Swarm and Evolutionary Computation 68 (2022) 101016

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Survey Paper

Multi-objective enhanced memetic algorithm for green job shop scheduling with uncertain times

Sezin Afsar[a], Juan José Palacios[a], Jorge Puente[a], Camino R. Vela[a,*], Inés González-Rodríguez[b]

[a] Department of Computing, University of Oviedo, Campus of Gijón, Gijón, 33204, Spain
[b] Dep. of Mathematics, Statistics and Computing, University of Cantabria, Av. Los Castros s/n, Santander, 39011 Spain

ARTICLE INFO

Keywords:
Job shop scheduling
Fuzzy durations
Multi-objective
Makespan
Non-processing energy
Memetic algorithm

ABSTRACT

The quest for sustainability has arrived to the manufacturing world, with the emergence of a research field known as *green scheduling*. Traditional performance objectives now co-exist with energy-saving ones. In this work, we tackle a job shop scheduling problem with the double goal of minimising energy consumption during machine idle time and minimising the project's makespan. We also consider uncertainty in processing times, modelled with fuzzy numbers. We present a multi-objective optimisation model of the problem and we propose a new enhanced memetic algorithm that combines a multiobjective evolutionary algorithm with three procedures that exploit the problem-specific available knowledge. Experimental results validate the proposed method with respect to hypervolume, ϵ-indicator and empirical attaintment functions.

1. Introduction

Scheduling problems appear in numerous domains, among others, engineering, management science or distributed and parallel computing. One of the most relevant problems is the job shop problem in its numerous variants, since it is considered to be a reference for many practical applications (for instance, wafer fabs in the semiconductor industry often function as job shops) [1]. Also, it is an NP-hard problem, thus posing a challenge to the research community [2].

The most common objective in the literature consists in finding solutions minimising the execution time span of the project, known as makespan. However, more recently, a growing environmental awareness has translated into concerns about the carbon footprint of everyday industrial process. Additionally, energy consumption in high-volume manufacturing constitutes a significant cost item in important industries (pharmaceutical, chemical, food-processing, moulding etc[3,4]. [5]). Minimising energy consumption is thus essential to control the environmental impact of a project while it can also bring significant cost reductions. However, energy saving should not be obtained at the cost of sacrificing service levels. For this reason, the concepts of energy-aware and green scheduling have emerged as relevant research topics in recent years [6]. In this new context, production scheduling can play a key role in reducing the energy consumption of factories and industrial plants as well as fossil fuels use in hybrid production systems by incorporating energy efficiency to the range of objectives under consideration [3,7,8]. In particular, for those production systems where machines cannot be completely turned off during idle periods (since restarting them may require a high energy consumption, or frequent turn on and off may damage the machines), the stand-by energy consumption incurred by idle machines constitutes an energy cost that should be minimised [9].

Additionally, mainstream scheduling approaches usually assume that all activity durations are precisely known in advance and do not change as the solution is being executed. Still, in many real-world applications these variables are subject to perturbations or changes, causing optimal solutions to the original problem to be of little or no use in practice [10,11]. This is why there exists an increasing interest on taking into consideration uncertainty in some of the input variables. To this end, fuzzy sets provide an interesting framework to tackle uncertainty in scheduling [12]. In particular, the fuzzy job shop problem, a job shop with uncertain durations modelled using fuzzy numbers, has received increasing attention during the last years [13,14]. The best-known solutions for the fuzzy job shop in terms of makespan have been obtained by a memetic algorithm combining a genetic algorithm and tabu search [15].

The simultaneous minimisation of makespan and objective functions related to energy consumption has been the goal of several recent contributions in different deterministic scheduling problems: unrelated parallel machine [16–19], different variants of the permutation flowshop [6,20–24], job shop [17] [25], and more complex production scheduling [26]. However, to the best of our knowledge, energy costs in scheduling

* Corresponding author.:
E-mail addresses: afsarsezin@uniovi.es (S. Afsar), palaciosjuan@uniovi.es (J.J. Palacios), puente@uniovi.es (J. Puente), crvela@uniovi.es (C.R. Vela), gonzalezri@unican.es (I. González-Rodríguez).

https://doi.org/10.1016/j.swevo.2021.101016
Received 6 August 2020; Received in revised form 4 September 2021; Accepted 3 November 2021
Available online 17 November 2021

under uncertainty are only addressed in [27], where a multi-objective genetic algorithm is proposed to optimise both the total weighted tardiness and the non-processing energy consumption.

The literature on deterministic scheduling with energy and makespan includes a number of successful and diverse solving methods based on mathematical optimisation [4,20,23], constraint programming [4,17,23], constructive heuristics [6,23], several bioinspired metaheuristics such as ant colony [21], differential evolution [19], multi-objective genetic algorithm hybridised with heuristic population initialisation [25] or whale-swarm [24], and memetic algorithms incorporating different types of local search [16,18,26]. Also, in an increasingly popular approach known as *mathheuristics*, the biobjective job shop minimising total weighted tardiness and energy consumption is tackled in [28] using a memetic algorithm incorporating a linear programming post-processing to further improve the energy consumption of a set of non-dominated solutions. More commonly, mathematical programming is embedded into metaheuristics to improve local search [29] or used to solve relaxed versions of the problem [30] or some sub-problems [31].

In this work we consider a multiobjective fuzzy job shop problem with two different objectives: minimising the makespan and minimising the non-processing energy. In so doing, we address the reduction of energy consumption in manufacturing (and the consequent improvement in environmental sustainability) without compromising service levels. Following a design principle of memetic algorithms [32], we propose a new hybrid metaheuristic combining functionalities from different search paradigms to achieve a synergetic behaviour.

We have a dominance-based evolutionary component, with a population of solutions that approach the optimal Pareto front from different directions. During the evolution, we incorporate two different single-objective local search procedures to optimise individual objectives: a tabu search for makespan and a heuristic for non-processing energy. Although having single-objective local search in a multiobjective memetic algorithm is not the most extended approach in the literature, it has nonetheless been successfully used in the past, for instance, in [33]. The weaker intensification provided by the heuristic method for the non-processing energy objective motivates incorporating a third intensification mechanism: a post-process based on a linear programming model making use of the commercial solver IBM ILOG CPLEX [34] to further improve the final set of non-dominated solutions. This obtains optimal energy values without altering job processing orders on each machine nor the makespan, an approach inspired by [28]. In the literature, we find multi-objective local search algorithms using neighbourhood structures specifically designed for only one of the objectives (the makespan), even if they include constructive heuristics at the end of the local search to improve the energy consumption of the built schedule [6,24]. We shall argue that a better exploitation is obtained by combining a single-objective and domain-specific local search for the makespan with a heuristic reduction strategy and a linear programming technique focused on the energy. This design is, on other hand, a natural consequence of the fact that the fuzzy makespan is a regular measure whereas the fuzzy non-processing energy is not. An extensive experimental study is conducted to validate the proposed method using three performance metrics for multiobjective optimization: hyper-volume, ϵ-indicator and empirical attaintment functions.

The contributions of this paper are the following:

- The bi-objective fuzzy job shop scheduling problem with makespan and non-processing energy minimisation is considered for the first time and a new mixed integer programming (MIP) model is defined.
- A new enhanced memetic algorithm to solve this problem is proposed. Although it builds on previous works and templates from the literature, non-trivial changes are introduced in different components of the algorithm to handle the uncertainty and the objectives considered here and the different techniques are combined in a novel manner to obtain a memetic procedure that makes a reasonable use of computational resources. In particular:
  - The evolutionary algorithm slightly modifies the NSGA-II template to introduce more diversity in the population and incorporate strategies that make use of problem-specific knowledge to intensify the search separately for both objectives.
  - The heuristic procedure to improve the NPE of a solution by performing right-shifts is based on a similar one from [27] but it necessarily differs from this previous procedure given the change in the objective function to be optimised simultaneously with NPE. How the right-shifts can be performed without affecting the feasibility of the schedule nor worsening the makespan is established in a new theoretical result.
  - A commercial solver is incorporated to solve a linear program (LP) as a final improvement step which is obtained by linearising the MIP using the solution of the evolutionary algorithm.
- An extensive experimental study is presented to validate the proposal.
- A set of benchmark instances and a suite of detailed results of the multiobjective algorithm on these instances is provided as reference for future work on this variant of the job shop problem.

The remaining of this paper is organised as follows. After introducing some preliminary concepts in Section 2, a formal statement of the multiobjective problem is given in Section 3, including a bi-objective MIP model. Section 4 presents a multiobjective enhanced memetic algorithm as solving method. Finally, Section 5 reports experimental results to empirically evaluate the performance of the proposed method and the synergy produced by its different components. Finally, some remarks and conclusions are given in Section 6.

2. Preliminaries

2.1. Multi-objective optimisation

A *multi-objective optimisation problem* [35] is an optimisation problem considering more than a single objective function $f_1, \dots, f_G$, $G \geq 2$. In addition to the usual *decision variable space* X of feasible solutions there exists an additional space called the *objective space* Z, so every solution $\boldsymbol{x} \in X$ is associated to an objective vector $\boldsymbol{z} = \boldsymbol{f}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \dots, f_G(\boldsymbol{x})) \in Z$. It is usually assumed that $Z \subseteq \mathbb{R}^G$, that is, f_k associates to each decision variable $\boldsymbol{x}$ a value $f_k(\boldsymbol{x}) \in \mathbb{R}$. It suffices however that $f_k(\boldsymbol{x})$ is in a totally ordered set $\langle L, \leq_L \rangle$, so f_k induces a complete pre-order $\leq_k$ (a complete, reflexive and transitive relation) in the set of decision variables defined as follows: $\boldsymbol{x} \leq_k \boldsymbol{y}$ if and only if $f_k(\boldsymbol{x}) \leq_L f_k(\boldsymbol{y})$. Given this mapping $\boldsymbol{f} : X \to Z$, solutions in X may be compared in terms of some ordering relation in the objective space Z, most commonly, a Pareto order.

