Journal of Artificial Intelligence Research 70 (2021) 169-204 Submitted 07/2020; published 01/2021

Cost-Optimal Planning, Delete Relaxation, Approximability, and Heuristics

Christer Bäckström CHRISTER.BACKSTROM@LIU.SE
Peter Jonsson PETER.JONSSON@LIU.SE
Department of Computer Science
Linköping University
SE-581 83 Linköping, Sweden

Sebastian Ordyniak SORDYNIAK@GMAIL.COM
School of Computing
University of Leeds
LS2 9JT, Leeds, United Kingdom

Abstract

Cost-optimal planning is a very well-studied topic within planning, and it has proven to be computationally hard both in theory and in practice. Since cost-optimal planning is an optimisation problem, it is natural to analyse it through the lens of approximation. An important reason for studying cost-optimal planning is heuristic search; heuristic functions that guide the search in planning can often be viewed as algorithms solving or approximating certain optimisation problems. Many heuristic functions (such as the ubiquitious h^+ heuristic) are based on *delete relaxation*, which ignores negative effects of actions. Planning for instances where the actions have no negative effects is often referred to as *monotone planning*. The aim of this article is to analyse the approximability of cost-optimal monotone planning, and thus the performance of relevant heuristic functions. Our findings imply that it may be beneficial to study these kind of problems within the framework of parameterised complexity and we initiate work in this direction.

1. Introduction

We divide the introduction into two parts where we describe the background in Section 1.1 and present our results in Section 1.2.

1.1 Background

In *cost-optimal* planning, the input is a planning instance where each action has a weight, and the task is to find a plan with minimum total weight or report that no plan exists. From the viewpoint of computational complexity both satisficing and cost-optimal STRIPS planning are **PSPACE**-complete problems (Bylander, 1994). It has been observed, though, that cost-optimal planning is typically much harder to solve in practice than satisficing planning. This observation has been one major factor behind the intensive search for improved cost-optimal planners during recent years. Even though cost-optimal planning is a highly interesting and relevant problem in itself, there is another usage that is probably more important. Many state-of-the-art planners are based on heuristic search, and the heuristic function is very often based on solving or approximating cost-optimal planning. Pommeren-

ing and Helmert (2012) summarise the connections between cost-optimal planning, delete relaxation, approximability, and heuristics as follows.

> The delete-relaxation heuristic h^+ is a well-informed heuristic for classical planning. It is defined as the optimal solution of a task where all negative effects are ignored. Since it is intractable, it is seldom used in practice and approximations are used instead.

The aim of this article is to analyse these connections in greater detail. It is important to assign the term "approximation" the correct meaning in this context. The interpretation should be strict: an approximation algorithm (for a minimisation problem) always produces a solution whose measure is at most α times the optimal one (where α may be a constant or a function of some parameter based on the given instance). On top of this, we want our algorithm to produce the approximative solution substantially faster than it would take to find an optimal solution. Since monotone planning, i.e. planning with actions having empty delete lists, is **NP**-complete, we will restrict ourselves to polynomial-time computable approximation algorithms. We concentrate on the parameter $|V|$, i.e. the number of variables and there are many different reasons for this. It is a parameter that is easy to understand, and, historically, it has been the most common parameter for analysing quantities such as the run time of planning algorithms or the approximability of planning problems, cf. (Bäckström & Jonsson, 2017; Kronegger et al., 2019; Jonsson, 1999). We see that $|V|$ has a close connection to the size of underlying state-transition graph (the state space has size at most $O(2^{|V|} \cdot |A|)$ where $|A|$ is the number of actions) and it is thus a good parameter for estimating problem complexity. Another reason is that one is mainly interested in measuring approximability using parameters that do not "grow too rapidly". To exemplify, assume a class of planning problems X is approximable within $p(|V|)$. Since $|V| \leq ||\mathbb{P}||$ (where $||\mathbb{P}||$ is instance size), this result immediately implies that X is approximable within $p(||\mathbb{P}||)$, too. However, the implication in the other direction does not hold in general and it is thus reasonable to consider the result based on $|V|$ to be stronger than the result based on $||\mathbb{P}||$.

We begin by a quick comparison of unrestricted and monotone Strips planning. Approximating the length of optimal solutions for unrestricted Strips planning is a very hard problem: for every $\varepsilon < 1$, it is **PSPACE**-hard to approximate the length within the (very liberal) bound $2^{|V| \cdot (1-|V|^{-\varepsilon})}$, where $|V|$ denotes the number of variables (Jonsson, 1999). This result is essentially the best possible since such a plan may have length $2^{|V|} - 1$ at most. The situation may seem much better in monotone planning since the length can be approximated within $|V|$. This is an illusion, however, since if a monotone planning instance is solvable, then it always has a solution of length at most $|V|$. We are thus interested to prove or disprove the existence of approximation algorithms that achieve $o(|V|)$-approximations, and we describe a number of such results in this article. An outline of the results will be given in the next section.

The connections between approximation algorithms for cost-optimal planning and planning heuristics are well understood: approximation algorithms for cost-optimal planning can be viewed as admissible[1] heuristic functions, cf. the paper by Betz and Helmert (2009)

1. A heuristic function is *admissible* if it never overestimates the cost.

or Section 2.4 in the paper by Aghighi et al. (2016). Loosely speaking, the better approximation bound that is achieved by an approximation algorithm, the smaller the gap between the heuristic value and the optimal value. Similarly, a strong nonapproximability result for a class of planning instances implies that the gap is large—for any polynomial-time computable heuristic function, there are instances that the function will severely underestimate. One may thus view approximation bounds as a way of measuring how well-informed heuristics possibly can be when applied to a certain set of Strips instances.

1.2 Our Results

We continue by briefly discussing the results presented in this article. From now on, we only consider monotone planning unless otherwise stated.

Approximability. Monotone planning comes in two different flavours: (1) where only positive preconditions are allowed (corresponding to restricted classical Strips with preconditions that are sets of unnegated variables, add lists that are sets of unnegated variables, and empty delete-lists) and (2) where free preconditions are allowed (corresponding to restricted state-variable planning with two-valued domains $\{0, 1\}$ and actions that only increase variable values). We will see that (1) and (2) behave quite differently. Assume we consider instances with free preconditions and where the action weights are bounded by $\beta(|V|)$ for some function β. Our results will show that this bound on action weights is directly connected to approximability. Under fairly mild assumptions on β, it is **NP**-hard to approximate cost-optimal planning within a bound that is very close to $\beta(|V|)$ and the problem is polynomial-time approximable within $|V| \cdot \beta(|V|)$ (Theorem 9). The hardness result holds even for instances restricted to having one precondition and one effect. These bounds are not useful for approximating minimum plan length since every weight can be assumed to be 1 in that case. We thus present the following complementary result: length-optimal planning is approximable in polynomial time within $|V|$ but it is **NP**-hard to approximate it within $|V|^{1-\varepsilon}$ for every $\varepsilon > 0$ (Theorem 11). The hardness result holds even if we restrict ourselves to instances having actions with at most one precondition and one effect.

The situation becomes different for monotone planning restricted to positive preconditions (Section 4). The currently best known approximation bound states that it is **NP**-hard to approximate the length of optimal plans within $(1 - o(1)) \cdot \ln |V|$. We show that plan length is **NP**-hard to approximate within $g_c(|V|)$ for all $c < 1/2$, where g_c is a certain function that grows slower than all polynomial functions but faster than all polylogarithmic functions (Watel & Weisser, 2016). In particular, $g_c(|V|)$ grows *much* faster than $(1-o(1)) \cdot \ln |V|$. This is thus a significant strengthening of the previous bound. This result holds when restricted to instances having at most two preconditions and one effect. Unfortunately, the result is not accompanied by an improved upper bound (the currently best known approximability bound is $|V|$) so there is still a substantial gap between the lower and upper bound. Closing this gap appears difficult since there are certain theoretical barriers that need to be overcome. This leads us to consider monotone planning with only one precondition—this class of planning problems have been considered by, for instance, Keyder and Geffner (2009) when constructing heuristic functions. We prove that cost-optimal planning can be approximated within $|V|^{\varepsilon}$ for every $\varepsilon > 0$ and that this result holds even when an unbounded number of effects are allowed. Good approximation bounds can thus

	at most one prec.	unbounded number of prec.
positive prec.	approx: $\|V\|^{\varepsilon}$, $\varepsilon > 0$ nonapprox: $\log^{2-\varepsilon} \|V\|$, $\varepsilon > 0$	approx: $\|V\| \cdot \beta(\|V\|)$ nonapprox: $g_c(\|V\|)$, $c < 1/2$
free prec.	approx: $\|V\| \cdot \beta(\|V\|)$ nonapprox: close to $\beta(\|V\|)$	

Figure 1: Summary of (non)approximability results for COP with respect to preconditions

	at most one prec.	unbounded number of prec.
positive prec.	approx: $\|V\|^{\varepsilon}$, $\varepsilon > 0$ nonapprox: $\log^{2-\varepsilon} \|V\|$, $\varepsilon > 0$	approx: $\|V\|$ nonapprox: $g_c(\|V\|)$, $c < 1/2$
free prec.	approx: $\|V\|$ nonapprox. $\|V\|^{1-\varepsilon}$, $\varepsilon > 0$	

Figure 2: Summary of (non)approximability results for LOP with respect to preconditions

be obtained by choosing ε close to 0. We additionally prove that plan length cannot be approximated within $\log^{2-\varepsilon} |V|$ for any $\varepsilon > 0$, unless **NP** has quasi-polynomial Las-Vegas algorithms (i.e. every problem in **NP** has a probabilistic algorithm with expected quasi-polynomial run-time and zero error probability.) These results are proven in Theorems 22 and 23, respectively.

We use COP to denote cost optimal planning and LOP to denote length optimal planning. Our (non)approximability results for COP are summarised in Figure 1 and the results for LOP can be found in Figure 2. All approximability results hold for instances where the actions may have an unbounded number of effects while all nonapproximability results hold for instances where actions have at most one effect. One can thus note (and this may be slightly surprising) that the (non)approximability of these problems is closely connected to the type and number of preconditions while the effects have a negligible impact.

Heuristics. We will now sketch how to transfer the (non)-approximability results to the world of heuristics. Our presentation is partly based on Sec. 2.4 in the paper by Aghighi et al. (2016). We ignore the costs of actions for the moment in order to streamline the presentation, i.e. we only consider heuristics that estimate plan length. Note that if we analyse heuristic functions that take costs into account then we can only obtain worse approximation bounds (compared to the case when we restrict ourselves to plan length) since such heuristics are applicable to a larger set of possible inputs. Let $OPT(\cdot)$ denote the measure of an optimal solution to a given instance of an optimisation problem. Given a Strips instance $\mathbb{P} = \langle V, A, I, G \rangle$ and a state s, we define the perfect heuristic function h^* as usual: $h^*(\mathbb{P}, s) = OPT(\langle V, A, s, G \rangle)$. We begin by considering admissible heuristics. Let us consider Strips instances with free preconditions. Our results say that for arbitrary $\varepsilon > 0$, there is no polynomial-time computable function f satisfying

$$OPT(\mathbb{P}) \leq f(\mathbb{P}) \leq |V|^{1-\varepsilon} \cdot OPT(\mathbb{P})$$

unless **P**=**NP**. Assume now that h is a polynomial-time computable and admissible heuristic function for Strips. Here it is important to note that we allow h to be *any* such function, and it is not required to be based on delete relaxation. Even if we restrict ourselves to monotone instances $\mathbb{P}$, the function h cannot achieve better than

$$\frac{h^*(\mathbb{P}, s)}{|V|^{1-\varepsilon}} \leq h(\mathbb{P}, s) \leq h^*(\mathbb{P}, s)$$

for any $\varepsilon > 0$; if so, the function $h(\mathbb{P}, s) \cdot |V|^{1-\varepsilon}$ would violate the nonapproximability result. Clearly, h cannot perform better on unrestricted Strips instances so the bound holds in the general case. Let us view this from a slightly different angle. We let h^+ denote the delete relaxation heuristic: we may view it as a function that takes a Strips instance $\langle V, A, I, G\rangle$ and a state s, and computes the shortest plan for the instance (V, A^+, s, G) where A^+ contains the actions in A with emptied delete lists. The function h^+ is **NP**-hard to compute so we may want to approximate it. If there exists an admissible heuristic function h satisfying

$$\frac{h^+(\mathbb{P}, s)}{|V|^{1-\varepsilon}} \leq h(\mathbb{P}, s) \leq h^+(\mathbb{P}, s),$$

for any $\varepsilon > 0$ then h is not polynomial-time computable by our results.

If we consider Strips instances with only positive preconditions, then we have the bounds $\frac{h^*(\mathbb{P},s)}{g_c(|V|)} \leq h(\mathbb{P}, s) \leq h^*(\mathbb{P}, s)$ and $\frac{h^+(\mathbb{P},s)}{g_c(|V|)} \leq h(\mathbb{P}, s) \leq h^+(\mathbb{P}, s)$, for $c < 1/2$. This may appear to be much better since g_c is a function that grows slower than any polynomial function. As pointed out earlier, there are unfortunately no known polynomial-time algorithms that achieve a better bound than $|V|$. We conclude that polynomial-time computable approximations of h^+ are necessarily quite weak. One may exemplify with the *max heuristic* $h^{\max}$ that is a well-known admissible heuristic that can be computed in polynomial time. The heuristic is unfortunately known for not being very informative, cf. Helmert and Domshlak (2009) and Ghallab et al. (2016, p. 39). Our result shows that this is a property that it necessarily shares with *all* admissible and polynomial-time computable heuristics that attempt to approximate h^+—such functions cannot even produce a polylogarithmic approximation of h^+ under plausible complexity-theoretic assumptions.

The situation is more complex for non-admissible heuristics and full details together with slightly sharper bounds can be found in Section 5. In brief, we prove that no polynomial-time computable function h can achieve $\frac{OPT(\mathbb{P})}{|V|^{\varepsilon}} \leq h(\mathbb{P}) \leq |V|^{1-\varepsilon} \cdot OPT(\mathbb{P})$ for any $0 < \varepsilon < 1$ when considering instances with free preconditions. We may, for instance, set ε to $1/2$ and arrive at the bound $\frac{OPT(\mathbb{P})}{\sqrt{|V|}} \leq h(\mathbb{P}) \leq \sqrt{|V|} \cdot OPT(\mathbb{P})$. If we restrict ourselves to positive preconditions, then we have the bound $\frac{OPT(\mathbb{P})}{\log^k |V|} \leq h(\mathbb{P}) \leq \log^k(|V|) \cdot OPT(\mathbb{P})$ for arbitrary $k \geq 0$. The fact that the overestimation may be quite substantial—at least polylogarithmic in the number of variables—has been observed in practice; for instance, Domshlak et al. (2015) write the following about a particular nonadmissible polynomial-time computable heuristic function.

> Foremost, we observe in Section 5.1 that h^{RB}_{repair} is prone to dramatic overestimation even in trivial examples...

Our results give a solid theoretical explanation for this behaviour.

Fixed-parameter tractability *Parameterised complexity* is an important branch of computational complexity that focuses on classifying computational problems according to their inherent difficulty with respect to multiple parameters, and the complexity of a problem is then measured as a function of those parameters. This allows for a more fine-grained classification of computationally hard problems than in classical computational complexity where the complexity of a problem is only measured in the input size. Of particular interest are *fixed-parameter tractable* problems, i.e. solvable in $f(k) \cdot poly(n)$ time where f is an arbitrary function, k is some parameter, and n is the (ordinary) size of the instance. For instances where the parameter k is sufficiently small, we thus have access to an efficient solution method. Furthermore, we know that the dependence on the parameter k is limited— it is an important difference between an algorithm running in $f(k) \cdot poly(n)$ time and an algorithm running in, say, n^k time.

