Bi-objective optimisation approaches to Job-shop problem with power requirements

Matthieu Gondran, Sylverin Kemmoe, Damien Lamy, Nikolay Tchernev

Version of Record: https://www.sciencedirect.com/science/article/pii/S0957417420305777
Manuscript_06aae2f1373cc56cdc5da81f675b73ab

Bi-objective optimisation approaches to Job-shop problem with power requirements

Matthieu Gondran[a], Sylverin Kemmoe[b], Damien Lamy[c,*], Nikolay Tchernev[a]

[a] Laboratoire d'Informatique de Modélisation et d'Optimisation des Systèmes Université Clermont Auvergne, France (e-mail: {gondran, tchernev}@ isima.fr)
[b] Centre de Recherche Clermontois en Gestion et Management Université Clermont Auvergne, France (e-mail: Sylverin.KEMMOE_TCHOMTE@uca.fr)
[c] Mines Saint-Etienne, Institut Henri Fayol, F - 42023 Saint-Etienne, France (e-mail: damien.lamy@emse.fr)

Nowadays, a large focus is given to mass personalisation, and multiple path shop floors are suited to such production environments. Hence, this paper deals with the Job-shop scheduling problem that is used for modelling a manufacturing system. Meanwhile, a large attention is given to energy consumption of production systems, but few works consider power requirements of the production systems in order to process operations. In order to contribute in filling this gap, this paper considers the problem where the objective is to minimise both the total completion time of all operations and the instant available power required to process these operations. The problem results in the Bi-objective Job-shop Problem with Power Requirements (Bi-JSPPR). The goal of this paper is to provide a Pareto frontier of schedules minimising both criteria, considering that operations may consume a lot of power at the beginning of the process (consumption peak), more than its consumption after a while, which allows to model power profiles of manufacturing operations. To solve the problem two metaheuristic approaches are investigated: a hybrid Non-dominated Sorting Genetic Algorithm (NSGA-II) and an iterated Greedy Randomized Adaptive Search Procedure coupled with an Evolutionary Local Search (iGRASP×ELS). An efficient local search procedure is specifically designed to improve the quality of solutions in the Pareto frontier of the hybrid NSGA-II (hNSGA-II). Computational experiments and statistical tests are conducted to demonstrate the efficiency of the approaches. Results show that both approach are complementary, having the hNSGA-II showing better average performances, while the iGRASP-ELS is better when high peak power consumption are considered.

Keywords: Scheduling; Job-shop; Power Threshold; Metaheuristics; hNSGA-II; iGRASP×ELS.

1. Introduction

Optimising energy consumption is one of the major concerns for companies, which tend to improve their energy-efficiency (Gutowski et al., 2005), and also for the society that is directly impacted by any type of manufacturing processes. According to the U.S. Energy Information Administration, more than 50% of the global delivered energy in 2010 was consumed by companies ('Annual Energy Outlook 2016', 2016) which still have limited solutions at their disposal. To improve the energy consumption of production systems, (Rager, Gahm, & Denz, 2015) suggest that two types of measures can be taken: technological and/or organisational. Technological measures deal with new machines or manufacturing processes, whereas organisational measures focus on improving the

*Corresponding author

existing system, leading to energy reduction. Most of scheduling problems in the literature consist in the minimisation of the makespan (total treatment time of all operations), total-weighted tardiness and other time-related objective functions. Until recently, only a few works have been dealing with energy optimisation as an essential criterion in scheduling. However, energy minimisation in manufacturing systems is an increasing search topic in the literature as stressed in (Giret, Trentesaux, & Prabhu, 2015), where several approaches can be found including but not limited to minimisation of Time-of-Use pricings (Luo, Du, Huang, Chen, & Li, 2013; Shrouf, Ordieres-Meré, García-Sánchez, & Ortega-Mier, 2014), total energy consumption (Dai, Tang, Giret, Salido, & Li, 2013; Liu, Dong, Lohse, Petrovic, & Gindy, 2014), peak power threshold (Bruzzone, Anghinolfi, Paolucci, & Tonelli, 2012) or carbon emissions (Fang, Uhan, Zhao, & Sutherland, 2011). Recently, (Kemmoe, Lamy, & Tchernev, 2017) presented a mixed integer linear program and metaheuristic approaches for a mono-objective Job-shop with a variable power threshold. The present research work relies on a similar mathematical formalisation but its purpose is to provide solutions to a Job-shop problem considering the following two objectives: minimising the makespan and minimising the power required to realise all the operations. Two metaheuristics are proposed which aim at providing good Pareto frontiers for the Bi-objective Job-shop Problem with Power Requirements (Bi-JSPPR). The paper differs from the literature and the previous work as follows:

- The problem consists in taking into account the makespan and a power threshold as an optimisation criterion. This approach is very different from the one that minimises the total energy cost (which can actually be deduced from the schedule obtained with consideration of a power threshold) as it considers in a deeper way the input power a production system really needs, which has a direct impact in power generation for suppliers (Merkert et al., 2015) and is related to critical peak pricing approaches (Gahm, Denz, Dirr, & Tuma, 2016).

- Two metaheuristics are designed for the problem. The first metaheuristic is an evolution of the GRASP×ELS (Kemmoe et al., 2017), and called iGRASP×ELS, and the second one is based on a Non-dominated Sorting Genetic Algorithm (NSGA-II (Deb, Pratap, Agarwal, & Meyarivan, 2002)), and called a hybrid NSGA-II.

- A population-based local search is proposed for improving the performance of the NSGA-II. The hybrid NSGA-II that uses this local search proves to be more performant than the NSGA-II alone.

- A set of instances for the problem under study is proposed, ranging from 50 to 300 operations. The two metaheuristics are assessed on these instances and compared using three different criteria : the degree of Pareto optimality (Zitzler, Deb, & Thiele, 2000), the distribution of solutions along the fronts (Tan, Goh, Yang, & Lee, 2006), the hypervolume distance to a lower bound set (Ehrgott & Gandibleux, 2001; Yen & He, 2013) and the number of solutions in Pareto frontiers. The comparison includes statistical tests (two paired samples).

The objective of this paper is to provide decision makers and production managers with a tool able to provide different solutions given two conflicting objectives: productivity and energy efficiency. Such solutions related to expert systems have spread in the literature on this topic (Abedi et al. 2020; Gong et al. 2020). In the current study, the proposed approach looks at the decision problem from two points of view considering both positions of a company: supplier and customer perspectives. The main supplier factors are the productivity and customer services with delivery on time. As a customer, criteria are confidence in social responsibility, safe and durable product, and emission of pollutants i.e. power requirements. The main difference with other research projects lies in this specific environmental objective as it allows a better smoothening of the energy consumption over time. Hence, given specific situations, especially in the context of smoothening, it is possible to design a set of possible solutions the manager can rely on for scheduling operations without jeopardising productivity. Also, it can be useful if companies express a desire for energy savings and to adjust contracts with the energy provider accordingly, without compromising productivity. The designed tool can be upgraded with other specific constraints in order to further help decision making process. Also, this approach can be easily coupled with simulation tools to assess the performance of solutions considering other constraints that are difficult to discretise and consider in the optimisation module. The optimisation module relying on the iGRASP×ELS has been part of a wider industrial project, where solutions of the Pareto fronts have been evaluated by simulation in order to provide the decision maker more robust solutions.

The remaining of this paper is as follows: in the next section an overview of research works concerning energy efficient manufacturing is presented. In section 3, the problem under study is introduced. In section 4, the metaheuristics and important related algorithms are presented. To assess the reliability of the approach for the problem, a computational experiment relying on 40 instances is given in section 5. Finally, the last section is devoted to the conclusion and research directions.

2. Related works

As (Liu et al., 2014) noted, Job-shop production systems are prevalent in the industry. Hence, research on energy efficient Job-shop has developed during the past years. (Salido et al., 2016) observed that energy efficient systems offer more robustness and are less sensitive to machine failures. The correlation between makespan, energy and robustness is studied in their work that considers a Job-shop problem. Machines may have variable processing speeds. This has an impact in energy consumption: fast machining will need more energy, resulting in reduced treatment times, whereas slow machining needs less energy. With such a production system, the lost time due to a breakdown can be caught up by adjusting processing speed of the machine with repercussions on energy consumption. (Liu et al., 2014) proposed a NSGA-II to address a Job-shop problem where both the total tardiness and the total energy consumption are minimised by reducing the idle times of machines.

A Job-shop problem involving different states for machines is studied in the work of (May, Stahl, Taisch, & Prabhu, 2015). The three main objectives are the Makespan, the Total Energy Consumption and the Worthless Energy Consumption, which corresponds to the energy lost in non-productive states of machines. They proposed a Green Genetic Algorithm (GGA) to solve the problem. They also investigated other energy-oriented objectives including but not limited to the ratio between Worthless Energy Consumption (WEC) and Useful Energy Consumption (UEC), or the energy used during idle states. (Liu, Dong, Lohse, & Petrovic, 2015) worked on a Job-shop production system subject to rolling blackout policies (period of time where it is not allowed to use energy). A NSGA-II algorithm is proposed to provide Pareto frontiers where three criteria are optimised: total weighted tardiness, total electricity costs and total energy consumption. (Zhang & Chiong, 2016) proposed to solve a Job-shop problem where the objective is to minimise energy consumption and the weighted total tardiness having machines with variable speeds. A Multiobjective Genetic Algorithm (MOGA), integrating two local search procedures depending on the optimised objective, is proposed. For solutions with a good quality considering the energy criterion, makespan-based local search is applied, while energy-based local search is applied in the other case. Individuals with good quality on both criteria are improved by the two local searches, while worst solutions are not improved. This principle reduces the computational time allocated to local search phases, and the results are better than a metaheuristic without local search. (Tang & Dai, 2015) studied a Job-shop problem where machine speeds can vary in order to reduce the total energy consumption of the production system while preserving the processing order of operations which is a given data. A mathematical formalisation and a genetic algorithm are proposed to solve the problem. (Lei, Zheng, & Guo, 2016) proposed a metaheuristic in order to reduce the total energy consumption and the workload balance in a Flexible Job-shop environment with variable-speed machines. (He, Li, Wu, & Sutherland, 2015) are interested in a Flexible Job-shop where the overall system consumption must be minimised by assigning the most suitable machine to each operation and reducing idle time. A linear model is given as well as a metaheuristic (Nested Partition Algorithm – NPA). Two scenarios are applied: (i) minimising total energy consumption or (ii) minimising both the energy and the makespan. (Moon & Park, 2014) formulated two Flexible Job-shop problems. In the first one, the objective is to jointly minimise the cost related to production and electricity consumption. In the second problem, the authors investigate an approach that can reduce greenhouse gas emissions and save energy by including battery operating costs in the objective function and considering renewable energy sources. (Wu & Sun, 2018) addressed a flexible Job-shop problem where the objective is to minimise the makespan, the total energy consumption and the number of switch on/off. An NSGA-II and a green heuristic are applied. In (Gong, Deng, Gong, Liu, & Ren, 2018) a flexible job-shop is investigated where objectives are related to the minimisation of the total worker cost, the makespan, and the maximisation of a green production indicator. This indicator takes into account energy consumption, noise emissions, recycling

of tool chips and safety of operations. A hybrid Genetic Algorithm and a NSGA-II are investigated to solve the problem.

