the possibly negative cost function madates 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

To empirically evaluate the practical potential of our approach, we first create a benchmark set for oversubscription planning.

Creating a Benchmark Set for OSP

Since no official, publicly available benchmark set for oversubscrition planning is currently available, we had to create one. We created a benchmark suite similar to Domshlak and Mirkis (2015), based on the collection of classical International Planning Competition (IPC) domains. However, in contrast to previous approaches, 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 the information available at `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 either 25%, 50%, 75%, or 100% of the best known solution cost for the classical planning task, resulting in four variants for each classical planning domain. In the following, we refer to these numbers as different domain suites. Every fact in the goal of the classical planning task, we assigned the utility of 1, every other fact the value of 0.

We briefly describe how we modified the PDDL specification. We extended PDDL by two additional sections in the problem file. The first section (:BOUND contains the bound on the solution cost, while the second section contains the utility function. The second section (:UTILITY allows to provide 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 oversubscription planning tasks. Both the PDDL domain collection and the adapted translator are available on demand.

Transforming SAS$^+$ to SAS

Fast Downward translates PDDL into SAS$^+$ representation, which is more compact than SAS. Thus, to apply our techniques to the problems in our benchmark set, we need to transform these tasks to 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.

	25%		50%		75%		100%	
Coverage	Bl	M&S	Bl	M&S	Bl	M&S	Bl	M&S
airport	**20**	9	**16**	9	**15**	9	**15**	9
miconic	85	85	**56**	55	**50**	49	45	45
mprime	**13**	12	**10**	9	7	7	6	6
mystery	10	10	**9**	8	7	7	7	7
scanalyzer08	**13**	12	12	12	**12**	11	**12**	11
scanalyzer11	**10**	9	9	9	**9**	8	**9**	8
tetris14	**17**	2	**14**	2	**10**	2	**8**	2
tidybot11	**20**	1	**20**	1	**16**	1	**13**	1
tidybot14	**20**	0	**17**	0	**12**	0	**6**	0
woodwork08	**25**	24	12	12	9	9	7	7
woodwork11	**18**	17	7	7	4	4	2	2
pipes-notank	**40**	18	**29**	18	**20**	17	14	**15**
pipes-tank	**28**	25	18	**19**	14	**15**	**11**	10
depot	15	15	8	**9**	6	**8**	4	**6**
openstacks08	29	**30**	24	**27**	23	**26**	22	**25**
openstacks11	20	20	17	**18**	17	**18**	17	**18**
openstacks14	19	19	10	**11**	5	**9**	3	**8**
parcprinter08	15	**16**	12	**13**	10	**12**	9	**10**
parcprinter11	11	**12**	8	**9**	6	**8**	5	**6**
parking11	10	10	1	**2**	0	**2**	0	**2**
parking14	11	11	0	**4**	0	**4**	0	**4**
satellite	8	**9**	6	6	3	3	3	3
Sum equal	587	587	450	450	376	376	349	349
Sum all	1044	953	765	710	631	605	567	554

Table 1: Per-domain coverage comparison of the blind heuristic (Bl) and merge-and-shrink (M&S) for the four domain suites. Top part depicts domains with advantage to the blind heuristic in all suites, middle part depicts domains with mixed results, while bottom parts shows domains with advantage to the merge-and-shrink heuristic in all suites.

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:

