Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Plan Reordering and Parallel Execution — A Parameterized Complexity View

Meysam Aghighi, Christer Bäckström
Department of Computer and Information Science
Linköping University
Linköping, Sweden
{meysam.aghighi, christer.backstrom} at liu.se

Abstract

Bäckström has previously studied a number of optimization problems for partial-order plans, like finding a minimum deordering (MCD) or reordering (MCR), and finding the minimum parallel execution length (PPL), which are all **NP**-complete. We revisit these problems, but applying parameterized complexity analysis rather than standard complexity analysis. We consider various parameters, including both the original and desired size of the plan order, as well as its width and height. Our findings include that MCD and MCR are **W**[2]-hard and in **W**[P] when parameterized with the desired order size, and MCD is fixed-parameter tractable (fpt) when parameterized with the original order size. Problem PPL is fpt if parameterized with the size of the non-concurrency relation, but para-**NP**-hard in most other cases. We also consider this problem when the number (k) of agents, or processors, is restricted, finding that this number is a crucial parameter; this problem is fixed-parameter tractable with the order size, the parallel execution length and k as parameter, but para-**NP**-hard without k as parameter.

1 Introduction

A *total-order (t.o.) plan* specifies one single total order for executing its actions, while a *partial-order (p.o.) plan* specifies a partial order on its actions and allows for executing them in any order that is a linearization of the partial order. A p.o. plan is more flexible than a t.o. plan, since the decision on an exact action order can be postponed until runtime, possibly subject to constraints not known at planning time. A p.o. plan may also be interpreted as a parallel plan, where mutually unordered actions can be executed simultaneously, possibly subject to further concurrency constraints. Planners may also be divided into total-order planners and partial-order planners, depending on whether they work on and produce a t.o. plan or a p.o. plan. Furthermore, one may use a p.o. planner and then linearize the result into a t.o. plan, or one may use a t.o. planner and then convert the result into a p.o. plan. A number of greedy algorithms for the latter appeared in the literature in the early 1990's (Régnier and Fade 1991; Veloso, Pérez, and Carbonell 1990; Kambhampati and Kedar 1994). Bäckström (1998) considered this from a computational point of view, defining *deordering* of a plan, a subset of the plan order such that the plan is still valid, and *reordering* of a plan, any partial order such that the plan is still valid. The optimization problems of finding a minimum size deordering or reordering of a plan were both found **NP**-complete. Hence, neither of the previous greedy deordering algorithms could produce a minimum deordering. However, one of the greedy algorithms has later been found to often produce optimal solutions in practice (Muise, McIlraith, and Beck 2011). Also other methods to solve these problems in practice have been proposed, e.g. using MaxSAT encodings and solvers (Muise, McIlraith, and Beck 2012).

The method of first generating a t.o. plan and then deordering this into a p.o. plan, usually by a non-optimal greedy algorithm, has since been applied to a number of diverse planning domains, including workflow planning in business process reengineering (Rodríguez-Moreno et al. 2007), automatic generation of narratives (Haslum 2012), composition of web services (Madhusudan and Uttamsingh 2006), flexible plan execution and monitoring (Muise, McIlraith, and Beck 2011) and plan repair (Scala and Torasso 2015) of numeric plans. The deordering concept can also be found in distributed control systems (Pannek 2013). Furthermore, Cimatti, Micheli, and Roveri (2015) presented a strong temporal planner for actions with uncontrollable durations, based on constructing a t.o. plan and then consider the reorederings of it.

One may also consider other optimality criteria for p.o. plans than minimum order size. One alternative is to maximize the number of linearizations. Since this is also a difficult problem, one often considers various proxy functions in practice, cf. Say, Ciré, and Beck (2016). Siddiqui and Haslum (2015) take a slightly different approach, considering blocks of actions that can be unordered with respect to each other, and use this to construct an anytime algorithm for deordering plans.

While finding a (usually non-optimal) de- or reordering of a t.o. plan has proven sufficient in practice in many applications, this is not always the case. For instance, Kvarnström (2011) considers multi-agent UAV planning and generate p.o. plans directly, claiming that post-relaxation of a t.o. plan is not sufficient for achieving optimality in this domain. Another case where the method seems to be problematic is planning for multiple agents that have both private and public actions (Maliah, Shani, and Stern 2016). Fur-

thermore, when studying this from a computational point of view, Bäckström and Jonsson (2011) have conjectured that if all scheduling constraints are known at planning time, then it is more efficient to generate a plan together with an optimal parallel schedule as one computation, instead of first generating a p.o. plan that allows for an optimal schedule and then find this schedule.

Since the approach of first finding a plan and then improving its order has proven feasible and attractive in practice, despite these latter cases and arguments, it is desirable to gain a better understanding of this approach. One way to do that is to use more fine-grained analysis tools than standard complexity theory. In this paper, we will thus make a preliminary investigation into applying parameterized complexity analysis to the problems defined by Bäckström (1998). This allows for sharper and more detailed results and distinctions, and parameterized analysis has recently been used to achieve such refined pictures of planning (Kronegger, Pfandler, and Pichler 2013; de Haan, Roubícková, and Szeider 2013; Bäckström et al. 2015; Aghighi and Bäckström 2015; Kronegger, Ordyniak, and Pfandler 2015).

We primarily study the problems defined by Bäckström (1998), but using parameterized complexity instead. The following are some of our results. Finding a minimum deordering (MCD) and a minimum reordering (MCR) are previously known to be **NP**-complete. We show that both are **W**[2]-hard and in **W**[P] if parameterized with the size of the resulting order. Parameterized with the original order, MCD is fixed-parameter tractable, while we could only prove that MCR is in **W**[P]. We also find that the size of the non-concurrency relation is a very useful parameter for finding minimum parallel executions of plans; the problem is fixed-parameter tractable with this parameter, but para-**NP**-hard for most other parameters. We also study the case of a restricted number of agents, or processors, finding this number to be a crucial parameter in this case.