Our previous hardness results indicate that bounding parameters such as action weight and the number of preconditions and/or effects is not sufficient for obtaining polynomial-time approximation algorithms with good performance and, naturally, not for obtaining exact algorithms either. We thus turn our attention towards finding parameters that may serve as the basis for fixed-parameter tractable algorithms for monotone planning (Section 6). It is a plausible working hypothesis that the planning process becomes more difficult if there are many actions that achieve the same effect, and it is known that there are efficiently solvable special cases if there is only one action that can achieve a particular effect (Bäckström & Nebel, 1995; Bäckström et al., 2015). Inspired by this, we first show that directly bounding the number of actions that achieve an effect cannot lead to a fixed-parameter tractable algorithm. However, a fixed-parameter tractable algorithm exists if we use a variant of this parameter and only consider instances with *unary* actions, i.e. actions that only have one effect. Unary actions are interesting for many different reasons and we merely point out that heuristics that exploit acyclic causal graphs are based on unary action. A well-known example is the original heuristic used by the Fast Downward planner (Helmert, 2006). Even though our results may appear preliminary, we view this approach as a viable research direction towards a better understanding of monotone planning. An interesting aspect of our results is that they are only based on the effects of actions, thus making them very different from the approximability results presented in Sections 3 and 4.

2. Preliminaries

This section is divided into three parts. In the first two parts, we introduce general Strips planning and the optimisation framework, respectively, and we present some useful results concerning monotone planning in the third part. Note that we allow negative preconditions in our basic formalism, thus making it equivalent to the SAS^+ formalism restricted to two-valued variable domains. We begin by providing some mathematical terminology and notation that will be frequently used throughout the article. We let log denote the logarithm

function with base 2 and we let ln denote the logarithm function with base e. We follow common practice and let $\log^k n$ denote $(\log n)^k$. A sequence of objects $x_1, x_2, \ldots, x_n$ is written $\langle x_1, x_2, \ldots, x_n \rangle$, with $\langle\rangle$ denoting the empty sequence. We write $|X|$ to denote the cardinality of a set X or the length of a sequence X, i.e., the number of elements in X, and we write $||X||$ to denote the size of x, i.e., the number of bits of the representation of X.

2.1 Planning

We will only consider binary variables (i.e. propositional atoms) in this article. Let V be a set of variables. A state (over V) is a subset $s \subseteq V$, interpreted such that a variable $v \in V$ is true in s if and only if $v \in s$. The negation of a variable v is denoted $\overline{v}$ and a literal is either a variable or the negation of a variable. A set S of literals over V is consistent if there is no variable $v \in V$ such that both $v \in S$ and $\overline{v} \in S$. A consistent set of literals over V is referred to as a partial state (over V). Let s be a state and t a partial state. Then t is satisfied in a state s if (1) $v \in s$ for every $v \in t$ and (2) $v \notin s$ for every $\overline{v} \in t$. The update operator $\ltimes$ is defined such that $r = s \ltimes t$ is a state defined such that for every variable $v \in V$, $v \in r$ if and only if either (1) $v \in t$ or (2) $v \in s$ and $\overline{v} \notin t$.

Definition 1. A *Strips instance* is a tuple $\mathbb{P} = \langle V, A, I, G \rangle$ where V is a set of variables, A is a set of *actions*, I is a state over V and G is a partial state over V. Each action a in A has a *precondition* $\mathrm{pre}(a)$ and an *effect* $\mathrm{eff}(a)$, which are both partial states over V, together with a *weight* $\mathrm{w}(a)$ that is a member of $\mathbb{N}^+$. For any two states $s, t \subseteq V$ and action $a \in A$, *a is from s to t* if
(1) s satisfies $\mathrm{pre}(a)$ and
(2) $t = s \ltimes \mathrm{eff}(a)$.
Let ω be an action sequence over A. Then ω is a *plan* for $\mathbb{P}$ if $\omega = \langle a_1, \ldots, a_\ell \rangle$ and there are states $s_0, \ldots, s_\ell \subseteq V$ such that
(1) $s_0 = I$,
(2) s_ℓ satisfies G and
(3) a_i is from s_{i-1} to s_i for all i $(1 \leq i \leq \ell)$.
The sequence $\langle s_0, \ldots, s_\ell \rangle$ is referred to as the *state sequence* for ω from s_0. A plan ω for $\mathbb{P}$ is a *shortest plan* for $\mathbb{P}$ if there is no other plan ω' for $\mathbb{P}$ such that $|\omega'| < |\omega|$.

We extend the notion of weight as follows: $\mathrm{w}(A) = \sum_{a \in A} \mathrm{w}(a)$ and if $\omega = \langle a_1, \ldots, a_\ell \rangle$ is an action sequence over A, then $\mathrm{w}(\omega) = \sum_1^\ell \mathrm{w}(a_i)$. We also define the function $\mathrm{w}_{\max} : A \to \mathbb{N}^+$ such that $\mathrm{w}_{\max}(A) = \max_{a \in A} \mathrm{w}(a)$ and if $\omega = \langle a_1, \ldots, a_\ell \rangle$ is an action sequence over A, then $\mathrm{w}_{\max}(\omega) = \max_{1 \leq i \leq \ell} \mathrm{w}(a_i)$. A plan ω for $\mathbb{P}$ is an *optimal plan* for $\mathbb{P}$ if there is no other plan ω' for $\mathbb{P}$ such that $\mathrm{w}(\omega') < \mathrm{w}(\omega)$.

We use the notation $a \colon \mathrm{pre}(a) \overset{w}{\Rightarrow} \mathrm{eff}(a)$ to define an action a with precondition $\mathrm{pre}(a)$, effect $\mathrm{eff}(a)$ and weight w, or $a \colon \mathrm{pre}(a) \Rightarrow \mathrm{eff}(a)$ when the weight is implicitly assumed to be 1. We restrict ourselves to positive weights since zero weights and negative weigths often cause anomalies, cf. Richter and Westphal (2010) and Benton et al. (2010). However, we do believe that considering actions with zero and negative weights may be an interesting direction for future research. We will sometimes ignore weights entirely and consider the optimal plan length, i.e., the smallest number of actions, rather than the minimal plan weight. These two measures are obviously equivalent if we assume that every action has

weight 1, and this implies that cost-optimal planning can never be easier to approximate than length-optimal planning.

We define the following computational problems for Strips instances.

Plan Existence(PE)
Type: Decision problem.
Instance: A Strips instance $\mathbb{P}$.
Question: Does $\mathbb{P}$ have a plan?

Bounded Plan Existence(BPE)
Type: Decision problem.
Instance: A Strips instance $\mathbb{P}$ and a positive integer k.
Question: Does $\mathbb{P}$ have a plan ω with weight $\mathrm{w}(\omega) \leq k$?

Note that BPE can be used for checking if there is a plan of length k or less by assigning unit weight to the actions. For a class X of Strips instances, we write $\mathsf{PE}(X)$ and $\mathsf{BPE}(X)$ to denote PE and BPE, respectively, restricted to the instances in X.

We will mainly consider subclasses that are systematically defined by imposing various restrictions on the preconditions and effects of actions. For such classes we will use the notation $\textsc{Strips}(x, y)$ where x corresponds to restrictions on the precondition and y to restrictions on the effect. Each of these can be a positive integer or either of the symbols "k" and "$*$", optionally followed by a plus sign. An integer specifies a fixed maximal number of literals in the condition, while k specifies a maximal number of literals that is a parameter of the class. The $*$ symbol means that the number of literals is unrestricted. If followed by a plus sign then all literals must be positive. For instance, $\textsc{Strips}(2, *+)$ is the class of instances where the precondition has at most two literals (which may be positive or negative) and the effect can contain an arbitrary number of positive literals. We write, for instance, $\mathsf{PE}(x, y)$ and $\mathsf{BPE}(x, y)$ for PE and BPE restricted to the class $\textsc{Strips}(x, y)$. Sometimes, we also add a third element, *sat*, to the tuples to denote the restriction to solvable instances only. For instance, $\textsc{Strips}(1, *+, sat)$ denotes the class $\textsc{Strips}(1, *+)$ restricted to those instances that have at least one plan.

2.2 Optimisation and Approximation

An *optimisation problem* takes an instance and generates a solution that is optimal according to some specified *measure* such as plan length. Alternatively, we may ask only for the actual measure of an optimal solution and not for the solution itself: this is known as *non-constructive optimisation*. An optimisation problem can be either a *minimisation problem* or a *maximisation problem*, depending on if the measure should be minimised or maximised. We will exclusively consider minimisation problems in this article.

Let OPT be a function that denotes the measure of an optimal solution for an instance. Let f be a function that computes solutions to instances, and let m be a function that computes the measure of a solution. For instance, we may consider Strips instances, let the function f be a planner, and let m compute the length of a given plan—this is the setting for approximating the minimum length of Strips plans. For a minimisation problem, we

say that f approximates OPT within a factor α if

$$OPT(\mathbb{I}) \;\leq\; m(f(\mathbb{I})) \;\leq\; \alpha \cdot OPT(\mathbb{I})$$

for all instances $\mathbb{I}$. That is, the measure of the solution produced by f never underestimates the optimal value but it is at most a factor α larger than the optimal value. In the non-constructive version of approximation (*nc-approximation*), one let f be a function from instances to numerical values instead of instances. That is, we do not require concrete solutions to be computed. The definition is then very similar to that of ordinary approximation: for a minimisation problem, we say that f nc-approximates OPT within a factor α if

$$OPT(\mathbb{I}) \;\leq\; f(\mathbb{I}) \;\leq\; \alpha \cdot OPT(\mathbb{I})$$

for all instances $\mathbb{I}$.

The approximation factor α is often also a function of the instance. Furthermore, the purpose of approximation is that the approximation function f should be easier to compute than the actual optimum OPT. Formally, this can be defined as follows for the case of minimisation.

Definition 2. Let X be some set and let $g : X \to \mathbb{N}_\infty$ (where $\mathbb{N}_\infty = \mathbb{N} \cup \{\infty\}$) be an arbitrary function. We say that g is **NP**-hard if $\mathbf{NP} \subseteq \mathbf{P}^g$ (i.e. if every **NP**-complete problem can be solved by some polynomial-time algorithm having oracle access to g).

Let $g : X \to \mathbb{N}_\infty$ be a function that we wish to minimise. Let $f : X \to \mathbb{N}_\infty$ and $\alpha : X \to \mathbb{N}_\infty$ be functions. We say that f *approximates* g *within* α if for every $\mathbb{I} \in X$,

$$g(\mathbb{I}) \;\leq\; f(\mathbb{I}) \;\leq\; \alpha(\mathbb{I}) \cdot g(\mathbb{I}).$$

We additionally say that g is ***NP**-hard to approximate within* α if every function f that approximates g within α is **NP**-hard.

Note that $\mathbf{NP} \subseteq \mathbf{P}^g$ if and only if co**NP** $\subseteq \mathbf{P}^g$, so there is no need to consider co**NP**-hardness in this context.

We define the following two minimization problems for Strips planning.

Length-Optimal Planning (LOP)
Type: Minimzation problem.
Instance: A Strips instance $\mathbb{P}$.
Solution: A plan ω for $\mathbb{P}$.
Measure: The number of actions $|\omega|$ in ω.

Cost-Optimal Planning (COP)
Type: Minimzation problem.
Instance: A Strips instance $\mathbb{P}$.
Solution: A plan ω for $\mathbb{P}$.
Measure: The total cost $\mathrm{w}(\omega)$ of ω.

2.3 Monotone Planning

We say that a Strips instance is *monotone* if all actions have only positive effects, i.e. we consider the instance set Strips$(*, *+)$. In monotone planning, it consequently holds that $s \ltimes \mathrm{eff}(a) = s \cup \mathrm{eff}(a)$ for all states s and actions a. Some of the most important consequences of this are summarised in the following lemma.

Lemma 3. *Let $\mathbb{P} = \langle V, A, I, G\rangle$ be a Strips(*,*+) instance, let $\omega = \langle a_1, \ldots, a_\ell\rangle$ be a plan for $\mathbb{P}$ and let $\langle s_0, \ldots, s_\ell\rangle$ be the state sequence for ω from $s_0 = I$. Then,*

(a) $s_i = I \cup \mathrm{eff}(a_1) \cup \ldots \cup \mathrm{eff}(a_i)$, for all i $(1 \leq i \leq \ell)$, and

(b) $s_i \subseteq s_j$ for all i, j $(0 \leq i < j \leq \ell)$.

If ω is also an optimal plan, then

(c) $s_i \subset s_j$ for all i, j $(0 \leq i < j \leq \ell)$ and

(d) each $a \in A$ occurs at most once in ω.

Proof. We prove (a)-(d) below.

(a) Immediate since $s \ltimes \mathrm{eff}(a) = s \cup \mathrm{eff}(a)$ when $\mathrm{eff}(a)$ contains no negative literals.

(b) Immediate from (a).

(c) Suppose $s_{i-1} = s_i$ for some i $(1 \leq i \leq \ell)$. It follows from (b) that a_i is redundant in ω, so $\omega' = \langle a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_\ell\rangle$ is also a plan for $\mathbb{P}$. However, $\mathrm{w}(\omega) = \mathrm{w}(\omega') + \mathrm{w}(a_i)$ and $\mathrm{w}(a_i) > 0$ so $\mathrm{w}(\omega') < \mathrm{w}(\omega)$, which contradicts that ω is an optimal plan for $\mathbb{P}$. It follows that $s_{i-1} \neq s_i$ for all i $(1 \leq i \leq \ell)$.

(d) Suppose $a_i = a_j$ where $i < j$. Then $s_i = s_{i-1} \cup \mathrm{eff}(a_i)$ and $s_i \subseteq s_{j-1}$ according to (b), so $\mathrm{eff}(a_j) = \mathrm{eff}(a_i) \subseteq s_i \subseteq s_{j-1}$ and it follows that $s_j = s_{j-1} \cup \mathrm{eff}(a_j) = s_{j-1}$. This contradicts (c) and, thus, that ω is optimal, so we conclude that no action occurs more than once in ω.

□

Another consequence of monotonicity is that there is no need to consider negative goals. If an atom is set true by an action in a plan, then it can never be set to false again. Hence, no plan can contain an action that sets an atom that also occurs as a negative goal, so the instances can always be preprocessed by removing such actions.

We next use Lemma 3 for obtaining certain bounds on plan lengths and costs.

Corollary 4. *Let $\mathbb{P} = \langle V, A, I, G\rangle$ be a Strips(*,*+) instance and let $\omega = \langle a_1, \ldots, a_\ell\rangle$ be an optimal plan for $\mathbb{P}$. Then, (1) $|\omega| \leq |V|$ and (2) $\mathrm{w}(\omega) \leq |V| \cdot \mathrm{w}_{\max}(A)$.*

Proof. (1) follows from Lemma 3c and (2) follows from (1). □

The bounds in Corollary 4 immediately implies that the problems PE, BPE, LOP, and COP are in **NP** when restricted to monotone Strips instances. The bounds do not hold for general planning where we allow negative effects: there is, for instance, an infinite sequence of Strips$(*, 1)$ instances $\mathbb{P}_1, \mathbb{P}_2, \ldots$ such that $\mathbb{P}_i$ contains i variables and $2i$ actions, and the length of the shortest plan for $\mathbb{P}_i$ is $2^i - 1$ (Bäckström & Nebel, 1995, Theorem 2)

The computational complexity of monotone Strips planning with restricted preconditions and effects is quite well understood.

