# Parameterized Complexity and Kernel Bounds for Hard Planning Problems

Christer Bäckström[1], Peter Jonsson[1], Sebastian Ordyniak[2], and Stefan Szeider[3⋆]

[1] Department of Computer Science, Linköping University, Linköping, Sweden
[2] Faculty of Informatics, Masaryk University, Brno, Czech Republic
[3] Institute of Information Systems, Vienna University of Technology, Vienna, Austria

**Abstract.** The *propositional planning* problem is a notoriously difficult computational problem. Downey et al. (1999) initiated the parameterized analysis of planning (with plan length as the parameter) and Bäckström et al. (2012) picked up this line of research and provided an extensive parameterized analysis under various restrictions, leaving open only one stubborn case. We continue this work and provide a full classification. In particular, we show that the case when actions have no preconditions and at most $e$ postconditions is fixed-parameter tractable if $e \leq 2$ and W[1]-complete otherwise. We show fixed-parameter tractability by a reduction to a variant of the Steiner Tree problem; this problem has been shown fixed-parameter tractable by Guo et al. (2007). If a problem is fixed-parameter tractable, then it admits a polynomial-time self-reduction to instances whose input size is bounded by a function of the parameter, called the *kernel*. For some problems, this function is even polynomial which has desirable computational implications. Recent research in parameterized complexity has focused on classifying fixed-parameter tractable problems on whether they admit polynomial kernels or not. We revisit all the previously obtained restrictions of planning that are fixed-parameter tractable and show that none of them admits a polynomial kernel unless the polynomial hierarchy collapses to its third level.

## 1 Introduction

The propositional planning problem has been the subject of intensive study in knowledge representation, artificial intelligence and control theory and is relevant for a large number of industrial applications [13]. The problem involves deciding whether an *initial state*—an $n$-vector over some set $D$–can be transformed into a *goal state* via the application of *operators* each consisting of *preconditions* and *post-conditions* (or *effects*) stating the conditions that need to hold before the operator can be applied and which conditions will hold after the application of the operator, respectively. It is known that deciding whether an instance has a solution is Pspace-complete, and it remains at least NP-hard under various

⋆ Research supported by the ERC, grant reference 239962.

restrictions [6, 3]. In view of this intrinsic difficulty of the problem, it is natural to study it within the framework of Parameterized Complexity which offers the more relaxed notion of *fixed-parameter tractability* (FPT). A problem is fixed-parameter tractable if it can be solved in time $f(k)n^{O(1)}$ where $f$ is an arbitrary function of the parameter and $n$ is the input size. Indeed, already in a 1999 paper, Downey, Fellows and Stege [8] initiated the parameterized analysis of propositional planning, taking the minimum number of steps from the initial state to the goal state (i.e., the length of the solution plan) as the parameter; this is also the parameter used throughout this paper. More recently, Bäckström et al. [1] picked up this line of research and provided an extensive analysis of planning under various syntactical restrictions, in particular the syntactical restrictions considered by Bylander [6] and by Bäckström and Nebel [3], leaving open only one stubborn class of problems where operators have no preconditions but may involve up to $e$ postconditions (effects).

### New Contributions

We provide a full parameterized complexity analysis of propositional planning without preconditions. In particular, we show the following dichotomy:

(1) Propositional planning where operators have no preconditions but may have up to $e$ postconditions is fixed-parameter tractable for $e \leq 2$ and W[1]-complete for $e > 2$.

W[1] is a parameterized complexity class of problems that are believed to be not fixed-parameter tractable. Indeed, the fixed-parameter tractability of a W[1]-complete problem implies that the Exponential Time Hypothesis fails [7, 11]. We establish the hardness part of the dichotomy (1) by a reduction from a variant of the $k$-Clique problem. The case $e = 2$ is known to be NP-hard [6]. Its difficulty comes from the fact that possibly one of the two postconditions might set a variable to its desired value, but the other postcondition might change a variable from a desired value to an undesired one. This can cause a chain of operators so that finally all variables have their desired value. We show that this behaviour can be modelled by means of a certain problem on Steiner trees in directed graphs, which was recently shown to be fixed-parameter tractable by Guo, Niedermeier and Suchy [15]. We would like to point out that this case (0 preconditions, 2 postconditions) is the only fixed-parameter tractable case among the NP-hard cases in Bylander's system of restrictions (see Table 1).

Our second set of results is concerned with bounds on problem kernels for planning problems. It is known that a decidable problem is fixed-parameter tractable if and only if it admits a polynomial-time self-reduction where the size of the resulting instance is bounded by a function $f$ of the parameter [10, 14, 12]. The function $f$ is called the *kernel size*. By providing upper and lower bounds on the kernel size, one can rigorously establish the potential of polynomial-time preprocessing for the problem at hand. Some NP-hard combinatorial problems such as $k$-Vertex Cover admit polynomially sized kernels, for others such as

| | $e = 1$ | $e = 2$ | fixed $e > 2$ | arbitrary $e$ |
|---|---|---|---|---|
| $p = 0$ | in P<br>in P | in FPT*<br>NP-C | W[1]-C*<br>NP-C | W[2]-C<br>NP-C |
| $p = 1$ | W[1]-C<br>NP-H | W[1]-C<br>NP-H | W[1]-C<br>NP-H | W[2]-C<br>PSPACE-C |
| fixed $p > 1$ | W[1]-C<br>NP-H | W[1]-C<br>PSPACE-C | W[1]-C<br>PSPACE-C | W[2]-C<br>PSPACE-C |
| arbitrary $p$ | W[1]-C<br>PSPACE-C | W[1]-C<br>PSPACE-C | W[1]-C<br>PSPACE-C | W[2]-C<br>PSPACE-C |

**Table 1.** Complexity of BOUNDED PLANNING, restricting the number of preconditions ($p$) and effects ($e$). The problems in FPT do not admit polynomial kernels. Results marked with * are obtained in this paper. All other parameterized results are from [1] and all classical results are from [6].

