Repairing Schedules by Removing Waiting Times: A Parameterized Complexity Analysis

Niels Grüttemeier
Fraunhofer IOSB-INA, Lemgo, Germany

Klaus Heeger
Fraunhofer IOSB-INA, Lemgo, Germany

Abstract

We consider the problem of repairing production schedules in a job-shop setting by reducing pre-planned waiting times. Herein, a schedule of all jobs is given. To compensate unforeseen disturbances, this schedule contains waiting times between the execution of two consecutive tasks of a job. Further, we assume that the schedule temporarily overloads some machines, e.g. due to reduced machine capacities because of worker sickness or (partially) broken machines. We study the problem of removing as few waiting times as possible in order to eliminate the machine overloads. After formalizing this problem, we perform an extensive analysis of its parameterized complexity with respect to several natural parameters, resulting in a detailed picture of the problem's complexity.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms; Theory of computation → Problems, reductions and completeness; Mathematics of computing → Combinatorics

Keywords and phrases Job shop, parallel machines, reactive scheduling

Digital Object Identifier 10.4230/LIPIcs.WADS.2025.31

Funding *Niels Grüttemeier*: Supported by the project *Datenfabrik.NRW*, a project by *KI.NRW*, funded by the Ministry for Economics, Innovation, Digitalization and Energy of the State of North Rhine-Westphalia (MWIDE).
Klaus Heeger: Supported by the German Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (BMUV) under the project *Smart-E-Factory*.

1 Introduction

Scheduling is a crucial task in the industrial production of goods, helping to achieve key goals such as low production costs or on-time deliveries. Therefore, scheduling is an established and very important area in the intersection of theoretical computer science, operations research, and combinatorial optimization. In classical scheduling problems, one is usually given a set of jobs (and their characteristics) to be processed in the future, and the task is to compute an "optimal" (with respect to some specified criterium) schedule of the jobs. However, many real-world scenarios are not adequatly covered by this approach, for instance when a large number of unexpected obstacles (e.g., machines breaking down, workers calling in sick, arrival of urgent customer orders, etc.) may compromise the initial plan. Recomputing a new schedule from scratch after each such obstacle is hardly an option in practice, as reassignments of jobs and operations to different resources require cancellations of expensive set-ups and lead to shop floor nervousness [21].

Instead of rescheduling, there are also other ways to handle unexpected obstacles in production planning. One such approach is scheduling under uncertainties, containing several different models of uncertainties in scheduling. One of the most common such models is stochastic scheduling, where the values of job parameters (most commonly of the processing time) are independent random variables instead of numbers, see e.g. [8, 19]. Another model is multi-scenario scheduling, where a finite number of different scenarios needs to be considered

t	1	2	3	4	5	6	7	8
A	3	2	2	2	2	2	0	1
B	1	1	2	0	2	2	1	1
1					A	x	x	B
2					A	B		
3			A	x	A	B		
4	A	x	x	A	B			
5	A	A	x	A	B			

→

t	1	2	3	4	5	6	7	8
A	3	2	2	2	2	2	0	1
B	1	1	2	0	2	2	1	1
1					A	x	x	B
2					A	B		
3			A	A	B			
4	A	A	B					
5	A	A	x	A	B			

Figure 1 An example of an instance of WTR (left) together with a repaired schedule (right). There are time steps $t = 1, \ldots, 8$ and jobs $1, \ldots, 5$ that are processed on two machines A and B. The upper part shows the machine capacities at the time steps, the lower part shows the jobs aligned at their starting time steps. At time step $t = 5$, the capacity constraints are violated, since three jobs are processed on machine A. Removing the grey marked symbols x in Job 3 and Job 4 converts the schedule into a schedule satisfying all capacity constraints.

(see [18] for a survey). Another approach is online scheduling, where the machine breakdowns appear one after another and one has to update the schedule immediately, see e.g. [2, 10]. Another approach is to *repair* a schedule. In this approach, one starts with a given schedule and applies small changes to it, adapting the schedule to the new situation. This is done by heuristically applying generic operations on the given schedule [20, 21] or by applying data-driven techniques from the field of machine learning [14, 16, 22]. Relatively new approaches combine scheduling under uncertainties with repairing policies [13]. For a very recent survey of scheduling problems in general, we refer to Agnetis et al. [1].

While the vast majority of research on repairing schedules focusses on heuristic approaches and their experimental evaluation, this work aims to lay a foundation for the theoretical study of these problems. In order to do so, we first formalize our scheduling problem as a precise mathematical problem which we call Waiting Time Removal (WTR) using notations from stringology. The idea of WTR is that in a case where unexpected resource breakdowns occur, updated information about resource availability and job durations is perfectly known and one has to adapt the old solution to the new situation. WTR is motivated by real-world production planning scenarios in German mid-sized companies. In this work, we study the computational complexity of WTR, focussing on the parameterized complexity.

An example of an instance of WTR together with a repaired schedule is illustrated in Figure 1. Intuitively, our scheduling problem WTR models the following scenario: there are multiple types of different machines (e.g. machine types A and B). Similar to a job shop, each job has to be processed on machines of different types in a job-specific order for a specific amount of time. However, in contrast to the standard job shop model, there may be waiting times planned between the steps, serving as a "buffer" to cover unexpected delays. To model this, each job is represented by a string enlisting the required machines (repeating a machine if it needs to be processed for multiple time units) and the planned waiting times. As an example, the string $AAxB$ corresponds to a job that first needs to be processed on machine A for two time units, then has one time unit waiting time (encoded by the letter "x"), and finally needs to be processed on machine B for one time unit. At each time step, a different number of machines of each type is available (e.g. at time $t = 1$, three machines of type A might be available, meaning that up to three different jobs can be processed on machines of type A); this number will be called the *capacity* of the machine at that time step. The capacities model e.g. the breakdown of machines or workers required to operate the machine calling in sick. Additionally, each job is assigned with a starting time. The goal is to remove a small number of waiting times from the jobs such that the resulting

Table 1 Our results. Each cell corresponds to parameterization by the sum of the two corresponding values. The diagonal cells correspond to parameterization by n, k, $|\Sigma|$, or T alone.

	n	k	$\lvert\Sigma\rvert$	T
n	XP (Thm. 4) W[1]-h (Thm. 3)			
k	FPT (Cor. 5) No Poly Ker. (Cor. 10)	XP (Obs. 2) W[2]-h (Cor. 9)		
$\lvert\Sigma\rvert$	XP (Thm. 4) W[1]-h (Thm. 3)	XP (Obs. 2) W[2]-h (Cor. 9)	Para NP-h (Thm. 3)	
T	FPT (Obs. 1) Poly Kernel (Obs. 1)	XP (Obs. 2) W[1]-h (Thm. 11)	FPT (Thm. 7) No Poly Ker. (Thm. 8)	Para NP-h (Thm. 6)

schedule satisfies the capacity constraints posed on the machine types in the all steps. The idea behind keeping the number of waiting time removals small is that maintaining many waiting times increases the robustness against future disruptions.

