A recursive algorithm for a pipeline maintenance scheduling problem

Assia Boumahdaf, Michel Broniatowski

A recursive algorithm for a pipeline maintenance scheduling problem

Assia Boumahdaf* Michel Broniatowksi†
Laboratoire de Statistique Théorique et Appliquée,
Université Pierre et Marie Curie

Abstract

This paper deals with the problem of preventive maintenance (PM) scheduling of pipelines subject to external corrosion defects. The preventive maintenance strategy involves an inspection step at some epoch, together with a repair schedule. This paper proposes to determine the repair schedule as well as an inspection time minimizing the maintenance cost. This problem is formulated as a binary integer non-linear programming model and we approach it under a decision support framework. We derive a polynomial-time algorithm that computes the optimum PM schedule and suggests different PM strategies in order to assist practitioners in making decision.

1 Introduction

Gas pipelines are facilities intended for the transport of natural gas at high pressure. Pipelines carry natural gas from the extraction area to export area and are buried under the ground in inhabited zones. A major threat for their integrity is metal-loss corrosion. To maintain safe pipeline condition, preventive maintenance (PM) is performed. Poor pipeline management can cause leaks and leads to human and environmental damage, as well as monetary loss.

As states Zhou in [14] a pipeline management program consists firstly in detecting the corrosion defects with appropriate equipment, secondly in evaluating the probability of failure based on the primary inspection results, and lastly in repairing the defects if necessary. Defects which do not call into question system integrity are not immediately repaired, and will be considered for PM, which consists of identifying the next inspection time, together with a repair schedule. This paper deals with the PM problem in gas pipeline from an economic point of view. We aim to investigate the optimal PM schedule which minimizes the operational cost. Large scale maintenance activities have significant cost. There are not only the costs related to inspection and repairs, but also the cost due to production losses during maintenance. When maintenance activities are conducted, gas flow in pipelines must be interrupted for security measures, generating a significant out-of-service cost. Hence, the operational cost estimate includes the cost due to inspection, repairs, and the cost due to out-of-service of gas pipelines.

Several models and methodologies establishing optimal PM schedules can be found in the literature. A number of papers have been published using a reliability approach. In the corroding gas pipeline context, we refer the reader to [12] for a recent survey about this subject. In their paper, they address the problem of predicting the reliability of pipelines with imperfect repairs in order to assist pipeline operators in making the most appropriate PM decision. Hong [5] uses also a reliability analysis to estimate the probability of failure, together with optimal inspection and maintenance schedules, based on minimization of the total cost including inspection, repairs, and

*e-mail: assia.boumahdaf@gmail.com
†e-mail: michel.broniatowski@upmc.fr

the cost of failure. In a more general context, Kallen [7] determines the optimal inspection and replacement policy which minimize the expected average costs using an adaptive Bayesian decision model.

To the best of our knowledge, there have been no previous economic and deterministic studies on PM of gas pipelines. However, this issue arises in a wide variety of areas, one of which is power plants. This is an important issue because a failure in a power station may cause an overall breakdown, and significant customer dissatisfaction. Canto [10] considers the problem of selecting the period for which the facilities should be taken offline for preventive maintenance. He models this problem with a binary integer linear program, and solved it using optimization software. The same author [1] solves the problem using Benders decomposition technique. Megow [9] derives a model for the problem of turnaround scheduling. They propose an optimization algorithm within a decision support framework that computes the schedule minimizing the resource cost.

Preventive maintenance problems also arise in the medical field. Vlah Jerić and Figueira [13] consider the issue of scheduling medical treatments for patients. They formulate the problem as a binary integer programming model, and solve it using a meta-heuristic algorithm. [11] propose a decision support system for resource scheduling. Chern *et al.* [2] consider health examination scheduling. They model this problem using a binary integer programming model, and propose solutions based on a heuristic algorithm.

There are many other fields that deal with PM scheduling problems; in the military context, we can mention [6], who have developed a dynamic approach for scheduling PM of engine modules. Maintenance scheduling problems involving machines have been investigated by Hadidi *et al.* [4] and Keha *et al.* [8] for instance, and in a paper factory by Garg *et al.*[3].

In this paper, we assume that we have collected information about defects which were not handled during the primary maintenance management program described above by Zhou [14]. This information consists of an acceptable limit date for repairs; we call this date *the deadline*. In the rest of this paper, the repairs not handled during the primary inspection, with their associated deadlines, will be called the primary repair schedule, and are the starting point of our study. This paper models the economic preventive maintenance scheduling problem in gas pipelines as a binary integer non-linear programming model, and presents an algorithmic solution based on dynamic programming. This algorithm is performed in polynomial time and computes the global solution; it proposes also a class of alternative solutions which may assist industrial personnel in making decisions.

The remainder of this paper is organized as follows. The model is formulated in Section 2. Section 3 models the PM problem by a binary nonlinear program. Section 4 focuses on the algorithmic solution. Computational results are presented in Section 5, and we conclude in Section 6

2 Problem description

After a primary inspection defined at time $t = 0$, a long term horizon is fixed, which we denote $T^* \in \mathbb{N}^*$; no repairs will be handled after T^*, which may be thought of as the maximal time before the next inspection. The next inspection may be happen at time t, $t \in \{1, 2, \ldots, T^*\}$. Define the inspection interval by $\Delta t = (0, t]$. During the primary inspection, a number of corrosion defects are detected. Some of them, considered as unacceptable for the safety of the pipeline, are repaired immediately. Those which do not call into question the pipeline integrity are not immediately repaired; to each of these is associated a deadline corresponding to the limit date for repair. So, the pipeline manager must plan repair activities no later than this deadline for each specific defect. Beyond this date, the safety of the pipeline is seriously compromised. Assume that we have knowledge of these deadlines and the number of defects to be repaired for each of these

dates. We may thus define what we have called in Section 1 the *primary repair schedule*:

$$\mathscr{P} = \{(n_1, T_1), \ldots, (n_N, T_N)\}, \tag{1}$$

where $N > 0$ is the number of different deadlines. For $i = 1, \ldots N$, n_i is the number of defects to be repaired before their deadline T_i, with $n_i > 0$, $T_i < T^*$ and $T_1 < \ldots < T_N$ (with $T_0 = 0$). From the primary repair schedule $\mathscr{P}$ defined by (1), we seek the next optimal inspection time, denoted by t^*, belonging to the set $\{1, \ldots, T^*\}$, and the optimal repair program within the inspection interval Δt^* that minimizes the operational cost.

2.1 Operational cost

Preventive maintenance activities include inspection and repairs. Moreover, gas pipelines must be interrupted for safety during maintenance activities. These activities generate a significant out-of-service cost corresponding to the financial loss due to the inactivity of the gas pipeline. The costs to be considered are the inspection cost, denoted by C_{insp}, the repair cost C_{rep}, and the out-of-service cost, C_{out}. Hence, the operational cost is

$$C_{tot} = C_{insp} + C_{rep} + C_{out}. \tag{2}$$

Remark 2.1. *The out-of-service cost C_{out} represents the financial loss due to repair activities. The inspection cost also takes into account an out-of-service cost.*

Let $\mathscr{P} = \{(n_i, T_i), i = 1, \ldots, N\}$ be the primary repair schedule and let t, $t \in \{1, \ldots, T^*\}$ be any fixed inspection time. The total cost (2) depends on both t and the repair plan within the inspection interval Δt. Denote by N_t the number of deadlines T_i, $i \in \{1, \ldots, N\}$ within Δt, *i.e.*,

$$N_t = card\{i \in \{1, \ldots, N\} : T_i \leq t\}. \tag{3}$$

When the next inspection is planned at time t, the operational cost related to $\mathscr{P}$, $C(t, \mathscr{P})$ is given by

$$C(t, \mathscr{P}) = C_{insp}(t) + \sum_{i=1}^{N_t} n_i C_{rep}(T_i) + \sum_{i=1}^{N_t} C_{out}(T_i). \tag{4}$$

Let us first make some assumptions about the costs defined above. We will consider two economic parameters: the discount rate and the inflation rate. For a given initial cost, for example, with initial inspection cost denoted C^0_{inp}, the inspection cost at time $t > 0$ will be given by

$$C_{insp}(t) = C^0_{inp} \times \left(\frac{1 + r_i}{1 + r_d}\right)^t,$$

where r_i and r_d are respectively the inflation rate and discount rate. Moreover, we assume that $r_i < r_d$. Thus, the function $t \mapsto C_{insp}(t)$ is decreasing. This will also be true for $C_{rep}(.)$ and $C_{out}(.)$.

The PM scheduling problem is twofold; the first part consists of selecting the next inspection time within the set $\{1, 2, \ldots, T^*\}$, and the second, to plan the repair schedule within Δt in order to minimize the total cost. In the following, the optimal solution will be denoted by $(t^*, \mathscr{P}^*_t)$. This problem is highly combinatorial. It consists of finding the next inspection time within $\{1, \ldots, T^*\}$, and the least expensive repair program among all possible programs achieved from $\mathscr{P}$, under the constraint that a defect cannot be repaired after its deadline. Thus, finding the exact solution in a short time cannot be expected. However as we will seen in the forthcoming section, by exploiting properties of the model (Section 3.1), we will be able to reduce the space of feasible solutions. This

combinatorial optimization problem will be modeled using a binary nonlinear programming model in Section 3.2, and an effective algorithmic solution will be proposed in Section **??**.