$k$-PATH an exponential kernel is the best one can hope for [4]. We examine all planning problems that we have previously been shown to be fixed-parameter tractable on whether they admit polynomial kernels. Our results are negative throughout. In particular, it is unlikely that the FPT part in the above dichotomy (1) can be improved to a polynomial kernel:

(2) Propositional planning where operators have no preconditions but may have up to 2 postconditions does not admit a polynomial kernel unless co-NP $\subseteq$ NP/poly.

Recall that by Yap's Theorem [17] co-NP $\subseteq$ NP/poly implies the (unlikely) collapse of the Polynomial Hierarchy to its third level. We establish the kernel lower bound by means of the technique of *OR-compositions* [4]. We also consider the "PUBS" fragments of planning as introduced by Bäckström and Klein [2]. These fragments arise under combinations of syntactical properties (postunique (P), unary (U), Boolean (B), and single-valued (S); definitions are provided in Section 3).

(3) None of the fixed-parameter tractable but NP-hard PUBS restrictions of propositional planning admits a polynomial kernel, unless co-NP $\subseteq$ NP/poly.

According to the PUBS lattice (see Figure 1), only the two maximal restrictions PUB and PBS need to be considered. Moreover, we observe from previous results that a polynomial kernel for restriction PBS implies one for restriction PUB. Hence this leaves restriction PUB as the only one for which we need to show a super-polynomial kernel bound. We establish the latter, as above, by using OR-compositions.

The full proofs of statements marked with $\star$ are omitted due to space restrictions and can be found at http://arxiv.org/abs/1211.0479.

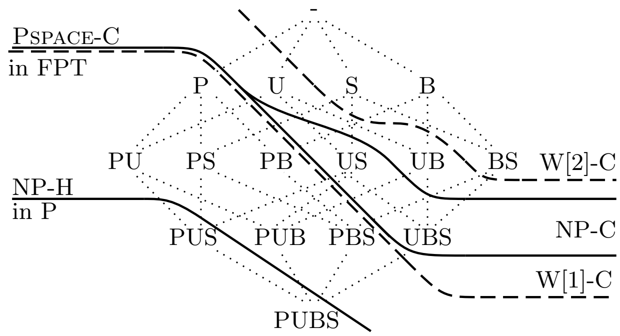


**Fig. 1.** Complexity of BOUNDED PLANNING for the restrictions P, U, B and S illustrated as a lattice defined by all possible combinations of these restrictions [1]. As shown in this paper, PUS and PUBS are the only restrictions that admit a polynomial kernel, unless the Polynomial Hierarchy collapses.

## 2 Parameterized Complexity

We define the basic notions of Parameterized Complexity and refer to other sources [9, 11] for an in-depth treatment. A *parameterized problem* is a set of pairs $\langle \mathbb{I}, k \rangle$, the *instances*, where $\mathbb{I}$ is the main part and $k$ the *parameter*. The parameter is usually a non-negative integer. A parameterized problem is *fixed-parameter tractable (FPT)* if there exists an algorithm that solves any instance $\langle \mathbb{I}, k \rangle$ of size $n$ in time $f(k)n^c$ where $f$ is an arbitrary computable function and $c$ is a constant independent of both $n$ and $k$. FPT is the class of all fixed-parameter tractable decision problems.

Parameterized complexity offers a completeness theory, similar to the theory of NP-completeness, that allows the accumulation of strong theoretical evidence that some parameterized problems are not fixed-parameter tractable. This theory is based on a hierarchy of complexity classes $\mathrm{FPT} \subseteq \mathrm{W}[1] \subseteq \mathrm{W}[2] \subseteq \cdots$ where all inclusions are believed to be strict. An *fpt-reduction* from a parameterized problem $P$ to a parameterized problem $Q$ if is a mapping $R$ from instances of $P$ to instances of $Q$ such that (i) $\langle \mathbb{I}, k \rangle$ is a YES-instance of $P$ if and only if $\langle \mathbb{I}', k' \rangle = R(\mathbb{I}, k)$ is a YES-instance of $Q$, (ii) there is a computable function $g$ such that $k' \leq g(k)$, and (iii) there is a computable function $f$ and a constant $c$ such that $R$ can be computed in time $O(f(k) \cdot n^c)$, where $n$ denotes the size of $\langle \mathbb{I}, k \rangle$.