Theorem 5. *The following results are proved by Bylander (1994).*

1. *BPE$(0, 2+)$ is in **P** (Theorem 4.8),*
2. *BPE$(0, 3+)$ is **NP**-complete (Theorem 4.4),*
3. *BPE$(1+, 1+)$ is **NP**-complete (Corollary 4.3),*
4. *PE$(*+, *+)$ is in **P** (Footnote 4), and*
5. *PE$(1, 1+)$ is **NP**-complete (Corollary 3.6).*

The picture is less clear when it comes to approximation properties of monotone planning. As a warm-up, we will provide some general upper bounds for approximation of monotone Strips planning. Even though the bounds may seem trivial, we will see later on that some of them are close to being optimal.

Theorem 6. *LOP$(*+, *+)$ can be approximated within $|V|$ and COP$(*+, *+)$ can be approximated within $|V| \cdot \mathrm{w}_{\max}(A)$.*

Proof. That PE$(*+, *+)$ is in **P** since it can be solved by a simple greedy algorithm is well-known—one adds an arbitrarily chosen action a to the end of the plan whenever a is applicable in the state achieved by the current plan. We will use an algorithm that is very close to the algorithm formulated by Bäckström and Jonsson (2017) (see Figure 3). There are only two differences.

1. We begin the algorithm by checking whether the initial state satisfies the goal state or not. This ensures that the algorithm returns the empty plan if this is indeed the case.
2. We never add an action to the plan if it does not change the current state. This way, we guarantee that the plan does not contain more than $|V|$ actions.

Our approximation algorithm(s) work as follows. Given an $(*+, *+)$ instance, we first check whether it has a solution or not by using GreedyPlan. If it has a solution, then we output the plan ω, and otherwise we report that there is no solution. We verify that this algorithm achieves the required bounds. Let ω' denote an optimal plan, i.e. it has either shortest length (for the LOP case) or it has lowest cost (for the COP case).

Let us now consider the LOP bound. If $|\omega| = 0$, then $\omega = \omega'$ and the algorithm returns the optimal answer. Otherwise, we see that $1 \leq |\omega'| \leq |\omega| \leq |V|$ since each action in ω changes at least one variable. We conclude that the algorithm approximates the instance

within $|V|$. The COP bound can be proved analogously: If $|\omega| = 0$, then $\omega = \omega' = \langle\rangle$ so $\mathrm{w}(\omega) = \mathrm{w}(\omega') = 0$ and the algorithm returns the optimal answer. Otherwise, we see that $1 \leq \mathrm{w}(\omega') \leq \mathrm{w}(\omega) \leq |V| \cdot \mathrm{w}_{\max}(A)$.

□

```
1 function GreedyPlan(⟨V, A, I, G⟩)
2   if G is satisfied by I then return ⟨⟩
3   s := I; ω := ⟨⟩
4   while there is some a ∈ A such that pre(a) is satisfied by s and s ≠ s ⋉ eff(a) do
5     s := s ⋉ eff(a)
6     Add a to the end of ω
7     Remove a from A
8   if G is satisfied by s then return ω
9   else reject
```

Figure 3: Greedy algorithm for solving $(*+, *+)$ instances

We now turn our attention to Strips$(*, *+)$ instances where negative preconditions are allowed. An important difference compared to Strips$(*+, *+)$ is that PE$(*, *+)$ is an **NP**-complete problem, and it is consequently computationally hard to verify whether a given instance has a solution or not. This implies that approximation is only meaningful for instances that are known to have a plan. Furthermore, the approximation results will be non-constructive since we do not have access to any plans solving the given instance—we only know that there exist at least one valid plan.

Theorem 7. *LOP$(*, *+, sat)$ can be nc-approximated within $|V|$ and COP$(*, *+, sat)$ can be nc-approximated within $|V|\mathrm{w}_{\max}(A)$.*

Proof. Every $(*, *+, sat)$ instance has a solution ω of length at most $|V|$ that contains at most $|A|$ actions by Corollary 4. The proof is thus analogous to the proof of Theorem 6. The actual plan ω is unknown so this is a non-constructive approximation result. □

3. Free Preconditions

The goal of this section is to analyse the approximability of COP$(*, *+)$ and LOP$(*, *+)$. We begin with the COP$(*, *+)$ problem in Section 3.1. We find that the approximability of COP$(*, *+)$ is strongly correlated to $\mathrm{w}_{\max}(A)$, i.e. the weight of the heaviest action. The proof for this result is not directly applicable to LOP$(*, *+)$ so we analyse LOP$(*, *+)$ in Section 3.2 and present an alternative proof.

The basis for the results in this section is the following variant of 3-SAT, which is well-known to be **NP**-complete (Berman, Karpinski, & Scott, 2003, Theorem 1).

(3,B2)-SAT
Type: Decision problem.
Instance: A propositional formula φ in conjunctive normal form such that (1) every clause of φ has exactly three distinct literals and (2) every literal occurs in exactly two clauses.
Question: Is φ satisfiable?

We assume without loss of generality that F contains no repeated clauses and that n is a multiple of 3. This implies that a (3,B2)-SAT formula F with n variables contains exactly $4n/3$ clauses since each variable give rise to four literal occurrences—two negated and two unnegated.

3.1 Cost-Optimal Planning

Aghighi et al. (2016) have proved that it is **NP**-hard to nc-approximate $\mathsf{COP}(3, 2+, sat)$ within $2^{|V|^p}$ for every $p \geq 0$. The proof by Aghighi et al. is based on a reduction from the directed Hamiltonian path problem and it is not clear how to generalise it to the $\mathsf{COP}(1, 1+, sat)$ problem. In response to this, we present an almost equally strong non-approximability result for $\mathsf{COP}(1, 1+, sat)$ in this section. The heart of our proof is the following construction.

Lemma 8. *For every integer $n > 0$, every (3,B2)-SAT formula F with n variables and every function $f : \mathbb{N}^+ \to \mathbb{N}^+$, there exists a Strips(1,1+,sat) instance $\mathbb{P}_{F,f} = \langle V, A, I, G\rangle$ such that*

(a) $\mathbb{P}_{F,f}$ can be computed in polynomial time in the size of F if $f(n) \in O(2^{n^p})$ for some fixed constant p,

(b) $|V| = K \cdot n$ where $K = 2 + 4/3$,

(c) $\mathbb{P}_{F,f}$ has a minimal weight plan with weight $\leq n + 4n/3$ if F is satisfiable,

(d) every minimal weight plan for $\mathbb{P}_{F,f}$ has weight $\geq f(K \cdot n)$ if F is not satisfiable, and

(e) the weights in $\mathbb{P}_{F,f}$ are bounded from above by $f(K \cdot n)$.

Proof. Let F be a (3,B2)-SAT formula over the variables $x_1, \ldots, x_n$ and clauses $C_1 \ldots C_m$ (where $m = 4n/3$). Let ℓ_i^j denote the j:th literal in clause C_i. Define the Strips(1,1+,sat) instance $\mathbb{P}_{F,f} = \langle V, A, \varnothing, G\rangle$ such that

- $V = \{x_1^t, \ldots, x_n^t, x_1^f, \ldots, x_n^f, c_1, \ldots, c_m\}$,
- $G = \{c_1, \ldots, c_m\}$,

and A contains the following actions:

- $true_i\colon \{\overline{x}_i^f\} \Rightarrow \{x_i^t\}$, for all i $(1 \leq i \leq n)$
- $false_i\colon \{\overline{x}_i^t\} \Rightarrow \{x_i^f\}$, for all i $(1 \leq i \leq n)$
- $vfy_i^j\colon \begin{cases} \{x_q^t\} \Rightarrow \{c_i\}, & \text{if } \ell_i^j = x_q \\ \{x_q^f\} \Rightarrow \{c_i\}, & \text{if } \ell_i^j = \overline{x_q} \end{cases}$, for all i, j $(1 \leq i \leq m,\ 1 \leq j \leq 3)$
- $sat_i\colon \varnothing \overset{f(K\cdot n)}{\Rightarrow} \{c_i\}$, for all i $(1 \leq i \leq m)$.

Proofs of the cases in the lemma follow:

(a) The weights for the sat_i actions need roughly n^p bits each to be written down. With this in mind, it is straightforward to verify that $\mathbb{P}_{F,f}$ can be computed in polynomial time.

(b) Obvious from the construction.

(c) Assume F is satisfiable and let $g : \{x_1, \ldots, x_n\} \to \{0, 1\}$ be a satisfying assignment. A shortest plan for $\mathbb{P}_{F,f}$ must contain at most one of the actions true_i and false_i for each variable x_i (depending on the satisfying assignment g), and exactly one of the actions $vfy_i^1, \ldots, vfy_i^3$ for each clause C_i (depending on g and C_i). This plan is of length at most $m + n \leq 4n/3 + n$, since $m = 4n/3$.

(d) Assume that F is not satisfiable. Then at least one goal variable c_i cannot be set to true by using a vfy_i^j action. Since c_i must be true in the goal, the plan must include action set_i, which has weight $f(K \cdot n)$.

(e) The sat_i actions have weight $f(K \cdot n)$ while all other actions have weight 1. □

Let $\beta : \mathbb{N} \to \mathbb{N}$ be a function and let $\mathbb{I}_\beta$ denote the set of $\mathsf{COP}(*, *+, sat)$ instances such that if $\langle V, A, I, G\rangle \in \mathbb{I}_\beta$, then $\mathrm{w}_{\max}(A) \leq \beta(|V|)$.

Theorem 9. *Arbitrarily choose some integer $k \geq 0$ and a constant $\varepsilon > 0$. Let $K = 2 + 4/3$ be the constant from Lemma 8. Let $\beta : \mathbb{N} \to \mathbb{N}$ denote a polynomial-time computable and non-decreasing function such that $(7n/3 + \varepsilon) \leq \beta(n) \leq 2^{n^k}$ for all $n \in \mathbb{N}$. Then, the following hold:*

1. *$\mathsf{COP}(\mathbb{I}_\beta)$ can be nc-approximated in polynomial time within $|V| \cdot \beta(|V|)$ and*
2. *$\mathsf{COP}(\mathbb{I}_\beta)$ is **NP**-hard to nc-approximate within $\beta(|V|/K)/(7|V|/3 + \varepsilon)$, even if restricted to $(1, 1+, sat)$ instances.*

Proof. The approximability bound follows directly from Theorem 7 so we concentrate on the non-approximability bound. We simplify the notation by letting $\beta'(x) = \beta(x/K)$ and then prove non-approximability within $\beta'(|V|)/(7|V|/3 + \varepsilon)$. The proof is by exhibiting a polynomial-time reduction from (3,B2)-SAT. Suppose $\mathsf{COP}(\mathbb{I}_\beta)$ can be nc-approximated within $\beta'(|V|)/(7|V|/3+\varepsilon)$ for some $\varepsilon > 0$ by algorithm $\mathcal{A}$. Let F be an arbitrary (3,B2)-SAT formula with n variables and $m = 4n/3$ clauses, and assume without loss of generality that n/K is an integer. Compute the $\mathsf{COP}(1, 1+, sat)$ instance $\mathbb{P}_{F,\beta'} = \langle V, A, I, G\rangle$ according to Lemma 8. This can be done in polynomial time by Lemma 8(a) since we require that $\beta(n) \leq 2^{n^k}$ for some fixed k and this holds for β', too, since β is required to be non-decreasing. The largest possible weight in $\mathbb{P}_{F,\beta'}$ is $\beta'(|V| \cdot K) = \beta(|V|)$ by Lemma 8(e) so $\mathbb{P}_{F,\beta'}$ is a member of $\mathbb{I}_\beta$.

Let OPT denote the optimal plan weight for $\mathbb{P}_{F,\beta'}$ and let W be the weight reported by algorithm $\mathcal{A}$. First suppose that F is satisfiable. We then know from Lemma 8(c) that $OPT \leq 4n/3 + n = 7n/3$. Hence, we get

$$W \leq \frac{\beta'(|V|)}{7|V|/3 + \varepsilon} \cdot OPT \ \leq \frac{\beta'(|V|)}{7|V|/3 + \varepsilon} \cdot \frac{7n}{3}.$$

Suppose instead that F is not satisfiable: it follows from Lemma 8(d) that $\beta'(K \cdot n) \leq OPT \leq W$. We know that $|V| = K \cdot n$ by Lemma 8(b) and consequently that $|V| > n$ since

$K > 1$. This implies that

$$\frac{\beta'(|V|)}{7|V|/3+\varepsilon} \cdot \frac{7n}{3} \leq \frac{\beta'(|V|)}{7n/3+\varepsilon} \cdot \frac{7n}{3} < \beta'(|V|) = \beta'(K \cdot n) \leq OPT \leq W.$$

We can consequently check whether F is satisfiable by nc-approximating $\mathbb{P}_{F,\beta'}$ within the bound, computing $\beta'(|V|)$, and comparing W with $\frac{\beta'(|V|)}{7|V|/3+\varepsilon} \cdot \frac{7n}{3}$. Recall that the function β is required to be polynomial-time computable and that this implies polynomial-time computability of β', too. It follows that $\mathbf{NP} \subseteq \mathbf{P}^{\mathcal{A}}$. □

It is only Strips$(1, 1+, sat)$ instances that are constructed in Lemma 8 so the previous bound holds when restricted to such instances. We remind the reader that the previous theorem is only applicable if β is at least linearly increasing: $\beta(|V|)/(7|V|/3+\varepsilon) \geq 1$ must hold since, otherwise, the approximation algorithm is required to produce a solution that is *strictly better* than the optimal solution. This explains why Theorem 9 is not applicable to the LOP problem and that we need to present a slightly different proof for LOP.

3.2 Length-Optimal Planning

We continue by proving that LOP$(1, 1+, sat)$ is not approximable within $|V|^{1-\varepsilon}$ for any $\varepsilon > 0$. The outline of the proof is identical to the proof for COP$(1, 1+, sat)$ but it is based on a slightly more involved construction.

Lemma 10. *For every (3,B2)-SAT formula F with n variables and every integer $\gamma \geq 2$, there exists a Strips(1,1+,sat) instance $\mathbb{P}_{F,\gamma} = \langle V, A, I, G\rangle$ such that*

(a) $\mathbb{P}_{F,\gamma}$ can be computed in polynomial time in the size of F;

(b) $|V| = n^\gamma + 4n/3 + 2n$

(c) $\mathbb{P}_{F,\gamma}$ has a shortest plan of length $\leq 4n/3 + n$ if F is satisfiable and

(d) every plan for $\mathbb{P}_{F,\gamma}$ has length $\geq n^\gamma$ if F is not satisfiable.

Proof. Let F be a (3,B2)-SAT formula over the variables $x_1, \ldots, x_n$ and clauses $C_1 \ldots C_m$ where $m = 4n/3$. Let ℓ_i^j denote the j:th literal in clause C_i. Arbitrarily choose an integer $\gamma \geq 2$ and let $h = n^\gamma$.

We construct the Strips$(1, 1+, sat)$ instance $\mathbb{P}_{F,\gamma} = \langle V, A, \varnothing, G\rangle$ as follows. Let

- $V = \{x_1^t, \ldots, x_n^t, x_1^f, \ldots, x_n^f, c_1, \ldots, c_m, y_1, \ldots, y_h\}$,
- $G = \{c_1, \ldots, c_m\}$,

and let A contain the following actions:

- $true_i$: $\{\overline{x}_i^f\} \Rightarrow \{x_i^t\}$, for all i $(1 \leq i \leq n)$
- $false_i$: $\{\overline{x}_i^t\} \Rightarrow \{x_i^f\}$, for all i $(1 \leq i \leq n)$
- vfy_i^j: $\begin{cases} \{x_q^t\} \Rightarrow \{c_i\}, & \text{if } \ell_i^j = x_q \\ \{x_q^f\} \Rightarrow \{c_i\}, & \text{if } \ell_i^j = \overline{x_q} \end{cases}$, for all i, j $(1 \leq i \leq m,\ 1 \leq j \leq 3)$

- $count_1 \colon \varnothing \Rightarrow \{y_1\}$
- $count_i \colon \{y_{i-1}\} \Rightarrow \{y_i\}$, for all i $(2 \leq i \leq h)$
- $sat_i \colon \{y_h\} \Rightarrow \{c_i\}$, for all i $(1 \leq i \leq m)$.

