Prognostics-based Scheduling to Extend a Distributed Platform Production Horizon under Service Constraint: Model, Complexity and Resolution.

Nathalie Herr, Jean-Marc Nicod, Christophe Varnier

INSTITUT FEMTO-ST

UMR CNRS 6174

Prognostics-based Scheduling to Extend a Distributed Platform Production Horizon under Service Constraint: Model, Complexity and Resolution

Version 1

Nathalie Herr — Jean-Marc Nicod — Christophe Varnier

Rapport de Recherche n° RR–FEMTO-ST–2577

DÉPARTEMENT AS2M – June 12, 2014

Prognostics-based Scheduling to Extend a Distributed Platform Production Horizon under Service Constraint: Model, Complexity and Resolution

Version 1

Nathalie Herr , Jean-Marc Nicod , Christophe Varnier

Département AS2M

PHM

Rapport de Recherche no RR –FEMTO-ST–2577 June 12, 2014 (43 pages)

Abstract: In the field of production scheduling, this paper addresses the problem of optimizing the useful life of a heterogeneous distributed platform composed of identical parallel machines and which has to provide a given production service. Each machine is supposed to be able to provide several throughputs corresponding to different operating conditions. The purpose is to provide a production scheduling that maximizes the production horizon. The use of Prognostics and Health Management (PHM) results in the form of Remaining Useful Life (*RUL*) allows to adapt the schedule to the wear and tear of machines. This work comes within the scope of Prognostics Decision Making (DM). The key point is to configure the platform, i.e., to select the appropriate profile for each machine during the whole production horizon so as to reach a total throughput based on a customer demand as long as possible.

In the homogeneous case, the Longest Remaining Useful Life first algorithm (LRUL) is proposed to find a solution and its optimality is proven. The NP-Completeness of the general case is then shown. A Binary Integer Linear Programming (BILP) model which allows to find optimal solutions for fixed time horizons has been defined. As solving such a BILP is NP-Complete, solutions can however be computed in reasonable time only for small size instances of the problem. Many heuristics are then proposed to cope with large scale decision problems and are compared through simulation results. Exhaustive simulations assess the efficiency of these heuristics. Distance to the theoretical maximal value comes indeed close to 5% for the most efficient ones.

Key-words: Sheduling, Parallel machines, Preventive Maintenance, Remaining Useful Life, Prognostics Decision Making, Integer Linear Programming, Optimal Solution, Heuristics

FEMTO-ST Institute, AS2M research department
TEMIS, 24 rue Alain Savary, F-25000 BESANÇON FRANCE
Tél : (33 3) 81 40 28 01 – Fax : (33 3) 81 40 28 09 – e-mail : isabelle.gabet@ens2m.fr

Ordonnancement adaptatif de tâches basé sur le pronostic pour l'optimisation de l'horizon de production sous contraintes

Version 1

Résumé :

Mots-clés : Ordonnancement, Machines parallèles, Maintenance prévisionnelle, Durée de vie résiduelle, Décision, Programme linéaire en nombres entiers, Solutions optimale, Heuristiques

FEMTO-ST Institute, AS2M research department
TEMIS, 24 rue Alain Savary, F-25000 BESANÇON FRANCE
Tél : (33 3) 81 40 28 01 – Fax : (33 3) 81 40 28 09 – e-mail : isabelle.gabet@ens2m.fr

Prognostics-based Scheduling to Extend a Distributed Platform Production Horizon under Service Constraint: Model, Complexity and Resolution

Nathalie Herr , Jean-Marc Nicod , Christophe Varnier

June 12, 2014

Abstract

In the field of production scheduling, this paper addresses the problem of optimizing the useful life of a heterogeneous distributed platform composed of identical parallel machines and which has to provide a given production service. Each machine is supposed to be able to provide several throughputs corresponding to different operating conditions. The purpose is to provide a production scheduling that maximizes the production horizon. The use of Prognostics and Health Management (PHM) results in the form of Remaining Useful Life (RUL) allows to adapt the schedule to the wear and tear of machines. This work comes within the scope of Prognostics Decision Making (DM). The key point is to configure the platform, i.e., to select the appropriate profile for each machine during the whole production horizon so as to reach a total throughput based on a customer demand as long as possible.

In the homogeneous case, the Longest Remaining Useful Life first algorithm (LRUL) is proposed to find a solution and its optimality is proven. The NP-Completeness of the general case is then shown. A Binary Integer Linear Programming (BILP) model which allows to find optimal solutions for fixed time horizons has been defined. As solving such a BILP is NP-Complete, solutions can however be computed in reasonable time only for small size instances of the problem. Many heuristics are then proposed to cope with large scale decision problems and are compared through simulation results. Exhaustive simulations assess the efficiency of these heuristics. Distance to the theoretical maximal value comes indeed close to 5% for the most efficient ones.

1 Introduction

An impressive amount of work has been done in the field of production scheduling. The most common cases studied are single machine scheduling, identical, uniform and unrelated parallel machine scheduling and open shop, flow shop and job shop scheduling [14]. As far as a lot of different hypothesis can be taken into account and combined in different ways, the number of possibilities is very huge. The problem tackled in this report concerns the scheduling of a platform composed of many heterogeneous parallel machines, performing independent and identical tasks. All the machines are supposed to be independent and of similar type. At each time, the platform has to deliver a given global throughput based on a customer demand. The total provided throughput is determined by the sum of each throughput of machines that are currently running. The platform can be seen as a distributed environment where machines fulfill a shared global task. The purpose is to manage the platform and implement a production schedule which allows to provide at least a given service level requested by consumers as long as possible.

All the machines are not supposed to be in use at any time because the target throughput can be reached by using only a subset of the machines within the platform or because some

machines are not available. In literature on scheduling theory, machines are commonly assumed to be continuously available during the whole considered production horizon [25, 19]. This assumption may not be valid in a real production situation due to wear and tear on machines which involves shutdowns due to breakdowns or maintenance operations [29]. In the considered problem, as each machine is assumed to be independent, the breakdown of one of them does not necessarily entail a shutdown of the whole platform. Maintenance is nevertheless required in the long term. Guarantying the availability of production systems is although an important requirement, especially in the power generation domain [8]. Shutdown periods can be minimized by gathering maintenance operations. This allows to reduce the costs due to the use of material and human resources and to production shutdown periods as well. To manage such a grouping, some maintenance actions might need to be postponed. Dietl et al. [11] proposed for instance to match the time to failure of different tools used in each station of a transfer line by derating them in such a way that a maximum of tools can be maintained at the same time. Grouping maintenance actions can also be necessary because maintenance is challenging. For instance, many works have been carried out to optimize the maintenance of wind farms [22, 4]. For the maintenance of such systems, especially offshore ones, complex aspects like weather conditions, requirement of non-traditional resources, skilled technicians, expensive hired services or spare parts have indeed to be considered. Kovacs et al. [22] proposed a mixed-integer programming formulation for the problem of optimizing the scheduling of maintenance actions for wind farms. Minimization of maintenance costs has been studied by Besnard et al. [4] who proposed an opportunistic maintenance optimization model for offshore wind power systems.

It is assumed that the platform can be totally shutdowned for maintenance and that the needed service is provided by an other platform when maintenance is performed. All the machines can in that case be maintained in the same time. The objective is then to maximize the production horizon of the whole platform between two maintenance periods. The key point is to be able to take the wear and tear of machines into consideration in the scheduling process. Prognostics and Health Management (PHM) can comply with these needs in that it can help to know the time left before occurrence of a failure. Its Prognostics phase is indeed dedicated to estimate the Remaining Useful Life (RUL) of machines in service [28, 23]. The use of PHM results is furthermore consistent with our objectives in that PHM aims at maintaining equipment operational performance over time, improving their usage and increasing their availability while avoiding failures and minimizing maintenance costs [5, 16].

The organisation of the report is as follows: Section 2 discusses related work. The tackled problem is detailed in Section 3 and is illustrated through a motivating example in Section 4. An optimal resolution and complexity results are then provided in Section 5. An optimal approach using a BILP is described in Section 6 and sub-optimal solutions are proposed in Section 7. The provided heuristics are then compared through simulation results (Section 8). This work is finally concluded in Section 9.

2 Related work

As pointed out by Haddad et al. [16], PHM has been shown to provide many benefits for the health management of systems such as avoiding failures, increasing availability, minimizing loss of remaining life, optimizing resource usage or reducting no-fault-founds. These benefits are strongly tied to the decision part of PHM process whose main purpose is to determine appropriate maintenance actions in response to prognostics predictions [18, 1]. The post-prognostics decision process concentrates appropriate decisions onto one equipment whereas Prognostic Decision Making (PDM) extends decisions to a whole system. Prognostic Decision Making aims also at choosing an appropriate system configuration [2]. Our work falls within this latter case.

Temporal segmentation for decision framework has been introduced by Bonissone et al. [5]. They identified three types of decisions in the segment dealing with multiple and repeated decisions: tactical, operational and strategic. According to the frequency on which the decisions have to be taken, diagnostics and prognostics fit with tactical level (seconds, minutes, hours). Decisions for process control in such timeframes suits with on-line scheduling and rescheduling. The part that concerns frequency from nanoseconds to seconds describes configurations that are encountered in electronic, electro-mechanical and control domains. Operational level is adapted to lower frequency decision process as production or maintenance planning and off-line scheduling. Our work falls within these two short-term and mid-term levels of the decision making process, in which many applications are studied. We can cite the aerospace domain [2, 7] and applications on wind turbines [16], electronic systems [28] or cutting tools [6].

Most of the studies proposed in the literature focus on maintenance planification. PHM enables indeed maintenance to be planned on the basis of actual component or system health state [7]. Many contributions are proposed in the form of maintenance policies that minimize life cycle costs. Sandborn et al. [28] endeavor to determine when scheduled maintenance makes sense for electronic systems. Haddad et al. [16] proposed an optimization consisting in finding an optimum subset of offshore turbines to be maintained, given information on their degradation, availability requirement and costs constraints. Balaban et al. [2] developed a prototype algorithm that uses probabilistic methods and prognostics information in generation of action policies for aerospace applications. In the same area, a PHM and Maintenance data integration tool that enables various available diagnostic and prognostics methods to be used in a real environment has been proposed by Camci et al. [7] for fighter aircrafts. Asmai et al. [1] used the data-driven approach to implement an intelligent maintenance prognosis tool. Incorporated into the maintenance decision process, this tool can be used to recommend better maintenance planning. In the same paper, it is pointed out that acknowledging the *RUL* information can also be very useful for production scheduling. Indeed, this quantity gives information about the status of equipment before proceeding with new production jobs. This can help avoiding material waste and production loss due to equipment breakdown in the middle of an operation. Decisions could therefore take several forms: immediate machine shutdown in order to avoid further damage, machine operation modification in such a way as to reduce the load, continuation of normal operation [16], preventive intervention, production rescheduling, etc. The use of prognostics results in the form of *RUL* can then be extended to modify operational conditions or mission profiles in order to accomplish the main objectives of the mission [23, 20]. Balaban et al. [3] proposed such an application on a hardware testbed based on a planetary rover platform and considering many fault modes such as mechanical deterioration, electronic faults or low remaining battery charge. The objective is not only to determine the *RUL* of a component, but also to suggest actions that can optimize vehicle maintenance, ensure mission safety, or extend mission duration. The idea that is conveyed is the following: if decisions are made with respect to the system health evolution over time, the mission effectiveness can be maximized before energy and health budgets are exceeded. In case of a fault occurrence, a new mission plan may have to be defined. Reconfiguration of the vehicle can also be considered in order to extend the *RUL* of the affected component as long as needed to ensure achievement of the mission objectives.

Such kind of reconfiguration that affects the system production rate can be found in scheduling literature. Variable-speed scheduling is for instance a generalization of standard multiple machine scheduling because not only the assignment of jobs to machines has to be managed, but also the time used by jobs on machines [32, 24, 31]. Tooling machines are for example variable-speed machines, insofar as they can be run at different speeds [11]. The notion of reconfiguration can also be found in the field of scientific computing and more precisely in scheduling of multi-

processor tasks. Processors capable of global Dynamic Voltage and Frequency Scaling (DVFS) have been developed and allow the manipulation of the voltage and frequency when the computational load is not perfectly balanced [30, 21]. Many applications have been proposed in the literature [27, 9, 33]. Three main objectives can be highlighted among these papers using reconfiguration. The first one is the makespan, i.e., total length of the schedule, minimization [32, 24, 27]. The second one is the minimization of energy consumption [30, 21, 9, 33]. This objective is linked with the third one that consists in minimizing production costs [32, 30].

The objective set in this paper can not be classified in these three categories. The point is to configure a set of machines so as to maximize the production horizon. A second objective is to use all the considered machines to their full potential in order to minimize maintenance costs by grouping maintenance operations. Scheduling which is taken into consideration differs furthermore from the general definition. We seek indeed to schedule considering prognostics information. We consider prognostics-based scheduling, which can be defined as a scheduling that takes the wear and tear of equipments into account and that adapts to remaining useful life (RUL). Scheduling appears then to be part of the PHM Decision Process, as far as prognostics results are used to determine the length of time intervals between two maintenance operations. Prognostics-based scheduling complies with main goal of scheduling that is achieving an optimal usage of resources. Such a kind of scheduling could furthermore be adaptive, as it may respond to disruptions or to knowledge of new informations dynamically over time [10].

3 Problem statement

3.1 Framework

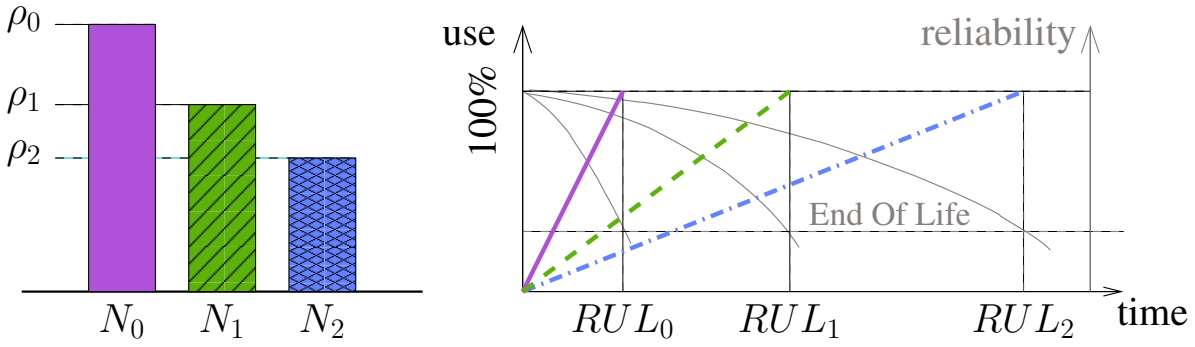
The application that is addressed in this paper is based on a platform composed of a set $\mathscr{M}$ of m machines M_j, with $j \in J = \{1, \ldots, m\}$ ($\mathscr{M} = \{M_1, \ldots, M_m\}$), performing independent and identical tasks. All the machines can be used in parallel as a global system. Machines are supposed to be always supplied with power or raw material required for the production. The provided result is a given service level that is measured as a throughput, i.e., number of pieces performed or amount of matter (a) treated per unit of time (ut). At each time the global throughput ρ_{tot} provided by the platform is determined by the sum of each throughput of machines M_j that are currently running. Note that the platform has to deliver a given global throughput $\sigma = \sigma(t)$. This latter one is based on a customer demand, which can be variable and defined as a function of time. The platform can be seen as a distributed environment where machines that are currently running fulfill a shared global task such that $\rho_{tot} \geq \sigma$.

3.2 Controlled running profiles

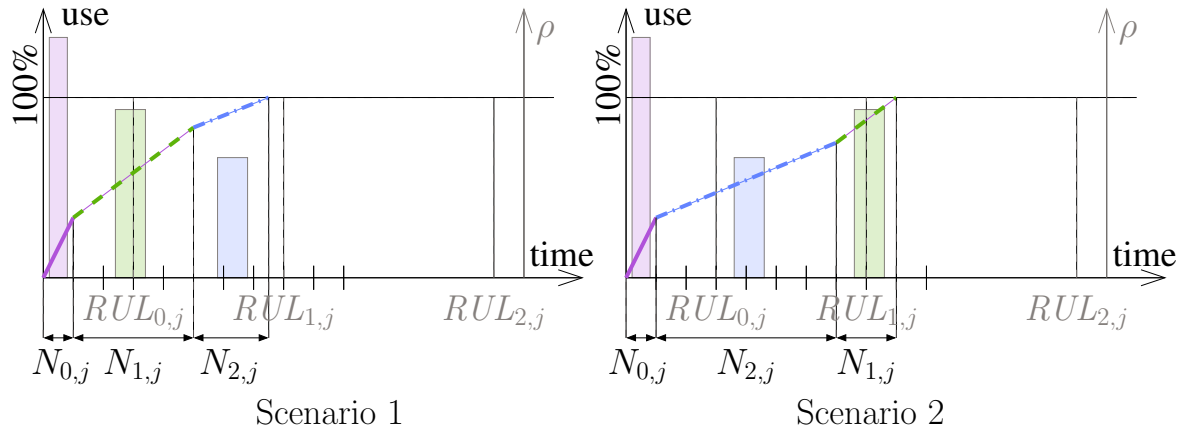
As previously developed, the performance of a machine may vary during its use and this variation can be controlled, for instance through voltage, power or speed scaling. We propose to exploit this characteristic to optimize the use of the considered platform. Each machine is supposed to be able to provide several throughputs. In a PHM context, each throughput corresponds to a certain operating condition. It is moreover assumed that each machine is monitored and associated with a prognostics module that gives a RUL value depending on both its past and its future usage. As highlighted by Elghazel and al. [12], the way to consider operating conditions, especially future ones in RUL estimation, still needs deep studying. So we assume in this paper that each needed RUL value is known and that RUL evolution depends on the operating conditions, that is on the running profile in which the machine is used.