Even though the number of studies on Job-shop with the objective of reducing energy consumptions is increasing, the literature shows that energy-saving objectives are not frequently addressed (Akbar & Irohara, 2018; May et al., 2015; Salido et al., 2016). Studies on multipath shop floors mainly concern total energy consumption rather than peak power limitations as shown in Table 1 where several recent contributions (not exhaustive) that rely on population based approaches (NSGA-II, HGA, …) are exposed. Hence, in this paper an optimisation approach is proposed to minimise the power threshold and the makespan in a Job-shop-like manufacturing environment. To our knowledge, the Bi-JSPPR is one of the first attempt focusing on this issue for Job-shop manufacturing systems. It is further defined in the next section.

Table 1
Recent contributions on multipath shop floors with energy considerations.

reference	Time related criteria	Total energy consumption	Peak power limitation	Energy pricing	Other criteria	Solving approach
(Liu et al., 2014)	x	x				NSGA-II
(Liu et al., 2015)	x	x		x		NSGA-II
(He et al., 2015)	x	x				NPA
(May et al., 2015)	x	x				GGA
(Salido et al., 2016)	x	x			x	GA
(Zhang & Chiong, 2016)	x	x				MOGA
(Wu & Sun, 2018)	x	x			x	NSGA-II
(Gong et al., 2018)	x	x			x	NSGA-II NHGA
(Zeng et al., 2018)	x	x				PSO + NSGA-II+TS

3. Problem definition and notations

A lot of research attention has been given to solve the classical Job-shop problem (Ku & Beck, 2016). In such a problem, a set of jobs $j \in J$ has to be processed. Each job j consists in a succession of operations on a set M of machines. In the classical deterministic Job-shop, $n = |J| \times |M|$ operations must be scheduled. Each operation, noted O_i, $i \in [1, n]$, must be processed by one machine $m \in M$. Machines cannot process more than one operation at a time. No preemption is allowed for the operations. All the jobs and machine resources are available at the beginning of schedule. As the machines are shared among the jobs, a sequence of job operations on each machine must be defined. In the classical Job-shop an optimal solution corresponds to a schedule with the minimal makespan noted c_{max}.

In this paper, all operations have a job and machine dependent power profile that varies according to the process progression. By discretising this power profile into energy-blocs (Rager et al., 2015; Weinert, Chiotellis, & Seliger, 2011), any operation can be viewed as a succession of sub-operations. Each sub-operation corresponds to a specific power consumption during the production process. Thus, in this paper the modelling approach used by (Kemmoe et al., 2017) is adopted, which allows to represent the power profile where each operation O_i is modelled as k sub-operations that are noted $O_{i,k}$. Each $O_{i,k}$ has a processing time, noted $P_{i,k}$, and a power consumption $W_{i,k}$. Each starting date of a sub-operations $O_{i,k}$ is noted $s_{i,k}$. Depending on the value of k, this allows a generic approach by taking into account several power profiles for operations if needed. However, as stated in (Rager et al., 2015), having a precise modelling of operations increase the complexity of a problem. To reduce this complexity, an assumed loss of information is considered by addressing the case where $k = 2$. As no preemption is allowed, Time-lags are considered for modelling no-wait constraints between the sub-operations of a same operation O_i, which is similar to the approach used by (Rager et al., 2015) in the context of parallel machines with the objective of reducing the total energy consumption and not the instant power required to process operations. Finally, a variable noted w_{max} models the power threshold that must not be exceeded over time. The objective is then to find solutions minimising both c_{max} and w_{max} in such a Job-shop problem by considering the power profiles of operations. The complete mathematical formalisation is included in the Appendix.

In Figure 1, two Gantt diagrams of a same problem with two different makespans and power thresholds are exposed. In Figure 1(A) the threshold (94) leads to a better makespan (69) than in Figure 1(B) where the power threshold equals 53, and the makespan equals 77. From these figures it can be seen that the constant power threshold is not needed all over the schedule. For instance it can be reduced to 60 power units after 25 time units in Figure 1(A). It is the same for Figure 1(B), where the threshold can be reduced to 30 power units during the interval [46; 59] on the time axis. Hence, a planning manager can even go further by using a solution and adapting it in order to negotiate variable power thresholds, which can induce a reduction in pricings. However, as it will not be possible to have a better makespan for the maximum power threshold, and to preserve a good trade-off between computation time and solution quality from the viewpoint of the makespan criterion, the choice is made to focus only on the maximum power threshold. Hence, the optimisation of the final power threshold is left to a post-optimisation framework if desired.

As can be stressed, it is difficult to compare the above solutions: it is not possible to say whether the solution with the highest power threshold and a lower makespan is better or worse than the solution with the lowest power threshold and the higher makespan value. The Job-shop with one additional resource (Agnetis, Flamini, Nicosia, & Pacifici, 2011) were shown to be NP-Hard. Therefore the problem under study in this paper is also NP-Hard and it is time consuming to get one solution on the Pareto frontier. Hence, finding all exact solutions composing such a Pareto front would

take a high computational effort as stressed by (Fang et al., 2011). Because of this, the use of metaheuristics is appropriate.

From the literature review, it appears that evolutionary algorithms are widely used because of their behaviour in finding Pareto frontiers as they work on a population of solutions (Dai et al., 2013; May et al., 2015).

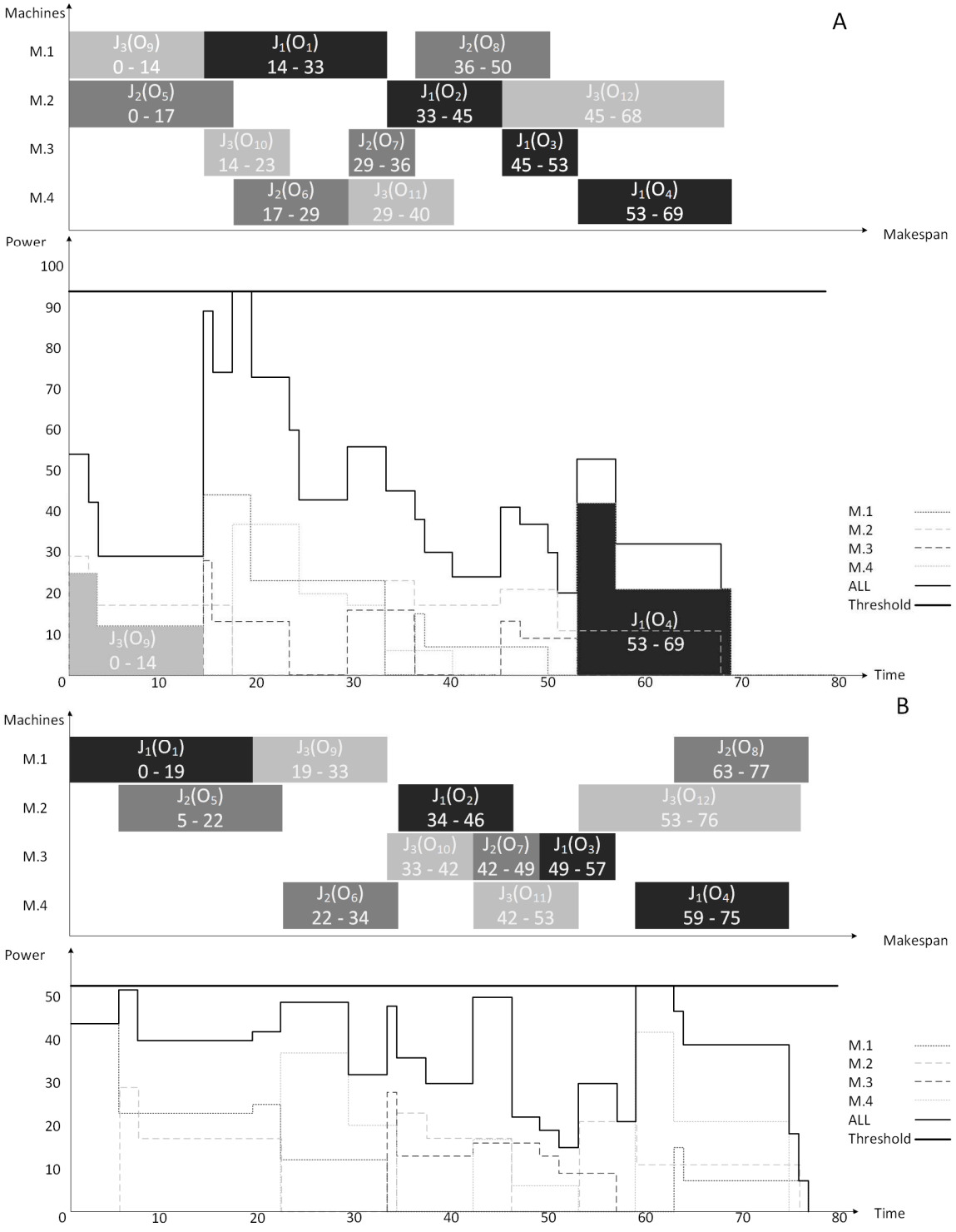

Fig. 1. Gantt charts and power loads of a problem with 94 power threshold (A) or 53 power threshold (B).

Among these algorithms, the Non-dominated Sorting Genetic Algorithm (NSGA-II) introduced by (Deb et al., 2002), is chosen in order to build the Pareto frontier of solutions for the Bi-JSPPR since this metaheuristic has shown its efficiency in solving real-world problems with up to three objectives (Bechikh, Elarbi, & Ben Said, 2017; Zhou et al., 2011). This metaheuristic is hybridised with a local search operator in order to create Pareto fronts with better solutions. Another metaheuristic is proposed, called iterated GRASP×ELS, as the GRASP×ELS has shown its efficiency in finding good solutions in past researches (Chassaing et al., 2014; Duhamel, Lacomme, Prins, & Prodhon, 2010; Kemmoe et al., 2017). These metaheuristics and related operators are presented in the following section.

4. Hybrid NSGA-II, iterated GRASP×ELS and related algorithms

As noted in the work of (Lacomme, Prins, & Sevaux, 2006), several multi-objective algorithms are available in literature and the number keeps increasing. Hence, choosing the most appropriate one for a given multi-objective optimisation problem is not obvious. In this paper two metaheuristics are developed in order to build Pareto fronts for the Bi-JSPPR. The first one is an evolution of the GRASP×ELS and is called iterated GRASP×ELS (iGRASP×ELS); the second one is based on a hybrid NSGA-II (hNSGA-II). It appears that metaheuristics generally share some common elements or procedures; in the problem under study, the coding of solutions and the evaluation algorithm are the same for the iGRASP×ELS and the hNSGA-II. Before presenting these two metaheuristics, the representation of solutions and the evaluation procedure are introduced in the following sub-section.