Proofs of the cases in the theorem follow:

(a) $\mathbb{P}_{F,\gamma}$ can obviously be computed in polynomial time.

(b) We see that $|V| = h + 2n + m$ so the bounds follow by observing that $h = n^\gamma$ and $m \leq 4n/3$.

(c) Assume F is satisfiable and let $f : \{x_1, \ldots, x_n\} \to \{0, 1\}$ be a satisfying assignment. A shortest plan for $\mathbb{P}_F$ must contain at most one of the actions true_i and false_i, depending on f, for each variable x_i, and exactly one of the actions $vfy_i^1, \ldots, vfy_i^3$ for each clause C_i, depending on C_i and f. This plan is of length at most $m + n \leq 4n/3 + n$, since $m = 4n/3$.

(d) Assume that F is not satisfiable. Then at least one goal variable c_i cannot be set to true by using a vfy_i^j action. Since c_i must be true in the goal, the plan must include action sat_i, which requires the plan to include all $count_i$ actions. Under that requirement, the plan is as short as possible only if it sets all goal atoms with *sat* actions. It follows that any shortest plan is of length $h + m = n^\gamma + m \geq n^\gamma$. □

Theorem 11. *$\mathsf{LOP}(*, *+)$ can be nc-approximated in polynomial time within $|V|$ and for every $\varepsilon > 0$, it is **NP**-hard to nc-approximate $\mathsf{LOP}(1, 1+, sat)$ within $|V|^{1-\varepsilon}$.*

Proof. The approximability bound follows from Theorem 7. The hardness proof is by a polynomial-time reduction from (3,B2)-SAT. Suppose $\mathsf{LOP}(1, 1+, sat)$ can be nc-approximated within $\alpha = |V|^{1-\varepsilon}$ for some $0 < \varepsilon < 1$ by an algorithm $\mathcal{A}$. Choose a constant integer $\gamma > 4/\varepsilon$.

Let F be an arbitrary (3,B2)-SAT formula with n variables. Construct a corresponding $\textsc{Strips}(1, 1+, sat)$ instance $\mathbb{P}_{F,\gamma} = \langle V, A, I, G \rangle$ according to Lemma 10. Let OPT denote the optimal plan length for $\mathbb{P}_F$ and let W be the weight reported by algorithm $\mathcal{A}$.

First suppose that F is satisfiable. We know from Lemma 10(c) that $OPT \leq 7n/3$ so $OPT \leq n^2$ when n is sufficiently large. We also know from Lemma 10(b) that $n^\gamma \leq |V|$. Hence,

$$W \leq 7n/3 \cdot |V|^{1-\varepsilon} \leq n^2 \cdot |V|^{1-\varepsilon} = (n^\gamma)^{2/\gamma} \cdot |V|^{1-\varepsilon} \leq |V|^{2/\gamma} \cdot |V|^{1-\varepsilon}.$$

We continue by supposing that F is not satisfiable. It follows from Lemma 10(d) that $n^\gamma \leq OPT \leq W$, and $|V| \leq n^\gamma + 4n/3 + 2n \leq 2n^\gamma$ holds for sufficiently large n since $\gamma \geq 2$. Hence, $|V|/2 \leq n^\gamma \leq OPT \leq W$ so it is sufficient to prove that

$$|V|^{1-\varepsilon} \cdot |V|^{2/\gamma} < \frac{|V|}{2}. \tag{1}$$

We see that $1 - \varepsilon + \frac{2}{\gamma} \leq 1 - \varepsilon + \frac{2}{4/\varepsilon} = 1 - \frac{\varepsilon}{2}$ so the lefthand side of (1) is bounded from above by $|V|^{1-\varepsilon/2}$. It follows that the inequality holds for sufficiently large $|V|$. From this point, the proof is analogous to the proof of Theorem 9. □

4. Positive Preconditions

The currently strongest non-approximability result for monotone planning restricted to positive preconditions has been proven by Aghighi et al. (2016).

Theorem 12. *It is **NP**-hard to approximate* $\mathsf{LOP}(0,*+)$ *within any function* α *such that* $\alpha(\langle V, A, s_I, s_G\rangle) = (1 - o(1)) \cdot \ln |V|$.

Note that this result holds for instances with *no* preconditions. For this (fairly uninteresting) class of instances, the result is basically tight: Aghighi et al. (2016) have demonstrated how to modify the $\ln n + 1$-approximation algorithm for Set Cover (Chvátal, 1979) into an $\ln n + 1$-approximation algorithm for $\mathsf{LOP}(0,*+)$. The restricted nature of $\mathsf{LOP}(0,*+)$—disallowing preconditions is a *severe* restriction—indicates that stronger approximation bounds are obtainable if we allow preconditions. We analyse such bounds in Section 4.1. In particular, we present a strong nonapproximability result for $\mathsf{LOP}(2+,1+)$. To describe the result, we define the function

$$g_c(n) = 2^{\log^{\left(1-\frac{1}{(\log\log n)^c}\right)} n}.$$

It is well-known, and not hard to verify, that g_c grows slower than all polynomial functions but faster than all polylogarithmic functions; concrete calculations can be found in the paper by Watel and Weisser (2016). Our result demonstrates that $\mathsf{LOP}(2+,1+)$ is **NP**-hard to approximate within $g_c(|V|)$ for all $c < 1/2$. Hence, our new result is substantially stronger than the bound presented by Aghighi et al. (2016). We note that we do not have any stronger lower bound for $\mathsf{COP}(2+,1+)$. It is not uncommon for these kind of problems to have the same approximability for both the unweighted and weighted version: for instance, Set Cover can be approximated within $\ln n + 1$ in both cases (Chvátal, 1979). Thus, it is not inconceivable that $\mathsf{LOP}(2+,1+)$ and $\mathsf{COP}(2+,1+)$ are indistinguishable from an approximability point of view.

These results raise several important questions such as the following: in what cases do we have strictly better bounds than the upper bounds in Theorem 6? This question appears to be very difficult to answer. It is not hard to see that there is a close connection between $\mathsf{LOP}(*+,1+)$ and the length of propositional Horn proofs; each action can basically be viewed as a Horn clause. Whether the minimal length of such proofs can be approximated within $n^{1-\varepsilon}$ for some $\varepsilon > 0$ or not is an important open question in theoretical computer science, and such a result would imply improved bounds for many computational problems (Alekhnovich et al., 2001; Dinur & Safra, 2004; Goldwasser & Motwani, 1999). Furthermore, the so-called *projection games conjecture* (Moshkovitz, 2015) is closely related to the approximability of these kinds of problems (via the Label Cover problem). More information about the difficulties of improving the (non)-approximability bounds of some of the problems that we use in the sequel can be found in the dissertation by Goldwasser (1997, Sec. 5.5.2).

In Section 4.2, we analyse problems where actions are allowed to have at most one precondition. This class of problems is partly motivated by work on the *local Steiner tree* heuristics (Keyder & Geffner, 2009). Our results are two-fold: $\mathsf{LOP}(1+,1+)$ is not approximable within $\log^{2-\varepsilon} |V|$ for any $\varepsilon > 0$ (and this result still beats Theorem 12) but

$\mathsf{COP}(1+,*+)$ is approximable within $O(|V|^{\varepsilon})$ for arbitrary constant $\varepsilon > 0$. $\mathsf{COP}(1+,*+)$ is our first example of a natural class of planning instances (besides the precondition-free class of $\textsc{Strips}(0,*+)$ instances) where we improve upon the trivial upper bound given by Theorem 6.

4.1 Problems with Several Preconditions

The basis for our nonapproximability result is a scheduling problem. Similar problems have been utilised for studying planning problems earlier by, for instance, Keyder et al. (2010). We are not aware of any prior examples where they have been used for analysing the approximability of planning, though.

Minimum And/Or Scheduling (AOS)
Type: Minimization problem.
Instance: A set $T = \{t_1, \ldots, t_n\}$ of *tasks*, a set $P = \{P_1, \ldots, P_n\}$ of *predecessor task sets*, where $P_i \subseteq T$ for each $P_i \in P$, and a distinguished *goal task* t_g. Each task is labelled as either an *AND-task* or an *OR-task*.
Solution: A set $\{t_g\} \subseteq T' \subseteq T$ and a total order $<$ on T' such that for all $t_i \in T'$:
1) if t_i is an AND-task, then $t_j \in T'$ and $t_j < t_i$ for all $t_j \in P_i$;
2) if t_i is an OR-task, then $t_j \in T'$ and $t_j < t_i$ for at least one $t_j \in P_i$;
3) if $t_i \neq t_g$ then $t_g \not< t_i$.
Measure: The size $|T'|$ of T'.

Note that the predecessor task sets may be empty. In fact, every solvable instance of AOS contains at least one task with an empty predecessor task set.

Theorem 13. *AOS is* ***NP****-hard to approximate within $g_c(|T|)$ for arbitrary $c < 1/2$, even if $|P_i| \leq 2$ for all $P_i \in P$.*

Proof. Goldwasser and Motwani (1999, Theorem 12) derived a nonapproximability result for this problem based on a one-to-one correspondence between AOS and the Label Cover problem. Dinur and Safra (2004) later proved that it is **NP**-hard to approximate Label Cover within $g_c(n)$ for all $c < 1/2$ so this result immediately carries over to AOS (Dinur and Safra (2004) mention explicitly this as one of several consequences of their result.) □

The AOS result provides a fairly direct way of proving non-approximability of $\mathsf{LOP}(2+,1+)$.

Theorem 14. *The following hold:*

1. *$\mathsf{COP}(*+,*+)$ can be approximated in polynomial time within $|V| \cdot \mathrm{w}_{\max}(A)$,*
2. *$\mathsf{LOP}(*+,*+)$ can be approximated in polynomial time within $|V|$, and*
3. *it is* ***NP****-hard to approximate $\mathsf{LOP}(2+,1+)$ within $g_c(|V|)$ for arbitrary $c < 1/2$.*

Proof. The approximability results (1) and (2) are immediate from Theorem 6. To prove the nonapproximability result (3), let $\mathbb{A} = \langle T, P, t_g \rangle$ be an arbitrary AOS instance where $T = \{t_1, \ldots, t_n\}$ is a set of tasks with a set $P = \{P_1, \ldots, P_n\}$ of predecessor task sets

and where $t_g \in T$ is the goal task. We assume that $|P_i| \leq 2$ for all $P_i \in P$ since the non-approximability bound in Theorem 13 holds under this restriction. Construct a corresponding Strips(2+,1+) instance $\mathbb{P} = \langle V, A, \varnothing, \{t_g\}\rangle$, where $V = T$ and A is defined such that for each $t_i \in T$,

- if t_i is an AND-task, then A contains the action $a_i : P_i \Rightarrow \{t_i\}$, and
- if t_i is an OR-task, then A contains the action $a_i^k : \{t_k\} \Rightarrow \{t_i\}$ for each task $t_k \in P_i$.

Let ω be a shortest plan for $\mathbb{P}$. Then there is obviously some optimal schedule $\langle T', <\rangle$ for $\mathbb{A}$ such that ω contains one action for each task in T' and ω corresponds to a topological sorting of these tasks. We see that that $OPT(\mathbb{P}) = OPT(\mathbb{A})$ and the instance $\mathbb{P}$ can clearly be computed in polynomial time. It follows that approximating $\mathsf{LOP}(2+, 1+)$ within $g_c(|V|)$ is **NP**-hard since it is **NP**-hard to approximate within $g_c(|T|)$ by Theorem 13. □

4.2 Problems with One Precondition

The results in this section are based on *Steiner trees*. Steiner tree problems form a large and diverse family of well-studied combinatorial optimisation problems. One particularly well-known variant is the *Steiner tree problem in graphs*, where the input is an undirected graph with non-negative edge weights together with a subset of vertices (known as the *terminals*), and the task is to find a subtree of the graph that has minimum total weight and that contains all terminals. It is not hard to verify that if the instance contains exactly two terminals, the problem reduces to finding the shortest path between the terminals. Furthermore, if all vertices are terminals, then the problem is equivalent to the minimum spanning tree problem. The connection between this Steiner problem and planning is well-known: the Steiner tree problem in graphs can be polynomial-time transformed into the $\mathsf{COP}(1+, 1+)$ problem. The details of this transformation can be found in the paper by Keyder and Geffner (2009).

The Steiner tree problem can be approximated within 1.39 (Byrka et al., 2013) so it is not very useful for proving strong nonapproximability results. We will instead use the Directed Steiner Tree problem—this allows us to prove a $\log^{2-\varepsilon} |V|$ bound for the $\mathsf{LOP}(1+, 1+)$ and $\mathsf{COP}(1+, 1+)$ problems. First recall that an *arborescence* is a directed graph that has a distinguished root vertex u and where there is exactly one directed path from u to any other vertex v.

Directed Steiner Tree (DST)
Type: Minimization problem.
Instance: A directed graph $G = \langle V, E\rangle$, a *root vertex* $r \in V$ and a set $Z \subseteq V$ of *terminals*. Each edge $e \in E$ has a non-negative weight $\mathrm{w}(e)$.
Solution: A subgraph $T = \langle V_T, E_T\rangle$ of G that is an arborescence with r as root such that $Z \subseteq V_T$.
Measure: The sum $\mathrm{w}(T) = \sum_{e \in E_T} \mathrm{w}(e)$ of edge weights in E_T.

Let $\mathbf{ZTIME}(f)$ denote the set of problems that can be solved by a probabilistic algorithm with expected runtime $f(n)$ and zero error probability.

Theorem 15 (Halperin and Krauthgamer (2003))**.** *The* DST *problem is not approximable within* $\log^{2-\varepsilon} n$ *for any* $\varepsilon > 0$*, unless* $\boldsymbol{NP} \subseteq \boldsymbol{ZTIME}(n^{polylog(n)})$*, where* $n = |V|$*.*

It is considered unlikely that $\mathbf{NP} \subseteq \mathbf{ZTIME}(n^{polylog(n)})$; an immediate consequence is that SAT can be solved in subexponential time by a randomised algorithm and, consequently, the randomised exponential time hypothesis is false. We will consider a variant of DST in the sequel: define HKDST to be the restriction of DST where $Z = V$ and $\mathrm{w}(e)$ is an integer such that $1 \leq \mathrm{w}(e) \leq |V|$.

Lemma 16. *The* HKDST *problem is not approximable within* $\log^{2-\varepsilon} n$ *for any* $\varepsilon > 0$*, unless* $\boldsymbol{NP} \subseteq \boldsymbol{ZTIME}(n^{polylog(n)})$*, where* $n = |V|$

Proof. The proof of the non-approximability result for DST by Halperin and Krauthgamer (2003) is based on input graphs that are complete trees of some arity d and height H, that is, the number of vertices is $n = \sum_{i=0}^{H} d^i \geq d^H$. All edge weights are on the form $1/(c^h)$ where c and h are integers such that $2 \leq c \leq d$ and $1 \leq h \leq H$. This can be scaled to integers between 1 and c^H, but $c^H \leq d^H < n$ so $\mathrm{w}(e) \leq n$ for all edges e. Halperin and Krauthgamer furthermore assume $Z = V$ and their proof for DST thus holds also for HKDST. □

4.2.1 Chain Reduction

We will now introduce a simple general method, *chain reduction*, for translating weighted Strips instances into unweighted instances in a solution-preserving way. The idea is similar to the count actions in Lemma 10 but there are important differences. We may view the count actions as simulating actions with weight h. Unfortunately, we need to simulate many actions—we would have m count actions of weight h where m is the number of clauses in formula F. By using chain reduction, we would end up with $h \cdot m$ actions and $h \cdot m$ variables. The solution in Lemma 10 is more economical since it only introduces h actions and h variables (and this is apparently a prerequisite for the proof of Theorem 11 to go through). However, the method in Lemma 10 is not generally applicable.