3 Mathematical model

3.1 Some properties of the model

Before formulating the mathematical model as a binary integer nonlinear program, it is worth noting a number of simple properties. We will use the fact that we can only do repairs early, and not late; the repair and the out-of-service costs decrease with time; anticipating repairs where other repairs are planned does not add an out-of-service cost. These properties will allow us to reduce the space of repair schedules to explore to $(1-2^{N+1})/(1-2)-1$. Let us introduce some notation that we will use in the following. Denotes by $\mathscr{D}_k$ the set of deadlines (including 0) up to T_k, for $k = 1, \ldots, N-1$,

$$\mathscr{D}_k = \{T_1, \ldots, T_k\} \cup \{0\}.$$

We denote by $\mathscr{D}$ (the subscript N will be omitted) the set of all deadlines (including 0), *i.e.*,

$$\mathscr{D} = \{T_1, \ldots, T_N\} \cup \{0\}.$$

Property 1. *All defects with same deadline will be repaired at the same time.*

For example, 3 defects have to repaired before year 15. Thus, we will repair the 3 defects at the same time, either at year 0 (during the primary inspection), at year 1, 2, and so on, up to year 15. This means that we will not split up repairs, since this action increases the total cost, adding a repair cost and/or an out-of-service cost. This fact is expressed in the following proposition.

Proposition 3.1. *Let t be an inspection time, $t \in \{1, \ldots T^*\}$, and let $\mathscr{P} = \{(n_i, T_i), i = 1, \ldots, N\}$ be the primary repair program with total cost $C(t, \mathscr{P})$. Let N_t be the number of deadlines within Δt defined by (3). For any set of defects $n_1, n_2, \ldots, n_N$, the new repair program defined after splitting a set is more expensive than $\mathscr{P}$.*

Proof. We will prove this proposition by considering only one set n_k, with $k \in \{1, \ldots N\}$. The result will be applied when considering several sets. We consider the kth set of defects n_k, such that $n_k \geq 2$, which must be repaired before T_k. Suppose that we have split the set n_k into two other sets, $n_k^{'}$ and $n_k^{''}$, such that $n_k = n_k^{'} + n_k^{''}$, where $n_k^{'} \geq 1$ is the number of repairs that will be performed early at time τ with $\tau < T_k$. Without loss of generality, assume that the $n_k^{''} \geq 1$ defects will be repaired at time T_k. We assume initially that $k \leq N_t$, i.e., the n_k defects are within Δt.

- If $\tau \notin \mathscr{D}_{k-1}$, define the new repair schedule as

$$\mathscr{P}_1 = \left\{(n_1, T_1), \ldots, (n_k^{'}, \tau), \ldots, (n_k^{''}, T_k), (n_{k+1}, T_{k+1}), \ldots, (n_N, T_N)\right\}. \tag{5}$$

Noting that planning repairs before deadlines add an out-of-service cost, the total cost of $\mathscr{P}_1$ is given by

$$C(t, \mathscr{P}_1) = C_{insp}(t) + \sum_{i=1}^{N_t} n_i C_{rep}(T_i) - n_k^{'} C_{rep}(T_k) + n_k^{'} C_{rep}(\tau) + \sum_{i=1}^{N_t} C_{out}(T_i) + C_{out}(\tau).$$

Using the fact that $C_{rep}(T_k) < C_{rep}(\tau)$, we get

$$C(t, \mathscr{P}) - C(t, \mathscr{P}_1) = n_k^{'}[C_{rep}(T_k) - C_{rep}(\tau)] - C_{out}(\tau) < 0.$$

• If $\tau \in \mathscr{D}_{k-1}$, then there exists i such that $\tau = T_i < T_k$. The new repair plan is given by

$$\mathscr{P}_2 = \left\{(n_1, T_1), \ldots, (n_i + n_k^{'}, T_i), \ldots, (n_k^{''}, T_k), (n_{k+1}, T_{k+1}), \ldots, (n_N, T_N)\right\}. \tag{6}$$

Remark next that repairs made earlier than absolutely necessary at a time where other repairs are planned does not add an out-of-service cost. Thus, the total cost of $\mathscr{P}_2$ is given by

$$C(t, \mathscr{P}_2) = C_{insp}(t) + \sum_{i=1}^{N_t} n_i C_{rep}(T_i) - n_k^{'} C_{rep}(T_k) + n_k^{'} C_{rep}(T_i) + \sum_{i=1}^{N_t} C_{out}(T_i).$$

Thus,

$$C(t, \mathscr{P}) - C(t, \mathscr{P}_2) = n_k^{'}[C_{rep}(T_k) - C_{rep}(T_i)] < 0.$$

Assume now that $k > N_t$, i.e., the set n_k is not within Δt, but we have the opportunity to perform early the set $n_k^{'}$ within Δt. In this case, all repair schedules that we can define will be more expensive than $\mathscr{P}$ because we add to $\mathscr{P}$ an out-of service cost depending on whether $\tau \in \mathscr{D}_{N_t}$ or not, and on the monotonicity of $C_{rep}(.)$. Therefore, in all cases the cost of $\mathscr{P}$ is less expensive than all repair plans defined by splitting. □

Remark 3.1. *We have supposed that the $n_k^{''}$ repairs take place at T_k. We could have decided to repairs early, but the associated repair schedule would be more expensive than (5) and (6).*

Remark 3.2. *We have added an out-of-service cost at time τ to (3.1) because no repair was planned at time τ. Thus, moving forward repairs from deadlines generates a cost due to the unavailability of gas from the pipeline.*

Remark 3.3. *For a given inspection time t, when the set of n_k defects is not in Δt (i.e., $k > N_t$), moving some repairs into the inspection interval is more expensive than repairing the set n_k at time T_k, because we add an out-of-service cost, and $C_{rep}(.)$ is decreasing.*

Remark 3.4. *We also observe that the first part of the proof suggests that repairs from after deadlines done early generate more expensive repair plan schedules than $\mathscr{P}$. This property will be proved below.*

Conclusion 1. *Proposition 3.1 allows for all split scenarios to be rejected, thus reducing the space of feasible solutions. Moreover, we can restrict our attention to deadlines within Δt in order to build the optimal repair senario.*

Property 2. *There is no monetary advantage to repair defects outside the times $\mathscr{D}$.*

For example, we fix the horizon time $T^* = 20$, and the next inspection at $t = 18$. Denote by $\mathscr{P} = \{(1,3),(2,10),(3,15)\}$ the primary repair schedule with total cost $C(t, \mathscr{P})$ given by

$$C(18, \mathscr{P}) = C_{insp}(18) + C_{rep}(3) + 2C_{rep}(10) + 3C_{rep}(15) + C_{out}(3) + C_{out}(10) + C_{out}(15).$$

Let $\mathscr{P}_1$ be a new repair schedule where we have repaired early the third set of defects with deadline 15 at year τ such that $\tau \notin \mathscr{D}_2 = \{3, 10\} \cup \{0\}$ and $\tau < 15$. The total cost of $\mathscr{P}_1$ is

$$C(18, \mathscr{P}_1) = C_{insp}(18) + C_{rep}(3) + 2C_{rep}(10) + 3C_{rep}(\tau) + C_{out}(3) + C_{out}(10) + C_{out}(\tau).$$

Hence,

$$C(18, \mathscr{P}) - C(18, \mathscr{P}_1) = 3\underbrace{(C_{rep}(15) - C_{rep}(\tau))}_{<0} + \underbrace{(C_{out}(15) - C_{out}(\tau))}_{<0}.$$

This leads to the following Proposition.

Proposition 3.2. *Let t be a fixed inspection time, with $t \in \{1, \ldots, T^*\}$, and let $\mathscr{P}$ be the primary repair program with total cost $C(t, \mathscr{P})$. Then, the total cost related to any repair schedules such that some defects are repaired before their deadline, is more expensive than $\mathscr{P}$.*

Proof. We prove the result for only one set of defects repaired early. According to Proposition 3.1, repairs with same deadline are handled at the same time. Let $\mathscr{P}_1$ be a repair schedule such that n_k repairs are done early at time τ such that $\tau < T_k$. Assume furthermore that $\tau \notin \mathscr{D}_{k-1}$. The total cost of $\mathscr{P}_1$ is

$$C(t, \mathscr{P}_1) = C_{insp}(t) + \sum_{i=1, i\neq k}^{N_t} n_i C_{rep}(T_i) + n_k C_{rep}(\tau) + \sum_{i=1, i\neq k}^{N_t} C_{out}(T_i) + C_{out}(\tau).$$

Thus,

$$C(t, \mathscr{P}) - C(t, \mathscr{P}_1) = n_k[C_{rep}(T_k) - C_{rep}(\tau)] - C_{out}(\tau) < 0.$$

This result can then be applied when considering several blocks, in order to conclude the proof. □

Conclusion 2. *Proposition 3.2 allows for all repair schedules that include repairs done before their deadline to be rejected.*

Remark 3.5. *At this stage, we have moved from a very large space of schedules to a space of $(N_t+1)!$ possible schedules for a given inspection time t. Indeed, using Conclusion 1 and Conclusion 2, we can repair the set n_1 either at time T_1 or earlier at time T_0 (two choices); the set n_2 can be repaired either at time T_2, or either at time T_1 or either at time T_0. For the last set n_{N_t}, we have $N_t + 1$ possibilities. Thus, we obtain $(N_t + 1)!$ plans to choose between if the next inspection is planned at time t.*

The next proposition states that the space of repair schedules may be reduced to 2^{N_t} for a given inspection time t. Denote by $\mathscr{P}_t^j$ the repair program such that the N_t repairs (the last set of defects within Δt) has been done early at time T_j, $j = 0, \ldots, N_t - 1$,

$$\mathscr{P}_t^j = \{(n_0, T_0), (n_1, T_1), \ldots, (n_j + n_{N_t}, T_j), (n_{j+1}, T_{j+1}), \ldots, (n_{N_t-1}, T_{N_t-1})\}, \quad n_0 = 0. \tag{7}$$

For $j = 0, \ldots, N_t - 1$, the associated total cost is given by

$$C_t^j = C_{insp}(t) + n_1 C_{rep}(T_1) + \ldots + (n_j + n_{N_t}) C_{rep}(T_j) + \ldots + n_{N_t-1} C_{rep}(T_{N_t-1}) + \sum_{i=1}^{N_t-1} C_{out}(T_i).$$

Proposition 3.3. *The number of feasible solutions is 2^{N_t} for a given inspection time $t \in \{1, \ldots, T^*\}$.*

Proof. We shall prove this proposition by induction on N_t. For $N_t = 1$, i.e., the inspection is planned so that there is only one set n_1 (with deadline T_1) within Δt. In this case, the proposition is proved. We shall prove the result for $N_t = 2$ and suppose that the n_1 repairs are planned. We shall prove that there are two repair scenarios for the set of n_2 defects. We consider two cases. In the first, the n_1 repairs take place at time T_1; in the second, they are moved to time $T_0 = 0$.