2 Parameterized Complexity

Parameterized complexity provides a more fine-grained complexity analysis compared to the traditional complexity analysis, and it is intended to give results that are more compatible with practical experience.

A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{Z}_0$, where Σ is a finite alphabet and $\mathbb{Z}_0$ is the set of non-negative integers. An *instance* of the problem is a pair $\langle \mathbb{I}, k \rangle$, where $\mathbb{I}$ is a string over Σ^* and k is the *parameter*. For a combination of more parameters $k_1, \ldots, k_p$, the second component of the instance is considered as their sum $k = k_1 + \ldots + k_p$. A parameterized problem is *fixed-parameter tractable (fpt)* if there exists an algorithm that solves every instance $\langle \mathbb{I}, k \rangle$ in time $f(k) \cdot |\mathbb{I}|^c$ where f is a computable function and c is a constant. **FPT** is the class of all fixed-parameter tractable decision problems. An *fpt reduction* from a parameterized language $L_1 \subseteq \Sigma_1^* \times \mathbb{Z}_0$ to another parameterized language $L_2 \subseteq \Sigma_2^* \times \mathbb{Z}_0$ is a mapping $R : \Sigma_1^* \times \mathbb{Z}_0 \to \Sigma_2^* \times \mathbb{Z}_0$ such that: (1) $\langle \mathbb{I}, k \rangle \in L_1$ iff $\langle \mathbb{I}', k' \rangle = R(\mathbb{I}, k) \in L_2$; (2) there is a computable function f and a constant c such that R can be computed in time $f(k) \cdot |\mathbb{I}|^c$; and (3) there is a computable function g such that $k' \leq g(k)$.

In addition to **FPT**, there are other classes defined for fixed-parameter intractability. The $\mathbf{W}[i]$ classes are defined by the WEIGHTED SATISFIABILITY PROBLEM for restricted circuits, where **W**[P] is the case of arbitrary circuits. The class para-**NP** consists of all parameterized problems that can be solved in *non*-deterministic time $f(k) \cdot |\mathbb{I}|^c$, for some computable function f and a constant c. The following relation between the parameterized complexity classes is known: $\mathbf{FPT} \subseteq \mathbf{W}[1] \subseteq \mathbf{W}[2] \subseteq \mathbf{W}[3] \subseteq \cdots \subseteq \mathbf{W}[P] \subseteq$ para-**NP**. On the other hand, the only known relation between parameterized and the standard complexity classes is $\mathbf{P} \subseteq \mathbf{FPT}$. To show para-**NP**-hardness in the paper, we use Theorem 2.14 in Flum and Grohe (2006): A parameterized problem is para-**NP**-hard if it has an **NP**-hard slice, i.e. if setting all parameters to constant values result in a non-parameterized problem that is **NP**-hard. To show **W**[P]-membership, we use Def 3.1 of Flum and Grohe (2006): A parameterized problem is in **W**[P] if it can be solved by some NTM in $f(k).n^c$ steps of which at most $h(k) \cdot \log(n)$ steps are non-deterministic, where f and h are computable functions, c is a constant and n is the instance size. For a detailed account of parameterized complexity, see Downey and Fellows (1999) or Flum and Grohe (2006).

3 Planning Framework

We follow Bäckström (1998) and use an axiomatic planning framework for membership results. We assume there is a *planning problem instance (ppi)* Π and that Π includes a specification of some set of *operators*. A *partial-order (p.o.) plan* for Π is a tuple $P = \langle A, \prec \rangle$, where A is a set of *actions* and $\prec$ is an order on A such that its transitive closure $\prec^+$ is a partial order. An action is an *occurrence* of an operator in Π, inheriting its definition. All actions in A are unique, but two or more actions can be occurrences of the same operator, i.e. they have the same behavior. We further say that P is a *total-order (t.o.) plan* if $\prec^+$ is also a total order. A t.o. plan corresponds to a unique sequence of the actions in A, i.e. a unique execution order. We further assume there is a *validity test* for Π that decides if a total-order plan is a valid solution for Π or not, and we say that a t.o. plan is Π-*valid* if it is a solution for Π. A p.o. plan is implicitly defined as Π-valid if all of its topological sortings are Π-valid. Just like Bäckström (1998), we will assume that validity of p.o. plans can be decided in polynomial time.

For hardness proofs, we will use propositional STRIPS in order to make the results as strong as possible. This formalism satisfies that p.o. plans can be validated in polynomial time (Nebel and Bäckström 1994, Corollary 11). We will use the notation $a : p \to e$ to define that a is an action with precondition p and effect e.

The size of a plan order, $|\prec|$, is defined as the number of relation tuples in it. Bäckström (1998) suggested that it may be beneficial to standardize the plan order to be either a transitive closure or a reductive closure. However, just as in his case, neither is a necessary assumption for our results. A *chain* is a subset $\{a_0, a_1, a_2, \ldots, a_l\}$ of A such that for every $1 \leq i \leq l$ we have $a_{i-1} \prec a_i$. In other words, a chain is a set of actions every two of which are comparable. An *antichain* is a set of actions no two of which are comparable.

4 Least-constrained Plans

For any two partial-order plans $P = \langle A, \prec \rangle$ and $Q = \langle A, \prec' \rangle$, and a PPI Π, we say Q is a *reordering* of P wrt. Π if and only if both P and Q are Π-valid. The MINIMUM-CONSTRAINED REORDERING (MCR) is defined as follows:

> INSTANCE: A PPI Π, a Π-valid partial order plan $P = \langle A, \prec \rangle$ and a positive integer k.
> QUESTION: Does P have a Π-valid reordering $Q = \langle A, \prec' \rangle$ such that $|\prec'| \leq k$?

