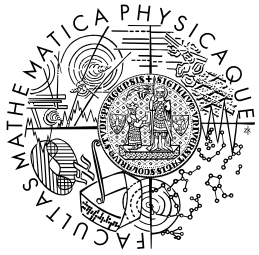

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

**MASTER THESIS**

Iva Hamerníková

# Fixed interval scheduling problems with endogenous uncertainty

Department of Probability and Mathematical Statistics

Supervisor of the master thesis: doc. RNDr. Martin Branda Ph.D.

Study programme: Mathematics

Study branch: Probability, Mathematical Statistics and Econometrics

Prague 2020

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date .....  .....

Author's signature

I would like to thank to my supervisor, doc. RNDr. Martin Branda Ph.D., for all his valuable advice and for his efforts and communication despite the complicated situation. I would also like to thank to my friends and family for their support, patience and willingness to discuss optimization problems even at midnight.

Title: Fixed interval scheduling problems with endogenous uncertainty

Author: Iva Hamerníková

Department: Department of Probability and Mathematical Statistics

Supervisor: doc. RNDr. Martin Branda Ph.D., Department of Probability and Mathematical Statistics

Abstract: This thesis is focused on the fixed interval scheduling (FIS) problems with random delays. Firstly, we introduce the concept of FIS problems and the exogenous and endogenous uncertainty. In the next chapter we will summarize the FIS problems under decision dependent randomness and their relation to the robust coloring. We will extend previous results with proposing a new FIS problem with maintenance. This problem is a specific case of a decision-dependent probabilities as it allows to use a specific type of a job - the maintenance, which positively impacts the probability distributions of job delays. We start with defining a problem, where maintenance must be assigned only before regular jobs and then we propose the general case, when maintenances appears during the whole processing period. We show why this approach leads to an optimal solution and provide a detailed example of a small problem. We also discuss some extensions of our problem. Finally, we conduct a numerical study. We solve the FIS maintenance problem with the *Cplex* solver for a few different settings of inputs. It seems that the maintenance is useful only for certain settings, such as jobs with high probability of having a delay or the price of outsourcing being much higher than the cost of maintenance. It is also shown that the problem quickly becomes too computationally demanding and it works well only for relatively small settings.

Keywords: fixed interval scheduling, decision dependent randomness, maintenance

# Contents

# Introduction

In this thesis we will focus on a subclass of scheduling problems, the fixed interval scheduling (FIS) problems. This class of problems assumes that starting and finishing times of jobs are given and our goal is to assign them to available machines. In practise we often observe random delays of jobs which have negative impact on our schedule and therefore the goal is often to find the schedule which is the least likely to delay. In this thesis we will introduce a new approach to solve this problem with an assumption that there exists a special kind of job which for a given price improves the probability distribution of delays.

Our motivation for this approach can be demonstrated on an example of a logistic centre shipping off packages. There are scheduled arrivals of trucks to deliver the package, however there is always a possibility of a delay in finding the appropriate order and therefore the truck having to wait, which is associated with a certain penalty (e.g. paying the driver more or giving the customer a discount). However, there is the option to rearrange the warehouse, which will lead into a quicker dispatch. Our decision to carry out the rearrangement then positively affects the probability distribution of delays. On the other hand this will mean additional expenses and maybe even a delay caused by the rearrangement, we are actually solving a multi-objective problem.

The first chapter is dedicated to introduction into FIS problems and decision dependent randomness, which means that our decisions about assignments are affecting the resulting probability distribution.

We will follow the work [Branda et al., 2015], which shows the equivalence between robust coloring problems (RCP) and FIS problems and in the second chapter we will introduce a new FIS maintenance problem based on the RCP formulation. In this case we have special jobs - maintenances, whose assignment allow us to positively affect the probability distributions of random delays. Using the maintenance is then a powerfull tool to reduce job overlaps. We will show the problem formulation and interpret constraints, we will also provide theoretical results proving that this formulation will help us achieve better values of objective functions. To demonstrate our problem, we will conclude the chapter with detailed example of a smaller instance of the problem. We are aware of limitations of our problem therefore we will also discuss some possible extensions of the problem.

In the last chapter we will provide results of a numerical study and consider when the maintenance problem brings improved results. We will also discuss different settings of parameters which were used and how they affected the results.

# 1. Fixed interval scheduling problems and stochastic programming

## 1.1 Fixed interval scheduling

Scheduling problems in general focus on solving the task how to assign certain processes to given resources, for example how to assign computing processes to different servers or how to assign lectures in available classrooms. These processes are called *jobs* and available resources for them are called *machines*.

Fixed interval scheduling problems, which will be in the next parts of thesis abbreviated as FIS, are a specific subclass of scheduling problems. For each job we are given the information about its starting time and its length (or its defined finishing time). We want to assign jobs to machines such that we can start processing each job in its given starting times, otherwise there is a certain penalty. This type of tasks has a wide application in real life problems, for example we want to schedule arrivals of trains to a station and their assignment to platforms. The arrival of a train is given and if it has to wait for an available platform we need to return money to passengers.

If the starting times and finishing times are given and cannot change, we speak about *deterministic scheduling* problems. There are different modifications of this problem. A typical task is to search the minimum number of machines to process all jobs or in case of fixed number of jobs searching for a schedule which is optimal with respect to priority of jobs.

In this thesis we will discuss the case of *stochastic scheduling*. In this case we consider that each job has a random processing time, specifically a random non-negative delay. Then we want to minimize the number of times when a delay affected processing of a following job, this case is called a *job overlap* and our task is to minimize the expected value of overlaps. In this type of problems we often consider a possibility of outsourcing, which means, that for a certain price we can hire more machines to process overlapping jobs.

Another problem might be to optimize the reliability of a schedule, i.e. maximize the probability that there will be no overlaps during the processing time.

Scheduling problems might be divided into subclasses according to the use of *preemption*. The book [Rossi et al., 2006] defined them as follows.

- The non-preemptive scheduling, where each job must be processed without interruption from its start to finish, i.e. jobs cannot be interrupted. The final schedule is a set of starting and finishing times for each machine.
- The preemptive scheduling, when jobs can be interrupted at any time to let some other job (typically one with a higher priority) to be processed instead. The final schedule is then more complicated then just the set of starting and finishing times.

In this thesis we will consider only the case of non-preemptive scheduling.

## 1.2 Exogenous and endogenous uncertainty in stochastic programming

Let us start with an explanation what the endogenous uncertainty is and how it differs from exogenous uncertainty. We will use formulations from the article Dupačová [2006].

Assume we have a stochastic programming problem in which we want to choose the best solution $x$ before a we observe the value of a random event $\omega \in \Omega$. For example we want to invest money in a certain stock, where $x$ means the decision which stock we choose, and $\omega$ is a change in the stock value. The cost of choosing $x$ is dependent on $\omega$ and can be described by a real-valued function $f(x, \omega)$. In the stock investing example the function $f$ describes the money lost in the investment.

Let $x$ be an element of $\mathcal{X}$, a nonempty finite closed subset of an euclidean space, let $P$ denote a know probability distribution of $\omega \in \Omega$. Then we write our optimization problem as

$$\min_{x \in \mathcal{X}} F(x; P) := \mathsf{E}_P f(x, \omega). \tag{1.1}$$

In case when the distribution of $\omega$ is independent on $x$ we speak about the *exogenous uncertainty*. However, if the distribution of $\omega$ depends on $x$, we call it the *endogenous uncertainty* or the *decision-dependent randomness*. Then we shall rewrite our problem with explicitly stating the dependence of distribution of $\omega$ on $x$ in the following form:

$$\min_{x \in \mathcal{X}} F(x) := \int_{\Omega} f(x, \omega) P_x(d\omega). \tag{1.2}$$

Assume that the expected value exist and is finite for all $x \in \mathcal{X}$ and also that the optimal solution exists so the above formulation is well defined.

This decision dependent model brings new complications because properties of the objective function have changed compared to the independent model. For example, convexity in $x$ of the function $f(x, \omega) \forall \omega \in \Omega$ would be sufficient to derive the convexity of $F(x; P)$, however, in the dependent model it does not imply the convexity of the objective function $F(x; P_x)$.

Under certain circumstances we can remove the dependence by a suitable transformation of $P_x$. Suppose there exist densities $p(x, \omega)$ of probability distributions $P_x$ with respect to a common probability measure $\mathcal{Q}$. Then the objective function in the problem 1.2 can be rewritten as

$$F(x) = \int_{\Omega} f(x, \omega) p(x, \omega) Q(d\omega) = \int_{\Omega} \tilde{f}(x, \omega) Q(d\omega),$$

where $\tilde{f}(x, \omega) = f(x, \omega) p(x, \omega)$ and the probability distribution $\mathcal{Q}$ is decision independent. The properties of the objective function still depend on the type of dependence of $P$ on $x$. In a special case described in the article Varaiya and Wets [1988] it is shown that if the dependence on $x$ can be written as $P(\omega; x) = Q(\omega + Hx)$ (where $Q$ is a known probability distribution function and $H$ is a given matrix of the appropriate size), then the problem can be rewritten in the following form:

$$\min_{x \in \mathcal{X}} \int f(x, \zeta - Hx) Q(d\zeta).$$

Then properties of the objective function depend on properties of $f(x, \omega)$ viewed as a function of $(x, \omega)$ jointly. When the dependence of probability distribution $P$ on the decision variable $x$ can be removed by a suitable transformation of the distribution, it is called the push-in technique. Unfortunately, such a transformation is not always available.

The article Dupačová [2006] also stresses that in the case of multistage stochastic problems with decision-dependent probabilities a special attention is needed, because the first stage decisions as well as the decisions from later stages may all affect the information available to the decision maker. Decisions at each stage may have impact on the probability in the subsequent stage of the problem. It is even possible that decisions influence at which time the uncertainty gets resolved which means that nonanticipativity conditions are decision dependent.

## 1.3 Decision dependent probabilities

The recent paper Hellemo et al. [2018] comes with a new taxonomy of endogenous uncertainty in stochastic programming. The first one is the class called *decision-dependent probabilities*, which can be generally described by the equation (1.2). We are trying to solve a problem with an unknown probability distribution, while the true distribution depends on decision variables. Specifically, only probabilities depend on decision variables and nonanticipativity constraints are not manipulated by decision variables. In the following chapters we will be discussing a problem from this class.

The other class discussed by the article are *decision-dependent information schema* problems, when the time of the information revelation depends on decision variables. Unlike the previous class, this type of problems might affect nonanticipativity conditions. There is also third class of problems which includes combinations of both decision-dependent probabilities and information schema.

The article by Dupačová [Dupačová, 2006] identifies two important sub-classes of a decision-dependent probability problem:

- decision-dependent parameters, when the probability distribution is known and the decision influences only parameters, and
- decision-dependent distribution selection, when there is a finite fixed set of probability distributions and the decision may be related to the choice of a probability distribution from this set.

The article [Hellemo et al., 2018] adds a third subclass, the decision-dependent distribution distortion. In this subclass we are given some prior set of probabilities for a distribution with known parameters. Decision variables control how these probability distributions are distorted.

# 2. Maintenance in the decision dependent FIS problem

In this chapter we will discuss a specific subclass of a decision dependent FIS problem. Our main task is to optimize a schedule of jobs on independent identical machines. All of these jobs have a certain probability of a delay. However in this chapter we will introduce a maintenance - a special job which can be applied to a machine and as a result decreases possibility of a delay of all following jobs processed by this machine. We will consider an offline schedule, which means that the schedule is prepared before the processing starts and we are using only information known in this time.

## 2.1 Decision dependent FIS - problem formulation

First of all we will start with formulation of a problem without maintenance jobs. Scheduling jobs with an unknown random delay is a special case of decision dependent problem. Following the formulation 1.2 from the previous chapter, our problem can be generally written as

$$\min_{x\in\mathcal{X}} \mathsf{E}_{P_x}[q(x,\xi)] = \int_{\Xi} q(x,\xi)P_x(d\xi). \tag{2.1}$$

Next we assume that rather than the decision vector $x$ we have a vector $(x, z)$ in which only the $z$ part of the decision vector influences the underlying probability distribution.

To formulate the problem properly let us assume that all starting and processing times of jobs are given and known, what we do not known are random delays of jobs and their distribution. The random delay represents several causes of a delay, a delay in the starting time, a delay in the finishing time or a delay due to an unexpected machine failure. However, we allow that job ends on time, in other words it is probable that the delay equals zero. We want to process all jobs while each machine can process at most one job at time and for this purpose we assume that the set of machines is sufficiently big. We also assume that all machines are identical and we do not allow a preemption, which means that job cannot be interrupted.