A *kernelization* [11] for a parameterized problem $P$ is an algorithm that takes an instance $\langle \mathbb{I}, k \rangle$ of $P$ and maps it in time polynomial in $|\mathbb{I}| + k$ to an instance $\langle \mathbb{I}', k' \rangle$ of $P$ such that $\langle \mathbb{I}, k \rangle$ is a YES-instance if and only if $\langle \mathbb{I}', k' \rangle$ is a YES-instance and $|\mathbb{I}'|$ is bounded by some function $f$ of $k$. The output $\mathbb{I}'$ is called a *kernel*. We say $P$ has a *polynomial kernel* if $f$ is a polynomial. Every fixed-parameter tractable problem admits a kernel, but not necessarily a polynomial kernel.

An *OR-composition algorithm* for a parameterized problem $P$ maps $t$ instances $\langle \mathbb{I}_1, k\rangle, \ldots, \langle \mathbb{I}_t, k\rangle$ of $P$ to one instance $\langle \mathbb{I}', k'\rangle$ of $P$ such that the algorithm runs in time polynomial in $\sum_{1 \leq i \leq t} |\mathbb{I}_i| + k$, the parameter $k'$ is bounded by a polynomial in the parameter $k$, and $\langle \mathbb{I}', k'\rangle$ is a YES-instance if and only if there is an $1 \leq i \leq t$ such that $\langle \mathbb{I}_i, k\rangle$ is a YES-instance.

**Proposition 1 (Bodlaender, et al. [4])** *If a parameterized problem $P$ has an OR-composition algorithm, then it has no polynomial kernel unless* co-NP $\subseteq$ NP/poly.

A *polynomial parameter reduction* from a parameterized problem $P$ to a parameterized problem $Q$ is an fpt-reduction $R$ from $P$ to $Q$ such that (i) $R$ can be computed in polynomial time (polynomial in $|\mathbb{I}| + k$), and (ii) there is a polynomial $p$ such that $k' \leq p(k)$ for every instance $\langle \mathbb{I}, k\rangle$ of $P$ with $\langle \mathbb{I}', k'\rangle = R(\langle \mathbb{I}, k\rangle)$. The *unparameterized version* $\tilde{P}$ of a parameterized problem $P$ has the same YES and NO-instances as $P$, except that the parameter $k$ is given in unary $1^k$.

**Proposition 2 (Bodlaender, Thomasse, and Yeo [5])** *Let $P$ and $Q$ be two parameterized problems such that there is a polynomial parameter reduction from $P$ to $Q$, and assume that $\tilde{P}$ is* NP*-complete and $\tilde{Q}$ is in* NP*. Then, if $Q$ has a polynomial kernel also $P$ has a polynomial kernel.*

## 3 Planning Framework

We will now introduce the SAS$^+$ formalism for specifying propositional planning problems [3]. We note that the propositional STRIPS language can be treated as the special case of SAS$^+$ satisfying restriction B (which will be defined below). More precisely, this corresponds to the variant of STRIPS that allows negative preconditions; this formalism is often referred to as PSN.

Let $V = \{v_1, \ldots, v_n\}$ be a finite set of *variables* over a finite *domain* $D$. Implicitly define $D^+ = D \cup \{\mathbf{u}\}$, where $\mathbf{u}$ is a special value (the *undefined value*) not present in $D$. Then $D^n$ is the set of *total states* and $(D^+)^n$ is the set of *partial states* over $V$ and $D$, where $D^n \subseteq (D^+)^n$. The value of a variable $v$ in a state $s \in (D^+)^n$ is denoted $s[v]$. A *SAS$^+$ instance* is a tuple $\mathbb{P} = \langle V, D, A, I, G\rangle$ where $V$ is a set of variables, $D$ is a domain, $A$ is a set of *actions*, $I \in D^n$ is the *initial state* and $G \in (D^+)^n$ is the *goal*. Each action $a \in A$ has a *precondition* $\mathrm{pre}(a) \in (D^+)^n$ and an *effect* $\mathrm{eff}(a) \in (D^+)^n$. We will frequently use the convention that a variable has value $\mathbf{u}$ in a precondition/effect unless a value is explicitly specified. Let $a \in A$ and let $s \in D^n$. Then $a$ is *valid in $s$* if for all $v \in V$, either $\mathrm{pre}(a)[v] = s[v]$ or $\mathrm{pre}(a)[v] = \mathbf{u}$. Furthermore, the *result of $a$ in $s$* is a state $t \in D^n$ defined such that for all $v \in V$, $t[v] = \mathrm{eff}(a)[v]$ if $\mathrm{eff}(a)[v] \neq \mathbf{u}$ and $t[v] = s[v]$ otherwise.

Let $s_0, s_\ell \in D^n$ and let $\omega = \langle a_1, \ldots, a_\ell\rangle$ be a sequence of actions. Then $\omega$ is a *plan from $s_0$ to $s_\ell$* if either (i) $\omega = \langle\rangle$ and $\ell = 0$ or (ii) there are states

$s_1, \ldots, s_{\ell-1} \in D^n$ such that for all $i$, where $1 \leq i \leq \ell$, $a_i$ is valid in $s_{i-1}$ and $s_i$ is the result of $a_i$ in $s_{i-1}$. A state $s \in D^n$ is a *goal state* if for all $v \in V$, either $G[v] = s[v]$ or $G[v] = \mathbf{u}$. An action sequence $\omega$ is a *plan for* $\mathbb{P}$ if it is a plan from $I$ to some goal state $s \in D^n$. We will study the following problem:

> BOUNDED PLANNING
> *Instance:* A tuple $\langle \mathbb{P}, k \rangle$ where $\mathbb{P}$ is a SAS$^+$ instance and $k$ is a positive integer.
> *Parameter:* The integer $k$.
> *Question:* Does $\mathbb{P}$ have a plan of length at most $k$?