When the objective is to minimise $\boldsymbol{f}(\boldsymbol{x})$ subject to $\boldsymbol{x} \in X$, for any two feasible solutions $\boldsymbol{x}, \boldsymbol{y} \in X$ $\boldsymbol{x}$ is said to *dominate* $\boldsymbol{y}$, denoted $\boldsymbol{x} \prec \boldsymbol{y}$, if and only if $\forall k = 1, \dots, G, f_k(\boldsymbol{x}) \leq f_k(\boldsymbol{y})$ and there exists at least one objective k^*, $1 \leq k^* \leq G$, such that $f_{k^*}(\boldsymbol{x}) < f_{k^*}(\boldsymbol{y})$. In other words, $\boldsymbol{x}$ is no worse than $\boldsymbol{y}$ in all objectives and is strictly better in at least one of them. We also say that $\boldsymbol{x}$ is *preferred to* $\boldsymbol{y}$ or *non-dominated* by $\boldsymbol{y}$ and that $\boldsymbol{y}$ is *dominated by* $\boldsymbol{x}$. A solution $\boldsymbol{x}^* \in X$ is *Pareto-optimal* if it is non-dominated by any other solution, that is, there exists no other solution $\boldsymbol{x} \in X$ such that $\boldsymbol{x} \prec \boldsymbol{x}^*$. The set of Pareto-optimal solutions receives the name of *Pareto-optimal set* or *Pareto set* in short, and its mapping in the objective space is called *Pareto front*.

2.2. The job shop problem

The classical *job shop scheduling problem* (JSP) consists in scheduling a set of jobs J on a set of machines M, subject to a set of constraints. There are *precedence constraints*, so each job $j \in J$ consists of a set of tasks or operations to be sequentially scheduled. There are also *resource constraints*, whereby each task of a job requires the uninterrupted and exclusive use of a machine for its whole processing time; o_{jm} represents

the operation of job j processed on machine m and p_{jm} represents its processing time. There is no recirculation, so for each job j the function $succ_j(m)$, representing the machine that succeeds machine m in the processing of job j, determines a linear order in the set of the machines that process job j, $M_j \subseteq M$. Additionally, in an energy-aware context, we follow [9] and assume that the energy consumption per time unit of a machine m while it is idle is given by its idle power level P_m^{idle}.

A solution to this problem is a *schedule*, i.e. an allocation of starting times for each task, s_{jm}, which, besides being *feasible* (in the sense that all precedence and resource constraints hold), is *optimal* according to some criteria, in our case, reducing the makespan and the non-processing energy consumption. Analogously to the unrelated parallel machine problem [16], in the job shop, minimising the makespan is related to an efficient use of the machines whereas minimising energy consumption implies a reduction of costs for industries as well as a cost-conscious use of environmental resources.

2.3. Fuzzy durations

It is commonly assumed that the exact processing time of a task is exactly known in advance. However, this is rarely the case in real-life applications. More often, there is an uncertain knowledge about the duration, possibly based on previous experience. The crudest representation of such uncertain knowledge would be a human-originated confidence interval. If some values appear to be more plausible than others, then a natural extension is a fuzzy interval or fuzzy number. The simplest model is a *triangular fuzzy number* (TFN), denoted $\hat{a} = (a^1, a^2, a^3)$, given by an interval $[a^1, a^3]$ of possible values and a modal value $a^2 \in [a^1, a^3]$. Its membership function takes the following triangular shape:

$$\mu_{\hat{a}}(x) = \begin{cases} \frac{x-a^1}{a^2-a^1} & : a^1 \leq x \leq a^2 \\ \frac{x-a^3}{a^2-a^3} & : a^2 < x \leq a^3 \\ 0 & : x < a^1 \text{ or } a^3 < x. \end{cases} \tag{1}$$

Triangular fuzzy numbers (or, more generally, fuzzy intervals) are widely used in scheduling as a model for uncertain processing times [12–14].

For solving the job shop problem, four arithmetic operations on TFNs are necessary: addition, substraction, product by a constant and the maximum. These are usually defined by extending the corresponding operations on real numbers using the Extension Principle, so for any pair of TFNs $\hat{a}$ and $\hat{b}$ and for any $r \in \mathbb{R}$, the first three operations can be expressed as follows:

$$\hat{a} + \hat{b} = (a^1 + b^1, a^2 + b^2, a^3 + b^3). \tag{2}$$

$$\hat{a} - \hat{b} = (a^1 - b^3, a^2 - b^2, a^3 - b^1). \tag{3}$$

$$r\hat{a} = (ra^1, ra^2, ra^3). \tag{4}$$

Obviously, the above operations are also applicable in the case where either $\hat{a}$ or $\hat{b}$ are in fact real numbers ($\hat{a} = (a, a, a)$ or $\hat{b} = (b, b, b)$) and subsume the usual arithmetic operations for real numbers.

Unfortunately, computing the extended maximum is not that simple and the set of TFNs is not even closed under this operation. Hence, it is common in the fuzzy scheduling literature to approximate the maximum of two TFNs component-wise:

$$\max(\hat{a}, \hat{b}) \approx (\max(a^1, b^1), \max(a^2, b^2), \max(a^3, b^3)). \tag{5}$$

Besides its extended use, several arguments can be given in favour of this approximation (cf. [14]).

The membership function $\mu_{\hat{a}}$ of a fuzzy time $\hat{a}$ may be interpreted as a possibility distribution on the real numbers [36], representing the set of more or less plausible, mutually exclusive values of a variable a (in our case, the underlying uncertain time). Since a degree of possibility can be viewed as an upper bound of a degree of probability, $\mu_{\hat{a}}$ also encodes a whole family of probability distributions. This allows to define the *expected value* of a TFN $\hat{a}$ as [37]:

$$E[\hat{a}] = \frac{1}{4}(a^1 + 2a^2 + a^3). \tag{6}$$

The expected value is linear and it induces a ranking $\leq_E$ in the set of TFNs [38], so for any two TFNs $\hat{a}, \hat{b}$, $\hat{a} \leq_E \hat{b}$ if and only if $E[\hat{a}] \leq E[\hat{b}]$. Clearly, if $\forall i \in 1, 2, 3, a^i \leq b^i$, then $\hat{a} \leq_E \hat{b}$.

3. Problem formulation

In this work we address a job shop scheduling problem with fuzzy durations (fuzzy JSP or FJSP in short) with the aim of finding a solution that is efficient in terms of energy consumption and, at the same time, finishes the whole project as soon as possible. Hence, we consider two objectives: minimising makespan and minimising energy-consumption. These two objectives have a conflicting nature [6,16,23] because there is usually a trade-off between optimising production w.r.t. completion time and minimising energy consumption.

3.1. Fuzzy makespan

A schedule s for a job shop problem may be determined by a decision variable $\boldsymbol{x} = (x_{ijm}, i, j \in \boldsymbol{J}, m \in M_i \cap M_j)$ representing a partial processing order of jobs on all machines, where x_{ijm} takes value 1 if job i immediately precedes job j on machine m and 0 otherwise (notice that, for fixed values of j and m, there exists at most one i such that $x_{ijm} = 1$). Such schedule (starting completion times of all tasks) may be easily computed based on $\boldsymbol{x}$ using a semi-active schedule builder [39]. For every task o_{jm} with processing time $\hat{p}_{jm}$, let $pred_j(m)$ be the machine that precedes machine m in the processing order of job j. Then, if $x_{ijm} = 1$, the earliest feasible starting time of o_{jm}, $\hat{s}_{jm}$ is a TFN given by:

$$\hat{s}_{jm} = \begin{cases} \hat{c}_{jpred_j(m)} & \text{if } \forall i, x_{ijm} = 0, \\ \max(\hat{c}_{jpred_j(m)}, \hat{c}_{im}) & \text{if } x_{ijm} = 1; \end{cases} \tag{7}$$

where $\hat{c}_{jpred_j(m)} = (0, 0, 0)$ if there is no machine preceding m in the processing order of job j (that is, o_{jm} is the first operation of the job j). The corresponding completion time $\hat{c}_{jm}$ is a TFN obtained as follows:

$$\hat{c}_{jm} = \hat{s}_{jm} + \hat{p}_{jm}. \tag{8}$$

The resulting schedule s is fuzzy in the sense that the starting and completion times of all tasks are fuzzy numbers, interpreted as possibility distributions on the values that these times may take. Notice however that there is no uncertainty regarding the order in which jobs are to be processed in machines.

The completion time of each job j in the schedule s is the completion time of the operation $o_{j\mu_j}$ of job j to be processed in μ_j, the last machine where job j is processed. It is given by:

$$\hat{C}_j = \hat{c}_{j\mu_j}. \tag{9}$$

The makespan of a schedule s is the maximum completion time of all jobs:

$$\hat{C}_{max} = \max_{j \in J} \hat{C}_j. \tag{10}$$

The makespan $\hat{C}_{\max}$ is a *regular performance measure*, that is, a measure that is non-decreasing w.r.t. job completion times.

3.2. Fuzzy non-processing energy

Regarding energy efficiency, we adopt an energy consumption model from [27], which is an extension to the fuzzy framework of the model initially proposed in [9]. This model assumes that machines cannot be turned off when idle (for example when turning on/off takes excessive energy or time) and that machine power levels remain constant while processing a task. In consequence, the objective of reducing the total electricity consumption of a job shop comes down to reducing the total

Table 1
Parameters.