Deordering of a p.o. plan is a special case of reordering where we require that $\prec' \subseteq \prec$. The problem MINIMUM-CONSTRAINED DEORDERING (MCD) is defined analogously to MCR. Bäckström (1998) showed that both MCD and MCR are **NP**-complete for planning formalisms that have a polynomial-time validity test for p.o. plans.

In this paper we will study both these problems using parameterized complexity, and we will consider the following parameters of the instance:

$n_\prec$	$\lvert\prec\rvert$, i.e. the size of the plan order
$h_\prec$	Height of $\prec$ (size of its longest chain)
$w_\prec$	Width of $\prec$ (size of the largest antichain)

We analogously define the solution parameters $n_{\prec'}$, $h_{\prec'}$ and $w_{\prec'}$, which are the desired values of the new order in the solution. Note that the instance parameters can be determined from the instance, while the solution parameters are values to optimize. Let π be a set of parameters. Then the notation π-MCR refers to the variant of MCR parameterized with the parameters in π. This notation is used analogously also for other problems. Note that we generally want as few parameters as possible for membership results and as many as possible for hardness results. If π-MCR is in a complexity class C and $\pi \subseteq \pi'$, then also π'-MCR is in C, i.e. adding parameters does not make the problem harder. Conversely, if π-MCR is C-hard and $\pi' \subseteq \pi$, then π'-MCR is also C-hard, i.e. removing parameters will not make the problem easier, so more parameters means a stronger result here.

Theorem 1. $\{n_\prec\}$-MCD *is in* **FPT**.

Proof. Let Π be a ppi, $P = \langle A, \prec \rangle$ be a Π-valid p.o. plan and k be an integer. We ask for a deordering of size k. Let $p = n_\prec$ be the parameter. There are at most $\binom{p}{0} + \binom{p}{1} + \ldots + \binom{p}{k} < (k+1)p^k$ different deorderings of P with size at most k. Since $k \leq p$ we have $(k+1)p^k \leq (p+1)p^p$ which is a function of p. By general assumption, Π-validity of each deordering can be verified in polynomial time, so the algorithm runs in time $f(p) \cdot q(n)$, where n is the instance size, f is some function and q is some polynomial. □

We will use the following version of Dilworth's Theorem (Hall 1998)[Thm.7.2.1]:

> **Dilworth's Theorem.** In any finite partially ordered set, the maximum number of elements in any antichain equals the minimum number of chains in any partition of the set into chains.

Observation 1. *The parameter combinations* $\{h_\prec, w_\prec\}$ *and* $\{h_{\prec'}, w_{\prec'}\}$ *are degenerate cases.*

Proof. Let $P = \langle A, \prec \rangle$ be a p.o. plan. According to Dilworth's Theorem there exists a partition of A into $w_\prec$ number of chains. On the other hand, the size of each chain is at most $h_\prec$, therefore we have $|A| \leq w_\prec h_\prec$. So $n_\prec \leq \binom{|A|}{2} \leq \binom{w_\prec h_\prec}{2}$. The same is true for the reordered/deordered partial order $\prec'$. If $|A| > w_{\prec'} h_{\prec'}$ there is no solution and if $|A| \leq w_{\prec'} h_{\prec'}$ every possible reordering/deordering can be checked in polynomial time. □

Now, every reordering/deordering of P can be generated and checked in time $f(w_\prec, h_\prec)$ for some function f, which means they would lie in **FPT** for any parameter combination that includes $\{w_\prec, h_\prec\}$. Although, in this paper we do not consider this as an **FPT** result, but rather a degenerate case and not a useful parameter combination. This also remains correct for all problems discussed in this paper. Note that in every deordering we have $h_{\prec'} \leq h_\prec$ and $w_{\prec'} \geq w_\prec$, whereas in reordering, none of these inequalities are guaranteed. This means that the parameter combination $\{h_\prec, w_{\prec'}\}$ also has this property, but only for plan deordering.

Theorem 2. $\{n_{\prec'}\}$-MCR *is in* **W***[P].*

Proof. Let $P = \langle A, \prec \rangle$ be a partial order Π-valid plan for some PPI Π. Let n be the instance size. Nondeterministically guess a reordering $\prec'$ of P wrt. Π such that $n_{\prec'} \leq k$. We need to guess at most $2k \log n$ bits because every action can be indexed in $\log n$ bits. Π-validity of $\langle A, \prec' \rangle$ is polynomial time by assumption. □

The following corollary is straightforward since in MCR we always have $n_{\prec'} \leq n_\prec$:

Corollary 1. $\{n_\prec\}$-MCR *and* $\{n_{\prec'}\}$-MCD *are in* **W***[P].*

Next we will use the DOMINATING SET problem which is **W**[2]-complete when parameterized with the set size (Downey and Fellows 1995):

> INSTANCE: A graph $G = (V, E)$ and an integer k.
> PARAMETER: k.
> QUESTION: Is there a set of k vertices $V' \subseteq V$ such that every vertex of G either belongs to V' or has a neighbor in V'?

Construction 1. *Let* $G = (V, E)$ *be an arbitrary graph such that* $V = \{v_1, \ldots, v_n\}$ *and let* $N(v)$ *denote the set of neighbors of vertex* v*. Construct a* PPI $\Pi = \langle V \cup \{g\}, A, \varnothing, \{g\} \rangle$ *and a* Π*-valid partial order plan* $P = \langle A, \prec \rangle$ *where* $A = \{a_1, a_2, \ldots, a_n, a\}$ *and for every* $i \in \{1, \ldots, n\}$, $a_i \prec a$*. Define* $a_i : \varnothing \to N(v_i) \cup \{v_i\}$ *and* $a : V \to g$*.*

The following construction is similar to Construction 1 with the difference that P is defined as a total order plan.