We define a running profile as a controlled machine profile involving a certain throughput and associated with a certain RUL. n running profiles are defined for each machine M_j: $N_{i,j} =$

$(\rho_{i,j}, RUL_{i,j})$, with $i \in I = \{0, \ldots, n-1\}$. Let $N_{0,j}$ be the nominal running profile of machine M_j, where immediate throughput $\rho_{0,j}$ and output $Q_{0,j} = \rho_{0,j} \times RUL_{0,j}$ are the most significant. This nominal running profile has the minimum RUL. By comparison, a sub-nominal profile provides a lowest throughput, but its associated RUL is longer (see figure 1(a)) such that $Q_{0,j} > Q_{1,j} > \ldots > Q_{n-1,j}$ with $\rho_{0,j} > \rho_{1,j} > \ldots > \rho_{n-1,j}$ and $RUL_{0,j} < RUL_{1,j} < \ldots < RUL_{n-1,j}$. Each running profile corresponds to an operating condition and impacts differently the wear and tear of the machine and therefore its operational time. Considering several running profiles seems to be interesting in that the combination of two or more running profiles allows to reach an operational time that is greater than when considering only the nominal running profile $N_{0,j}$. Without taking efficiency into account, Figure 1(b) shows that it is possible to run a machine M_j for longer than the RUL of the nominal profile $RUL_{0,j}$ by using three different running profiles $N_{0,j}$, $N_{1,j}$ and $N_{2,j}$. Of course, in counter part, the amount of work done with this machine in both proposed scenarios is lower than it would be with the nominal profile only.



(a) Characteristics of running modes



(b) Use of several running modes

Figure 1: Running modes

It is assumed that the RUL evolution is not impacted by the order of the selected $N_{i,j}$ among the machine lifespan. The second scenario proposed in Figure 1(b) shows that an appropriate selection of $N_{i,j}$ allows to extend the useful life even beyond $RUL_{1,j}$. As showed in the motivating example in Section 4, one can take advantage of the use of many running profiles to optimize the scheduling of a set of machines.

3.3 Decision problem

The problem tackled in this paper is the optimization of the useful life of a platform such as defined in the framework (see Section 3.1). The objective is to provide a prognostics-based scheduling as defined in Section 2 by configuring the platform so as to reach the demand as long as possible. All the machines are not supposed to be in use at any time because of their RUL or because the target throughput σ can be achieved by using only a subset of the available machines within the platform. RUL is assumed to be constant in time when the machine is not used. It is moreover assumed that overproduction should be avoided as far as possible. Overproduction leads indeed either to costly stocks or to losses if the production can not be stored. Allowing overproduction can however allow to extend a platform useful life (see Scenario *S2* in Section 4). The key point is then to be able to find the appropriate configuration for each machine during its lifespan.

One way to tackle the problem consists in discretizing the time into periods ΔT. This approach is not so far from realistic constraints, since one can imagine that one period could be one day or one week in a real case. The production horizon $\mathcal{T}$ can then be expressed as follows: $\mathcal{T} = K \times \Delta T$, with ΔT the length of one time period and K the number of periods for which the demand level σ is reached. If the demand σ is a function of time $\sigma(t)$, we assume that $\sigma(t) = \sigma_k$ is a constant value within the period k for all t and all k such that $(k-1)\Delta T < t \leq k\Delta T$ and $1 \leq k \leq K$.

Considering discretized time, the problem consists in choosing, for each period of time k, a subset of machines to be used and an associated running profile for each of them. Using the notations defined in this section, the problem tackled here can be described by the following notational form: $\text{MAXK}(\sigma_k \,|\, \rho_{i,j} \,|\, RUL_{i,j})$. This general notation stands for the problem of finding a schedule that maximizes the production horizon $K\Delta T$, considering a required service level σ_k as a global throughput to be reached for each time period k $(1 \leq k \leq K\})$ and a set of machines M_j $(1 \leq j \leq m)$, each with n running profiles $N_{i,j} = (\rho_{i,j}, RUL_{i,j})$ for $(0 \leq i < n)$. This notation allows to express different associated problems as the one that considers a homogeneous platform with a constant demand σ such that each machine M_j has only one running profile and provides the same production level $\rho_{0,j} = \rho_j = \rho$ and $RUL_{0,j} = RUL_j$ $(n = 1)$: $\text{MAXK}(\sigma \,|\, \rho \,|\, RUL_j)$.

4 Motivating example

Before going further, it may be helpful to consider a naive motivating example. This example aims at illustrating that using a machine within its more efficient profile is not always suitable for improving the useful life of the platform. The purpose is to show that considering different running profiles for machines can be a good deal to extend a platform production horizon while respecting a given targeted global throughput, even if the efficiency decreases when the useful life increases.

Let us consider four machines ($\mathscr{M} = \{M_1, M_2, M_3, M_4\}$) with which we plan to produce at least a global throughput reaching at least the demand $\sigma = 450$ as long as possible. At $t = 0$, M_1 is able either to produce $\rho_{0,1} = 450$ for one period of time or to produce $\rho_{1,1} = 125$ for three periods of time; at $t = 0$, each other machine M_j $(j = 2, 3, 4)$ is able to produce either $\rho_{0,j} = 350$ for one period of time or $\rho_{1,j} = 75$ for three periods of time (see Table 1). These profiles respect the profile model introduced before ($\rho_{i,j} \times RUL_{i,j} < \rho_{i-1,j} \times RUL_{i-1,j}$). $\forall t > 0$, each profile $N_{i,j}$ $(0 \leq i < n$ and $1 \leq j \leq m)$ depends on the usage of M_j in the past. The selected running profiles ($N_{i,j} = (\rho_{i,j}, RUL_{i,j})$) are given in Table 1 as a function of time for the three scenarios presented in Figure 2.

Table 1: Running profile ($N_{i,j} = (\rho_{i,j}, RUL_{i,j})$) for each machine and for each of the 3 scenarios shown in Figure 2 as a function of time

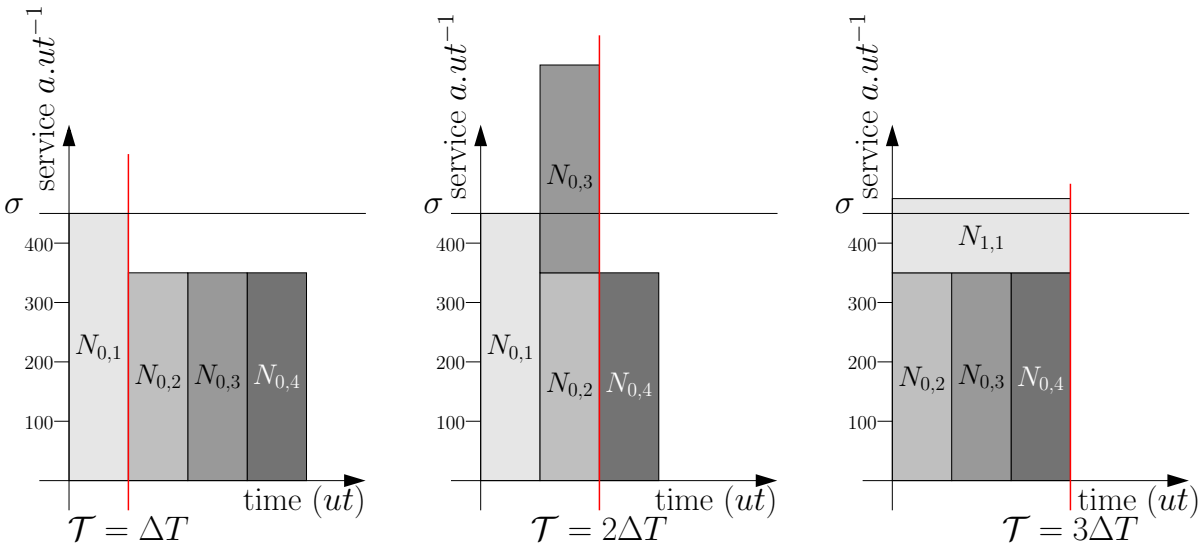
		$t=0$	$t=\Delta T$	$t=2\Delta T$	$t=3\Delta T$
scenario *S1* ($\mathcal{T}=\Delta T$)	M_1	(450,1)	(450,0)	(450,0)	(450,0)
		(125,3)	(125,0)	(125,0)	(125,0)
	M_2	(350,1)	(350,1)	(350,0)	(350,0)
		(75,3)	(75,3)	(75,0)	(75,0)
	M_3	(350,1)	(350,1)	(350,1)	(350,0)
		(75,3)	(75,3)	(75,3)	(75,0)
	M_4	(350,1)	(350,1)	(350,1)	(350,1)
		(75,3)	(75,3)	(75,3)	(75,3)
scenario *S2* ($\mathcal{T}=2\Delta T$)	M_1	(450,1)	(450,0)	(450,0)	(450,0)
		(125,3)	(125,0)	(125,0)	(125,0)
	M_2	(350,1)	(350,1)	(350,0)	(350,0)
		(75,3)	(75,3)	(75,0)	(75,0)
	M_3	(350,1)	(350,1)	(350,0)	(350,0)
		(75,3)	(75,3)	(75,0)	(75,0)
	M_4	(350,1)	(350,1)	(350,1)	(350,0)
		(75,3)	(75,3)	(75,3)	(75,0)
scenario *S3* ($\mathcal{T}=3\Delta T$)	M_1	(450,1)	(450,0)	(450,0)	(450,0)
		(125,3)	(125,2)	(125,1)	(125,0)
	M_2	(350,1)	(350,0)	(350,0)	(350,0)
		(75,3)	(75,0)	(75,0)	(75,0)
	M_3	(350,1)	(350,1)	(350,0)	(350,0)
		(75,3)	(75,3)	(75,0)	(75,0)
	M_4	(350,1)	(350,1)	(350,1)	(350,0)
		(75,3)	(75,3)	(75,3)	(75,0)

The main idea of the first scenario *S1* is to run each machine using its more efficient profile without allowing overproduction. One can see in Figure 2(a) that in this case the platform runs for four periods of time but the targeted throughput is achieved only for one period ΔT by using M_1. For each of the three last periods ($t > \Delta T$), the delivered throughput does not reach the demand σ since $\rho_{0,2}, \rho_{0,3}, \rho_{0,4} < \sigma$. Considering the objective, the useful life of the platform, i.e., the scheduling horizon, is then one period ($\mathcal{T} = \Delta T$) for this scenario *S1*.

When overproduction is allowed (Scenario *S2*), two machines can be used in parallel and the scheduling horizon is increased to two periods (see Figure 2(b) with $\mathcal{T} = 2\Delta T$). M_1 is used for $0 \leq t < \Delta T$ and M_2 and M_3 are used in parallel for $\Delta T \leq t < 2\Delta T$. For $t \geq 2\Delta T$, $RUL_{0,2} = 0$ and $RUL_{0,3} = 0$ and $\rho_{0,4} < \sigma$. If the production is stopped after two periods, some potential still remains. The machine M_4 has indeed never been used and does not need maintenance yet. The schedule proposed in Figure 2(b) is optimal under the previous assumptions.

Machines should then be used in another running profile to extend the production horizon. One can see in Figure 2(c) that the third scenario *S3* consisting in using the machine M_1 with a lower throughput allows to reach the targeted throughput for three periods. Indeed, by using M_1 for three periods with a throughput $\rho_{1,1} = 125$, its contribution can be added to the one of one of the other machines ($\rho_{0,j} = 350$ with $j = 2, 3, 4$) at each period. After having been used for one period, for each M_j with $j \geq 2$, $RUL_{0,j} = RUL_{1,j} = 0$. After the third

period, $RUL_{1,1} = RUL_{0,1} = 0$ for machine M_1 also. Because of the small number of alternative scenarios, it is easy to see that any other schedule can not reach the constraint $\sigma = 450$ for a larger number of periods ($\mathcal{T} \geq 3\Delta T$). $K = 3$ for $\mathcal{T} = 3\Delta T$ is then an optimal solution to the problem MAXK($\sigma_k \mid \rho_{i,j} \mid RUL_{i,j}$). Compared with *S2*, an extra period has been allowed in *S3* although the efficiency of M_1 has been reduced.



(a) Scenario *S1* (b) Scenario *S2* (c) Scenario *S3*

Figure 2: Motivating example

5 Optimal resolution and complexity results

Several classes of the general problem can be considered, depending on the machine throughput values and their *RUL*. In the following, some complexity results are demonstrated for different problem classes.

5.1 Homogeneous equipment case

First, consider the MAXK($\sigma \,|\, \rho \,|\, RUL_j$) problem. In this case, the total required throughput is constant (σ) and of the form $(q-1)\rho < \sigma \leq q\rho$ ($q \in \mathbb{N}^*$), with q the minimal number of parallel machines necessary to reach σ for each time period and ρ the throughput provided by each machine M_j. This problem can be stated as follows: how can q machines, from a set of m machines, be used at each time period so as to maximize the schedule horizon?

Let's consider the problem with discrete preemption allowed. In this case, the use of each machine may be started or interrupted at the beginning of each time period. In the following, we denote $\mathscr{K}(\mathscr{M}, q)$ the horizon of the optimal schedule for MAXK($\sigma \,|\, \rho \,|\, RUL_j$). Let $RUL_j(k)$ be the *RUL* of machine M_j at time $k \times \Delta T$. A relaxation of this problem can also be considered when the time is considered as continuous. In this case, preemption is allowed at each time and the maximal production horizon is denoted $\mathscr{K}_{cont}(\mathscr{M}, q) \times \Delta T$, with:

$$\mathscr{K}_{cont}(\mathscr{M}, q) = \frac{\sum_{1 \leq j \leq m} RUL_j(0)}{q} \tag{1}$$

$\mathscr{K}_{cont}(\mathscr{M}, q)$ corresponds to an upper bound for $\mathscr{K}(\mathscr{M}, q)$, reachable in only very limited scenarios. It can be seen as a geometric resolution which does not always comply with a permitted running of machines. As an example, consider three machines M_1, M_2 and M_3, with $RUL_1(0) = 1$, $RUL_2(0) = 1$ and $RUL_3(0) = 4$. According to Equation 1, if two machines are used in parallel ($q = 2$),

$$\mathscr{K}_{cont}(\mathscr{M}, q) = \frac{1 + 1 + 4}{2} = 3.$$

Machines M_1 and M_2 can for instance be used in parallel for one unit of time. Then, at time ΔT, $RUL_1(1) = RUL_2(1) = 0$ and $RUL_3(1) = 4$. To reach $\mathscr{K}_{cont}(\mathscr{M}, q) = 3$, machine M_3 has to be used in parallel with itself during two periods, which is actually not possible.

5.1.1 An associated problem

The problem at stake in this section appears to be very similar to a classical parallel machine scheduling problem. Pinedo [26] considered the problem $P_q|prmp|C_{max}$ where m jobs have to be scheduled on q parallel resources with the objective of minimizing the makespan C_{max}. A parallel can be drawn between the two problems. The q machines running in parallel that we consider here are equivalent to the q parallel resources considered by Pinedo. The m machines that have to be used are equivalent to the m jobs that Pinedo aimed at scheduling. $RUL_{0,j}$ of a machine M_j is equal to a job processing time (p_j) and entirely wear out a machine is the same as finish a job. The main difference is the objective function. Pinedo minimized the duration of the schedule while we aim at maximizing the schedule horizon. Our problem can however be reduced to the classical problem of parallel machines detailed earlier. In [26], Pinedo suggested to use the LRPT algorithm (Longest Remaining Processing Time first) to treat this problem and provided the following theorem: "*LRPT yields an optimal schedule for $P_q|prmp|C_{max}$ in discrete time*" [26]. With no precedence constraint between tasks and if preemption is allowed, the LRPT schedule is by construction active, nondelay and with no idle-time. This result is consistent with our requirement to meet the fulfillment of the demand for each time period.

5.1.2 Optimal greedy algorithm

The MAXK$(\sigma \,|\, \rho \,|\, RUL_j)$ problem can be optimally solved using the Longest Remaining Useful Life first greedy algorithm (LRUL). This algorithm is based on the same principle as the LRPT (Longest Remaining Processing Time first) algorithm proposed by Pinedo [26] to find an optimal schedule for the classical parallel machine scheduling problem $P_q|prmp|C_{max}$ in discrete time. LRUL algorithm consists in choosing first the machines having the longest remaining useful life, when discrete preemption is allowed. At the beginning of each time period k ($1 \leq k \leq K$), the q machines having the longest RUL at time k are scheduled for one period.

5.1.3 maxK$(\sigma \,|\, \rho \,|\, \boldsymbol{RUL}_j)$ complexity

To find the complexity we first need to demonstrate some lemmas.

Lemma 1. *If $q < m$ and $max_{1 \leq j \leq m}(RUL_j) \geq \mathscr{K}_{cont}(\mathscr{M}, q)$, then $\mathscr{K}(\mathscr{M}, q) = \mathscr{K}(\mathscr{M}', q - 1)$, with $\mathscr{M}' = \mathscr{M} \setminus \{M_{j'} \text{ s.t. } RUL_{j'} = max_{1 \leq j \leq m}(RUL_j)\}$.*

Lemma 1 states that solving a problem complying with its conditions is equivalent to solving the same problem without considering the machine with the greatest RUL. This reduction to an equivalent problem is illustrated in Figure 3.