We will be using the following notation.

- $\mathcal{T} = [0, T]$ – interval from starting time 0 to end time $T$, we consider the time $t$ as a continuous variable,
- $\mathcal{C}$ – the set of machines,
- $\mathcal{J}$ – the set of jobs, for each one of them we define:
  - $s_j$ – starting time of the job $j$, $\mathcal{S}$ represents the set of all starting times,

- $f_j^0$ – the prescribed completion time,
- $D_j(\xi)$ – the random delay, a non-negative value, where $D_j(\xi) = 0$ has a positive probability,
- $f_j(\xi) = f_j^0 + D_j(\xi)$ - random finishing time composed of prescribed finishing time and a random delay,

- $x_{jc}$ – a binary decision variable, $j \in \mathcal{J}, c \in \mathcal{C}$, equals one if the job $j$ is assigned to the machine $c$, zero otherwise.

Let us note that we assume a job cannot have processing time equal to 0 and $s_j < f_j^0, \forall j \in \mathcal{J}$.

We want to use the formulation of our problem with minimizing the expected number of overlaps caused by random delay. For this purpose we shall introduce $\tilde{y}_{jc}(\xi)$ - an integer variable representing the number of jobs which machine $c$ cannot process because of a random delay caused by a job $j$.

The appropriate formulation of a problem of minimizing the expected count of overlaps was shown by Branda in the article [Branda et al., 2015]:

$$\min_{x,\tilde{y}} \mathsf{E}_P \left[ \sum_{c\in\mathcal{C}} \sum_{j\in\mathcal{J}} \tilde{y}_{jc}(\xi) \right] \tag{2.2a}$$

$$\text{s.t.} \sum_{s_j \le t < f_j} x_{jc} \le 1, t \in \mathcal{S}, c \in \mathcal{C}, \tag{2.2b}$$

$$\sum_{c\in\mathcal{C}} x_{jc} = 1, j \in \mathcal{J}, \tag{2.2c}$$

$$\sum_{k: f_j^0 \le s_k < f_j(\xi)} x_{kc} \le \tilde{y}_{jc}(\xi) + |\mathcal{J}|(1 - x_{jc}), c \in \mathcal{C}, j \in \mathcal{J}, \xi \in \Xi, \tag{2.2d}$$

$$x_{jc} \in 0, 1, c \in \mathcal{C}, j \in \mathcal{J}, \tag{2.2e}$$

$$\tilde{y}_{jc}(\xi) \in \mathbb{N}, c \in \mathcal{C}, j \in \mathcal{J}, \xi \in \Xi. \tag{2.2f}$$

First two constraints (2.2b) and (2.2c) ensure that at most one job in each time is assigned to a machine and that each job is assigned to a certain machine. The third constraint (2.2d) expresses the number of unprocessed jobs on a machine $c$. These jobs could not have started in starting times $s_k$, because the job $j$ was delayed. The constraint is valid only if the job $j$ is assigned to the machine $c$, otherwise $\tilde{y}_{jc}(\xi)$ is set to zero, thank to the other term $|\mathcal{J}|(1 - x_{jc})$ on the right side of the inequality.

The objective function (2.2a) then represents the number of jobs, which were not processed on their initially assigned machine because of delays. If the outsourcing is possible, than it can be interpreted as the extra capacity which has to be bought to process all jobs (after multiplication by the outsourcing price). In case that the outsourcing is not possible, it might be a better approach to search for a schedule with the highest reliability. Procedures for this were shown at [Branda, 2018].

### 2.1.1 Two-stage reformulation

Let us define a general two-stage stochastic problem:

$$\begin{aligned} \min \quad & c^T x + \mathsf{E}\left[q^T(\xi) y(\xi)\right] \\ \text{s.t.} \quad & Ax = b, \\ & T(\xi)x + Wy(\xi) = h(\xi), \\ & x \in \mathbb{R}^n_+, y \in \mathbb{R}^m_+. \end{aligned} \tag{2.3}$$

In this formulation, $x$ is the first stage decision variable, $y$ is the second stage decision variable and we can assume they are real (non-negative) or modify the formulation of the problem assuming they are integer. The variable $x$ represents the decision we have to make in the first stage, before a realization of a random element $\xi$. At the second stage, the result of $\xi$ is already known and we are making the decision $y$. $T(\xi)$ is called the random technology matrix, $W$ is the weighting matrix and vectors $b, h(\xi)$ are some suitable right hand side vectors. Vectors $c$ and $q(\xi)$ represent costs of the first stage and second stage decision, respectively. We are minimizing the costs of decision $x$ (known at the first stage) and expected costs of the decision $y$ (unknown before the realization of $\xi$). If the term $W$ is an identity matrix, then constraints with $y$ can be eliminated and this simplified problem is called the simple recourse.

Now we see that, with certain modifications, the problem (2.2) can be considered as an integer two stage stochastic problem, where $\tilde{y}_{jc}(\xi)$ are second stage decision variable, and $x_{jc}$ are the first stage variables.

From the third constraint (2.2d) we obtain the relationship:

$$\tilde{y}_{jc}(\xi) \geq \sum_{k: f_j^0 \leq s_k < f_j(\xi)} x_{kc} - |\mathcal{J}|(1 - x_{jc}), c \in \mathcal{C}, j \in \mathcal{J}, \xi \in \Xi.$$

Since we have no other constraint on $\tilde{y}_{jc}(\xi)$, only that it is an integer, the minimal value is achieved when this inequality becomes equality or when $\tilde{y}_{jc}(\xi)$ equals zero and the other side of the inequality is negative. Minimizing a non-decreasing function of $\tilde{y}_{jc}(\xi)$ is then equal to minimizing the function of

$$\left[ \sum_{k: f_j^0 \leq s_k < f_j(\xi)} x_{kc} - |\mathcal{J}|(1 - x_{jc}) \right]_+ .$$

Let us define a random technology matrix $T_{jk}$ with the following formula:

$$T_{jk}(\xi) = \begin{cases} \mathbb{I}_{[D_j(\xi) > s_k - f_j^0]}, \text{if } s_k \geq f_j^0, \\ 0, \text{otherwise.} \end{cases}$$

An element of the technology matrix equals 1 if the random delay $D_j(\xi)$ for the $j$-th job is greater than the time window between the starting time $s_k$ of the $k$-th job and the prescribed finishing time $f_j^0$ of the $j$-th job.

Now we can rewrite the original problem as a simple recourse:

$$\begin{aligned}
&\min_x \mathsf{E}_\xi \left[ \sum_{c\in\mathcal{C}} \sum_{j\in\mathcal{J}} \left[ \sum_{k\in\mathcal{J}} T_{jk}(\xi) x_{kc} - |\mathcal{J}|(1-x_{jc}) \right]_+ \right], \\
&\text{s.t.} \sum_{s_j \le t < f_j} x_{jc} \le 1, t \in \mathcal{S}, c \in \mathcal{C}, \\
&\sum_{c\in\mathcal{C}} x_{jc} = 1, j \in \mathcal{J}, \\
&x_{jc} \in 0,1, c \in \mathcal{C}, j \in \mathcal{J}, \\
&\tilde{y}_{jc}(\xi) \in \mathbb{N}, c \in \mathcal{C}, j \in \mathcal{J}, \xi \in \Xi
\end{aligned} \tag{2.4}$$

## 2.2 Relation to robust coloring problems

In the graph theory, coloring problems deal with assigning colors to vertices of a graph in such a way that no two adjacent vertices share the same color. Also in the FIS problem with given number of machines we are dealing with similar problem - how to assign jobs to machines so that two subsequent jobs assigned to the same machine do not overlap. The reformulation of a FIS problem minimizing the expected number of overlaps to a robust coloring problem was shown in the article Branda et al. [2015].

The vertices of this graph are jobs $\mathcal{J}$ and the edges are overlapping jobs. The set of edges can be divided into two sets: $E$, which includes edges between jobs which overlap with probability equal to one (because of their prescribed starting and finishing times) and a set $\overline{E}$ of jobs overlapping with probability less than one. Precisely:

$$\begin{aligned}
E &= \{(j,j\prime) : P(s_{j\prime} < f_j^0 + D_j(\xi)) = 1\} \\
&= \{(j,j\prime) : s_j \le s_{j\prime} \wedge s_{j\prime} \le f_j^0\} \\
\overline{E} &= \{(j,j\prime) : P(s_{j\prime} \le f_j^0 + D_j(\xi)) \le 1\} \\
\Leftrightarrow \overline{E} &= \{(j,j\prime) : P(D_j(\xi) \ge s_{j\prime} - f_j^0) \le 1\} \\
\Leftrightarrow \overline{E} &= \{(j,j\prime) : s_{j\prime} > f_j^0\}
\end{aligned}$$

Clearly, $E \cap \overline{E} = \emptyset$.

The set of machines $\mathcal{C}$ represents the set of colours. Although traditionally coloring problems aim to find the chromatic number of a graph, now we assume that the coloring of the graph $(\mathcal{J}, \mathcal{E})$ is possible with respect to the set of hard edges $E$. In other words we assume the set of machines is sufficient, which means that we have at least as many machines as is the chromatic number of the graph constructed as described above. Furthermore, in a robust coloring problem we define a penalty $q_{jj\prime}$ which penalizes the overlap of jobs $j$ and $j\prime$ caused by a random delay $D_j(\xi)$. Our task is assign colors (machines) to vertices (jobs) such that we minimize the penalization. The definition of edges implies that we have an oriented graph (the orientation represents the order of jobs), however coloring problems are the same for oriented an not oriented graphs.

Then the problem can be written in the following way introduced by Yaňez and Ramirez in the article 2003:

$$\min_{x,y} \sum_{\{j,j\prime\}\in\overline{E}} q_{jj\prime} y_{jj\prime}$$

$$\text{s.t.} \sum_{c\in\mathcal{C}} x_{jc} = 1, j \in \mathcal{J},$$
$$x_{jc} + x_{j\prime c} \leq 1, \{j, j\prime\} \in E, c \in \mathcal{C},$$
$$x_{jc} + x_{j\prime c} \leq 1 + y_{jj\prime}, \{j, j\prime\} \in \overline{E}, c \in \mathcal{C},$$
$$x_{jc} \in \{0, 1\}, c \in \mathcal{C}, j \in \mathcal{J},$$
$$y_{jj\prime} \in \{0, 1\}, \{j, j\prime\} \in \overline{E}. \tag{2.5}$$

The objective function penalizes possibly overlapping jobs if they are assigned to the same machine. To assign a penalty we use the variable $y_{jj\prime}$, which equals one if and only if vertices $j$ and $j\prime$ share the same color $c$ (in the scheduling terminology jobs $j$ and $j\prime$ are assigned to the same machine $c$).

We want to prove that the robust coloring formulation is equivalent to the two stage formulation (2.2). Let us define the penalty as $q_{jj\prime} = P(D_j(\xi) > s_{j\prime} - f_j^0)$. Branda in 2015 provides the proof for this equivalence which we will show here.

**Lemma 1.** *The two-stage stochastic programming formulation 2.2 of minimizing the expected number of overlaps is equivalent to the robust coloring problem 2.5 with the penalties* $q_{jj\prime} = P(D_j(\xi) > s_{j\prime} - f_j^0)$.

*Proof.* The constraints of the problems define identical sets of feasible solutions. The first condition guarantees that each job is assigned to exactly one machine. The second constraint in 2.5 ensures, that at each time at most one job is assigned to a machine, this is a result from the definition of the set $E$. The third constraint, together with the binary definition of $y_{jj\prime}$, is equivalent to the inequality $\sum_{k\in\mathcal{J}} T_{jkc}(\xi)x_{kc} - |\mathcal{J}|(1 - x_{jc}) \leq \tilde{y}_{jc}(\xi)$.