Construction 2. *Let* G*,* V*,* N*,* Π *and* A *be defined the same as Construction 1. Construct* $P = \langle A, \prec \rangle$*, a* Π*-valid total order plan such that for each* $i \in \{1, \ldots, n-1\}$, $a_i \prec a_{i+1}$ *and* $a_n \prec a$*.*

Theorem 3. *The following problems are* **W***[2]-hard:*

1. $\{n_{\prec'}, h_\prec, h_{\prec'}\}$-MCD
2. $\{n_{\prec'}, w_\prec, h_{\prec'}\}$-MCD
3. $\{n_{\prec'}, h_\prec, h_{\prec'}\}$-MCR
4. $\{n_{\prec'}, w_\prec, h_{\prec'}\}$-MCR

Proof. 1. By a reduction from DOMINATING SET using Construction 1. Each action a_i corresponds to adding vertex v_i to the dominating set and action a is a validity check on the set. Thus, it is easy to observe that G has a dominating set of size k if and only if P has a deordering $P' = \langle A, \prec' \rangle$ of size k. It can also be seen that $h_\prec = h_{\prec'} = 2$.
2. Similar to part (1) but instead use Construction 2. Note that $w_\prec = 1$ and $h_{\prec'} \leq k + 1$.
3. By using Construction 1. Similar to part (1), if G has a dominating set of size k, then P has a reordering of size k. For the other direction, assume $P' = \langle A, \prec' \rangle$ be a Π-valid reordering of size k. Since a is the only action that sets g, every topological sorting of P' must be of the form $\langle \Omega, a, \Omega' \rangle$ where $\{\Omega, \Omega'\}$ is a partition of $\{a_1, \ldots, a_n\}$. Among all topological sortings of P', let $\langle \omega, a, \omega' \rangle$ be the one with fewest number of actions preceding a ($\omega = \arg\min|\Omega|$) and let $\omega = \langle a_{\pi_1}, a_{\pi_2}, \ldots, a_{\pi_l} \rangle$. For every $1 \leq i \leq l$ we must have a list of actions $\{x_1, \ldots, x_j\}$ such that $a_{\pi_i} \prec' x_1 \prec' \ldots \prec' x_j \prec' a$, otherwise ω did not contain a_{π_i}. Therefore $l \leq n_{\prec'} \leq k$. Now since the preconditions of a must be satisfied by actions in ω, $V \subseteq \bigcup_{1 \leq i \leq l} (N(v_{\pi_i}) \cup \{v_{\pi_i}\})$ which means that $\{v_{\pi_1}, v_{\pi_2}, \ldots, v_{\pi_l}\}$ is dominating set for G. We have $h_\prec = 2$ and $h_{\prec'} \leq k + 1$.
4. Similar to part (3) but instead use Construction 2. Also, $w_\prec = 1$ and $h_{\prec'} \leq k + 1$. □

Note that the parameter $h_{\prec'}$ is always redundant in case 1, since we always have $h_{\prec'} \leq h_\prec$ in a deordering. By removing the non-constant parameters from the above theorem, we get the following corollary:

Corollary 2. *The following problems are para-**NP**-hard:*

1. $\{h_\prec, h_{\prec'}\}$-MCD
2. $\{w_\prec\}$-MCD
3. $\{h_\prec\}$-MCR
4. $\{w_\prec\}$-MCR

5 Parallel Plans

A *parallel plan* (pplan) is a tuple $P = \langle A, \prec, \# \rangle$ where $\langle A, \prec \rangle$ is a p.o. plan and $\#$ is an irreflexive and symmetric relation on A. $\#$ is called the *non-concurrency* relation. For every two actions a and b, $a\#b$ indicates that they cannot be executed in parallel. We define the size of the non-concurrency relation $n_\#$ as the number of pairs $a, b \in A$ s.t. $a\#b$. We also define the non-concurrency graph $G_\# = (A, E)$ where $(a, b) \in E$ if and only if $a\#b$. Let $d : A \mapsto \mathbb{N}$ denote the duration of each action. A *parallel execution* of a pplan P is a function $r : A \mapsto \mathbb{N}$, denoting the release time for every action in A, such that: 1) If $a \prec b$ then $r(a) + d(a) \leq r(b)$, and 2) If $a\#b$ then either $r(a) + d(a) \leq r(b)$ or $r(b) + d(b) \leq r(a)$. The length of a parallel execution or $\max_{a \in A}\{r(a) + d(a)\}$ denotes the latest finishing time of any action. We define the *parallel length* of a pplan P to be the minimum length over all possible parallel executions of P. In this paper, for simplicity, we assume that the duration of every action is one.

The problem PARALLEL PLAN LENGTH (PPL) is defined as follows:

INSTANCE: A PPI Π, a Π-valid parallel plan $P = \langle A, \prec, \# \rangle$ and a positive integer l_p.
QUESTION: Does P have a parallel execution of length at most l_p?

PPL is **NP**-complete (Bäckström 1998). We will now analyze the parameterized complexity of PPL considering the following parameters:

l_p	Length of the parallel plan
$n_\#$	Size of the non-concurrency relation
$\Delta_\#$	Maximum degree of $G_\#$

Lemma 1. *The parallel length of the pplan $P = \langle A, \prec, \varnothing \rangle$ is equal to $h_\prec$.*

Proof. Let k be the parallel length of P. Trivially, $h_\prec \leq k$. On the other hand, a greedy execution of actions (at each iteration, execute all actions a s.t. every action b that $b \prec a$ is already executed) shows that $k \leq h_\prec$ so, $k = h_\prec$. □

Lemma 2. *$h_\prec$ can be computed in polynomial time.*

Proof. The plan order is a DAG and computing $h_\prec$ is equivalent to finding the longest path in it. □

Theorem 4. *$\{n_\#\}$-PPL is in* **FPT**.