Related Work. To the best of our knowledge, our introduced problem WTR has not yet been studied.

One framework that is conceptually related to the idea of repairing given schedules is the approach of *parameterized local search*: The idea is to apply a limited number of changes on a given schedule to improve a target function value. Balzereit et al. [3] study parameterized local search for a single machine scheduling problem. While their approach mostly serves as a subroutine in a hill-climbing setting, it can also be used to repair schedules.

Another framework related to repairing schedules is *reoptimization* [4, 17]: The idea of reoptimization is to compute approximate solutions of perturbed instances from given optimal solutions of an original instance. There are constant factor approximations for reoptimization settings of multi-machine makespan minimization under the constraint that some jobs are not allowed to be scheduled parallel [17]. Boria and Della Croce [4] provide constant factor approximizations for multiple min-sum scheduling problems with release dates.

Our Results. We initiate the theoretical study of WTR. As WTR is NP-hard even if the number of allowed waiting time removals is unbounded (Theorem 3), the problem is inapproximable, motivating our parameterized complexity analysis. Considering the most natural parameterizations (by the number n of jobs, the number k of deletions, the time horizon T, and the alphabet size $|\Sigma|$ and combinations thereof), we derive a complete picture of the parameterized complexity of WTR. An overview of the results is shown in Table 1.

Notably, while most of our hardness results hold already for alphabets of constant size, the paraNP-hardness of WTR parameterized by T (Theorem 6) is contrasted with the fixed-parameter tractability with respect to $T + |\Sigma|$ (Theorem 7). A further notable result is that parameterizing with the number n of jobs is not sufficient to ensure fixed-parameter tractability (Theorem 3), but only results in an XP-algorithm (Theorem 4).

2 Preliminaries

We use standard notations from parameterized complexity [5].

For integers $a \leq b$, we let $[a, b] := \{i \in \mathbb{N} \mid a \leq i \leq b\}$. We let $\circ$ denote the concatenation of strings, and we let $|w|$ denote the length of a string w. Furthermore, for $i \in \mathbb{N}$, we let $w[i]$ be the ith symbol of w or the empty word if $i \notin [1, |w|]$.

Let Σ be a finite alphabet, let $x \notin \Sigma$, and let $w \in (\Sigma \cup \{x\})^*$ be a string with $m \in \mathbb{N}$ occurrences of the symbol x. For $R \subseteq [1, m]$, we let $w - R$ denote the string after removing the ith occurrences of x for each $i \in R$ from w. As an example, consider $axbbxcx - \{1, 3\} = abbxc$, where we remove the first and the third occurrence of x while keeping the second occurrence of x. Throughout this work, we call such set R an *x-removal set*.

Problem Definition. Recall that we aim to study a problem where the goal is to adjust a given plan by removing waiting times. In the following, we formalize this idea.

Let Σ be a finite alphabet, let $x \notin \Sigma$, and let $T \in \mathbb{N}$ be a number of time steps. A *T-step capacity bound for* Σ is a mapping $c : \Sigma \times [1, T] \to \mathbb{N}_0$. Intuitively, $c(\sigma, t)$ corresponds to the capacity of symbol σ at time step t. A *job* (w, s) is a tuple consisting of a string $w \in (\Sigma \cup \{x\})^*$ and a starting time $s \in \mathbb{N}$ that satisfies $s + |w| - 1 \leq T$. Intuitively, the inequality guarantees that every job is completed by time step T. Given a set of jobs $\mathcal{J} := \{(w_i, s_i) \mid i \in [1, n]\}$, a symbol $\sigma \in \Sigma$ and a time step $t \in [1, T]$, we define the *load of symbol σ at time step t* as

$$\lambda_\sigma^t := |\{(w, s) \in J \mid w[t - s + 1] = \sigma\}|.$$

A set $\mathcal{J}$ of jobs *satisfies the capacity constraints* c if $\lambda_\sigma^t \leq c(\sigma, t)$ for every combination of $\sigma \in \Sigma$ and $t \in [1, T]$. For a job J and some $t \in [s, s + |J| - 1]$, we say that the $(t - s + 1)$th letter of J *is processed* at time t.

Given a set of jobs $\mathcal{J} := \{(w_i, s_i) \mid i \in [1, n]\}$, we let $\#x_i$ denote the number of occurrences of the symbol x in the string w_i for $i \in [1, n]$. The problem WTR is defined as follows.

Waiting Time Removal (WTR)
Input: A number of time steps T, a set $\mathcal{J} := \{(w_i, s_i) \mid i \in [1, n]\}$ of jobs, a T-step capacity bound c, and an integer k.
Question: Does there exist a family of sets $R_1, \ldots, R_n$ with $R_i \subseteq [1, \#x_i]$ for each $i \in [1, n]$ and $\sum_{i=1}^{n} |R_i| \leq k$ such that the job set $\mathcal{J}' := \{(w_i - R_i, s_i) \mid i \in [1, n]\}$ satisfies the capacity constraints c?

For notational convenience, we define for a job $J = (w, s)$ its length $|J| := |w|$, its starting time $s(J) := s$, and $J - R := (w - R, s)$ for $R \subseteq [1, \#x_i]$.

Note that $T \cdot n$ essentially bounds the size of an instance $(T, \mathcal{J}, c, t)$ of WTR: Each w_i has length at most T and thus, $|\mathcal{J}|$ can be encoded by encoding at most $T \cdot n$ symbols. Moreover, the load of a symbol can never exceed n and thus, c can be encoded using $\mathcal{O}(T^2 \cdot n \log(n))$ bits. Therefore, we can observe the following.

▶ **Observation 1.** *WTR admits a polynomial kernel for $T + n$.*

Furthermore, note that WTR can be solved by the following simple brut-force algorithm: Enumerate all size-k subsets of occurrences of the symbol x in the input instance, and check whether removing precisely these occurrences provides a modified set $\mathcal{J}'$ that satisfies the capacity constraints c. This simple brute-force strategy implies the following.

▶ **Observation 2.** *WTR can be solved in $|I|^{k+\mathcal{O}(1)}$ time, where $|I|$ denotes the input size.*

Graph Theory. We use standard notation from graph theory, see e.g. [12]. More precisely, all appearing graphs are undirected unless stated otherwise. A graph $G = (V, E)$ is *k-partite* if its vertex set can be partitioned into k independent sets, i.e., $V = V_1 \cup \ldots \cup V_k$ with $V_i \cap V_j = \emptyset$ for each $i \neq j$ and for each set V_i, there is no edge whose endpoints are both contained in V_i. For a graph $G = (V, E)$ and $V_1, V_2 \subseteq V$, we denote by $E[V_1, V_2]$ the edges with one endpoint in V_1 and the other endpoint in V_2. For a vertex $v \in V$, we denote by $\delta(v)$ the set of edges incident to v.

3 Parameterization by the Number of Jobs

We first study the parameterization by the number n of jobs. While the total number of jobs naturally appears to be a large parameter in real-world applications, we start by showing that – somewhat surprisingly – WTR is unlikely to be fixed-parameter tractable for the parameter n.