Therefore, we can discuss only the relation of the objective functions, in particular a contribution of a node $j$. We consider $N_j$ subsequent jobs of a job $j$ which are processed by a machine $c$, i.e. $x_{jc} = x_{k_1c} = \cdots = x_{k_{N_j}c} = 1$ with $f_j^0 \leq s_{k_1} < f_{k_1}^0 \cdots \leq s_{k_{N_j}} < f_{k_{N_j}c}^0$. We can obtain an explicit expression of the expectation of $y_{jc}(\xi)$ :

$$\begin{aligned}
\mathsf{E}_\xi[\tilde{y}_{jc}(\xi)] &= \sum_{n=1}^{N_j-1} n \cdot P(s_{k_n} - f_j^0 < D_j(\xi) \leq s_{k_{n+1}} - f_j^0) \\
&\quad + N_j \cdot P(D_j(\xi) > s_{k_{N_j}} - f_j^0) \\
&= \sum_{n=1}^{N_j-1} n \cdot (P(D_j(\xi) > s_{k_n} - f_j^0) - P(D_j(\xi) > s_{k_{n+1}} - f_j^0)) \\
&\quad + N_j \cdot P(D_j(\xi) > s_{k_{N_j}} - f_j^0) \\
&= \sum_{n=1}^{N_j} P(D_j(\xi) > s_{k_n} - f_j^0) \\
&= \sum_{n:\{j,k_n\}\in\overline{E}} q_{jk_n}
\end{aligned}$$

Thus, we have obtained the equality of objective functions

$$\mathsf{E}_\xi \left[ \sum_{c\in\mathcal{C}} \sum_{j\in\mathcal{J}} \tilde{y}_{jc}(\xi) \right] = \sum_{\{j,j\prime\}\in\overline{E}} q_{jj\prime} y_{jj\prime}.$$

□

In the article Yáñez and Ramirez [2003] authors prove that the robust coloring problem is NP-hard and classic integer programming algorithms work only for problems of smaller instances. Authors then propose using a modification of genetic algorithm for larger problems as it achieves solutions in a feasible time. Branda in the article Branda et al. [2015] used *Gams* to obtain exact solutions of smaller instances of the problem and then a modified tabu search algorithm to achieve approximate solutions of larger instances.

## 2.3 Maintenance

In the previous part we defined a FIS problem with the objective to minimize the expected number of overlaps and we have shown that it can be rewritten as a robust graph coloring problem. Now we want to introduce a new element, the *maintenance* jobs. Maintenance is a specific job with given starting time, expected finishing time and also a possibility of a random delay, however if the maintenance is carried out on the given machine, it decreases the probabilities of delays of all following jobs processed by the machine after the maintenance. The maintenance also has a certain cost and unlike other jobs does not have to be processed.

This is in practice an important yet not examined case. We can imagine, for example, that we need to schedule certain projects between employees with the possibility to send and employee to a paid training, which will help him work more efficiently. In this case we are trying to balance the increased cost for trainings with the decreased penalty for delayed projects.

Let us provide a formal notation. Assume that we have a set of available maintenance jobs $\mathcal{J}^M$ and each of them is defined by a starting time $s_m$ and a prescribed finishing time $f_m^0$, each maintenance costs a price $c_m$. This general definitions allows random delays of maintenance jobs as well. Let us note the set of regular jobs as $\mathcal{J}^I$, then the set of all jobs is composed of this two subsets: $\mathcal{J} = \mathcal{J}^M \cup \mathcal{J}^I$.

Let us assume that possible overlaps of jobs might be solved by outsourcing (paying for another machines to process them) for a fixed price $p$ for an additional machine. Our objective is then to minimize expenses composed of costs for maintenance and the price paid for outsourcing. Using the maintenance more often increases price paid for the service, however it also decreases price paid for possible outsourcing needed, it is a case of a multi-objective optimization.

The expected number of job overlaps was defined as

$$\min_x \mathsf{E}_{P(x)} \left[ \sum_{c\in\mathcal{C}} \sum_{j\in\mathcal{J}} \left[ \sum_{k:f_j^0 \le s_k < f_j(\xi)} x_{kc} - |\mathcal{J}|(1-x_{jc}) \right]_+ \right].$$

Therefore the expected price paid for outscouring of overlapping jobs is

$$p\,\mathsf{E}_{P(x)} \left[ \sum_{c\in\mathcal{C}} \sum_{j\in\mathcal{J}} \left[ \sum_{k:f_j^0 \le s_k < f_j(\xi)} x_{kc} - |\mathcal{J}|(1-x_{jc}) \right]_+ \right],$$

where $P(x)$ is the probability distribution of delays dependent on the decision variables $x$. The costs of maintenance application has a more straightforward formulation:

$$\sum_{m \in \mathcal{J}^M} c_m \sum_{c \in \mathcal{C}} x_{mc}.$$

The problem of minimizing costs caused by job delays in our case on maintenance assignment can be formulated as follows:

$$\min_x \sum_{m \in \mathcal{J}^M} c_m \sum_{c \in \mathcal{C}} x_{mc} + p \, \mathsf{E}_{P(x)} \left[ \sum_{c \in \mathcal{C}} \sum_{j \in \mathcal{J}} \left[ \sum_{k : f_j^0 \leq s_k < f_j(\xi)} x_{kc} - |\mathcal{J}|(1 - x_{jc}) \right]_+ \right] \tag{2.6a}$$

$$\text{s.t.} \sum_{c \in \mathcal{C}} x_{jc} = 1, j \in \mathcal{J}^I, \tag{2.6b}$$

$$\sum_{c \in \mathcal{C}} x_{mc} \leq 1, m \in \mathcal{J}^M, \tag{2.6c}$$

$$x_{jc} \in \{0, 1\}, c \in \mathcal{C}, j \in \mathcal{J}. \tag{2.6d}$$

The conditions represent the demand to process all regular jobs while maintenance jobs do not have to be processed at all.

The goal of maintenance is to improve the distribution of delays $P$ to a "better" distribution $\tilde{P}$. This means that after assigning and processing the maintenance on a selected machine, we assume that the probability of not finishing a job before next job starts (creating an overlap) is smaller than before. Let us write it as $\tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0) < P(D_j(\xi) > s_{j\prime} - f_j^0)$. To introduce this improvement into the problem specification, we shall define a new variable $\Delta_{jj\prime}$ with the following formula:

$$\Delta_{jj\prime} = P(D_j(\xi) > s_{j\prime} - f_j^0) - \tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0), \{j, j\prime\} \in \overline{E}.$$

Because the set $\overline{E}$ includes overlaps between all jobs, it includes also overlaps between maintenances and regular jobs. However we do not want to apply the improvement caused by a maintenance $m$ to the overlap of this maintenance $m$ and a following job $j\prime$. Therefore to have a well-defined problem, we set

$$\Delta_{jj\prime} = 0, \text{if } j \in \mathcal{J}^M$$

Obviously, $\Delta_{jj\prime} \in [0, 1]$.

For example, if the original probability distribution of delays is exponential with parameter $\lambda$, then $P(D_j(\xi) > s_{j\prime} - f_j^0) = exp(-\lambda(s_{j\prime} - f_j^0))$ and the improved distribution $\tilde{P}$ is exponential with parameter $\tilde{\lambda}$, then the improvement means that $\tilde{\lambda} \geq \lambda$.

Previously we have shown the equivalence between a robust coloring problem and fixed interval scheduling, now we will want to modify it to include the maintenance periods. We already know that we can write

$$
\begin{aligned}
&\mathsf{E}_{P(x)}\left[\sum_{c\in\mathcal{C}}\sum_{j\in\mathcal{J}}\left[\sum_{k:f_j^0\le s_k<f_j(\xi)} x_{kc} - |\mathcal{J}|(1-x_{jc})\right]_+\right] \\
&= \mathsf{E}_{\xi}\left[\sum_{c\in\mathcal{C}}\sum_{j\in\mathcal{J}}\tilde{y}_{jc}(\xi)\right] \\
&= \sum_{\{j,j\prime\}\in\overline{E}} q_{jj\prime}y_{jj\prime},
\end{aligned}
$$

where $q_{jj\prime} = P(D_j(\xi) > s_{j\prime} - f_j^0)$. However, if the maintenance was applied to a machine $c$ before processing jobs $j$ and $j\prime$, then penalty function is rather $\tilde{q}_{jj\prime} = \tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0)$. We will use the variable $\Delta_{jj\prime}$ defined earlier to rewrite the expected number of overlaps as

$$
\left(\sum_{\{j,j\prime\}\in\overline{E}} q_{jj\prime}y_{jj\prime} - \sum_{\{j,j\prime\}\in\overline{E}} \Delta_{jj\prime}z_{jj\prime}\right),
$$

where $z_{jj\prime}$ is a binary variable which for a machine $c$ and maintenance $m$ equals 1 if and only if the maintenance $m$ was applied to the machine $c$ before consequent jobs $j$ and $j\prime$.

### 2.3.1 Maintenance at the beginning

We will start with a special case when all maintenance jobs must be assigned only at the beginning of the processing time. Then, if a maintenance is applied to a machine, all following jobs on this machine will be processed with the improved probability distribution $\tilde{P}$. We will be using the variable $z_{jj\prime}$ to label edges $\{j, j\prime\}$ which will be affected by the maintenance. Because the improvement brought by the maintenance cannot be applied to the edge between the maintenance itself and a following job, we will use notation $\overline{E}_I = \{(j, j\prime) : s_{j\prime} \ge f_j^0, j, j\prime \in \mathcal{J}^I\}, \overline{E}_I \subset \overline{E}$ to distinguish edges that can be influence by a maintenance. To have a well defined problem we set values of variable $z_{jj\prime}$ to zero, if $j = m, m \in \mathcal{J}^M$.

Our original intention was to define the problem as follows. However, we found out we need to propose different constraints for the variable $z_{jj\prime}$. We provide the original formulation (2.7) and then new constraints with the explanation why the former did not work.

$$\min_{x,y,z} \sum_{m \in \mathcal{J}^M} c_m \sum_{c \in \mathcal{C}} x_{mc} + p \left( \sum_{\{j,j\prime\} \in \overline{E}} q_{jj\prime} y_{jj\prime} - \sum_{\{j,j\prime\} \in \overline{E}} \Delta_{jj\prime} z_{jj\prime} \right) \tag{2.7a}$$

$$\text{s.t.} \sum_{c \in \mathcal{C}} x_{jc} = 1, j \in \mathcal{J}^I, \tag{2.7b}$$

$$\sum_{c \in \mathcal{C}} x_{mc} \leq 1, m \in \mathcal{J}^M, \tag{2.7c}$$

$$\sum_{m \in \mathcal{J}^M} x_{mc} \leq 1, c \in \mathcal{C}, \tag{2.7d}$$

$$x_{jc} + x_{j\prime c} \leq 1, \{j, j\prime\} \in E, c \in \mathcal{C}, \tag{2.7e}$$

$$x_{jc} + x_{j\prime c} \leq 1 + y_{jj\prime}, \{j, j\prime\} \in \overline{E}, c \in \mathcal{C}, \tag{2.7f}$$

$$x_{jc} + x_{mc} \geq 1 + z_{jj\prime}, \{j, j\prime\} \in \overline{E}_I, m \in \mathcal{J}^M, c \in \mathcal{C}, \tag{2.7g}$$

$$x_{jc} + x_{j\prime c} \geq 2 z_{jj\prime}, \{j, j\prime\} \in \overline{E}_I, c \in \mathcal{C}, \tag{2.7h}$$

$$z_{jj\prime} = 0, j \in \mathcal{J}^M, \{j, j\prime\} \in \overline{E}_I, \tag{2.7i}$$

$$x_{jc} \in \{0, 1\}, c \in \mathcal{C}, j \in \mathcal{J} = \mathcal{J}^M \cup \mathcal{J}^I, \tag{2.7j}$$

$$y_{jj\prime} \in \{0, 1\}, \{j, j\prime\} \in \overline{E}, \tag{2.7k}$$

$$z_{jj\prime} \in \{0, 1\}, \{j, j\prime\} \in \overline{E} \quad . \tag{2.7l}$$

The objective is to minimize the expenses caused by either paying for maintenance or paying for additional machines to cover job overlaps. The first condition (2.7b) demands that all regular jobs are processed. The second condition (2.7c) specifies that maintenance jobs might not have to be processed and that each job or maintenance might be processed by only one machine. The third condition (2.7d) was added to prevent assigning more maintenances on one machine, we want to choose only one maintenance with its effect lasting for the rest of the processing period. The fourth condition (2.7e) forbids to assign two surely overlapping jobs to one machine, while the fifth condition (2.7f) allows to two possibly overlapping jobs to be assigned to one machine for the penalty $q_{jj\prime}$.