First case. We have $\mathscr{P}_t = \{(n_1, T_1), (n_2, T_2)\}$ with total cost $C(t, \mathscr{P}_t)$ given by

$$C(t, \mathscr{P}_t) = C_{insp}(t) + \sum_{i=1}^{2} n_i C_{rep}(T_i) + \sum_{i=1}^{2} C_{out}(T_i).$$

The total cost associated with $\mathscr{P}_t^1 = \{(n_1 + n_2, T_1)\}$ is

$$C_t^1 = C_{insp}(t) + (n_1 + n_2)C_{rep}(T_1) + C_{out}(T_1),$$

and the cost related to $\mathscr{P}_t^0 = \{(n_2, T_0), (n_1, T_1)\}$ is

$$C_t^0 = C_{insp}(t) + n_2 C_{rep}(T_0) + n_1 C_{rep}(T_1) + C_{out}(T_1).$$

When comparing C_t^1 with C_t^0, we obtain

$$C_t^1 - C_t^0 = n_2 \left[C_{rep}(T_1) - C_{rep}(T_0)\right] < 0,$$

and thus can rule out the repair plan $\mathscr{P}_t^0$. When comparing $C(t, \mathscr{P}_t)$ with C_t^1, we have

$$C(t, \mathscr{P}_t) - C_t^1 = n_2 \underbrace{\left[C_{rep}(T_2) - C_{rep}(T_1)\right]}_{<0} + C_{out}(T_2).$$

In this case, we cannot conclude which is the best program in terms of minimal cost. Consequently, for a given inspection time t, we may repair the set n_2 either at its deadline T_2 ($\mathscr{P}_t$) or early at time T_1 ($\mathscr{P}_t^1$).

Second case. Set $\tilde{\mathscr{P}}_t = \{(n_1, T_0), (n_2, T_2)\}$ with total cost $C(t, \tilde{\mathscr{P}}_t)$ given by

$$C(t, \tilde{\mathscr{P}}_t) = C_{insp}(t) + n_1 C_{rep}(T_0) + n_2 C_{rep}(T_2) + C_{out}(T_2).$$

The cost related to $\tilde{\mathscr{P}}_t^1 = \{(n_1, T_0), (n_2, T_1)\}$ is

$$C_t^1 = C_{insp}(t) + n_1 C_{rep}(T_0) + n_2 C_{rep}(T_1) + C_{out}(T_1),$$

and the total cost associated with $\tilde{\mathscr{P}}_t^0 = \{(n_1 + n_2, T_0)\}$ is

$$C_t^0 = C_{insp}(t) + (n_1 + n_2)C_{rep}(T_0).$$

Since $C(t, \tilde{\mathscr{P}}_t) - C_t^1 = n_2 \left[C_{rep}(T_2) - C_{rep}(T_1)\right] + C_{out}(T_2) - C_{out}(T_1) < 0$, we can rule out the schedule $\tilde{\mathscr{P}}_t^1$. Furthermore, when comparing $C(t, \tilde{\mathscr{P}}_t)$ with C_t^0, we have

$$C(t, \tilde{\mathscr{P}}_t) - C_t^0 = n_2 \left[C_{rep}(T_2) - C_{rep}(T_0)\right] + C_{out}(T_2),$$

which does not allow us to conclude anything, so the proposition is proved for $N_t = 2$.

Now, let t be an inspection time such that $N_t = k + 1$. Assuming that the set $n_1, \ldots, n_k$ are planned, we will consider only one configuration. We suppose that the sets of $n_1, \ldots, n_k$ defects take place respectively at times $T_1, \ldots, T_k$. We shall prove that for a given inspection time t, there are two choices for positioning the $(k+1)$th repairs. Using the fact that $C_{rep}(T_k) < C_{rep}(T_{k-1}) < \ldots < C_{rep}^0$, and the fact that repairing early when other repairs are planned does not add an out of service cost, yields

$$C_t^k < C_t^{k-1} < \ldots < C_t^0.$$

Thus, we can rule all repair plans such that the n_{k+1} repairs are done at an earlier date than T_k. There remain only two repair schedules to compare, $\mathscr{P}_t$ and $\mathscr{P}_t^k$, with respective total costs $C(t, \mathscr{P}_t)$ and C_t^k.

$$C(t, \mathscr{P}_t) - C_t^k = n_k \underbrace{\left[C_{rep}(T_{k+1}) - C_{rep}(T_k)\right]}_{<0} + C_{out}(T_{k+1}).$$

We cannot conclude which is the best. Consequently, we may either plan the n_{k+1} repairs at year T_k (schedule $\mathscr{P}_t^k$) or at year T_{k+1} (schedule $\mathscr{P}_t$). □

Conclusion 3. *Proposition 3.3 states that for a given inspection time t, the number of schedules to consider is 2^{N_t}. For a given set of defects n_k, $k \in \{1, \dots, N_t\}$, there are two repair strategies. Either the set n_k is handled at its deadline T_k, or earlier at the previous deadline, where repairs are already planned.*

The next proposition plays a crucial role in designing the algorithm which will solve the optimization problem (8) in the forthcoming section. Let $\tilde{\mathscr{P}}$ be any repair schedule achieved from $\mathscr{P}$.

Proposition 3.4. *For any repair schedule $\tilde{\mathscr{P}}$, the function $s \in (0, T^*) \mapsto C_{tot}(s, \tilde{\mathscr{P}})$ is not convex and has local minima at points $s = \{T_j - 1,\ j = 1, \dots, N\} \cup \{T^*\}$. The optimal inspection t^* is found at one of these.*

Proof. The total cost function $C_{tot}(., \tilde{\mathscr{P}})$ is the sum of a decreasing function $C_{insp}(.)$ and a step function $(C_{rep} + C_{out})(.)$ with jump discontinuities at $T_j - 1$ for $j = 1, \dots, N$. Then, the total cost function $C_{tot}(., \mathscr{P})$ is increasing within $[T_j - 1, T_j]$ and decreasing within $[T_j, T_{j+1} - 1]$ and $[T_{N_t}, T^*]$. Thus $s \mapsto C_{tot}(s, \mathscr{P})$ cannot be convex and has local minima at $T_j - 1$, $j = 1, \dots, N$. Therefore, the global minimum is found at one of these dates. □

Remark 3.6. *Note that if $T_1 = 1$, $T_1 - 1$ should not be considered as a candidate for the next inspection because it would coincide with the primary inspection.*

Conclusion 4. *Proposition 3.4 suggests that the optimal solution for the PM schedule problem should be chosen for an inspection time among $\{T_j - 1, j = 1, \dots, N\} \cup \{T^*\}$ instead of $\{1, \dots, T^*\}$. This implies that the space of feasible solutions may be reduced to $\sum_{k=1}^{N} 2^k$ repair plans to consider.*

This proposition pushes us to solve the PM scheduling problem in a support decision framework, by building an algorithm that proposes a set of repair schedules related to inspection times within $t \in \{T_j - 1, j = 1, \dots, N\} \cup \{T^*\}$, which includes the optimal PM schedule $(t^*, \mathscr{P}_t^*)$, in order to support decisions of pipeline managers.

3.2 Formulating PM as an integer programming problem

The problem described above can be formulated as a binary integer nonlinear programming model, taking into account all the previous propositions, and certain constraints that we shall now describe.

We are looking for an optimal inspection time t^* such that $t^* \in \{T_1 - 1, \dots, T_N - 1, T^*\}$, and a repair schedule within Δt^* that minimizes the total cost. In the following, we introduce two decision variables a_i and b_j, for $i = 1, \dots, N+1$, and $j = 0, \dots, N$, defined by:

$$a_i = \begin{cases} 1 & \text{if an inspection is planned in year } T_i - 1 \\ 0 & \text{otherwise,} \end{cases}$$

$$a_{N+1} = \begin{cases} 1 & \text{if an inspection is planned in year } T^* \\ 0 & \text{otherwise,} \end{cases}$$

and

$$b_j = \begin{cases} 1 & \text{if repairs are planned in year } T_j \\ 0 & \text{otherwise,} \end{cases}$$

where $T_0 = 0$. Denote by a and b the variables

$$a = (a_1, \dots, a_{N+1}) \quad \text{and} \quad b = (b_0, \dots, b_N).$$

Both vectors must satisfy certain constraints, which we now describe. The first ensures that on the time interval $(0, T^*]$, there is only one inspection after the primary inspection. Note that if $T_1 = 1$, we cannot plan the next inspection at time 0, that is during the primary inspection; thus, define the variable α such that

$$\alpha = \begin{cases} 1 \text{ if } T_1 \neq 1 \\ 0 \text{ otherwise.} \end{cases}$$

Therefore, the constraint stating that there is only one inspection on $(0, T^*]$ is reflected as:

$$\alpha a_1 + \sum_{j=2}^{N+1} a_j = 1.$$

For example, if $T^* = 10$, $N = 4$ and $T_1 \neq 1$, the variable $a = (0, 0, 0, 0, 1)$ of length 5 means that an inspection is planned at time T^*. However, if $T_1 = 1$, the vector a has length 4, and $a = (0, 1, 0, 0)$ means that the inspection is planned at year $T_3 - 1$.