Proof. Let Π be a PPI and $P = \langle A, \prec, \# \rangle$ a Π-valid pplan. Let a and b be two actions such that $a\#b$. Without loss of generality, assume in the optimal parallel execution of P, a is executed earlier than b, i.e. $r(a) + d(a) \leq r(b)$. If we remove $a\#b$ from the non-concurrency relation and instead, add $a \prec b$ to the partial order, the minimum parallel length of the new pplan will be the same. Therefore, for every $a\#b$ it is enough to check both $a \prec b$ and $b \prec a$ and then the minimum parallel length of P will be the minimum length over all $2^{n_\#}$ generated pplans. Each one of these pplans has an empty non-concurrency relation. Therefore, with respect to Lemmas 1 and 2 this is an fpt algorithm. □

We will continue by showing that PPL and graph coloring are highly connected. GRAPH K-COLORING (CHROMATIC NUMBER) is defined as follows:

INSTANCE: Graph $G = (V, E)$ and a positive integer $k \leq |V|$.
QUESTION: Is G k-colorable, i.e. does there exist a function $f : V \to \{1, 2, \ldots, k\}$ such that $f(v) \neq f(u)$ whenever $(v, u) \in E$?

GRAPH K-COLORING is **NP**-complete even for $k = 3$ and planar graphs having no vertex degree exceeding 4 (Garey and Johnson 1979)[GT4]. Let $\chi_\#$ denote the chromatic number of $G_\#$.

Lemma 3. *The parallel length of the pplan $P = \langle A, \varnothing, \# \rangle$ is equal to $\chi_\#$.*

Proof. Let Π be a PPI and let $P = \langle A, \varnothing, \# \rangle$ be a Π-valid pplan. Let l_p be the parallel length of P. Also let $f : A \to \{1, 2, \ldots, k\}$ be a k-coloring for $G_\#$. If we define the release time of every action equal to its color ($r(a) = f(a)$), then clearly this is a valid parallel execution of P because no two actions $a\#b$ have the same color (= release time). So, $l_p \leq k$. Similarly, by assigning the release time of actions as colors, every parallel execution gives a valid coloring. Therefore, $l_p \geq k$ and hence, $l_p = k$. □

Theorem 5. *Let l_p denote be the parallel length of the pplan $P = \langle A, \prec, \# \rangle$. Then*

$$\max\{h_\prec, \chi_\#\} \leq l_p \leq h_\prec + n_\#$$

Proof. By ignoring $\#$ we get $h_\prec \leq l_p$ and by ignoring $\prec$ we get $\chi_\# \leq l_p$ (Lemmas 1 and 3). So, $\max\{h_\prec, \chi_\#\} \leq l_p$. To prove the upper bound, first assume that the non-concurrency relation is empty, then using Lemma 1, P has a parallel execution of length $h_\prec$. Now in the current parallel execution, for every two actions a and b that $a\#b$ and $r(a) = r(b)$, we add a new iteration to the parallel execution and shift action a and every action with a later release time one unit forward, i.e. $r(a) := r(a) + 1$ and also for every x such that $r(x) > r(a)$, $r(x) := r(x) + 1$. By repeating this process at most $n_\#$ times it is guaranteed that no two such a and b actions will be found. Therefore, we always have $l_p \leq h_\prec + n_\#$. □

The upper bound in the above theorem is achievable: let Π be a PPI, $P = \langle A, \prec, \# \rangle$ a Π-valid pplan and $A = \bigcup_{1 \leq i \leq n} \{a_i, b_i\}$. For each i, let $a_i \prec a_{i+1}$, $a_i \prec b_{i+1}$, $b_i \prec a_{i+1}$, $b_i \prec b_{i+1}$ and $a_i \# b_i$. For every two actions a_i and a_j where $i < j$ we have $a_i \prec a_j$; and for every two actions a_i and b_j where $i \leq j$, we either have $a_i \prec b_j$ $(i < j)$ or $a_i \# b_j$ $(i = j)$. Therefore no two actions in A can be executed in parallel and hence the parallel length for this pplan is $|A| = 2n$. On the other hand, the partial order contains two chains ($a_1 \prec \ldots \prec a_n$ and $b_1 \prec \ldots \prec b_n$) of length n. So, the parallel length is equal to the upper bound in Theorem 5, $h_\prec + n_\# = n + n = 2n$.

Theorem 6. $\{n_\prec, l_p, \Delta_\#, h_\prec\}$-PPL *is para-**NP**-hard.*

Proof. By a reduction from the **NP**-complete problem GRAPH 3-COLORING to the slice $\{n_\prec = 0, l_p = 3, \Delta_\# = 4, h_\prec = 1\}$-PPL. Let $G_\# = (V, E)$ be an arbitrary planar graph with vertex degree at most 4 and let $V = \{v_1, v_2, \ldots, v_n\}$. Construct a PPI $\Pi = \langle V, A, \varnothing, V \rangle$ and a parallel plan $P = \langle A, \varnothing, \# \rangle$ where $A = \{a_1, a_2, \ldots, a_n\}$ and $a_i : \overline{v_i} \rightarrow v_i$. Define $a_i \# a_j$ if and only if $(v_i, v_j) \in E$. It is easy to see that P is a Π-valid plan and that G has a 3-coloring if and only if P has a parallel execution of size 3 or less. Note that if two actions can be executed in the same iteration of the parallel execution, their corresponding vertices can be colored with the same color and vice versa. We finally note that $n_\prec = 0$ and $h_\prec = 1$ since the plan order is empty. □

6 Reordering Parallel Plans

In this section we investigate the optimization of pplans by reordering and deordering. To do so, we define MINIMUM PARALLEL REORDERING (MPR):

> INSTANCE: A PPI Π, a Π-valid parallel plan $P = \langle A, \prec, \# \rangle$ and a positive integer l_p.
> QUESTION: Does P have a Π-valid reordering with a parallel execution of length at most l_p?