Proof. If $max_{1\leq j\leq m}(RUL_j) \geq \mathscr{K}_{cont}(\mathscr{M}, q)$, the machine with the largest RUL is used all through the scheduling horizon. The maximal number of periods that can be completed, say $\mathscr{K}(\mathscr{M}, q)$, is not limited by this machine. $\mathscr{K}(\mathscr{M}, q)$ can then be found by solving the problem without taking into account the machine with the largest RUL and with a reduced number of machines $(q-1)$ necessary to reach the updated required throughput $(\sigma - \rho)$. That concludes the proof. □

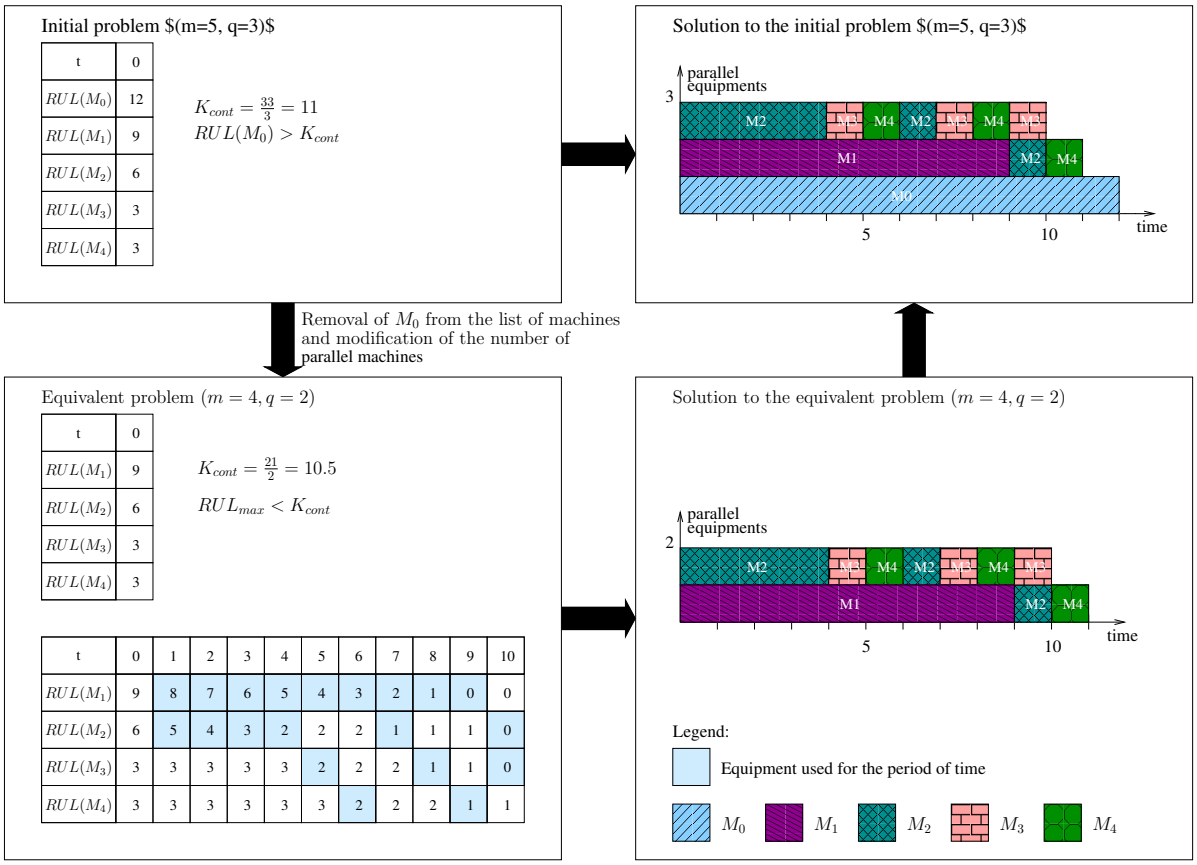


Figure 3: Simplification of a problem falling within the scope of Lemma 1

Lemma 2. *If $q < m$, $RUL_j(0) \geq 1$ $(1 \leq j \leq m)$ and $max_{1\leq j\leq m}(RUL_j(0)) \leq \mathscr{K}_{cont}(\mathscr{M}, q)$, an optimal schedule can be found to the* $\text{MAXK}(\sigma \,|\, \rho \,|\, RUL_j)$ *problem using the Largest Remaining Useful Life first greedy algorithm (LRUL) and $\mathscr{K}(\mathscr{M}, q) = \lfloor \mathscr{K}_{cont}(\mathscr{M}, q) \rfloor$.*

Proof. By construction, by using the LRUL algorithm and since $RUL_j(0) \geq 1$ for all $1 \leq j \leq m$, it always exists q machines such that $RUL_j(k) \geq 1$ for all $0 \leq k \leq k_1$, with

$$k_1 = \left\lfloor \frac{\sum_{1\leq j\leq m} (RUL_j(0) - 1)}{q} \right\rfloor.$$

When $k = k_1 + 1$, $RUL_j(k_1 + 1) \leq 1$ for all $1 \leq j \leq m$ and

$$\sum_{1\leq j\leq m} RUL_j(k_1) = m + r, \quad \text{with} \quad r = \sum_{1\leq j\leq m} (RUL_j(0) - 1) \mod q.$$

At time $k_1\Delta T$, k_1 periods have been completed and the demand can be reached for $\lfloor (m+r)/q \rfloor$ more periods. It is then possible to find a schedule that uses the LRUL algorithm and that completes $k_1 + \lfloor (m+r)/q \rfloor$ periods. We have:

$$\begin{aligned} k_1 + \left\lfloor \frac{m+r}{q} \right\rfloor &= \left\lfloor \frac{\sum_{1\leq j\leq m} (RUL_j(0)-1)}{q} \right\rfloor + \left\lfloor \frac{m+r}{q} \right\rfloor \\ &= \frac{\sum_{1\leq j\leq m} (RUL_j(0)-1)}{q} - \frac{r}{q} + \frac{m+r}{q} - \frac{r'}{q} \quad \text{with} \quad r' = (m+r) \mod q. \\ &= \frac{\sum_{1\leq j\leq m} RUL_j(0) - m + m - r + r}{q} - \frac{r'}{q} \\ &= \frac{\sum_{1\leq j\leq m} RUL_j(0)}{q} - \frac{r'}{q} \end{aligned}$$

And finally:

$$\begin{aligned} \mathscr{K}(\mathscr{M}, q) &= k_1 + \left\lfloor \frac{m+r}{q} \right\rfloor \\ &= \left\lfloor \frac{\sum_{1\leq j\leq m} RUL_j(0)}{q} \right\rfloor = \lfloor \mathscr{K}_{cont}(\mathscr{M}, q) \rfloor . \end{aligned}$$

That concludes the proof. □

Theorem 1. *An optimal solution to the problem* MAXK$(\sigma \,|\, \rho \,|\, RUL_j)$ *can be computed in polynomial time in* $O(\mathscr{K}_{cont}(\mathscr{M}, q) \times m \times log(m))$.

Proof. We aim at showing the complexity of finding an optimal schedule for MAXK$(\sigma \,|\, \rho \,|\, RUL_j)$ in any cases.

- Straightforward cases:

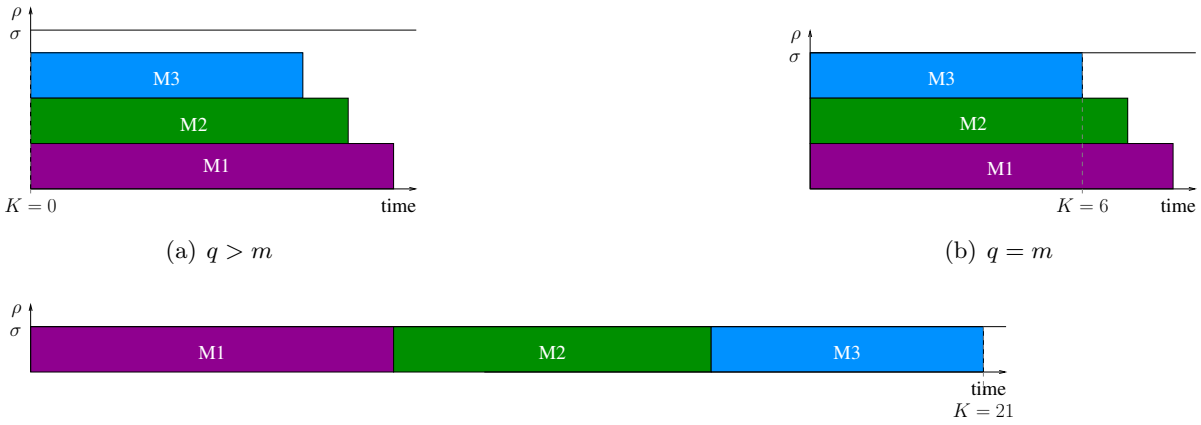
$$\mathscr{K}(\mathscr{M}, q) = \begin{cases} 0 & \text{if } q > m \ (2.1) \\ \sum_{1\leq j\leq m} RUL_j(0) & \text{if } q = 1 \ (2.2) \\ min_{1\leq j\leq m}(RUL_j(0)) & \text{if } q = m \ (2.3) \end{cases} \tag{2}$$

If $q > m$, all the available machines are not enough to provide the total required throughput σ. The production horizon is then zero (see Equation (2.1) and Figure 4(a)).

If $q = 1$, only one machine can be used at once in each time period. All the machines can then be used totally, one after another. The maximal production horizon is then obtained by adding the initial RUL from all the machines (see Equation (2.2) and Figure 4(c)).

If $q = m$, all the available machines have to be used in parallel to reach the demand σ. The maximal production horizon is then limited by the lowest RUL (see Equation (2.3) and Figure 4(b)).

For these three first cases, the solution of MAXK$(\sigma \,|\, \rho \,|\, RUL_j)$ can thus be found respectively in $O(1)$, $O(m)$ and $O(m)$.



(a) $q > m$ (b) $q = m$

(c) $q = 1$

Figure 4: Straightforward cases

- Other cases:

$$\mathscr{K}(\mathscr{M}, q) = \left\lfloor \frac{\sum_{1 \leq j \leq m} RUL_j(0)}{q} \right\rfloor \tag{3}$$

if $q < m$ and $RUL_j(0) \geq 1$ $(1 \leq j \leq m)$

We assume that we are in the conditions defined within Lemma 2. Otherwise, Lemma 1 insures that it exists an equivalent problem without machines which RUL are greater than $\mathscr{K}_{cont}(\mathscr{M}, q)$. According to Lemma 2, an optimal schedule to the problem is obtained thanks to LRUL algorithm with a horizon given by Equation (3). As maximum $\mathscr{K}_{cont}(\mathscr{M}, q)$ sorting of the m machines have to be performed during the scheduling, the complexity of the used algorithm is in $O(\mathscr{K}_{cont}(\mathscr{M}, q) \times m \times log(m))$. □

5.2 NP-Completeness in the general case

The general case is $\text{MAXK}(\sigma_k \,|\, \rho_{i,j} \,|\, RUL_{i,j})$. We prove in this section that this problem is NP-Complete in the strong sense.

Theorem 2. *Finding an optimal solution to the problem* $\text{MAXK}(\sigma_k \,|\, \rho_{i,j} \,|\, RUL_{i,j})$ *is NP-Complete in the strong sense.*

Proof. The NP-Completeness of the general problem will be demonstrated by proving that the special case $\text{MAXK}(\sigma \,|\, \rho_j \,|\, 1)$ is NP-hard in the strong sense.

Let us consider the following decision problem: given a horizon of K periods, is there a schedule that allocates machines over time such that the demand σ is reached for every period k $(1 \leq k \leq K)$? In other words, if $\mathscr{M}_k \subset \mathscr{M}$ is the set of machines that are scheduled within the period k, $\forall k \leq K$, is $\sum_{j|M_j \in \mathscr{M}_k} \rho_j \geq \sigma$? The problem is in NP: given a schedule of K periods of time, it is easy to check in polynomial time whether it is valid or not. The NP-Completeness is obtained by reduction from 3-PARTITION [13] which is NP-Complete in the strong sense.

Let us consider an instance $\mathcal{I}_1$ of 3-PARTITION: given an integer B and $3K$ positive integers $a_1, a_2, \ldots, a_{3K}$ such that for all $j \in \{1, \ldots, 3K\}$, $B/4 < a_j < B/2$ and with $\sum_{j=1}^{K} a_j = KB$, does exist a partition $I_1, \ldots, I_K$ of $\{1, \ldots, 3K\}$ such that for all $k \in \{1, \ldots, K\}$, $|I_k| = 3$ and $\sum_{j \in I_k} a_j = B$? We build the following instance $\mathcal{I}_2$ of our problem with K periods, each period

k having a length of time $\Delta T = 1$ and with a demand $\sigma_k = \sigma = B$ for $1 \leq k \leq K$. There are $3K$ machines M_j in $\mathscr{M}$ with $RUL_j = 1$ and $\rho_j = a_j$ for all $1 \leq j \leq 3K = m$. Clearly, the size of $\mathcal{I}_2$ is polynomial in the size of $\mathcal{I}_1$. We now show that $\mathcal{I}_1$ has a solution if and only if $\mathcal{I}_2$ does.

Suppose first that $\mathcal{I}_1$ has a solution. For $1 \leq k \leq K$, machine M_j is assigned to $\mathscr{M}_k$ within the period k with $j \in I_k$ and $\rho_j = a_j$. Then, we have $\sum_{j|M_j \in \mathscr{M}_k} \rho_j = \sigma = \sum_{j \in I_k} a_j = B$ and therefore the constraint on the demand is respected for the K periods. We have a solution to $\mathcal{I}_2$.

Suppose that $\mathcal{I}_2$ has a solution. Let $\mathscr{M}_k$ be the set of machines allocated to the period k such that for all machines $M_j \in \mathscr{M}_k$ with $j \in I_k$, $\sum_{j \in I_k} \rho_j = \sigma = B$. Because of ρ_j, $|\mathscr{M}| = |I_k| = 3$. Since the demand σ has to be reached for the K periods, the solution is a 3-PARTITION.

We have proven that the problem $\text{MAXK}(\sigma \,|\, \rho_j \,|\, 1)$ is NP-Complete in the strong sense. Since this problem is a special case of $\text{MAXK}(\sigma_k \,|\, \rho_{i,j} \,|\, RUL_{i,j})$, it is sufficient to prove the NP-Completeness of $\text{MAXK}(\sigma_k \,|\, \rho_{i,j} \,|\, RUL_{i,j})$. This concludes the proof. □

5.3 Bounds

To improve the bounding accuracy and to speed up the solver, particularly for large instances, upper and lower bounds for $\mathscr{K}(\mathscr{M}, q)$ are supplied for the general $\text{MAXK}(\sigma \,|\, \rho_{i,j} \,|\, RUL_{i,j})$ problem.

5.3.1 Upper bound

An upper bound KMAX can be provided as defined in Equation (4). This equation is a generalization of Equation (1) that defines an upper bound $\mathscr{K}_{cont}(\mathscr{M}, q)$ for the special case considered in Section 5.1. If all the machines are used with their running profile that provides the best output (say $Q_{i,j} = \rho_{i,j} \times RUL_{i,j}$) and if the total required throughput σ is constant over time, then KMAX is the theoretical maximal number of periods for which the demand σ can be reached. This upper bound is only reachable under very restrictive conditions, i.e., if no overproduction is performed during the whole scheduling horizon and if no potential remains at the end of the schedule.

$$\text{KMAX} = \left\lfloor \frac{\sum_{1 \leq j \leq m} \max_{0 \leq i < n} (\rho_{i,j} RUL_{i,j})}{\sigma} \right\rfloor \tag{4}$$

In practice, KMAX is never reached. In the same way as for $\mathscr{K}_{cont}(\mathscr{M}, q)$ for the homogeneous equipment case (see Equation 1 in Section 5.1), the construction of this upper bound can be seen as a geometric filling of a rectangle whose width corresponds to the demand σ and whose length is maximal knowing the previous width (see Figure 5). The surface area of this rectangle is then commensurate with the global potential of the set of machines. The solution that is obtained does not comply with two important constraints of the general problem. The time discretizing is first not observed. The time is considered as continuous and preemption is allowed at each moment. A change in the configuration (machines/running profiles) is then allowed at each time. The global potential is secondly spread in the rectangle without considering the throughput characteristics of each machine. Throughputs that can be provided by each machine are then not necessarily observed at each time.

5.3.2 Lower bounds

Many lower bounds can be defined. In each scenario, all the machines are supposed to be used in their nominal running profile $N_{0,j}$ providing the maximal throughput $\rho_{0,j}$ and associated with the minimum RUL, $RUL_{0,j}$.

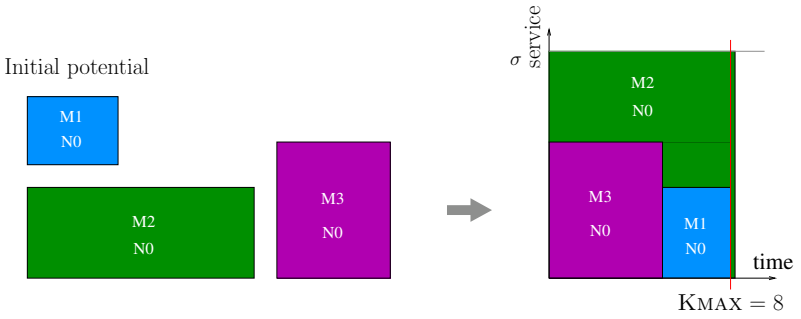


Figure 5: Construction of KMAX

First lower bound, KMIN_1, is obtained when all the available machines are used in parallel. The maximal production horizon is then limited by the lowest RUL (see Equation (5) and Figure 6(a)).

$$\text{KMIN}_1 = min_{1 \leq j \leq m}(RUL_{min,j}) \tag{5}$$

For the second lower bound, machines are supposed to be sorted by a descending order of their $RUL_{0,j}$. We are searching for the first machine M_{j^*} with $j^* = j_{min}$ such that:

$$\sum_{j=0}^{j^*} \rho_{0,j} \geq \sigma.$$

KMIN_2 corresponds to the RUL of machine M_{j^*} (see Equation (6) and Figure 6(b)).

$$\text{KMIN}_2 = RUL_{0,j^*} \tag{6}$$

A third lower bound that extends the principle used for KMIN_1 can be defined (see Figure 6(c)). The idea is to form many groups of machines such as each group allows to reach the demand σ. Worst case is considered for each group, that is, the overproduction is supposed to be maximal. In this case, the total throughput is the following:

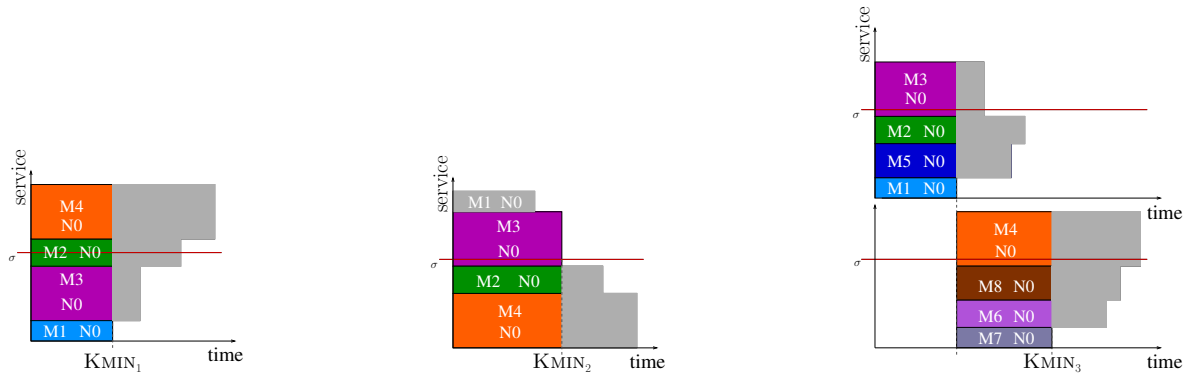
$$\rho_{tot} = \sigma + max_{1 \leq j \leq m}(\rho_{0,j}) - 1$$

We are searching for the number of groups of machines j_3 that can be built under the previous assumptions. j_3 is given by:

$$j_3 = \left\lfloor \frac{\sum_{j=1}^{m} \rho_{0,j}}{\rho_{tot}} \right\rfloor = \left\lfloor \frac{\sum_{j=1}^{m} \rho_{0,j}}{\sigma + max_{1 \leq j \leq m}(\rho_{0,j}) - 1} \right\rfloor$$

The production horizon is limited by the minimal RUL in each group of machines and KMIN_3 is then defined by Equation (7), assuming the machines are sorted by an ascending order of their RUL.

$$\text{KMIN}_3 = \sum_{j=1}^{j_3} RUL_{0,j} \tag{7}$$



(a) KMIN$_1$ (b) KMIN$_2$ (c) KMIN$_3$

Figure 6: Lower bounds

6 Optimal approach

Let's consider the following scheduling problem: MAXK$(\sigma \mid \rho_{i,j} \mid RUL_{i,j})$. A first approach based on an exact resolution method can be considered to cope with this problem.