Definition 17. Let $\mathbb{P} = \langle V, A, I, G \rangle$ be a Strips(*,*+) instance. For each $a \in A$, define the sets V_a and A_a as follows:

- If $w(a) = 1$, then $V_a = \varnothing$ and $A_a = \{a^1\}$, where $a^1 \colon \mathrm{pre}(a) \Rightarrow \mathrm{eff}(a)$.
- Otherwise, let $w = \mathrm{w}(a)$ and define $V_a = \{v_a^1, \ldots, v_a^{w-1}\}$ and $A_a = \{a^1, \ldots, a^w\}$ where
  $a^1 \colon \{v_a^1\} \Rightarrow \mathrm{eff}(a)$
  $a^i \colon \{v_a^{i+1}\} \Rightarrow \{v_a^i\}$ for all i $(1 < i < w)$ and
  $a^w \colon \mathrm{pre}(a) \Rightarrow \{v_a^{w-1}\}$

Also define the sets $V_R = V \cup (\cup_{a \in A} V_a)$ and $A_R = \cup_{a \in A} A_a$. The *chain reduction* of $\mathbb{P}$ is the Strips instance $\mathbb{P}_R = \langle V_R, A_R, I, G \rangle$.

Intuitively, the chain reduction removes every action a with weight $w > 1$ and replace it with a set of w actions where each action has weight 1, and this is done in a solution-preserving way. One should observe that if $\mathbb{P}$ is a Strips(1+, 1+) instance, then so is the chain reduction of $\mathbb{P}$; this fact will be important later on. The size of a chain reduction can easily be inferred from the definition.

Proposition 18. *Let $\mathbb{P} = \langle V, A, I, G\rangle$ be an arbitrary STRIPS(*,*+) instance and let $\mathbb{P}_R = \langle V_R, A_R, I, G\rangle$ be the chain reduction of $\mathbb{P}$. Then,*

(a) $|V_R| = |V| + \mathrm{w}(A) - |A|$ *and*

(b) $|A_R| = \mathrm{w}(A)$.

The action weights can, in principle, be arbitrarily large and this implies that the size of a chain reduction can be arbitrarily large, too. This motivates the next definition.

Definition 19. A class X of STRIPS instances is *polynomially bounded* if there exists a polynomial p such that $\mathrm{w}(A) \leq p(||\mathbb{P}||)$ for every instance $\mathbb{P} = \langle V, A, I, G\rangle$ in X.

For polynomially bounded subclasses of STRIPS chain reductions behave reasonably well.

Lemma 20. *Let X be a polynomially bounded subclass of STRIPS(*+,*+), let $\mathbb{P} = \langle V, A, I, G\rangle$ be an arbitrary instance in X, and let $\mathbb{P}_R = \langle V_R, A_R, I, G\rangle$ be the chain reduction of $\mathbb{P}$. Then:*

(a) $\mathbb{P}_R$ is computable in polynomial time in $||\mathbb{P}||$.

(b) $\mathbb{P}_R$ has a plan if and only if $\mathbb{P}$ has a plan.

(c) $OPT(\mathbb{P}_R) = OPT(\mathbb{P})$.

(d) If ω_R is an optimal plan for $\mathbb{P}_R$, then $|\omega_R| \leq \mathrm{w}(A)$.

(e) Given an optimal plan ω_R for $\mathbb{P}_R$, a plan ω for $\mathbb{P}$ such that $\mathrm{w}(\omega) \leq |\omega_R|$ can be computed in polynomial time.

Proof. (a) It is immediate from Definition 17 that $\mathbb{P}_R$ can be computed in polynomial time since $\mathbb{P}$ is polynomially bounded.

(b-c) Immediate from Definition 17.

(d) According to Lemma 3, we may assume that no action occurs more than once so $|\omega_R| \leq |A_R| = \mathrm{w}(A)$.

(e) Let ω_R be an optimal plan for $\mathbb{P}_R$. According to Lemma 3 no action needs to occur more than once, so we can remove all multiple occurences of actions. Each action b in ω_R belongs to $A_a \subseteq A_R$ for some $a \in A$. Obviously, b cannot contribute to the goal unless all actions in A_a are in ω_R. Furthermore, a^1 can occur in ω_R only if all actions in A_a occur in ω_R since ω_R is a plan. Hence, we can construct a plan ω for $\mathbb{P}$ as follows: Start with $\omega = \langle\rangle$ and step through ω_R from beginning to end. For each action b in ω_R, if $b = a^1$ for some $a \in A$, then add a to the end of ω. It follows from (2) that this is a polynomial time computation and it is obvious that $\mathrm{w}(\omega) \leq |\omega_R|$ since $|A_a| = \mathrm{w}(a)$ for every $a \in A$. □

(a) and (b) says that the construction is a polynomial-time reduction, (c) says that the reduction preserves optimal solutions and (d)-(e) says that solutions for $\mathbb{P}_R$ can be converted back to solutions for $\mathbb{P}$ in polynomial time.

4.2.2 Non-approximability of Strips(1+, ∗+) Instances

We will now prove non-approximability results for Strips(1+, ∗+). The basic result is that there is a polynomial-time reduction HKDST to COP(1+, 1+) that preserves optimal solutions (Lemma 21). This reduction proves almost directly that COP(1+, 1+) cannot be approximated within $\log^{2-\varepsilon}|V|$ for any $\varepsilon > 0$, unless $\mathbf{NP} \subseteq \mathbf{ZTIME}(n^{polylog(n)})$. This result is combined with chain reductions (in Theorem 22) for proving the same non-approximability bound for LOP(1+, 1+).

Lemma 21. *There is a polynomial reduction from HKDST to COP(1+, 1+) that preserves optimal solutions.*

Proof. Let D be an instance of HKDST with a directed graph $H = \langle V, E\rangle$, a root vertex $r \in V$ and the terminal set V as input. Define the corresponding (1+,1+) Strips instance $\mathbb{P}_D = \langle V, A, \{r\}, V\rangle$ where A contains the action $a_e : \{u\} \overset{\mathrm{w}(e)}{\Rightarrow} \{v\}$ for each edge $e = \langle u, v\rangle \in E$. We claim the following:

a) If ω is an optimal plan for $\mathbb{P}_D$, then D has a DST T such that $\mathrm{w}(T) = \mathrm{w}(\omega)$.

b) If T is an optimal DST for D, then $\mathbb{P}_D$ has a plan ω such that $\mathrm{w}(\omega) = \mathrm{w}(T)$.

c) The construction above is a polynomial reduction from HKDST to COP(1+, 1+).

Proofs of these claims follow:

a) Suppose $\omega = \langle a_1, \ldots, a_\ell\rangle$ is an optimal plan for $\mathbb{P}_D$ and $\langle s_0, \ldots, s_\ell\rangle$ is the corresponding state sequence from $s_0 = \{r\}$. Since $\mathbb{P}_D$ is a (1+,1+) instance and ω is optimal it follows from Lemma 3 that $s_{i-1} \subseteq s_i$ and $|s_i \setminus s_{i-1}| = 1$ for $1 \leq i \leq \ell$. Hence, there is an order $v_0, \ldots, v_\ell$ on the variables in V such that $v_0 = r$ and $\{v_i\} = s_i \setminus s_{i-1} = \mathrm{eff}(a_i)$ for $1 \leq i \leq \ell$. That is, $s_i = \{v_0, \ldots, v_i\}$ for $1 \leq i \leq \ell$. Let ω_i denote the prefix $\langle a_1, \ldots, a_i\rangle$ of ω. We claim that for each i such that $0 \leq i \leq \ell$, the graph H contains a DST $T_i = \langle s_i, E_i\rangle$ for the terminal set s_i such that $\mathrm{w}(T_i) = \mathrm{w}(\omega_i)$. Proof by induction over the length of plan prefixes:

Base case: $(i = 0)$ We have $\omega_0 = \langle\rangle$, so $\mathrm{w}(\omega_0) = 0$. Let $T_0 = \langle s_0, \varnothing\rangle = \langle\{r\}, \varnothing\rangle$, which is a DST for s_0. Obviously, $\mathrm{w}(T_0) = 0 = \mathrm{w}(\omega_0)$.

Induction: (i > 0) Suppose the claim holds for some k such that $0 \leq k < \ell$. Then there is a DST $T_k = \langle s_k, E_k\rangle$ for s_k such that $\mathrm{w}(T_k) = \mathrm{w}(\omega_k)$. Then consider action a_{k+1}. Obviously, $\mathrm{pre}(a_{k+1}) \subseteq s_k$ since ω is a plan. Furthermore, $\mathrm{pre}(a_{k+1}) = \{v_j\}$ for some $v_j \in V$ by definition of a_{k+1}, so it follows that $\mathrm{pre}(a_{k+1}) \subseteq s_j$ and that $j \leq k$ since $s_j = \{v_0, \ldots, v_j\} \subseteq \{v_0, \ldots, v_k\} = s_k$. We also know that $\mathrm{eff}(a_{k+1}) = \{v_{k+1}\}$, so it follows from the definition of a_{k+1} that $\langle v_j, v_{k+1}\rangle \in E$. Let $E_{k+1} = E_k \cup \{\langle v_j, v_{k+1}\rangle\}$ and $T_{k+1} = \langle s_{k+1}, E_{k+1}\rangle$. By assumption, $T_k = \langle s_k, E_k\rangle$ is a DST for s_k so T_{k+1} must be a DST for s_{k+1}. Finally, $\mathrm{w}(T_{k+1}) = \mathrm{w}(T_k) + \mathrm{w}(\langle v_j, v_{k+1}\rangle) = \mathrm{w}(\omega_k) + \mathrm{w}(a_{k+1}) = \mathrm{w}(\omega_{k+1})$.

It follows that the claim holds for all prefixes of ω, so H contains a DST $T = T_\ell$ such that $\mathrm{w}(T) = \mathrm{w}(T_\ell) = \mathrm{w}(\omega_\ell) = \mathrm{w}(\omega)$.

b) Suppose $T = \langle V, E_T\rangle$ is an optimal DST for D. Let $\nu = v_0, \ldots, v_\ell$ be a topological sorting of T, where $v_0 = r$ since T is a tree directed from the root r. Let $s_i = \{v_0, \ldots, v_i\}$ for $0 \leq i \leq \ell$. Formally speaking, the set s_i consists of vertices in T and it is not a state. However, there is an obvious one-to-one correspondence between sets of tree vertices and states so we can view such a set as a state whenever it is convenient. For each i such that $1 \leq i \leq \ell$, there is some $j < i$ such that $\langle v_j, v_i\rangle \in E_T$ since ν is a topological sorting of the tree T.

Hence, there is an action $a_i \in A$ such that $\mathrm{pre}(a_i) = \{v_j\}$ and $\mathrm{eff}(a_i) = \{v_i\}$, since $E_T \subseteq E$ by definition of DSTs. Furthermore, $\mathrm{pre}(a_i) \subseteq s_j$ so $s_j \subseteq s_{i-1}$ and $s_i = s_{i-1} \cup \mathrm{eff}(a_i)$. This holds for all i such that $1 \leq i \leq \ell$ so let $\omega = \langle a_1, \ldots, a_\ell \rangle$, which must then be a plan with state sequence $\langle s_0, \ldots, s_\ell \rangle$ from $s_0 = \{r\}$. It is immediate that $\mathrm{w}(\omega) = \mathrm{w}(T)$.

c) Instance $\mathbb{P}_D$ can be constructed from D in polynomial time and it follows from (a) that an optimal DST for D can be extracted from an optimal plan for $\mathbb{P}_D$ in polynomial time.

It follows that this construction is a polynomial reduction that preserves optimal solutions. □

Theorem 22. *$\mathsf{LOP}(1+,1+)$ and $\mathsf{COP}(1+,1+)$ cannot be approximated within $\log^{2-\varepsilon}|V|$ for any $\varepsilon > 0$, unless $\mathbf{NP} \subseteq \mathbf{ZTIME}(n^{polylog(n)})$.*

Proof. Assume $\mathbf{NP} \not\subseteq \mathbf{ZTIME}(n^{polylog(n)})$. We prove the COP case by a reduction from HKDST. Let $\varepsilon > 0$ be an arbitrary constant and suppose there is a polynomial-time algorithm $\mathcal{A}$ that approximates $\mathsf{COP}(1+,1+)$ within $\log^{2-\varepsilon}|V|$. Let $\mathbb{D}$ be an arbitrary instance of HKDST with input graph $H = \langle V, E \rangle$, root vertex r and terminal set V. Let $\mathbb{P}_D = \langle V, A, \{r\}, V \rangle$ be the corresponding Strips (1+,1+) instance according to the construction in Lemma 21. Since $\mathcal{A}$ can approximate $\mathsf{COP}(1+,1+)$ within $\log^{2-\varepsilon}|V|$, it can thus also approximate HKDST within $\log^{2-\varepsilon}|V|$. However, this contradicts Lemma 16. It follows that $\mathcal{A}$ cannot exist and, thus, that $\mathsf{COP}(1+,1+)$ cannot be approximated within $\log^{2-\varepsilon}|V|$.

Proof of the LOP case by reduction from the class X of COP instances resulting from the reduction from HKDST above. Let $\varepsilon > 0$ be an arbitrary constant and suppose there is a polynomial-time algorithm $\mathcal{A}'$ that approximates $\mathsf{LOP}(1+,1+)$ within $\log^{2-\varepsilon}|V|$. Let $\mathbb{P} = \langle V, A, \{r\}, V \rangle$ be an arbitrary instace of X and let $\mathbb{P}_R = \langle V_R, A_R, \{r\}, V \rangle$ be the chain reduction of $\mathbb{P}$. Note that $\mathbb{P}_R$ is a Strips$(1+,1+)$ instance since the instances in X are Strips$(1+,1+)$ instances. Without loss of generality, assume the graph H satisfies that $|E| < |V|^2$, since loops can be removed, and that $|E| \geq |V| - 1$, since $\mathbb{D}$ is otherwise trivially unsolvable. Then $\mathbb{P}$ satisfies that $|V| - 1 \leq |A| < |V|^2$ and $\mathrm{w}(A) \leq |A||V| < |V|^3$. Applying Proposition 18 then yields that

$$|V_R| = |V| + \mathrm{w}(A) - |A| \leq \mathrm{w}(A) + 1 \leq |V|^3.$$

By assumption, $\mathcal{A}'$ can approximate $\mathsf{LOP}(X)$ within $\log^{2-\varepsilon}|V_R|$ and the class X is polynomially bounded, so it follows from Lemma 20 that $\mathcal{A}'$ can be used together with the reduction to approximate $\mathsf{COP}(X)$ within $\log^{2-\varepsilon}|V|^3$. Furthermore,

$$\log^{2-\varepsilon}|V|^3 = 3^{2-\varepsilon}\log^{2-\varepsilon}|V| \leq 9\log^{2-\varepsilon}|V|$$

and there is some $\varepsilon' > 0$ such that $9\log^{2-\varepsilon}|V| \leq \log^{2-\varepsilon'}|V|$ for large $|V|$. However, then $\mathsf{COP}(X)$ can be approximated within $\log^{2-\varepsilon'}|V|$, which contradicts Theorem 15. It follows that $\mathcal{A}'$ cannot exist and, thus, that $\mathsf{LOP}(1+,1+)$ cannot be approximated within $\log^{2-\varepsilon}|V|$. □

We have seen that the difference between the approximability of $\mathsf{COP}(1, 1+, sat)$ and $\mathsf{LOP}(1, 1+, sat)$ is huge: the former is not approximable within an exponential bound (given that weights are sufficiently large) while the latter is approximable within a linear bound. We will see in the next section that there is no corresponding gap when comparing $\mathsf{COP}(1+, *+)$ and $\mathsf{LOP}(1+, *+)$ since both problems can be approximated within $|V|^{\varepsilon}$ for arbitrary $\varepsilon > 0$.