Next conditions were added to model the dependence of the probability distribution on the decision. The resulting probability distribution for a delay of a job $j$ will be either $P$ or $\tilde{P}$, depending on the value of variables $y_{jj\prime}$ and $z_{jj\prime}$. The sixth and seventh constraints (2.7g) and (2.7h) specify when the variable $z_{jj\prime}$ can be equal to one. That is in a case of $x_{jc}$ and $x_{mc}$ equal to one (the maintenance was applied on the chosen machine and the job $j$ is assigned to this machine) and in case of $x_{jc}$ and $x_{j\prime c}$ equal to one (the pair of possibly overlapping jobs $j$ and $j\prime$ was assigned to this machine). Unfortunately this formulation of conditions does not work well for certain inputs. If neither the job $j$ nor the maintenance $m$ are assigned to a machine $c$, then the condition

$$x_{jc} + x_{mc} \geq 1 + z_{jj\prime}, \{j, j\prime\} \in \overline{E}, m \in \mathcal{J}^M, c \in \mathcal{C}$$

becomes invalid. If we put this formulation of our problem into a solver it will result either in warning that there is no feasible solution or obviously wrong solutions such as assigning maintenances to machines where there are no overlaps. Therefore we propose a different formulation.

It is necessary to formulate constraints with $z_{jj\prime}$ alternatively by introducing an auxiliary variable $\tilde{z}_{jj\prime mc}$, which is equal to one if a maintenance $m$ was applied to machine $c$ and the pair of jobs $j$ and $j\prime$ will be processed on this machine with an improved probability of delay $\tilde{P}$:

$$\begin{aligned}
&x_{jc} + x_{j\prime c} + x_{mc} \leq 2 + \tilde{z}_{jj\prime mc}, \{j, j\prime\} \in \overline{E}_I, m \in \mathcal{J}^M, c \in \mathcal{C},\\
&x_{jc} + x_{j\prime c} \geq 2\tilde{z}_{jj\prime mc}, \{j, j\prime\} \in \overline{E}_I, m \in \mathcal{J}^M, c \in \mathcal{C},\\
&x_{jc} + x_{mc} \geq 2\tilde{z}_{jj\prime mc}, \{j, j\prime\} \in \overline{E}_I, m \in \mathcal{J}^M, c \in \mathcal{C},\\
&z_{jj\prime} = \sum_{m \in \mathcal{J}^M} \sum_{c \in \mathcal{C}} \tilde{z}_{jj\prime mc}, \{j, j\prime\} \in \overline{E}_I.
\end{aligned} \tag{2.8}$$

The first condition forces $\tilde{z}_{jj\prime mc}$ to equal one, if for a machine $c$ the maintenance was applied and possibly overlapping jobs $j$ and $j\prime$ were also assigned to this machine. This allows us to apply the improvement $\Delta_{jj\prime}$ in the objective function. On the other hand, following two conditions reassure, that $\Delta_{jj\prime}$ is not applied in any other case, as they force $\tilde{z}_{jj\prime mc}$ to equal zero if at least one of $x_{jc}, x_{j\prime c}, x_{mc}$ do not equal zero. In other words, if neither maintenance nor one of the overlapping jobs were assigned to this machine, the improvement $\Delta_{jj\prime}$ will not be applied in the objective function. Thanks to the exclusivity of assignment (a pair of jobs can be assigned to only one machine as well as the maintenance can be assigned to only one machine), the value of $z_{jj\prime}$ can be obtained as a sum over auxiliary values $\tilde{z}_{jj\prime mc}$. This equation has a straightforward interpretation - the improvement of a delay can be applied on a pair of jobs if there exists one specific machine to which both the maintenance and the pair of overlapping jobs were assigned.

This proposed definition is more strict and does not allow hypothetical cases such as $x_{jc} = 1, x_{j\prime c} = 1, x_{mc} = 1, z_{jj\prime} = 0$ for a given machine $c$, maintenance job $m$ and possibly overlapping jobs $j$ and $j\prime$. This particular case was allowed (i.e. equation are valid for this case) by constraints in the original formulation, however, due to the minimization of a negative term, the objective function would choose $z_{jj\prime}$ equal to one whenever possible and this case would not have be chosen as an optimal solution.

If we use this proposed specification of constraints, we can see that variables $y_{jj\prime}$ and $z_{jj\prime}$ are now rather auxiliary variables representing specific values of the decision variable $x$.

$$y_{jj\prime} = 1 \Leftarrow (x_{jc} = 1 \wedge x_{j\prime c} = 1), \{j, j\prime\} \in \overline{E}, c \in \mathcal{C}$$

$$\tilde{z}_{jj\prime mc} = 1 \Leftrightarrow (x_{jc} = 1 \wedge x_{j\prime c} = 1 \wedge x_{mc} = 1), \{j, j\prime\} \in \overline{E}, m \in \mathcal{J}^M, c \in \mathcal{C}$$

This way we can reject some combinations of decision variables earlier.

Moreover, the formulation of the problem can be modified to reflect limited resources for the maintenance. Suppose we have a budget $\Gamma$, then we can remove the maintenance cost term from the objective function and add the condition $\sum_{m \in \mathcal{J}^M} c_m \sum_{c \in \mathcal{C}} x_{mc} \leq \Gamma$. The other conditions remain the same. This is actually one example of an alternative formulation of a multi-criterion problem.

### 2.3.2 Maintenance in the planning period

While in the previous section we assumed that the maintenance could be applied only before all regular jobs, now we will focus on a more general case, in which the maintenance period can be whenever during the processing period $[0, T]$.

We still suppose that a job preemption is not allowed.

We need to define a set $\overline{E}_m \subset \overline{E}$. This set contains only those pairs of possibly overlapping jobs which would be processed after the maintenance period. Recall that a maintenance $m$ is defined by its starting time $s_m$ and prescribed finishing time $f_m^0$. Then the subset $\overline{E}_m$ for a given $m \in \mathcal{J}^M$consists of all jobs with planned start after the end of maintenance.

$$\overline{E}_m = \{(j, j\prime) \in \overline{E} : s_j \geq f_m^0\}$$

Let us remind the definition of the set

$$\overline{E} = \{(j, j\prime) \in \mathcal{J} \times \mathcal{J} : s_{j\prime} \geq f_j^0, j \in \mathcal{J} = \mathcal{J}^M \cup \mathcal{J}^I\}.$$

The set $\overline{E}$ contains all pairs of possibly overlapping edges between jobs of all types - regular one and maintenances as well. In the general case we suppose a maintenance can also have a random delay with a certain penalization. On the other hand the set $\overline{E}_m$ contains only those pairs of jobs which we can apply the improvement $\Delta_{jj\prime}$ to, if the maintenance $m$ was provided before them. Clearly, $\overline{E}_m \subset \overline{E}$.

Then the robust coloring formulation can be written similarly as in the previous case. We are already using proposed constraints with the auxiliary variable $\tilde{z}_{jj\prime mc}$ Note that this time when using the variable $z_{jj\prime}$ we can apply the improvement only to those jobs processed after the maintenance period, so we need to set $z_{jj\prime}$ as zero for all edges $\{j, j\prime\}$ which are not included in any set $\overline{E}_m$.

$$\min_{x,y,z} \sum_{m\in\mathcal{J}^M} c_m \sum_{c\in\mathcal{C}} x_{mc} + p\left(\sum_{\{j,j\prime\}\in\overline{E}} q_{jj\prime}y_{jj\prime} - \sum_{\{j,j\prime\}\in\overline{E}} \Delta_{jj\prime}z_{jj\prime}\right) \tag{2.9a}$$

$$\text{s.t.} \sum_{c\in\mathcal{C}} x_{jc} = 1, j \in \mathcal{J}^I, \tag{2.9b}$$

$$\sum_{c\in\mathcal{C}} x_{mc} \leq 1, m \in \mathcal{J}^M, \tag{2.9c}$$

$$\sum_{m\in\mathcal{J}^M} x_{mc} \leq 1, c \in \mathcal{C}, \tag{2.9d}$$

$$x_{jc} + x_{j\prime c} \leq 1, \{j, j\prime\} \in E, c \in \mathcal{C}, \tag{2.9e}$$

$$x_{jc} + x_{j\prime c} \leq 1 + y_{jj\prime}, \{j, j\prime\} \in \overline{E}, c \in \mathcal{C}, \tag{2.9f}$$

$$x_{jc} + x_{j\prime c} + x_{mc} \leq 2 + \tilde{z}_{jj\prime mc}, \{j, j\prime\} \in \overline{E}_m, m \in \mathcal{J}^M, c \in \mathcal{C}, \tag{2.9g}$$

$$x_{jc} + x_{j\prime c} \geq 2\tilde{z}_{jj\prime mc}, \{j, j\prime\} \in \overline{E}_m, m \in \mathcal{J}^M, c \in \mathcal{C}, \tag{2.9h}$$

$$x_{jc} + x_{mc} \geq 2\tilde{z}_{jj\prime mc}, \{j, j\prime\} \in \overline{E}_m, m \in \mathcal{J}^M, c \in \mathcal{C}, \tag{2.9i}$$

$$z_{jj\prime} = \sum_{m:\{j,j\prime\}\in\overline{E}_m} \sum_{c\in\mathcal{C}} \tilde{z}_{jj\prime mc}, \{j, j\prime\} \in \cup_{m\in\mathcal{J}^M}\overline{E}_m, \tag{2.9j}$$

$$z_{jj\prime} = 0, \text{if } \{j, j\prime\} \notin \cup_{m\in\mathcal{J}^M}\overline{E}_m, \tag{2.9k}$$

$$x_{jc} \in \{0, 1\}, c \in \mathcal{C}, j \in \mathcal{J} = \mathcal{J}^M \cup \mathcal{J}^I, \tag{2.9l}$$

$$y_{jj\prime} \in \{0, 1\}, \{j, j\prime\} \in \overline{E}, \tag{2.9m}$$

$$z_{jj\prime} \in \{0, 1\}, \{j, j\prime\} \in \overline{E}. \tag{2.9n}$$

The interpretation of constraints is the same as in the previous section, only this time the improvement can be applied only to pairs of jobs from $\overline{E}_m$, which is different for each maintenance $m$.

In case of limited resources for maintenance, it is possible to modify the problem by removing the maintenance cost part from the objective function and adding a constraint $\sum_{m\in\mathcal{J}^M} c_m \sum_{c\in\mathcal{C}} x_{mc} \leq \Gamma$.

If all maintenances $m \in \mathcal{J}^M$ have their starting and finishing time set before starting times of regular jobs $j \in \mathcal{J}^I$ we obtain the case of maintenance at the beginning which was described in the previous section.

### 2.3.3 Theoretical results

We have defined a general FIS problem with maintenance, now we want to show that this problem is well-defined and with the usage of maintenance periods we can obtain results which are at least as good as results from the original robust coloring problem without maintenance.

Let us start with a properly defined assumption on maintenance.

**Assumption 1.** *Suppose, that we have a FIS problem with random delays of jobs $D_j(\xi)$ with a probability distribution $P$. Furthermore suppose, that there exist maintenance jobs $m \in \mathcal{J}^M$ with the property that after the application of the maintenance to a a machine all random delays of jobs processed by this machine have a different probability distribution $\tilde{P}$. The probability distribution after maintenance satisfies the condition $\tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0) < P(D_j(\xi) > s_{j\prime} - f_j^0)$.*

**Lemma 2.** *The proposed FIS problem with maintenance* (2.9)*, where maintenance exist according to the assumption is well-defined and leads to the optimal solution of the two-stage problem* (2.6)*.*

*Proof.* Thanks to the results of Lemma 1 we already know that the expected number of overlaps can be described with the usage of penalties and soft edges in the corresponding interval graph. Now we want to discuss what the objective function will look like with maintenance.