$i, j \in J$	jobs
$m \in M$	machines
$M_j \subseteq M$	set of machines that process job j
n_m	total number of jobs processed by machine m
$l \in \{1,2,3\}$	TFN components
$succ_j(m)$	the machine that succeeds machine m in the processing order of job j
$\widehat{p}_{jm}$	processing time of job j on machine m (TFN)
μ_j	the last machine that processes job j
P_m^{idle}	idle power level of a machine m
W	a sufficiently large number

non-processing energy (NPE), i.e. the energy consumed when machines are idle (on, but not processing any job). In [27], it was shown that the fuzzy NPE value is heavily dependent on the task processing order in each machine and the uncertainty present in the starting and completion times of every task.

Let $\boldsymbol{x}$ be the processing order of jobs processed on all machines, representing a schedule $\boldsymbol{s}$ as explained in Section 3.1, and let us suppose $x_{ijm} = 1$, that is, i and j are jobs consecutively processed on machine m. The *idle time* between jobs i and j on machine m according to $\boldsymbol{x}$ is a fuzzy quantity $\widehat{I}_{ijm}$ measuring the gap between the completion of task o_{im} and the start of task o_{jm}. Considering that in a feasible schedule the minimum possible gap has to be zero, we have that:

$$\widehat{I}_{ijm} = \max\{\widehat{0}, \widehat{s}_{jm} - \widehat{c}_{im}\}, \tag{11}$$

and $\widehat{I}_{ijm} = (0,0,0)$ if $x_{ijm} = 0$. Clearly, the total idle time for a machine m is the sum of all idle times between every two consecutive jobs on that machine and the *total non-processing energy* is a TFN defined as:

$$\widehat{\text{NPE}} = \sum_{m \in M} \left(P_m^{idle} \sum_{i,j \in J} \widehat{I}_{ijm} \right). \tag{12}$$

Unlike the makespan, the NPE (both the deterministic and fuzzy version) is not a regular performance measure. For instance, it can increase if we decrease the starting time of the first task in a machine while maintaining all other tasks (hence, job completion times) unchanged.

3.3. Bi-objective fuzzy job shop problem

In summary, we consider a bi-objective job shop scheduling problem with uncertain task processing times taking the form of triangular fuzzy numbers where the objective vector for any feasible schedule $\boldsymbol{s}$ is

$$\left(\widehat{C}_{max}, \widehat{\text{NPE}}\right). \tag{13}$$

The problem can be denoted $J|\widehat{p}_o|\left(\widehat{C}_{\max}, \widehat{\text{NPE}}\right)$ according to the usual $\alpha|\beta|\gamma$ notation introduced in [40].

Notice that both objective values $\widehat{C}_{\max}$ and $\widehat{\text{NPE}}$ belong to the set of TFNs, for which the relation $\leq_E$ provides a ranking, as explained in Section 2.3. Thus, even if the objective space is not a subset of $\mathbb{R}^2$, the Pareto front (or its approximation by a solving method) can still be visualised in $\mathbb{R}^2$ by identifying each objective with its expected value.

3.4. Bi-objective mixed integer programming model

It is possible to give a Mixed Integer Programming (MIP) model for the fuzzy JSP with makespan and NPE minimisation based on the $\leq_E$ ranking for fuzzy numbers. The parameters and variables of the model are given in Tables 1 and 2 respectively.

The first objective function in the model (Eq. (14)) corresponds to minimising the makespan and the second one, to minimising the non-processing energy, where minimisation is defined according to the ranking provided by $\leq_E$.

$$\min_{\substack{\widehat{I},\widehat{C} \\ \widehat{s},x}} \left(E[\widehat{C}_{\max}], E[\widehat{\text{NPE}}] \right) \tag{14}$$

s.t.

$$s^l_{j\, succ_j(m)} \geq s^l_{jm} + p^l_{jm} \qquad \forall j \in J, m \in M_j, \; l \in \{1,2,3\} \tag{15}$$

$$C^l_{\max} \geq s^l_{j\mu_j} + p^l_{j\mu_j} \qquad \forall j \in J, l \in \{1,2,3\} \tag{16}$$

$$s^l_{jm} \geq s^l_{im} + p^l_{im} - W \cdot (1 - x_{ijm}) \qquad \forall i,j \in J, m \in M_j \cap M_i, \; l \in \{1,2,3\} \tag{17}$$

$$I^1_{ijm} \geq s^1_{jm} - (s^3_{im} + p^3_{im}) - W(1 - x_{ijm}) \qquad \forall i,j \in J, m \in M_j \cap M_i \tag{18}$$

$$I^2_{ijm} \geq s^2_{jm} - (s^2_{im} + p^2_{im}) - W(1 - x_{ijm}) \qquad \forall i,j \in J, m \in M_j \cap M_i \tag{19}$$

$$I^3_{ijm} \geq s^3_{jm} - (s^1_{im} + p^1_{im}) - W(1 - x_{ijm}) \qquad \forall i,j \in J, m \in M_j \cap M_i \tag{20}$$

$$\text{NPE}^l = \sum_{m \in M} P_m^{idle} \Big(\sum_{\substack{i,j \in J \\ i \neq j}} I^l_{ijm} \Big) \qquad l \in \{1,2,3\} \tag{21}$$

$$\sum_{\substack{i,j \in J \\ i \neq j}} x_{ijm} = n_m - 1 \qquad \forall m \in M_j \cap M_i \tag{22}$$

$$\sum_{\substack{i \in J \\ i \neq j}} x_{ijm} \leq 1 \qquad \forall j \in J, m \in M_j \cap M_i \tag{23}$$

$$\sum_{\substack{j \in J \\ j \neq i}} x_{ijm} \leq 1 \qquad \forall i \in J, m \in M_j \cap M_i \tag{24}$$

$$x_{ijm} \in \{0,1\} \qquad \forall i,j \in J, m \in M_j \cap M_i \tag{25}$$

$$0 \leq I^1_{ijm} \leq I^2_{ijm} \leq I^3_{ijm} \qquad \forall i,j \in J, m \in M_j \cap M_i \tag{26}$$

$$0 \leq s^1_{jm} \leq s^2_{jm} \leq s^3_{jm} \qquad \forall j \in J, m \in M_j \tag{27}$$

Constraint (15) prevents the overlapping of the execution of two consecutive tasks from job j, stating that the task of job j to be processed on machine $succ(m)$ cannot start until after the job's task on machine m is completed. Constraint (16) defines the makespan. Constraint (17) ensures that a machine processes only one job at a time, so job j does not start being processed on machine m before the machine has finished processing job i, its immediate predecessor on m. Constraints (18–20) ensure that $\widehat{I}_{ijm}$ takes the value of the idle time between jobs i and j only if x_{ijm} is 1. Constraint (21) ensures that NPE will be the sum across all machines of the energy consumption of each machine while it is idle. Constraint (22) enforces that for machine m, there can be at most $n_m - 1$ precedence constraints (e.g. if 5 tasks are processed on a machine, there can be at most 4 idle intervals, so there will be $5 \times 4 = 20$ binary x variables for that machine but only 4 of them should be 1). Constraint (23) states that for each machine m and job j, there can be at most one job that comes immediately before job j (0 if it is the first job to be processed on that machine) and constraint (24) states that for each machine m and job i, there can be at most one job that comes immediately after job i (0 if it is the last job to be processed on that machine). Notice that constraints (22)-(25) ensure that x_{ijm} takes value 1 if and only if job i immediately precedes job j on machine m. Finally, constraints (26) and (27) ensure that idle times and starting times are non-negative and proper TFNs.

Table 2
Decision variables.

x_{ijm}	takes value 1 if job i immediately precedes job j on machine m, 0 otherwise
$\hat{s}_{jm}$	starting time of job j on machine m (TFN)
$\hat{I}_{ijm}$	idle time between consecutive jobs i and j on machine m (TFN)
$\hat{C}_{\max}$	makespan (TFN)
$\widehat{NPE}_{\max}$	non-processing energy (TFN)

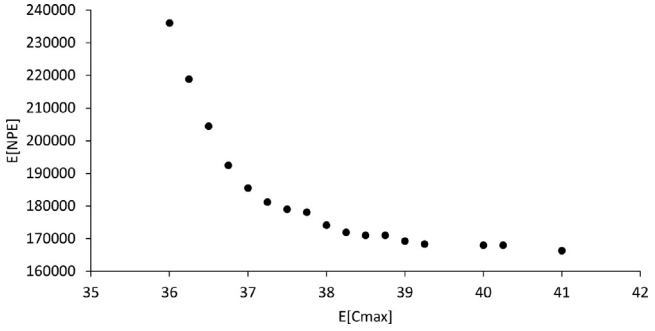

Fig. 1. Pareto front of a small instance solved by ϵ-constraint method.

3.5. Conflict between the objectives

In a bi-objective optimization problem like ours, if the two objective functions are correlated, the problem is called *trivial*, meaning that a single solution can simultaneously optimize both objectives [35]. Intuitively, one might think that decreasing the makespan could help in reducing the non-processing energy consumption, so it is natural to wonder if our problem falls into this trivial category. However, we shall see that this is not the case.