We will consider the following four syntactical restrictions, originally defined by Bäckström and Klein [2].

- **P** (postunique): For each $v \in V$ and each $x \in D$ there is at most one $a \in A$ such that $\mathrm{eff}(a)[v] = x$.
- **U** (unary): For each $a \in A$, $\mathrm{eff}(a)[v] \neq \mathbf{u}$ for exactly one $v \in V$.
- **B** (Boolean): $|D| = 2$.
- **S** (single-valued): For all $a, b \in A$ and all $v \in V$, if $\mathrm{pre}(a)[v] \neq \mathbf{u}$, $\mathrm{pre}(b)[v] \neq \mathbf{u}$ and $\mathrm{eff}(a)[v] = \mathrm{eff}(b)[v] = \mathbf{u}$, then $\mathrm{pre}(a)[v] = \mathrm{pre}(b)[v]$.

For any set $R$ of such restrictions we write $R$-BOUNDED PLANNING to denote the restriction of BOUNDED PLANNING to only instances satisfying the restrictions in $R$. Additionally we will consider restrictions on the number of preconditions and effects as previously considered in [6]. For two non-negative integers $p$ and $e$ we write $(p, e)$-BOUNDED PLANNING to denote the restriction of BOUNDED PLANNING to only instances where every action has at most $p$ preconditions and at most $e$ effects. Table 1 and Figure 1 summarize results from [6, 3, 1] combined with the results presented in this paper.

## 4 Parameterized Complexity of $(0, e)$-Bounded Planning

In this section we completely characterize the parameterized complexity of BOUNDED PLANNING for planning instances without preconditions. It is known [1] that BOUNDED PLANNING without preconditions is contained in the parameterized complexity class W[1]. Here we show that $(0, e)$-BOUNDED PLANNING is also W[1]-hard for every $e > 2$ but it becomes fixed-parameter tractable if $e \leq 2$. Because $(0, 1)$-BOUNDED PLANNING is trivially solvable in polynomial time this completely characterized the parameterized complexity of BOUNDED PLANNING without preconditions.

### 4.1 Hardness Results

**Theorem 1** $(0, 3)$-BOUNDED PLANNING *is* W[1]*-hard.*

*Proof.* We devise a parameterized reduction from the following problem, which is W[1]-complete [16].

Multicolored Clique
*Instance:* A $k$-partite graph $G = (V, E)$ with partition $V_1, \dots, V_k$ such that $|V_i| = |V_j| = n$ for $1 \le i < j \le k$.
*Parameter:* The integer $k$.
*Question:* Are there vertices $v_1, \dots, v_k$ such that $v_i \in V_i$ for $1 \le i \le k$ and $\{v_i, v_j\} \in E$ for $1 \le i < j \le k$? (The graph $K = (\{v_1, \dots, v_k\}, \{\,\{v_i, v_j\} : 1 \le i < j \le k\,\})$ is a $k$-*clique* of $G$.)

Let $\mathbb{I} = (G, k)$ be an instance of this problem with partition $V_1, \dots, V_k$, $|V_1| = \dots = |V_k| = n$ and parameter $k$. We construct a $(0, 3)$-Bounded Planning instance $\mathbb{I}' = (\mathbb{P}', k')$ with $\mathbb{P}' = \langle V', D', A', I', G' \rangle$ such that $\mathbb{I}$ is a Yes-instance if and only if so is $\mathbb{I}'$.

We set $V' = V(G) \cup \{\, p_{i,j} : 1 \le i < j \le k \,\}$, $D' = \{0, 1\}$, $I' = \langle 0, \dots, 0 \rangle$, $G'[p_{i,j}] = 1$ for every $1 \le i < j \le k$ and $G'[v] = 0$ for every $v \in V(G)$. Furthermore, the set $A'$ contains the following actions:

- For every $v \in V(G)$ one action $a_v$ with $\mathrm{eff}(a_v)[v] = 0$;
- For every $e = \{v_i, v_j\} \in E(G)$ with $v_i \in V_i$ and $v_j \in V_j$ one action $a_e$ with $\mathrm{eff}(a_e)[v_i] = 1$, $\mathrm{eff}(a_e)[v_j] = 1$, and $\mathrm{eff}(a_e)[p_{i,j}] = 1$.

Clearly, every action in $A'$ has no precondition and at most 3 effects.