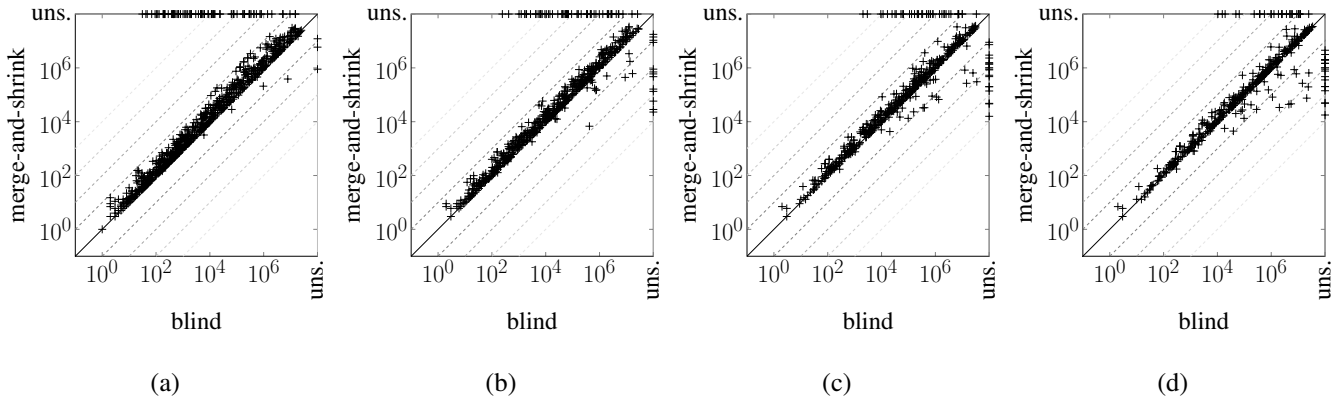

Figure 2: Comparison of the number of expansions performed with the blind and the merge-and-shrink heuristics for different problem suites, (a) 25%, (b) 50%, (c) 75%, and (d) 100%.

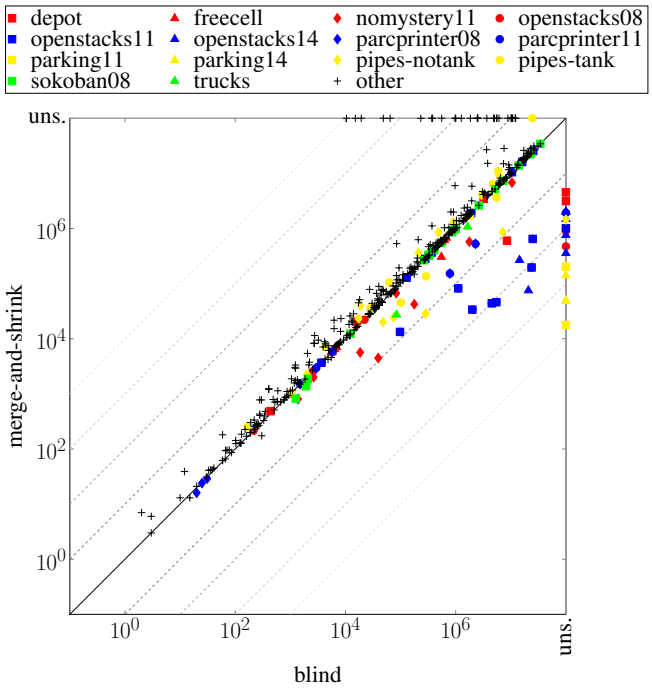

Figure 3: Domain-wise comparison of the number of expansions performed with the blind and the merge-and-shrink heuristics for the 100% problem suite. Domains where merge-and-shrink exhibits better performance in terms of the number of expansions are emphasized.

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

M&S A merge and shrink approach to compute $h^{\alpha}_{\Theta}(s, \mathbf{b})$. For step (I) we used the bisimulation based shrinking, and as merge strategy SCC-DFP (Sievers *et al.* 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 3.

For a fair comparison, we set the guidance heuristic in all our approaches to the blind heuristic. 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.

Results

Table 1 shows the per-domain coverage, 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 summed in the "Sum equal" row. Overall, the baseline still achieves the higher coverage, with the difference getting smaller towards suites with larger cost bounds, namely, 91 for suite 25, 55 for suite 50, 26 for suite 75, and 13 for suite 100. We note that the domains AIRPORT, TETRIS, TIDYBOT11, TIDYBOT14, PIPESWORLD-NOTANKAGE, and PIPESWORLD-TANKAGE are responsible for most of the difference, due to the construction of merge-and-shrink abstraction not being finished within the time bound. With the exception of these 6 domains, merge-and-shrink loses at most one task in coverage per suite. Looking at the bottom part of the table, there are several domains where the performance improves significantly, across the suites. The improvement is getting larger towards suites with larger cost bounds. This is consistent with the overall results, hinting that merge-and-shrink would be beneficial for larger cost bounds.