▶ **Theorem 3.** *For an instance of WTR, deciding the existence of a deletion family (of arbitrary size) is NP-hard and W[1]-hard parameterized by the number of jobs, even if $|\Sigma| = 2$.*

Proof. We reduce from Multicolored Independent Set which is NP-hard [11] and W[1]-hard parameterized by solution size ℓ [15]. In Multicolored Independent Set, one is given an ℓ-partite graph $G = (V_1 \cup \ldots \cup V_\ell, E)$ with $|V_1| = |V_2| = \ldots = |V_\ell|$, and the task is to find a multicolored independent set, i.e., a set of ℓ pairwise non-adjacent vertices containing exactly one vertex from each V_i. Let $G = (V_1 \cup \ldots \cup V_\ell, E)$ be an instance of Multicolored Independent Set. We fix an arbitrary ordering $e_1, \ldots, e_m$ of the edges of G as well as of the vertices of each color class $V_i = \{v_i^1, \ldots, v_i^{|V_1|}\}$. For each color class V_i and each edge $e_p \in E$, we define the $|V_1|$-letter substring w_i^p to be

$$w_i^p := \begin{cases} 0^{j-1}10^{|V_1|-j} & \text{if } e_p \in \delta(v_i^j) \text{ for some } j \in [1, |V_1|] \\ 0^{|V_1|} & \text{otherwise.} \end{cases}$$

Using these substrings, we create, for each $i \in [1, \ell]$, one job $J_i = (w_i, 1)$ where

$$w_i := x^{|V_1|-1} \circ \bigcirc_{p=1}^{m} w_i^p .$$

We define the $(m \cdot |V_1| + |V_1| - 1)$-capacity bound c: For every $p \in [1, m]$, letter 1 has capcity one at time $p \cdot |V_1|$. All other capacities are infinite. See Figure 2 for an example. The reduction clearly runs in polynomial time and the number of jobs equals ℓ.

We continue by showing correctness. First note that for any set $R_i \subseteq [1, |V_1| - 1]$, job $J_i - R_i$ contributes to the load of letter 1 at time $p \cdot |V_1|$ if and only if $v_i^{|R_i|+1}$ is adjacent to e_p. Given a multicolored independent set $I = \{v_1^{j_1}, \ldots, v_\ell^{j_\ell}\}$, we construct a deletion set $\mathcal{R} = \{R_1, \ldots, R_n\}$ as follows: For each job J_i, we let $R_i = [1, j_i - 1]$. To see that this $\{(w_i - R_i, 1) \mid i \in [1, n]\}$ satisfies the capacity constraints, note that only letter 1 has a finite capacity (at times $t = p \cdot |V_1|$ for $p \in [1, m]$), so it suffices to show that the load of letter 1 does not exceed its capacity. Assume towards a contradiction that there is some time t where the load of letter 1 exceeds its capacitiy, i.e., $t = p \cdot |V_1|$ for some $p \in [1, m]$ and there are two jobs $J_i - R_i$ and $J_{i'} - R_{i'}$ both contributing to the load of 1 at time t. This implies that e_p is incident to both $v_i^{j_i}$ and $v_{i'}^{j_{i'}}$, a contradiction to I being an independent set.

We conclude with the reverse direction, so let $\mathcal{R} = \{R_1, \ldots, R_n\}$ be a deletion family satisfying the capacity constraints. Let $j_i := |R_i| \in [1, |V_1| - 1]$. We claim that $I = \{v_1^{j_1}, \ldots, v_\ell^{j_\ell}\}$ is an independent set. Assume towards a contradiction that $e_p = \{v_i^{j_i}, v_{i'}^{j_{i'}}\}$ for some $p \in [m]$ and $i, i' \in [1, \ell]$. Then at time $p \cdot |V_1|$, both $J_i - R_i$ and $J_{i'} - R_{i'}$ contribute to the load of letter 1, a contradiction to the feasibility of the schedule. ◀

		$\{v_1^1, v_2^1\}$			$\{v_1^3, v_2^2\}$			$\{v_2^2, v_3^1\}$			$\{v_2^3, v_3^2\}$			$\{v_1^2, v_3^3\}$			$\{v_1^3, v_3^1\}$		
x	x	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1
x	x	1	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0
x	x	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	1	0	0

$I = \{v_1^1, v_2^2, v_3^3\}$

x	x	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1
x	1	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	
0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	1	0	0		

Figure 2 The upper part shows an example for the reduction for $\ell = 3$ and the graph containing edges $\{v_1^1, v_2^1\}$, $\{v_1^3, v_2^2\}$, $\{v_2^2, v_3^1\}$, $\{v_2^3, v_3^2\}$, $\{v_1^2, v_3^3\}$, and $\{v_1^3, v_3^1\}$. Red columns correspond to time steps where the capacity of letter 1 equals one while all other time steps have unboundend capacity. The vertical lines separate the different "blocks" of $|V_1|$ letters corresponding to an edge. The lower part depicts a solution corresponding to the independent set $\{v_1^1, v_2^2, v_3^3\}$.

While WTR is presumably not fixed-parameter tractable for n, we show that – on the positive side – WTR can be solved in polynomial time for constant values of n by providing the following XP algorithm.

▶ **Theorem 4.** *WTR can be solved in $O(n \cdot T^{2n+1})$ time.*

Proof. We give a dynamic program solving the problem. The dynamic programming table τ has entries of the form $\tau[(t, p_1, \ldots, p_n)]$ for every suitable tuple $(t, p_1, \ldots, p_n)$, where we call $(t, p_1, \ldots, p_n)$ with $t \in [0, T+1]$ and $p_i \in [0, |J_i|+1]$ for each $i \in [1, n]$ *suitable* if, for every $i \in [n]$, it holds that $p_i = 0$ if and only if $t < s(J_i)$. Each entry stores the minimum size of a deletion set such that the load of any symbol for any time $t' \leq t$ does not exceed its capacity and at time t (where each letter has capacity 0 for $t' \in \{0, T+1\}$), the p_ith letter of J_i is processed. Herein, $p_i = 0$ encodes that no letter of J_i is processed until time t and $p_i = |J_i| + 1$ encodes that all letters from J_i have already been processed until time t. The table is computed as follows: We initialize the table by setting

$$\tau[(0, p_1, \ldots, p_n)] = \begin{cases} 0 & \text{if } p_1 = \ldots = p_n = 0 \\ \infty & \text{otherwise.} \end{cases}$$

For the update step, we introduce some additional notation. We say that a suitable tuple $(t-1, p_1', \ldots, p_n')$ *is compatible with* a suitable tuple $(t, p_1, \ldots, p_n)$ if, for every $i \in [n]$,

- $p_i' \leq p_i$,
- if $p_i \notin \{0, |J_i|+1\}$, then $p_i' < p_i$ and $w_i[p_i'+1] = w_i[p_i'+2] = \ldots = w_i[p_i-1] = x$, and
- for each $\sigma \in \Sigma$, we have $|\{i \mid s_i[p_i] = \sigma\}| \leq \lambda_\sigma^t$.