First, a small instance for FJSP with 6 jobs and 6 machines from [41] is solved with the ϵ-constraint method [42] using CPLEX. Ideal points ($E[\hat{C}_{\max}]^I$ and $E[\widehat{NPE}]^I$) are obtained by solving a MIP with the corresponding objective considering the constraints (15)-(27) whereas Nadir points ($E[\hat{C}_{\max}]^N$ and $E[\widehat{NPE}]^N$) are computed by solving the same MIPs with the additional constraints $E[\widehat{NPE}] \leq E[\widehat{NPE}]^I$ and $E[\hat{C}_{\max}] \leq E[\hat{C}_{\max}]^I$, respectively. The ϵ-constraints are introduced on $E[\hat{C}_{\max}]$ from $E[\hat{C}_{\max}]^I$ to $E[\hat{C}_{\max}]^N$. Since $\hat{C}_{\max}$ is a TFN, the minimum possible change in its expected value is chosen as ϵ value, which is 0.25. All the resulting problems are optimally solved and the Pareto front for the original instance is shown in Fig. 1. From this figure, it is clear that the two objectives are not correlated since it is not possible to decrease one of them without deteriorating the other one.

It is in fact possible to have cases where the energy consumption increases when the makespan is reduced. Indeed, let us consider a minimal working example with two machines and two jobs consisting of two tasks each, such that $p_{11} = (2,2,2)$, $p_{12} = (10,12,14)$, $p_{21} = (2,4,8)$ and $p_{22} = (1,2,3)$ and $P_1^{idle} = P_2^{idle} = 1$.

In a solution where job 1 is executed first on machine 1 and second on machine 2, that is, $x_{121} = x_{212} = 1$ and $x_{211} = x_{122} = 0$, the starting times are $\hat{s}_{11} = \hat{0}$, $\hat{s}_{21} = (2,2,2)$, $\hat{s}_{22} = \hat{0}$ and $\hat{s}_{12} = (1,2,3)$. Hence, the makespan is $\hat{C}_{\max} = (12,14,17)$ and machine idle times are $I_{121} = (0,0,1)$ and $I_{212} = (0,0,2)$, so $\widehat{NPE} = (0,0,3)$.

On the other hand, if job 1 is to be executed second on machine 1 and first on machine 2, that is, $x_{211} = x_{122} = 1$ and $x_{121} = x_{212} = 0$, the starting times turn out to be $\hat{s}_{21} = \hat{0}$, $\hat{s}_{11} = (2,4,8)$, $\hat{s}_{12} = \hat{0}$ and $\hat{s}_{22} = (10,12,14)$. In this case, the makespan has reduced to $\hat{C}_{\max} = (12,14,16)$, but now machine idle times are $I_{211} = (0,0,6)$ and $I_{122} = (0,0,4)$, so $\widehat{NPE}$ increases to (0,0,10).

To further illustrate the behaviour of the NPE under uncertainty, let us consider a final example with two jobs consisting of one task each so both tasks are executed on the same machine and with task processing times $\hat{p}_{11} = (2,2,2)$, and $\hat{p}_{21} = (2,4,8)$.

If job 1 is to be executed first and then job 2, that is, $x_{121} = 1$ and $x_{211} = 0$, the starting and completion times for both tasks will be $\hat{s}_{11} = \hat{0}$, $\hat{c}_{11} = (2,2,2)$, $\hat{s}_{21} = (2,2,2)$ and $\hat{c}_{21} = (4,6,10)$. Thus, the machine idle time between both tasks is $\hat{I}_{121} = \max\{\hat{0}, \hat{s}_{21} - \hat{c}_{11}\} = \max\{\hat{0}, (2,2,2) - (2,2,2)\} = \hat{0}$. Hence, for this processing order, the objective function values will be $\hat{C}_{\max} = (4,6,10)$ and $\widehat{NPE} = \hat{0}$.

On the other hand, if job 2 is to be executed first, followed by job 1, that is, $x_{211} = 1$ and $x_{121} = 0$, the starting and completion times for both tasks turn out to be $\hat{s}_{21} = \hat{0}$, $\hat{c}_{21} = (2,4,8)$, $\hat{s}_{11} = (2,4,8)$ and $\hat{c}_{11} = (4,6,10)$ and the machine idle time will be $\hat{I}_{211} = \max\{\hat{0}, \hat{s}_{11} - \hat{c}_{21}\} = \max\{\hat{0}, (2,4,8) - (2,4,8)\} = (0,0,6)$. Therefore, for this alternative processing order, the makespan value will still be $\hat{C}_{\max} = (4,6,10)$ (the same as with the first processing order) whereas the non-processing energy changes to $\widehat{NPE} = (0,0,6P_1^{idle})$.

We see here that not only is it possible for the energy to grow when the makespan is reduced but, also, due to the presence of uncertainty, the task processing order crucially influences the resulting non-processing energy costs, even if the makespan remains unchanged. This might contradict our intuition on the solution behaviour regarding energy costs and, quite importantly, it differs from the behaviour of non-processing energy costs in a deterministic job shop problem.

4. Multi-objective enhanced memetic algorithm

To solve the fuzzy job shop problem described in Section 3, we propose a multiobjective enhanced memetic algorithm. In its core lies a population-based method, based on the the well-known high-level template for multi-objective genetic algorithm NSGA-II [43] endowed with problem-specific coding, decoding and recombination operators. Since the synergy produced in MAs by combining different population-based and local-search metaheuristics has proved to be very powerful in solving scheduling problems [18,44,45], we propose to complement the genetic algorithm's diversification capabilities, with three domain-dependent, single-objective and single-solution intensification procedures: a tabu search (TS) for makespan (explained in Section 4.2), a heuristic procedure (HER) to reduce NPE (Section 4.3) and a linear programming postprocess (Section 4.4) also aimed at reducing NPE. We believe this to be a reasonable way of incorporating the problem-domain knowledge given the different nature of the two objectives considered herein. Indeed, most problem-dependent neighbourhood structures for FJSP that exhibit nice properties, such as feasibility or connectivity, are only defined for regular performance measures, which includes the makespan but not the NPE. Hence, instead of using a myopic neighbourhood for energy (or simply not having neighbours for energy) in a multi-objective local search, we opt for a single-objective local search exploring promising areas of the search space in terms of makespan followed by a heuristic refinement of the resulting schedules in terms of NPE during the evolution, followed by a final refinement also in terms of NPE.

The memetic algorithm starts from a randomly generated population. Unlike in NSGA-II, selection is not performed using tournament. Instead, a lower selective pressure is obtained by randomly pairing all individuals in the population, so every one has a chance to pass its genes onto the following generation thus contributing to promote diversity and avoid premature convergence. Then, each pair is combined using crossover and mutation, so two offspring are obtained. During the evolutionary process, the tabu search (TS) and the heuristic procedure

HER are applied to improve the individuals of the population, following a Lamarckian learning model [32]. Specifically, at each iteration, once an offspring population is obtained, each offspring is improved in two steps. First, the TS uses a neighbourhood structure specifically designed to improve the makespan of each individual so changes produced by the TS in the processing order of operations are translated back into the individual. Second, the solution produced by the TS is subject to a heuristic method to reduce its non-processing energy consumption. A new population is then formed by applying non-dominated sort and crowding distance to the union of the old population and the set of twice-improved offspring. An additional mechanism to preserve diversity and avoid premature convergence is introduced at this step as explained in Section 4.1.

The evolutionary process is left to run until the stopping criterion is met. Instead of having a fixed number of generations, as in NSGA-II, a more dynamic criterion is adopted, so the evolution continues until a fixed number of consecutive iterations $\max_{\text{MOEMA}}$ pass without improvement, that is, without adding a new solution to the set of non-dominated solutions. Once the evolution process is finished, the resulting non-dominated solutions are further improved to obtain optimal non-processing energy values using an exact MIP commercial solver. More precisely, from each schedule we obtain a linear programming version of the problem by maintaining the job processing orders established by the schedule. The commercial solver then modifies task starting times so each schedule is guaranteed to have optimal non-processing energy consumption without increasing the makespan.

A pseudocode description of the resulting multiobjective enhanced memetic algorithm, called MOEMA hereafter, can be found in Algorithm 1.

Algorithm 1 Multi-Objective Enhanced Memetic Algorithm MOEMA.

Require: An FJSP instance
Ensure: A schedule
1: $Best \leftarrow \emptyset$ *//set of non-dominated solutions so far*
2: Generate a pool P_0 of random solutions.*//initial population*
3: Evaluate each chromosome of P_0 using InsertionSGS
4: Apply TS to each schedule and rebuild the chromosomes
5: Apply HER to each schedule and update the vector of fitness values
6: $i \leftarrow 0$ *//Counter for iterations without changes*
7: **while** $i < \max_{\text{MOEMA}}$ **do**
8: *//Obtain offspring from current population* P_i
9: $\text{Off}(P_i) \leftarrow$ Apply selection and recombination to P_i
10: Evaluate $\text{Off}(P_i)$;
11: *//Improve offspring with lamarckism*
12: Apply TS to all schedules in $\text{Off}(P_i)$ and rebuild the chromosomes
13: Apply HER to all schedules in $\text{Off}(P_i)$ and update fitness values
14: *//Obtain new population and update set of non-dominated solutions*
15: $P_{i+1} \leftarrow$ Apply non-dominated and crowding distance sorting to $P_i \cup \text{Off}(P_i)$
16: $Best \leftarrow$ non-dominated solutions in $Best \cup P_{i+1}$
17: **if** *Best* remains unchanged **then**
18: $i \leftarrow i + 1$;
19: **else**
20: $i \leftarrow 0$;
21: **end if**
22: **end while**
23: *//LP post-processing of non-dominated solutions*
24: **for** each solution s in *Best* **do**
25: $LP(s) \leftarrow$ obtain linear programming model from s
26: $s \leftarrow$ apply mathematical solver to $LP(s)$
27: **end for**
28: $Best \leftarrow$ non-dominated solutions in *Best//update Best*
29: **return** *Best*

donor: (1 2 2 3 3 2 1 1 3)
receiver: (1 ~~1~~ ~~3~~ 2 2 1 ~~2~~ ~~3~~ 3)
offspring: (1 2 3 3 2 1 2 1 3)

Fig. 2. Example of GOX, with the selected substring of the donor underlined and the corresponding deleted genes crossed out in the receiver.