6.1 Decision problem

The decision problem can be described as follows: does exist a schedule to achieve the constant given service σ during a given number of time periods K, considering the current health state of all machines, i.e., the value of $\{RUL_{i,j}$ s.t. $0 \leq i \leq N-1$ and $1 \leq j \leq M\}$?

For this problem, denoted DP_K, one can propose an Integer Linear Program $(ILP(DP_K))$ using binary variables.

6.1.1 Variables

Let $a_{i,j,k}$ s.t. $0 \leq i \leq n-1, 1 \leq j \leq m$ and $1 \leq k \leq K$ be the variables of the decision problem. $a_{i,j,k}$ is defined as a bivalent variable. $a_{i,j,k} = 1$ if machine M_j is used with the running profile N_i during the period k ; $a_{i,j,k} = 0$ otherwise.

Using this set of variables involves that machines are supposed to be able to change of running profile at the beginning of each period.

6.1.2 Constraints

The constraints of the decision problem DP_K should express the need to reach the required production throughput and the limitation on the use of the machines, due to their functioning and to their RUL.

The first set of constraints concerns the production throughput. The required service σ should be reached for all time periods. This can be expressed by the following inequalities:

$$\forall k, \quad \sum_{j=1}^{M} \sum_{i=1}^{N} (a_{i,j,k} \times \rho_{i,j}) \geq \sigma \tag{8}$$

The second set of constraints sets that a machine M_j can only be used once per period k, in only one running profile N_i:

$$\forall k, \ \forall j, \quad \sum_{i=1}^{N} a_{i,j,k} \leq 1 \tag{9}$$

Finally, the last set of constraints is due to the remaining useful life of each machine. We can consider that during a given period k, if a machine M_j is used with the running profile N_i, then it cuts the remaining useful life by $\Delta T / RUL_{i,j}$. Consequently, due to the value of $RUL$of the machine M_j, the following inequalities express that each machine could not be used more than its initial RUL:

$$\forall j, \quad \sum_{i=1}^{n} \frac{\sum_{k=1}^{K} a_{i,j,k} \times \Delta T}{RUL_{i,j}} \leq 1 \tag{10}$$

6.2 Associated optimization problem

The previous described Integer Linear Program, denoted $ILP(\sigma, \mathscr{M}, K)$ allows without any objective function to answer the following question: does exist a configuration of all the machines such that the required throughput σ could be reached during at least K periods ?

6.2.1 Minimizing the production loss

One can use this model to obtain solutions where an objective would be to limit the loss of production. What is called loss of production here is the difference between the resulting throughput of a given configuration (list of machines used during the period of time, with the running profile for each machines) and the required one ($\rho_{i,j} - \sigma$). The production loss for a given period k is then as follows:

$$\phi_k = \sum_{j=1}^{M} \sum_{i=1}^{N} (a_{i,j,k} \times \rho_{i,j}) - \sigma \tag{11}$$

A first optimization problem that can be addressed is consequently the minimization of the total production loss $\sum_{k=1}^{K} \phi_k$. This will almost correspond to keep a maximum potential of production for the set of machines $\mathscr{M}$ as long as possible.

6.2.2 Maximizing the production horizon

As presented in section 3, we propose to solve the problem where the set of machines $\mathscr{M}$ has to produce the given global throughput σ as long as possible. Besides the previous model can compute a solution to reach a global throughput σ, it is not sufficient since it is designed for a given number of periods K. It can nevertheless be useful to determine the greatest number of periods during which a given platform $\mathscr{M}$ is able to produce a throughput greater or equal to the demand σ. First, one can determine two bounds of this number. The first one is an upper bound, KMAX, defined in Section 5.3.1 (see Equation 4). The worst lower bound KMIN is 0. The three lower bounds defined by Equations (5),(6) and (7) (see section 5.3.2) can be considered. If a heuristic algorithm can provide a solution, the corresponding production horizon could also be considered as a better lower bound.

Since one can compute these two bounds, KMAX and KMIN, finding the maximum number of periods that can be reached for a given demand σ with a given platform $\mathscr{M}$ can be done using a dichotomy search approach. This approach is detailed in Algorithm 1.

$ILP(DP_K)$ is in fact a Binary Linear Program and solving such a binary linear program is NP-Complete. However, as shown in Section 8, efficient solvers as [17] or [15] are able to give solutions in limited time only for small problem instances. For more realistic problem sizes, defining scalable heuristics is mandatory. Thanks to the previous ILP, a validation of these heuristics is proposed in section 8 for small size instances of the problem.

Algorithm 1: Dichotomy search procedure to find the maximum number of periods K using the $ILP(DP_K)$

Remark: for this algorithm, we call $ILP(\sigma, \mathscr{M}, K)$ the integer linear program described in section 6.1 and $LP(\sigma, \mathscr{M}, K)$ its rational relaxation
$K_{min} \leftarrow$ KMIN
$K_{max} \leftarrow$ KMAX
while $K_{max} - K_{min} > 0$ **do**
  $K \leftarrow (K_{min} + K_{max})/2$
  if $LP(\sigma, \mathscr{M}, K)$ *has a solution* **then**
    if $ILP(\sigma, \mathscr{M}, K)$ *has at least one solution* **then**
      $K_{min} \leftarrow K$
    else
      $K_{max} \leftarrow K$
  else
    $K_{max} \leftarrow K$
return K

7 Sub–optimal approaches

The optimal solution can not be computed using the BIP defined in the previous section as soon as platforms with a large number of machines and/or with a large number of running profiles are considered. In order to deal with large scale problems, four polynomial time heuristics that allocate for each period of time enough machines to reach the targeted throughput as long as possible are then proposed. Each heuristics follows its own strategy to select the machines and an associated running profile for each of them so as to define its contribution to the global production within the current period. A first strategy consists in defining the schedule period by period. A new selection of machines is performed for each period and is applied only for one period of time. The *RUL* of the selected machines are updated at the end of each period to take their usage into account. The whole process is then iterated until the set of available machines is not able to reach σ any more. An other strategy consists in applying the same selection on many periods. The number of periods on which a solution can be applied is limited by the selected machine having the smallest *RUL*. Each *RUL* of the selected machines is then updated as for the first strategy and the process is iterated with the remaining set of available machines. The number of periods K that are successfully completed represents the useful life of the platform.

One can see that the strategy working by group of periods cannot be used as it is when a variable demand σ_k is considered. It could nevertheless easily be adapted by applying each selection on the minimum of the two following values: (1) smallest *RUL* of the selected machines and (2) length of the time interval in which the targetted demand remains constant. Regardless of the two strategies, three different types of heuristics can be distinguished. First heuristics provides a random schedule. Second type comprises greedy heuristics and last one uses a dynamic programming based algorithm. An illustration based on the initial set of machines described in Figure 7 is given for each heuristics in the following sections.

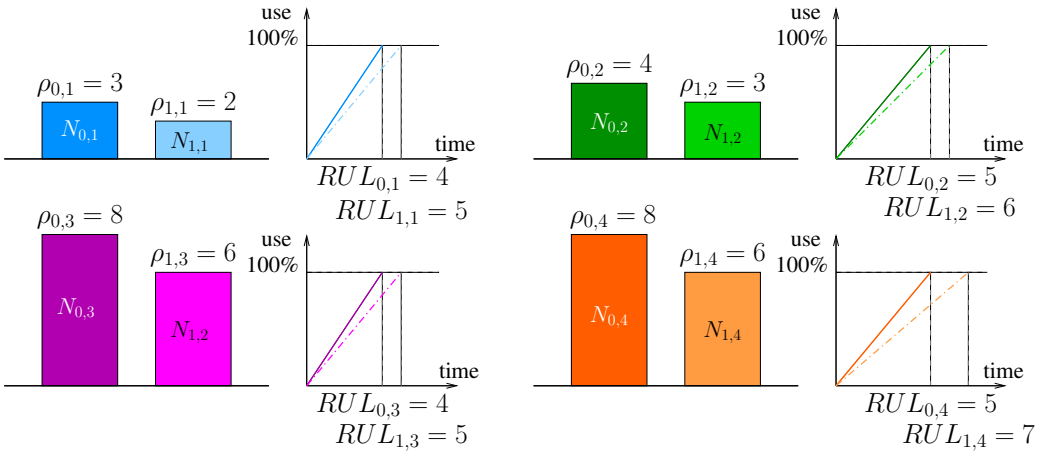


Figure 7: Initial set of machines

7.1 Basic heuristics

7.1.1 H–RAND: Random assignment heuristics

This first heuristics builds the schedule period by period. For each period k, just as much couples machine/running profile as needed are randomly selected among those that are available so as to reach at least the demand σ. The process, detailed in Algorithm 2, is stopped as soon as a period can not be completed. This can happen even if there is enough potential left to reach the demand σ. In each period, the remaining available machines depends indeed on the first choices made by the heuristics.

This naive heuristics mainly serves as a basis for comparison and assesses the interest of defining more complex and smart heuristics to extend the useful life of the system to a number of periods closed to the optimal one. A solution provided by H–RAND and based on the set of machines described in Figure 7 is given in Figure 8.

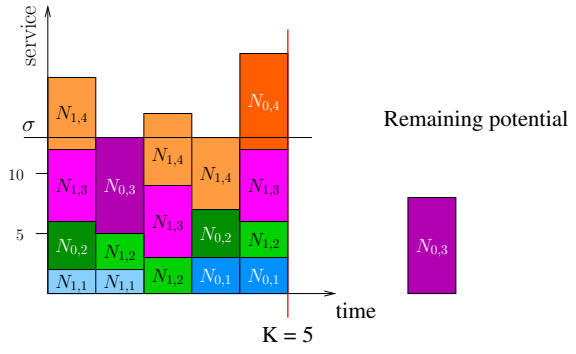


Figure 8: Schedule obtained with H–RAND

7.1.2 H–LRF: Largest *RUL* First heuristics

This heuristics works by group of periods and aims at considering each machine M_j using its profile associated to the largest RUL, that is the profile providing the lowest throughput: $N_{n-1,j} = (\rho_{n-1,j}, RUL_{n-1,j})$. We recall that $\rho_{n-1,j} = \min_{0 \leq i < n}(\rho_{i,j})$ and $RUL_{n-1,j} = \max_{0 \leq i < n}(RUL_{i,j})$ for any machine $M_j \in \mathscr{M}$. A subset of machines having the greatest RUL is selected until the

Algorithm 2: H–RAND: Random assignment heuristics

Data: σ: throughput demand
ρ_{max}: maximal throughput reachable with the considered set of machines
$list$: list of available couples machine/running profile $N_{i,j}$, with $RUL_{i,j} \geq 1$
sol_k: list of couples machine/running profile $N_{i,j}$ selected for the period k
ρ_{tot}: total throughput provided by the solution sol_k
input : σ
set of initial couples $N_{i,j} = (\rho_{i,j}, RUL_{i,j})$ for each machine j and running profile i
output: $sol = (sol_1, sol_2, \ldots, sol_k)$: list of solutions for each completed period

```
repeat
    k ← 0
    ρ_max ← Σ_{j|M_j∈ℳ} ρ_{max,j}
    sol ← ∅
    ρ_tot ← 0
    if ρ_max ≥ σ then
        while (ρ_tot < σ) & (list is not empty) do
            N_{i,j} ← choose randomly a machine and an associated running profile in list
            if the selected machine M_j is not already used in sol_k then
                sol_k ← sol_k + N_{i,j}
                ρ_tot ← ρ_tot + ρ_{i,j}
                list ← list − N_{i,j}
    Use the selected machines for one period of time
    Update the RUL of the selected machines to take their usage into account
    k++
until ρ_tot ≥ σ
return sol = (sol_1, ..., sol_k)
```

global throughput ρ_{tot} reaches at least the demand σ. If the available machines are not sufficient to reach σ ($\rho_{tot} < \sigma$), all the available machines are selected and the throughput of the machine M_j having the largest RUL is increased by selecting $N_{i-1,j}$ in spite of $N_{i,j}$ if possible. This process is iterated within the current period until $\rho_{tot} \geq \sigma$. While the global throughput ρ_{tot} exceeds the demand σ, the selected machine providing the maximal throughput $\rho_{max,j}$ such that $\rho_{max,j}$ is lower or equal to the overproduction (i.e., $\rho_{max} < \rho_{tot} - \sigma$) is erased from the solution. The solution is applied to the maximum number of periods, which corresponds to the smallest RUL of the solution. The process, detailed in Algorithm 3 and illustrated in Figure 9, is stopped as soon as a period can not be completed anymore.

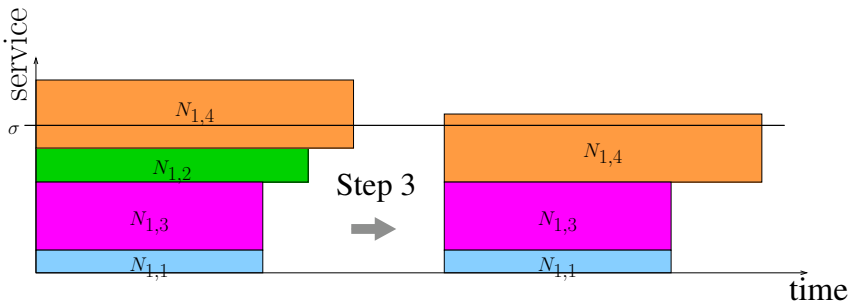


Figure 9: H–LRF operating principle

Figure 10 represents a solution provided by H–LRF for the set of machines considered in this section (see Figure 7).

Algorithm 3: H–LRF: Largest RUL First heuristics

Data: σ: throughput demand
ρ_{max}: maximal throughput reachable with the considered set of machines
$list$: list of available couples machine/running profile $N_{i,j}$, with $RUL_{i,j} \geq 1$
sol_k: list of couples machine/running profile $N_{i,j}$ selected for the period k
ρ_{tot}: total throughput provided by the solution sol_k
input : σ
set of initial couples $N_{i,j} = (\rho_{i,j}, RUL_{i,j})$ for each machine j and running profile i
output: $sol = (sol_1, sol_2, \ldots, sol_k)$: list of solutions for each completed period

repeat
  $k \leftarrow 0$
  $\rho_{max} \leftarrow \sum_{j|M_j \in \mathscr{M}} \rho_{max,j}$
  if $\rho_{max} \geq \sigma$ **then**
    $sol_k \leftarrow$ all available machines in their running profile associated to the largest RUL $(N_{n-1,j} = (\rho_{min,j}, RUL_{max,j}))$
    $\rho_{tot} \leftarrow$ total throughput provided by sol_k
    while $\rho_{tot} < \sigma$ **do**
      Select the machine in sol_k having the maximal RUL in a more nominal running profile ($N_{i-1,j}$ in spite of $N_{i,j}$)
    while $\rho_{tot} > \sigma$ **do**
      $sol_k \leftarrow sol_k -$ $N_{i,j}$ with machine M_j providing the maximal throughput $\rho_{max,j}$ from the solution such that $\rho_{max,j} < \rho_{tot} - \sigma$
  Use the selected machines for a number of periods equal to the smallest RUL of sol_k
  Update the RUL of the selected machines to take their usage into account
  $k++$
until $\rho_{tot} \geq \sigma$
return $sol = (sol_1, \ldots, sol_k)$

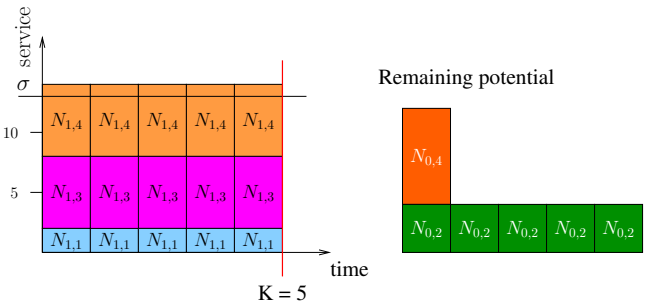


Figure 10: Schedule obtained with H–LRF

7.1.3 H–HOF: Highest Output First heuristics

The heuristics H–HOF is based on the same principle as H–LRF but each machine M_j is configured with its most efficient profile $N_{0,j} = (\rho_{0,j}, RUL_{0,j})$. We recall that $\rho_{0,j} = \max_{0 \leq i < n}(\rho_{i,j})$ and $RUL_{0,j} = \min_{0 \leq i < n}(RUL_{i,j})$ for any machine $M_j \in \mathscr{M}$. Two options can be considered. First one, H–HOFlt (Highest Output First, lowest throughput first), selects the machines having the lowest throughput first and second one, H–HOFht (Highest Output First, highest throughput first), these having the highest throughput. For both options, a subset of the smallest number of machines that allows to reach σ is selected by the corresponding order. Then, as long as possible, the machine whose RUL is the smallest from the selected machines (say M_l) is iteratively chosen and its throughput is decreased from the running profile $N_{i,j}$ to the sub-