Intuitively, this means that if the p_i'th letter of job J_i is processed at time $t-1$, then the p_ith letter can be processed at time t (after deleting all x in between; the second item ensures that only x are in between the p_i'th and p_ith letter). We update the table as follows:

$$\tau[(t, p_1, \ldots, p_n)] = \min_{(t-1, p_1', \ldots, p_n') \text{ compatible with } (t, p_1, \ldots, p_n)} \tau[(t-1, p_1', \ldots, p_n')] + \sum_{i : p_i \neq p_i'} (p_i - p_i' - 1)\,.$$

An optimal solution can be read from $\tau[(T+1, |J_1|+1, \ldots, |J_n|+1)]$ using traceback. More formally, we have a yes-instance of WTR if and only if $\tau[(T+1, |J_1|+1, \ldots, |J_n|+1)] \leq k$.

The dynamic programming table has $O(T \cdot \Pi_{i=1}^{n}(|J_i|+2)) = O(T^{n+1})$ entries, each of which can be computed in $O(n \cdot T^n)$ time. Thus, the dynamic program runs in $O(n \cdot T^{2n+1})$ time.

It remains to show correctness. First, we show by induction on t that if $\tau[(t, p_1, \ldots, p_n)] = q < \infty$, then there is a deletion family of size q such that no load exceeds its capacity at any time $t' \leq t$ and at time t, the p_ith letter of J_i is scheduled. This clearly holds for $t = 0$. For $t \geq 1$, let $a := (t-1, p'_1, \ldots, p'_n)$ such that $\tau[(t, p_1, \ldots, p_n)] = \tau[a] + \sum_{i:p_i \neq p'_i}(p_i - p'_i - 1)$. This entry $\tau[a]$ corresponds to a deletion family $\mathcal{R}'$. By induction, for jobs $\{J_i - R'_i\}$, no load is exceeded for any time $t' < t$. By the second condition, the $p'_i + 1, \ldots, p_i - 1$th letters of J_i are all x, say the r_ith to r'_ith occurrences of x. Setting $\mathcal{R} = \{R'_i \cup [r_i, r'_i] \mid i \in [n]\}$ for each $i \in [n]$ results in a deletion family of size $\tau[a] + \sum_{i:p_i \neq p'_i}(p_i - p'_i - 1) = q$. By the third condition, no load is exceeded at time t. By induction, no load is exceeded at any time $t' \leq t-1$ for $\mathcal{R}'$; consequently, also for $\mathcal{R}$ no load is exceeded at any time $t' \leq t-1$ for $\mathcal{R}$. Thus, $\mathcal{R}$ is a deletion family of size q with the desired properties.

Next, we show that if there is a deletion family $\mathcal{R}$ of size q such that at time t, for each $i \in [1, n]$ the p_ith letter of J_i is scheduled, and no load is exceeded for any $t' \leq t$, then $\tau[(t, p_1, \ldots, p_n)] \leq q$. We again do so by induction on t. For $t = 0$, the statement is obvious, so consider $t > 0$. The deletion family $\mathcal{R}$ is also a deletion family where at time $t-1$, the p'_ith letter of J_i is scheduled for some $p'_1, \ldots, p'_n$. By the second condition, the $p'_i + 1, \ldots, p_i - 1$th letters of J_i are all x, say the r_ith to r'_ith occurrences of x. Setting $R'_i := R_i \setminus [r_i, r'_i]$ for $i \in [1, n]$ therefore results in a deletion set $\mathcal{R}' = \{R'_1, \ldots, R'_n\}$ of size $q - \sum_{i:p_i \neq p'_i}(p_i - p'_i - 1)$. By induction, we have $\sum_{i=1}^{n} |R'_i| \geq \tau[(t-1, p'_1, \ldots, p'_n)]$. Thus, we have

$$q = \sum_{i=1}^{n} |R'_i| + \sum_{i:p_i \neq p'_i} (p_i - p'_i - 1) \geq \tau[(t-1, p'_1, \ldots, p'_n)] + \sum_{i:p_i \neq p'_i} (p_i - p'_i - 1)$$
$$\geq \tau[(t, p_1, \ldots, p_n)] .$$

The correctness follows. ◀

Adding a simple pruning rule to the dynamic program from Theorem 4 yields fixed-parameter tractability by the combined parameter $n + k$:

▶ **Corollary 5.** *WTR parameterized by $n + k$ can be solved in $O(T \cdot n^{2k+1})$ time.*

Proof. For every job $J_i = (w_i, s_i)$, the deletion set R_i has size at most k. Thus, at each time t, at most k different letters from w_i can be scheduled, implying that at most k different values for p_i need to be considered. Consequently, the number of DP entries to consider is $O(T \cdot n^k)$, and each of these entries can be computed in $O(n \cdot n^k)$ time. Therefore, WTR can be solved in $O(T \cdot n^{2k+1})$ time. ◀

4 Parameterization by the Number of Time Steps

We next study parameterization by the number T of time steps. We start by showing that WTR cannot be solved in polynomial time for constant values of T unless P = NP. Motivated by this strong hardness result, we then study parameterizations by the combined parameters of $T + |\Sigma|$ and $T + k$.

▶ **Theorem 6.** *WTR is NP-hard even if T is constant.*

Proof. We provide a reduction from Independent Set. In Independent Set, one is given a graph $G = (V, E)$ together with an integer ℓ. The question is whether there exists a set $I \subseteq V$ of pairwise non-adjacend vertices with $|I| \geq \ell$. Independent Set is well-known to be NP-hard even if G has maximum degree 3 [7].

Let $(G = (V, E), \ell)$ be an instance of Independent Set, where G has maximum degree 3. We describe how to construnct an equivalent instance of WTR in polynomial time. To this end, we assume that G has a proper edge coloring with four colors: Due to Vizing's Theorem [23], we can partition the edge set E into four sets E_1, E_2, E_3, and E_4 such that for each $i \in [1, 4]$, no two edges in E_i share one endpoint. Since the proof of Vizing's Theorem provides an algorithm computing the partition in polynomial time, we may use the sets $E_1, \ldots, E_4$ for our construction.

To construct an instance $(T, \mathcal{J}, c, k)$ of WTR, we first set $k := \ell$ and we set $T := 10$. Furthermore, we define the alphabet $\Sigma := \{0\} \cup E$. Next, given some $v \in V$ and some $i \in [1, 4]$, we let $w_v^i := 00$ if v has no incident edge in E_i, and we let $w_v^i := 0e$ otherwise, where e is the unique incident edge in E_i. We then define $w_v := x \circ w_v^1 \circ w_v^2 \circ w_v^3 \circ w_v^4 \circ 0$ and set $\mathcal{J} := \{(w_v, 1) \mid v \in V\}$. Note that for every $v \in V$, we have $|w_v| = 10$.

We complete the construction by defining the capacity c. We set $c(0, 10) := |V| - \ell$. Furthermore, for every $i \in \{2, 4, 6, 8\}$ and $e \in E$ we define $c(e, i) := 1$. All remaining capacities are set to ∞.