A second constraint encodes the fact that we cannot plan repairs simultaneously at times T_0 and T_1. Thus,

$$b_0 + b_1 = 1.$$

Indeed, Proposition 3.3 states that there are two options when planning repairs. Suppose that we decide to repair the n_1 defects at time T_1 (at their deadline). Then, the n_2 defects may be repaired either at time T_2 (their deadline) or time T_1, but not at time $T_0 = 0$. Suppose now that the n_1 repairs were moved at time T_0, then the 2 repairs may be planned at time T_2 or at time T_0, but not at time T_1. Since defects with the same deadline have to be repaired at the same time, we have at most N sets of repairs:

$$1 \leq \sum_{j=0}^{N} b_j \leq N.$$

The PM scheduling problem is then the following:

$$\begin{aligned} &\text{Minimize} \quad C_{tot}(a_1, \ldots, a_{N+1}, b_0, \ldots, b_N) \\ &\text{subject to} \begin{cases} a_i \in \{0, 1\} \text{ for all } i = 1, \ldots, N+1 \\ b_j \in \{0, 1\} \text{ for all } j = 0, \ldots, N \\ \alpha a_1 + \sum_{i=2}^{N+1} a_i = 1, \quad \alpha \in \{0, 1\} \\ b_0 + b_1 = 1 \\ 1 \leq \sum_{j=0}^{N} b_j \leq N, \end{cases} \end{aligned} \tag{8}$$

where the objective function is given by

$$\begin{aligned} C(a_1, \ldots, a_{N+1}, b_0, \ldots, b_N) &= \alpha a_1 C_{insp}(T_1 - 1) + \sum_{i=2}^{N} a_i C_{insp}(T_i - 1) + a_{N+1} C_{insp}(T^*) \\ &+ \sum_{j=0}^{N-1} b_j (\Pi_{i=1}^{j}(1 - a_i))(n_j + (1 - b_{j+1})(n_{j+1} + (1 - b_{j+2})(n_{j+2} + \ldots + (n_N(1 - b_N)) \ldots))C_{rep}(T_j) \\ &+ b_N \times (\Pi_{i=1}^{N}(1 - a_i)) \times n_N C_{rep}(T_N) + \sum_{j=1}^{N} b_j (\Pi_{i=1}^{j}(1 - a_i)) C_{out}(T_j), \end{aligned} \tag{9}$$

with the conventions $n_0 = 0$ and $\Pi_{i=1}^{T_0}(1-a_i) = 1$. We illustrate this objective function with the following example.

Example 3.1. *We fix the time horizon T^* as $T^* = 10$ and the number of deadlines N as $N = 3$. Set $T_1 = 2$, $T_2 = 4$ and $T_3 = 8$, and for all $i \in \{1,2,3\}$, we set $n_i = 1$. Thus, the primary PM (1) is given by*

$$\mathscr{P} = \{(1,T_1),(1,T_2),(1,T_3)\}.$$

Suppose that we want an inspection to take place at year $t = T_3 - 1 = 7$. Since $T_1 \neq 1$ then $\alpha = 1$, and thus

$$a = (0,0,1,0).$$

Suppose furthermore that we want to do repairs with deadline $T_2 = 4$ early, at year $T_1 = 2$; then, the new PM schedule $\mathscr{P}_t^1$ (7), with $N_t = 2$, is $\mathscr{P}_t^1 = \{(1+1,T_1)\}$; thus the vector b is defined by

$$b = (0,1,0,0).$$

To perform the total cost of this program, a and b are substituted into (9), with

$$a = (0,0,1,0) \qquad \text{and} \qquad b = (0,1,0,1).$$

The calculation of the inspection cost for $\mathscr{P}_t^1$ gives

$$\begin{aligned} C_{insp}(a) &= a_1 C_{insp}(T_1 - 1) + a_2 C_{insp}(T_2 - 1) + a_3 C_{insp}(T_3 - 1) + a_4 C_{insp}(T^*) \\ &= a_3 C_{insp}(T_3 - 1) = 1 \times C_{insp}(7). \end{aligned}$$

The calculation of the reparation cost is somewhat complicated because it has to take into account whether or not the deadlines are within the inspection interval $\Delta t = (0, T_3 - 1]$, and the total number of repairs for each deadline. Noting that T_0, T_1 and T_2 are within Δt, we multiply the variables b_0, b_1 and b_2 respectively by $\Pi_{i=1}^{0}(1-a_i) = 1$, $(1-a_1) = 1$ and $(1-a_1)(1-a_2) = 1$. Since $T_3 \notin \Delta t$, the variable b_3 will be multiplied by $(1-a_1)(1-a_2)(1-a_3) = 0$, and does not contribute to the total cost, even if $b_3 = 1$. The total number of repairs for each deadline is expressed by a non-linear term that involves n_j and $(1-b_j)$. Thus, the calculation of the repair cost yields

$$\begin{aligned} C_{rep}(a,b) &= b_1 \left((1-a_1)\right) \left(n_1 + (1-b_2)(n_2 + (n_3(1-b_3)))C_{rep}(T_1)\right. \\ &= 1 \times \left((1-0)\right) \left(n_1 + (1-0)(n_2 + n_3(1-1)))C_{rep}(T_1)\right. \\ &= (n_1 + n_2) C_{rep}(T_1). \end{aligned}$$

The out-of-service cost is given by

$$C_{out}(a,b) = b_1 \times (1-a_1) C_{out}(T_1) = 1 \times C_{out}(T_1).$$

We obtain

$$C(a,b) = C_{insp}(6) + 2C_{rep}(T_1) + C_{out}(T_1),$$

which is the total cost of $\mathscr{P}_t^1$.

Remark 3.7. *Proposition 3.4 states that the optimal solution of (8) is achieved for an inspection time belonging to $\{T_1 - 1, \ldots, T_N - 1, T^*\}$. Then, the size of the space of feasible solutions may be reduced to $\sum_{k=0}^{N} 2^k - 1$. Indeed, if $t = T_1 - 1$, then $N_t = 0$ and there is no repair plan to explore. If $t = T_2 - 1$, then $N_t = 1$ and there are 2^1 repair plans to consider. If $t = T_3 - 1$, 2^2 repair schedules have to be considered, which yields the result.*

4 Solution-finding strategy

The PM scheduling problem consists of finding an optimal inspection time t^* such that $t^* \in \{T_1 - 1, \ldots, T_N - 1, T^*\}$, and a repair schedule within Δt^* that minimizes the total cost. The search for the optimal solution requires considering $(1-2^{N+1})/(1-2)-1$ possible repair plans. We will see in this section how to reduce this number to $(N+1)(N+2)/2-1$. We outline an algorithm in the context of the support decision framework, which computes the optimal PM schedule, together with alternative solutions, in order to propose to the pipeline manager a set of "good" solutions. A naive way to solve the problem is to look through, for a fixed inspection time $t \in \{T_j - 1, j = 1, \ldots, N, T^*\}$, all of the 2^{N_t} feasible solutions, and save any of those with the best objective function. For large values of N, such a method is not efficient. To quantify the efficiency of our algorithm, we have developed two other algorithms related to the problem (8)). The first investigates all repair schedules when $t \in \{T_j - 1, j = 1, \ldots, N, T^*\}$, and the second, which will serve as a benchmark, looks through all repair plans when $t \in \{1, 2, \ldots, T^*\}$. This will be developed in Section 5.

4.1 Construction of the algorithm

The aim of this section is to develop a strategy which builds a search tree and returns the optimal schedule for a given inspection time. In order to design our algorithm, we shall use the following example. Over a time horizon $T^* = 20$, we consider $N = 3$ sets of defects with cardinality n_1, n_2, n_3, respectively, with associated deadlines T_1, T_2 and T_3, and we will fix the inspection time as $t = T^*$. Note that this search tree will not necessarily find the global optimum. In order to find that, we have to build all $N + 1$ search trees, each related to inspection times, and compare the associated total costs.

4.1.1 First and second generations

The inspection time is $t = T^*$, thus the number of set of defects that will be considered is $N_t = 3$. We need to look through $2^3 = 8$ repair schedules. We shall represent the various repair plans by a tree, where each branch represents one strategy. For example, the branch $br(2)$ corresponds to the repair plan $\{(n_1, T_1), (n_2 + n_3, T_2)\}$ (see Figure 1).

The strategy is to look through all 8 repair plans in order to discard the most expensive branches. After removing some branches, a final tree will remain, corresponding to potential solutions; the least expensive will be the optimal for fixed inspection times (but not necessarily the global optimum). Note that since $N_t = 3$, the final tree will have three generations. This methodology will allow us to draw rules for an iterative construction of the final search tree. Figure 8 depicts such a final tree.