In the following, we describe in more detail the algorithm's components: the representation chosen to code chromosomes together with the decoding and evaluation procedure and the genetic operators; the tabu search, including the neighbourhood structure and the mechanisms used in this algorithm to speed-up the search; the heuristic energy reduction procedure; and the solution post-processing using linear programming.

4.1. Problem-specific evolutionary operators

The NSGA-II template needs to be furnished with problem-specific operators, including the coding and the decoding of individuals, together with their evaluation. Here, we propose that a genotype codifies a solution to the FJSP as a permutation with repetitions [46]. This is a permutation of the set of tasks, where each task o_{jm} is represented by its job number j. One advantage of this coding scheme is that it refers only to feasible schedules and, conversely, every feasible schedule can be thus encoded.

For instance, in a JSP instance with 3 jobs and 3 tasks per job, if a schedule induces the task processing order $o_{11} - o_{21} - o_{22} - o_{31} - o_{32} - o_{23} - o_{12} - o_{13} - o_{33}$, this can be coded as the chromosome (1 2 2 3 3 2 1 1 3) (and viceversa).

During the evaluation phase, every chromosome is decoded by generating an associated schedule and computing the vector of fitness values $(\widehat{C}_{\max}, \widehat{\text{NPE}})$. A good knowledge and understanding of subspaces of schedules and their properties is of key importance when designing decoding strategies. The reason is that in many cases, obtaining schedules pertaining to a particular subspace allows to reduce the search space without loosing the possibility of finding optimal solutions. Of particular interest is the subspace of *active schedules* [1]. These are feasible schedules such that no task may start earlier without delaying the starting time of another. It is common practice in deterministic scheduling to restrict the search of an optimal solution to this subspace, since it is guaranteed to contain an optimal solution for any regular performance measure, in particular, for the makespan. This is still the case in the fuzzy job shop, where active fuzzy schedules can be obtained from a chromosome using the algorithm ActiveSGS from [39]. This algorithm is shown to be complete in the set of active schedules (that is, it can be used to generate all schedules in this category) and, therefore, it can generate an optimal solution.

Regarding other genetic operators, selection is performed with random pairs. For recombination, we propose to use Generalised Order Crossover (GOX), a specialised crossover operator for the JSP coding based on permutations [47] together with an inversion mutation operator [48], since they have shown good performance in multiobjective FJSP problems [49]. To generate an offspring from two parent chromosomes (referred to as donor and receiver respectively), GOX randomly selects a substring from the donor choromosome. Then, all genes of that substring are deleted in the receiver (taking into account the relative position of job numbers in the chromosome). Finally, GOX implants the donor's substring into the receiver at the position where the first gene of the substring occurred before deletion in the receiver. An example of GOX for a problem of 3 jobs and 3 tasks per job can be seen in Fig. 2. To generate a second offspring, it suffices to swap the parents' roles.

Regarding the inversion mutation, two positions in the chromosome are randomly selected and the subsequence of genes between these two positions are inverted. An example of this mutation can be seen in Fig. 3.

Finally, the replacement strategy follows the NSGA-II schema with an additional diversity preserving mechanism. Starting from a popula-

original chromosome: (1 2 2 3 3 2 1 1 3)
mutated chromosome: (1 2 2 1 2 3 3 1 3)

Fig. 3. Example of inversion mutation, with the substring between the two selected positions to be inverted underlined.

tion P_i, a pool of solutions $\mathrm{Off}(P_i)$ is built from P_i by applying selection, crossover and mutation to the chromosomes in P_i and, following decodification, improvement via tabu search and heuristic energy reduction. Then, the next generation P_{i+1} is obtained from $P_i \cup \mathrm{Off}(P_i)$. As a first step, individuals with identical objective function values are filtered from $P_i \cup \mathrm{Off}(P_i)$ to avoid repeated elements. This yields a new set of individuals P'. P_{i+1} is then obtained from the selection operator based on non-dominated sorting and crowded distance sorting: P_{i+1} is initialised with the non-dominated solutions from P'; if P_{i+1} is not yet complete, the non-dominated solutions are removed from P' and the process is repeated with the remaining ones until $|P_{i+1}| \geq |P_i|$, where $|P|$ denotes the size of a population P. If $|P_{i+1}|$ exceeds $|P_i|$, a crowding distance criterion is applied to filter out the least diverse solutions from the last set of solutions added to P_{i+1}.

4.2. Tabu search

Tabu search [50,51], is an advanced local search technique that allows selecting non-improving neighbours in order to escape from local optima. It is often used in combination with population-based metaheuristics in what constitutes a classical memetic algorithm, so the tabu search provides exploitation while the other metaheuristic provides exploration.

A key component for the success of a local search is the use of problem-dependent neighbourhood structures. Several neighbourhoods have been proposed in the literature for FJSP based on reversing critical arcs in a graph representation of an schedule. This representation, known as disjunctive graph, varies slightly depending on the variant of the problem under consideration. Here we use the disjunctive graph structure and criticality concepts defined for the FJSP with makespan minimisation in [52]. Each schedule can be represented by a solution graph in a similar manner to deterministic schedules: each node corresponds to a task and there are two additional nodes representing the start and the end of the project; so-called conjunctive (directed) arcs connect the start node with the first task of each job, each task with its successor in its job and the last task of each job with the end node, while so-called disjunctive (directed) arcs connect each task with its immediate successor in the machine (according to the schedule). There is however a crucial difference: all arcs are weighted with the processing time of the task at the origin, a TFN. In consequence, the concepts of critical path and critical arc are significantly redefined taking into account the particulars of TFN arithmetic. Specifically, the introduction of uncertainty produces a considerable rise in the number of critical arcs. Given a solution graph and the redefined concepts of criticality, we consider a neighbourhood structure from [53] based on moves that reverse arcs at the extreme of critical blocks (that is, maximal sequences of tasks in a critical path requiring the same machine). All neighbours thus generated are feasible and, most importantly, solutions generated by reversing any other arcs can never improve the expected makespan. Fig. 4 illustrates a move of this neighbourhood structure. Dashed arcs represent precedence between tasks in a machine while solid arcs represent precedence between tasks in a job. In bold in the original solution (before move) we see the arc between two tasks at the extreme of a critical block, o_{im} and o_{jm}. This is reversed to form the neighbour (after move) and, in doing so, arcs from the predecessor in the machine of o_{im} (denoted PM_{im}) and to the succesor in the machine of o_{jm} (denoted SM_{jm}) also change. On the other hand, arcs from job predecessors (denoted PJ) and to job successors (SJ) of both tasks do not change. For further detail the interested reader is referred to [52,53].

The TS procedure using the above neighbourhood structure can be seen in Algorithm 2. It incorporates two mechanisms to speed-up the

Algorithm 2 Tabu search algorithm.

Require: A *FJSP* instance, a feasible initial solution s and a maximum number of iterations max_{TS}
Ensure: A schedule
1: $best \leftarrow s$ *//Best solution found so far*
2: $TList \leftarrow \emptyset$ *//Tabu list (initially empty)*
3: $i \leftarrow 0$ *//Number of iterations without improvement*
4: **while** $i < max_{\mathrm{TS}}$ **do**
5: Compute $\mathcal{N}$ the neighbourhood of s
6: Sort $\mathcal{N}$ in ascending order using the makespan estimate $\mathrm{LB}(\widehat{C}_{\max})$
7: *//Find the best neighbour* s^*
8: $\widehat{C}^*_{\max} \leftarrow (W, W, W)$
9: **for each** $s' \in \mathcal{N}$ **do**
10: **if** $\mathrm{LB}(\widehat{C}_{\max}(s')) <_E \widehat{C}^*_{\max}$ **then**
11: $v \leftarrow$ arc reversed in s to form s'
12: **if** $v \notin TList$ **or** $\widehat{C}_{\max}(s') <_E \widehat{C}_{\max}(best)$ **then**
13: **if** $\widehat{C}_{\max}(s') <_E \widehat{C}^*_{\max}$ **then** *//found a better neighbour*
14: $\widehat{C}^*_{\max} \leftarrow \widehat{C}_{\max}(s')$
15: $s^* \leftarrow s'$
16: **end if**
17: **end if**
18: **end if**
19: **end for**
20: *//Update tabu list, best solution so-far, iteration count and current solution*
21: $v^* \leftarrow$ arc reversed in s to form s^*
22: Update $TList$ with v^*
23: **if** $\widehat{C}_{\max}(s^*) <_E \widehat{C}_{\max}(best)$ **then**
24: $best \leftarrow s^*$
25: $i \leftarrow 0$
26: **else**
27: $i \leftarrow i + 1$
28: **end if**
29: $s \leftarrow s^*$
30: **end while**
31: **return** *best*

evaluation of the neighbours. The first one is based on computing the fuzzy makespan after a move using forward propagation in the graph representing the solution (as proposed in [53]). This avoids a full evaluation, recomputing only the solutions components that were modified by the move (as it is desirable in local search embedded in MAs [54]). The second speed-up mechanism is a neighbour filtering strategy adapted from [55] that uses low-cost lower bound estimates of the makespan, denoted $\mathrm{LB}(\widehat{C}_{\max})$, to discard non-improving neighbours without fully evaluating them (lines 6–9 of Algorithm 2). This filtering reduces the size of the neighbourhood and thus increases the chances for improvement. Filtering and constraint propagation are particularly important in the fuzzy framework as the number of feasible neighbours of a solution is considerably larger than in the crisp case due to the increased number of critical arcs. The tabu list stores tabu moves. The stopping criterion is a number max_{TS} of consecutive iterations without improving the current solution.