The theorem will follow after we have shown the that $G$ contains a $k$-clique if and only if $\mathbb{P}$ has a plan of length at most $k' = \binom{k}{2} + k$. Suppose that $G$ contains a $k$-clique with vertices $v_1, \dots, v_k$ and edges $e_1, \dots, e_{k''}$, $k'' = \binom{k}{2}$. Then $\omega' = \langle a_{e_1}, \dots, a_{e_{k''}}, a_{v_1}, \dots, a_{v_k} \rangle$ is a plan of length $k'$ for $\mathbb{P}'$. For the reverse direction suppose that $\omega'$ is a plan of length at most $k'$ for $\mathbb{P}'$. Because $I'[p_{i,j}] = 0 \neq G'[p_{i,j}] = 1$ the plan $\omega'$ has to contain at least one action $a_e$ where $e$ is an edge between a vertex in $V_i$ and a vertex in $V_j$ for every $1 \le i < j \le k$. Because $\mathrm{eff}(a_{e=\{v_i,v_j\}})[v_i] = 1 \neq G[v_i] = 0$ and $\mathrm{eff}(a_{e=\{v_i,v_j\}})[v_j] = 1 \neq G[v_j] = 0$ for every such edge $e$ it follows that $\omega'$ has to contain at least one action $a_v$ with $v \in V_i$ for every $1 \le i \le k$. Because $k' = \binom{k}{2} + k$ it follows that $\omega'$ contains exactly $\binom{k}{2}$ actions of the form $a_e$ for some edge $e \in E(G)$ and exactly $k$ actions of the form $a_v$ for some vertex $v \in V(G)$. It follows that the graph $K = (\{\, v : a_v \in \omega \,\}, \{\, e : a_e \in \omega \,\})$ is a $k$-clique of $G$. $\square$

### 4.2 Fixed-Parameter Tractability

Before we show that $(0, 2)$-Bounded Planning is fixed-parameter tractable we need to introduce some notions and prove some simple properties of $(0, 2)$-Bounded Planning. Let $\mathbb{P} = \langle V, D, A, I, G \rangle$ be an instance of Bounded Planning. We say an action $a \in A$ has an effect on some variable $v \in V$ if $\mathrm{eff}(a)[v] \neq \mathbf{u}$, we call this effect *good* if furthermore $\mathrm{eff}(a)[v] = G[v]$ or $G[v] = \mathbf{u}$ and we call the effect *bad* otherwise. We say an action $a \in A$ is *good* if it has only good effects, *bad* if it has only bad effects, and *mixed* if it has at least one good and at least one bad effect. Note that if a valid plan contains a bad action then this action can always be removed without changing the validity of the plan.

Consequently, we only need to consider good and mixed actions. Furthermore, we denote by $B(V)$ the set of variables $v \in V$ with $G[v] \neq \mathbf{u}$ and $I[v] \neq G[v]$.

The next lemma shows that we do not need to consider good actions with more than 1 effect for $(0,2)$-Bounded Planning.

**Lemma 1 ($\star$)** *Let $\mathbb{I} = \langle \mathbb{P}, k \rangle$ be an instance of* $(0,2)$-Bounded Planning. *Then $\mathbb{I}$ can be fpt-reduced to an instance $\mathbb{I}' = \langle \mathbb{P}', k' \rangle$ of* $(0,2)$-Bounded Planning *where $k' = k(k+3)+1$ and no good action of $\mathbb{I}'$ effects more than one variable.*

**Theorem 2** $(0,2)$-Bounded Planning *is fixed-parameter tractable.*

*Proof.* We show fixed-parameter tractability of $(0,2)$-Bounded Planning by reducing it to the following fixed-parameter tractable problem [15].

> Directed Steiner Tree
> *Instance:* A set of nodes $N$, a weight function $w : N \times N \to (\mathbb{N} \cup \{\infty\})$, a root node $s \in N$, a set $T \subseteq N$ of terminals , and a weight bound $p$.
> *Parameter:* $p_M = \frac{p}{\min\{\, w(u,v) \,:\, u,v \in N \,\}}$.
> *Question:* Is there a set of arcs $E \subseteq N \times N$ of weight $w(E) \leq p$ (where $w(E) = \sum_{e \in E} w(e)$) such that in the digraph $D = (N, E)$ for every $t \in T$ there is a directed path from $s$ to $t$? We will call the digraph $D$ a *directed Steiner Tree (DST)* of weight $w(E)$.

Let $\mathbb{I} = \langle \mathbb{P}, k \rangle$ where $\mathbb{P} = \langle V, D, A, I, G \rangle$ be an instance of $(0,2)$-Bounded Planning. Because of Lemma 1 we can assume that $A$ contains no good actions with two effects. We construct an instance $\mathbb{I}' = \langle N, w, s, T, p \rangle$ of Directed Steiner Tree where $p_M = k$ such that $\mathbb{I}$ is a Yes-instance if and only if $\mathbb{I}'$ is a Yes-instance. Because $p_M = k$ this shows that $(0,2)$-Bounded Planning is fixed-parameter tractable.

We are now ready to define the instance $\mathbb{I}'$. The node set $N$ consists of the root vertex $s$ and one node for every variable in $V$. The weight function $w$ is $\infty$ for all but the following arcs: (i) For every good action $a \in A$ the arc from $s$ to the unique variable $v \in V$ that is effected by $a$ gets weight 1. (ii) For every mixed action $a \in A$ with some good effect on some variable $v_g \in V$ and some bad effect on some variable $v_b \in V$, the arc from $v_b$ to $v_g$ gets weight 1.

We identify the root $s$ from the instance $\mathbb{I}$ with the node $s$, we let $T$ be the set $B(V)$, and $p_M = p = k$.