The problem 2.5 is a case of maintenance problem when $\mathcal{J}^M = \emptyset$ and consequently $\overline{E}_m = \emptyset$. Then all variables $x_{mc}, \tilde{z}_{jj\prime mc}$ and $z_{jj\prime}$ are set to zero and constraints on the problem 2.9 define the same feasible set as constraints in the original RCP problem. The objective function is then

$$\min_{x,y,z} \sum_{m\in\mathcal{J}^M} c_m \sum_{c\in\mathcal{C}} 0 + p\left(\sum_{\{j,j\prime\}\in\overline{E}} q_{jj\prime}y_{jj\prime} - \sum_{\{j,j\prime\}\in\overline{E}} \Delta_{jj\prime}\cdot 0\right)$$

with some positive price $p$, which is equivalent to the RCP objective function

$$\min_{x,y} \sum_{\{j,j\prime\}\in\overline{E}} q_{jj\prime}y_{jj\prime}.$$

Let us assume that the set $\mathcal{J}^M$ is not empty, i.e. there exists at least one maintenance $m$. Suppose that we have a given machine $c$ and certain jobs $j_1, j_2, \dots j_k \in \mathcal{J}^I$ assigned to this machine. Assuming that jobs are ordered, for a pair $j, j\prime \in \mathcal{J}$ of subsequent jobs the objective function is equal to:

$$\sum_{m\in\mathcal{J}^M} c_m \cdot x_{mc} + p\left(q_{jj\prime}y_{jj\prime} - \Delta_{jj\prime}z_{jj\prime}\right).$$

Later we will show a detailed breakdown of possible values of variables. If we substitute $q_{jj\prime}$ and $\Delta_{jj\prime}$ with their definition we obtain the formula:

$$\begin{aligned} &c_m \cdot x_{mc} + \\ &+ p\left(P(D_j(\xi) > s_{j\prime} - f_j^0)y_{jj\prime} - (P(D_j(\xi) > s_{j\prime} - f_j^0) - \tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0))z_{jj\prime}\right). \end{aligned} \tag{2.10}$$

Thanks to the formulation of constraints related to $z_{jj\prime}$, (which serves as an indicator of edges improved by maintenance) we know that this decision variable may equal one if and only if a maintenance $m$ was applied on machine $c$ before the job $j$. Let us look at all possible cases for the given machine $c$, a maintenance $m$ and pair of jobs $j, j\prime$.

1. The maintenance was not applied on the machine $c$ and neither of jobs $j, j\prime$ or only one of them was assigned to this machine. Then $y_{jj\prime} = 0$ and $z_{jj\prime} = 0$.
   In this case the value of objective function 2.10 equals

   $$p \cdot P(D_j(\xi) > s_{j\prime} - f_j^0)y_{jj\prime} = 0,$$

   which is the same result as from the RCP problem for a pair $j, j\prime$ which was not assigned to the machine $c$.

2. The maintenance was not applied on the machine $c$ and both jobs $j, j\prime$ were assigned to this machine. Then $y_{jj\prime} = 1$ and $z_{jj\prime} = 0$.
   Now the value of objective function 2.10 equals

   $$p \cdot P(D_j(\xi) > s_{j\prime} - f_j^0) y_{jj\prime} = p \cdot P(D_j(\xi) > s_{j\prime} - f_j^0),$$

   which is equivalent to the RCP objective function (multiplication by a positive constant $p$ does not change results).

3. The maintenance was applied on the machine $c$, however neither of jobs $j, j\prime$ or only one of them was assigned to this machine. Then $y_{jj\prime} = 0$ and $z_{jj\prime} = 0$.
   In this case the maintenance does not affect the pair $j, j\prime$, because it was assigned to a different machine. The value of objective function for this pair is 0, which is the same result as from the original RCP problem for pair of jobs not consequent on a machine.

4. The maintenance was applied on the machine $c$ (thus $x_{mc} = 1$) and both jobs $j, j\prime$ were assigned to this machine. Then $y_{jj\prime} = 1$ and $z_{jj\prime} = 1$.
   Now the maintenance will affect the value of objective function 2.10 so it will be equal to:

   $$\begin{aligned}
   &c_m \cdot 1 + \\
   &p\left(P(D_j(\xi) > s_{j\prime} - f_j^0) \cdot 1 - (P(D_j(\xi) > s_{j\prime} - f_j^0) - \tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0)) \cdot 1\right) = \\
   &c_m + p\left(P(D_j(\xi) > s_{j\prime} - f_j^0) - P(D_j(\xi) > s_{j\prime} - f_j^0) + \tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0)\right) = \\
   &c_m + p \cdot \tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0).
   \end{aligned}$$

   We know that the distribution $\tilde{P}$ is better than the distribution $P$ and $p \cdot \tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0) \leq p \cdot P(D_j(\xi) > s_{j\prime} - f_j^0)$.

With this knowledge we can see that for each pair of consequent jobs (they were assigned to the same machine) the objective function has a value either $p \cdot P(D_j(\xi) > s_{j\prime} - f_j^0)$ if they are processed on a machine without maintenance, or $c_m + p \cdot \tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0)$ if they are processed on a machine with maintenance applied before, yet the price for maintenance $c_m$ is counted only once, not for every pair of jobs.

Suppose that we already have values of all decision variables $x, y, z$. Let us define sets $\overline{E}_{m-}$, which contains all pairs of consequent jobs not affected by any maintenance and $\overline{E}_{m+}$, which contains all pairs of consequent jobs affected by a maintenance $m$. Clearly $\overline{E}_{m-} \cap \overline{E}_{m+} = \emptyset$. The value of objective function is then:

$$\sum_{\{j,j\prime\} \in \overline{E}_{m-}} p \cdot P(D_j(\xi) > s_{j\prime} - f_j^0) + \sum_{\{j,j\prime\} \in \overline{E}_{m+}} p \cdot \tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0) + \sum_{\{m: x_{mc} = 1\}} c_m$$

The first term is the term from the original RCP problem, the other two are related to the use of maintenance. If the price for maintenance is too high or the improvement to small compared to the price of outsourcing $p$, the maintenance will never be applied and we will obtain the solution of the original RCP problem. Which we already know is equivalent rewrite of the two stage problem.

However, if the maintenance is applied, it increases the penalty for delay and we will achieve a lower value of the objective function. Thus the FIS maintenance problem will never find worse solution than the original RCP problem and the formulation (2.9) is a valid reformulation of the two-stage problem.

□

### 2.3.4 Example

In this section we will illustrate the FIS maintenance problem on an example. Consider that we have 2 machines available and we need to assign to them 7 jobs of which 6 are regular jobs and the remaining one is a maintenance. Suppose that we have given starting a finishing times in the table 2.1.

| job | starting time | prescribed finish time |
|---|---|---|
| $j_1$ | 0:00 | 0:15 |
| $j_2$ | 0:00 | 0:20 |
| $m$ | 0:25 | 0:30 |
| $j_3$ | 0:45 | 1:00 |
| $j_4$ | 1:05 | 1:15 |
| $j_5$ | 1:10 | 1:25 |
| $j_6$ | 1:30 | 1:40 |

Table 2.1: The list of starting and finishing times of jobs.

From this data we shall construct the graph.

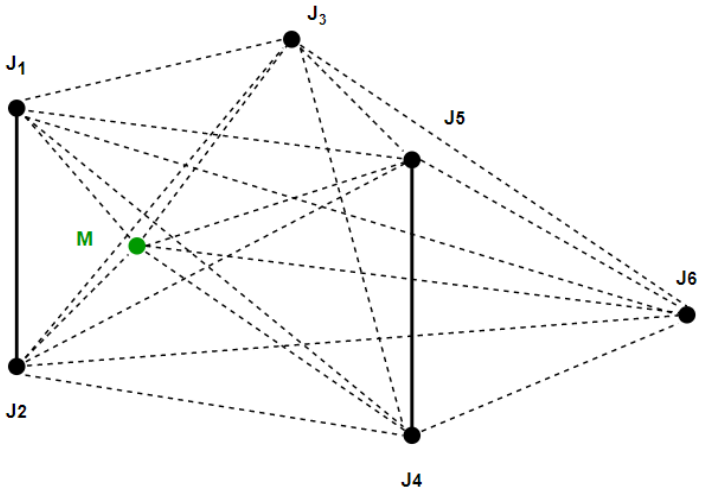


Figure 2.1: Graph of the maintenance problem.

The picture 2.1 shows the corresponding interval graph of all jobs and edges between them. Hard edges (between surely overlapping jobs) are represented by continuous lines and soft edges (between possibly overlapping jobs) are represented by dashed lines. Each soft edge between jobs $j$ and $j\prime$ is associated with a penalty $q_{jj\prime} = P(D_j(\xi) > s_{j\prime} - f_j^0)$. If a certain pair of jobs is assigned to the same

machine, then those appropriate vertices have the same colour and the variable $y_{jj\prime}$ for the soft edge between them equals one, which results in adding the penalty $q_{jj\prime}$ to the value of the objective function.

To get us a better understanding how the maintenance works, let us look at the picture 2.2.

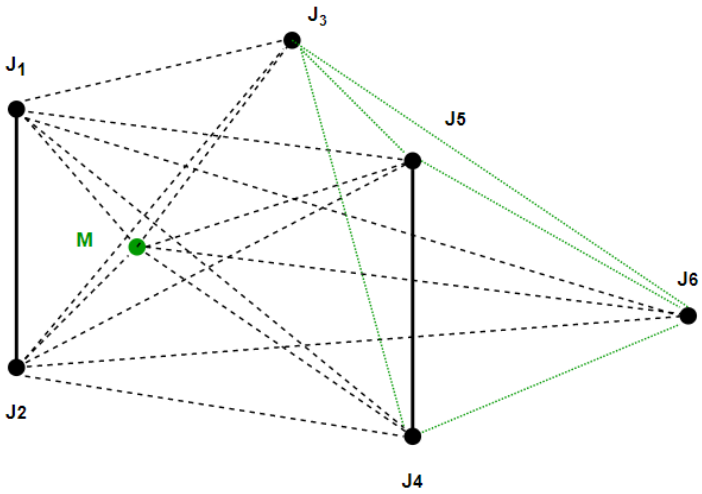


Figure 2.2: Graph of the maintenance problem, edges belonging to the set $\overline{E}_m$ have green color.

This figure depicts which edges belong to the set $\overline{E}_m$, i.e. which edges might be improved by the maintenance $m$. Note, that in compliance with the definition of the set $\overline{E}_m$, it contains only pairs of jobs processed after the maintenance is finished, not the edges between the maintenance itself and a following job. If the maintenance $m$ is assigned to the same machine as the pair $j, j\prime$, then the variable $z_{jj\prime}$ equals one and the improvement $\Delta_{jj\prime}$ might be used resulting in a lower penalty $\tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0)$ for this pair of jobs.

Let us assume that the optimal schedule is to process jobs $j_1, j_3$ and $j_5$ on the machine $c_1$ and jobs $j_2, j_4$ and $j_6$ on the machine $c_2$. If the maintenance is assigned to the first machine, it will have a positive effect on the penalty between jobs $j_3$ and $j_5$. If it will be assigned to the second machine, it will impact the penalty between jobs $j_4$ and $j_6$. On the other hand we allow the possibility of a delay of the maintenance $m$. This means that assigning a maintenance to the machine $c_1$ would add penalties for the pairs $\{m, j_3\}$ and $\{m, j_5\}$ to the objective function or penalties for pairs $\{m, j_4\}$ and $\{m, j_6\}$ in case of maintenance assigned to the machine $c_2$.

Finally, let us assume that the optimal solution is to use maintenance for the machine $c_1$. The schedule is described by first-stage variables $x_{jc}, j \in \mathcal{J}, c \in \mathcal{C}$. In our case $x_{j_1c_1}, x_{j_3c_1}, x_{j_5c_1}, x_{mc_1}, x_{j_2c_2}, x_{j_4c_2}$ and $x_{J_6c_2}$ equals one while all other variables $x_{jc}$ are equal to zero. The value of the objective function can be expressed as:

$$c_m + p(q_{j_1j_3} + q_{j_1m} + q_{j_15} + q_{mj_3} + q_{mj_5} + q_{mj_3} - \Delta_{mj_3} + q_{j_2j_4} + q_{j_2j_6} + q_{j_4j_6}),$$

all other terms equal zero. We can see that the maintenance contributes negatively to the objective function by adding terms $c_m + p(q_{mj_3} + q_{mj_5})$, on the other

hand it helps by subtracting $p\Delta_{m3}$ from the objective function.

## 2.4 Extensions of the FIS problem with maintenance

In the previous section we introduced the decision dependent problem with maintenance. We defined two special case of the problem, in the first one we were only deciding whether to apply the maintenance before all regular jobs and in the second one we worked with the possibility to assign the maintenance during the processing period. In this part of the thesis we will discuss other possible extensions of our problem.

### 2.4.1 Non-identical machines

Before this section we have always assumed that all machines are indentical. However, in practice we may often need to distinguish between machines as not each machine might be suitable to process all jobs. Our problem can be modified to reflect this case.