4.1. Solution representation and evaluation

For the Job-shop problem, a coding of solutions known as "permutation with repetition" (Bierwirth, 1995) is generally used (Liu et al., 2014; Salido et al., 2016). This vector (chromosome) noted λ in the following, consists in a succession of job numbers (the genes) where each occurrence of a job corresponds to a complete operation that must be scheduled (to take into account the no-wait constraints, all the sub-operations of an operation are scheduled at once). In Figure 1(A), the Gantt diagram can be expressed as a vector of 12 operations: [O_9;O_5;O_1;O_2;O_{10};O_6;O_{11};O_7;O_3;O_{12};O_4;O_8]. Each operation can be converted to its job number. For instance, a vector λ of a solution as presented in Figure 1(A) can be encoded this way: [3 2 1 1 3 2 3 2 1 3 1 2]. Reading the vector from left to right, the l^{th} occurrence of a job's index number refers to the l^{th} operation in the processing plan of this job. The first 3 in position 1 in the chromosome stands for the first operation of Job 3 (O_9), the second 3 in position 5 stands for the second operation of Job 3, and so on. Hence, if sub-operations are considered, a conversion of the chromosome in a list of operations would be [$O_{9,1}$-$O_{9,2}$;$O_{5,1}$-$O_{5,2}$;$O_{1,1}$-$O_{1,2}$;$O_{2,1}$-$O_{2,2}$;$O_{10,1}$-$O_{10,2}$;$O_{6,1}$-$O_{6,2}$;$O_{11,1}$-$O_{11,2}$;$O_{7,1}$-$O_{7,2}$;$O_{3,1}$-$O_{3,2}$; $O_{12,1}$-$O_{12,2}$; $O_{4,1}$-$O_{4,2}$; $O_{8,1}$-$O_{8,2}$]. The evaluation procedure of such a sequence taking into account both physical and power constraints is given in Algorithm 1.

In Algorithm 1, lines 4-8 refer to the starting date of an operation according to its job sequence (second operation of J_1 must be started after completion of the first operation of J_1). Lines 9-13 compute the starting date according to previously scheduled operations in the required machine (for instance, in Figure 1(B) the first operation of J_3 occur after the first operation of J_1 in the machine M_1). Finally, lines 14-18 compute starting dates of the operation according to the available instant power (in Figure 1(B) the last operation of J_2 starts at 63 because the last operation of J_1 has its peak power ending at 63). Hence the power threshold is considered as a constraint during the evaluation of λ and can lead to a delay in the starting dates of the operations. In the end, a predecessor operation (operation already scheduled, and that must be finished before starting the current operation) is assigned to each operation. After decoding a chromosome with the evaluation procedure presented in Algorithm 1, a solution can be expressed as a Gantt chart as in Figure 1.

Algorithm 1: Evaluation

Input/Output

S : structure storing the solution that is evaluated;
maxPower : maximum available power to schedule all operations

Variables

job, op : job treated and its operation to schedule;
machine : machine for the operation;
job_release_date, machine_release_date, *power_release_date*: temporary starting dates of an operation;
start_date : final starting date of an operation;

Begin

1. **FOR** each *job* occurrence in *S*.λ **DO**
2. *op* ≔ operation corresponding to *job*'s occurrence;
3. *machine* ≔ required machine to process operation
4. **IF** (*op* is first in *job*'s sequence) **THEN**
5. *job_release_date* ≔ 0;
6. **ELSE**
7. *job_release_date* ≔ end date of previous operation in *job*'s sequence;
8. **END IF**
9. **IF** (no operation scheduled on *machine*) **THEN**
10. *machine_release_date* ≔ 0;
11. **ELSE**
12. *machine_release_date* ≔ end date of previous operation on machine;
13. **END IF**
14. **IF** (enough power to schedule operation without exceeding *maxPower*) **THEN**
15. *power_release_date* ≔ 0;
16. **ELSE**
17. *power_release_date* ≔ date of next power release;
18. **END IF**
19. *start_date* ≔ max(*start_date_conj*, *machine_release_date*, *power_release_date*);
20. Compute all relevant information and actual makespan and store it into *S*;
21. **END FOR**

End

This evaluation procedure is used in the iGRASP×ELS and in the hNSGA-II. In the following sub-section, the hNSGA-II is first introduced.

4.3. Hybrid NSGA-II

This section describes the required components to elaborate an appropriate metaheuristic approach for the Bi-JSPPR based on a NSGA-II hybridised with a local search procedure (hNSGA-II).

The NSGA-II has been introduced by (Deb et al., 2002). What made the NSGA-II popular is its easiness to be obtained from a Genetic Algorithm; it has been successfully applied to many multi-objective combinatorial problems (Ahmadi, Zandieh, Farrokh, & Emami, 2016; Lin & Yeh, 2012; Liu et al., 2014). The result of the NSGA-II consists in a set of non-dominated solutions, the closest possible to the optimal Pareto frontier. At each iteration, the algorithm tends to improve this front using principles of genetic algorithms (crossover and mutations). Two main operators compose the NSGA-II: the non-dominated sorting procedure and the crowding tournament procedure. These procedures are not reminded here as they are not specific to the problem. For more information on NSGA-II, the reader can refer to the initial publication of (Deb et al., 2002).

4.3.1. Initialise population

The first stage in a NSGA-II metaheuristic is to generate the initial population. For the studied problem, individuals of the population are randomly generated. To this purpose, each individual is defined by a random repetition vector λ, and by a random power threshold (*maxPower* in Algorithm 1) in order to evaluate the solution, as the makespan depends on the power threshold for each solution.

4.3.2. Generation of new children solutions

In evolutionary algorithms, two important operators are used to generate offspring and diversify the population: the crossover and the mutation. These two operators are presented in the following points.

(i) Crossover

The single-point crossover based on the job occurrences in the sequence of job operations is used as the crossover operator. Given two parents P_1 and P_2, selected using a binary tournament, the operator generates two children C_1 and C_2 by the following procedure:

1. Randomly generate a point X on both parents P_1 and P_2. The position of this point is the same in both chromosomes.

2. Copy the job occurrences from the beginning of $P_1(P_2)$ until X into $C_1(C_2)$.

3. Complete $C_1(C_2)$ after X by reading $P_2(P_1)$ from left to right and by adding each occurrence of a job if the corresponding occurrence is not already present in $C_1(C_2)$.

For example, given [321132321312] and [121133322132] two feasible parent chromosomes. If X is generated in 5th position, then two children are constructed as in Figure 2.

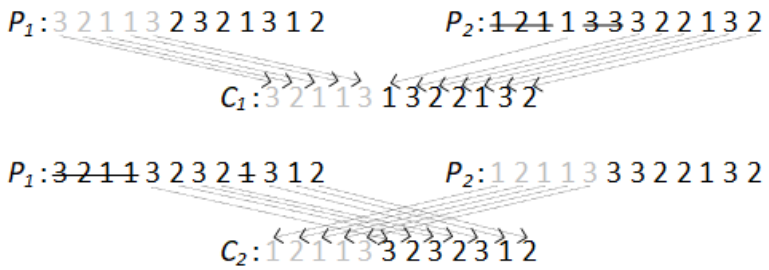

Fig. 2. Example of the construction of two children solutions.

(ii) Mutation operator

In this work, the swap mutation operator is used. In this operator, two different arbitrary genes of the children chromosomes are chosen and the values are exchanged. For instance, in the first children previously generated during the crossover procedure ([321131322132]), a mutation could consist in permuting jobs in 4th and 11th positions, resulting in the new child [321**3**31322**1**12]. For the mutation operator in the NSGA-II metaheuristic, an expected power consumption of the offspring is randomly chosen in the interval constructed with the power consumption of each parent. This expected power consumption will be used in the local search procedure presented in the next sub-section.

In single-objective optimisation, it is well known that a standard GA must be hybridised with a local search procedure to be able to compete with metaheuristics like Tabu Search (Lacomme et al., 2006). However, the local search must not lead to a premature convergence of solutions into one single point of the solution space, and then disturb the search mechanism of the multi-objective GA. The next section presents moves and the general structure of the local search procedure proposed for the problem that encompasses both criteria.

4.3.3. Local search procedure for hybrid NSGA-II

In its basis formulation, the NSGA-II does not include a Local search algorithm. This procedure aims at improving the quality of solutions while keeping the Pareto frontier well scattered. The local search phase consists in exploring the neighbourhood of a solution by considering several power thresholds in order to explore the neighbourhood of the current solution and obtain several points in the solution space to design a better Pareto frontier. This aims at exploring the neighbourhood of a solution as in Figure 3, where an example of possible improved solutions after a search process with four output solutions is given.

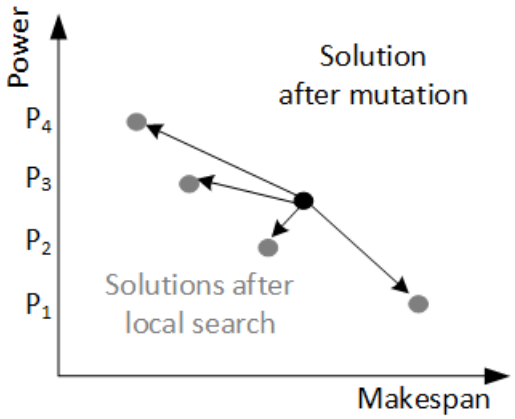

Fig. 3. Possible positions of improved solutions.

The structure of the local search procedure is given in Algorithm 2 where a population, named *paretoLS*, of non-dominated solutions is built from a given solution *S*. The size of the population constructed from *S* is equal to the number *maxIter*. In order to obtain these solutions, the power threshold evolves by step in the local search. This step is computed by considering a given ratio between the minimum and maximum power required by the solutions in the actual Pareto front (line 1). Then, the power threshold *pow* is initialised to a value lower than the power obtained randomly during the mutation phase (P_1 in Figure 3). The validity of *pow* is checked (line 2). Considering the current power threshold (*pow*), *S* is improved by doing permutations of operations in a critical path (lines 6-22). A critical path π in a schedule of a solution is a set of operations that cannot be delayed without negatively affecting the makespan.

In a critical path only specific operations can be exchanged: operations that are scheduled on the same machine (machine related disjunction), or operations delayed because of the power threshold (power related disjunction). The neighbourhood that is explored in this algorithm extends the one proposed by (Van Laarhoven, Aarts, & Lenstra, 1992) for the classical Job-shop since theirs is only relying on machine related disjunctions. This neighbourhood relies on the notion of critical block proposed by (Grabowski, Nowicki, & Zdrzalka, 1986) where a critical block is a set of consecutive operations scheduled on a same machine. In this study, the notion of machine critical block is extended to any couple of operations (O_i, O_j) if the operation O_j is delayed because of O_i. When such a couple (O_i, O_j) is detected in a critical path, the local search applies a neighbourhood operator to get a new solution (line 10). This neighbourhood operator is noted *N* and is defined as follows: Let *S* be a solution of a problem. Let O_i and O_j be two operations in a critical block *B* of a critical path π of *S*. A neighbouring solution *S'* is obtained from *S*, by reversing the processing order of O_i and O_j. At line 24, the power threshold *pow* is updated according to the power step computed at line 1, and *pow* is then used as the new power threshold in order to obtain another improved solution. This procedure is applied to each child obtained after crossover and mutation.