Algorithm 4: H–HOF: Highest Output First heuristics

Data: σ: throughput demand
ρ_{max}: maximal throughput reachable with the considered set of machines
$list$: list of available couples machine-running profile $N_{i,j}$, with $RUL_{i,j} \geq 1$
sol_k: list of couples machine-running profile $N_{i,j}$ selected for the period k
ρ_{tot}: total throughput provided by the solution sol_k
input : σ
set of initial couples $N_{i,j} = (\rho_{i,j}, RUL_{i,j})$ for each machine j and running profile i
output: $sol = (sol_1, sol_2, \ldots, sol_k)$: list of solutions for each completed period

repeat
  $k \leftarrow 0$
  $\rho_{max} \leftarrow \sum_{j|M_j \in \mathscr{M}} \rho_{max,j}$
  if $\rho_{max} \geq \sigma$ **then**
    while $\rho_{tot} < \sigma$ **do**
      $sol_k \leftarrow sol_k + N_{i,j}$ with M_j the available machine providing the lowest (resp. the highest) throughput in its more efficient running profile $N_{i,j}$ for the first version H–HOFlt (resp. the second version H–HOFht)
    while $\rho_{tot} \geq \sigma$ **do**
      Decrease the contribution of the machine M_j in sol_k having the minimal RUL by modifying its running profile from the chosen one $N_{i,j}$ to the following one providing a lowest throughput $N_{i+1,j}$, only if ρ_{tot} remains $\geq \sigma$
  Use the selected machines for a number of periods equal to the smallest RUL of sol_k
  Update the RUL of the selected machines to take their usage into account
  $k++$
until $\rho_{tot} \geq \sigma$
return $sol = (sol_1, \ldots, sol_k)$

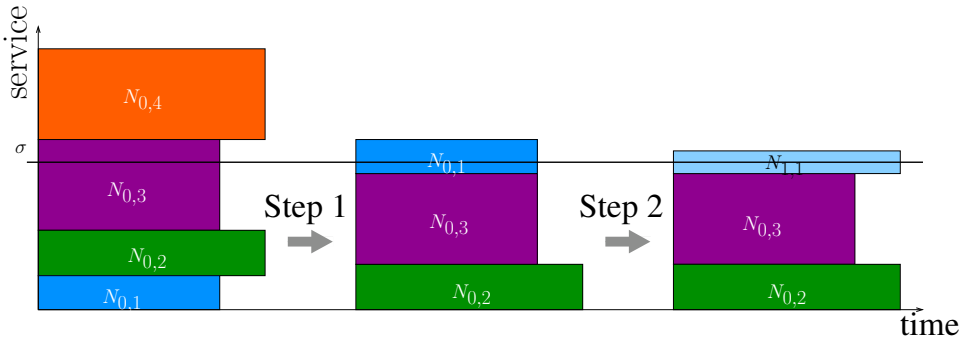
nominal profile $N_{i+1,j}$, but only if the overall throughput remains greater than σ. This allows to increase the number of periods on which the solution can be applied. As for H–LRF, the number of completed periods is indeed given by the selected machine M_l which RUL is the smallest. RUL values are then updated for every machine and the process is repeated until enough machines are available to reach σ.

This process (see Algorithm 4 and illustrations in Figure 11 and Figure 12) is iterated as long as there is enough potential left to reach the demand for a minimum of one period.

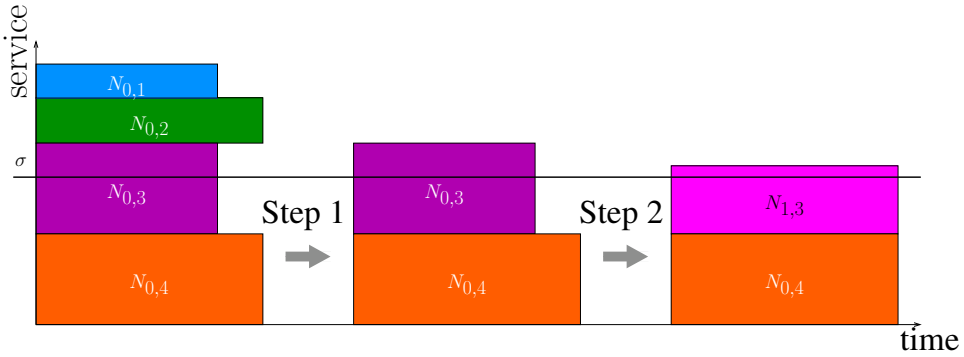
7.1.4 H–DP: Dynamic Programming based heuristics

H–DP is a more sophisticated heuristics. It aims at minimizing the production loss. If one period is considered, the problem is to find the best subset of machines and to configure each machine in the best running profile in order to reach at least the production demand with the smallest overproduction. A Knapsack-like algorithm is proposed so as to make the choice between all the available couples within the current period. The differences with the classical Knapsack problem is first that the sum of the value ($\rho_{i,j}$) of the selected objects (subset of couples machine/running profile) should be greater or equal to the knapsack weight (σ). Secondly, each object (M_j) can have several values ($\rho_{i,j}$, $0 \leq i \leq n-1$) and at most one could be selected. The objective of the Knapsack-like problem is to minimize the sum of the machine value in the case where this sum exceeds the knapsack weight σ.

The algorithm developed to implement H–DP (see Algorithms 5) is the classical two-dimensional dynamic programming based approach. Each available machine is successively considered, following an ascending order of their throughput. This sorting allows to minimize



(a) Version 1: H–HOFlt



(b) Version 2: H–HOFht

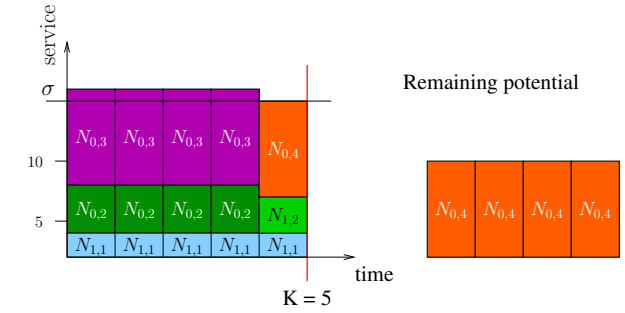
Figure 11: H–HOF operating principle

the number of recorded solutions and therefore minimizes both the memory needed and the processing time. Machines with the same throughput are also sorted in descending order of their *RUL*. Each running profile of each machine is successively considered, from the last one providing the minimal throughput to the most nominal one. Performing this sorting before each search for solution allows to wear out the set of machines homogeneously. Due to this turnover in the use of machines, a maximum of different machines are kept available for the last periods and the production horizon is extended.

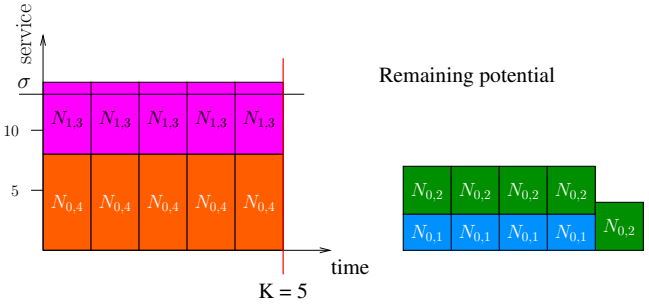
For each machine M_j, the targeted throughput σ' is iterated from 1 to σ. For each value of σ', each available profile $N_{i,j} = (\rho_{i,j}, RUL_{i,j})$ $(0 \leq i \leq n-1)$ of M_j is considered to select or not the current machine with its right configuration regarding the objective. To define the objective value let's introduce some notations: let $ov_i(\sigma', j)$ be the overall throughput obtained by the j first machines using both the j^{th} machine with its i^{th} profile and the optimal configuration considering the $j-1$ first machines obtained for a target throughput of $\sigma' - \rho_{i,j}$; let $OV_i(\sigma, j)$ be a valide overall throughput and $+\infty$ otherwise ; finally let $OV(\sigma', j)$ be the optimal (minimal) throughput that exceeds the target demand σ' using a subset of the j first machines. The expression of the optimal value is the following:

$$ov_i(\sigma', j) = OV(\sigma' - \rho_{i,j}, j-1) + \rho_{i,j} \quad \text{with} \quad 1 \leq i \leq n$$

$$OV_i(\sigma', j) = \begin{cases} ov_i(\sigma', j) & \text{if } ov_i(\sigma', j) \geq \sigma' \\ +\infty & \text{otherwise} \end{cases}$$



(a) Version 1: H–HOFlt



(b) Version 2: H–HOFht

Figure 12: Schedules obtained with H–HOF

$$OV(\sigma', j) = \min\left(OV(\sigma', j-1), \min_{0 \leq i \leq n-1} OV_i(\sigma', j)\right)$$

The minimal throughput for the current period is given at the position $OV(\sigma, m)$ of the 2D matrix OV used by the algorithm. Thanks to the storage of each choice that is made for every couple (σ', j) when the algorithm is running, the algorithm is able to reconstruct the way to obtain the optimal schedule. Should two or more equivalent schedules be found, the algorithm chooses the solution with fewer machines.

As illustrated in Figure 13, H–DP minimizes the overproduction as long as possible. While the schedule found for each time period is optimal considering the preceding choices, the global schedule is not necessary optimal. This can be seen in Section 8, in which all the proposed heuristics are compared to upper bounds and among themselves through their reached production horizon.

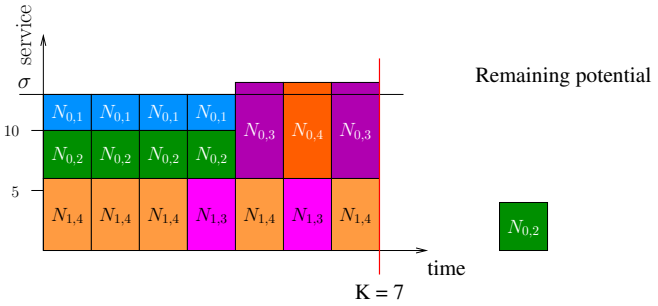


Figure 13: Schedule obtained with H–DP

Algorithm 5: H–DP: Dynamic Programming based heuristics

Data: σ: throughput demand
ρ_{max}: maximal throughput reachable with the considered set of machines
$list$: list of available couples machine-running profile $N_{i,j}$, with $RUL_{i,j} \geq 1$
$sol_{k,\sigma'}$: list of couples machine-running profile $N_{i,j}$ selected for the demand σ' for the period k
$\rho_{k,\sigma'}$: total throughput provided by the solution $sol_{k,\sigma'}$
σ_{init}: first throughput considered for each machine
σ_{cumul}: last throughput considered for each machine
input : σ
set of initial couples $N_{i,j} = (\rho_{i,j}, RUL_{i,j})$ for each machine j and running profile i
output: $sol = (sol_{1,\sigma}, sol_{2,\sigma}, \ldots, sol_{k,\sigma})$: list of solutions for each completed period

repeat
  $k \leftarrow 0$
  $sol_{k,*} \leftarrow \emptyset$
  $\rho_{max} \leftarrow \sum_{j|M_j \in \mathscr{M}} \rho_{max,j}$
  if $\rho_{max} \geq \sigma$ **then**
    Sort the machines M_j in an ascending order of their nominal throughput $\rho_{0,j}$ and in a descending order of their most sub-nominal RUL ($RUL_{n-1,j}$) for same throughputs
    $\sigma_{init} \leftarrow 1$
    foreach *machine* M_j **do**
      if M_j *is the last machine in the list* **then**
        $\sigma_{init} \leftarrow \sigma$
        $\sigma_{cumul} \leftarrow \sigma$
      else
        $\sigma_{cumul} \leftarrow \rho_{max,j} + \sum_{\text{all previous } M_{j'}} \rho_{max,j'}$
      for $\sigma' \leftarrow \sigma_{init}$ **to** $\min(\sigma_{cumul}, \sigma)$ **do**
        for $i \leftarrow n-1$ **to** 0 **do**
          $sol_{k,\sigma'} \leftarrow$ select the best solution for σ'
  Use the selected machines for one period of time
  Update the RUL of the selected machines to take their usage into account
  $k++$
until $\rho_{tot} \geq \sigma$
return $sol = (sol_{1,\sigma}, \ldots, sol_{k,\sigma})$

7.2 Enhancement: Repair step

According to the results presented in Section 8, optimal solutions can not be found using the sub-optimal approaches proposed in Section 7.1. The distance to the optimum is indeed always positive, which means that there is a scope for improvement. There is also usually remaining potential, i.e., many machines can still be used as their RUL is greater than 0 at the end of most of the schedules obtained with basic heuristics. The corresponding production horizons could then be extended by using this remaining potential even if the remaining machines can not produce enough to reach an additional period of time.

7.2.1 Strategy

We propose to enhance the results obtained with basic heuristics by performing a revision of the schedules. This can be done because the schedules are built up offline. The repair is based on a greedy algorithm and can be applied on a given schedule obtained with any basic heuristics and made up of at least one time period ($K \geq 1$). The basic idea is to swap machines whose RUL remain greater than zero for other machines used in the initial schedule whose RUL are equal

to zero for $k > K$. The recovery of these machines allows to increase the number of remaining machines that can be used in parallel and allows potentially to reach the demand for one or more additional period(s).

Let $\mathscr{M}_{RUL\neq 0}$ and $\mathscr{M}_{RUL=0}$ be two subsets of machines whose RUL are respectively greater than zero or equal to zero for $k > K$. The greedy repair algorithm consists in building $\mathscr{M}_{RUL\neq 0}$ by considering machines in the decreasing order of their RUL (for $k > K$) with the smallest size and in finding the first period k_{swap} such that every machine in $\mathscr{M}_{RUL\neq 0}$ does not appear within the period k_{swap} and it exists a subset of machines $\mathscr{M}_{k_{swap}} \subset \mathscr{M}_{RUL=0}$ that are used within period k_{swap} and $\sum_{M_j \in \mathscr{M}_{k_{swap}}} \rho_{*,j} \leq \sum_{M_j \in \mathscr{M}_{RUL\neq 0}} \rho_{0,j}$. Then the two subsets $\mathscr{M}_{k_{swap}}$ and $\mathscr{M}_{RUL\neq 0}$ are swapped. The process is reiterated until an extra period can be added to the schedule to obtain $K+1$ periods that reach σ. The algorithm ends when no more swap can be performed.

7.2.2 Illustration of the repair principle

The repair process can be seen on the following very simple example. Let's consider three machines with one running profile and the caracteristics showed in Figure 14. The schedule obtained with H–DP can be seen in Figure 15(a). One can see that the machine M_3 is never used. There is a remaining potential, but no additional period can be completed because the remaining machine is not powerful enough to reach the demand alone. The machine M_3 is not used in the first period of the schedule, so it can be exchanged with machine M_2 for one period. There is now an overproduction in the first period of the schedule, but also two different machines available. The demand can then be reached for one more period by using machines M_3 et M_2 in parallel (see Figure 15(b)). The same exchange can be done in the second scheduled period. This allows to get the machine M_2 back for one period and to increase anew the number of completed periods K by one (see Figure 15(c)). On this example, applying the repair on the H–DP schedule allows to use all the machines entirely and to extend the production horizon from $4\Delta T$ to $6\Delta T$.

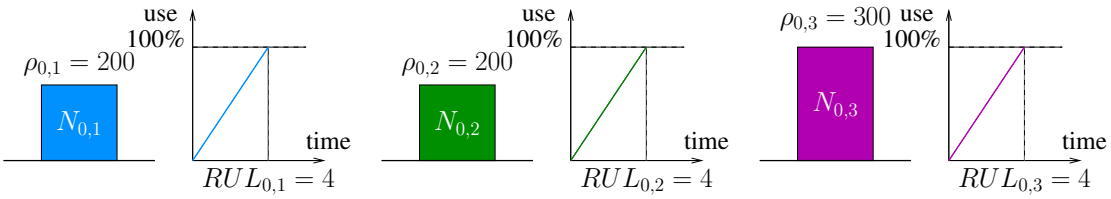


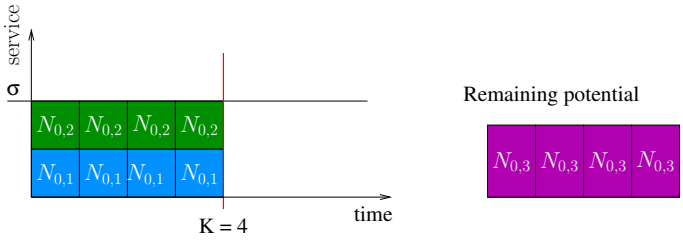
Figure 14: Set of machines for repair illustration

Repair will be performed on the results of the three most efficient heuristics, i.e., H–LRF, H–HOFht and H–DP. Using both the principle of each previously described heuristics and the repair step allows to obtain three new heuristics: H–LRF-R, H–HOF-R and H–DP-R.

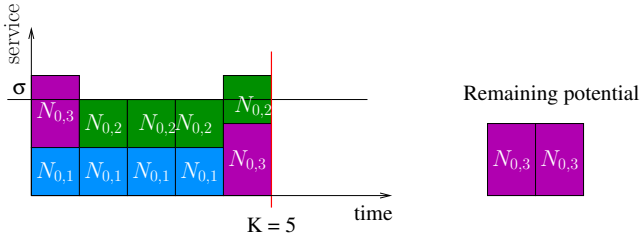
All the proposed heuristics are compared through their results in section 8. The quality of the solutions they provide is also appreciated in this same section by comparison to the lower and upper bounds defined in section 5.3.

8 Simulation results

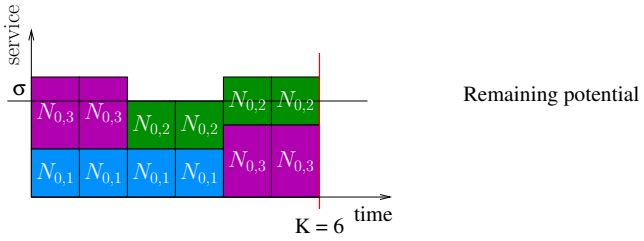
Both proposed approaches previously described (optimal and heuristics ones) have been validated on random problem instances. These have been generated using a simulator and config-



(a) H–DP schedule



(b) H–DPR - Step 1



(c) H–DPR - Step 2

Figure 15: Repair strategy

ured with many parameters. First one is the number of machines constituting the platform: $m \in \{10, 25, 50\}$. Second one sets the number of running profiles with which each machine can be used: $n \in \{1, 2, 5, 10\}$.