Before we prove the correctness, we provide some intuition. Since $c(0, 10) = |V| - \ell$ and every w_v ends with the symbol 0 at time step 10, we need to remove the unique occurrence of x in ℓ strings, which then corresponds to the choice of vertices in a solution of Independent Set. The capacities $c(e, i) := 1$ for $i \in \{2, 4, 6, 8\}$ then guarantee that no pair of chosen vertices are incident with the same edge.

We conclude the proof by showing that (G, ℓ) is a yes-instance of Independent Set if and only if $(T, \mathcal{J}, c, k)$ is a yes-instance of WTR.

Let (G, ℓ) be a yes-instance of Independent Set. Then, there is a set $I \subseteq V$ containing ℓ vertices that are pairwise non-adjacent. We construct a solution of the WTR instance as follows: for each $v \in I$, we remove the unique occurrence of x in w_v and let $\tilde{w}_v$ denote the resulting string. Note that $\tilde{w}_v[t] = w_v[t + 1]$ for $t \in [1, 9]$. Note that we removed $|I| = \ell = k$ occurrences of x. We show that the modified job set satisfies the capacity constraints c: First, observe that $\lambda_0^{10} = |V| - |I| = V - \ell = c(0, 10)$. Second, let $i \in \{2, 4, 6, 8\}$ and assume towards a contradiction that $c(e, i) \geq 2$ for some symbol $e \in \Sigma$. Then, there exist $\tilde{w}_v$ and $\tilde{w}_u$ with $\tilde{w}_v[i] = \tilde{w}_u[i] = e$. Then, we have $w_v[i+1] = w_u[i+1] = e$ and thus, by the construction of $\mathcal{J}$, the vertices v and u are both incident with the same edge e in G. This contradicts the fact that I consists of pairwise non-adjacent vertices. Consequently, we have $c(e, i) \leq 1$ for every $i \in \{2, 4, 6, 8\}$ and $e \in E$. Since all other capacities are ∞, we conclude that the modified job set satisfies the capacity constraints c and thus, $(T, \mathcal{J}, c, k)$ is a yes-instance of WTR.

Conversely, let $(T, \mathcal{J}, c, k)$ be a yes-instance of WTR. Then, there is a subset of jobs $\mathcal{J}' \subseteq \mathcal{J}$ with $|\mathcal{J}'| \leq k = \ell$ such that removing the unique occurrence of x in the corresponding strings results in a plan that satisfies the capacity constraints c. Observe that $|\mathcal{J}'| = \ell$ since otherwise $\lambda_0^{10} = |V| - |\mathcal{J}'| > |V| - \ell = c(0, 10)$. Let $w_{v_1}, \ldots, w_{v_\ell}$ be the strings of the jobs in $\mathcal{J}'$. Then, $I := \{v_1, \ldots, v_\ell\}$ is a set of vertices of G with $|I| = \ell$. It reamins to show that the vertices in I are pairwise non-adjacent. Assume towards a contradiction that two vertices v_1 and v_2 of I are connected by an edge e. Recall that the edge set E is partitioned into the sets $E_1, \ldots, E_4$. Without loss of generality we assume $e \in E_1$. Then, by the construction of the jobs we have $w_{v_1}[3] = w_{v_2}[3] = e$. Removing the single occurrence of x in w_{v_1} and w_{v_2}

provides strings $\tilde{w}_{v_1}$ and $\tilde{w}_{v_2}$ with $\tilde{w}_{v_1}[2] = \tilde{w}_{v_2}[2] = e$. Since $c(e, 2) = 1$, this contradicts the fact that the modified job set satisfies the capacity constraints c. Consequently, all vertices in I are pairwise non-adjacent and thus, (G, ℓ) is a yes-instance of Independent Set. ◀

Note that the alphabet size of the instance constructed in the proof of Theorem 6 is roughly the size of the input graph of the Independent Set instance. Recall that in our application the symbols in Σ correspond to distinct machine types on which the jobs need to be processed. In a real-world production setting it is reasonable that there are only few distinct machines types. Motivated by this observation, we study the combined parameter $T + |\Sigma|$ and show that WTR is FPT for this parameterization.

▶ **Theorem 7.** *WTR parameterized by $T + |\Sigma|$ is FPT.*

Proof. Let $(T, \mathcal{J}, c, k)$ be an instance of WTR. We show fixed-parameter tractability by reformulating WTR as an integer linear program (ILP) where the number of variables is bounded by a function only depending on T and $|\Sigma|$. It is well-known that ILP parameterized by the number of variables is in FPT [9]. To this end, we introduce the notion of job types. Recall that a job $(w, s) \in \mathcal{J}$ consists of a string $w \in (\Sigma \cup \{x\})^*$ and a starting time $s \in [1, T]$. We say that two jobs *have the same type* $\tau = (w_\tau, s_\tau)$ if they have the same string w_τ and the same starting time s_τ. We denote the number of jobs of type τ by n_τ. For each job type τ, we consider possible x-removal sets $R \subseteq [1, \#w_\tau]$. We provide the ILP formulation by specifying the variables, the constraints, and the target function.

Variables. For each combination of a job type τ and a possible removal set $R \subseteq [1, \#w_\tau]$, we introduce an integer variable $Y_{(\tau,R)}$. The intuitive idea is that the value of $Y_{(\tau,R)}$ equals the number of jobs with type τ whose internal x-deletions correspond to the (possibly empty) removal set R. More precisely, a solution set $\mathcal{J}'$ contains exactly $Y_{(\tau,R)}$ jobs of the form $(w_\tau - R, s_\tau)$.

Constraints. For each type τ, we add the following constraint, ensuring that all jobs of type τ are scheduled:

$$\sum_{R \subseteq [1, \#w_\tau]} Y_{(\tau,R)} = n_\tau \,.$$

For each combination of $\sigma \in \Sigma$ and $t \in [1, T]$, we introduce the following constraint:

$$\sum_{\text{Type } \tau = (w_\tau, s_\tau)} \sum_{\substack{R \subseteq [1, \#w_\tau] \\ (w_\tau - R)[t - s_\tau + 1] = \sigma}} Y_{(\tau,R)} \leq c(\sigma, t) \,.$$

The left-hand side corresponds to the load of symbol σ at time step t. Thus, these constraints guarantee that the solution satisfies the capacity constraints c: Intuitively, each string w_τ has been modified by a (possibly empty) x-removal set R. For a given type τ, the inner sum counts all modified strings of type τ that have symbol σ at time step t. Since we sum up these values over all possible types, the left-hand side corresponds to the load of symbol σ at time step t.

Target Function. The target function has the form

$$\sum_{\text{Type } \tau} \sum_{R \subseteq [1, \#w_\tau]} |R| \cdot Y_{(\tau,R)}.$$

Recall that intuitively each value of $Y_{(\tau,R)}$ equals the number of jobs with type τ whose internal x-deletions correspond to the (possibly empty) removal set R. Thus, for each combination of a type τ and a removal set R, we delete exactly $Y_{(\tau,R)} \cdot |R|$ symbols. Since we sum up over all τ and R, the target function models the total number of x-deletions.