Algorithm 2: local-search-hNSGA-II

Input

S : sequence to explore;
maxP : maximum power threshold in the actual Pareto front;
minP : minimum power threshold in the actual Pareto front;
minPow : minimum power threshold to obtain a feasible solution;
nbStep : number of improved solutions;
maxIter : maximum number of call to Local-search procedure;

Output

paretoLS : non-dominated solutions obtained from *S* ;

Variables

tmpS, savS : temporary solutions;
pow : current power for the local search process;
stepP : step to update pow at each iteration of the search process;
cptIter : loop variable;

Begin

1. *cptIter* ≔ 0; *paretoLS* ≔ ∅; *stepP* ≔ (*maxP* – *minP*)/*nbStep*;
2. *pow* ≔ max(*minPow*; *S*.power-(*maxIter*/2)**stepP*); //*power* is given from the mutation
3. **WHILE** *cptIter* < *maxIter* **DO**
4. *tmpS* ≔ *S*;
5. evaluate *tmpS* with *pow*;
6. *op* ≔ last operation in critical path of *tmpS*; *father* ← father of *op*;
7. **WHILE** father != *0* **DO //local search phase on disjunctive arcs**
8. **IF** job(*op*) != job(*father*) **THEN** //op and father can be scheduled on different machines
9. *savS* ≔ *tmpS*;
10. permute *op* and *father* in *savS;* //*Neighbourhood N*
11. evaluate *savS* with *Pow;*
12. **IF** *savS.makespan* < *tmpS.makespan* **THEN**
13. *tmpS* ≔ sav*S;*
14. *op* ≔ last operation in critical path of *tmpS*;
15. **ELSE**
16. *op* ≔ father of *op*;
17. **END IF**
18. **ELSE**
19. *op* ≔ father of *op*;
20. **END IF**
21. *father* ≔ father of *op*;
22. **END WHILE**
23. add *tmpS* in *paretoLS*; eliminate dominated solutions;
24. *pow* ≔ *pow* + *stepP*;
25. *cptIter* ≔ *cptIter* + 1;
26. **END WHILE**
27. **return** *paretoLS*;

End

The complete hNSGA-II procedure is given in Algorithm 3, where the local search is included just after crossover and mutation (at line 6). The population *Pop* in the NSGA-II is updated by

applying the classical non-dominated sorting procedure and margin procedure on the initial population merged with all the solutions obtained after the local search process.

Algorithm 3: hNSGA-II

Output

pFront : non-dominated elements obtained during procedure;

Variables

ns : size of the populations;
Pop : population of individuals;
newPop : temporary population for crossover and mutations;

Begin

1. Generate a first population *Pop* with *ns* solutions;
2. Sort each solution of *Pop* into non-dominated fronts;
3. Compute margins of each solution in *Pop*;
4. **DO**
5. Apply crossover and mutation to generate *ns* children solutions;
6. Apply **local-search-hNSGA-II**;
6. Sort each solution of *Pop* into non-dominated fronts; //takes into account solutions after local search
7. Compute margins of each solution in *Pop*;
8. Sort *Pop* in decreasing order of *fronts* and *margins*;
9. Erase each solution that is not in the first *ns* solutions of *Pop* after sorting;
10. **WHILE** no stopping criterion is met
11. *pFront* ≔ *Pop.front*(1); //these are the non-dominated solutions in Pop
12. **return** *pFront*;

End

4.2. iGRASP×ELS metaheuristic

A second metaheuristic is used for the problem; it is based on a GRASP×ELS, which is a metaheuristic proposed by (Prins, 2009). However, the GRASP×ELS is generally used for single objective optimisation problems for which it has shown to be effective, as in (Kemmoe et al., 2017) in the context of a variable power threshold. In this study, a bi-objective problem is considered. In order to deal with the structure of the GRASP×ELS, it is chosen to apply this metaheuristic iteratively with several power thresholds (i.e. the problem is reduced to a single objective case). Hence, the GRASP×ELS is transformed into an iterated GRASP×ELS (iGRASP×ELS) in which a criterion (the makespan) is optimised while considering the other one fixed (the power threshold). This iGRASP×ELS is not presented in this paper as it does not differ very much from the initial GRASP×ELS proposed in (Kemmoe et al., 2017) for a variable power threshold. However some points must be highlighted. Indeed, each GRASP×ELS consists in searching one point in the Pareto frontier. The best solution found during a GRASP×ELS phase is inserted in the Pareto front if it is a non-dominated solution. All solutions that may be dominated are erased from this population. The way the power threshold is computed from one GRASP×ELS phase to another consists in adding a step value equal to the difference between a maximal power threshold that is computed with a first execution of a GRASP×ELS without considering the power threshold objective (*maxP*), and the

minimal power threshold under which no solution exists (*minP*). This value is then divided by the maximum allowed size of the Pareto frontier (*paretoMaxSize*): $step = \frac{maxP - minP}{paretoMaxSize}$.

5. Computational experiment

5.1. Experimental settings

Both the proposed hNSGA-II and iGRASP×ELS have been implemented in C++, and tested on a PC with an Intel Core i7-4800MQ 2.7GHz, running Windows 7 operating system. If several papers have addressed energy efficiency in scheduling, they do not clearly consider the power profiles of operations and they are focused on minimising total energy consumption rather than peak power consumption. This implies the definition of new instances and specific methods. Hence, as no instance of reasonable size existed in the literature for the considered problem, the generation of test instances was required. A set of 40 standard benchmark instances with 50 to 300 operations, designed by Lawrence (Lawrence, 1984), is available from the OR-library for the classical JSSP. However, these instances do not take into account power requirements of operations. Therefore, this dataset is adapted to the Bi-JSPPR by introducing additional factors such as duration and consumption of the power peak, and nominal power consumption after the end of the peak. This new dataset is available on http://damienlamy.com/works/Energy/JSPPR/BiObjective/.

The original processing times of operations are kept the same and are used as the total processing time of operations. The power data for these instances have been randomly generated by considering that given the total duration of an operation P_i, the nominal power consumption $W_{i,2}$ and the peak power consumption $W_{i,1}$ are expressed as follows: $W_{i,2}$ is uniformly distributed on the closed interval $[0; \frac{P_i}{2}]$. $W_{i,1}$ is uniformly distributed on the closed interval $[W_{i,2}; P_i]$. The duration of the peak power consumption ($P_{i,1}$) is in $[0; \frac{P_i}{3}]$. For each instance, and for each metaheuristic, 5 replications have been made.

5.2. Adopted performance measures

To assess the quality of a multi-objective optimisation algorithm, different measures can be considered than the only criterion used in single objective optimisation (Collette & Siarry, 2002). In the present work, four measures are used: *Hyperarea Ratio (HR)* as in (Collette & Siarry, 2002), the *degree of Pareto optimality* (*RN),* the *distribution of solutions along the Pareto front* (*TS*) as in (Zhang & Chiong, 2016), and the number of non-dominated solutions (NS^M) in a given Pareto front PS^M. The first three measures, which require explanations, are reminded below:

(1) The measure of the deviation between a Lower Bound Set (LBS) and Pareto fronts through hypervolume comparison: *Hyperarea Ratio* (*HR*). Each element of the LBS (m, w) is computed using

the formula: $m = \frac{\sum_{i \in V} \sum_{k \in SO_i} P_{ik} W_{ik}}{w}$, where w is a possible value for the power threshold, ranging from the minimum power threshold (required to execute any operation) to the power threshold where the problem is equal to the classical Job-shop. This approach allows a simple estimation of the LBS. The referenced point for computing the hypervolumes *HV* is *nadir,* which is obtained for all Pareto fronts given by an algorithm *M* on an instance (i.e. *nadir* is the point formed with the worst makespan and the worst power of each Pareto front returned by *M*). Finally, HR is computed as follows: $HR = 100 * \frac{HV^{LBS} - HV^{M}}{HV^{LBS}}$. This value is computed for each replication of a metaheuristic *M* in order to have average values on all replications.

(2) The degree of Pareto optimality (*RN*) of the obtained solutions: $RN^M = \frac{NS^{M^*}}{NS^M}$, where NS^{M^*} denotes the number of solutions in PS^M that are not dominated by any solution of another algorithm. This measure is also called *C*-metric in (Zitzler et al., 2000).

(3) The distribution of solutions along the Pareto front (*TS*), with $TS^M = \frac{1}{\overline{D}} \sqrt{\frac{1}{NS^M} \sum_{i \in PS^M} (D_i - \overline{D})^2}$, where D_i is the Euclidian distance in the search space between the solution *i* and the nearest solution *i'* in PS^M: $D_i = \sqrt{\left(Obj_1(i) - Obj_1(i')\right)^2 + \left(Obj_2(i) - Obj_2(i')\right)^2}$ and with $\overline{D} = \frac{1}{NS^M} \sum_{i \in PS^M} D_i$. A smaller value of *TS* suggests that solutions are more evenly distributed in the Pareto front. This metric is related to the spacing metric of (Tan et al., 2006).

5.3. Tuning parameters

The parameter values for the hNSGA-II are set as follows: the population size n = 50; the stopping criterion is a number of iterations iter_max=1000; the crossover probability cp = 0.8; the mutation probability mp = 0.2. These parameters were obtained after a Design of Experiment (DOE) consisting in testing different parameters on a subset of all the instances. This subset consists in 8 instances with different number of jobs or machines. Tested parameters are as follows: n={50; 100}, *iter_max*={500; 750; 1000}, cp={0.6; 0.7; 0.8; 0.9}, mp={0.1; 0.2; 0.3} resulting in 72 possible configurations. The average values of *NS*, *RN* and *TS* are computed for each set and the configuration with the best overall performances for the hNSGA-II is selected for all the other instances. For the Local Search, the number of steps (*nbStep*) is experimentally fixed at 40, while the maximum number of explored solutions (*maxIter*) is fixed at 20 to avoid large computation time. A maximum computation time for each execution is set at 1500 seconds.

For the iGRASP×ELS, the same approach is used with tested parameters as follows: the number of GRASP iterations (*nb_grasp*) is ranged from 20 to 40 with steps of 10. The number of ELS iterations (*nb_els*) is ranged from 75 to 150 with steps of 25, and the number of neighbours generated (*nb_n*) is ranged from 20 to 25 with steps of 1. This DOE resulted in the following parameters:

nb_grasp=20, *nb_els*=100, and *nb_n*=23. As this metaheuristic is not dedicated to bi-objective problems, it is applied iteratively with different power thresholds. To avoid extended computation times, a time limit for each GRASP×ELS is set to 50 seconds. Each power threshold is computed by adding a step value computed as in section 4.2. The algorithm stops after finding 30 different solutions (i.e. $paretoMaxSize$). Hence, the maximum computation time for an execution is 1500 seconds.

Because of the selected parameters, the average computation time of all instances is equal for both metaheuristics.

5.4. Computational results on the hNSGA-II and the iGRASP×ELS

Some frequently used notations are reproduced here for quick reference:

$\overline{HR}$: average hypervolume deviation between computed fronts and Lower Bound Set;

$\overline{TS}$: average distribution of solutions along the Pareto front;

$\overline{RN}$: average degree of Pareto optimality.

$\overline{NS}$: average number of non-dominated solutions.

The results are presented in Table 2 where the column *Ins.* represents the instance treated. The number of jobs is noted nbJ, the number of machines is noted nbM, while n is the number of operations. $\overline{HR}$ represents the average value for HR (area deviation between Lower Bound Set and Pareto fronts); $\overline{TS}$ is the average value for TS (evenness of the distribution of solutions along the Pareto front) and $\overline{RN}$ is the average value for RN (degree of Pareto optimality). Finally, $\overline{NS}$ denotes the average number of non-dominated solutions. Best values are in bold.