The demand σ_k has been assumed to be constant during the whole scheduling process. Only one demand value σ has then been associated to each problem configuration. Many demand values corresponding to different problem instances have however been tested. These values have been defined as follows:

$$\sigma = \alpha \times \rho_{tot,max}, \tag{12}$$
$$\text{with} \quad \rho_{tot,max} = \sum_{1 \leq j \leq m} \rho_{max,j}$$

the maximal total throughput reachable with the considered set of machines and α a load varying between 30% and 90%.

Each result is the average of 20 random instances corresponding to one problem configuration as defined before. Each instance corresponds to a different platform, which contains m machines randomly selected among a bank of initial machines providing different throughputs. Even if a platform may include machines with same throughputs, these machines can be differentiated by their RUL. It is indeed assumed that machines can have different states of health and therefore different RUL values at the beginning of the scheduling process.

The MAXK($\sigma \mid \rho_{i,j} \mid RUL_{i,j}$) problem is considered. All the heuristics proposed in Section 7 are first compared with each other through the number of completed periods K. The best heuristics are then compared to the optimal approach based on the ILP described in Section 6 for small size instances of the problem. For large instances, results obtained with these heuristics are compared to the upper bound KMAX defined in Section 5.3.1. Results are represented as a function of the load (α).

8.1 Comparison of heuristics

In the following figures, the number of completed periods K is represented as a function of the load $\alpha = \sigma/\rho_{tot,max}$ varying between 30% and 90%.

8.1.1 Basic heuristics

On the basis on many tests on different sets of machines, it appears that one version of H–HOF is as efficient as the other one. For the rest of the study, only the version selecting the machines having the highest throughputs first, H–HOFht, will then be considered.

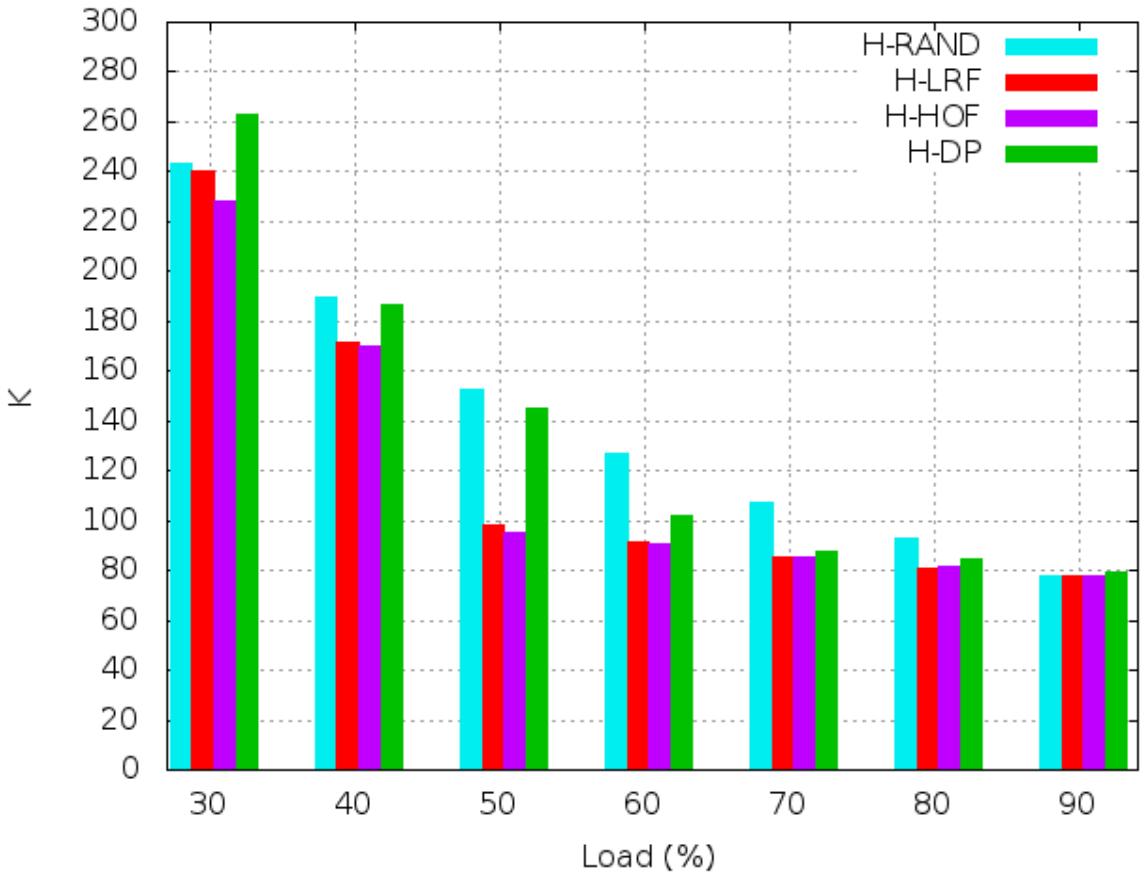


Figure 16: Average number of periods completed (K) depending on the load - $N = 1$ running profile, $M = 10$ machines

It appears that the random assignment heuristics provides results that are not so bad when only one running profile is considered. One can see in Figure 16 that H–RAND provides the greater K value for loads between 40% to 80%. This can be explained by the fact that this

heuristics does not select useless machine. The maximal overproduction is then equal to the maximal available throughput ρ_{max} minus 1. H–RAND is however not reliable for high loads ($\alpha \geq 50\%$) when the number of running profiles is increased. The number of possibilities for the choice of couples machine/profile increases with the number of profiles. If many machines are selected in profiles providing to low throughputs, the remaining machines may not be sufficient to reach the demand, even in their nominal profile. Results already decrease when taking into account two running profiles (see Figure 17). The heuristics H–RAND will then not be considered in next simulation results.

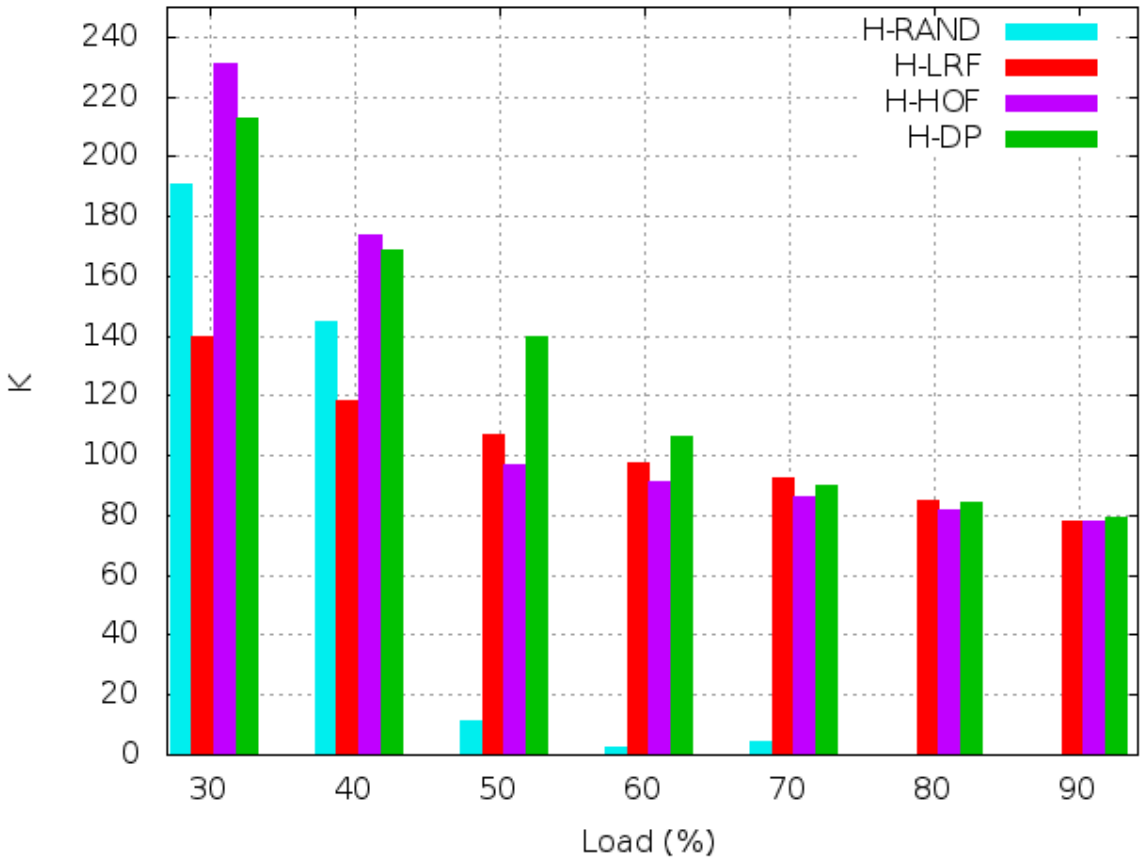


Figure 17: Average number of periods completed (K) depending on the load - N = 2 runnnig profiles, M = 10 machines

Considering many running profiles can however be interesting. One can indeed see in Figure 18 that the production horizon K increases with N when H–LRF is used.

Variation of the number of running profiles and of the number of machines seems to have no significant effect on the results provided by H–HOF (see Figures 18, 19 and20). This heuristics favours indeed the nominal running profiles. Considering the same machine, the nominal running profile provides the same throughput and is associated with the same *RUL* whatever the number of running profiles considered.

H–DP appears to give the best results for low loads α varying between 30% and 50% (see Figures 19 and 20). For high loads ($\alpha > 50\%$), the highest production horizons are obtained with H–LRF (see Figures 19 and 20).

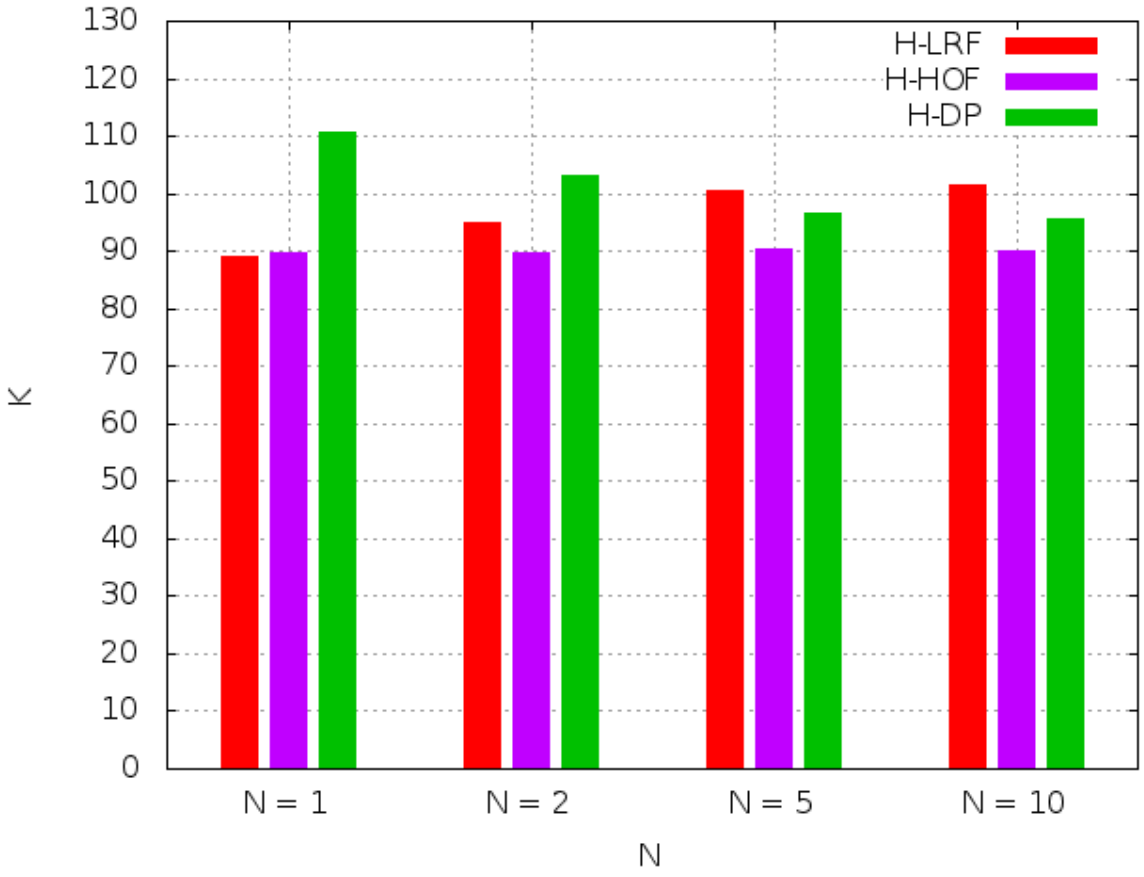


Figure 18: Average number of periods completed (K) depending on the number of running profiles (N) - M = 10 machines, load = 60%

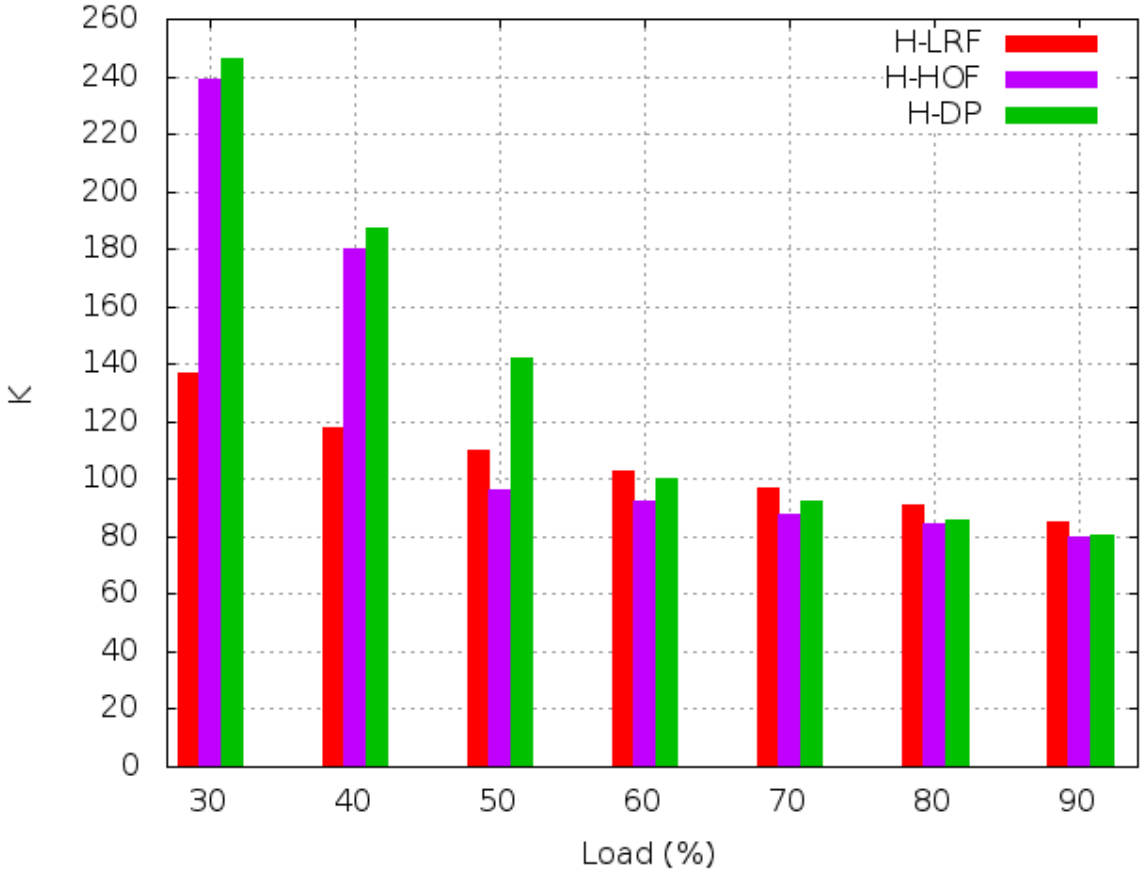


Figure 19: Average number of periods completed (K) depending on the load - N = 5 running profiles, M = 10 machines

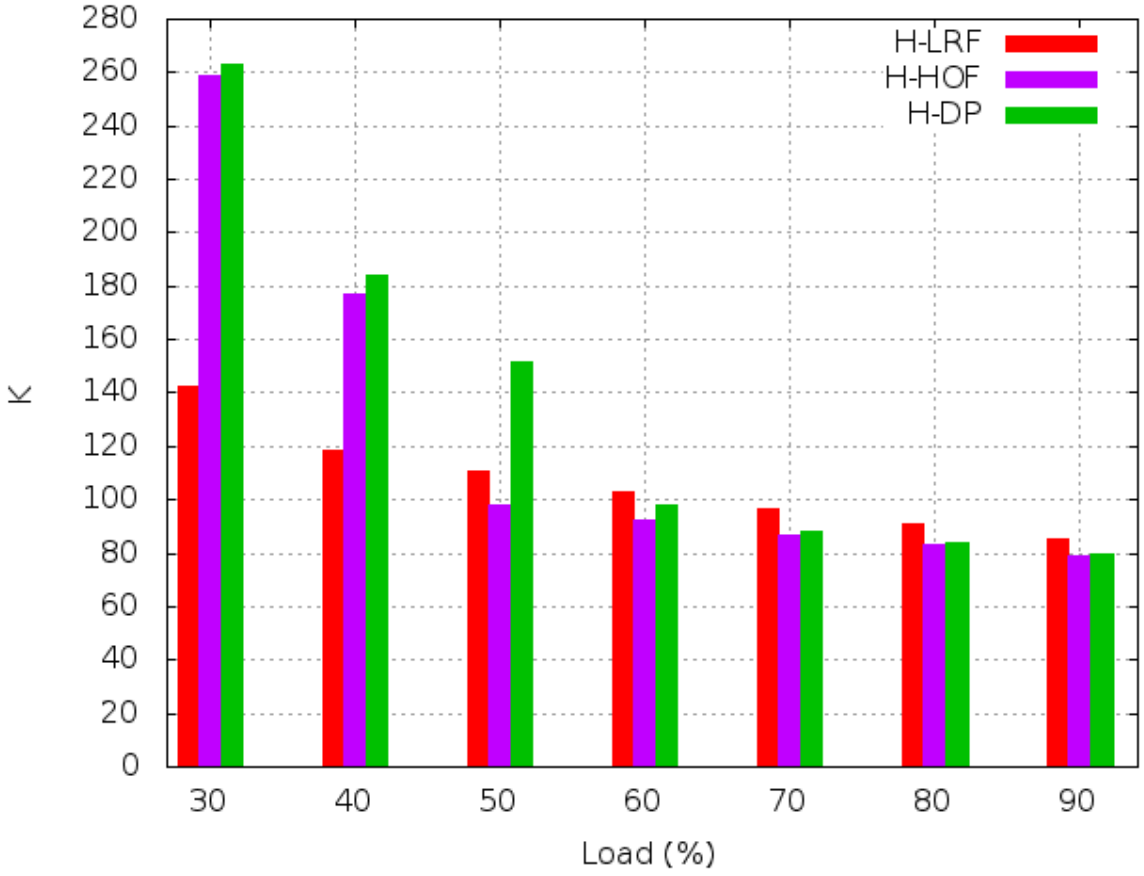


Figure 20: Average number of periods completed (K) depending on the load - N = 5 running profiles, M = 25 machines

8.1.2 Improvement obtained with repair

Results of each heuristics with repair (HR–*) are normalized with the results of the corresponding heuristics without repair (H–*). Figure 21 shows the improvement provided by the repair regardless of the efficiency of each heuristics.