Number of Variables. Note that the number of strings in $(\Sigma \cup \{x\})^*$ in an instance with maximum time step T is bounded by $(|\Sigma|+1)^T$, and that we have at most T starting positions. Consequently, there are at most $T \cdot (|\Sigma|+1)^T$ possible job types $\tau = (w_\tau, s_\tau)$. Furthermore, since $|w_\tau| \leq T$ for all τ, there are at most 2^T possible x-removal sets $R \subseteq [1, \#w_\tau]$. Thus, the number of variables $Y_{(\tau,R)}$ is bounded by $T \cdot (|\Sigma|+1)^T \cdot 2^T$. By a classical result by Lenstra [9], an ILP can be solved FPT-time with respect to the number of variables. This implies that the above ILP can be solved in FPT-time with respect to $T + |\Sigma|$. ◀

We complement the FPT result for $T + |\Sigma|$ by showing that WTR is unlikely to admit a problem kernel of polynomial size for this parameterization.

▶ **Theorem 8.** *WTR parameterized by $T+|\Sigma|$ does not admit a polynomial kernel unless* NP $\subseteq$ coNP/poly.

Proof. We prove the statement by providing a polynomial parameter transformation from Set Cover. In Set Cover, one is given a universe of elements U, a familiy $\mathcal{F} \subseteq 2^U$ of subsets of U, and an integer ℓ. The question is whether there exists a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ such that $|\mathcal{F}'| \leq \ell$ and $\bigcup_{F \in \mathcal{F}'} F = U$. Set Cover parameterized by $|U|$ does not admit a polynomial kernel unless NP $\subseteq$ coNP/poly [6].

Let $(U, \mathcal{F}, \ell)$ be an instance of Set Cover and fix an arbitrary ordering $U := \{u_1, \ldots, u_{|U|}\}$ of the elements in the universe. We describe how to construct an equivalent instance $(T, \mathcal{J}, c, k)$ of WTR for the alphabet $\Sigma = \{0, 1\}$. We set $k := \ell$ and we set $T := 2 \cdot |U| + 1$. Note that T is polynomial in $|U|$. Next, we define the job set $\mathcal{J}$. Given some $F \in \mathcal{F}$ and some $u \in U$, we let $w_u^F := 01$ if $u \in F$ and $w_u^F := 00$ if $u \notin F$. We define the job set $\mathcal{J} := \{(w^F, 1) \mid F \in \mathcal{F}\}$, where

$$w_F := x \circ \bigcirc_{i=1}^{|U|} w_{u_i}^F.$$

Note that $w_F[2i+1] = 1$ if and only if $u_i \in F$ and we have $w_F[2i] = 0$ for every $i \in [1, |U|]$.

We finish the construction by defining the $(2 \cdot |U| + 1)$-capacity bound c: For every $i \in [1, |U|]$, we set $c(1, 2i+1) := |\{F \in \mathcal{F} \mid u \in F\}| - 1$. All remaining $c(\sigma, t)$ are set to ∞.

We next show that $(U, \mathcal{F}, \ell)$ is a yes-instance of Set Cover if and only if $(T, \mathcal{J}, c, \ell)$ is a yes instance of WTR.

Let $(U, \mathcal{F}, \ell)$ be a yes-instance of Set Cover. Then, there exists a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ with $|\mathcal{F}'| \leq \ell$ such that for each $i \in [1, |U|]$, there is one $F \in \mathcal{F}'$ with $u_i \in F$. We construct a solution of the WTR instance as follows: for each $F \in \mathcal{F}'$, we remove the unique occurrence of x in w^F, and let $\tilde{w}^F$ denote the resulting string. Note that $\tilde{w}^F[t] = w^F[t+1]$ for every $t \in [1, T-1]$. We show that the modified job set satisfies the capacity constraints c. Since all other capacities are ∞, it suffices to show that for each $i \in [1, |U|]$ the load of symbol 1 at time step $2i+1$ is not larger than $c(1, 2i+1)$. Since $\tilde{w}^F[2i+1] = w^F[2i+2] = 0$ for all $F \in \mathcal{F}'$, we have $\lambda_1^{2i+1} = |\{F \in \mathcal{F} \setminus \mathcal{F}' \mid u_i \in F\}|$. Since each u_i is contained in at least one $F \in \mathcal{F}'$, we conclude that $\lambda_1^{2i+1} \leq |\{F \in \mathcal{F} \mid u_i \in F\}| - 1 = c(1, 2i+1)$. Therefore, $(T, \mathcal{J}, c, \ell)$ is a yes instance of WTR.

Conversely, let $(T, \mathcal{J}, c, k)$ be a yes-instance of WTR. Then, there is a subset $\mathcal{J}' \subseteq \mathcal{J}$ of jobs with $|\mathcal{J}'| \leq \ell$ such that removing the unique occurrence of x in the corresponding strings results in a plan that satisfies the capacity constraints c. Let $w^{F_1}, \ldots, w^{F_{|\mathcal{J}'|}}$ be the

strings of the jobs in $\mathcal{J}'$. Then, $\mathcal{F}' := \{F_1, \ldots, F_{|\mathcal{J}'|}\}$ is a subfamily of $\mathcal{F}$ of size at most ℓ. Assume towards a contradiction that there is some $u_i \in U$ with $u_i \notin \bigcup_{F \in \mathcal{F}'} F$. Then, for each $F \in \mathcal{F}$ with $u_i \in F$, the occurrence of x in w^F has not been removed. Consequently, the load of symbol 1 at time step $2i+1$ is $|\{F \in \mathcal{F} \mid u_i \in F\}| > c(1, 2i+1)$. This contradicts the fact that removing the x in the strings of $\mathcal{J}'$ results in a job set that satisfies the capacity constraints c. Therefore, $(U, \mathcal{F}, \ell)$ is a yes-instance of Set Cover. ◀

The following two observations can be made by a taking closer look at the instance constructed in the proof of Theorem 8: First, the parameter k equals the solution size of the Set Cover instance. Since Set Cover is known to be W[2]-hard when parameterized by solution size [5], this implies the following.

▶ **Corollary 9.** *WTR parameterized by k is W[2]-hard even if $|\Sigma| = 2$.*

Second, the parameter $n + k$ is upper-bounded by $2 \cdot |\mathcal{F}|$, where $\mathcal{F}$ is the set familiy in the Set Cover instance. Since Set Cover does not admit a polynomial kernel for $|\mathcal{F}|$ [6], this implies the following.

▶ **Corollary 10.** *WTR does not admit a polynomial kernel for $n+k$ unless* NP $\subseteq$ coNP/poly *even if $|\Sigma| = 2$.*

Next, consider the parameterization by $T + k$. Recall that WTR is XP for k by Observation 2 and thus, WTR is XP for $T + k$. We complement this result by showing W[1]-hardness.