4.2.3 Approximability of Strips(1+, *+) Instances

The goal of this section is to prove that $\mathsf{COP}(1+, *+)$ is approximable within $|V|^{\varepsilon}$ for arbitrary $\varepsilon > 0$. Once again, the proof is based on Steiner trees; the result is proven by a reduction to the Directed Group Steiner Tree problem (Charikar et al., 1999).

Directed Group Steiner Tree (DGST)
Type: Minimization problem.
Instance: A directed graph $G = \langle V, E\rangle$, a root vertex $r \in V$ and a set $C = \{g_1, \ldots, g_k\}$, where $g_i \subseteq V$ for all i $(1 \leq i \leq k)$. The g_i sets are called *groups*. Each edge $e \in E$ has a non-negative weight $\mathrm{w}(e)$.
Solution: A subgraph $T = \langle V_T, E_T\rangle$ of G that is an arborescence with r as root and that contains at least one vertex from each group in C.
Measure: The sum $\mathrm{w}(T)$ of edge weights in E_T.

Given a Strips instance $\mathbb{P} = \langle V, A, I, G\rangle$ and $v \in V$, we let $(A)_v$ denote the actions in A such that $v \in \mathrm{eff}(a)$. Furthermore, we let $\mathrm{eff}(A') = \{v \in \mathrm{eff}(a) \mid a \in A'\}$ for arbitrary $A' \subseteq A$.

Theorem 23. *$\mathsf{COP}(1+, *+)$ can be approximated within $O(|G \setminus I|^{\varepsilon})$ (and consequently within $O(|V|^{\varepsilon})$) for arbitrary $\varepsilon > 0$.*

Proof. Charikar et al. (1999) have proved that DGST is approximable within $O(|C|^{\varepsilon})$ for any $\varepsilon > 0$. It is thus sufficient to prove that there is a polynomial-time reduction from $\mathsf{COP}(1+,{}^*+)$ to DGST that preserves optimal solutions. We prove this by a reduction in two steps. The first step enables us to always use the initial state $\varnothing$—this simplifies the reduction to DGST that is carried out in the second step.

Let $\mathbb{P} = \langle V, A, I, G\rangle$ be an arbitrary Strips$(1+, *+)$ instance. For the first step, construct the corresponding Strips instance $\mathbb{P}_I = \langle V, A_I, \varnothing, G\rangle$, where $A_I = A \cup \{a_I\}$ and a_I is defined as $a_I \colon \varnothing \overset{0}{\Rightarrow} I$. We have previously excluded zero weights from Strips instances but we make an exception here for improved readability. Note that a zero weight is used only for the new action that is not present in the original action set A. Clearly, an action sequence $\langle a_1, \ldots, a_\ell\rangle$ is a plan for $\mathbb{P}$ if and only if $\langle a_I, a_1, \ldots, a_\ell\rangle$ is a plan for $\mathbb{P}_I$, and $\mathrm{w}(\langle a_I, a_1, \ldots, a_\ell\rangle) = \mathrm{w}(\langle a_1, \ldots, a_\ell\rangle)$.

For the second step, let $\mathbb{D}$ be a corresponding DGST instance defined by a directed graph $H = \langle A_I, E\rangle$, the root vertex a_I and a collection C of subsets of A_I defined as follows:

- $E = \{\langle a, b\rangle \mid a, b \in A_I \text{ and } \mathrm{pre}(b) \subseteq \mathrm{eff}(a)\}$, i.e. E contains the edge $\langle a, b\rangle$ for each pair of actions a, b such that a can provide the precondition for b.
- The weights on E are defined such that $\mathrm{w}(\langle a, b\rangle) = \mathrm{w}(b)$ for each edge $\langle a, b\rangle \in E$.

- $C = \{(A_I)_v \mid v \in G\}$, i.e. for each goal atom $v \in G$, a DGST for $\mathbb{D}$ must contain at least one vertex $a \in A_I$ such that $v \in \text{eff}(a)$.

In the sequel, we use the same convention as in the proof of Theorem 21: there is an one-to-one correspondence between sets of tree vertices and states, and we view sets of tree vertices as states whenever convenient. We claim that:

(a) If ω is an optimal plan for $\mathbb{P}_I$, then $\mathbb{D}$ has a DGST T such that $\text{w}(T) = \text{w}(\omega)$.

(b) If T is an optimal DGST for $\mathbb{D}$, then $\mathbb{P}_I$ has a plan ω such that $\text{w}(\omega) = \text{w}(T)$.

(c) $\mathbb{D}$ can be computed from $\mathbb{P}_I$ in polynomial time.

Proofs of these claims follow:

(a) Suppose $\omega = \langle a_1, \ldots, a_\ell \rangle$ is an optimal plan for $\mathbb{P}_I$ and $\langle s_0, \ldots, s_\ell \rangle$ is the state sequence for ω from $s_0 = \varnothing$. We can assume that $a_1 = a_I$ without losing generality. Let $A_T = \{a_1, \ldots, a_\ell\}$ and define the edge set E_T on A_T as follows. For each i, where $1 < i \leq \ell$, it must hold that $\text{pre}(a_i) \subseteq s_{i-1}$ since ω is a plan. Furthermore, $\mathbb{P}$ is a (1+,*+) instance so either $\text{pre}(a_i) = \varnothing$ or $\text{pre}(a_i) = \{v\}$. In the first case, $\text{pre}(a_i) \subseteq s_j$ for any j such that $0 \leq j \leq \ell$, so choose $j = 1$. In the second case, we know that $\text{pre}(a_i) \subseteq s_{i-1}$ but $\text{pre}(a_i) \not\subseteq s_0 = \varnothing$, so there is some smallest j such that $0 < j < i$, $v \in s_j$ and $v \in \text{eff}(a_j)$ (note here that monotonicity implies that once variable v is a member of state s_j, then s_j is a member of all succeeding states). In either case, let $\langle a_j, a_i \rangle \in E_T$. Now, let $T = \langle A_T, E_T \rangle$. There is obviously a path in T from a_1 to a_i for every i such that $1 < i \leq \ell$. However, for each such i, there is exactly one j such that $\langle a_j, a_i \rangle \in E_T$, so there is exactly one path from a_1 to each a_i, i.e. T is a directed tree with a_1 as root.

It further holds that $G \subseteq \text{eff}(\omega) = \text{eff}(A_T)$ since $s_0 = \varnothing$. Hence, for every $v \in G$, there is some $a \in A_T$ such that $v \in \text{eff}(a)$, i.e. $a \in (A_I)_v$. It follows that T is a DGST for D and it is immediate that $\text{w}(T) = \text{w}(A_T \setminus \{a_I\}) = \text{w}(A_T) = \text{w}(\omega)$.

(b) Suppose $T = \langle A_T, E_T \rangle$ is an optimal DGST for $D_{\mathbb{P}}$. Let $\omega_I = \langle a_1, \ldots, a_\ell \rangle$ be an arbitrary topological sorting of T. By definition, T is a directed tree with a_I as root, so $a_1 = a_I$. Furthermore, T is directed away from the root, so for every i, where $1 < i \leq \ell$, there is some j such that $1 \leq j < i$ and $\langle a_j, a_i \rangle$. However, T is a subgraph of H so $\langle a_j, a_i \rangle \in E$, i.e. $\text{pre}(a_i) \subseteq \text{eff}(a_j)$. Hence, $\text{pre}(a_i) \subseteq s_{i-1}$ according to Lemma 3. Furthermore, A_T contains at least one action $a \in g$ for each group $g \in C$, i.e. it contains at least one action $a \in A_v$ for every $v \in G$. Hence, $G \subseteq \text{eff}(A_T)$ and, thus, $G \subseteq s_\ell$. It follows that ω is a plan for $\mathbb{P}$. Obviously, $\text{w}(\omega) = \text{w}(A_T) = \text{w}(A_T \setminus \{a_I\}) = \text{w}(T)$.

(c) Immediate.

It follows that this construction is a polynomial reduction from $\mathsf{COP}(1+, *+)$ to DGST that preserves optimal solutions. □

5. Non-admissible Heuristics

We consider non-admissible heuristics in this section, i.e. heuristics that are allowed to overestimate the cost. We are consequently interested in heuristic functions h that satisfy

$$\frac{h^*(\mathbb{P}, s)}{c(|V|)} \leq h(\mathbb{P}, s) \leq d(|V|) \cdot h^*(\mathbb{P}, s)$$

for some functions c, d. Non-admissible heuristics are not very useful for cost-optimal planning but they are frequently used in satisficing and suboptimal planning—a well-known example is the non-admissible h^{FF} heuristic used in the FF planning system (Hoffmann & Nebel, 2001).

Given a LOP instance $\mathbb{P} = (V, A, I, G)$ and a function $p : \mathbb{N} \to \mathbb{N}$, let $\mathbb{P}^p$ denote a LOP instance consisting of the union of $p(|V|)$ disjoint copies of $\mathbb{P}$. Here, a disjoint copy $\mathbb{P}' = (V', A', I', G')$ of $\mathbb{P} = (V, A, I, G)$ is an instance where each variable has been renamed so that $V \cap V' = \varnothing$ and A, I, G have been modified accordingly. Given two planning instances $\mathbb{P} = (V, A, I, G)$ and $\mathbb{P}' = (V', A', I', G')$ such that $V \cap V' = \varnothing$, the union of $\mathbb{P}$ and $\mathbb{P}'$ is the planning instance $\mathbb{P}'' = (V'', A'', I'', G'')$ where $V'' = V \cup V'$, $A'' = A \cup A'$, the initial state I'' is defined such that it combines I and I' over the variable set I'', and G'' is defined analogously. Note the following:

1. if $\mathbb{P}$ is an $\mathsf{LOP}(a, b)$ instance, then $\mathbb{P}^p$ is an $\mathsf{LOP}(a, b)$ instance,
2. $\mathbb{P}^p$ contains $p(|V|) \cdot |V|$ variables,
3. if p is a polynomial, then $\mathbb{P}^p$ can be computed in polynomial time, and
4. $OPT(\mathbb{P}^p) = p(|V|) \cdot OPT(\mathbb{P})$.

Theorem 24. *Arbitrarily choose $0 < \varepsilon, \varepsilon' < 1$ such that $\varepsilon' + \varepsilon \cdot \varepsilon' + \varepsilon^3 < 1$. Assume there is a function h from $\mathsf{LOP}(1, 1+, sat)$ instances to the integers such that*

$$\frac{OPT(\mathbb{P})}{|V|^{\varepsilon}} \leq h(\mathbb{P}) \leq |V|^{\varepsilon'} \cdot OPT(\mathbb{P}).$$

Then h is not polynomial-time computable unless $\boldsymbol{P} = \boldsymbol{NP}$. In particular, if h satisfies

$$\frac{OPT(\mathbb{P})}{|V|^{\zeta}} \leq h(\mathbb{P}) \leq |V|^{1-\zeta} \cdot OPT(\mathbb{P})$$

for any $0 < \zeta < 1$, then h is not polynomial-time computable unless $\boldsymbol{P} = \boldsymbol{NP}$.

Proof. Let $p : \mathbb{N} \to \mathbb{R}$ denote the function $p(x) = x^{\varepsilon}$. Given a Strips instance $\mathbb{P}$, the instance $\mathbb{P}^p$ is, formally speaking, not defined since the defintion require the co-domain of p to contain integers only. To simplify the presentation of the proof, we carry on viewing p as a function from $\mathbb{N}$ to $\mathbb{R}$: we note that $0 \leq |p(x) - \lfloor p(x) \rfloor| < 1$ and this minimal difference is irrelevant for the proof.

Let $\mathbb{P} = \langle V, A, I, G \rangle$ denote an arbitrary $\mathsf{LOP}(1, 1+, sat)$ instance. We know that $\mathbb{P}^p$ contains $|V| \cdot |V|^{\varepsilon}$ variables so

$$\frac{OPT(\mathbb{P}^p)}{(|V| \cdot |V|^{\varepsilon})^{\varepsilon}} \leq h(\mathbb{P}^p) \leq (|V| \cdot |V|^{\varepsilon})^{\varepsilon'} \cdot OPT(\mathbb{P}^p) \Rightarrow$$

$$\frac{OPT(\mathbb{P}) \cdot |V|^{\varepsilon}}{(|V| \cdot |V|^{\varepsilon})^{\varepsilon}} \leq h(\mathbb{P}^p) \cdot |V|^{\varepsilon} \leq (|V| \cdot |V|^{\varepsilon})^{\varepsilon'} \cdot OPT(\mathbb{P}) \cdot |V|^{\varepsilon} \Rightarrow$$

$$\frac{OPT(\mathbb{P})}{(|V|^{\varepsilon})^{\varepsilon}} \leq h(\mathbb{P}^p) \cdot |V|^{\varepsilon} \leq (|V| \cdot |V|^{\varepsilon})^{\varepsilon'} \cdot OPT(\mathbb{P}) \cdot |V|^{\varepsilon} \Rightarrow$$

$$OPT(\mathbb{P}) \leq h(\mathbb{P}^p) \cdot |V|^{\varepsilon^3} \leq (|V| \cdot |V|^{\varepsilon})^{\varepsilon'} \cdot OPT(\mathbb{P}) \cdot |V|^{\varepsilon} \cdot |V|^{\varepsilon^2}$$

Recall that $\mathbb{P}^p$ is polynomial-time computable. Now, Theorem 11 implies that h is not polynomial-time computable since this would enable us to approximate $\mathsf{LOP}(1,1+)$ within $|V|^{\varepsilon'+\varepsilon\cdot\varepsilon'+\varepsilon^3}$ in polynomial time even though $\varepsilon' + \varepsilon \cdot \varepsilon' + \varepsilon^3 < 1$

For the second part of the theorem, we need to find out when $(1-\zeta)+\zeta(1-\zeta)+\zeta^3 < 1$. The inequality can be rewritten as $\zeta^3 < \zeta^2$ so it always holds when $0 < \zeta < 1$. □

Theorem 24 naturally carries over to less restricted problems such as $\mathsf{LOP}(*, *+, sat)$ and $\mathsf{COP}(*, *+, sat)$. If we restrict ourselves to positive preconditions, then we get slightly weaker results.

Theorem 25. *Assume there is a function h from $\mathsf{LOP}(2+, 1+)$ instances to the integers such that*

$$\frac{OPT(\mathbb{P})}{\log^k |V|} \leq h(\mathbb{P}) \leq g_c(|V|) \cdot OPT(\mathbb{P})$$

for some $k \geq 0$ and $c < 1/2$. Then h is not polynomial-time computable unless $\boldsymbol{P} = \boldsymbol{NP}$.