4.3. Heuristic reduction of non-processing energy consumption

In [27] a study is conducted to reduce idle times and the associated $\widehat{\mathrm{NPE}}$ value in a fuzzy schedule based on the concept of local right shift, which intuitively consists in "moving a task block to the right on the Gantt chart while preserving the task sequence".

Now, if we perform a local right shift of o_{im}, the task of job i to be processed on machine m, we delay its starting time and, hence, its com-

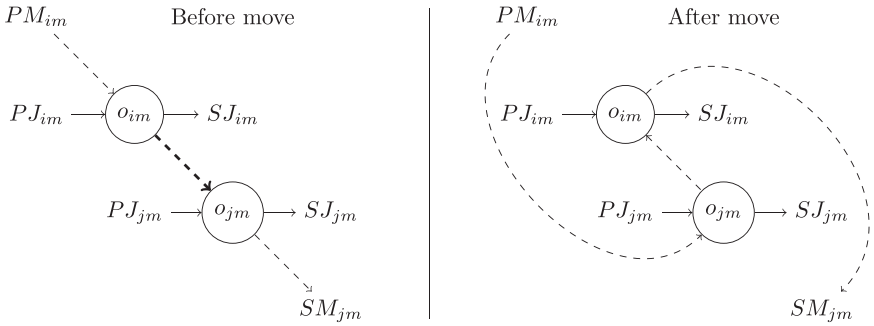

Fig. 4. Graphic representation of a move from the neighbourhood used in the TS.

pletion time. To preserve feasibility, we need to take into account that the execution of o_{im} cannot overlap with the execution of tasks succeeding o_{im} in its job or its machine (if they exist). Clearly, the latest possible starting time of o_{im} satisfying machine feasibility $\widehat{lsM}_{im}$ is given by:

$$lsM^l_{im} = s^l_{jm} - p^l_{im}, l = 1, 2, 3 \tag{28}$$

if i is not the last job to be processed on m and j denotes the job to be processed on machine m right after i (i.e. $x_{ijm} = 1$), while $lsM^l_{im} = \infty$ otherwise. Analogously, the latest possible starting time of o_{im} satisfying job precedence constraints $\widehat{lsJ}_{im}$ is given by:

$$lsJ^l_{im} = s^l_{isucc_i(m)} - p^l_{im}, l = 1, 2, 3 \tag{29}$$

if o_{im} is not the last task in job i (i.e., m is not the last machine where i needs to be processed), with $lsJ^l_{im} = \infty$ otherwise.

Based on this, together with Lemmas 1 y 2 from [27], it is immediate to prove the following proposition, which establishes which is the largest local right shift of a task that does not increase neither $E[\widehat{\text{NPE}}]$ nor $E[\widehat{C}_{\max}]$, is not infeasible in the sense that it does not produce overlaps with the successor in the machine or the job and such that the new starting time is a TFN.

Proposition 1. *Let o_{im} be the task of job i to be processed on machine m. Then, the latest possible starting time for o_{im} obtained with a local right-shift of this task such that is it is feasible and it is non-increasing in $E[\widehat{\text{NPE}}]$ and in $E[\widehat{C}_{\max}]$ is given by the following:*

- *If i is the first job to be processed on machine m:*

$$\begin{aligned} s'^3_{im} &= \min\left\{lsM^3_{im}, lsJ^3_{im}\right\} \\ s'^2_{im} &= \min\left\{s'^3_{im}, lsM^2_{im}, lsJ^2_{im}\right\} \\ s'^1_{im} &= \min\left\{s'^2_{im}, lsM^1_{im}, lsJ^1_{im}\right\} \end{aligned} \tag{30}$$

- *If i is neither the first nor the last job to be processed on machine m and j is the job to be processed on machine m right after i (i.e. $x_{ijm} = 1$):*

$$\begin{aligned} s'^3_{im} &= \min\left\{\max\left\{s^3_{im}, s^1_{jm} - p^3_{im}\right\}, lsJ^3_{im}\right\} \\ s'^2_{im} &= \min\left\{s'^3_{im}, lsM^2_{im}, lsJ^2_{im}\right\} \\ s'^1_{im} &= \min\left\{s'^2_{im}, lsM^1_{im}, lsJ^1_{im}\right\} \end{aligned} \tag{31}$$

- *If i is the last job to be processed on machine m and k is the job to be processed on machine m right before i (i.e. $x_{kim} = 1$):*

$$\begin{aligned} s'^3_{im} &= s^3_{im} \\ s'^2_{im} &= s^2_{im} \\ s'^1_{im} &= \min\left\{s'^2_{im}, \max\left\{c^3_{km}, s^1_{im}\right\}, lsJ^1_{im}, C^1_{\max} - p^1_{im}\right\} \end{aligned} \tag{32}$$

As explained above, during the evaluation phase, every chromosome is decoded and an associated schedule is generated for which the vector of fitness values $(\widehat{C}_{\max}, \widehat{\text{NPE}})$ is computed. Then, the tabu search process may modify the processing order of jobs on the machines and, if this were the case, the schedule, the vector of fitness values and the chromosome are updated. We propose that, at this point, local right shifts are applied to tasks in inverse order in which they are scheduled based on Proposition 1, in an attempt to improve the $\widehat{\text{NPE}}$ value without changing $\widehat{C}_{\max}$. We shall refer to this strategy as Heuristic Energy Reduction, or HER for short. The detailed procedure can be seen in Algorithm 3, where PM_o and SM_o denote, respectively, the predecessor and successor of task o in the machine sequence.

Algorithm 3 Heuristic Energy Reduction procedure HER.

Require: a fuzzy JSP P, a schedule s that codifies a chromosome z
Ensure: a better or equal value for $\widehat{\text{NPE}}$ of s
1: θ is the array of tasks codified by s
2: **while** θ is not empty **do**
3: remove o the last task in θ
4: **if** o is the last task scheduled in a machine **then**
5: $s'^3_o = s^3_o$
6: $s'^2_o = s^2_o$
7: $s'^1_o = \min\{s'^2_o, \max\{c^3_{PM_o}, s^1_o\}, lsJ^1_o, C^1_{\max} - p^1_o\}$
8: **else** *//not the last task in a machine*
9: **if** o is the first task scheduled in a machine **then**
10: $s'^3_o = \min\{lsM^3_o, lsJ^3_o\}$
11: **else** *//not the first task in a machine*
12: $s'^3_o = \min\{\max\{s^3_o, s^1_{SM_o} - p^3_o\}, lsJ^3_o\}$
13: **end if**
14: $s'^2_o = \min\{s'^3_o, lsM^2_o, lsJ^2_o\}$
15: $s'^1_o = \min\{s'^2_o, lsM^1_o, lsJ^1_o\}$
16: **end if**
17: **end while**
18: Compute $\widehat{\text{NPE}}$ of s';
19: Update the vector of objective values of z with the new $\widehat{\text{NPE}}$;

Notice that the job processing orders on the different machines remain unchanged in this algorithm and only some starting and completion times may change. Thus, there is not change in the chromosome when HER procedure is applied, it may only change the vector of fitness value.

The idea of using a heuristic method to improve the energy consumption of an already built schedule is not new, as shown, for instance, in [6,21,24,28]. However, the heuristic method is highly dependent on the problem at hand, so different proposals are not comparable. For example, in [27] it has been shown how the heuristic procedure must be specifically designed from scratch when uncertainty is incorporated to the problem, since this single change makes the already-existing proposal for the deterministic counterpart of the problem [28] simply non-applicable.

4.4. Linear programming postprocessing

Mathematical models are frequently used to obtain the optimal solution for small-size instances in scheduling problems [23,56,57]. For larger instances, however, MIP solvers are often unable to reach even a feasible solution within reasonable time limits and metaheuristics are instead the most successful approaches.

Trying to have the best of both worlds, in MOEMA we integrate a MIP commercial solver that uses the set of non-dominated solutions obtained after the evolutionary process as starting points (see lines 18 to

20 in Algorithm 1). Since the tabu search embedded in the evolutionary process produces an intensive minimisation of the makespan objective, we propose that the solver considers only the $\widehat{\text{NPE}}$ value and devotes itself to searching for more energy-efficient solutions. More precisely, the first objective in (14) of the MIP model from Section 3.4 is removed and, instead, the $E[\widehat{C}_{\max}]$ value of the schedule is introduced as an upper bound in a new constraint.