▶ **Theorem 11.** *WTR parameterized by $T + k$ is W[1]-hard.*

Proof. We reduce from Multicolored Clique parameterized by solution size ℓ. In Multicolored Clique, one is given an ℓ-partite graph $G = (V_1 \cup \ldots \cup V_\ell, E)$ with $|V_1| = |V_2| = \ldots = |V_\ell|$, and the task is to find a multicolored clique, i.e., a set of ℓ pairwise adjacent vertices containing exactly one vertex from each V_i. Multicolored Clique is W[1]-hard parameterized by ℓ, even if there is some $r \in \mathbb{N}$ so that each vertex has exactly r neighbors in every other color class [5]. Let $G = (V_1 \cup \ldots \cup V_\ell, E)$ be an instance of Multicolored Clique where each vertex v has exactly r neighbors in every other color class. We let $f : [1, \binom{\ell}{2}] \to \binom{[1,\ell]}{2}$ be an arbitrary bijection to fix an ordering of the set $\binom{[1,\ell]}{2} := \{X \subseteq [1, \ell] \mid |X| = 2\}$ containing all two-element subsets of $[1, \ell]$. We use the alphabet $\Sigma = V \cup \{0, 1\} \cup \{z_i \mid i \in [\ell]\}$. We add the following jobs:

- for each $v \in V_i$, we add one job $J_v = (w_v, 1)$ where w_v starts with xz_i. Afterwards, for each $q \in [1, \binom{\ell}{2}]$, we append the following four letters:
  - if $i \notin f(q)$, then we append 0000,
  - if $f(q) = \{i, i'\}$ for some $i' > i$, then we append $0v00$, and
  - if $f(q) = \{i', i\}$ for some $i' < i$, then we append $000v$.
- for each edge $e = \{u, v\} \in E$ with $u \in V_i$ and $v \in V_{i'}$ for some $i < i'$, we add one job $J_e = (xu0v, 4 \cdot (f(\{i, i'\}) - 1) + 2)$.

The capacities are as follows:

- for each $i \in [1, \ell]$, letter z_i has capacity 1 at time 1 and capacity $|V_1| - 1$ at time 2,
- for each $v \in V$, letter v has capacity r at every time step, and
- all other capacities are set to be ∞.

Finally, we set $k := \ell + \binom{\ell}{2}$. Note that $T = 4 \cdot \binom{\ell}{2} + 2$. As the reduction can clearly be computed in polynomial time, it remains to show correctness.

Before we show the correctness, we provide some intuition. The capacities of z_i at time 1 ensures that at most one deletion occurs in jobs $\{J_v \mid v \in V_i\}$, while the capacities of z_i at time 2 ensure that at least one deletion occurs in $\{J_v \mid v \in V_i\}$. Therefore, exactly one deletion occurs in $\{J_v \mid v \in V_i\}$ for every $i \in [1, \ell]$. These ℓ vertices with a deletion correspond to a multicolored clique: Since there exists some unique $v_i \in V_i$ such that the x in J_{v_i} is deleted, this implies that all occurrences of v_i in J_{v_i} move one time step earlier. Thereby, for each $q \in [1, \binom{\ell}{2}]$ with $i \in f(q)$, letter v_i from J_{v_i} "collides" with letter v_i in J_e the r edges e incident to v_i in $E[V_i, V_{i'}]$ where $i' = f(q) \setminus \{i\}$. Because the capacity of v_i is r at each time step, this implies that a deletion has to occur in J_e for some edge $e \in E[V_i, V_{i'}]$ incident to v_i. As the budget allows for $k - \ell = \binom{\ell}{2}$further deletions, this implies that there is exactly one edge in $E[V_i, V_{i'}]$ with a deletion for $\{i, i'\} \in \binom{[1,\ell]}{2}$. Since this edge needs to be incident to both v_i and $v_{i'}$, it follows that $\{v_i \mid i \in [1, \ell]\}$ is a multicolored clique.

We first show that if G is a yes-instance to Multicolored Clique, then $(T, \mathcal{J}, c, k)$ is yes-instance of WTR. Given a clique $\{v_1, \ldots, v_\ell\}$ with $v_i \in V_i$, we construct a solution to the WTR instance as follows: We remove the unique occurrence of x in job J_{v_i} for each $i \in [1, \ell]$ and in job $J_{\{v_{i_1}, v_{i_2}\}}$ for $i_1 < i_2 \in [1, \ell]$. We call the resulting strings $\tilde{w}_v$ for $v \in V$ and $\tilde{w}_e$ for $e \in E$. As we removed k occcurences of x in total, it remains to show that the capacity constraints are satisfied. For letter z_i, we have a load of 1 at time 1 (from job J_{v_i}) and a load of $|V_1| - 1$ at time to (from jobs J_v for $V \in V_i \setminus \{v_i\}$). Thus, the capacities of z_i are satisfied. For $v \in V \setminus \{v_1, \ldots, v_\ell\}$, the maximum load at any time is $\deg(v) = r$, so its capacity is satisfied. We continue with letter v_i for $i \in [1, \ell]$. At time $4 \cdot (q-1) + 3$, the load of v_i is either 0 (if $i \notin f(q)$ or $f(q) = \{i, i'\}$ for some $i' < i$) or r (if $f(q) = \{i, i'\}$ for some $i' > i$; in this case, job J_{v_i} and $r - 1$ edge jobs (all edges from $E[V_i, V_{i'}] \cap \delta(v_i)$ except for $\{v_i, v_{i'}\}$). In both cases, the capacity constraint is satisfied. Symetrically, at time $4 \cdot (q-1) + 5$, the load is either 0 (if $i \notin f(q)$ or $f(q) = \{i, i'\}$ for some $i' > i$) or r (if $f(q) = \{i, i'\}$ for some $i' < i$; in this case, job J_{v_i} and $r - 1$ edge jobs (all edges from $E[V_i, V_{i'}] \cap \delta(v_i)$ except for $\{v_i, v_{i'}\}$). In both cases, the capacity constraint is satisfied.

Lastly, we show that if $(T, \mathcal{J}, c, k)$ is a yes-instance to WTR, then G is a yes-instance to Multicolored Clique. Since $(T, \mathcal{J}, c, k)$ is a yes-instance, there is a subset $\mathcal{J}' \subseteq \mathcal{J}$ of jobs with $|\mathcal{J}'| \leq \ell$ such that removing the unique occurrence of x in the corresponding strings results in a plan that satisfies the capacity constraints c. As the capacity of z_i for $i \in [\ell]$ at time step 2 is $|V_1| - 1$, it follows that for each $i \in [1, \ell]$, there exists some $v_i \in V_i$ with $J_{v_i} \in \mathcal{J}'$. As $c(z_i, 1) = 1$, this v_i is unique.