(1) Focusing on the $\overline{HR}$ column, it can be seen that the hNSGA-II is closer in average to the Lower Bound Set (LBS) than the iGRASP×ELS. Results are better on more than 77% of the instances. After conducting a paired sample t-test, results show that there is a difference between the two approaches as hypothesis *H0* (i.e. mean differences equal to 0) can be rejected (p-value < 0.0001). However, it should be mentioned that, in order to have comparable computation times between the two approaches, the iGRASP×ELS returns less values in the Pareto fronts. This obviously has an impact in the computation of hypervolumes as less area is covered by the solutions in the front. Two scenarios can be considered to improve the quality of returned fronts in the iGRASP×ELS: computing more solutions (increase $paretoMaxSize$), or allowing a larger search time for each point. In both cases the Pareto frontiers are improved, at the cost of an increased computation time which is a difficult trade-off to deal with in optimisation algorithms.

Table 2

Results obtained with hNSGA-II and iGRASP×ELS.

				hNSGA-II				iGRASP×ELS			
Ins.	nbJ	nbM	n	$\overline{HR}$	$\overline{TS}$	$\overline{RN}$	$\overline{NS}$	$\overline{HR}$	$\overline{TS}$	$\overline{RN}$	$\overline{NS}$
la01_jsppr	10	5	50	**27.80**	1.01	**0.69**	**33.4**	31.71	**0.64**	0.02	21
la02_jsppr	10	5	50	27.81	0.75	0.19	**27.2**	**26.92**	**0.66**	**0.37**	20.8
la03_jsppr	10	5	50	26.61	0.69	**0.35**	**28.6**	**25.25**	**0.59**	0.18	17.8
la04_jsppr	10	5	50	21.24	1.27	**0.33**	**45.8**	**21.07**	**1.09**	0.32	23.4
la05_jsppr	10	5	50	**48.27**	0.68	**0.27**	**8.4**	50.80	**0.40**	0.03	6
la06_jsppr	15	5	75	**33.12**	0.95	**0.62**	**27.6**	36.67	**0.53**	0.07	16.4
la07_jsppr	15	5	75	**32.44**	0.71	**0.76**	**22.8**	38.77	**0.64**	0	13.8
la08_jsppr	15	5	75	**27.52**	0.84	**0.51**	**29.6**	30.33	**0.79**	0.06	17.6
la09_jsppr	15	5	75	**34.70**	0.83	**0.53**	**15.6**	39.97	**0.53**	0	10.8
la10_jsppr	15	5	75	**34.47**	0.66	**0.58**	**25**	37.56	**0.59**	0.03	15.4
la11_jsppr	20	5	100	**26.98**	0.77	**0.81**	**32.8**	31.80	**0.72**	0.02	21
la12_jsppr	20	5	100	**31.41**	0.74	**0.65**	**24**	35.03	**0.56**	0	13.6
la13_jsppr	20	5	100	**25.34**	0.87	**0.80**	**31.4**	29.26	**0.77**	0	19
la14_jsppr	20	5	100	**27.62**	1.08	**0.64**	**35**	29.89	**0.57**	0.06	20.4
la15_jsppr	20	5	100	**26.67**	1.12	**0.71**	**25,4**	32.03	**0.67**	0	16.6
la16_jsppr	10	10	100	**13.00**	**0.69**	**0.81**	**50**	13.88	0.80	0.24	17.6
la17_jsppr	10	10	100	**13.50**	**0.71**	**0.80**	**50**	15.78	0.83	0.20	21.4
la18_jsppr	10	10	100	**12.79**	**0.69**	**0.82**	**50**	14.59	0.80	0.19	20.8
la19_jsppr	10	10	100	**14.22**	**0.61**	**0.86**	**50**	16.06	0.86	0.23	20.4
la20_jsppr	10	10	100	**11.95**	**0.65**	**0.76**	**50**	13.16	0.87	0.36	20.6
la21_jsppr	15	10	150	**13.49**	**0.64**	**0.89**	**50**	14.56	0.99	0.25	24.8
la22_jsppr	15	10	150	13.97	**0.63**	**0.80**	**50**	**13.40**	0.84	0.30	23.4
la23_jsppr	15	10	150	**11.94**	**0.62**	**0.86**	**50**	13.76	0.80	0.08	24.8
la24_jsppr	15	10	150	**13.48**	**0.63**	**0.83**	**50**	14.44	0.87	0.14	24.2
la25_jsppr	15	10	150	**17.13**	**0.69**	**0.98**	**50**	18.30	0.79	0.28	24.2
la26_jsppr	20	10	200	**12.08**	**0.65**	**0.91**	**50**	13.26	0.93	0.21	24.6
la27_jsppr	20	10	200	15.18	**0.59**	**0.80**	**50**	**14.43**	0.86	0.34	24.2
la28_jsppr	20	10	200	11.60	**0.58**	**0.84**	**50**	**11.57**	0.93	0.29	23.6
la29_jsppr	20	10	200	**16.56**	**0.61**	**0.94**	**50**	17.50	0.95	0.19	24.6
la30_jsppr	20	10	200	**14.37**	**0.67**	**0.87**	**50**	15.52	0.90	0.23	26.8
la31_jsppr	30	10	300	**15.86**	**0.64**	**0.86**	**50**	18.21	0.74	0.18	24.4
la32_jsppr	30	10	300	11.59	**0.68**	0.56	**50**	**11.49**	1.01	0.56	27
la33_jsppr	30	10	300	**14.75**	**0.61**	**0.74**	**50**	15.59	0.84	0.30	24.4
la34_jsppr	30	10	300	15.95	**0.63**	**0.75**	**50**	**15.51**	0.90	0.48	24.4
la35_jsppr	30	10	300	**15.24**	**0.66**	**0.65**	**50**	15.73	0.79	0.35	23.8
la36_jsppr	15	15	225	**13.13**	**0.68**	**0.91**	**50**	14.15	0.94	0.21	23.8
la37_jsppr	15	15	225	**12.27**	**0.58**	**0.82**	**50**	12.91	0.98	0.27	26
la38_jsppr	15	15	225	**14.23**	**0.60**	**0.91**	**50**	14.61	0.91	0.30	23.6
la39_jsppr	15	15	225	**11.59**	**0.66**	**0.93**	**50**	11.85	0.99	0.27	23.8
la40_jsppr	15	15	225	12.12	**0.69**	**0.74**	**50**	**12.04**	0.94	0.29	24.6
Average:				**19.85**	**0.73**	**0.73**	**41.57**	21.48	0.80	0.20	21.14

(2) Focusing on the $\overline{RN}$ column, it can be stressed that the solutions found by the hNSGA-II are of higher quality than the solutions achieved by the iGRASP×ELS. Remember that this criterion corresponds to the percentage of solutions that are not dominated by any solution of any run of the other metaheuristic. Considering this criterion, Pareto frontiers are better in almost all the instances with the hNSGA-II. Furthermore, in average, 80% of the iGRASP×ELS solutions are dominated by at least one of the solutions provided by the hNSGA-II. In the meantime, only 26% of all solutions of the hNSGA-II are dominated by at least one of the iGRASP×ELS solutions. The statistical test confirms the observation, as hypothesis *H0* can be rejected (p-value < 0.0001). The Figure 4 summarises the

results for the $\overline{RN}$ criterion. In this figure, the rectangles correspond to the average value of $\overline{RN}$, and the error lines correspond to the standard deviation among these results. The figure shows that the hNSGA-II is clearly better on the dataset. However, the $\overline{RN}$ criterion, considered as an aggregated result, does not show that the iGRASP×ELS performs better when the power threshold is high and where the hNSGA-II has difficulties in finding solutions as stressed in Figure 5. This last point implies that the iGRASP×ELS, and hence the GRASP×ELS, can be used when searching solutions with low makespan values and high power consumptions as discussed in the following.

(3) Focusing on the $\overline{TS}$ column, it appears that solutions obtained with the hNSGA-II are distributed more evenly than solutions found by the iGRASP×ELS. The average *TS* value corresponding to the iGRASP×ELS is larger than the one of the hNSGA-II (0.80 compared to 0.73). Actually, the hNSGA-II provides better scattered Pareto frontiers in more than 60% of all the instances. This is due to the specific NSGA-II procedures (ranking and margin) that help having good Pareto fronts. However, statistical test shows that hypothesis *H0* cannot be rejected and so results are comparable on this criterion.

(4) Finally, considering the average number of non-dominated solutions in the different Pareto fronts, the hNSGA-II shows better average performances. This is mainly due to the behaviour of the algorithms, as the iGRASP×ELS searches for solutions according to a pre-computed step value. This step value is designed empirically to avoid large computation times for this metaheuristic, as it would be very difficult to obtain a valuable Pareto front in an acceptable duration if all possible thresholds were to be tested. Also, it allows to have the same overall computational time with the hNSGA-II for better comparisons. Hence, a trade-off has been considered between the number and the duration of each iGRASP×ELS run to build the Pareto frontier. Meanwhile, the NSGA-II does not suffer from this constraint, and the non-dominated sorting procedure ensures a better management of the population. In more than 62.5% of test cases, the whole final population constitutes the Pareto front, which ensures a large number of alternatives for a decision maker.

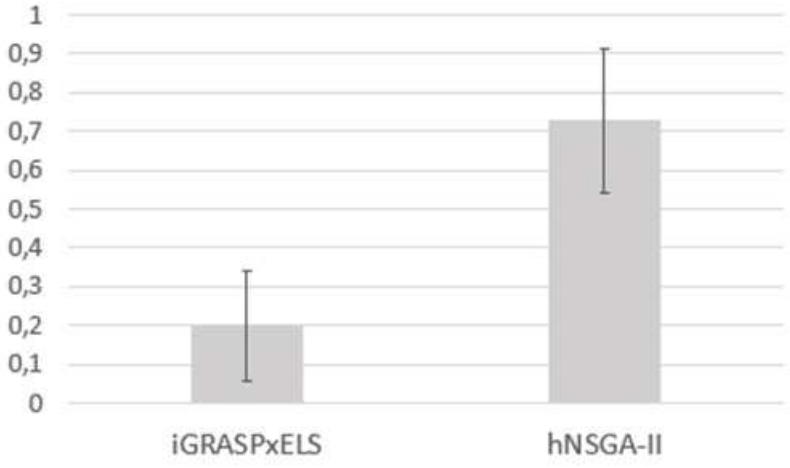

Fig. 4. Average values for the RN criterion (blue rectangles) and standard deviations (error lines).

Considering the different criteria used to compare the two approaches, results show that the hNSGA-II is in average better than the iGRASP×ELS because of three criteria: the $\overline{RN}$, the $\overline{HR}$ and $\overline{NS}$ values. The overall better performance of NSGA-II allows to provide decision makers with larger sets of possible solutions. In both cases, multicriteria decision approaches can be applied to choose the best suited solutions to a given problem.

As can be stressed, even though the hNSGA-II apparently outperforms the iGRASP×ELS in most of cases, this last metaheuristic seems more effective on some instances (i.e. la27_jsppr, la34_jsppr, la40_jsppr – according to $\overline{HR}$ criterion). Actually, the iGRASP×ELS still has several advantages that are not visible in the presented results. Indeed, in almost all instances this metaheuristic presents better results when considering high power thresholds as stressed in Figure 5.