Notice that in the MIP model from Section 3.4 there is only one set of integer variables, namely $\boldsymbol{x} = \{x_{ijm} : i, j \in \boldsymbol{J}, m \in \boldsymbol{M}\}$, defining the job-processing order on the machines. Hence, once a job order is fixed for every machine, as it happens on line 19 of Algorithm 1, the model becomes a linear program (LP). In that case, the resulting model finds optimal values for the starting times, makespan and idle-times with respect to $\boldsymbol{x}$. Let us denote them $\hat{s}(\boldsymbol{x})$, $\widehat{C}_{\max}(\boldsymbol{x})$ and $\widehat{I}(\boldsymbol{x})$ respectively, to highlight the fact that they depend on the job order fixed by $\boldsymbol{x}$, which is no longer a decision variable, but a parameter fixed in advance from a schedule $\boldsymbol{s}$. As a result, constraint (17) becomes

$$s^l_{jm}(\boldsymbol{x}) \geq s^l_{im}(\boldsymbol{x}) + p^l_{im} \quad \forall i, j \in \boldsymbol{J}, \forall m \in \boldsymbol{M}_j \cap \boldsymbol{M}_i, x_{ijm} = 1, l \in \{1, 2, 3\} \quad (33)$$

Similarly, constraints (18)–(20) would now be

$$I^1_{ijm}(\boldsymbol{x}) \geq s^1_{jm}(\boldsymbol{x}) - (s^3_{im}(\boldsymbol{x}) + p^3_{im}) \quad \forall i, j \in \boldsymbol{J}, m \in \boldsymbol{M}_j \cap \boldsymbol{M}_i, x_{ijm} = 1 \quad (34)$$

$$I^2_{ijm}(\boldsymbol{x}) \geq s^2_{jm}(\boldsymbol{x}) - (s^2_{im}(\boldsymbol{x}) + p^2_{im}) \quad \forall i, j \in \boldsymbol{J}, m \in \boldsymbol{M}_j \cap \boldsymbol{M}_i, x_{ijm} = 1 \quad (35)$$

$$I^3_{ijm}(\boldsymbol{x}) \geq s^3_{jm}(\boldsymbol{x}) - (s^1_{im}(\boldsymbol{x}) + p^1_{im}) \quad \forall i, j \in \boldsymbol{J}, m \in \boldsymbol{M}_j \cap \boldsymbol{M}_i, x_{ijm} = 1 \quad (36)$$

Also, constraints (22)–(25) disappear when the values of the variables $\boldsymbol{x}$ are fixed while constraints (15), (16), (21), (26) and (27) remain essentially the same other than the change in the notation of the variables. Finally, the solution $\boldsymbol{s}$ found after the evolutionary process is set to be an initial starting point for this linear model that aims at minimising the non-processing energy consumption without increasing the initial expected makespan value. This translates in the addition of the following constraint:

$$E[\widehat{C}_{\max}(\boldsymbol{x})] \leq E[\widehat{C}^{init}_{\max}] \quad (37)$$

where $E[\widehat{C}^{init}_{\max}]$ is the expected makespan for the non-dominated solution $\boldsymbol{s}$.

It could be thought that incorporating the LP solver makes the heuristic component HER redundant in MOEMA. We will however show in the experimental study that this is not the case. The heuristic procedure is applied to every individual in the population along the evolutionary process and it thus helps in obtaining solutions with reasonably good NPE values with very little computational effort. This in turn helps in guiding the multi-objective evolution towards promising regions of the search space. In consequence, the non-dominated solutions fed to the mathematical solver provide a good starting point so the solver can reach in few seconds optimal energy values for the given job processing orders.

Alternatively, we have run some experiments where the original MIP model is considered and only the makespan and non-processing values from the non-dominated solution are taken as upper bounds. However, in thie case the MIP solver needs an excessively large amount of time to solve the problem. The commercial solver IBM ILOG CPLEX [34] is used both for preliminary empirical tests on the MIP model and for solving the LP formulation.

5. Experimental study

We now present an experimental study of the behaviour of the proposed algorithm MOEMA. As pointed out in [32], the success of MAs is most likely a direct consequence of the synergy of the different search approaches they incorporate. To assess this synergy effect in our hybrid algorithm, we shall first compare different versions of the evolutionary part of MOEMA, namely, the population-based algorithm on its own or incorporating either one of or both the TS procedure and the heuristic HER. In a second stage, we will evaluate the contribution of the LP post-processing.

5.1. Problem instances, metrics and running conditions

For the experimental study, we shall consider twelve FJSP instances from the literature [14] to which we incorporate machine idle power consumption values. They are fuzzy versions of 12 well-known benchmark problems for deterministic job shop which are considered hard to solve: FT10 (size 10×10), FT20 (20×5), La21, La24, La25 (15×10), La27, La29 (20×10), La38, La40 (15×15), and ABZ7, ABZ8, ABZ9 (20×15), where the notation $n \times m$ indicates that the problem instance has n jobs and m machines. Machine idle power consumption levels P^{idle}_k are those proposed by [9] and used in [27].

Comparisons between any two multi-objective versions of the solving method will be made either in terms of hypervolume (HV) and the unary additive ϵ indicator ($I_{\epsilon+}$) [58] with pair-wise differences between empirical attainment functions (EAF) [59] using the visualisation tool from [60].

In absence of the optimal Pareto front, PO^*, we approximate the reference set by the non-dominated elements in the union of all sets of solutions obtained across all experiments, denoted S, as proposed in [48]. All these reference sets, together with the detailed results obtained by the proposed MOEMA are included as supplementary material. Also, when using these metrics, it is advised to normalise the values of the objective functions. If $f_i(s)$ denotes the value of the i-th objective function for a solution $\boldsymbol{s} \in S$, then its normalised value is given by:

$$f_i(\boldsymbol{s}) = \frac{f_i(\boldsymbol{s}) - f_i^-(S)}{f_i^+(S) - f_i^-(S)}, \quad (38)$$

where $f_i^-(S) = \min\{f_i(\boldsymbol{s}) : \boldsymbol{s} \in S\}$ is a lower bound of the objective function f_i in the set S, and $f_i^+(S) = 1.05 \times \max\{f_i(\boldsymbol{s}) : \boldsymbol{s} \in S\}$ is an upper bound thereof. The correction factor 1.05 helps avoiding null HV values, which can be troublesome in different cases.

All the experiments reported in this section correspond to a C++ implementation and run on a PC with Intel Xeon Gold 6132 processor at 2.6 Ghz and 128 Gb RAM with Linux (CentOS v6.10).

As a result of a preliminary parametric analysis, the parameter setup for MOEMA is the following: population size 100, GOX crossover with probability 1.0, inversion mutation with probability 0.1, $max_{\text{MOEMA}} = 25$ consecutive iterations without improvement as stopping criterion for the genetic algorithm and $max_{\text{TS}} = 10$ as stopping criterion for the local search and tabu list size equal to $8 + (n + m)/3$ (which is the largest list size adopted in [61]). The commercial solver runs for each instance until it reaches the optimal value, taking less than one second per solution.

To illustrate how the dynamic stopping criterion affects the convergence of the algorithm, Fig. 5 shows for instance La27 the evolution of the multiobjective metrics HV and $I_{\epsilon+}$ for a single run of MOEMA when max_{MOEMA} is set to 25 and to 50. We can appreciate that after the generation where the algorithm stops when $max_{\text{MOEMA}} = 25$ the improvement in multiobjective metrics is marginal. Indeed, when changing from 25 to 50, HV improves less than 0.72% and $I_{\epsilon+}$ improves less than 0.83% and, regarding the objective functions, $E[\widehat{C}_{\max}]$ improves less than 0.5% and $E[\widehat{NPE}]$ improves less than 1.2%.

The behaviour on this particular instance is illustrative of the algorithm's behaviour on all instances. On average, going from $max_{\text{MOEMA}} = 25$ to $max_{\text{MOEMA}} = 50$ implies 57.2% more iterations with a 55% CPU time increase. However, the improvement in $E[\widehat{C}_{\max}]$ and $E[\widehat{NPE}]$ is below 0.05% and 1.06% respectively, HV improves 1.2% and $I_{\epsilon+}$ 5.6%. In summary, there is a considerable increase in computational effort for a marginal improvement in solution quality.

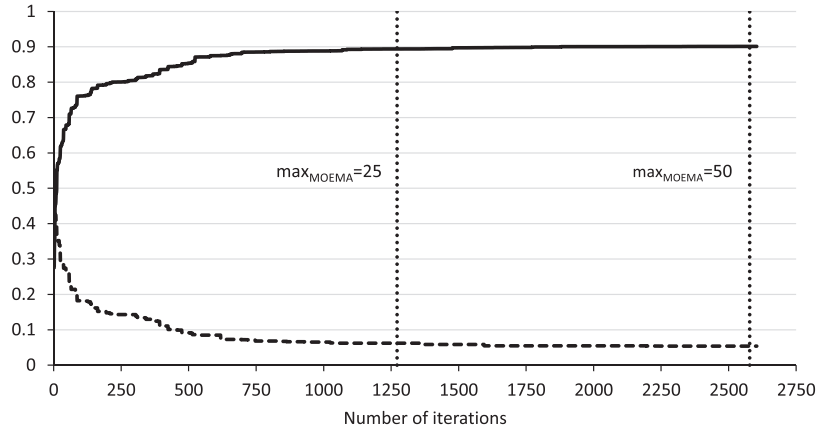

Fig. 5. Evolution of the hypervolume (solid line) and the ϵ-indicator (dashed line) for instance La27 along a single run of MOEMA when the stopping criterion is $max_{\text{MOEMA}} = 25$ or $max_{\text{MOEMA}} = 50$. The y axis corresponds to the multiobjective metrics value and the x-axis, to the iterations of the algorithm.