MINIMUM PARALLEL DEORDERING (MPD) is defined similarly. Both problems are **NP**-complete if p.o. plans can be validated in polynomial time (Bäckström 1998).

Theorem 7. $\{n_\prec, n_\#\}$-MPD *is in* **FPT**.

Proof. A combination of Theorems 1 and 4: We check $2^{n_\#}$ new partial orders gained from removing the non-concurrency relation. Each new partial order has size at most $p = n_\prec + n_\#$. There are at most $f(p) = (p+1)p^p$ different deorderings of each partial order and Π-validity check can be done in polynomial time for each of them. So, the final algorithm runs in time $g(n_\prec, n_\#)q(n)$ for some function g, some polynomial q and instance size n. □

Theorem 8. $\{n_{\prec'}, n_\#\}$-MPR *is in* **W***[P]*.

Proof. Let (Π, P, l_p) be an instance of MPR where Π is a PPI and $P = \langle A, \prec, \# \rangle$ is a partial order Π-valid plan. Let n be the instance size. Nondeterministically, guess a reordering $\prec'$ of P wrt. Π. Since every action can be indexed in $\log n$ bits, we need to guess at most $2n_{\prec'} \log n$ bits. To check the optimal parallel execution length of $\langle A, \prec', \# \rangle$, similar to the proof of Theorem 4 we check all possibilities for each pair of actions $a\#b$. This can be done in time $O(2^{n_\#} p(n))$ for some polynomial p. □

Corollary 3. $\{n_{\prec'}, n_\#\}$-MPD *is in* **W***[P]*.

Theorem 9. *For every set of partial order parameters δ that contains $n_{\prec'}$, the following fpt reduction holds*

$$\delta\text{-MCD} \leq_{fpt} \delta \cup \{n_\#, l_p\}\text{-MPD}$$

Proof. Let $P = \langle A, \prec \rangle$ be a Π-valid p.o. plan for some PPI Π, and let δ be a set of parameters including $n_{\prec'}$. Let $Q = \langle A, \prec' \rangle$ be any deordering of P. Define the Π-valid pplan $Q' = \langle A, \prec', \varnothing \rangle$. Q' is a deordering of P of size $n_{\prec'}$. According to Lemma 1, the parallel length of Q' is $l_p = h_{\prec'} \leq n_{\prec'} + 1$. This proves an fpt reduction from δ-MCD to $\delta \cup \{n_\#, l_p\}$-MPD. □

Similarly, we can have a reduction for reordering and then combined with Theorem 3:

Corollary 4. *For $\delta = \{n_\#, l_p, n_{\prec'}\}$, the following problems are* **W***[2]-hard:*

1. $\delta \cup \{h_\prec, h_{\prec'}\}$-MPD
2. $\delta \cup \{w_\prec, h_{\prec'}\}$-MPD
3. $\delta \cup \{h_\prec, h_{\prec'}\}$-MPR
4. $\delta \cup \{w_\prec, h_{\prec'}\}$-MPR

Theorem 10. δ-MPR *and* δ-MPD *are para-**NP**-hard for* $\delta = \{n_\prec, n_{\prec'}, l_p, \Delta_\#, h_\prec, h_{\prec'}\}$.

Proof. Similar to proof of Theorem 6 by a reduction from GRAPH 3-COLORING to a slice of MPR with $n_\prec = n_{\prec'} = 0$, $h_\prec = h_{\prec'} = 1$, $l_p = 3$ and $\Delta_\# = 4$. □

7 Parallel k-Processor Plans

PPL presumes the access to an infinite number of processors, or agents. However, this is not a realistic assumption. For instance, in multi-agent planning there may be fewer agents available than the number of actions that can be executed in parallel. A similar case may arise in web service composition, where the number of processors is limited. In this section we define PPL$_k$, a variation of PPL limited to k processors (or agents). But we need to first, define the k-processor parallel execution of a pplan.

Let k be an integer. A *k-processor parallel execution* of a pplan is a parallel execution such that for each time $t \in \mathbb{N}$ there are at most k actions a such that $r(a) \leq t < r(a) + d(a)$. The *k-processor parallel length* of a pplan is the minimum length over all possible k-processor parallel executions of it. PARALLEL K-PROCESSOR PLAN LENGTH (PPL$_k$) has the following definition:

INSTANCE: A PPI Π, a Π-valid parallel plan $P = \langle A, \prec, \# \rangle$, a positive integer l_p and a number of processors k.
QUESTION: Does P have a k-processor parallel execution of length at most l_p?

The parameters we consider for PPL$_k$ are the same parameters we used for PPL, i.e. $n_\prec$, $h_\prec$, $w_\prec$, l_p, $n_\#$ and $\Delta_\#$, plus k, the number of processors.

Theorem 11. $\{l_p, n_\#, k\}$-PPL$_k$ *is in* **FPT**.