**Claim 1 ($\star$)** *$\mathbb{P}$ has a plan of length at most $k$ if and only if $\mathbb{I}'$ has a DST of weight at most $p_M = p = k$.*

The theorem follows. $\square$

## 5 Kernel Lower Bounds

Since $(0,2)$-Bounded Planning is fixed-parameter tractable by Theorem 2 it admits a kernel. Next we provide strong theoretical evidence that the problem does not admit a polynomial kernel. The proof of Theorem 3 is based on an OR-composition algorithm and Proposition 1.

**Theorem 3 ($\star$)** $(0, 2)$-Bounded Planning *has no polynomial kernel unless* co-NP $\subseteq$ NP/poly.

In previous work [1] we have classified the parameterized complexity of the "PUBS" fragments of Bounded Planning. It turned out that the problems fall into four categories (see Figure 1): (i) polynomial-time solvable, (ii) NP-hard but fixed-parameter tractable, (iii) W[1]-complete, and (iv) W[2]-complete. The aim of this section is to further refine this classification with respect to kernelization. The problems in category (i) trivially admit a kernel of constant size, whereas the problems in categories (iii) and (iv) do not admit a kernel at all (polynomial or not), unless W[1] = FPT or W[2] = FPT, respectively. Hence it remains to consider the six problems in category (ii), each of them could either admit a polynomial kernel or not. We show that none of them does.

According to our classification [1], the problems in category (ii) are exactly the problems $R$-Bounded Planning, for $R \subseteq \{P, U, B, S\}$, such that $P \in R$ and $\{P, U, S\} \nsubseteq R$.

**Theorem 4** *None of the problems $R$-*Bounded Planning *for $R \subseteq \{P, U, B, S\}$ such that $P \in R$ and $\{P, U, S\} \nsubseteq R$ (i.e., the problems in category (ii)) admits a polynomial kernel unless* co-NP $\subseteq$ NP/poly.

The remainder of this section is devoted to establish Theorem 4. The relationship between the problems as indicated in Figure 1 greatly simplifies the proof. Instead of considering all six problems separately, we can focus on the two most restricted problems $\{P, U, B\}$-Bounded Planning and $\{P, B, S\}$-Bounded Planning. If any other problem in category (ii) would have a polynomial kernel, then at least one of these two problems would have one. This follows by Proposition 2 and the following facts:

1. The unparameterized versions of all the problems in category (ii) are NP-complete. This holds since the corresponding classical problems are strongly NP-hard, hence the problems remain NP-hard when $k$ is encoded in unary (as shown by Bäckström and Nebel [3]);
2. If $R_1 \subseteq R_2$ then the identity function gives a polynomial parameter reduction from $R_2$-Bounded Planning to $R_1$-Bounded Planning.

Furthermore, the following result of Bäckström and Nebel [3, Theorem 4.16] even provides a polynomial parameter reduction from $\{P, U, B\}$-Bounded Planning to $\{P, B, S\}$-Bounded Planning. Consequently, $\{P, U, B\}$-Bounded Planning remains the only problem for which we need to establish a superpolynomial kernel lower bound.

**Proposition 3 (Bäckström and Nebel [3])** *Let $\mathbb{I} = \langle \mathbb{P}, k \rangle$ be an instance of $\{P, U, B\}$-*Bounded Planning*. Then $\mathbb{I}$ can be transformed in polynomial time into an equivalent instance $\mathbb{I}' = \langle \mathbb{P}', k' \rangle$ of $\{P, B, S\}$-*Bounded Planning *such that $k = k'$.*

Hence, in order to complete the proof of Theorem 4 it only remains to establish the next lemma.

**Lemma 2** $\{P, U, B\}$-Bounded Planning *has no polynomial kernel unless* co-NP $\subseteq$ NP/poly*.*

*Proof.* Because of Proposition 1, it suffices to devise an OR-composition algorithm for $\{P, U, B\}$-Bounded Planning. Suppose we are given $t$ instances $\mathbb{I}_1 = \langle \mathbb{P}_1, k \rangle, \dots, \mathbb{I}_t = \langle \mathbb{P}_t, k \rangle$ of $\{P, U, B\}$-Bounded Planning where $\mathbb{P}_i = \langle V_i, D_i, A_i, I_i, G_i \rangle$ for every $1 \leq i \leq t$. It has been shown in [1, Theorem 5] that $\{P, U, B\}$-Bounded Planning can be solved in time $O^*(S(k))$ (where $S(k) = 2 \cdot 2^{(k+2)^2} \cdot (k+2)^{(k+1)^2}$ and the $O^*$ notation suppresses polynomial factors). It follows that $\{P, U, B\}$-Bounded Planning can be solved in polynomial time with respect to $\sum_{1 \leq i \leq t} |\mathbb{I}_i| + k$ if $t > S(k)$. Hence, if $t > S(k)$ this gives us an OR-composition algorithm as follows. We first run the algorithm for $\{P, U, B\}$-Bounded Planning on each of the $t$ instances. If one of these $t$ instances is a Yes-instance then we output this instance. If not then we output any of the $t$ instances. This shows that $\{P, U, B\}$-Bounded Planning has an OR-composition algorithm for the case that $t > S(k)$. Hence, in the following we can assume that $t \leq S(k)$.