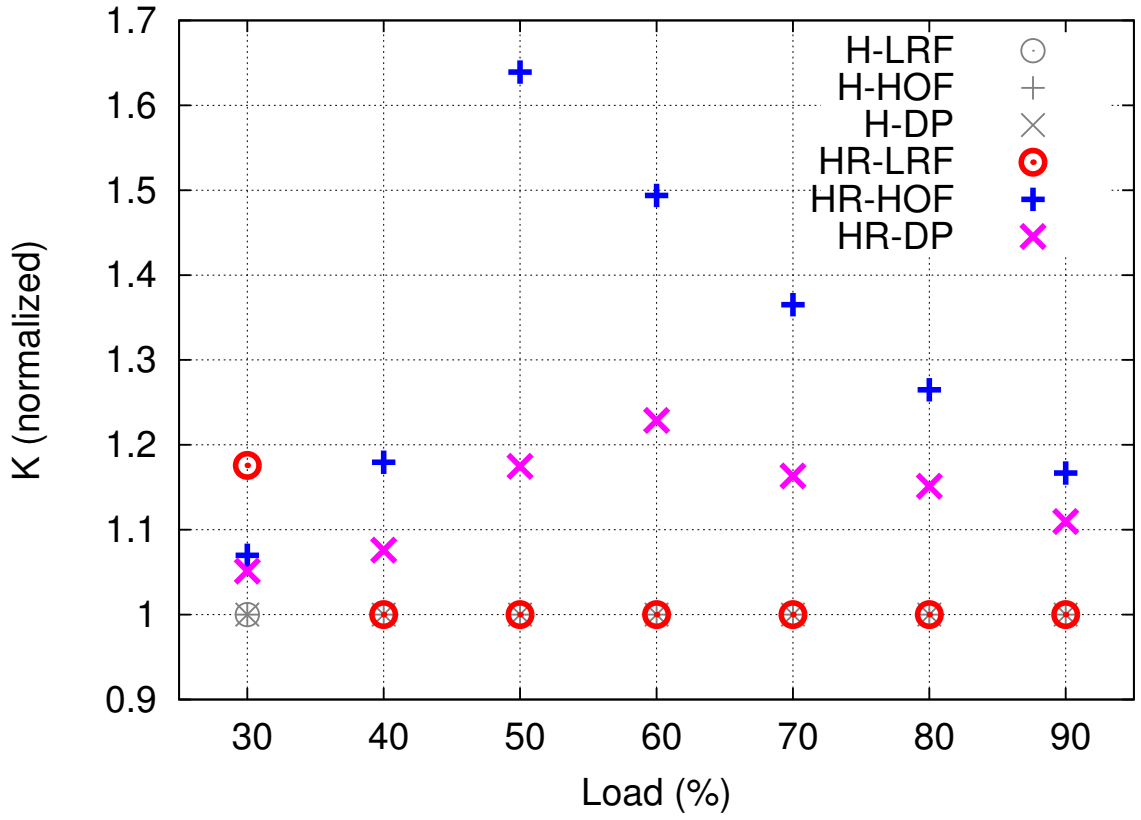


Figure 21: Improvement obtained with repair - N = 5 running profiles, M = 25 machines

A first observation reveals that quite no repair can be done on the results obtained with H–LRF. This heuristics favors indeed lowest throughputs. A significant number of machines is then necessary to reach the demand. Even if some machines remain at the end of the initial schedule, only few swaps can be done because quite all the machines are used in each period. This can also explain the decrease of the repair efficiency with high loads for all the heuristics.

The repair efficiency is maximal for H–HOF. One can see in Figure 28 that this heuristics gives the worse results without repair. Less completed periods entails more remaining potential at the end of the schedule. More swaps can then be done during the repair pocess.

8.2 Comparison to the optimal

Optimal solutions can be found in limited time for small size instances of the problem, with $N \leq 3$, $M \leq 5$ and $K \leq 20$. Results obtained with heuristics can then be compared to optimal ones only for these cases. The MAXK$(\sigma \mid \rho_{i,j} \mid RUL_{i,j})$ problem is considered in this section.

Considering the set of machines described in Figure 22, Table 2 shows the upper bound and the number of periods completed when using a linear program allowing rational solutions (K-LP) and when using an ILP (K-ILP) for many demand values σ. K-LP reaches most of the time the previously defined upper bound KMAX. As explained in Section 5.3.1, this upper bound is obtained by considering only the global potential of all the machines. The corresponding schedule complies then neither with the time discretizing nor with the maximal throughput of each machine. The LP relaxation of the ILP does not either comply with the time discretizing and the throughput values in each profile for each machine. Schedules provided by LP and by the strategy allowing to reach KMAX are not consistent with a permitted use of machines, as machines can provide rational fractions of their initial fixed throughputs during only fractions of periods. Solutions provided by the LP relaxation does however respect the maximal throughput that can be provided by each machine. This limitation explains why K-LP = KMAX−1 in some cases.

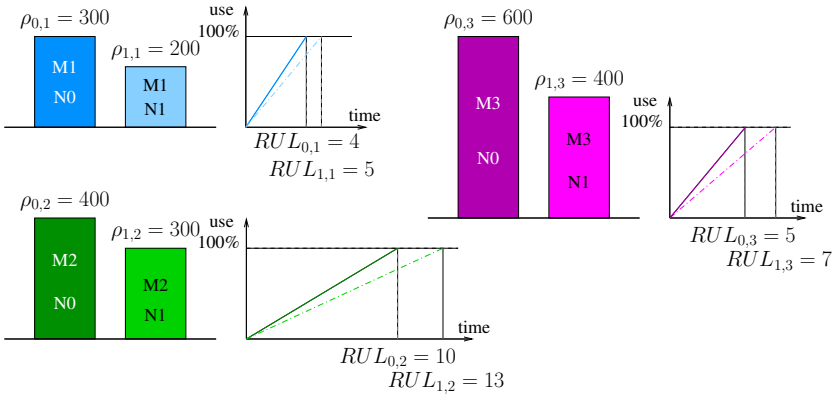


Figure 22: Initial set of machines ($M = 3$ machines, $N = 2$ running profiles)

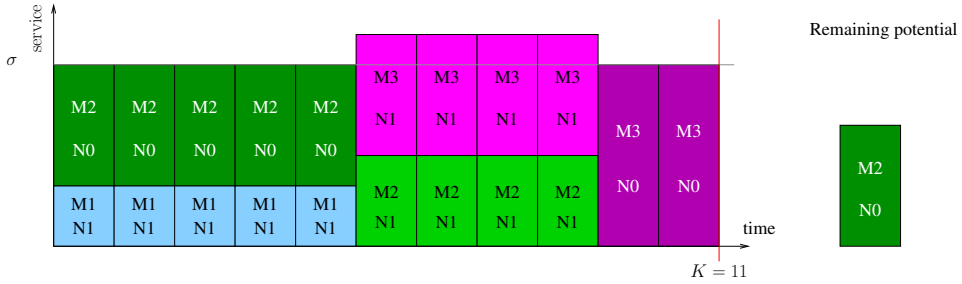
Table 2: Optimal results for many demand values σ considering the set of machines described in Figure 22

σ	KMAX	K-LP	K-ILP
300	27	27	24
500	16	16	12
600	13	13	11
700	11	11	11
800	9	8	6
900	8	7	6
1000	7	6	5

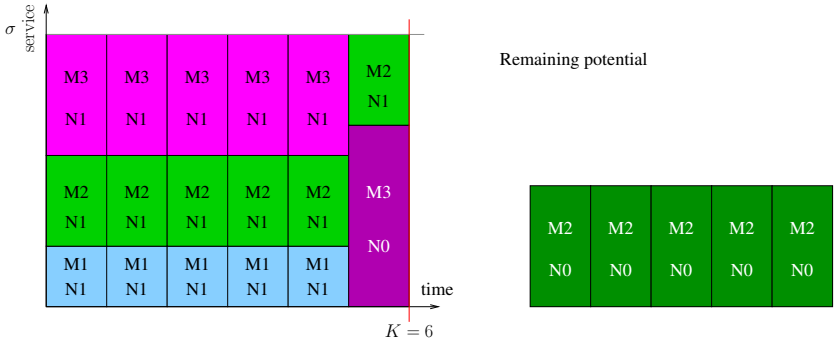
While LP uses the machines in such a way as to reach exactly the demand σ during the whole schedule horizon, the ILP complies with their fixed throughput characteristics. In some cases, this implies overproduction. For some scenarios, the ILP can provide many solutions.

In that case, the solution which minimizes the overproduction is considered in the following examples.

One can easily find optimal solutions for the MAXK($\sigma \mid \rho_{i,j} \mid RUL_{i,j}$) problem considering the machines described in Figure 22 and simple demand values. Figure 23 shows optimal solutions for two needed global throughputs $\sigma = 600\,a.ut^{-1}$ and $\sigma = 900\,a.ut^{-1}$. Similarly, Figure 24 shows sub-optimal solutions for the same cases, obtained with H–DP. The number of completed periods with H–DP is less than the optimal one, even if no overproduction is made in schedules provided by H–DP.



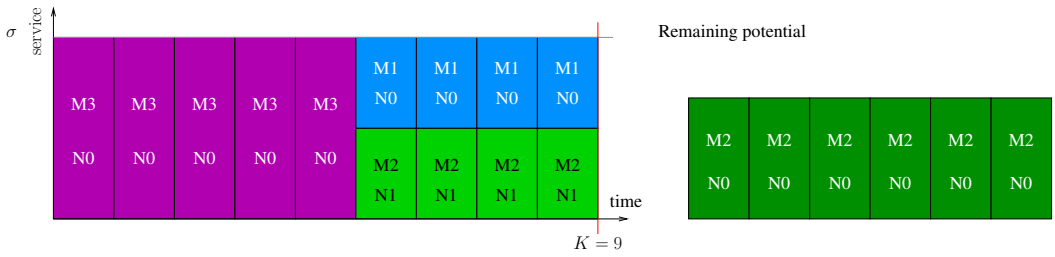
(a) ILP - $\sigma = 600\,a.ut^{-1}$



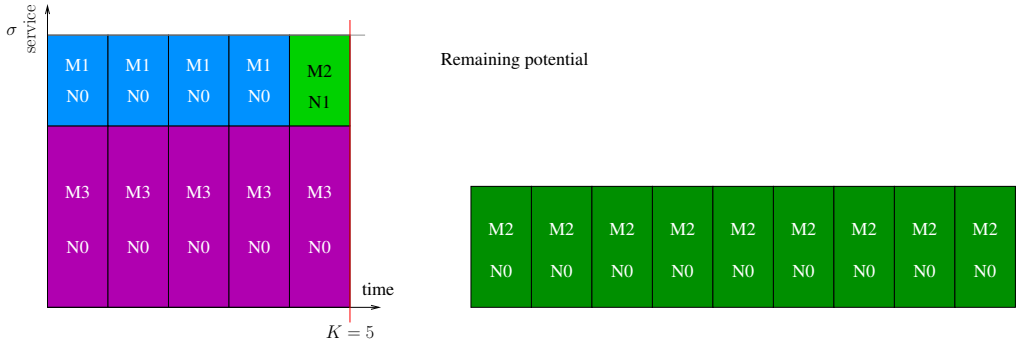
(b) ILP - $\sigma = 900\,a.ut^{-1}$

Figure 23: Optimal schedules considering the set of machines described in Figure 22

For high loads ($\alpha \geq 70\%$), K-ILP comes closer to the first lower bound KMIN$_1$ defined in Section 5.3.2. In these cases, ILP follows indeed quite the same strategy: all the machines are used in parallel to reach the demand σ.



(a) H–DP - $\sigma = 600\, a.ut^{-1}$



(b) H–DP - $\sigma = 900\, a.ut^{-1}$

Figure 24: Sub-optimal schedules obtained with H–DP considering the set of machines described in Figure 22

8.3 Comparison to an upper bound

In the following figures, distance of the number of completed periods K to the theoretical upper bound KMAX is represented as a function of the load $\alpha = \sigma/\rho_{tot,max}$ varying between 30% and 90%. Both results obtained without and with repair are represented for each heuristics. We recall that the horizon of the optimal solution is less than KMAX$\times\Delta T$. Results are then actually better than showed in the following figures.

8.3.1 Basic heuristics

One can see in Figures 25, 26 and 27 that all the heuristics excepting H–RAND are at least at 50% from KMAX, 30% for H–DP. This is promising since the upper bound KMAX is reasonably not reachable. In the best cases, H–DP is at 10% from the maximal value. With high loads and a great number of running profiles, H–LRF gets also close to 10%.

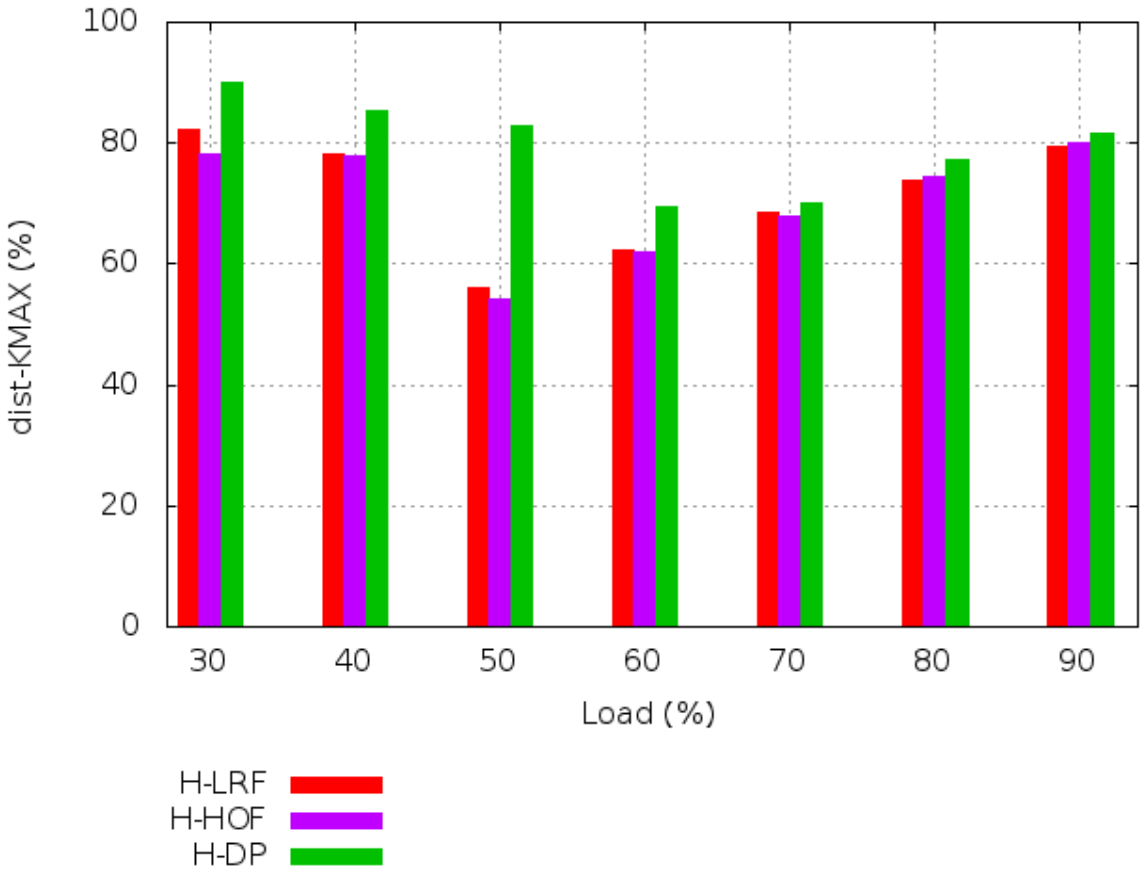


Figure 25: Distance to the theoretical maximal value (KMAX) depending on the load - N = 1 running profile, M = 10 machines