Assume we have an incidence matrix $A$ of dimension $|\mathcal{J}| \times |\mathcal{C}|$ with an element $a_{j,c}$ equal to one if and only if the job $j$ can be processed by the machine $c$. Then we need to add the constraint $x_{jc} \leq a_{jc}, j \in \mathcal{J}$ into our problem. This will allow to assign the job $j$ to the machine $c$ if and only if it is suitable for processing it. Note that the set of all jobs $\mathcal{J} = \mathcal{J}^M \cup \mathcal{J}^I$ contains both maintenances and regular jobs, which means we might also restrict the assignment of maintenances to machines.

This condition will have impact on narrowing down the solution space, it might also reduce the set $\overline{E}$ (or the set $\overline{E}_m$) as some possibly overlapping jobs might now be forbidden to be assigned to a certain machine. Otherwise the problem formulation remains the same.

### 2.4.2 Machine dependent delays

Another extension of this problem is when we consider the distributions of delays depend on machines. The motivation for this assumption comes from the observation that some machines might more reliable than others. Let $D_{jc}(\xi)$ be the random delay of a job $j$ processed by a machine $c$. The true finishing time of this particular job and machine is then $f_{jc}(\xi) = f_j^0 + D_{jc}(\xi)$.

Let us define a random technology matrix $T_{jkc}(\xi)$ in a similar way as before:

$$T_{jkc}(\xi) = \begin{cases} \mathbb{I}_{[D_{jc}(\xi) > s_k - f_j^0]}, \text{if } s_k \geq f_j^0, \\ 0, \text{otherwise.} \end{cases}$$

The two stage problem can be written the same way as the problem 2.4. Let us find out, if we can rewrite is as a robust coloring problem.

The objective function to minimize is the expected number of overlaps

$$\mathsf{E}_\xi \left[ \sum_{c \in \mathcal{C}} \sum_{j \in \mathcal{J}} \tilde{y}_{jc}(\xi) \right].$$

Consider a job $j$ processed on machine $c$ with $N_j$ subsequent jobs. We use the same steps as in the proof of 1 and obtain the equation:

$$\mathsf{E}_\xi\left[\tilde{y}_{jc}(\xi)\right] = \sum_{n=1}^{N_j} P(D_{jc}(\xi) > s_{k_n} - f_j^0)$$

for the given job $j$ and machine $c$.

We have derived the formulation for a given job and machine, now we want to express the expected number of overlaps for all jobs and machines.

$$\mathsf{E}_\xi\left[\sum_{c\in\mathcal{C}}\sum_{j\in\mathcal{J}}\tilde{y}_{jc}(\xi)\right] = \sum_{c\in\mathcal{C}}\sum_{j\in\mathcal{J}}\mathsf{E}_\xi\left[\tilde{y}_{jc}(\xi)\right]$$

$$= \sum_{c\in\mathcal{C}}\sum_{j\in\mathcal{J}}\sum_{k:s_k\geq s_j} x_{jc}x_{kc}P(D_{jc}(\xi) > s_k - f_j^0) \tag{2.11}$$

We obtained a quadratic (in the decision variable $x$) objective function. To simplify the formulation in a similar way as before, we need to define sets of possibly overlaping jobs dependent on machine $c$ :

$$\overline{E}_c = \{(j, j\prime) : P(D_{jc}(\xi) \geq s_{j\prime} - f_j^0) \leq 1\}, c \in \mathcal{C}$$

Additionally, we define

$$q_{jj\prime c} = P(D_{jc}(\xi) \geq s_{j\prime} - f_j^0).$$

Let us define the variable $y_{jj\prime c}$, which equals one if and only if both jobs $j$ and $j\prime$ were assigned to the machine $c$.

The whole problem can be written in a simplified formulation as:

$$\min_{x,y}\sum_{c\in\mathcal{C}}\sum_{\{j,j\prime\}\in\overline{E}_c} q_{jj\prime c}y_{jj\prime c}$$

$$\text{s.t.}\sum_{c\in\mathcal{C}} x_{jc} = 1, j \in \mathcal{J},$$

$$x_{jc} + x_{j\prime c} \leq 1, \{j, j\prime\} \in E, c \in \mathcal{C},$$

$$x_{jc} + x_{j\prime c} \leq 1 + y_{jj\prime c}, \{j, j\prime\} \in \overline{E}, c \in \mathcal{C},$$

$$x_{jc} + x_{j\prime c} \geq 2y_{jj\prime c}, \{j, j\prime\} \in \overline{E}, c \in \mathcal{C},$$

$$x_{jc} \in \{0, 1\}, c \in \mathcal{C}, j \in \mathcal{J},$$

$$y_{jj\prime c} \in \{0, 1\}, \{j, j\prime\} \in \overline{E}, c \in \mathcal{C}. \tag{2.12}$$

However, because we cannot remove the dependence on $c$ (i.e. the sum over $c$) it is not a case of a robust coloring problem but a generalization of it.

### 2.4.3 Maintenance with limited duration

In the chapter 3 we have assumed that the improvement of the maintenance lasts till the end of the processing period. If we are schedulling jobs with long processing time, it might be necessary to assume that the the positive effect brought by maintenances lasts only for a certain time period. Our problem can be modified to reflect this situation in the definition of subsets $\overline{E}_m$. Let us define $T_m$ as the time when the maintenance $m$ stops affecting delays of jobs. Only those pairs of jobs schedulled after the maintenance was processed and before its' effect ends have the improved probability of delay $\tilde{P}$. This will be reflected in the new defition of sets:

$$\overline{E}_m = \{(j, j\prime) \in \overline{E} : s_j \geq f_m^0 \wedge s_{j\prime} \leq T_m\}.$$

Otherwise the problem formulation remains the same.

# 3. Numerical study

In this chapter we present results of a numeric study solving the FIS maintenance problem introduced in chapter 3. The source code is a part of the electronic attachments uploaded with this thesis.

We used software *GAMS* version 24.0.2 and the study was conducted on PC with the operation system *Microsoft Windows 10 Pro*, processor *AMD Ryzen 7 2700X Eight-Core Processor* 3,7 Ghz and RAM 64 GB. The problem was solved using the solver *Cplex* version 12.5.0.0., which uses a default branch and cut algorithm. We did not change any setting apart from prolonging the maximum allowed running time of the solver to one hour. We assume that investing a larger amount of time to solving the problem would not be efficient, however it is possible to set a longer time limit.

The input information about jobs and their processing times were simulated in the program *R,* version 4.0.1. To generate solvable tasks we used the method described in thesis [Jeliga, 2019]. We employ a number of auxiliary machines $\tilde{M}$ and generate a certain number $\tilde{J}$ of jobs for each one of them. The total number of jobs is then equal to $J = \tilde{M} \cdot \tilde{J}$ and the number of machines available is chosen as $M > \tilde{M}$, in our study we chose $M = \tilde{M} + 1$. This way we have guaranteed that there exists more than one feasible solution. Job processing times $f_j^0 - s_j$ were simulated using the exponential distribution with parameter $\lambda_1$, the breaks between jobs $s_{j\prime} - f_j^0$ were simulated from exponential distribution with parameter $\lambda_2$. For the random delay we suppose it has the following distribution function:

$$F_j(x) = P_j(D_j(\xi) \leq x) = p_j + (1 - p_j)(1 - e^{-\lambda_j x}), x \geq 0,$$

where $p_j$ is the probability of finishing the job $j$ in time. With this formulation we can exactly express penalties $q_{jj\prime}$.

$$q_{jj\prime} = P(D_j(\xi) > s_{j\prime} - f_j^0) = 1 - P(D_j(\xi) \leq s_{j\prime} - f_j^0)$$

$$= 1 - p_j + (1 - p_j)(-1 + e^{-\lambda_j(s_{j\prime} - f_j^0)}) = (1 - p_j)(e^{-\lambda_j(s_{j\prime} - f_j^0)})$$

Without the loss of generality, we decided to set the probability of a maintenance having a delay as 0, i.e. set the parameter $p_m = 1, m \in \mathcal{J}^M$. The theoretical formulation allows that each job $j$ has a different distribution of delays $D_j$, so this will not have any negative impact on the calculation.

The improved distribution after the maintenance application $\tilde{P}$ was set either proportionally as

$$\tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0) = 0.5 \cdot P(D_j(\xi) > s_{j\prime} - f_j^0)$$

or with the formula

$$\tilde{P}_j(D_j(\xi) \leq x) = \tilde{p}_j + (1 - \tilde{p}_j)(1 - e^{-\tilde{\lambda}_j x}), x \geq 0,$$

where parameters $\tilde{p}_j$ and $\tilde{\lambda}_j$ are different values then $p_j$ and $\lambda_j$, which determine the distribution $P$. Additionally, to follow the rule that $P(D_j(\xi) > s_{j\prime} - f_j^0) > \tilde{P}(D_j(\xi) > s_{j\prime} - f_j^0)$ we will suppose that $\tilde{p}_j > p_j$ and $\lambda_j = \tilde{\lambda}_j$.

## 3.1 The dimension of the problem

In our FIS maintenance problem formulation 2.9 we introduced a rather wide set of constraints. This might become a technical limitation, because commercial software usually has a restriction on number of constraints and variables (for example $GAMS$ demo license can solve only problems up to 2000 constraints and 2000 variables). Also having more constraints might require higher computational capacity.

Our problem works with decision variables $x_{jc}$, their number equals $|\mathcal{J}| \cdot |\mathcal{C}|$ and variables $y_{jj\prime}$ and $z_{jj\prime}$, for both of them their number equals $|\overline{E}|$. We also declared auxiliary variables $z_{jj\prime mc}$, which was defined for sets $\overline{E}_m$, therefore their number equals to $\sum_{m\in\mathcal{J}^M} |\overline{E}_m| \cdot |\mathcal{C}|$. In total we have

$$|\mathcal{J}| \cdot |\mathcal{C}| + 2|\overline{E}| + \sum_{m\in\mathcal{J}^M} |\overline{E}_m| \cdot |\mathcal{C}|$$

variables in the problem.

Let us count the total number of constraints equations directly from the formula 2.9:

$$|\mathcal{J}^I| + |\mathcal{J}^M| + |\mathcal{C}| + |E| \cdot |\mathcal{C}| + |\overline{E}| \cdot |\mathcal{C}| +$$
$$+3 \sum_{m\in\mathcal{J}^M} |\overline{E}_m| \cdot |\mathcal{C}| + \underbrace{|\cup_{m\in\mathcal{J}^M} \overline{E}_m| + |\overline{E} \setminus \cup_{m\in\mathcal{J}^M} \overline{E}_m|}_{|\overline{E}_m|}$$

Later we will use a few different settings for the numeric study with ten different generated scenarios for each setting. The size of sets of soft edges $\overline{E}$ and hard edges $E$ as well as soft edges to improve $\overline{E}_m$ always depend on the specific input (there might be extreme cases when for example jobs would be starting and finishing so that there are no hard edges), therefore we will provide average number of equations in each setting.

## 3.2 Conditions on soft edges

During the numeric study we encountered a problematic behaviour of variables $y_{jj\prime}$, which serves as an indicator of active soft edges. Conditions on $y_{jj\prime}$ are given by the relation

$$x_{jc} + x_{j\prime c} \leq 1 + y_{jj\prime}, \{j, j\prime\} \in \overline{E}, c \in \mathcal{C},$$

which allows $y_{jj\prime}$ to equal one even when the variables $x_{jc}$ or $x_{j\prime c}$ equal zero for some of the jobs $j, j\prime$ and a machine $c$. However, the objective function is trying to minimize terms $q_{jj\prime}y_{jj\prime}$ and therefore set $y_{jj\prime}$ as zero, whenever possible. When we decided to set that maintenances have no delays, it means that penalties related to edges $m, j\prime$ are set to zero and variables $y_{mj\prime}$ might equal one even in cases when it does not make sense. This case happened during the numeric study, all variables $y_{mj\prime}, m \in \mathcal{J}^M, \{j, j\prime\} \in \overline{E}$ obtained the value 1, irrespective of the actual assignment of maintenances and jobs on machines.

We are interested mainly in the value of the objective function and the assignment of jobs given by variables $x_{jc}$ and this behaviour of $y_{jj\prime}$ has no impact on them. Only the interpretation of $y_{jj\prime}$ (the value one represents that there are soft edges between following jobs) may be misleading. We decided leave it in the numeric study as it is, however, there are options how to fix this problem.