In order to look beyond the coverage, Figure 2 depicts the comparison in terms of the number of node expansions

performed by the branch-and-bound search algorithm. The cost bound increases, from left to right. Figure 2 (a) shows the expansions for the suite 25, where there is a clear advantage to the blind heuristic, but as we move to larger bounds, the advantage becomes moderate, and then turns into somewhat complementary results in Figure 2 (d) for suite 100. Note that, in contrast to the classical optimal planning with A^*, here a dominating heuristic does not guarantee a smaller number of expansions. However, when the blind heuristic has a smaller number of expansions, it is always within one order of magnitude. For the other case, when merge-and-shrink dominates in the number of expansions, it can get to two orders, and more.

Further focusing on suite 100, the per-domain expansions can be observed in Figure 3. Improvement over the baseline can be observed in many domains, in particular in some domains where this improvement is not reflected in the overall coverage, probably due to the costly pre-search abstraction computation. These include FREECELL, NOMYSTERY, PIPESWORLD-NOTANKAGE, PIPESWORLD-TANKAGE, SOKOBAN08, and TRUCKS. There are additional 8 domains where the improvement in expansions is reflected in the coverage, namely DEPOTS, OPENSTACKS08, OPENSTACKS11, OPENSTACKS14, PARC-PRINTER08, PARC-PRINTER11, PARKING11, and PARKING14.

Conclusions and Future Work

In this work we have introduced a reformulation of an oversubscription planning task to a classical planning task with two cost functions on operators, allowing to ease the adaptation of the existing heuristics for classical planning to the oversubscription planning setting. We have shown with the merge-and-shrink heuristic how such an adaptation can be done. Our experimental evaluation shows the feasibility of such an approach. In order to perform the experimental evaluation, in the absense of a standard benchmark set and a PDDL fragment for describing oversubscription planning tasks, we have introduced such a fragment and created the benchmark set, as well as provided a translator from PDDL to a multi-valued variables formalism SAS, which is used internally by most modern planners. By adapting the Fast Downward planning framework, with many classical planning heuristics implemented, to oversubscription planning formalism we have simplified the future adaptation of classical planning heuristics to oversubscription planning via the suggested reformulation.

In future work we intend to investigate such adaptations. Further, we intend to investigate the interplay between the reformulation and heuristics 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 (Mirkis and Domshlak 2013; 2014) and explore the connections between the reformulation and their approach. In addition, we would like to explore various heuristics for nodes ordering in the branch-and-bound search. Last, but not least, we would like to adapt the existing search pruning techniques for classical planning (Domshlak *et al.* 2012; Alkhazraji *et al.* 2012) to the branch-and-bound search over the oversubscription planning tasks.

References

Yusra Alkhazraji, Martin Wehrle, Robert Mattmüller, and Malte Helmert. A stubborn set algorithm for optimal planning. In Luc De Raedt, Christian Bessiere, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter Lucas, editors, *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI 2012)*, pages 891–892. IOS Press, 2012.

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

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

Blai Bonet and Héctor Geffner. Planning as heuristic search. *Artificial Intelligence*, 129(1):5–33, 2001.

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

Carmel Domshlak and Vitaly Mirkis. Deterministic oversubscription planning as heuristic search: Abstractions and reformulations. *Journal of Artificial Intelligence Research*, 52:97–169, 2015.

Carmel Domshlak, Michael Katz, and Alexander Shleyfman. Enhanced symmetry breaking in cost-optimal planning as forward search. In Lee McCluskey, Brian Williams, José Reinaldo Silva, and Blai Bonet, editors, *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling (ICAPS 2012)*, pages 343–347. AAAI Press, 2012.

Carmel Domshlak, Jörg Hoffmann, and Michael Katz. Red-black planning: A new systematic approach to partial delete relaxation. *Artificial Intelligence*, 221:73–114, 2015.

Stefan Edelkamp. Planning with pattern databases. In Amedeo Cesta and Daniel Borrajo, editors, *Proceedings of the Sixth European Conference on Planning (ECP 2001)*, pages 84–90. AAAI Press, 2001.

Patrik Haslum and Héctor Geffner. Admissible heuristics for optimal planning. In Steve Chien, Subbarao Kambhampati, and Craig A. Knoblock, editors, *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling (AIPS 2000)*, pages 140–149. AAAI Press, 2000.

Patrik Haslum, Adi Botea, Malte Helmert, Blai Bonet, and Sven Koenig. Domain-independent construction of pattern database heuristics for cost-optimal planning. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence (AAAI 2007)*, pages 1007–1012. AAAI Press, 2007.