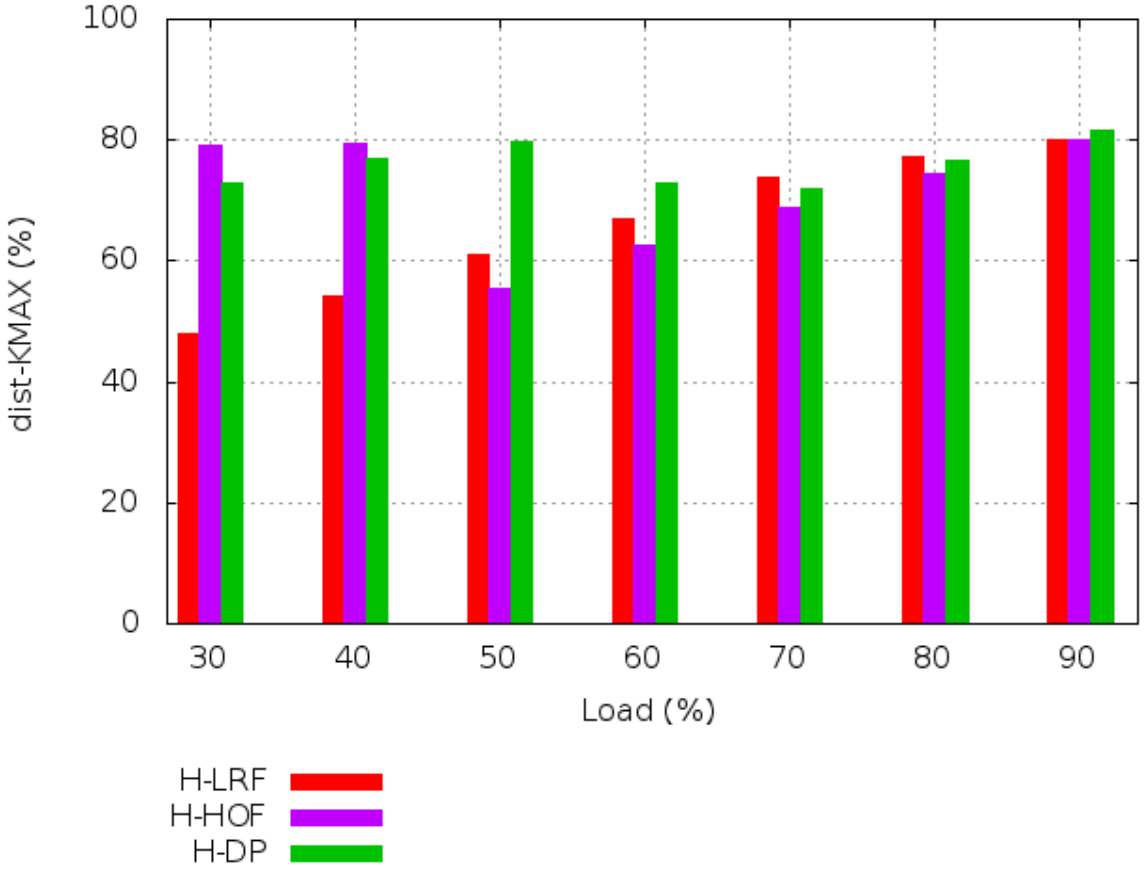


Figure 26: Distance to the theoretical maximal value (KMAX) depending on the load - N = 2 running profiles, M = 10 machines

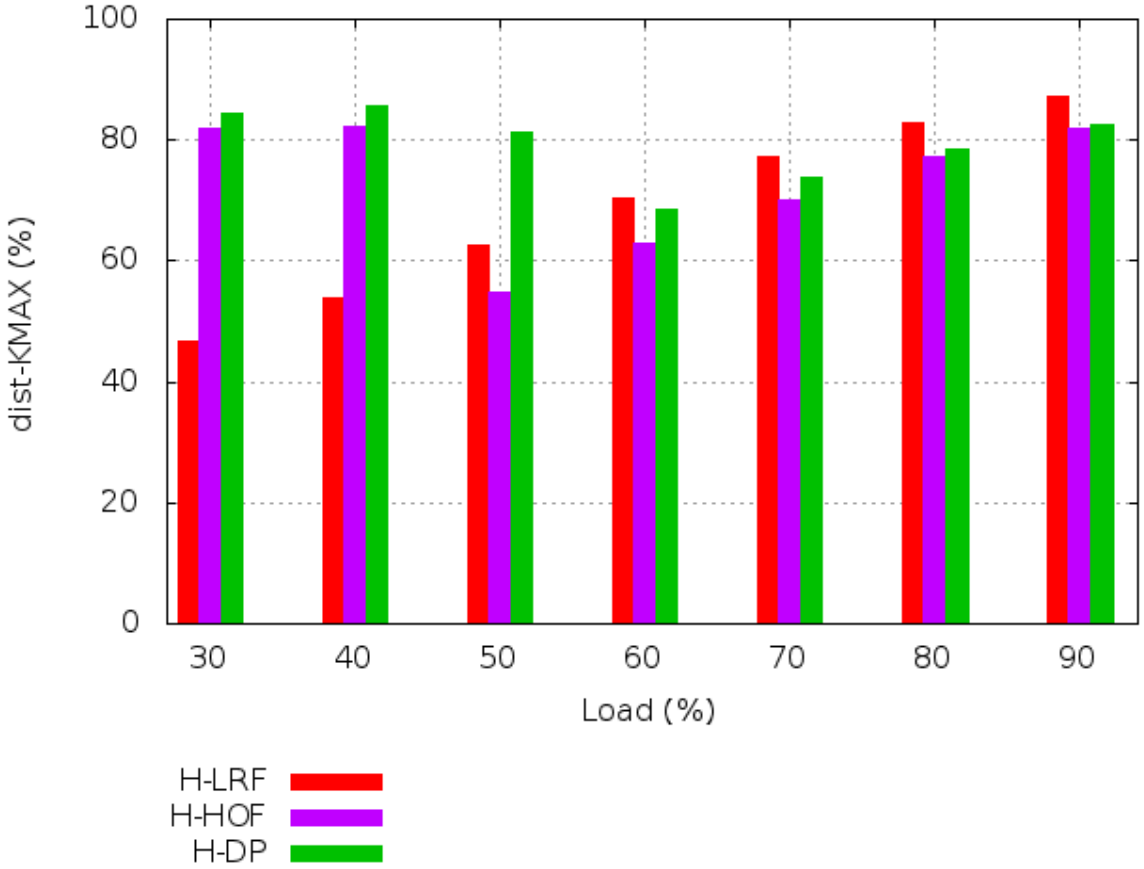


Figure 27: Distance to the theoretical maximal value (KMAX) depending on the load - N = 5 running profiles, M = 10 machines

8.3.2 Improvement obtained with repair

As discussed in previous section, repair does not improve results provided by H–LRF for loads greater than 30%. One can see in Figure 28 that the repair allows all the same to enhance results provided by H–HOF (resp. H–DP) to in average 94% (resp. 93%) from KMAX. When swaps of machines can be done, the repair enhances then results close to the theoretical maximal value whatever the efficiency of the initial heuristics.

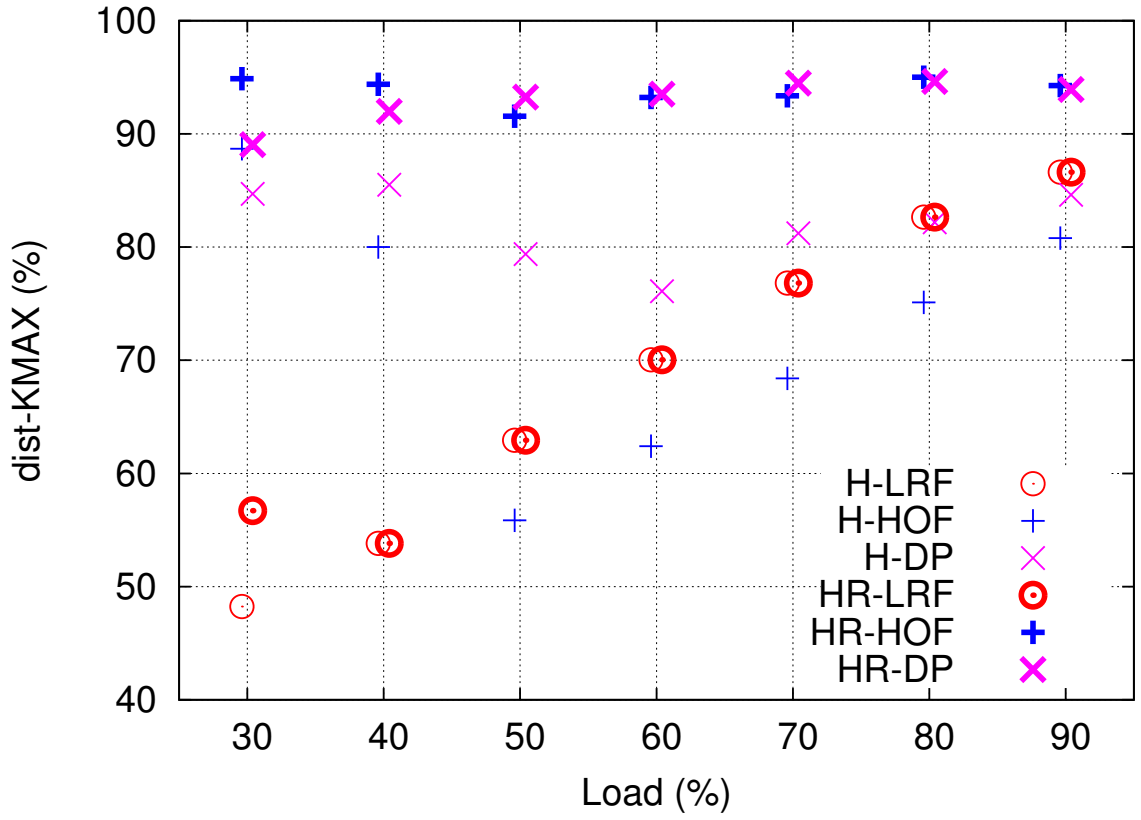


Figure 28: Improvement obtained with repair - $N = 5$ running profiles, $M = 25$ machines

Even if H–HOF provides bad results in comparison to H–DP, with repair, HR–HOF becomes as much effective as HR–DP and presents the advantages of needing little computation time (Computation parameters: Processor Intel Core™ i5-3550 CPU 3.30GHz×4, 15.6 Gio, 64 bits). Computation time increases with the load for all the heuristics. H–LRF and H–HOF provide schedules in less than 20 ms whereas H–DP needs on average 4 min. With repair, the compute times of HR–LRF and HR–HOF are in average respectively 40 ms and 80 ms. HR–DP needs 7 min in average to be processed.

9 Conclusion and future work

A new approach of scheduling using prognostics results has been investigated in this report. We have proposed scheduling algorithms using several operating conditions for each machine of a heterogeneous platform so as to extend the global operational time. We have shown that we are able to prolong as long as possible the production horizon by managing the usage of the resource thanks to the knowledge of each machine remaining useful life.

Prognostics-based scheduling has been proposed to configure sets of machines in compliance with the objective. This particular scheduling makes use of prognostics results in the form of *RUL* to adapt the provided schedule to the real state of the machines. It is part of the last step of the PHM process, i.e., Decision Making. Since the optimal solution can only be reached by running a time consuming Binary Integer Linear Program, several sub-optimal heuristics have been presented to solve the considered decision problem in polynomial time. Efficiency of these heuristics has been assessed by numerous exhaustive simulations.

As future work, we plan to explore continuous use of machines. Indeed, none of the proposed solutions guarantees that a machine will be used during its whole operational time without a planned shutdown. Taking this constraint into account is challenging in some production context. When some machines are running, as fuel cells, shutting down their production for a short period incurs extra costs.

Taking maintenance tasks into account within prognostics-based schedules is also a very interesting issue. In the best case scenario, optimization of the maintenance policy could allow to provide a steady-state scheduling.

References

[1] Siti Azirah Asmai, Burairah Hussin, and Mokhtar Mohd. Yusof. A framework of an intelligent maintenance prognosis tool. In *2nd International Conference on Computer Research and Development, ICCRD 2010*, number art. no. 5489525, pages 241 – 245, 2010.

[2] Edward Balaban and Juan J. Alonso. An approach to prognostic decision making in the aerospace domain. In *Annual Conference of the Prognostics and Health Management Society*, 2012.

[3] Edward Balaban, Sriram Narasimhan, Matthew Daigle, José Celaya, Indranil Roychoudhury, Bhaskar Saha, Sankalita Saha, and Kai Goebel. A mobile robot testbed for prognostic-enabled autonomous decision making. In *Annual Conference of the Prognostics and Health Management Society*, 2011.

[4] François Besnard, Michael Patriksson, Ann-Brith Strömberg, Adam Wojciechowski, and Lina Bertling. An optimization framework for opportunistic maintenance of offshore wind power system. In *IEEE Powertech*, 2009.

[5] Piero P. Bonissone and Naresh Iyer. Soft computing applications to prognostics and health management (phm): Leveraging field data and domain knowledge. In *IWANN*, pages 928–939, 2007.

[6] Fatih Camci and Ratna B. Chinnam. Health-state estimation and prognostics in machining processes. *IEEE Transactions on Automation Science and Engineering*, 7 (3):581 – 597, July 2010.

[7] Fatih Camci, G. Scott Valentine, and Kelly Navarra. Methodologies for integration of phm systems with maintenance data. In *IEEE Aerospace Conference*, volume 1 - 9 of *IEEE AEROSPACE CONFERENCE PROCEEDINGS*, pages 4110 – 4118, 2007.

[8] F.G. Carazas and G.F.M. Souza. Risk-based decision making method for maintenance policy selection of thermal power plant equipment. *Energy*, 35:964–975, 2010.

[9] G. Chen, K. Malkowski, M. Kandemir, and P. Raghavan. Reducing power with performance constraints for parallel sparse applications. In *Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium, Boston, USA*, 2005.

[10] Jie Chen and F. Frank Chen. Adaptive scheduling in random flexible manufacturing systems subject to machine breakdowns. *International Journal of Production Research*, 41, No. 9:1927–1951, 2003.

[11] Clemens Dietl and Uwe K. Rakowsky. An operating strategy for high-availability multistation transfer lines. *International Journal of Automation and Computing*, 2:125 – 130, 2006.

[12] Wiem Elghazel, Noureddine Zerhouni, Jacques Bahi, Kamal Medjaher, M. Hakem, and C. Guyeux. Dependability of sensor networks for prognostics and health management. Technical report, FEMTO-ST, dpt AS2M, 25000 Besançon, FRANCE, 2013.

[13] M. R. Garey and D. S. Johnson. *Computers and Intractability, a Guide to the Theory of NP-Completeness*. W.H. Freeman & Co, 1979.

[14] R.L. Graham, E.L. Lawler, J.K. Lenstra, and A.H.G. Rinnooy Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, 5:287–326, 1979.

[15] GUROBI. Optimizer reference manual, 2012.

[16] Gilbert Haddad, Peter Sandborn, and Michael Pecht. A real options optimization model to meet availability requirements for offshore wind turbines. In *MFPT: The Applied Systems Health Management Conference, Virginia Beach, Virginia*, 2011.

[17] High-performance mathematical programming solver for linear programming, mixed integer programming, and quadratic programming. `http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/`, 2012.

[18] Naresh Iyer, Kai Goebel, and Piero Bonissone. Framework for post-prognostic decision support. In *Proceedings of 2005 IEEE Aerospace Conference*, 2005.

[19] Min Ji, Yong He, and T.C.E. Cheng. Single-machine scheduling with periodic maintenance to minimize makespan. *Computers & Operations Research*, 34:1764–1770, 2007.

[20] Mohamed-Hedi Karray, Brigitte Chebel-Morello, and Noureddine Zerhouni. A trace based system for decision activities in cbm process. In *Proceedings of IEEE International Conference on Prognostics and Health Management, Gaithersburg, Maryland*, 2013.

[21] Hideaki Kimura, Mitsuhisa Sato, Yoshihiko Hotta, Taisuke Boku, and Daisuke Takahashi. Empirical study on reducing energy of parallel programs using slack reclamation by dvfs in a power-scalable high performance cluster. In *Proceedings of IEEE International Conference on Cluster Computing, Barcelona, Spain*, 2006.

[22] Andras Kovacs, Gabor Erdos, Laszlo Monostori, and Zsolt Janos Viharos. Scheduling the maintenance of wind farms for minimizing production loss. In *Proceedings of the 18th IFAC World Congress, Milano (Italy)*, 2011.

[23] M. Lebold and M. Thurston. Open standards for condition-based maintenance and prognostic systems. In *5th Annual Maintenance and Reliability Conferance (MARCON), Gatlinburg, USA*, 2001.

[24] C.-Y. Lee and V.J. Leon. Machine scheduling with a rate-modifying activity. *European Journal of Operational Research*, 128:119–128, 2001.

[25] Ying Ma, Chengbin Chu, and Chunrong Zuo. A survey of scheduling with deterministic machine availability constraints. *Computers & Industrial Engineering*, 58:199–211, 2010.

[26] Michael Pinedo. *Scheduling - Theory, Algorithms, and Systems*. Prentice Hall Series in Industrial and Systems Engineering. Prentice Hall, 1995.

[27] Thomas Rauber and Gudula Rünger. Compiler support for task scheduling in hierarchical execution models. *Journal of Systems Architecture*, 45:483–503, 1998.

[28] Peter Sandborn. A decision support model for determining the applicability of prognostic health management (phm) approaches to electronic systems. In *Proc. Reliability and Maintainability Symposium (RAMS), Arlington, VA*, 2005.

[29] Günter Schmidt. Scheduling with limited machine availability. *European Journal of Operational Research*, 121:1–15, 2000.

[30] Greg Semeraro, Grigorios Magklis, Rajeev Balasubramonian, David H. Albonesi, Sandhya Dwarkadas, and Michael L. Scott. Energy-efficient processor design using multiple clock domains with dynamic voltage and frequency scaling. In *8th International Symposium on High-Performance Computer Architecture (HPCA), Cambridge, MA*, February 2002.

[31] Jiri Sgall. *Online algorithms*, volume 1442 of *Lecture notes in computer science*, chapter 9. On-line Scheduling, pages 196–231. Springer-Verlag, Berlin, Germany, 1998.

[32] Michael A. Trick. Scheduling multiple variable-speed machines. *Operations Research*, 42:234–248, 1994.

[33] Lizhe Wang, Gregor von Laszewski, Jai Dayal, and Fugang Wang. Towards energy aware scheduling for precedence constrained parallel tasks in a cluster with dvfs. In *Proceedings of 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid), Melbourne, Australia*, 2010.

FEMTO-ST INSTITUTE, headquarters
32 avenue de l'Observatoire - F-25044 Besançon Cedex FRANCE
Tél : (33 3) 81 85 39 99 – Fax : (33 3) 81 85 39 68 – e-mail: contact@femto-st.fr

FEMTO-ST - AS2M : TEMIS, 24 rue Alain Savary, F-25000 Besançon
FEMTO-ST - DISC : UFR Sciences - Route de Gray - F-25030 Besançon cedex France
FEMTO-ST - ENERGIE : Parc Technologique, 2 Av. Jean Moulin, Rue des entrepreneurs, F-90000 Belfort France
FEMTO-ST - MEC'APPLI : 24, chemin de l'épitaphe - F-25000 Besançon France
FEMTO-ST - MN2S : 32, rue de l'Observatoire - F-25044 Besançon cedex France
FEMTO-ST - OPTIQUE : UFR Sciences - Route de Gray - F-25030 Besançon cedex France
FEMTO-ST - TEMPS-FREQUENCE : 26, Chemin de l'Epitaphe - F-25030 Besançon cedex France

http://femto-st.fr