Given $\mathbb{I}_1, \dots, \mathbb{I}_t$ we will construct an instance $\mathbb{I} = \langle \mathbb{P}, k' \rangle$ of $\{P, U, B\}$-Bounded Planning as follows. For the construction of $\mathbb{I}$ we need the following auxiliary gadget, which will be used to calculate the logical "OR" of two binary variables. The construction of the gadget uses ideas from [3, Theorem 4.15]. Assume that $v_1$ and $v_2$ are two binary variables. The gadget $\mathrm{OR}_2(v_1, v_2, o)$ consists of the five binary variables $o_1$, $o_2$, $o$, $i_1$, and $i_2$. Furthermore, $\mathrm{OR}_2(v_1, v_2, o)$ contains the following actions:

– the action $a_o$ with $\mathrm{pre}(a_o)[o_1] = \mathrm{pre}(a_o)[o_2] = 1$ and $\mathrm{eff}(a_o)[o] = 1$;
– the action $a_{o_1}$ with $\mathrm{pre}(a_{o_1})[i_1] = 1$, $\mathrm{pre}(a_{o_1})[i_2] = 0$ and $\mathrm{eff}(a_{o_1})[o_1] = 1$;
– the action $a_{o_2}$ with $\mathrm{pre}(a_{o_2})[i_1] = 0$, $\mathrm{pre}(a_{o_2})[i_2] = 1$ and $\mathrm{eff}(a_{o_2})[o_2] = 1$;
– the action $a_{i_1}$ with $\mathrm{eff}(a_{i_1})[i_1] = 1$;
– the action $a_{i_2}$ with $\mathrm{eff}(a_{i_2})[i_2] = 1$;
– the action $a_{v_1}$ with $\mathrm{pre}(a_{v_1})[v_1] = 1$ and $\mathrm{eff}(a_{v_1})[i_1] = 0$;
– the action $a_{v_2}$ with $\mathrm{pre}(a_{v_2})[v_2] = 1$ and $\mathrm{eff}(a_{v_2})[i_2] = 0$;

We now show that $\mathrm{OR}_2(v_1, v_2, o)$ can indeed be used to compute the logical "OR" of the variables $v_1$ and $v_2$. We need the following claim.

**Claim 2 ($\star$)** *Let* $\mathbb{P}(\mathrm{OR}_2(v_1, v_2, o))$ *be a* $\{P, U, B\}$-Bounded Planning *instance that consists of the two binary variables $v_1$ and $v_2$, and the variables and actions of the gadget* $\mathrm{OR}_2(v_1, v_2, o)$*. Furthermore, let the initial state of* $\mathbb{P}(\mathrm{OR}_2(v_1, v_2, o))$ *be any initial state that sets all variables of the gadget* $\mathrm{OR}_2(v_1, v_2, o)$ *to* $0$ *but assigns the variables $v_1$ and $v_2$ arbitrarily, and let the goal state of* $\mathbb{P}(\mathrm{OR}_2(v_1, v_2, o))$ *be defined by* $G[o] = 1$*. Then* $\mathbb{P}(\mathrm{OR}_2(v_1, v_2, o))$ *has a plan if and only if its initial state sets at least one of the variables $v_1$ or $v_2$ to* $1$*. Furthermore, if there is such a plan then its length is* $6$*.*

We continue by showing how we can use the gadget $\mathrm{OR}_2(v_1, v_2, o)$ to construct a gadget $\mathrm{OR}(v_1, \dots, v_r, o)$ such that there is a sequence of actions of $\mathrm{OR}(v_1, \dots, v_r, o)$ that sets the variable $o$ to 1 if and only if at least one of the external variables $v_1, \dots, v_r$ are initially set to 1. Furthermore, if there is such a sequence of actions then its length is at most $6\lceil \log r \rceil$. Let $T$ be a rooted binary tree with root $s$ that has $r$ leaves $l_1, \dots, l_r$ and is of smallest possible height. For every node $t \in V(T)$ we make a copy of our binary OR-gadget such that the copy of a leave node $l_i$ is the gadget $\mathrm{OR}_2(v_{2i-1}, v_{2i}, o_{l_i})$ and the copy of an inner node $t \in V(T)$ with children $t_1$ and $t_2$ is the gadget $\mathrm{OR}_2(o_{t_1}, o_{t_2}, o_t)$ (clearly this needs to be adapted if $r$ is odd or an inner node has only one child). For the root node with children $t_1$ and $t_2$ the gadget becomes $\mathrm{OR}_2(o_{t_1}, o_{t_2}, o)$. This completes the construction of the gadget $\mathrm{OR}(v_1, \dots, v_r, o)$. Using Claim 2 it is easy to verify that the gadget $\mathrm{OR}(v_1, \dots, v_r, o)$ can indeed be used to compute the logical "OR" or the variables $v_1, \dots, v_r$.