We are first interested in building and comparing the branches $br(1)$, $br(2)$, $br(5)$, $br(6)$. The nodes (n_1, T_0) and (n_1, T_1) are labeled with the total costs, denoted by $CT(1,0)$ and $CT(1,1)$ respectively, corresponding to the cost related to the *partial plans* $\{(n_1, T_0)\}$ and $\{(n_1, T_1)\}$(see Figure 1). The costs $CT(1,0)$ and $CT(1,1)$ are given by

$$CT(1,0) = C(t, \{(n_1, T_0)\}) = C_{insp}(t) + n_1 C_{rep}(T_0), \tag{10}$$

and

$$CT(1,1) = C(t, \{(n_1, T_1)\}) = C_{insp}(t) + n_1 C_{rep}(T_1) + C_{out}(T_1). \tag{11}$$

The total costs of the branches $br(1)$, $br(2)$, $br(5)$, $br(6)$ are

$$\begin{aligned}
C(t,\, br(1)) &= CT(1,1) + n_2 C_{rep}(T_2) + C_{out}(T_2) + n_3 C_{rep}(T_3) + C_{out}(T_3),\\
C(t,\, br(5)) &= CT(1,0) + n_2 C_{rep}(T_2) + C_{out}(T_2) + n_3 C_{rep}(T_3) + C_{out}(T_3),\\
C(t,\, br(2)) &= CT(1,1) + n_2 C_{rep}(T_2) + C_{out}(T_2) + n_3 C_{rep}(T_3),\\
C(t,\, br(6)) &= CT(1,0) + n_2 C_{rep}(T_2) + C_{out}(T_2) + n_3 C_{rep}(T_2).
\end{aligned}$$

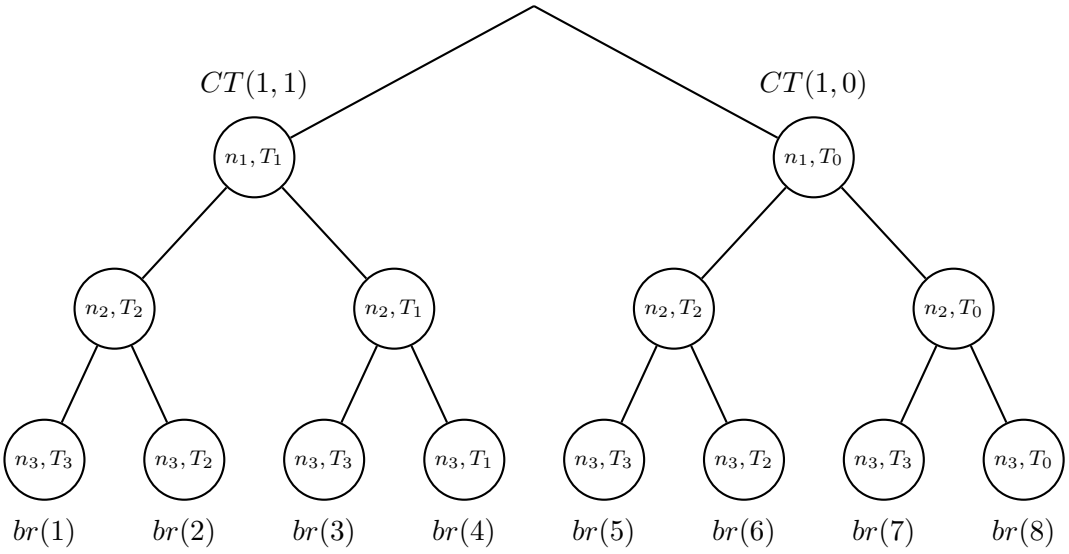

Figure 1: The search tree representing the 2^{N_t} PM schedules when $t = T^*$.

When comparing the repair schedules $br(1)$ with $br(5)$, and $br(2)$ with $br(6)$, we get

$$C(t,\, br(1)) - C(t,\, br(5)) = CT(1,1) - CT(1,0) = n_1(\underbrace{C_{rep}(T_1) - C_{rep}(T_0)}_{<0}) + \underbrace{C_{out}(T_1)}_{>0},$$

and

$$C(t,\, br(2)) - C(t,\, br(6)) = CT(1,1) - CT(1,0) = n_1(C_{rep}(T_1) - C_{rep}(T_0)) + C_{out}(T_1).$$

At this stage, we cannot conclude as to which is the most expensive program, so we must add a condition on

$$CT(1,1) - CT(1,0).$$

• If $CT(1,1) < CT(1,0)$, then the node (or the partial repair program) $\{(n_1, T_1)\}$ has the smallest cost function, and we get

$$C(t,\, br(1)) < C(t,\, br(5)) \qquad \text{and} \qquad C(t,\, br(2)) < C(t,\, br(6)).$$

The repair programs $br(5)$ and $br(6)$ are respectively more expensive than $br(1)$ and $br(2)$; the branches $br(5)$ and $br(6)$ are rejected, i.e., the child node (n_2, T_2) of (n_1, T_0) is discarded.

• If $CT(1,0) < CT(1,1)$, then the node (n_1, T_0) has the smallest cost function

$$C(t,\, br(1)) > C(t,\, br(5)) \qquad \text{and} \qquad C(t,\, br(2)) > C(t,\, br(6)).$$

The branches $br(1)$ and $br(2)$ are respectively more expensive than $br(5)$ and $br(6)$; hence $br(1)$ and $br(2)$ are rejected, i.e., the descendant of (n_1, T_1) is discarded, *i.e.*, (n_2, T_2).

We have so far compared the branches $br(1)$, $br(2)$, $br(5)$, $br(6)$. It remains to compare $br(3)$ with $br(7)$ and $br(4)$ and $br(8)$. As before, we calculate the total cost of the repair programs $br(3)$, $br(7)$, $br(4)$ and $br(8)$:

$$\begin{aligned}
C(t,\, br(3)) &= CT(1,1) + n_2 C_{rep}(T_1) + n_3 C_{rep}(T_3) + C_{out}(T_3),\\
C(t,\, br(7)) &= CT(1,0) + n_2 C_{rep}(T_0) + n_3 C_{rep}(T_3) + C_{out}(T_3),\\
C(t,\, br(4)) &= CT(1,1) + n_2 C_{rep}(T_1) + n_3 C_{rep}(T_1),\\
C(t,\, br(8)) &= CT(1,0) + n_2 C_{rep}(T_0) + n_3 C_{rep}(T_0).
\end{aligned}$$

When comparing $br(3)$ with $br(7)$, and $br(4)$ with $br(8)$, we obtain

$$C(t, br(3)) - C(t, br(7)) = CT(1,1) - CT(1,0) + n_2(\underbrace{C_{rep}(T_1) - C_{rep}(T_0)}_{<0}).$$
$$C(t, br(4)) - C(t, br(8)) = CT(1,1) - CT(1,0) + (n_2 + n_3)(C_{rep}(T_1) - C_{rep}(T_0)).$$

- If $CT(1,1) < CT(1,0)$, then

$$C(t, br(3)) < C(t, br(7)) \qquad \text{and} \qquad C(t, br(4)) < C(t, br(8)).$$

We may discard $br(7)$ and $br(8)$, that is, the descendant of (n_1, T_1), i.e., (n_2, T_0).

- If $CT(1,1) > CT(1,0)$, then

$$C(t, br(3)) - C(t, br(7)) = (\underbrace{CT(1,1) - CT(1,0)}_{>0}) + n_2(\underbrace{C_{rep}(T_1) - C_{rep}(T_0)}_{<0}),$$
$$C(t, br(4)) - C(t, br(8)) = (CT(1,1) - CT(1,0)) + (n_2 + n_3)(C_{rep}(T_1) - C_{rep}(T_0)).$$

Once again, we cannot conclude as to the most costly program, hence we keep the branches $br(3)$, $br(4)$, $br(7)$ and $br(8)$, and thus the nodes (n_2, T_1) and (n_2, T_0).

By combining these observations, we can define a rule in order to build the second generation. We begin with the search tree depicted in Figure 1, representing the $2^{N_t} = 8$ PM plans. The first generation is composed of the nodes (n_1, T_1) and (n_1, T_0), and labeled respectively with the cost functions $CT(1,1)$ and $CT(1,0)$, defined by (11) and (10).

- When $\{(n_1, T_1)\}$ has a smaller cost function than $\{(n_1, T_0)\}$, *i.e.*, when $CT(1,1) < CT(1,0)$, we discard the most expensive branches $br(5)$, $br(6)$, $br(7)$ and $br(8)$, and thus the node (n_1, T_0). Only four branches remain, among which the optimal program (for the fixed inspection time), as shown in Figure 2.

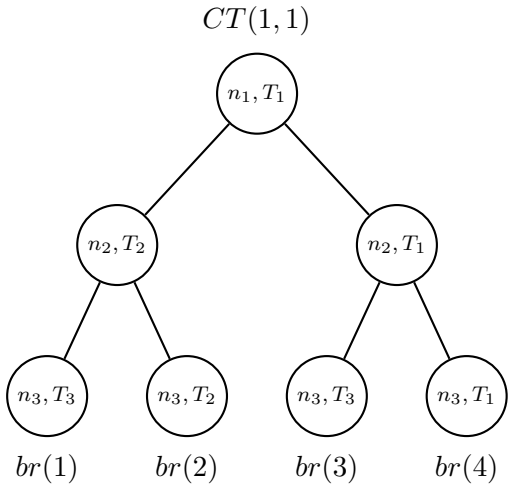

Figure 2: Search tree when $CT(1,1) < CT(1,0)$.