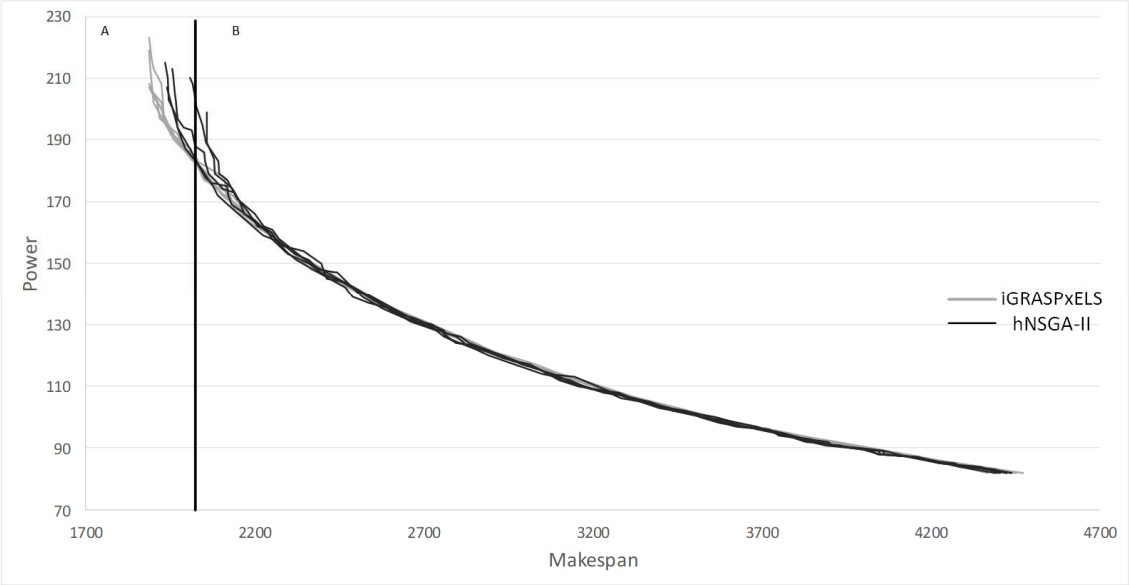

Fig. 5. Pareto fronts obtained after 5 replications of each metaheuristic on la35_jsppr.

Figure 5 shows an example of five different Pareto fronts per metaheuristic obtained with a typical execution of both algorithms on instance la35_jsppr. As can be stressed, the iGRASP×ELS finds better average solutions when the power threshold is high (part A of the figure), whereas the hNSGA-II finds better average solutions in the second part (part B), and is better considering all three criteria.

To complete the analysis of performance when high power thresholds are considered, and to further assess performances between multi-objective metaheuristics and metaheuristics specifically designed to address single objective problems, an investigation can be conducted on the extreme points of the Pareto front as suggested in (Abedi, Chiong, Noman, & Zhang, 2020). In the case of the iGRASP×ELS and hNSGA-II two objectives could be compared to solutions from the literature: the makespan, and the peak power consumption. Considering the makespan, best solutions in the Pareto front considering this criterion are given in Table 3. Results are compared with two specifically

designed algorithms, namely the Improved Shuffle Complex Evolution Algorithm (ISCEA) proposed in (Zhao, Zhang, Zhang, & Wang, 2015) and the New Island Model Genetic Algorithm provided in (Kurdi, 2016). In this table $\%BKS$ refers to the deviation between best found solutions and best known solutions from the literature, whereas $\overline{\%BKS}$ refers to the average deviation over the different runs. Results show that the iGRASP×ELS has a better behaviour than the hNSGA-II when considering only the makespan. When compared with ISCEA, it can be observed that this metaheuristic is slightly better than the iGRASP×ELS on $\%BKS$, but the iGRASP×ELS seems to have a better behaviour in average; it also seems to be preferable to NIMGA. Meanwhile, the hNSGA-II is clearly outperformed by the GRASP×ELS and the other metaheuristics on the makespan criterion. This suggests that the overall performance attained by the hNSGA-II is mainly due to the solutions with low power thresholds. Actually, it appears that the behaviour of this metaheuristic is largely dependent from the non-hierarchised local search procedure which tends to orientate the search towards higher makespan solutions with lower power thresholds. Concerning this second objective it is more difficult to compare with other studies, as this work is one of the only works dealing with this as an objective. Another work by (Masmoudi, Delorme, & Gianessi, 2019) deals with similar constraints, but it is focused on minimising costs, and it is a linear program solving problems with 36 and 50 operations, which is lower than the medium problems with up to 300 operations we are addressing here.

Table 3
Comparison of performances on makespan criterion between different metaheuristics.

	iGRASP×ELS		hNSGA-II		ISCEA		NIMGA	
	$\%BKS$	$\overline{\%BKS}$	$\%BKS$	$\overline{\%BKS}$	$\%BKS$	$\overline{\%BKS}$	$\%BKS$	$\overline{\%BKS}$
la01-05	0	0	0.54	0.91	0	0.68	0	0
la06-10	0	0	0	0	0	0	0	0
la11-15	0	0	0	0	0	0.01	0	0
la16-20	0	0.04	1.15	2.77	0.38	1.98	0.11	0.62
la21-25	0.57	1.13	3.43	6.85	0.57	1.36	0.95	2.91
la26-30	2.1	2.64	7.04	10.58	0.61	2.53	2.69	4.2
la31-35	0	0	2.11	4.23	0.02	0.16	0	0.03
la36-40	1.73	2.24	5.94	9.41	1.43	3.68	2.01	4.25
Mean:	0.55	0.76	2.53	4.34	0.38	1.3	0.72	1.50

Actually, the GRASP×ELS is very useful when a single solution is needed considering a given power threshold. For example, if a power supplier informs a customer that the available power during a given period is reduced to a lower level than the one contracted (because of breakdowns, or maintenance tasks), it is up to the enterprise to adjust the schedule conformingly to the new power threshold. In such a scenario, it is possible to use the GRASP×ELS used in the iGRASP×ELS because its intensification scheme is better suited than the hNSGA-II, which relies on a genetic algorithm (Kemmoe et al., 2017). Furthermore, the iGRASP×ELS requires less memory space, since no population is used during the local search phases, which could be useful in order to deal with large

scale instances. This is exposed in Figure 6, which shows the $\overline{RN}$ value of the Pareto fronts when considering only solutions with high power thresholds. This final figure proves the GRASP×ELS is able to produce non-dominated solutions with high power thresholds, and hence the complementarity of the two proposed metaheuristics.

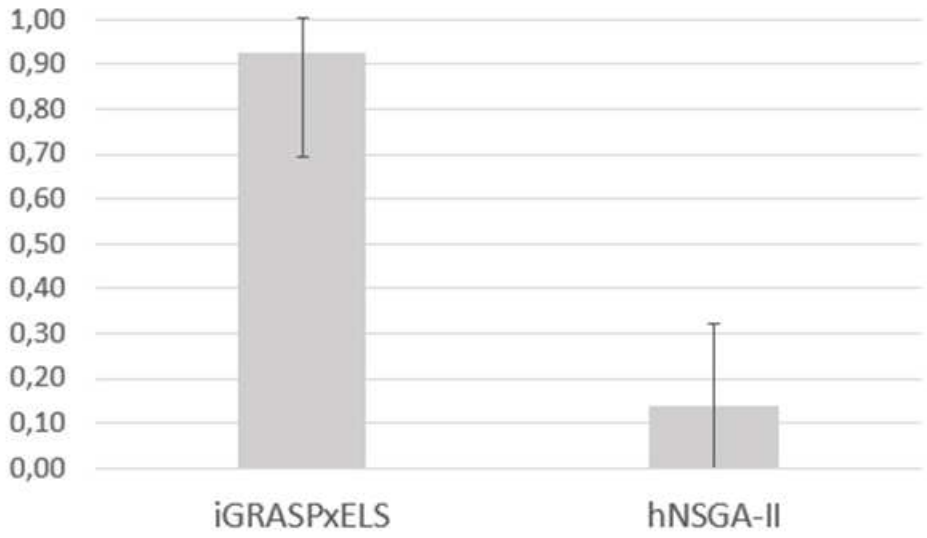

Fig. 6. Average values for the RN when focusing on solutions with more than 150 operations and with high power thresholds.

To demonstrate the importance of the local search in the hNSGA-II, a comparison with the classical NSGA-II (without local search) is also given in Table 4. It is clear from this last table that the hNSGA-II outperforms the NSGA-II (all p-values are inferior to 0.05) which confirms the assumption of (Lacomme et al., 2006). Note that, compared to Table 2, $\overline{HR}$ changes for hNSGA-II, as the reference point *nadir* also changes.

Table 4
Comparison of performances between hNSGA-II and NSGA-II on Bi-JSPPR

				hNSGA-II			NSGA-II		
Ins.	nbjob	nbmac	n	$\overline{HR}$	$\overline{TS}$	$\overline{RN}$	$\overline{HR}$	$\overline{TS}$	$\overline{RN}$
la01-05	10	5	50	**22.55**	**0.88**	**0.90**	38.54	0.95	0
la06-10	15	5	75	**22.36**	0.80	**0.99**	40.63	0.80	0
la11-15	20	5	100	**20.43**	0.91	**0.99**	37.95	**0.81**	0
la16-20	10	10	100	**15.01**	**0.67**	**1.00**	31.45	0.89	0
la21-25	15	10	150	**15.78**	**0.64**	**1.00**	35.86	0.80	0
la26-30	20	10	200	**15.87**	**0.62**	**1.00**	34.02	0.74	0
la31-35	30	10	300	**15.07**	0.64	**1.00**	31.64	0.64	0
la36-40	15	15	225	**14.47**	**0.64**	**1.00**	33.93	0.70	0
Average :				**17.69**	**0.73**	**0.98**	35.5	0.79	0

If the results presented in this study concern a specific problem considering power requirements of production systems, some hypothesis have been made. They mainly concern the power profiles of operations that could consider more variability. This would however increase complexity of the

problem and a trade-off in the modelling could be explored. Another limitation concerns the values of the power profiles as it is very difficult to obtain data from manufacturers. Focusing on other environmental and social indicators could also be valuable for industrials. Also, the approach in this paper is deterministic as no stochasticity is considered. A continuation of this work in an innovative way would be to address and study the problem where operations have variable processing times and hence the objective would resume in finding more robust solutions. Also, as production systems are dynamic, designing reactive approaches based on dispatching rules, or fast heuristics dedicated to power minimisation could be investigated. The two approaches could be joined in order to explore predictive-reactive approaches. Finally, the paper concerns the optimisation of Job-shop floors, and the consideration of flexible manufacturing, where several machines are available, with different processing characteristics would be interesting. As the industry tends to move forward to reconfigurable manufacturing systems, this kind of production systems could also benefits from research on energy efficiency and scheduling approaches (Battaïa, Benyoucef, Delorme, Dolgui, & Thevenin, 2020).

To conclude this experiment, it appears that both approaches have pros and cons depending on the objective of the decision maker. If the focus is given to high productivity, the iGRASP×ELS will be preferred, whereas the hNSGA-II can be chosen when better performances on low power thresholds are searched for. The capacity of the hNSGA-II to produce a large amount of solutions is also an interesting characteristic for a decision maker. However, one drawback of the approach might concern the definition of the point where it is more relevant to use the iGRASP×ELS rather than the hNSGA-II considering specificities of each algorithm. Hence, based on obtained results, the definition of an adaptive optimiser, choosing the proper optimisation tool given a situation/objective could be an interesting search project.