We next show that $\{v_1, \ldots, v_\ell\}$ forms a multicolored clique. Consider any $q \in [1, \binom{\ell}{2}]$ and let $f(q) = \{i_1, i_2\}$ for some $i_1 < i_2$. At time $4 \cdot (q-1) + 3$, job $J_{v_{i_1}}$ contributes one to the load of v_{i_1}. For each edge $e \in E[V_{i_1}, V_{i_2}] \cap \delta(v_{i_1})$, job J_e contributes 1 to the load of v_{i_1} at time $4 \cdot (q-1) + 3$ if and only if $J_e \notin \mathcal{J}'$. As $|E[V_{i_1}, V_{i_2}] \cap \delta(v_{i_1})| = r = c(v_i, 4 \cdot (q-1) + 3)$, this implies that there is at least one edge $e \in E[V_{i_1}, V_{i_2}] \cap \delta(v_{i_1})$ with $J_e \in \mathcal{J}'$. Considering the letter v_{i_2} and time $4 \cdot (q-1) + 5$, symmetric arguments show that there is at least one edge $e \in E[V_{i_1}, V_{i_2}] \cap \delta(v_{i_2})$ with $J_e \in \mathcal{J}'$. As there are ℓ deletions in jobs J_v for $v \in V$ and $k - \ell = \binom{\ell}{2}$, this implies that for each $i_1 < i_2 \in [\ell]$, there is a unique $e \in E[V_i, V_{i'}] \cap \delta(v_{i_1}) \cap \delta(v_{i_2})$ with $J_e \in \mathcal{J}'$, and this edge is incident to both v_{i_1} and v_{i_2}, i.e., $e = \{v_{i_1}, v_{i_2}\}$. This implies that $\{v_1, \ldots, v_\ell\}$ is a multicolored clique. ◀

5 Conclusion

In this work, we initiated theoretical studies of repairing production schedules by the deletion of waiting times in a job shop setting by introducing a mathematical formulation of the problem. While our results mostly highlight the hardness of this problem, there are also few tractability results, for example when there are only few different machine types and a short time horizon is considered. We conclude with several proposals for further research questions:

- Instead of removing waiting times, there are also other natural operations to modify a schedule, for example swapping jobs. What is the complexity of the resulting problem?
- Our scheduling model is quite general, being based on job shop scheduling. Does basing the problem on simpler scheduling models, e.g. flow shops, result in a more tractable problem?
- Given that eliminating all capacity violations turned out to be quite hard, one might try to not directly search for a "perfect" schedule, but instead only a "better" one (i.e., one with fewer capacity violations). Can such a schedule be found using the framework of parameterized local search?
- While we study how to remove pre-planned waiting times, it might be interesting to study how these waiting times can be established in the first place. How can we create schedules that are robust in the sense that they can be repaired by removing waiting times?

References

1 Alessandro Agnetis, Jean-Charles Billaut, Michael Pinedo, and Dvir Shabtay. Fifty years of research in scheduling – Theory and applications. *European Journal of Operational Research*, 2025. doi:10.1016/j.ejor.2025.01.034.

2 Susanne Albers and Günter Schmidt. Scheduling with unexpected machine breakdowns. *Discret. Appl. Math.*, 110(2-3):85–99, 2001. doi:10.1016/S0166-218X(00)00266-3.

3 Kaja Balzereit, Niels Grüttemeier, Nils Morawietz, Dennis Reinhardt, Stefan Windmann, and Petra Wolf. Scalable neighborhood local search for single-machine scheduling with family setup times. *CoRR*, abs/2409.00771, 2024. doi:10.48550/arXiv.2409.00771.

4 Nicolas Boria and Federico Della Croce. Reoptimization in machine scheduling. *Theor. Comput. Sci.*, 540:13–26, 2014. doi:10.1016/J.TCS.2014.04.004.

5 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

6 Michael Dom, Daniel Lokshtanov, and Saket Saurabh. Kernelization lower bounds through colors and ids. *ACM Trans. Algorithms*, 11(2):13:1–13:20, 2014. doi:10.1145/2650261.

7 M. R. Garey, David S. Johnson, and Larry J. Stockmeyer. Some simplified NP-complete graph problems. *Theor. Comput. Sci.*, 1(3):237–267, 1976. doi:10.1016/0304-3975(76)90059-1.

8 Kevin D Glazebrook. Scheduling tasks with exponential service times on parallel processors. *J. Appl. Probab.*, 16(3):685–689, 1979.

9 Hendrik W. Lenstra Jr. Integer programming with a fixed number of variables. *Math. Oper. Res.*, 8(4):538–548, 1983. doi:10.1287/MOOR.8.4.538.

10 Bala Kalyanasundaram and Kirk Pruhs. Fault-tolerant scheduling. *SIAM J. Comput.*, 34(3):697–719, 2005. doi:10.1137/S0097539794261799.

11 Richard M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Problems*, pages 85–103, 1972. doi:10.1007/978-1-4684-2001-2_9.

12 Bernhard Korte and Jens Vygen. *Combinatorial Optimization*. Springer, 6th edition, 2018.

13 Patricio Lamas, Marcos Goycoolea, Bernardo K. Pagnoncelli, and Alexandra M. Newman. A target-time-windows technique for project scheduling under uncertainty. *Eur. J. Oper. Res.*, 314(2):792–806, 2024. doi:10.1016/J.EJOR.2023.10.027.

14 Arthur Müller and Lukas Vollenkemper. Reinforcement learning as an improvement heuristic for real-world production scheduling. In *Proceedings of the 23rd IEEE International Conference on Machine Learning and Applications (ICMLA '21)*. IEEE, 2024. doi:10.1109/ICMLA52953.2021.00085.

15 Krzysztof Pietrzak. On the parameterized complexity of the fixed alphabet shortest common supersequence and longest common subsequence problems. *J. Comput. Syst. Sci.*, 67(4):757–771, 2003. doi:10.1016/S0022-0000(03)00078-3.

16 George A. Rovithakis, Stelios E. Perrakis, and Manolis A. Christodoulou. Application of a neural-network scheduler on a real manufacturing system. *IEEE Trans. Control. Syst. Technol.*, 9(2):261–270, 2001. doi:10.1109/87.911378.

17 Markus W. Schäffter. Scheduling with forbidden sets. *Discret. Appl. Math.*, 72(1-2):155–166, 1997. doi:10.1016/S0166-218X(96)00042-X.

18 Dvir Shabtay and Miri Gilenson. A state-of-the-art survey on multi-scenario scheduling. *Eur. J. Oper. Res.*, 310(1):3–23, 2023. doi:10.1016/J.EJOR.2022.11.014.

19 Martin Skutella, Maxim Sviridenko, and Marc Uetz. Unrelated machine scheduling with stochastic processing times. *Math. Oper. Res.*, 41(3):851–864, 2016. doi:10.1287/MOOR.2015.0757.

20 Stephen F Smith. Reactive scheduling systems. *Intelligent scheduling systems*, pages 155–192, 1995.

21 V Subramaniam, AS Raheja, and K Rama Bhupal Reddy. Reactive repair tool for job shop schedules. *Int. J. Prod. Res.*, 43(1):1–23, 2005.

22 E. Szelke and G. Markus. A blackboard based perspective of reactive scheduling. In *Artificial Intelligence in Reactive Scheduling*, pages 60–77. Springer US, 1995. doi:10.1007/978-0-387-34928-2_6.

23 Vadim G Vizing. On an estimate of the chromatic class of a p-graph. *Diskret. analiz.*, 3:25–30, 1964.