Proof. We use the same assumption as in the proof of Theorem 24: $\log^k |V|$ is not necessarily an integer but this causes no problem in the proof. We see that

$$\frac{OPT(\mathbb{P}^{\log^k |V|})}{\log^k(\log^k |V| \cdot |V|)} \leq h(\mathbb{P}^{\log^k |V|}) \leq g_c(|V|) \cdot OPT(\mathbb{P}^{\log^k |V|}) \Rightarrow$$

$$\frac{OPT(\mathbb{P}) \cdot \log^k |V|}{\log^k(\log^k |V| \cdot |V|)} \leq h(\mathbb{P}^{\log^k |V|}) \leq g_c(|V|) \cdot OPT(\mathbb{P}) \cdot \log^k |V| \Rightarrow$$

$$OPT(\mathbb{P}) \leq \frac{h(\mathbb{P}^{\log^k |V|}) \cdot \log^k(\log^k |V| \cdot |V|)}{\log^k |V|} \leq g_c(|V|) \cdot OPT(\mathbb{P}) \cdot \log^k(\log^k |V| \cdot |V|)$$

We know from the paper by Watel and Weisser (2016) that g_c grows faster than every polylogarithmic function. This implies that

$$OPT(\mathbb{P}) \leq \frac{h(\mathbb{P}^{\log^k |V|}) \cdot \log^k(\log^k |V| \cdot |V|)}{\log^k |V|} \leq g_{c'}(|V|) \cdot OPT(\mathbb{P})$$

when $c < c' < 1/2$ and the instances under consideration contains sufficiently many variables. Recall that $\mathbb{P}^{\log^k |V|}$ is polynomial-time computable when k is fixed. Now, Theorem 14 implies that h is not polynomial-time computable since this would enable us to approximate $\mathsf{LOP}(2+,1+)$ within $g_{c'}(|V|)$ in polynomial time.

□

6. Fixed-Parameter Tractability

The results in the previous sections indicate that bounding parameters like action weight and the number of preconditions and/or effects is not sufficient for obtaining polynomial-time approximation algorithms with good performance, and certainly not for identifying polynomial-time solvable cases. It is thus natural to consider other parameters that are more powerful. We consider two parameters δ and χ in this section. Let $\mathbb{P} = \langle V, A, I, G\rangle$ denote an arbitrary Strips instance.

Parameter δ. Let $\delta(\mathbb{P}) = \max_{v\in V} |\{a \in A \mid \text{eff}(a) \cap \{v, \overline{v}\} \neq \varnothing\}|$, i.e. δ is the maximum number of actions that affects a variable.

Parameter χ. Define $X(\mathbb{P}) = \{a, b \in A \mid a \neq b \text{ and } \text{eff}(a) \cap \text{eff}(b) \neq \varnothing\}$ and let $\chi(\mathbb{P}) = |X(\mathbb{P})|$.

The parameters measure, in different ways, how many actions in total that may cause combinatorial explosion during action selection. The intuitive difference between δ and χ can be viewed as follows: δ is a local property that is applicable directly on the variable level while χ is a global property that takes the full set of actions into account. Instances satisfying $\delta(\mathbb{P}) = 1$ or $\chi(\mathbb{P}) = 0$ are often referred to as *post-unique* or *establisher-unique* in the literature (Bäckström & Nebel, 1995; Cooper et al., 2014). We let $\text{Strips}(*, *, P)$ denote the full set of post-unique instances. Post-unique instances are known for having favourably computational properties (Bäckström & Nebel, 1995; Bäckström et al., 2015) and at the same time being sufficiently powerful for modelling certain interesting industrial applications: for instance, chemical and pharmaceutical examples are presented by Cooper et al. (2012, 2013). However, post-uniqueness often make modelling difficult and it is thus desirable to identify restrictions that are easier to work with—bounded δ or χ are obvious candidates.

The parameter δ has been considered previously in the literature, cf. the article by Bäckström et al. (2015). It is, for instance, known that $\mathsf{LOP}(*, *)$ is fixed-parameter tractable in the parameter δ and the plan length k—this problem can be solved in $O(((\delta \cdot k)^{k+1} + \delta)|V|)$ time. Plan length is a useful parameter in many contexts but it is, for obvious reasons, not particularly relevant when considering heuristics that are supposed to estimate plan length. Our first result (Section 6.1) shows that δ in isolation is not sufficient for fixed-parameter tractability: the problem $\mathsf{LOP}(1+, 1+)$ is **NP**-hard even when restricted to $\delta = 2$. We note that Bylander (1994, Corr 3.4) has proved that $\mathsf{LOP}(1+, 1+)$ is **NP**-hard but his proof seems difficult to generalise to the case when $\delta = 2$.

Our second result (Section 6.2) shows that χ is a better parameter for fixed-parameter tractability: we prove that $\mathsf{COP}(*, 1+)$ is solvable in $2^{\chi(\mathbb{P})} \cdot poly(||\mathbb{P}||)$ time. We recall that $\mathsf{LOP}(1+, 1+)$ is **NP**-hard so $\mathsf{COP}(*, 1+)$ is obviously an intractable problem. We remark that the result only holds for *unary* actions, i.e. actions that only affects one variable. Instances based on unary actions have a long and interesting history in planning: many tractable subclasses of planning only work for unary actions and many heuristic functions make a relaxation to unary actions—in particular, heuristics that exploit acyclic causal graphs (such as the original heuristic used by the FD planner (Helmert, 2006)) are inherently based on unary action relaxations. Finally, one should note that $\mathsf{LOP}(*, 1, P)$ is

NP-hard (Bäckström & Nebel, 1995) so monotonicity is an essential property: the result cannot be generalised to non-monotone planning even in the case when χ is fixed to 0.

6.1 The Parameter δ

We show that $\mathsf{LOP}(1+,1+)$ is **NP**-hard even if we only consider instances where the parameter δ is bounded by 2. The basis for this result is the Vertex Cover problem.

Vertex Cover
Type: Decision problem.
Instance: An undirected graph $G = (V, E)$ and an integer $K \geq 0$.
Question: Does there exist a set $V' \subseteq V$ such that $|V'| \leq K$ and for each $\{u, v\} \in E$, $\{u, v\} \cap V' \neq \varnothing$?

Theorem 26. *$\mathsf{LOP}(1+,1+)$ is **NP**-hard even when restricted to instances with $\delta = 2$.*

Proof. Let $\langle (V, E), K\rangle$ be an arbitrary instance of Vertex Cover where $V = \{v_1, \ldots, v_k\}$. Define the Strips instance $\mathbb{P} = \langle W, A, \varnothing, G\rangle$ such that

- $W = V \cup E$,
- $G = E$,

and A contains the following actions:

- $set_i \colon \varnothing \Rightarrow v_i$, for all i $(1 \leq i \leq n)$
- $e_i \colon \{v_i\} \Rightarrow \{e\}$ and $e_j \colon \{v_j\} \Rightarrow \{e\}$ for all $e = \{v_i, v_j\} \in E$.

Note that $\mathbb{P}$ is a $\textsc{Strips}(1+,1+)$ instance with $\delta = 2$. We claim that $\mathbb{P}$ has a solution of length at most $K + |E|$ if and only if $((V, E), K)$ is a yes-instance.

Assume first that $((V, E), K)$ is indeed a yes-instance with cover $V' = \{v_{i_1}, \ldots, v_{i_p}\}$. Then, $\langle \mathrm{set}_{i_1}, \mathrm{set}_{i_2}, \ldots, \mathrm{set}_{i_p}\rangle$ concatenated with a suitable sequence of $|E|$ e_k actions is a valid plan for $\mathbb{P}$. This plan has total length $p + |E| \leq K + |E|$.

Assume instead that $\mathbb{P}$ has a solution ω with length at most $K + |E|$. This implies that $p \leq K$ actions $\mathrm{set}_{i_1}, \ldots, \mathrm{set}_{i_p}$ are included in ω. Hence, $V' = \{v_{i_1}, \ldots, v_{i_p}\}$ is a vertex cover of G that contains at most $p \leq K$ vertices. □

6.2 The Parameter χ

We will now prove that $\mathsf{COP}(*, 1+)$ can be solved in $2^{\chi(\mathbb{P})} \cdot poly(||\mathbb{P}||)$ time and, consequently, that it admits a fixed-parameter tractable algorithm when considering parameter χ. The algorithm for COP(*,1+) is presented in Figure 4. It is a brute-force algorithm that generates all sets of actions that may be used in a plan, checks whether the corresponding instance is post-unique, and if so solves it using an algorithm for $\mathsf{COP}(*, 1+, P)$. We simplify the presentation of the algorithm and the forthcoming proofs by using the following convention: unsolvable instances are assigned the optimal value ∞. An immediate consequence of the $2^{\chi(\mathbb{P})} \cdot poly(||\mathbb{P}||)$ bound is that $\mathsf{COP}(*, 1+)$ restricted to instances with $\chi(\mathbb{P}) \leq k \log(|V|)$ can be solved in polynomial time for arbitrary fixed k. Another consequence is that $\mathsf{COP}(*, 1+)$

restricted to instances with $\chi(\mathbb{P}) \leq |V|^{1-\varepsilon}$ can be solved in subexponential time for arbitrary fixed $\varepsilon > 0$.

We begin by presenting a tractability result for $\mathsf{COP}(*, 1+, P)$. We postpone proving this lemma until Section 6.3.

Lemma 27. *$\mathsf{COP}(*, 1+, P)$ can be solved in polynomial time.*

Theorem 28. *$\mathsf{COP}(*, 1+)$ can be solved in $2^{\chi(\mathbb{P})} \cdot poly(||\mathbb{P}||)$ time.*

Proof. The algorithm presented in Figure 4 clearly meets the time complexity requirement since $OPT(\mathbb{P}')$ can be computed in polynomial time by Lemma 27.

Let $\mathbb{P} = \langle V, A, I, G\rangle$ be a STRIPS$(*, 1+)$ instance. If $OPT(\mathbb{P}) = \infty$, i.e. $\mathbb{P}$ has no solution, then $OPT(\langle V, A', I, G\rangle) = \infty$ for every $A' \subseteq A$. This implies that the algorithm will return the correct value ∞.

Assume instead that $OPT(\mathbb{P}) < \infty$ and suppose that $\omega = \langle a_1, \ldots, a_n\rangle$ is an optimal plan. Let $B = \{a_1, \ldots, a_n\}$. We claim that $\langle V, B, I, G\rangle$ is post-unique. To see this, we assume to the contrary that there are two distinct actions a_i, a_j in ω such that $\mathrm{eff}(a_i) \cap \mathrm{eff}(a_j) \neq \varnothing$. The actions a_i and a_j are unary so $\mathrm{eff}(a_1) = \mathrm{eff}(a_2)$. That both a_1 and a_2 occur in an optimal plan contradicts Lemma 3(c).

Every post-unique subset of A is enumerated in line 4 so, in particular, the set B will occur in this enumeration. We conclude,that the algorithm return the optimal value $OPT(\langle V, B, I, G\rangle)$ since c (the value that is computed in line 6) equals $\mathrm{w}(\omega)$. □

```
1  function χ-Plan(⟨V, A, I, G⟩)
2    let X := X(ℙ) and A' := A \ X
3    let c := ∞
4    for every X' ⊆ X s.t. ℙ' = ⟨V, A' ∪ X', I, G⟩ is post-unique
6      let c := min(c, OPT(ℙ'))
7    return c
```

Figure 4: Algorithm for solving COP(*,1+)

6.3 Tractability of $\mathsf{COP}(*, 1+, P)$

We will now prove that $\mathsf{COP}(*, 1+, P)$ can be solved in polynomial time (Lemma 27). The bulk of the proof is devoted to proving that $\mathsf{COP}(*, 1+, P)$ is polynomial-time equivalent to $\mathsf{LOP}(*, 1+, P)$. In the final step, we merely verify that the algorithm for SAS$^+$-IAO (Jonsson & Bäckström, 1998) is suitable for solving $\mathsf{LOP}(*, 1+, P)$.

Let $\mathbb{P} = \langle V, A, I, G\rangle$ be a $\mathsf{COP}(*, 1+, P)$ instance and define $v^+ = \{a \in A \mid v \in \mathrm{eff}(a)\}$. Since $\mathbb{P}$ is post-unique, it follows that $0 \leq |v^+| \leq 1$ for every $v \in V$. We use an auxiliary function NA (where NA stands for *necessary actions*) for giving the proof of Lemma 27 a clearer structure. The definition of NA can be found in Figure 5. Given an instance $\mathbb{P}$ of $\mathsf{COP}(*, 1+, P)$, NA is intended to generate a set of actions that must be members of *any* plan for $\mathbb{P}$. It achieves this by a fixed-point computation where actions are iteratively added to the set N. We prove correctness in the next lemma and we note that there are similarities with *landmarks* (Keyder et al., 2010).

1 **function** NA($\langle V, A, I, G\rangle$)
2 **if** there is a $v \in G$ such that $v \notin I$ and $v^+ = \varnothing$ **then reject**
3 $N := \bigcup\{v^+ \mid v \in G \text{ and } v \notin I\}$
4 $N' := \varnothing$
5 **while** $N \neq N'$ **do**
6 $N' := N$
7 **if** there exists $a \in N$ such that $\overline{v} \in \text{pre}(a)$ and $v \in I$ **then reject**
8 **if** there exists $a \in N$ such that $v \in \text{pre}(a)$, $v \notin I$ and $v^+ = \varnothing$ **then reject**
9 **if** there exists $a \in N$ such that $v \in \text{pre}(a)$, $v \notin I$ and $v^+ \neq \varnothing$ **then** $N := N \cup \{v^+\}$
10 **return** N

Figure 5: Greedy algorithm for computing necessary actions

Lemma 29. *If $\omega = \langle a_1, \ldots, a_m\rangle$ is a plan for the $\mathsf{COP}(*, 1+, P)$ instance $\mathbb{P} = \langle V, A, I, G\rangle$, then* NA *does not reject* $\mathbb{P}$ *and* $\text{NA}(\mathbb{P}) \subseteq \{a_1, \ldots, a_m\}$.

Proof. Let $B = \{a_1, \ldots, a_m\}$. The instance $\mathbb{P}$ is post-unique so $0 \leq |v^+| \leq 1$ for every $v \in V$. We prove the result by induction over the number p of loops in lines 5–9. Let N_i denote the set N after i loops have been performed. We prove that $N_0 \subseteq N_1 \subseteq \cdots \subseteq N_p \subseteq B$ and that NA does not reject during the computation of $N_1, \ldots, N_p$.

Base case. $p = 0$. First note that NA cannot reject $\mathbb{P}$ in line 2: $v^+ \neq \varnothing$ since ω is a solution to $\mathbb{P}$. Thus, the set N_0 has been properly computed. It is obvious that $N_0 \subseteq B$ since the actions in N are the only actions that can achieve the goals in $G \setminus I$ due to post-uniqueness.

Induction hypothesis. $N_{p-1} \subseteq B$.

Induction step. We show that $N_p \subseteq B$ by analysing lines 7-9.

Line 7. If there exists $a \in N_{p-1}$ such that $\overline{v} \in \text{pre}(a)$ and $v \in I$, then ω cannot be a valid plan for $\mathbb{P}$. We know that $a \in B$ by the induction hypothesis and we also know that no action can make the variable v false due to monotonicity. Thus, the algorithm cannot reject in this line.

Line 8. If there exists $a \in N_{p-1}$ such that $v \in \text{pre}(a)$, $v \notin I$ and $v^+ = \varnothing$, then ω cannot be a valid plan for $\mathbb{P}$: we know that $a \in B$ by the induction hypothesis but the preconditions of a cannot be achieved.

Line 9. If there exists $a \in N_{p-1}$ such that $v \in \text{pre}(a)$ and $v \notin I$, then there is exactly one action (due to post-uniqueness) in A that can achieve the precondition v and this action is contained in v^+, so $N_p = N_{p-1} \cup \{v^+\}$. Since $N_{p-1} \subseteq B$ by the induction hypothesis, it follows that $N_p \subseteq B$, too. □

Next, we verify that NA is a monotonic function with respect to the goal state.