Proof. For every pair of actions a and b such that $a \# b$, we check both $a \prec b$ and $b \prec a$ which leads to checking all $2^{n_\#}$ different cases. There are at most $l_p \cdot k$ slots available for each action to be executed. Therefore, if $|A| > l_p k$ there is no solution. So, $|A| \leq l_p k$ and there are at most $O((l_p k)^{|A|}) = O((l_p k)^{l_p k})$ different schedules. Checking each schedule can be done in time $O(|A| + |\prec|) = O(|A|^2)$. In total we need time $O(2^{n_\#}(l_p k)^{l_p k})$. □

A special case of PPL$_k$ where each action has the same duration and the non-concurrency relation is an empty set, is usually referred to as Precedence Constrained Scheduling. PRECEDENCE CONSTRAINED SCHEDULING (PCS) is an **NP**-complete problem (Garey and Johnson 1979)[SS9] and it becomes **W**[2]-hard if parameterized by the number of processors (k) (Bodlaender and Fellows 1995):

INSTANCE: A set T of unit-length tasks, a partial order $\prec$ on T, a positive integer deadline D and a number of processors k.
PARAMETER: k.
QUESTION: Does there exist a k-processor schedule for T that meets the deadline D and obeys the precedence constraints, i.e., Does there exist a map $f : T \mapsto \{1, 2, \ldots, D\}$ such that for all $t, t' \in T$, $t \prec t'$ implies $f(t) < f(t')$, and for all i, $1 \leq i \leq D$, $|f^{-1}(i)| \leq k$?

PCS remains **NP**-complete for $D = 3$. (Lenstra and Kan 1978)

Theorem 12. $\{k\}$-PPL$_k$ *is* **W***[2]-hard.*

Proof. Proof by reduction from PRECEDENCE CONSTRAINED SCHEDULING. Let $\langle T, \prec, D, k \rangle$ be a scheduling instance and let $T = \{t_1, \ldots, t_n\}$. Construct a PPI $\Pi = \langle T, A, \varnothing, T \rangle$ and a parallel plan $P = \langle A, \prec, \varnothing \rangle$ where $A = \{a_1, \ldots, a_n\}$ and for every i, $a_i : \varnothing \to t_i$, and let $l_p = D$. Trivially, P is a Π-valid plan, otherwise $\prec$ would not be a valid partial order. It is easy to see that there exists a k-processor parallel execution of P of length l_p if and only if there exists a k-processor schedule for T that meets the deadline $D = l_p$. □

Theorem 13. $\{n_\#, \Delta_\#, l_p\}$-PPL$_k$ *is para-***NP***-hard.*

Proof. By a reduction from the **NP**-complete problem PRECEDENCE CONSTRAINED SCHEDULING to the slice $\{n_\# = 0, \Delta_\# = 0, l_p = 3\}$-PPL$_k$. From an arbitrary scheduling instance $(T, \prec, D = 3, k)$, build a PPI Π and a Π-valid pplan $P = \langle A, \prec, \varnothing \rangle$ similar to the proof of Theorem 12 and let $l_p = D = 3$. Then P has a k-processor parallel execution of length $l_p = 3$ if and only if T has a k-processor schedule that meets the deadline $D = 3$. □

The above result is very interesting in comparison to Theorem 4 and also complies well with the intuition that PPL bounded to a certain number of processors is much harder than the unbounded case.

8 Discussion

While $\{n_\prec, n_\#\}$-MPD is in **FPT**, we only know that $\{n_\prec, n_\#\}$-MPR is in **W**[P] (it follows from Theorem 8). It remains an open question to find a more precise complexity classification of the latter problem. It is, however, not obviously as easy as $\{n_\prec, n_\#\}$-MPD, since this problem has a much more restricted set of potential solutions; the number of deorderings is bounded in $n_\prec$, while the number of reorderings is not bounded in $n_\prec$ in the general case.

We can consider other parameters for PPL, like the clique number ($c_\#$) or the chromatic number ($\chi_\#$). But since

$$c_\# \leq \chi_\# \leq \Delta_\# + 1 \leq n_\# + 1$$

the hardness results for $\Delta_\#$ are stronger than for both $c_\#$ and $\chi_\#$. We could also consider parameters that capture more structure of the graph $G_\#$, for instance its treewidth, a property that has been recurringly successful as a parameter for many other problems. It is probably even more interesting to identify structural properties of both $\prec$ and $\#$ considered together.

In deordering we always have $n_{\prec'} \leq n_\prec$, whether if we optimize $n_{\prec'}$ or l_p. However, in reordering it is different. When we optimize $n_{\prec'}$ (MCR) we must have $n_{\prec'} \leq n_\prec$, otherwise the answer is automatically yes; and when we optimize l_p (MPR), $n_{\prec'} > n_\prec$ is also allowed.

There is a high correlation between PPL, PPL$_k$ and scheduling problems, specially in the absence of the non-concurrency relation. Finding $h_\prec$ is similar to critical path scheduling. Ullman (1975) showed that PCS remains **NP**-hard even on 2 processors if tasks of length 1 and 2 are allowed. This means we can add parameter k to the result in Theorem 13 if we allow task lengths 1 and 2. Mnich and Wiese (2015) applied parameterized complexity on scheduling problems and discovered several areas of fixed-parameter tractability. van Bevern et al. (2016) showed that PCS parameterized by partial-order width is **W**[2]-hard even on 2 processors if tasks of length 1 and 2 are allowed. One obvious way to continue the work is to consider the k-processor parallel length as a criterion for plan optimization.

Acknowledgments

Aghighi is partially supported by the National Graduate School in Computer Science (CUGS), Sweden. Bäckström is partially supported by the Swedish Research Council (VR) under grant 621-2014-4086.

References

Aghighi, M., and Bäckström, C. 2015. Cost-optimal and net-benefit planning - A parameterised complexity view. In *Proc. 24th Int'l Joint Conf. Artif. Intell. (IJCAI 2015), Buenos Aires, Argentina*, 1487–1493.

Bäckström, C., and Jonsson, P. 2011. All PSPACE-complete planning problems are equal but some are more equal than others. In *Proc. 4th Ann. Symp. Combinatorial Search (SOCS 2011), Castell de Cardona, Barcelona, Spain*.