- Under the condition $CT(1,1) > CT(1,0)$ (i.e., the node (n_1, T_0) has a smaller cost function than (n_1, T_1)), the most expensive branches are $br(1)$ and $br(2)$. In this case, only six branches remain $(br(3), \dots br(8))$ instead of eight, among which is found the optimal program (see Figure 3).

Now, we are able to define the strategy for designing the second generation. Suppose that we have built only the first generation, $\{(n_1, T_1), (n_1, T_0)\}$, as depicted in Figure 4. Nodes are labeled with cost function $CT(1,1)$ and $CT(1,0)$ respectively.

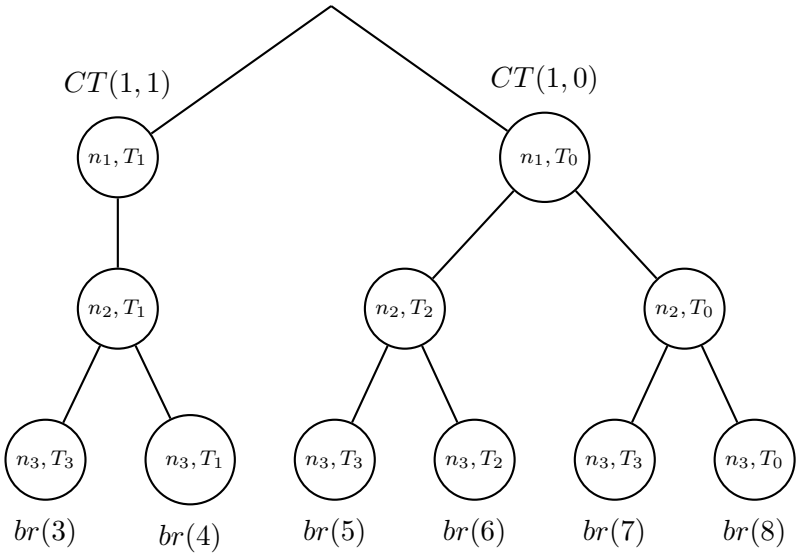

Figure 3: Search tree when $CT(1,1) > CT(1,0)$.

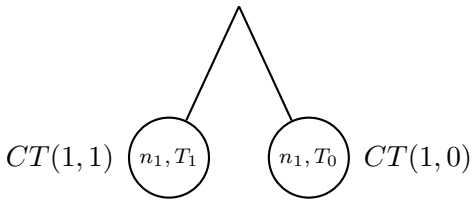

Figure 4: Construction of the first generation.

• If $CT(1,1) = \min\{CT(1,0), CT(1,1)\}$, we select (n_1, T_1) and create its two children: (n_2, T_2) and (n_2, T_1). Since $C_{rep}(T_1) < C_{rep}(T_0)$, we do not create the child of (n_1, T_0), i.e., (n_2, T_0) (see Figure 5).

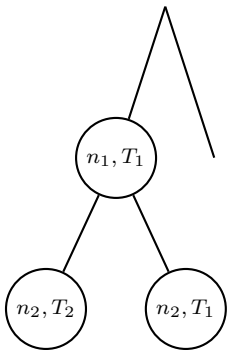

Figure 5: Second generation when $CT(1,1) = \min\{CT(1,0), CT(1,1)\}$.

Denote by $S(2)$ the set of all indices of deadlines at the second generation. Figure 5 gives $S(2) = \{1, 2\}$. Note that this set contains distinct deadlines. In the following, we will denote by $S(i)$ the set of indices of deadlines at the ith generation.

• If $CT(1,0) = \min\{CT(1,0), CT(1,1)\}$, we select the node (n_1, T_0) to create its two children,

(n_2, T_2) and (n_2, T_0). Since $C_{rep}(T_1) < C_{rep}(T_0)$, we generate the descendant of (n_1, T_1), (n_2, T_1) (see Figure 6). In this case $S(2) = \{0, 1, 2\}$.

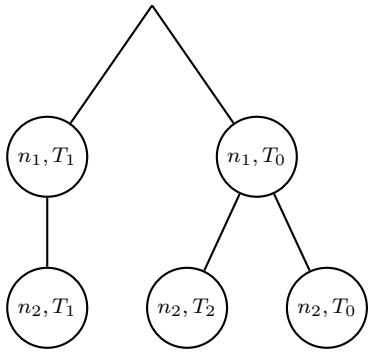

Figure 6: Second generation when $CT(1,0) = \min\{CT(1,0), CT(1,1)\}$.

4.1.2 Third generation

In order to construct the third generation, we start from Figure 3. As before, we will compare the total cost of the remaining repair programs, i.e., the branches $\{br(i), i = 3, ..., 8\}$, in order to remove the most expensive ones. Let $CT(2,0)$, $CT(2,1)$ and $CT(2,2)$ be respectively the total cost of the partial programs $\{(n_1 + n_2, T_0)\}$; $\{(n_1 + n_2, T_1)\}$ and $\{(n_1, T_0), (n_2, T_2)\}$. The nodes (n_2, T_0), (n_2, T_1) and (n_2, T_2) are respectively labeled with $CT(2,0)$, $CT(2,1)$ and $CT(2,2)$ (see Figure 7). These costs are given by

$$
\begin{aligned}
CT(2,0) &= C(t, \{(n_1, T_0), (n_2, T_0)\}) = CT(1,0) + n_2 C_{rep}(T_0), \\
CT(2,1) &= C(t, \{(n_1, T_1), (n_2, T_1)\}) = CT(1,1) + n_2 C_{rep}(T_1), \\
CT(2,2) &= C(t, \{(n_1, T_0), (n_2, T_2)\}) = CT(1,0) + n_2 C_{rep}(T_2) + C_{out}(T_2).
\end{aligned}
$$

The total costs of $\{br(i), i = 3, ..., 8\}$ are respectively

$$
\begin{aligned}
C(t, br(3)) &= CT(2,1) + n_3 C_{rep}(T_3) + C_{out}(T_3), \\
C(t, br(5)) &= CT(2,2) + n_3 C_{rep}(T_3) + C_{out}(T_3), \\
C(t, br(7)) &= CT(2,0) + n_3 C_{rep}(T_3) + C_{out}(T_3), \\
C(t, br(4)) &= CT(2,1) + n_3 C_{rep}(T_1), \\
C(t, br(6)) &= CT(2,2) + n_3 C_{rep}(T_2), \\
C(t, br(8)) &= CT(2,0) + n_3 C_{rep}(T_0).
\end{aligned}
$$

We will compare the branches $br(3)$, $br(5)$ and $br(7)$, as well as the branches $br(4)$, $br(6)$ and $br(8)$. Once again we put conditions on $CT(2,0)$, $CT(2,1)$ and $CT(2,2)$, in order to be able to conclude on the costly programs. Below, we deal with only one case:

$$CT(2,1) = \min(CT(2,0), CT(2,1), CT(2,2)).$$

When comparing the total costs of $br(3)$, $br(5)$ and $br(7)$, we obtain

$$C(t, br(3)) < C(t, br(5)) \qquad \text{and} \qquad C(t, br(3)) < C(t, br(7)).$$

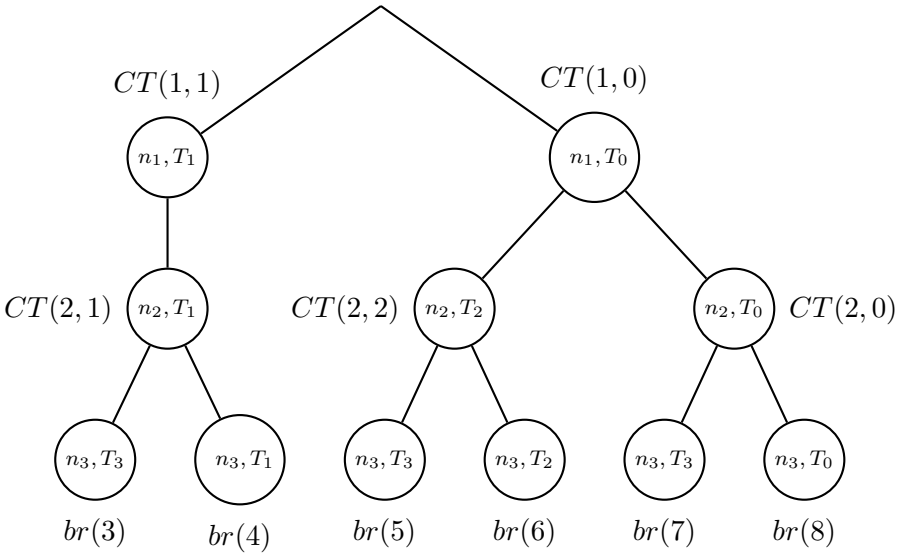

Figure 7: Search tree when $CT(1,1) > CT(1,0)$ with labels $CT(2,0)$, $CT(2,1)$, $CT(2,2)$.

Thus, $br(5)$ and $br(7)$ are deleted, which is the same as removing the nodes (n_3, T_3), which are the offspring of (n_2, T_2) and (n_2, T_0). Only one node (n_3, T_3) is left at the third generation. Since $C_{rep}(T_2) < C_{rep}(T_1) < C_{rep}(T_0)$, we have

$$C(t, br(4)) < C(t, br(8)).$$

Thus, we delete $br(8)$ and hence the node (n_2, T_0), since we have previously removed the node (n_3, T_3) (the descendant of (n_2, T_0)). Since $C_{rep}(T_1) > C_{rep}(T_2)$, we cannot make a conclusion about $br(6)$. Indeed,

$$C(t, br(4)) - C(t, br(6)) = \underbrace{CT(2,1) - CT(2,2)}_{<0} + n_3 \underbrace{(C_{rep}(T_1) - C_{rep}(T_2))}_{>0},$$

so we keep $br(6)$. Under the condition $CT(2,1) = \min(CT(2,0), CT(2,1), CT(2,2))$, the most expensive branches $br(5)$, $br(7)$ and $br(8)$, have been deleted. Thus, there remain only three PM schedules instead of eight, as depicted in Figure 8.