6. Conclusion

In this paper, the Bi-objective Job-shop Problem with Power Requirements (Bi-JSPPR) is investigated. Two objectives are optimised: the makespan and the power threshold. Optimising power threshold is an important objective, especially in enterprises where machines cannot be shut down, as few improvements could be done concerning the total energy consumptions in such an environment. In the problem studied in this work, each operation presents two types of power requirements to fit real world power consumption of machines. The instances generated represent production systems, where up to 300 operations must be scheduled. These instances are difficult to solve with a linear solver, and obtaining a Pareto front with such an approach is time consuming. Hence, two metaheuristics are proposed: a hybrid NSGA-II (hNSGA-II) and an iterated GRASP×ELS (iGRASP×ELS). The purpose of these metaheuristics is to obtain a near optimal Pareto frontier in a rather small computational effort. The local search added to the NSGA-II is an intensification scheme

that allows the hNSGA-II to achieve better results than the NSGA-II alone. This metaheuristic is compared to the iGRASP×ELS, which is based on a GRASP×ELS previously proposed for the search of single solutions when the power threshold is considered a given data (Kemmoe et al., 2017). The computational experiment shows the efficiency of the proposed approaches. As stressed by the results, the hNSGA-II performs better than the iGRASP×ELS. However, the GRASP×ELS used in the iGRASP×ELS can also be an interesting metaheuristic as it provides better solutions when the power threshold is high, and it can also be used if the goal is to find one solution considering a specific power threshold by increasing its computation time.

As stressed by (Giret et al., 2015), energy management of production systems still lacks decisional tools. The proposed approaches in this paper can be easily adapted to industrial cases because of the structure of Job-shop problems, which represent several and widely spread manufacturing systems (Salido et al., 2016). Furthermore, considering power optimisation as an important objective in classical scheduling problems will allow to design energy-efficient production systems. For this purpose, several research directions might be investigated. For instance, it could be interesting to add other objectives such as decreasing the total energy consumption or to include TOU pricings or real-time pricings in order to reduce the cost of the production. If the approach investigated in this paper consists in two objectives, it is also a first step in integrating several other environmental objectives such as waste minimisation or reusal (Le Hesran, Ladier, Botta-Genoulaz, & Laforest, 2019). As stressed in (Akbar & Irohara, 2018) other environmental and social indicators should be addressed, hence, considering an integrated problem with operators, rest allowance, and energy-efficient scheduling problems could be a promising direction. A first future work will consist in designing instances that consider operations with wider power behaviours in order to study the impact of discretisation on the quality and robustness of solutions (Rager et al., 2015). Designing methods for predictive-reactive scheduling is also a promising direction as production systems are dynamic and have to adapt to different situations during scheduling horizons. Extending the current research project to flexible manufacturing systems, where several machines are available for a given operation, having different processing characteristics, or even reconfigurable manufacturing systems, would be an interesting search topic. Finally, as the iGRASP×ELS consists in constructing the Pareto frontier by running several times the GRASP×ELS with different power thresholds, this metaheuristic could be parallelised in order to reduce computation times. For this purpose, the use of STORM could be an interesting direction (Avez, Lacomme, Lamy, Tchernev, & Phan, 2016), and future research works with joint optimisation and machine learning operations are to be investigated, especially for real production systems with thousands of operations to be scheduled.

Appendix A. Mathematical formalisation of the Job-shop problem with power constraints

Notations

The notations used in the problem are as follows:

H : a large positive number;

M : set of machines;

J : set of jobs;

V : set of all the operations ($|V| = |M|.|J|$);

i, j : indexes for the different operations ($i = O_i$);

J_i : job of operation i;

SO_i : set of sub-operations of operation i;

k, l : indexes representing the different sub-operations of operations;

m_i : machine required to process operation i, $m_i \in M$;

$P_{i,k}$: duration of k^{th} sub-operation of operation i;

$W_{i,k}$: power required for processing the k^{th} sub-operation of operation i;

w_{max} : maximum power that must never be exceeded;

c_{max} : completion date of all operations also called makespan of the schedule;

$s_{i,k}$: starting time of k^{th} sub-operation of operation i;

$x_{i,k,j,l}$: binary variable equal to 1 if k^{th} sub-operation of operation i is realised before the l^{th} sub-operation of operation j and equal to 0 otherwise;

$y_{i,k,j,l}$: binary variable equal to 1 if there is a non-null power flow from operation i to the operation j and equal to *0* otherwise;

$\varphi_{i,k,j,l}$: denotes the number of power units transferred from the k^{th}sub-operation of operation i to the l^{th} sub-operation of operation j. $\varphi_{0,0,j,l}$ concerns the power units transferred from the source node to other sub-operations;

Mathematical Formalisation

The first line (1) of the mathematical formalisation refers to the objective of the problem, which is the minimisation of both the completion time of all operations (makespan) and the power threshold:

$$\begin{matrix} Min\ c_{max} \\ Min\ w_{max} \end{matrix} \quad (1)$$

Job-shop model

$$s_{i,|SO_i|} - c_{max} \leq -P_{i,|SO_i|}, \forall i \in V \quad (2)$$

$$x_{i,|SO_i|,j,1} + x_{j,|SO_j|,i,1} = 1, \forall (i,j) \in V^2, m_i = m_j \quad (3)$$

$$s_{j,1} - s_{i,|SO_i|} \geq P_{i,|SO_i|}, \forall (i,j) \in V^2, i < j, J_i = J_j \quad (4)$$

$$s_{i,k} - s_{i,k-1} = P_{i,k-1}, \forall i \in V, \forall k \in SO_i, k > 1 \quad (5)$$

$$s_{j,1} - s_{i,|SO_i|} - Hx_{i,|SO_i|,j,1} \geq P_{i,|SO_i|} - H, \forall (i,j) \in V^2, m_i = m_j \quad (6)$$

The first set of constraints (2) gives the expression of the makespan, which must be greater or equal to the end date of all the operations (as operations are split into sub-operations, the last sub-operation of each operation is considered, hence the use of the cardinality of the subset SO_i). Constraints (3) represent the disjunctions constraints for operations occurring on the same machines. In these constraints, if two operations i and j of different jobs must be scheduled on the same machine, then i is processed before j, or j is processed before i. Constraints (4) define the starting dates of operations of a job according to its sequence of operations. Constraints (5) ensure that, each sub-operations referring to the same operation are processed without delays (i.e. no-wait constraints). Constraints (6) adjust the starting dates of operations that belong to different Jobs but need the same machine, as they cannot be processed simultaneously.

Power consumption model

$$\sum_{j\in V}\sum_{k\in SO_j}\varphi_{0,0,j,k} - w_{max} \leq 0 \quad (7)$$

$$\varphi_{0,0,j,l} + \sum_{i\in V\setminus j}\sum_{k\in SO_i}\varphi_{i,k,j,l} + \sum_{k=1}^{l-1}\varphi_{j,k,j,l} = W_{j,l}, \forall j \in V, \forall l \in SO_j \quad (1)$$

$$\sum_{j\in V\setminus i}\sum_{l\in SO_j}\varphi_{i,k,j,l} + \sum_{l=k+1}^{|SO_i|}\varphi_{i,k,i,l} \leq W_{i,k}, \forall i \in V, \forall k \in SO_i \quad (2)$$

$$\varphi_{i,k,j,l} - Hy_{i,k,j,l} \leq 0, \forall (i,j) \in V^2 \forall (k,l) \in \left(SO_i, SO_j\right) \quad (3)$$

$$y_{i,k,j,l} - \varphi_{i,k,j,l} \leq 0, \forall (i,j) \in V^2, \forall (k,l) \in \left(SO_i, SO_j\right) \quad (4)$$

$$s_{j,l} - s_{i,k} - Hy_{i,k,j,l} \geq P_{i,k} - H, \forall (i,j) \in V^2, \forall (k,l) \in \left(SO_i, SO_j\right), J_i \neq J_j \quad (5)$$

$$\varphi_{i,k,j,l} = 0, \forall (i,j) \in V^2, \forall (k,l) \in (SO_i, SO_j), (j < i, J_i = J_j) \vee (i = j, l < k) \quad (6)$$

The constraint (7) avoids to exceed the Power Threshold when processing the operations as it cannot be allocated more power to the operations than w_{max} . Constraints (8) ensure that the sum of power flows from sub-operations and initial power threshold is equal to the power needed for the k^{th} sub-operation of operation j. Constraints (9) ensure that the sum of power flows from the considered k^{th} sub-operation of operation i to other sub-operations never exceeds the power that was used for its processing. Constraints (10) ensure that if there is a power flow from k^{th} sub-operation of i to l^{th} sub-operation of j, then $y_{i,k,j,l} = 1$ (flow detection). If $y_{i,k,j,l} = 0$ then no flow is possible from k^{th} sub-operation of i to l^{th} operation of j. Constraints (11) stipulate that if there is no need of a flow from i to j ($\varphi_{i,k,j,l} = 0$), then necessarily $y_{i,k,j,l} = 0$; if $y_{i,k,j,l} = 1$, then $\varphi_{i,k,j,l} \geq 1$. Constraints (12) adjust the starting dates of sub-operations which need to wait before the end of previous operations, in order to not exceed the power threshold and receive a power flow from a previously scheduled operation. Constraints (13) stipulate that no flow is possible between two sub-operations i and j, if i and j belong to the same job and if i is processed before j.

References

Abedi, M., Chiong, R., Noman, N., & Zhang, R. (2020). A multi-population, multi-objective memetic algorithm for energy-efficient job-shop scheduling with deteriorating machines. *Expert Systems with Applications*, 113348. https://doi.org/10.1016/j.eswa.2020.113348

Agnetis, A., Flamini, M., Nicosia, G., & Pacifici, A. (2011). A job-shop problem with one additional resource type. *Journal of Scheduling*, *14*(3), 225–237. https://doi.org/10.1007/s10951-010-0162-4

Ahmadi, E., Zandieh, M., Farrokh, M., & Emami, S. M. (2016). A multi objective optimization approach for flexible job shop scheduling problem under random machine breakdown by evolutionary algorithms. *Computers & Operations Research*, *73*, 56–66. https://doi.org/10.1016/j.cor.2016.03.009

Akbar, M., & Irohara, T. (2018). Scheduling for sustainable manufacturing: A review. *Journal of Cleaner Production*, *205*, 866–883. https://doi.org/10.1016/j.jclepro.2018.09.100

Annual Energy Outlook 2016. (2016). *U.S. Energy Information Administration*.

Avez, G., Lacomme, P., Lamy, D., Tchernev, N., & Phan, R. (2016). First experiment of STORM for design of efficient optimization methods: Application to the Job-shop with Time-lags. *11th International Conference on Modeling, Optimization and Simulation*. Presented at the MOSIM, Montréal.

Battaïa, O., Benyoucef, L., Delorme, X., Dolgui, A., & Thevenin, S. (2020). Sustainable and Energy Efficient Reconfigurable Manufacturing Systems. In *Reconfigurable Manufacturing Systems: From Design to Implementation* (pp. 179–191). Springer Series in Advanced Manufacturing.