Lemma 30. *Let $\mathbb{P} = \langle V, A, I, G\rangle$ be a $\mathsf{COP}(*, 1+, P)$ instance. If* NA *does not reject* $\mathbb{P}$, *then* $\text{NA}(\mathbb{P}) = \bigcup_{p \in G} \text{NA}(\langle V, A, I, \{p\}\rangle)$.

Proof. It is obvious from the algorithm that $\text{NA}(\mathbb{P}) \supseteq \text{NA}(\langle V, A, I, \{p\}\rangle)$ for every $p \in G$, and, consequently,

$$\text{NA}(\mathbb{P}) \supseteq \bigcup_{p \in G} \text{NA}(\langle V, A, I, \{p\}\rangle).$$

We prove that $\mathrm{NA}(\mathbb{P}) \subseteq \bigcup_{p \in G} \mathrm{NA}(\langle V, A, I, \{p\}\rangle)$. Let a' denote an arbitrary action in $\mathrm{NA}(\mathbb{P})$. We show inductively that there exists a $g \in G$ such that $a' \in \mathrm{NA}(\langle V, A, I, \{g\}\rangle)$, and this readily implies that $\mathrm{NA}(\mathbb{P}) \subseteq \bigcup_{p \in G} \mathrm{NA}(\langle V, A, I, \{p\}\rangle)$. We do this by induction over n where n is the number of loops the algorithm makes before a' is added to N.

Base case. $n = 0$. If a' is added to N in line 3, then there exists a goal g such that $a' \in \mathrm{NA}(\langle V, A, I, \{g\}\rangle)$ and $\{a'\} \subseteq \bigcup_{p \in G} \mathrm{NA}(\langle V, A, I, \{p\}\rangle)$.

Induction step. Assume the claim holds for $n = k$, $k \geq 0$. We show that it holds for $n = k + 1$, too. In this case, a' is added to N in line 9 of the algorithm. Then there exists an action $a \in N$ such that a' establishes one of a's preconditions, and a was added to N after at most k loops. The induction hypothesis implies that there exists a $g \in G$ such that $a \in \mathrm{NA}(\langle V, A, I, \{g\}\rangle)$. Consequently, $a' \in \mathrm{NA}(\langle V, A, I, \{g\}\rangle) \subseteq \bigcup_{p \in G} \mathrm{NA}(\langle V, A, I, \{p\}\rangle)$. □

We continue by strengthening Lemma 29 with the aid of Lemma 30: given an instance $\mathbb{P}$, we show a strong correspondence between the actions in $\mathrm{NA}(\mathbb{P})$ and the actions in a shortest plan for $\mathbb{P}$.

Lemma 31. *If $\omega = \langle a_1, \ldots, a_m\rangle$ is a shortest plan for the $\mathsf{COP}(*, 1+, P)$ instance $\mathbb{P} = \langle V, A, I, G\rangle$, then $\mathrm{NA}(\mathbb{P}) = \{a_1, \ldots, a_m\}$.*

Proof. We know from Lemma 29 that NA does not reject $\mathbb{P}$. We prove the claim by induction over the number of goals t.

Base case. If $t = 0$, then the plan is empty and $\mathrm{NA}(\mathbb{P}) = \varnothing$.

Induction hypothesis. Assume the hypothesis holds for $t = p - 1$, i.e. every shortest plan for an instance $\langle V, A, I, G\rangle$ with $|G| = p - 1$ contains exactly the actions in $NA(\langle V, A, I, G\rangle)$.

Induction step. Consider a shortest plan $\omega = \langle a_1, \ldots, a_m\rangle$ for $\mathbb{P} = \langle V, A, I, G\rangle$ where $|G| = p$. Let s denote the state that the final action a_m is applied to, and let $s' = s \cap G$. Exactly one goal $g \in G$ does not hold in s' since the final action in a shortest plan always sets a goal and we are restricted to unary actions. Post-uniqueness imply that there is only one action that can achieve this goal, namely a_m. In particular, this implies that $a_m \in NA(\langle V, A, I, \{g\}\rangle)$

By the induction hypothesis, every shortest plan for $\langle V, A, I, G \setminus \{g\}\rangle$ contain exactly the actions in $NA(\langle V, A, I, G \setminus \{g\}\rangle)$ so

$$\{a_1, \ldots, a_{m-1}\} = NA(\langle V, A, I, G \setminus \{g\}\rangle) \Rightarrow$$

$$\{a_1, \ldots, a_{m-1}, a_m\} = NA(\langle V, A, I, G \setminus \{g\}\rangle) \cup \{a_m\} \Rightarrow$$

$$\{a_1, \ldots, a_{m-1}, a_m\} \subseteq NA(\langle V, A, I, G \setminus \{g\}\rangle) \cup NA(\langle V, A, I, \{g\}\rangle$$

since a_m is the only action that achieves g. We apply Lemma 30 and obtain

$$\{a_1, \ldots, a_m\} \subseteq NA(\langle V, A, I, G\rangle).$$

We conclude that $\{a_1, \ldots, a_m\} = NA(\langle V, A, I, G\rangle)$ with the aid of Lemma 29. □

We finally put the pieces together and prove the tractability result.

Proof. (of Lemma 27) Let $\mathbb{P} = \langle V, A, I, G\rangle$ be an arbitrary instance of $\mathsf{COP}(*, 1+, P)$. Let $\omega = \langle a_1, \ldots, a_m\rangle$ be a shortest plan for $\mathbb{P}$ and let $\psi = \langle b_1, \ldots, b_n\rangle$ be a plan for $\mathbb{P}$ with minimal weight. We immediately see that $\mathrm{w}(\psi) \leq \mathrm{w}(\omega)$. By Lemmas 29 and 31, we additionally see that

$$\{a_1, \ldots, a_m\} = \mathrm{NA}(\mathbb{P}) \subseteq \{b_1, \ldots, b_n\}.$$

This implies that $\mathrm{w}(\omega) \leq \mathrm{w}(\psi)$ since negative weights are not allowed. We conclude that $\mathrm{w}(\omega) = \mathrm{w}(\psi)$ and that $\mathsf{COP}(*, 1+, P)$ can be solved by an algorithm for $\mathsf{LOP}(*, 1+, P)$.

The problem $\mathsf{LOP}(*, 1+, P)$ can be solved by the polynomial-time algorithm for SAS^+-IAO by Jonsson and Bäckström (1998). To verify that it is indeed applicable, we first recall that we assume the initial state to be a total state (in the sense that every variable has a definite value) and, consequently, we have no actions that can turn an "unknown" variable value into a definite value. We are thus within the SAS^* framework and are free to use Theorem 7.8 by Jonsson and Bäckström (1998). We see that post-unicity implies property O, unarity is property U, and monotonicity implies property A^+ which in turn implies property A. Hence, $\mathsf{COP}(*, 1+, P)$ can be solved in polynomial time. □

7. Discussion

We have studied computational properties of cost-optimal planning restricted to monotone instances. Our main motivation is to obtain a better understanding of heuristics used in search-based planning. The results reveal that approximating monotone planning is in general **NP**-hard even under very liberal approximation bounds. However, the results also reveal that the success of certain heuristics (such as local Steiner tree heuristics) can be attributed to the fact that the underlying restricted approximation problem is easier to approximate than the general problem. Most of our work is directed towards admissible heuristics but many results carry over without too much effort to non-admissible heuristics. We see that non-admissible heuristics seem to be slightly easier to approximate than their admissible counterparts but the difference must, unfortunately, be considered fairly small.

With this in mind, we conclude that the tools of classical complexity theory are not fully adequate for analysing these kind of problems: The coarse partitioning of problems into tractable and **NP**-hard is not satisfactory since most subcases end up being **NP**-hard. We remind the reader that the span in time complexity for **NP**-hard problems is, as far as we know, enormous: there are problems that can be solved in subexponential $2^{o(n)}$ time and there are problems where no better bound than 2^{n^c} (for large values of c) is known. What one really would like to have is a more fine-grained way of studying these problems.

In response to this, we suggest that parameterised complexity may be a better tool and we initiate a parameterised study of the complexity of monotone cost-optimal planning. We concentrate on two simple but natural parameters δ, χ based on the number of actions that can achieve a particular effect. The hypothesis is that they measure the amount of non-determinism needed in the action selection process and that this is correlated to the computational hardness of the underlying problem. We show that fixed-parameter tractability is obtained (in the special case of unary actions) when using χ but not δ. This emphasises that identifying the right parameters is crucial for obtaining strong bounds on

computational costs. Needless to say, the identification of *useful* parameters is a challenging research problem.

Another challenging research direction is to connect the performance of heuristic functions with their time complexity. If we simplify things slightly, then classical complexity will only tell us that a heuristic function can achieve an approximation bound α in polynomial time while computing solutions with better bounds is an **NP**-hard problem (which will require superpolynomial time to solve). The switch to the parameterised setting gives us the opportunity to view the time complexity of a heuristic function as a function of its performance. This is clearly much more informative than the simple dichotomy that classical complexity offers us. One may illuminate this idea by considering Theorem 28. It shows that an *exact* solution to the $\mathsf{COP}(*, 1+)$ problem can be found in $2^{\chi(\mathbb{P})} \cdot poly(||\mathbb{P}||)$ time. The natural question here is whether faster algorithms can be constructed if it is sufficient to find an α-approximate solution. If this is indeed the case, how does the time complexity of such algorithms depend on the choice of α?

Acknowledgements

We thank the reviewers for highly valuable input. Christer Bäckström is partially supported by the Swedish Research Council (VR) under grant 621-2014-4086. Peter Jonsson is partially supported by VR under grant 2017-04112.

References

Aghighi, M., Bäckström, C., Jonsson, P., & Ståhlberg, S. (2016). Analysing approximability and heuristics in planning using the exponential-time hypothesis. In *Proc. 22nd European Conference on Artificial Intelligence (ECAI-2016)*, pp. 184–192.

Alekhnovich, M., Buss, S. R., Moran, S., & Pitassi, T. (2001). Minimum propositional proof length is NP-hard to linearly approximate. *Journal of Symbolic Logic*, *66*(1), 171–191.

Bäckström, C., & Jonsson, P. (2017). Time and space bounds for planning. *Journal of Artificial Intelligence Research*, *60*, 595–638.

Bäckström, C., Jonsson, P., Ordyniak, S., & Szeider, S. (2015). A complete parameterized complexity analysis of bounded planning. *Journal of Computer and System Sciences*, *81*(7), 1311–1332.

Bäckström, C., & Nebel, B. (1995). Complexity results for SAS^+ planning. *Computational Intelligence*, *11*, 625–656.

Benton, J., Talamadupula, K., Eyerich, P., Mattmüller, R., & Kambhampati, S. (2010). G-value plateaus: A challenge for planning. In *Proc. 20th International Conference on Automated Planning and Scheduling (ICAPS-2010)*, pp. 259–262.

Berman, P., Karpinski, M., & Scott, A. D. (2003). Approximation hardness of short symmetric instances of MAX-3SAT. *Electronic Colloquium on Computational Complexity (ECCC)*, *10*(49).

Betz, C., & Helmert, M. (2009). Planning with h^+ in theory and practice. In *32nd Annual German Conference on AI (KI-2009)*, pp. 9–16.

Bylander, T. (1994). The computational complexity of propositional Strips planning. *Artificial Intelligence*, *69*(1-2), 165–204.

Byrka, J., Grandoni, F., Rothvoß, T., & Sanità, L. (2013). Steiner tree approximation via iterative randomized rounding. *Journal of the ACM*, *60*(1), 6:1–6:33.

Charikar, M., Chekuri, C., Cheung, T.-Y., Dai, Z., Goel, A., Guha, S., & Li, M. (1999). Approximation algorithms for directed Steiner problems. *Journal of Algorithms*, *33*(1), 73–91.

Chvátal, V. (1979). A greedy heuristic for the set-covering problem. *Mathemathics of Operations Research*, *4*(3), 233–235.

Cooper, M. C., Maris, F., & Régnier, P. (2012). Tractable monotone temporal planning. In *Proc. 22nd International Conference on Automated Planning and Scheduling (ICAPS-2012)*, pp. 20–28.

Cooper, M. C., Maris, F., & Régnier, P. (2013). Relaxation of temporal planning problems. In *Proc. 20th International Symposium on Temporal Representation and Reasoning (TIME-2013)*, pp. 37–44.

Cooper, M. C., Maris, F., & Régnier, P. (2014). Monotone temporal planning: Tractability, extensions and applications. *Journal of Artificial Intelligence Research*, *50*, 447–485.

Dinur, I., & Safra, S. (2004). On the hardness of approximating label-cover. *Information Processing Letters*, *89*(5), 247–254.

Domshlak, C., Hoffmann, J., & Katz, M. (2015). Red-black planning: A new systematic approach to partial delete relaxation. *Artificial Intelligence*, *221*, 73–114.

Ghallab, M., Nau, D. S., & Traverso, P. (2016). *Automated Planning and Acting*. Cambridge University Press.

Goldwasser, M. (1997). *Complexity Measures for Assembly Sequences*. Ph.D. thesis, Stanford University, Stanford, CA, USA.

Goldwasser, M. H., & Motwani, R. (1999). Complexity measures for assembly sequences. *International Journal of Computational Geometry & Applications*, *9*(4/5), 371–418.

Halperin, E., & Krauthgamer, R. (2003). Polylogarithmic inapproximability. In *Proc. 35th Annual ACM Symposium on Theory of Computing (STOC-2003)*, pp. 585–594.

Helmert, M. (2006). The fast downward planning system. *Journal of Artificial Intelligence Research*, *26*, 191–246.

Helmert, M., & Domshlak, C. (2009). Landmarks, critical paths and abstractions: What's the difference anyway? In *Proc. 19th International Conference on Automated Planning and Scheduling (ICAPS-2009)*, pp. 162–169.

Hoffmann, J., & Nebel, B. (2001). The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, *14*, 253–302.

Jonsson, P. (1999). Strong bounds on the approximability of two pspace-hard problems in propositional planning. *Annals of Mathematics and Artificial Intelligence*, *26*, 133–147.

Jonsson, P., & Bäckström, C. (1998). State-variable planning under structural restrictions: Algorithms and complexity. *Artificial Intelligence*, *100*(1-2), 125–176.

Keyder, E., & Geffner, H. (2009). Trees of shortest paths vs. Steiner trees: Understanding and improving delete relaxation heuristics. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI-2009)*, pp. 1734–1739.

Keyder, E., Richter, S., & Helmert, M. (2010). Sound and complete landmarks for And/Or graphs. In *Proc. 19th European Conference on Artificial Intelligence (ECAI-2010)*, pp. 335–340.

Kronegger, M., Ordyniak, S., & Pfandler, A. (2019). Backdoors to planning. *Artificial Intelligence*, *269*, 49–75.

Moshkovitz, D. (2015). The projection games conjecture and the NP-hardness of $ln\ n$-approximating set-cover. *Theory of Computing*, *11*, 221–235.

Pommerening, F., & Helmert, M. (2012). Optimal planning for delete-free tasks with incremental LM-cut. In *Proc. 22nd International Conference on Automated Planning and Scheduling (ICAPS-2012)*, pp. 363–367.

Richter, S., & Westphal, M. (2010). The LAMA planner: Guiding cost-based anytime planning with landmarks. *Journal of Artificial Intelligence Research*, *39*, 127–177.

Watel, D., & Weisser, M.-A. (2016). A note on the inapproximability of the minimum monotone satisfying assignment problem. Tech. rep. HAL-01377704, HAL Archives-Ouverte.