The last stage consists in determining the optimal solution for the fixed inspection time $t = T^*$. In order to do this, it suffices to evaluate the nodes of the last generation by adding to $CT(2,1)$, firstly $n_3 C_{rep}(T_3) + C_{out}(T_3)$ (yielding the total cost of $br(3)$), and secondly $n_3 C_{rep}(T_1)$ (giving the total cost of $br(4)$). To get the total cost of $br(6)$, we add $n_3 C_{rep}(T_2)$ to $CT(2,2)$. The minimum value returns the optimum PM plan for the fixed inspection time $t = T^*$.

Remark 4.1. *Figure 8 gives* $S(3) = \{1, 2, 3\}$.

Remark 4.2. *The third generation has been built with*

1. *The two offspring of the node* (n_2, T_1) *(the node that gives the minimum of* $CT(2,i)$*,* $i = 0, 1, 2$*):* (n_3, T_3) *and* (n_3, T_1).

2. *The offspring of the node* (n_2, T_2) *(which satisfies* $C_{rep}(T_2) < C_{rep}(T_1)$*):* (n_3, T_2).

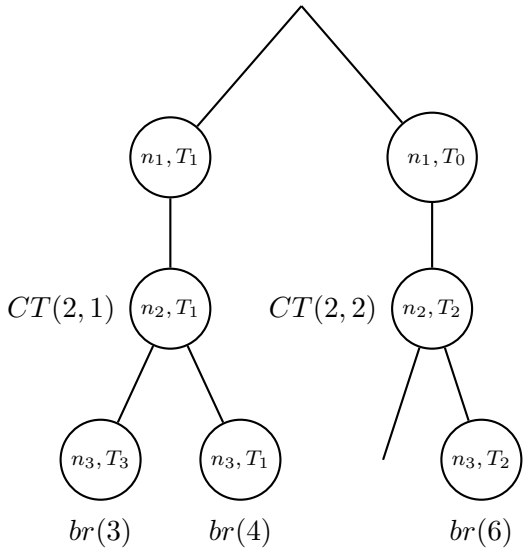

Figure 8: Final tree when $CT(1,1) > CT(1,0)$ and $CT(2,1) = \min\{CT(2,0), CT(2,1), CT(2,2)\}$.

Remark 4.3. *At each generation i, we build at most $(i+1)$ nodes (n_i, T_j) for $j = 0, \ldots, i$. For example, Figure 6 shows that the second generation is composed of nodes (n_2, T_0), (n_2, T_1) and (n_2, T_2), and Figure 8 shows that the third generation has been built with nodes (n_3, T_1), (n_3, T_2) and (n_3, T_3).*

Let us summarize the building strategy related to the example introduced in Section 4.1. We have initially fixed the inspection time as $t = T^*$, and we wish to build a tree with three generations, since $N_t = 3$. The first generation $\{(n_1, T_1), (n_1, T_0)\}$ is constructed and evaluated with $CT(1,1)$ and $CT(1,0)$. The node with the lowest cost generates its two children. The other node generates its descendant, provided that its repair cost calculated at its deadline is less than the repair cost calculated at the deadline of the least expensive node. The result of this first step is the second generation. At the end of the second stage, the third generation is built in the same manner as before, and gives the final tree containing three generation. The aim of the third and final stage is to determine the optimal repair plan for the given inspection time; for this, it suffices to evaluate the nodes at the last generation. The branch that returns the least expensive cost is the best repair schedule. Recall that this PM plan is not necessarily the global optimum, it could be a local minimum. In order to obtain the global optimum, we have to build trees related to inspection times $T_2 - 1$ and $T_3 - 1$ (the case $t = T_1 - 1$ is trivial because $N_t = 0$, i.e., there are no repairs, hence no tree). Each of such tree returns the best repair program. To get the (global) PM schedule, we have to compare the total costs of these best repair plans. As a result, with our algorithm, pipeline managers have the opportunity to choose one of these PM schedules. If they choose to inspect at year T^*, the algorithm will output the least expensive repair plan for this choice. The general algorithm is presented in the next section.

4.2 Algorithm

The following algorithm generalizes the previous example. It provides a search tree with $j - 1$ generations for a fixed inspection time $t = T_j - 1$. The construction of the remaining trees follows the same algorithm. At the end, we obtain $N + 1$ PM schedules, i.e., an inspection time together with the best related repair plan; the global minimum is obtained by comparing the total costs of these $N + 1$ repair plans.

Result: Optimal PM schedule for an inspection at year $t = T_j - 1$.
Initialization: Construction of the first generation $\{(n_1, T_1), (n_1, T_0)\}$;
Evaluation of partial repair programs $CT(1,1)$ and $CT(1,0)$;
Step 1: second generation;
if $\boldsymbol{CT(1,1) < CT(1,0)}$ **then**
  Both children of node (n_1, T_1), i.e., $\{(n_2, T_2), (n_2, T_1)\}$ are constructed;
  if $C_{rep}(T_0) > C_{rep}(T_1)$ **then**
    it is not necessary to build the offspring of (n_1, T_0)
  end
  The second generation is built;
else
  Both descendants of (n_1, T_0), i.e., $\{(n_2, T_2), (n_2, T_0)\}$ are built;
  if $C_{rep}(T_0) > C_{rep}(T_1)$ **then**
    the descendant of (n_1, T_1), i.e., (n_2, T_1), is built;
  end
  The second generation is built;
end
Step i: $(i+1)$th generation, $1 < i < j-1$;
The ith generation was built at the previous step: $\{(n_i, T_l), l \in S(i)\}$;
if $\boldsymbol{CT(i, \tilde{l}) = \min\{CT(l), l \in S(i)\}}$ **then**
  Both children of node $(n_1, T_{\tilde{l}})$ are constructed, i.e., $\{(n_{i+1}, T_{i+1}), (n_{i+1}, T_{\tilde{l}})\}$;
  while $\boldsymbol{C_{rep}(T_k) < C_{rep}(T_{\tilde{l}})}$, $\boldsymbol{k \in S(i) - \{\tilde{l}\}}$ **do**
    Construction of the descendant of (n_k, T_k), i.e.,(n_{i+1}, T_k)
  end
  The $(i+1)$th generation is built;
end
Step $j-1$: Evaluation of full repair plans;
The best PM schedule for an inspection at $t = T_j - 1$ is given.

Algorithm 1: Construction of the tree with $j-1$ generations.

This algorithm builds and estimates the costs of at most $(N+1)(N+2)/2 - 1$ repair schedules. Indeed, if $t = T_1 - 1$, the optimal program is that where there is no repair. In this case, the total cost coincides with the inspection cost evaluated at time t. If $t = T_2 - 1$, the algorithm builds and estimates the costs of at most two branches, and returns the best repair schedule, $b^*(T_2 - 1)$. If $t = T^*$, the algorithm builds and estimates the costs at most $N + 1$ branches, and returns the best repair schedule, $b^*(T^*)$. Thus, the optimal maintenance program $b^*(t^*)$ is the best among $\{b^*(T_2 - 1), \ldots, b^*(T^*).\}$

5 Computational results

In this section, we present a large number of examples to confirm the efficiency of the proposed algorithm. The characteristics of the 11 examples are shown in Table 1. The column labeled with N corresponds to the number of distinct deadlines, the third column designates the deadlines, and the last column the numbers of defects associated with each deadline. In other words this table gives 11 *primary repair schedules* defined in Section 2 (Definition 1). For example, the the fifth experience is composed of 7 different deadlines. The first four are $T_1 = 2$, $T_2 = 5$, $T_3 = 8$ and $T_4 = 15$. At year $T_1 = 2$ there is 1 repairs; at year $T_2 = 5$ there is 1 repair; at year $T_3 = 8$, 1 and at year $T_4 = 15$, 1.

Hence the primary is given by

$$\mathscr{P} = \{(n_1 = 2, T_1 = 2), (n_2 = 1, T_2 = 5), (n_3 = 1, T_3 = 8), (n_4 = 1, T_4 = 15)$$
$$(n_5 = 6, T_5 = 24), (n_6 = 5, T_6 = 26), (n_7 = 4, T_7 = 28)\}\,.$$

For the above primary repair schedule, the PM problem aims at finding the next (after the primary inspection) optimal inspection t^* within

$$\{T_j - 1,\ j = 1, \ldots, N, T^*\} = \{1, 4, 7, 15, 24, 25, 27, 30\}\,,$$

and the associated repair schedule minimizing the operational cost within Δt^*; here 30 designates the horizon time T^*. For this experience, our algorithm built at most $35 = (7+1)(7+2)/2 - 1$ repairs schedule and return 7 maintenance programme; one for each epoch: $4, 7, 15, 24, 25, 27,$ and 30.