Bechikh, S., Elarbi, M., & Ben Said, L. (2017). Many-objective Optimization Using Evolutionary Algorithms: A Survey. In S. Bechikh, R. Datta, & A. Gupta (Eds.), *Recent Advances in Evolutionary Multi-objective Optimization* (Vol. 20, pp. 105–137). https://doi.org/10.1007/978-3-319-42978-6_4

Bierwirth, C. (1995). A generalized permutation approach to job shop scheduling with genetic algorithms. *Operations-Research-Spektrum*, *17*(2–3), 87–92.

Bruzzone, A. A. G., Anghinolfi, D., Paolucci, M., & Tonelli, F. (2012). Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops. *CIRP Annals - Manufacturing Technology*, *61*(1), 459–462. https://doi.org/10.1016/j.cirp.2012.03.084

Chassaing, M., Fontanel, J., Lacomme, P., Ren, L., Tchernev, N., & Villechenon, P. (2014). A GRASP×ELS approach for the job-shop with a web service paradigm packaging. *Expert Systems with Applications*, *41*(2), 544–562. https://doi.org/10.1016/j.eswa.2013.07.080

Collette, Y., & Siarry, P. (2002). *Optimisation multiobjectif*. Paris: Eyrolles.

Dai, M., Tang, D., Giret, A., Salido, M. A., & Li, W. D. (2013). Energy-efficient scheduling for a flexible flow shop using an improved genetic-simulated annealing algorithm. *Robotics and Computer-Integrated Manufacturing*, *29*(5), 418–429. https://doi.org/10.1016/j.rcim.2013.04.001

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197.

Duhamel, C., Lacomme, P., Prins, C., & Prodhon, C. (2010). A GRASP×ELS approach for the capacitated location-routing problem. *Computers & Operations Research*, *37*(11), 1912–1923. https://doi.org/10.1016/j.cor.2009.07.004

Ehrgott, M., & Gandibleux, X. (2001). Bounds and Bound Sets for Biobjective Combinatorial Optimization Problems. In M. Köksalan & S. Zionts (Eds.), *Multiple Criteria Decision Making in the New Millennium* (Vol. 507, pp. 241–253). https://doi.org/10.1007/978-3-642-56680-6_22

Fang, K., Uhan, N., Zhao, F., & Sutherland, J. W. (2011). A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction. *Journal of Manufacturing Systems*, *30*(4), 234–240. https://doi.org/10.1016/j.jmsy.2011.08.004

Gahm, C., Denz, F., Dirr, M., & Tuma, A. (2016). Energy-efficient scheduling in manufacturing companies: A review and research framework. *European Journal of Operational Research*, *248*(3), 744–757. https://doi.org/10.1016/j.ejor.2015.07.017

Giret, A., Trentesaux, D., & Prabhu, V. (2015). Sustainability in manufacturing operations scheduling: A state of the art review. *Journal of Manufacturing Systems*, *37*, 126–140. https://doi.org/10.1016/j.jmsy.2015.08.002

Gong, G., Deng, Q., Gong, X., Liu, W., & Ren, Q. (2018). A new double flexible job-shop scheduling problem integrating processing time, green production, and human factor indicators. *Journal of Cleaner Production*, *174*, 560–576. https://doi.org/10.1016/j.jclepro.2017.10.188

Grabowski, J., Nowicki, E., & Zdrzalka, S. (1986). A block approach for single-machine scheduling with release dates and due dates. *European Journal of Operational Research*, *26*(2), 278–285.

Gutowski, T., Murphy, C., Allen, D., Bauer, D., Bras, B., Piwonka, T., … Wolff, E. (2005). Environmentally benign manufacturing: Observations from Japan, Europe and the United States. *Journal of Cleaner Production*, *13*(1), 1–17. https://doi.org/10.1016/j.jclepro.2003.10.004

He, Y., Li, Y., Wu, T., & Sutherland, J. W. (2015). An energy-responsive optimization method for machine tool selection and operation sequence in flexible machining job shops. *Journal of Cleaner Production*, *87*, 245–254. https://doi.org/10.1016/j.jclepro.2014.10.006

Kemmoe, S., Lamy, D., & Tchernev, N. (2017). Job-shop like manufacturing system with variable power threshold and operations with power requirements. *International Journal of Production Research*, *55*(20), 6011–6032. https://doi.org/10.1080/00207543.2017.1321801

Ku, W.-Y., & Beck, J. C. (2016). Mixed Integer Programming models for job shop scheduling: A computational analysis. *Computers & Operations Research*, *73*, 165–173. https://doi.org/10.1016/j.cor.2016.04.006

Kurdi, M. (2016). An effective new island model genetic algorithm for job shop scheduling problem. *Computers & Operations Research*, *67*, 132–142. https://doi.org/10.1016/j.cor.2015.10.005

Lacomme, P., Prins, C., & Sevaux, M. (2006). A genetic algorithm for a bi-objective capacitated arc routing problem. *Computers & Operations Research*, *33*(12), 3473–3493. https://doi.org/10.1016/j.cor.2005.02.017

Lawrence, S. (1984). *Resource constrained project scheduling: An experimental investigation of heuristic scheduling techniques (Supplement)*. Graduate School of Industrial Administration, Carnegie-Mellon University.

Le Hesran, C., Ladier, A.-L., Botta-Genoulaz, V., & Laforest, V. (2019). Operations scheduling for waste minimization: A review. *Journal of Cleaner Production*, *206*, 211–226. https://doi.org/10.1016/j.jclepro.2018.09.136

Lei, D., Zheng, Y., & Guo, X. (2016). A shuffled frog-leaping algorithm for flexible job shop scheduling with the consideration of energy consumption. *International Journal of Production Research*, *55*(11), 3126–3140. https://doi.org/10.1080/00207543.2016.1262082

Lin, Y.-K., & Yeh, C.-T. (2012). Multi-objective optimization for stochastic computer networks using NSGA-II and TOPSIS. *European Journal of Operational Research*, *218*(3), 735–746. https://doi.org/10.1016/j.ejor.2011.11.028

Liu, Y., Dong, H., Lohse, N., & Petrovic, S. (2015). Reducing environmental impact of production during a Rolling Blackout policy – A multi-objective schedule optimisation approach. *Journal of Cleaner Production*, *102*, 418–427. https://doi.org/10.1016/j.jclepro.2015.04.038

Liu, Y., Dong, H., Lohse, N., Petrovic, S., & Gindy, N. (2014). An investigation into minimising total energy consumption and total weighted tardiness in job shops. *Journal of Cleaner Production*, *65*, 87–96. https://doi.org/10.1016/j.jclepro.2013.07.060

Luo, H., Du, B., Huang, G. Q., Chen, H., & Li, X. (2013). Hybrid flow shop scheduling considering machine electricity consumption cost. *International Journal of Production Economics*, *146*(2), 423–439. https://doi.org/10.1016/j.ijpe.2013.01.028

Masmoudi, O., Delorme, X., & Gianessi, P. (2019). Job-shop scheduling problem with energy consideration. *International Journal of Production Economics*, *216*, 12–22. https://doi.org/10.1016/j.ijpe.2019.03.021

May, G., Stahl, B., Taisch, M., & Prabhu, V. (2015). Multi-objective genetic algorithm for energy-efficient job shop scheduling. *International Journal of Production Research*, *53*(23), 7071–7089. https://doi.org/10.1080/00207543.2015.1005248

Merkert, L., Harjunkoski, I., Isaksson, A., Säynevirta, S., Saarela, A., & Sand, G. (2015). Scheduling and energy – Industrial challenges and opportunities. *Computers & Chemical Engineering*, *72*, 183–198. https://doi.org/10.1016/j.compchemeng.2014.05.024

Moon, J.-Y., & Park, J. (2014). Smart production scheduling with time-dependent and machine-dependent electricity cost by considering distributed energy resources and energy storage. *International Journal of Production Research*, *52*(13), 3922–3939. https://doi.org/10.1080/00207543.2013.860251

Prins, C. (2009). A GRASP x evolutionary local search hybrid for the vehicle routing problem. In *Bio-inspired algorithms for the vehicle routing problem* (pp. 35–53). Retrieved from http://link.springer.com/chapter/10.1007/978-3-540-85152-3_2

Rager, M., Gahm, C., & Denz, F. (2015). Energy-oriented scheduling based on Evolutionary Algorithms. *Computers & Operations Research*, *54*, 218–231. https://doi.org/10.1016/j.cor.2014.05.002

Salido, M. A., Escamilla, J., Barber, F., Giret, A., Tang, D., & Dai, M. (2016). Energy efficiency, robustness, and makespan optimality in job-shop scheduling problems. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, *30*(03), 300–312. https://doi.org/10.1017/S0890060415000335

Shrouf, F., Ordieres-Meré, J., García-Sánchez, A., & Ortega-Mier, M. (2014). Optimizing the production scheduling of a single machine to minimize total energy consumption costs. *Journal of Cleaner Production*, *67*, 197–207. https://doi.org/10.1016/j.jclepro.2013.12.024

Tan, K. C., Goh, C. K., Yang, Y. J., & Lee, T. H. (2006). Evolving better population distribution and exploration in evolutionary multi-objective optimization. *European Journal of Operational Research*, *171*(2), 463–495. https://doi.org/10.1016/j.ejor.2004.08.038

Tang, D., & Dai, M. (2015). Energy-efficient approach to minimizing the energy consumption in an extended job-shop scheduling problem. *Chinese Journal of Mechanical Engineering*, *28*(5), 1048–1055. https://doi.org/10.3901/CJME.2015.0617.082

Van Laarhoven, P. J. M., Aarts, E. H. L., & Lenstra, J. K. (1992). Job Shop Scheduling by Simulated Annealing. *Operations Research*, *40*(1), 113–125.

Weinert, N., Chiotellis, S., & Seliger, G. (2011). Methodology for planning and operating energy-efficient production systems. *CIRP Annals - Manufacturing Technology*, *60*(1), 41–44. https://doi.org/10.1016/j.cirp.2011.03.015

Wu, X., & Sun, Y. (2018). A green scheduling algorithm for flexible job shop with energy-saving measures. *Journal of Cleaner Production*, *172*, 3249–3264. https://doi.org/10.1016/j.jclepro.2017.10.342

Yen, G. G., & He, Z. (2013). Performance Metrics Ensemble for Multiobjective Evolutionary Algorithms. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION*, 14.

Zeng, Z., Hong, M., Li, J., Man, Y., Liu, H., Li, Z., & Zhang, H. (2018). Integrating process optimization with energy-efficiency scheduling to save energy for paper mills. *Applied Energy*, *225*, 542–558. https://doi.org/10.1016/j.apenergy.2018.05.051

Zhang, R., & Chiong, R. (2016). Solving the energy-efficient job shop scheduling problem: A multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. *Journal of Cleaner Production*, *112*(4), 3361–3375. https://doi.org/10.1016/j.jclepro.2015.09.097

Zhao, F., Zhang, J., Zhang, C., & Wang, J. (2015). An improved shuffled complex evolution algorithm with sequence mapping mechanism for job shop scheduling problems. *Expert Systems with Applications*, *42*(8), 3953–3966. https://doi.org/10.1016/j.eswa.2015.01.007

Zhou, A., Qu, B.-Y., Li, H., Zhao, S.-Z., Suganthan, P. N., & Zhang, Q. (2011). Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, *1*(1), 32–49. https://doi.org/10.1016/j.swevo.2011.03.001

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, *8*(2), 173–195. https://doi.org/10.1162/106365600568202