Bäckström, C.; Jonsson, P.; Ordyniak, S.; and Szeider, S. 2015. A complete parameterized complexity analysis of bounded planning. *J. Comput. Syst. Sci.* 81(7):1311–1332.

Bäckström, C. 1998. Computational aspects of reordering plans. *J. Artif. Intell. Res.* 9:99–137.

Bodlaender, H. L., and Fellows, M. R. 1995. W[2]-hardness of precedence constrained k-processor scheduling. *Oper. Res. Lett.* 18(2):93–97.

Cimatti, A.; Micheli, A.; and Roveri, M. 2015. Strong temporal planning with uncontrollable durations: A state-space approach. In *Proc. 29th AAAI Conf. Artif. Intell., Austin, Texas, USA.*, 3254–3260.

de Haan, R.; Roubícková, A.; and Szeider, S. 2013. Parameterized complexity results for plan reuse. In *Proc. 27th AAAI Conf. Artif. Intell. (AAAI-2013), Bellevue, WA, USA*.

Downey, R. G., and Fellows, M. R. 1995. Fixed-parameter tractability and completeness i: Basic results. *SIAM Journal on Computing* 24(4):873–921.

Downey, R. G., and Fellows, M. R. 1999. *Parameterized Complexity*. Monographs in Computer Science. Springer.

Flum, J., and Grohe, M. 2006. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer.

Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.

Hall, M. 1998. *Combinatorial theory*, volume 71. John Wiley & Sons.

Haslum, P. 2012. Narrative planning: Compilations to classical planning. *J. Artif. Intell. Res.* 44:383–395.

Kambhampati, S., and Kedar, S. 1994. A unified framework for explanation-based generalization of partially ordered and partially instantiated plans. *Artif. Intell.* 67(1):29–70.

Kronegger, M.; Ordyniak, S.; and Pfandler, A. 2015. Variable-deletion backdoors to planning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, 3305–3312.

Kronegger, M.; Pfandler, A.; and Pichler, R. 2013. Parameterized complexity of optimal planning: A detailed map. In *Proc. 23rd Int'l Joint Conf. Artif. Intell. (IJCAI 2013), Beijing, China*, 954–961.

Kvarnström, J. 2011. Planning for loosely coupled agents using partial order forward-chaining. In *Proc. 21st Int'l Conf. Automated Planning and Scheduling (ICAPS 2011), Freiburg, Germany*.

Lenstra, J. K., and Kan, A. H. G. R. 1978. Complexity of scheduling under precedence constraints. *Operations Research* 26(1):22–35.

Madhusudan, T., and Uttamsingh, N. 2006. A declarative approach to composing web services in dynamic environments. *Decision Support Systems* 41(2):325–357.

Maliah, S.; Shani, G.; and Stern, R. 2016. Collaborative privacy preserving multi-agent planning. *Auton. Agent. Multi-Agent Syst.* 1–38.

Mnich, M., and Wiese, A. 2015. Scheduling and fixed-parameter tractability. *Math. Program.* 154(1-2):533–562.

Muise, C. J.; McIlraith, S. A.; and Beck, J. C. 2011. Monitoring the execution of partial-order plans via regression. In *Proc. 22nd Int'l Joint Conf. Artif. Intell. (IJCAI 2015), Barcelona, Catalonia, Spain*, 1975–1982.

Muise, C. J.; McIlraith, S. A.; and Beck, J. C. 2012. Optimally relaxing partial-order plans with maxsat. In *Proc. 22nd Int'l Conf. Automated Planning and Scheduling, (ICAPS 2012), Atibaia, São Paulo, Brazil*.

Nebel, B., and Bäckström, C. 1994. On the computational complexity of temporal projection, planning, and plan validation. *Artif. Intell.* 66(1):125–160.

Pannek, J. 2013. Parallelizing a state exchange strategy for noncooperative distributed NMPC. *Systems & Control Letters* 62(1):29–36.

Régnier, P., and Fade, B. 1991. Complete determination of parallel actions and temporal optimization in linear plans of action. In *Proc. European Workshop on Planning, (EWSP '91), Sankt Augustin, FRG*, 100–111.

Rodríguez-Moreno, M. D.; Borrajo, D.; Cesta, A.; and Oddi, A. 2007. Integrating planning and scheduling in workflow domains. *Expert Syst. Appl.* 33(2):389–406.

Say, B.; Ciré, A. A.; and Beck, J. C. 2016. Mathematical programming models for optimizing partial-order plan flexibility. In *Proc. 22nd European Conf. Artif. Intell. (ECAI 2016), The Hague, The Netherlands*, 1044–1052.

Scala, E., and Torasso, P. 2015. Deordering and numeric macro actions for plan repair. In *Proc. 24th Int'l Joint Conf. Arfif. Intell. (IJCAI-2015), Buenos Aires, Argentina*, 1673–1681.

Siddiqui, F. H., and Haslum, P. 2015. Continuing plan quality optimisation. *J. Artif. Intell. Res.* 54:369–435.

Ullman, J. D. 1975. Np-complete scheduling problems. *J. Comput. Syst. Sci.* 10(3):384–393.

van Bevern, R.; Bredereck, R.; Bulteau, L.; Komusiewicz, C.; Talmon, N.; and Woeginger, G. J. 2016. Precedence-constrained scheduling problems parameterized by partial order width. In *Discrete Optimization and Operations Research - 9th International Conference, DOOR 2016, Vladivostok, Russia, September 19-23, 2016, Proceedings*, 105–120.

Veloso, M.; Pérez, A.; and Carbonell, J. 1990. Nonlinear planning with parallel resource allocation. In *Workshop on Innovative Approches to Planning, Scheduling and Control, San Diego, CA, USA, 1990*, 207–212.