Malte Helmert and Carmel Domshlak. Landmarks, critical paths and abstractions: What's the difference anyway? In Alfonso Gerevini, Adele Howe, Amedeo Cesta, and Ioannis

Refanidis, editors, *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS 2009)*, pages 162–169. AAAI Press, 2009.

Malte Helmert, Patrik Haslum, and Jörg Hoffmann. Flexible abstraction heuristics for optimal sequential planning. In Mark Boddy, Maria Fox, and Sylvie Thiébaux, editors, *Proceedings of the Seventeenth International Conference on Automated Planning and Scheduling (ICAPS 2007)*, pages 176–183. AAAI Press, 2007.

Malte Helmert, Patrik Haslum, Jörg Hoffmann, and Raz Nissim. Merge-and-shrink abstraction: A method for generating lower bounds in factored state spaces. *Journal of the ACM*, 61(3):16:1–63, 2014.

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

Jörg Hoffmann and Bernhard Nebel. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302, 2001.

Erez Karpas and Carmel Domshlak. Cost-optimal planning with landmarks. In Craig Boutilier, editor, *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*, pages 1728–1733. AAAI Press, 2009.

Michael Katz and Carmel Domshlak. Optimal additive composition of abstraction-based admissible heuristics. In Jussi Rintanen, Bernhard Nebel, J. Christopher Beck, and Eric Hansen, editors, *Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling (ICAPS 2008)*, pages 174–181. AAAI Press, 2008.

Michael Katz and Carmel Domshlak. Implicit abstraction heuristics. *Journal of Artificial Intelligence Research*, 39:51–126, 2010.

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

Michael Katz and Vitaly Mirkis. In search of tractability for partial satisfaction planning. In Subbarao Kambhampati, editor, *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016)*, pages 3154–3160. AAAI Press, 2016.

Emil Keyder and Héctor Geffner. Heuristics for planning with action costs revisited. In *Proceedings of the 18th European Conference on Artificial Intelligence (ECAI 2008)*, pages 588–592, 2008.

Emil Keyder and Héctor Geffner. Soft goals can be compiled away. *Journal of Artificial Intelligence Research*, 36:547–556, 2009.

Emil Keyder, Silvia Richter, and Malte Helmert. Sound and complete landmarks for and/or graphs. In Helder Coelho, Rudi Studer, and Michael Wooldridge, editors, *Proceedings of the 19th European Conference on Artificial Intelligence (ECAI 2010)*, pages 335–340. IOS Press, 2010.

Vitaly Mirkis and Carmel Domshlak. Abstractions for oversubscription planning. In Daniel Borrajo, Subbarao Kambhampati, Angelo Oddi, and Simone Fratini, editors, *Proceedings of the Twenty-Third International Conference on Automated Planning and Scheduling (ICAPS 2013)*, pages 153–161. AAAI Press, 2013.

Vitaly Mirkis and Carmel Domshlak. Landmarks in oversubscription planning. In Torsten Schaub, Gerhard Friedrich, and Barry O'Sullivan, editors, *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI 2014)*, pages 633–638. IOS Press, 2014.

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

Florian Pommerening and Malte Helmert. A normal form for classical planning tasks. In Ronen Brafman, Carmel Domshlak, Patrik Haslum, and Shlomo Zilberstein, editors, *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling (ICAPS 2015)*, pages 188–192. AAAI Press, 2015.

Silvia Richter, Malte Helmert, and Matthias Westphal. Landmarks revisited. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI 2008)*, pages 975–982. AAAI Press, 2008.

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

Silvan Sievers, Martin Wehrle, and Malte Helmert. Generalized label reduction for merge-and-shrink heuristics. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI 2014)*, pages 2358–2366. AAAI Press, 2014.

David E. Smith. Choosing objectives in over-subscription planning. In Shlomo Zilberstein, Jana Koehler, and Sven Koenig, editors, *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS 2004)*, pages 393–401. AAAI Press, 2004.

Menkes van den Briel, Romeo Sanchez, Minh B. Do, and Subbarao Kambhampati. Effective approaches for partial satisfaction (over-subscription) planning. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI 2004)*, pages 562–569. AAAI Press, 2004.