We are now ready to construct the instance $\mathbb{I}$. $\mathbb{I}$ contains all the variables and actions from every instance $\mathbb{I}_1, \dots, \mathbb{I}_t$ and of the gadget $\mathrm{OR}(v_1, \dots, v_t, o)$. Additionally, $\mathbb{I}$ contains the binary variables $v_1, \dots, v_t$ and the actions $a_1, \dots, a_t$ with $\mathrm{pre}(a_i) = G_i$ and $\mathrm{eff}(a_i)[v_i] = 1$. Furthermore, the initial state $I$ of $\mathbb{I}$ is defined as $I[v] = I_i[v]$ if $v$ is a variable of $\mathbb{I}_i$ and $I[v] = 0$, otherwise. The goal state of $\mathbb{I}$ is defined by $G[o] = 1$ and we set $k' = k + 6\lceil \log t \rceil$. Clearly, $\mathbb{I}$ can be constructed from $\mathbb{I}_1, \dots, \mathbb{I}_t$ in polynomial time and $\mathbb{I}$ is a YES-instance if and only if at least one of the instances $\mathbb{I}_1, \dots, \mathbb{I}_t$ is a YES-instance. Furthermore, because $k' = k + 6\lceil \log t \rceil \leq k + 6\lceil \log S(k) \rceil = k + 6\lceil 1 + (k+2)^2 + (k+1)^2 \cdot \log(k+2) \rceil$, the parameter $k'$ is polynomial bounded by the parameter $k$. This concludes the proof of the lemma. □

## 6 Conclusion

We have studied the parameterized complexity of BOUNDED PLANNING with respect to the parameter plan length. In particular, we have shown that $(0, e)$-BOUNDED PLANNING is fixed-parameter tractable for $e \leq 2$ and W[1]-complete for $e > 2$. Together with our previous results [1] this completes the full classification of planning in Bylander's system of restrictions (see Table 1). Interestingly, $(0, 2)$-BOUNDED PLANNING turns out to be the only nontrivial fixed-parameter tractable case (where the unparameterized version is NP-hard).

We have also provided a full classification of kernel sizes for $(0, 2)$-BOUNDED PLANNING and all the fixed-parameter tractable fragments of BOUNDED PLANNING in the "PUBS" framework. It turns out that none of the nontrivial problems (where the unparameterized version is NP-hard) admits a polynomial kernel unless the Polynomial Hierarchy collapses. This implies an interesting *dichotomy* concerning the kernel size: we only have constant-size and superpolynomial kernels—polynomially bounded kernels that are not of constant size are absent.

## References


1. C. Bäckström, Y. Chen, P. Jonsson, S. Ordyniak, and S. Szeider. The complexity of planning revisited - a parameterized analysis. In J. Hoffmann and B. Selman, editors, *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada*. AAAI Press, 2012.
2. C. Bäckström and I. Klein. Planning in polynomial time: the SAS-PUBS class. *Comput. Intelligence*, 7:181–197, 1991.
3. C. Bäckström and B. Nebel. Complexity results for SAS+ planning. *Comput. Intelligence*, 11:625–656, 1995.
4. H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *J. of Computer and System Sciences*, 75(8):423–434, 2009.
5. H. L. Bodlaender, S. Thomassé, and A. Yeo. Kernel bounds for disjoint cycles and disjoint paths. In A. Fiat and P. Sanders, editors, *Algorithms - ESA 2009, 17th Annual European Symposium, Copenhagen, Denmark, September 7-9, 2009. Proceedings*, volume 5757 of *Lecture Notes in Computer Science*, pages 635–646. Springer Verlag, 2009.
6. T. Bylander. The computational complexity of propositional STRIPS planning. *Artificial Intelligence*, 69(1–2):165–204, 1994.
7. J. Chen, X. Huang, I. A. Kanj, and G. Xia. Strong computational lower bounds via parameterized complexity. *J. of Computer and System Sciences*, 72(8):1346–1367, 2006.
8. R. Downey, M. R. Fellows, and U. Stege. Parameterized complexity: A framework for systematically confronting computational intractability. In *Contemporary Trends in Discrete Mathematics: From DIMACS and DIMATIA to the Future*, volume 49 of *AMS-DIMACS*, pages 49–99. American Mathematical Society, 1999.
9. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer Verlag, New York, 1999.
10. M. R. Fellows. The lost continent of polynomial time: Preprocessing and kernelization. In *Parameterized and Exact Computation, Second International Workshop, IWPEC 2006, Zürich, Switzerland, September 13-15, 2006, Proceedings*, volume 4169 of *Lecture Notes in Computer Science*, pages 276–277. Springer Verlag, 2006.
11. J. Flum and M. Grohe. *Parameterized Complexity Theory*, volume XIV of *Texts in Theoretical Computer Science. An EATCS Series*. Springer Verlag, Berlin, 2006.
12. F. V. Fomin. Kernelization. In F. M. Ablayev and E. W. Mayr, editors, *Computer Science - Theory and Applications, 5th International Computer Science Symposium in Russia, CSR 2010, Kazan, Russia, June 16-20, 2010. Proceedings*, volume 6072 of *Lecture Notes in Computer Science*, pages 107–108. Springer Verlag, 2010.
13. M. Ghallab, D. S. Nau, and P. Traverso. *Automated planning - theory and practice*. Elsevier, 2004.
14. J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(2):31–45, Mar. 2007.
15. J. Guo, R. Niedermeier, and O. Suchý. Parameterized complexity of arc-weighted directed steiner problems. *SIAM J. Discrete Math.*, 25(2):583–599, 2011.
16. K. Pietrzak. On the parameterized complexity of the fixed alphabet shortest common supersequence and longest common subsequence problems. *J. of Computer and System Sciences*, 67(4):757–771, 2003.
17. C.-K. Yap. Some consequences of nonuniform conditions on uniform classes. *Theoretical Computer Science*, 26(3):287–300, 1983.