- Manually adjust the values of $y_{jj\prime}$ by adding this new constraint:

$$y_{jj\prime} = 0, j \in \mathcal{J}^M, \{j, j\prime\} \in \overline{E}.$$

  The advantage of this approach is, that we add only a limited number of equations and do not make our problem too much larger. The disadvantage is, that it is a manual adjustment and not a system solution. Also, it is necessary to define a new subset of $\overline{E}$ for this equation.

- Another solution is to define a new auxiliary variable $y_{jj\prime c}$ and use it to formulate new constraints instead of:

$$x_{jc} + x_{j\prime c} \leq 1 + y_{jj\prime}, \{j, j\prime\} \in \overline{E}, c \in \mathcal{C}.$$

  Those new constraints would be:

$$x_{jc} + x_{j\prime c} \leq 1 + y_{jj\prime c}, \{j, j\prime\} \in \overline{E}, c \in \mathcal{C},$$

$$x_{jc} + x_{j\prime c} \geq 2y_{jj\prime c}, \{j, j\prime\} \in \overline{E}, c \in \mathcal{C},$$

$$y_{jj\prime} = \sum_{c \in \mathcal{C}} y_{jj\prime c}, \{j, j\prime\} \in \overline{E}$$

  The advantage of this approach is that $y_{jj\prime}$ will be equal to one if and only if both jobs $j, j\prime$ were assigned to the same machine, regardless of their corresponding penalty. The obvious disadvantage is that we will add new variables and more equations to our problem.

## 3.3 Schedule with reliable jobs

Our first intention was to compare the maintenance problem with the robust coloring problem from the article [Branda et al., 2015]. Therefore we used the same values of parameters:

- $\lambda_1 = 0.2$ for the simulation of job lengths,
- $\lambda_2 = 0.05$ for the simulation of lengths of breaks between jobs and
- $\lambda_j = 0.2$ and $p_j = 0.9$ for the distribution of delays of regular jobs.

This setting assumes that jobs are very reliable and have very low probability of delay, typically the first non-zero number is at the third decimal spot. With this assumption the optimal solution will be using maintenance only when it is necessary. Moreover, we decided to set the improvement brought by maintenance as half of the penalty, specifically: $\Delta_{jj\prime} = q_{jj\prime}$, if the improvement was bigger, it could have been used more often.

Because the price for outsourcing and price for maintenance were not a part of the original article, we decided to experiment with it. It turned out that the maintenance was used only if the ratio between the outsourcing price and maintenance price was very high, such as 20000 : 2. This is in compliance with our expectation, for a very reliable jobs we do not need to take any extraordinary measures. However, in this study we want to show the effect of maintenance so we will use a parameter settings allowing us to assign maintenance more often.

Let us look at an example of a problem with 3 machines and 10 jobs of which jobs 3 and 5 are maintenance. The input data were simulated ten times and the table 3.1 shows how many times at least one maintenance was used in the solution with different values of the parameter $p$, while the price for maintenance remains fixed on values 2 and 3.

| **price of outsourcing** | 20 | 200 | 2000 | 20000 | 200000 |
|---|---|---|---|---|---|
| **number of cases maintenance used** | 0 | 0 | 2 | 5 | 9 |

Table 3.1: Usage of maintenance related to the price of outsourcing.

Based on this experiment we decide to use the setting of parameters when the ratio between the price for outsourcing and the price for maintenance is approximately ten thousand. We will test three different settings of parameters and generate 10 scenarios for each one. We manually choose and fix certain indexes of jobs to serve as maintenances.

- 3 machines and 10 jobs, simulated on $\tilde{J} = 5$ and $\tilde{M} = 2$. Jobs 3 and 5 are maintenances with prices 2 and 3, the price for outsourcing is 20000.
- 5 machines and 20 jobs, simulated on $\tilde{J} = 5$ and $\tilde{M} = 4$. Jobs $3, 5, 10$ and 12 are maintenances with prices $5, 3, 2$ and 3, the price for outsourcing is 20000.
- 5 machines and 32 jobs, simulated on $\tilde{J} = 8$ and $\tilde{M} = 4$. Jobs $2, 7, 15$ and 25 are maintenances with prices $10, 6, 3$ and 2 , the price for outsourcing is 20000.

All the above mentioned cases will be compared with the case when we cannot use the maintenance, i.e. $x_{mc} = 0, \forall m \in \mathcal{J}^M$.

The results are shown in tables below. The left part of the table refers to the problem with maintenance, the right is for comparison with the problem without maintenance. The column **Obj.f.** provides the value of the objective function, namely of the expression

$$\sum_{m \in \mathcal{J}^M} c_m \sum_{c \in \mathcal{C}} x_{mc} + p \left( \sum_{\{j,j\prime\} \in \overline{E}} q_{jj\prime} y_{jj\prime} - \sum_{\{j,j\prime\} \in \overline{E}} \Delta_{jj\prime} z_{jj\prime} \right)$$

or of the expression

$$p \cdot \sum_{\{j,j\prime\} \in \overline{E}} q_{jj\prime} y_{jj\prime}$$

in case of not using the maintenance. The column **Time** provides the information (in seconds) how much time the solver needed to obtain the solution and the column **Rel. gap** provides the ratio between absolute gap (the difference between the best possible and the final MIP solution) and the final MIP solution.

We are aware that that finding the exact solution is problematic even for relatively small instances of the problem. To be able to use this problem in practice we suggest using some heuristic, however they might not lead to the optimal solution.

| Scen. | With maintenance | | | Without maintenance | | |
|---|---|---|---|---|---|---|
| | Time | Obj.f. | Rel. gap | Time | Obj.f. | Rel. gap |
| **1** | 0.19 | 37.57 | 0.00 | 0.05 | 45.80 | 0.00 |
| **2** | 0.17 | 27.82 | 0.00 | 0.05 | 40.75 | 0.03 |
| **3** | 0.14 | 6.62 | 0.00 | 0.05 | 6.62 | 0.00 |
| **4** | 0.25 | 7.02 | 0.00 | 0.05 | 7.02 | 0.00 |
| **5** | 0.36 | 207.43 | 0.00 | 0.05 | 394.25 | 0.00 |
| **6** | 0.22 | 3.71 | 0.00 | 0.05 | 3.71 | 0.00 |
| **7** | 0.14 | 0.14 | 0.09 | 0.05 | 0.14 | 0.00 |
| **8** | 0.19 | 15.58 | 0.00 | 0.03 | 15.58 | 0.00 |
| **9** | 0.20 | 65.57 | 0.00 | 0.05 | 125.68 | 0.00 |
| **10** | 0.24 | 43.29 | 0.00 | 0.03 | 56.96 | 0.00 |

Table 3.2: Results of the numeric study for 10 jobs and 3 machines.

The table 3.2 summarizes the results for the first problem with 10 jobs and 3 machines. We can see that this small instance of the problem the solver achieved results relatively fast (around half a second), however without the maintenance the solution was obtained faster. This is expected due to large number of equations (in average 447 equations for a scenario). We also notice that the maintenance was used in scenarios $1, 2, 5, 9$ and 10 and we obtained lower values than in the problem without maintenance. This is especially visible in scenarios 5 and 9, when the results reached with maintenances are almost half of the value without maintenance. We can also see that most of the values of relative gaps equal zero, i.e. the algorithm almost always found the best possible solution.

| Scen. | With maintenance | | | Without maintenance | | |
|---|---|---|---|---|---|---|
| | Time | Obj.f. | Rel. gap | Time | Obj.f. | Rel. gap |
| **1** | 605.64 | 89.85 | 0.10 | 0.47 | 134.61 | 0.09 |
| **2** | 102.70 | 46.82 | 0.10 | 0.20 | 46.82 | 0.10 |
| **3** | 793.88 | 78.19 | 0.10 | 0.20 | 81.67 | 0.10 |
| **4** | 1870.77 | 119.32 | 0.10 | 0.72 | 156.37 | 0.10 |
| **5** | 1682.28 | 551.02 | 0.10 | 0.24 | 759.30 | 0.10 |
| **6** | 498.56 | 97.59 | 0.10 | 0.11 | 98.70 | 0.06 |
| **7** | 96.77 | 5.25 | 0.10 | 0.16 | 5.25 | 0.09 |
| **8** | 1212.38 | 138.64 | 0.10 | 0.27 | 157.16 | 0.10 |
| **9** | 173.94 | 4.28 | 0.10 | 0.38 | 4.28 | 0.10 |
| **10** | 7.63 | 42.83 | 0.10 | 0.09 | 52.77 | 0.01 |

Table 3.3: Results of the numeric study for 20 jobs and 5 machines.

As we can see in the table 3.3 for the problem with 20 jobs, the computing times increased dramatically. In average, there were 4420 equations for a scenario so it becomes more computationally demanding. Maintenance was used more often this time, in 7 out of 10 cases we were able to reach a better value of the objective function then without it. Relative gap is 10% for the majority of scenarios, which shows how difficult the task is. If we wanted lower values such as 1% the computation would demand more resources

| Scen. | With maintenance | | | Without maintenance | | |
|---|---|---|---|---|---|---|
| | Time | Obj.f. | Rel. gap | Time | Obj.f. | Rel. gap |
| **1** | 3601.47 | 470.93 | 10.80 | 7.47 | 572.96 | 0.10 |
| **2** | 3601.20 | 40.63 | 43.88 | 2.08 | 41.76 | 0.10 |
| **3** | 3601.16 | 332.06 | 20.08 | 3.75 | 372.80 | 0.10 |
| **4** | 3600.24 | 504.56 | 8.16 | 1.69 | 704.39 | 0.08 |
| **5** | 3601.81 | 1216.12 | 9.44 | 24.97 | 1460.15 | 0.10 |
| **6** | 3600.24 | 1398.96 | 2.14 | 1.89 | 1580.29 | 0.07 |
| **7** | 3600.28 | 211.25 | 7.50 | 3.69 | 276.63 | 0.10 |
| **8** | 3601.23 | 302.31 | 25.69 | 1.69 | 307.52 | 0.10 |
| **9** | 3602.94 | 251.21 | 16.69 | 5.19 | 327.54 | 0.10 |
| **10** | 3600.25 | 167.80 | 16.61 | 1.91 | 169.40 | 0.10 |

Table 3.4: Results of the numeric study for 32 jobs and 5 machines.

Finally, in the table 3.4 we can see results for our largest problems with 32 jobs. The computational requirements are definitely more problematic here, because in all cases the solver exceeded the time limit. There was in average 13245 equations for a scenario and 4892 defined variables. On the other hand we see that in all scenarios there was a better solution reached and because in some cases the relative gap is really high, we probably could have reached better results if we had prolonged the time limit.

## 3.4 Schedule with delaying jobs

In contrast to the previous section we will now focus on the case when the ratio between price for outsourcing and price for maintenances is not so high, however, regular jobs are very likely to have a delay and applying the maintenance will bring a dramatic improvement in their delay probabilities. Specifically, the price for outsourcing is set as 200, while prices for maintenances remain the same as before. We assume, that the probability distribution of random delays is

$$P_j(D_j(\xi) \leq x) = p_j + (1 - p_j)(1 - e^{-\lambda_j x}), x \geq 0, j \in \mathcal{J}^I$$

where $p_j = 0.05$, i.e. there is only a low chance of a job having no delay. The improved distribution $\tilde{P}$ is defined as

$$\tilde{P}_j(D_j(\xi) \leq x) = \tilde{p}_j + (1 - \tilde{p}_j)(1 - e^{-\lambda_j x}), x \geq 0, j \in \mathcal{J}^I$$

where $\tilde{p}_j = 0.95$, which means that after the maintenance application there is a great chance of having no delay. We still suppose that maintenances do not have delays, so that $P_m(D_m(\xi) \leq x) = 0, x \geq 0, m \in \mathcal{J}^M$

Again, we will look closely at three different settings, for each one of them we generate 10 scenarios.

- 3 machines and 10 jobs, simulated on $\tilde{J} = 5$ and $\tilde{M} = 2$. Jobs 3 and 5 are maintenances with prices 2 and 3, the price for outsourcing is 200.
- 5 machines and 20 jobs, simulated on $\tilde{J} = 5$ and $\tilde{M} = 4$. Jobs $3, 5, 10$ and 12 are maintenances with prices $5, 3, 2$ and 3, the price for outsourcing is 200.