We want to look at the computational effort required to solve the problem (8) when the number of deadlines N increases. To do this, we will compare our algorithm, which we call *tree algorithm*, with two other algorithms. The first, called the *simplified algorithm*, solves (8) and looks through complete repair plans when candidates for the next inspection are in $\{T_1 - 1, \ldots, T_N - 1, T^*\}$. According to Section 4.2, this method investigates $(1 - 2^{N+1})/(1 - 2) - 1$ schedules. The second one, called the *comprehensive algorithm*, solves the same problem but investigates 2^{N_t} schedules for each $t \in \{1, 2, \ldots, T^*\}$. This method provides an inefficient but complete set of PM schedules (inspection times and repair plans) which will serve as a benchmark for our algorithm and the simplified one. Both algorithms (simplified and comprehensive) output the best repair schedule for each inspection time $t \in \{T_1 - 1, \ldots, T_N - 1, T^*\}$ (and especially, the optimal PM plan $(t^*, \mathscr{P}_t^*)$).

The experimental design consists of entering manually for the three algorithms, the number of different deadlines N, the values of deadlines, and the number of defects observed for each deadline. The parameters T^*, discount rate, and inflation rate, are also entered manually and respectively set as $T^* = 30$, $r_d = 8\%$, $r_i = 1\%$. The full costs are expressed in $k€$ and are also manually entered. The initial inspection cost is $C^0_{insp} = 500$, the initial repair cost is $C^0_{rep} = 60$, and the initial out-of-service cost is $C^0_{out} = 300$. We compare the performance of our algorithm with the comprehensive and simplified algorithms. The computational tests were built with scilab 5.4.1 on a SONY computer with biprocessor, 2.30 GHz and 1 GB of RAM. The three algorithms were developed in a decision support framework. The tools provide a set of suitable solutions. Each solution is defined by a repair schedule, an inspection time belonging to $\{T_j - 1, j = 1, \ldots, N\} \cup \{T^*\}$, and a total cost. Table 2 illustrates the results. It compares the running times (indicated by the CPU) and the optimal cost of the all 11 experiments for each algorithm.

When we compare the outputs of our algorithm with both the comprehensive and simplified algorithms, we see that all the algorithms return the same optimal cost for each experience, whereas the running time is much less than for the others. Our algorithm was developed under a support decision framework. We have seen that its results can help a pipeline manager, in the sense that the algorithm provides a set of solutions that includes the optimal PM schedule. Table 3 below shows for $N = 7$, all PM schedules, i.e., all repair schedules associated with $t \in \{T_j - 1, j = 1, \ldots, 7\} \cup \{T^*\}$. The first column corresponds to the distinct inspection times, the second to the associated repair plans, and the last represents the total cost of these repair schedules within the inspection interval. We see that the optimal PM schedule suggests an inspection at year 23, for a total cost of 347.05704 $k€$. The tool gives then to the pipeline manager the following repair program within the inspection interval $]0, 23]$

$$\{(4, T_0), (0, T_1), (0, T_2), (0, T_3), (0, T_4)\}.$$

This repair schedule suggests to the practitioner to repair 4 defect during the primary inspection (at $T_0 = 0$)) and no repair at years $T_1 = 2$, $T_2 = 5$, $T_3 = 8$ and $T_4 = 15$. If the operator is not satisfied with this PM schedule, he has the opportunity to select another schedule. For example, he may choose to inspect later at year 27. In this case, the tool proposes a repair schedule with a total cost equal to 514.11211 $k€$, and the repair plan is given by

$$\{(4, T_0), (0, T_1), (0, T_2), (0, T_3), (0, T_4), (11, T_5), (0, T_6)\},$$

which means that 4 defect must be repair during the primary inspection and 11 repairs should be made at year $T_5 = 24$.

Example N^o	N	Deadlines T_j, $j = 1, \ldots N$	Number of defects
1	3	1,8,16	1,2,1
2	4	4,6,12,22	2,3,1,1
3	5	2,3,8,12,24	1,1,3,2,2
4	6	5,6,8,15,19,25	1,1,3,3,2,1
5	7	2,5,8,15,24,26,28	1,1,1,1,6,5,4
6	8	4,7,8,11,13,21,25,27	1,1,2,1,1,3,3,1
7	9	5,6,8,11,14,20,21,25,26	1,2,3,2,1,1,3,2,1
8	10	3,5,6,7,13,18,20,22,25,26	1,2,1,2,3,3,1,1,1,1
9	11	2,4,5,6,10,12,16,17,20,22,25	1,2,3,1,4,1,1,1,2,2,3
10	12	2,4,5,6,7,9,10,11,18,20,24,26	1,1,2,3,2,2,1,1,1,3,2,1
11	13	2,5,6,7,10,12,13,17,18,20,21,24,26	1,1,3,2,2,1,1,3,4,1,4,5,2

Table 1: Data for simulated examples.

N	Comprehensive algorithm		Simplified Algorithm		Tree Algorithm	
	CPU	Optimal cost	CPU	Optimal cost	CPU	Optimal Cost (€)
3	0.60	306972.81	0.078	306972.81	0.0156	306972.81
4	1.61	408943.08	0.22	408943.08	0.0156	408943.08
5	3.26	432790.34	0.44	432790.34	0.0312	432790.34
6	9.39	382437.51	1.37	382437.51	0.0312	382437.51
7	24.60	347057.04	3.88	347057.04	0.0468	347057.04
8	54.40	394468.88	9.42	394468.88	0.0624	394468.88
9	123.27	382437.51	24.08	382437.51	0.0624	382437.51
10	272.24	437285.67	57.24	437285.67	0.0780	437285.67
11	545.78	467592.59	124.16	467592.59	0.0780	467592.59
12	1138.46	467592.59	278.82	467592.59	0.1092	467592.59
13	2656.50	442437.51	725.95	442437.51	0.1248	442437.51

Table 2: Computational results.

Inspection time	Repair within Δt plans	Total cost
30	$\{(4, T_0), (0, T_1), (0, T_2), (0, T_3), (0, T_4), (15, T_5), (0, T_6), (0, T_7)\}$	547256.39
27	$\{(4, T_0), (0, T_1), (0, T_2), (0, T_3), (0, T_4), (11, T_5), (0, T_6)\}$	514112.11
25	$\{(4, T_0), (0, T_1), (0, T_2), (0, T_3), (0, T_4), (6, T_5)\}$	465784.98
23	$\{(4, T_0), (0, T_1), (0, T_2), (0, T_3), (0, T_4)\}$	347057.04
14	$\{(53, T_0), (0, T_1), (0, T_2), (0, T_3)\}$	375675.59
7	$\{(2, T_0), (0, T_1), (0, T_2)\}$	432790.34
4	$\{(1, T_0), (0, T_1)\}$	442437.51
1	$\{(0, T_0)\}$	467.59259

Table 3: Data for simulated examples when $N = 7$.

6 Conclusion

This paper focused on the PM problem in gas pipelines from the economics point of view. We have modeled this problem using binary integer nonlinear programming, which drastically shrinks the possible set of good repair schedules, by exploiting the problem's properties. In order to solve the problem, we have proposed an algorithm, based on dynamic programming, which finds the exact solution extremely quickly, along with a set of alternative solutions. As a result, managers of gas pipeline systems can consider various possible schedules, and choose alternative PM programs, and our algorithm assists them in making the most suitable decision in a short period of time.

References

[1] Salvador Perez Canto. Application of benders' decomposition to power plant preventive maintenance scheduling. *European Journal of Operational Research*, 184(2):759–777, 2008.

[2] Ching-Chin Chern, Pei-Szu Chien, and Shu-Yi Chen. A heuristic algorithm for the hospital health examination scheduling problem. *Eur. J. Oper. Res.*, 186(3):1137–1157, 2008.

[3] Harish Garg, Monica Rani, and S.P. Sharma. Preventive maintenance scheduling of the pulping unit in a paper plant. *Japan Journal of Industrial and Applied Mathematics*, 30(2):397–414, 2013.

[4] Laith A. Hadidi, Umar M. Al-Turki, and M. Abdur Rahim. Joint job scheduling and preventive maintenance on a single machine. *International Journal of Operational Research*, 13(2):174–184, 2012.

[5] Han Ping Hong. Inspection and maintenance planning of pipeline under external corrosion considering generation of new defects. *Structural Safety*, 21(3):203–222, 1999.

[6] Seong-Jong Joo. Scheduling preventive maintenance for modular designed components: A dynamic approach. *European Journal of Operational Research*, 192(2):512 – 520, 2009.

[7] M.J. Kallen and J.M. van Noortwijk. Optimal maintenance decisions under imperfect inspection. *Reliability Engineering & System Safety*, 90(2 â€“ 3):177 – 185, 2005.

[8] Ahmet B. Keha, Ketan Khowala, and John W. Fowler. Mixed integer programming formulations for single machine scheduling problems. *Computers & Industrial Engineering*, 56(1):357 – 367, 2009.

[9] Nicole Megow, Rolf H. Möhring, and Jens Schulz. Decision support and optimization in shutdown and turnaround scheduling. *INFORMS J. on Computing*, 23(2):189–204, April 2011.

[10] Salvador Perez Canto. Using 0/1 mixed integer linear programming to solve a reliability-centered problem of power plant preventive maintenance scheduling. *Optimization and Engineering*, 12(3):333–347, 2011.

[11] Katja Schimmelpfeng, Stefan Helber, and Steffen Kasper. Decision support for rehabilitation hospital scheduling. *OR Spectr.*, 34(2):461–489, April 2012.

[12] Yong Sun, Lin Ma, and Jon Morris. A practical approach for reliability prediction of pipeline systems. *European Journal of Operational Research*, 198(1):210–214, 2009.

[13] Silvija Vlah Jerić and José Rui Figueira. Multi-objective scheduling and a resource allocation problem in hospitals. *J. of Scheduling*, 15(5):513–535, October 2012.

[14] Wenxing Zhou. System reliability of corroding pipelines. *International Journal of Pressure Vessels and Piping*, 87(10):587 – 595, 2010.