- 5 machines and 32 jobs, simulated on $\tilde{J} = 8$ and $\tilde{M} = 4$. Jobs $2, 7, 15$ and $25$ are maintenances with prices $10, 6, 3$ and $2$ , the price for outsourcing is $200$.

Results are summarized in tables below with the same structure as in the previous section. It is again obvious that the finding the exact solution becomes problematic even for relatively small instances.

| **Scen.** | **With maintenance** | | | **Without maintenance** | | |
|---|---|---|---|---|---|---|
| | **Time** | **Obj.f.** | **Rel. gap** | **Time** | **Obj.f.** | **Rel. gap** |
| **1** | 0.19 | 4.24 | 0.00 | 0.05 | 4.35 | 0.00 |
| **2** | 0.33 | 2.35 | 0.00 | 0.05 | 3.87 | 0.00 |
| **3** | 0.16 | 0.63 | 0.08 | 0.05 | 0.63 | 0.00 |
| **4** | 0.27 | 0.67 | 0.00 | 0.06 | 0.67 | 0.00 |
| **5** | 0.52 | 7.30 | 0.00 | 0.06 | 37.45 | 0.00 |
| **6** | 0.31 | 0.35 | 0.00 | 0.05 | 0.35 | 0.00 |
| **7** | 0.17 | 0.01 | 0.00 | 0.05 | 0.01 | 0.00 |
| **8** | 0.28 | 1.48 | 0.00 | 0.05 | 1.48 | 0.00 |
| **9** | 0.17 | 2.64 | 0.00 | 0.05 | 11.94 | 0.00 |
| **10** | 0.34 | 4.76 | 0.00 | 0.05 | 5.41 | 0.00 |

Table 3.5: Results of the numeric study for 10 jobs and 3 machines.

Let us look at the smallest instance of the problem. The table 3.5 shows, that similarly as in the previous section, maintenance was used in half of the cases (scenarios $1, 3, 5, 9, 10$) to obtain better values of the objective function. We can notice that running times are slightly higher than in the table 3.2. Especially interesting is the scenario 5, where we achieved a significantly better result, however it also took the longest time to process. The relative gap is in all cases except 3 equal to zero, so the solution used is also the best possible.

| **Scen.** | **With maintenance** | | | **Without maintenance** | | |
|---|---|---|---|---|---|---|
| | **Time** | **Obj.f.** | **Rel. gap** | **Time** | **Obj.f.** | **Rel. gap** |
| **1** | 3600.11 | 6.14 | 5.81 | 0.45 | 12.79 | 0.10 |
| **2** | 813.58 | 4.45 | 0.10 | 0.31 | 4.45 | 0.06 |
| **3** | 1533.55 | 7.76 | 0.10 | 0.14 | 7.76 | 0.10 |
| **4** | 3600.14 | 10.48 | 16.57 | 0.61 | 14.86 | 0.09 |
| **5** | 3600.20 | 28.98 | 11.52 | 0.20 | 72.13 | 0.10 |
| **6** | 3600.20 | 7.11 | 4.38 | 0.14 | 9.27 | 0.07 |
| **7** | 480.98 | 0.50 | 0.10 | 0.11 | 0.50 | 0.08 |
| **8** | 3600.53 | 11.68 | 6.97 | 0.25 | 14.93 | 0.10 |
| **9** | 785.75 | 0.41 | 0.10 | 0.17 | 0.41 | 0.09 |
| **10** | 460.28 | 4.40 | 0.10 | 0.08 | 5.01 | 0.05 |

Table 3.6: Results of the numeric study for 20 jobs and 5 machines.

The table 3.6 summarizes results for the instance with 20 jobs. Compared to the previous section, we can notice that we have already reached the time limit for half of scenarios. Although we have the same average number of equations

and variables for a scenario, the solver needs more time and ends with a higher relative gap. For the scenarios $2, 3, 7$ and 9 we ended with the same result as without maintenance, in this case we also did not reach the time limit and the relative gap is small. Nevertheless, in other scenarios we managed to find a lower value of the objective function. The high values of relative gap indicate that it would be better to run the solution for a longer time.

| **Scen.** | **With maintenance** | | | **Without maintenance** | | |
|---|---|---|---|---|---|---|
| | **Time** | **Obj.f.** | **Rel. gap** | **Time** | **Obj.f.** | **Rel. gap** |
| **1** | 3600.22 | 25.93 | 38.53 | 12.70 | 54.43 | 0.10 |
| **2** | 3600.20 | 3.97 | 93.94 | 1.09 | 3.97 | 0.10 |
| **3** | 3601.64 | 24.42 | 70.75 | 2.58 | 35.42 | 0.10 |
| **4** | 3601.83 | 24.88 | 32.41 | 2.47 | 66.76 | 0.09 |
| **5** | 3601.05 | 48.15 | 43.12 | 16.94 | 138.71 | 0.10 |
| **6** | 3601.16 | 74.79 | 7.87 | 3.86 | 150.20 | 0.10 |
| **7** | 3600.95 | 18.35 | 34.84 | 2.33 | 26.28 | 0.10 |
| **8** | 3600.89 | 19.86 | 49.74 | 1.86 | 29.21 | 0.10 |
| **9** | 3602.36 | 20.92 | 89.66 | 6.58 | 31.05 | 0.10 |
| **10** | 3601.25 | 16.10 | 44.59 | 2.30 | 16.07 | 0.05 |

Table 3.7: Results of the numeric study for 32 jobs and 5 machines.

The largest instance with 30 is summarized in the table 3.7. We see that the time limit was used in all cases and relative gap is really high, so it would be better to give the solver more time or computational capacity. In some cases, such as the scenario 5, we managed to find a significantly lower value of the objective function, on the other hand, the solution found in the scenario 10 is even slightly worse than the original problem. This is probably due to errors in rounding.

The overall statistics for all cases are provided in the table 3.8. Columns **case** and **jobs** serve to identify the appropriate part of simulation, the case of reliable jobs and disproportion in prices of outsourcing and maintenance is called **ratio** and the case of delaying jobs is called **prob**.

| **case** | **jobs** | **avg. time** | **avg eqs** | **avg vars** | **avg gap** |
|---|---|---|---|---|---|
| **ratio** | 10 | 0.21 | 447.00 | 238.90 | 0.01 |
| **ratio** | 20 | 704.45 | 4219.50 | 1619.80 | 0.10 |
| **ratio** | 32 | 3601.08 | 13245.50 | 4892.10 | 16.10 |
| **prob** | 10 | 0.27 | 447.00 | 238.90 | 0.01 |
| **prob** | 20 | 2207.53 | 4219.50 | 1619.80 | 4.58 |
| **prob** | 32 | 3601.15 | 13245.50 | 4892.10 | 50.55 |

Table 3.8: Overall results for tested instances of the FIS maintenance problem

We can see that although working with exactly the same average number of equations and variables for an instance of the problem, the problem with delaying jobs took longer time to solve than the problem with prices disproportion. This is especially visible in the part with 20 jobs, while the prices disproportion modification managed to find a solution in a reasonable time, in the delaying jobs

version already half of the scenarios finished at exceeding the time limit and with a higher relative gap. We conclude that our problem is suitable only for instances with smaller dimensions.

## 3.5 Number of machines

During the setting phase of the numeric study we were experimenting with number of machines available for jobs. We used a relatively strict setup when the number of available machines is only one above the number of machines used for generating the instance. This is because with lower number of machines there will be more possibilities of job overlaps and therefore higher penalties, so the maintenance is more useful.

To demonstrate this occurrence, we provide the following table comparing the usage of maintenance in the case of 32 jobs when 4 of them are maintenances and all data were generated using 4 auxiliary machines (each with 8 jobs). The table 3.9 shows for many cases out of ten scenarios the maintenance was used, specifically that at least one maintenance job was assigned to a machine. We can see that the case with 5 jobs, which was described in the section above, maintenances were used a lot, namely in 9 out of 10 or even 10 out of 10 cases. When we increase the number of machines, maintenance is used less. It seems that the maintenance modification might be useful especially when we do not have many machines and therefore there is a higher risk of jobs overlap.

The same phenomenon of maintenances being less used if there are more available machines is valid also for smaller instances with 10 or 20 jobs.

| **type** | **number of jobs** | **machines** | **scenarios of which m. used** | |
|---|---|---|---|---|
| **ratio** | 32 | 5 | 10 | 9 |
| **ratio** | 32 | 6 | 10 | 7 |
| **ratio** | 32 | 8 | 10 | 7 |
| **prob** | 32 | 5 | 10 | 10 |
| **prob** | 32 | 6 | 10 | 9 |
| **prob** | 32 | 8 | 10 | 6 |

Table 3.9: Usage of maintenance related to the number of machines.

# Conclusion

In this thesis we discussed fixed interval scheduling problems under decision dependent randomness. In the first chapter we defines fixed interval scheduling problems and explained the decision dependent randomness. We mentioned the recently introduced taxonomy which divides decision dependent problems into three classes - decision dependent probabilities, decision dependent information schema and mixed problem. Our main focus in this thesis was a problem from the decision dependent probabilities class.

The next chapter introduces the fixed interval scheduling problem with random delays of jobs. We formulated the problem as minimizing the number of expected overlaps between jobs and showed the connection to the robust graph coloring problem. Then we defined a new element - the maintenance - a job which after being processed by a machine can improve the probability distribution of delays of all future jobs assigned to this machine. Firstly, we explored the case when maintenance jobs can be assigned only at the beginning of the planning period and therefore they impact all the following jobs. Then we investigated a more general case, when maintenance jobs can be assigned to machines during the whole planning period. We used the robust coloring formulation and modified it to reflect the impact of maintenances, with cost for maintenance at one side and possibly improved probabilities of delays on the other side. We also had to add new decision variable $z_{jj'}$ indicating the improvement brought by the maintenance and new sets of constraints for it. Finally, we illustrated the problem on a small example of 6 jobs, 2 machines and 1 maintenance.

The FIS maintenance problem can be modified to reflect different setting or assumptions. Chapter also 3 discusses certain extensions of our problem, such as the delays' distribution dependent on machines, non identical machines or maintenances with a limited time duration.

The final chapter is dedicated to the numeric study. We simulated input data in the software *R* and solved our problem with the *GAMS* software, solver *Cplex*. We discussed the setting of parameters and found out that maintenance is active in problems where there is a high ratio between the price for outsourcing and price for maintenance or in problems, where jobs are very likely to delay. We compared results of the FIS maintenance problem with the original RCP formulation. The FIS maintenance problem brings better results in the objective function, however due to having more constraints it requires more computational capacity. For the largest instances it even demands too much time that it is in practise not suitable to be used. We suggest that to use this problem in practice some heuristics should be used, otherwise it works well only for relatively small cases with 10 or 20 jobs.

# Bibliography

Martin Branda. Distributionally robust fixed interval scheduling on parallel identical machines under uncertain finishing times. *Computers & Operations Research*, 98, 06 2018. doi: 10.1016/j.cor.2018.05.025.

Martin Branda, Jan Novotný, and Asmund Olstad. Fixed interval scheduling under uncertainty – a tabu search algorithm for an extended robust coloring formulation. *Computers & Industrial Engineering*, 93, 12 2015. doi: 10.1016/j.cie.2015.12.021.

Jitka Dupačová. Optimization under exogenous and endogenous uncertainty. In *Proceedings of MME*, 09 2006. doi: 10.13140/2.1.2682.2089.

Lars Hellemo, Paul Barton, and Asgeir Tomasgard. Decision-dependent probabilities in stochastic programs with recourse. *Computational Management Science*, 08 2018. doi: 10.1007/s10287-018-0330-0.

Jan Jeliga. Minimax v úlohách rozvrhování za nejistoty. Masters thesis, Charles University, Faculty of Maths and Physics, Department of Probability and Mathematical Statistics, 2019.

Francesca Rossi, Peter van Beek, and Toby Walsh. *Handbook of Constraint Programming*. Elsevier Science Inc., USA, 2006. ISBN 9780080463803.

Pravin Varaiya and Roger J.-B. Wets. Stochastic dynamic optimization approaches and computation. Iiasa working paper, IIASA, Laxenburg, Austria, September 1988. URL `http://pure.iiasa.ac.at/id/eprint/3118/`.

Javier Yáñez and Javier Ramirez. The robust coloring problem. *European Journal of Operational Research*, 148:546–558, 02 2003. doi: 10.1016/S0377-2217(02)00362-4.