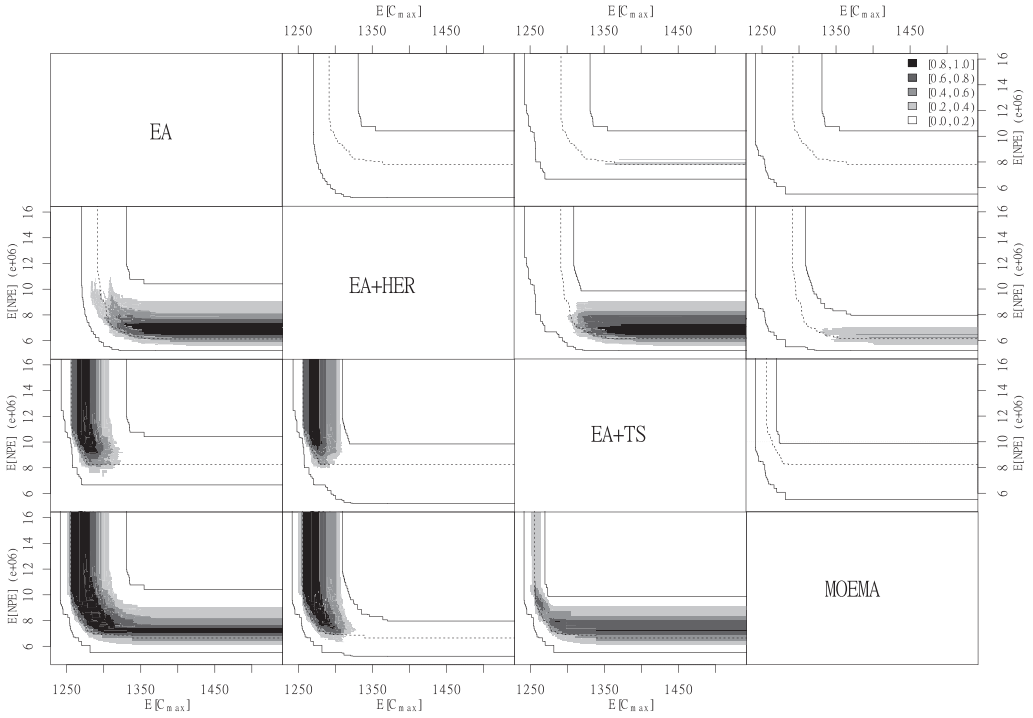

Fig. 6. Differences between EAF plots for instance.La27 .

5.2. Synergy between diversification and intensification during the evolution

A first series of experiments is oriented to asses the contribution of the intensification provided by TS and HER to the baseline evolutionary algorithm. We shall compare four versions of the solving method, depending on the components it incorporates: the evolutionary algorithm followed by the LP postprocessing without any intensification method during the evolution (referred to as EA in the following), the evolutionary algorithm incorporating TS and LP but not HER (EA+TS), the evolutionary algorithm incorporating HER but without TS followed by LP (EA+ HER) and the complete enhanced memetic algorithm MOEMA. For the sake of fair comparisons, the stopping criterion is changed so all methods are left to run for the same time as it takes MOEMA to converge.

Fig. 6 depicts for instance La27 pairwise comparisons for the four algorithms in terms of empirical attaintment functions (EAFs), plotting the difference in EAF values between each pair of algorithm. The behaviour on the remaining instances is similar. In the light of the mosaic in this figure, we can extract some conclusions regarding the different problem-specific components of MOEMA.

First, we can analyse the contribution of the TS component when it is incorporated to the plain evolutionary algorithm in order to intensify the search with respect to the makespan, that is, EA compared to EA+TS (cells (1,3) and (3,1) of the mosaic). According to these EAF plots, the probability that EA dominates EA+TS is null almost everywhere except for a very small area of the objective space (where makespan values are worse), where this probability remains very low (between 0.2 and 0.4). On the contrary, EA+TS clearly dominates EA in the region where the makespan is better (low values on the x-axis), with probability close to 1. This is an expected behaviour since TS focuses on optimising makespan.

Second, we can see what happens when TS is incorporated to the evolutionary component when this already includes the heuristic HER for energy reduction, that is, compare EA+HER to MOEMA (cells (2,4) and (4,2) of the mosaic). Here, the effect of incorporating TS is similar to the previous case: there is an area (corresponding to smaller non-processing energy values and larger makespan ones) where there is a low probability that EA+HER dominates MOEMA. On the other hand, MOEMA has a high probability (close to 1 in a big area) of dominating EA+HER both at the center of the objective space and in the area with small makespan values. Notice as well that the upper and lower lines are

located at lower values on the y-axis in this second case compared to the first one. This is a natural consequence of the fact that in this second case both algorithms incorporate the two energy reduction components (the heuristic HER and the LP post-processing), so all the solutions obtained with both algorithms have good energy values.

From these two pair-wise comparisons we can conclude that incorporating TS for makespan intensification into the search process significantly contributes to the quality of the obtained set of non-dominated solutions.

We can also analyse the effect of the heuristic component HER in an analogous manner. On the one hand, there is the pair-wise comparison between the plain evolutionary algorithm with LP postprocessing (EA) and the variant that also includes HER (EA+HER), that is, (cells (1,2) and (2,1) of the mosaic). On the other hand, we have the comparison between the hybrid algorithm without and with HER, that is, EA+TS versus MOEMA (cells (3,4) and (4,3)). The results are quite similar to those obtained above for TS: we can see that introducing the heuristic HER for energy reduction during the evolution has a strong influence in the quality of solutions at the center of the objective space and in the area with lowest non-processing energy values (low values on the y-axis). When HER is included in the solving method the obtained solutions dominate those obtained without HER in a big area of the objective space, with big probability in those areas corresponding to small non-processing energy values (lowest values of the y-axis). Notice that this is always the case even if we always apply a LP post-processing to find optimal energy values of the set of non-dominated solutions obtained after evolution.

It is also possible to analyse what happens when we incorporate only the TS method or HER into the evolutionary algorithm, that is, EA+TS versus EA+HER (cells (3,2) and (2,3) in the mosaic). As expected, EA+TS dominates EA+HER in the area with low makespan values while EA+HER dominates EA+TS in the area with lower non-processing energy cost.

Finally, we can compare the plain evolutionary algorithm with LP post-processing (EA) to the enhanced memetic algorithm incorporating both intensification features into the evolution (MOEMA). If we look at cells (1,4) and (4,1) in the mosaic, it is clear that EA is dominated by MOEMA in every region and never dominates it.

5.3. Contribution of LP post-processing

We analyse the synergy effect between the evolutionary part of MOEMA (that is, the multiobjective evolutionary algorithm incorporating TS and HER, denoted EA+TS+HER hereafter) and the exact LP method used as post-process in two different ways. First, we compare EA+TS+HER with MOEMA in terms of hypervolume and ϵ-indicator to assess the contribution of this post-process. Then, we analyse if the LP method benefits from using the evolved set of non-dominated solutions as starting points. To this end, we run the commercial solver for the LP formulation starting from a set of non-dominated solutions obtained by building schedules with the ActiveSGS procedure.

Table 3 summarises the results obtained after 30 runs of EA+TS+HER (the memetic algorithm without LP post-process) and MOEMA (the memetic algorithm enhanced with the LP post-process). For each instance, it shows the average hypervolume (HV) and ϵ-indicator ($I_{\epsilon+}$) values across the 30 sets of non-dominated solutions obtained by each algorithm, with standard deviation values between brackets. Additionally, a statistical test is run for each instance to assess whether differences between both methods are significant. Since the output of the LP optimisation is dependent on the output of EA+TS+HER, these are paired samples. However, they do not pass a Shapiro-Wilk normality test, so Wilcoxon Signed-Ranks tests are used to compare both methods across all instances. As a result, for each instance and metric we highlight in bold those values that are significantly better than their counterparts according to these tests.

Table 3
Comparison, in terms of HV and the $I_{\epsilon+}$ values between the enhanced memetic algorithm MOEMA and the same algorithm without the LP post-processing, EA+TS+HER. In bold, those values which are significantly better according to statistical tests.

	HV		$I_{\epsilon+}$	
Inst.	EA+TS+HER	MOEMA	EA+TS+HER	MOEMA
FT10	0.818 (0.024)	**0.866** (0.027)	0.197 (0.031)	**0.149** (0.033)
FT20	0.794 (0.071)	**0.849** (0.068)	0.210 (0.069)	**0.159** (0.061)
ABZ7	0.791 (0.043)	**0.816** (0.045)	0.204 (0.035)	**0.177** (0.036)
ABZ8	0.790 (0.057)	**0.824** (0.063)	0.172 (0.048)	**0.138** (0.051)
ABZ9	0.735 (0.071)	**0.748** (0.072)	0.170 (0.048)	**0.159** (0.046)
La21	0.826 (0.052)	**0.843** (0.050)	0.184 (0.045)	**0.167** (0.043)
La24	0.843 (0.045)	**0.866** (0.052)	0.176 (0.047)	**0.156** (0.053)
La25	0.769 (0.042)	**0.798** (0.046)	0.251 (0.049)	**0.220** (0.055)
La27	0.808 (0.056)	**0.826** (0.056)	0.164 (0.053)	**0.148** (0.051)
La29	0.767 (0.071)	**0.790** (0.074)	0.213 (0.069)	**0.190** (0.070)
La38	0.858 (0.055)	**0.890** (0.060)	0.149 (0.036)	**0.116** (0.041)
La40	0.899 (0.057)	**0.931** (0.053)	0.098 (0.045)	**0.069** (0.040)

We can see that incorporating the LP post-process clearly improves the performance of the algorithm both in terms of HV (3.6%) and $I_{\epsilon+}$ (16.0%) on average. In fact, MOEMA outperforms EA+TS+HER on every instance, with statistically significant differences in all cases for both metrics. With respect to CPU times, they depend strongly on the instance size but also on the ratio between number of jobs and number of machines (there is a high correlation between the CPU times and the value of the equation $(|J|/|M|) + |O|$ where O denotes the set of all operations). Additionally, the runtime increases only 2.8% on average when the LP post-process is incorporated.

Regarding the solutions obtained with the commercial solver for the LP formulation starting from a set of non-dominated random feasible solutions, in 10 runs the solver reaches the optimal solution with respect to non-processing energy maintaining the job-processing order on every machine in less than 1.5 seconds per solution. However, the obtained results are not competitive with those of MOEMA. Even if the energy reduction obtained by the LP solver is very significant (25% on average), the obtained solutions are dominated by all solutions of all fronts obtained in all runs of MOEMA. In fact, almost any solution given by MOEMA dominates all the solutions obtained by the LP solver). The difference in favour of MOEMA is so blatant that it renders a table of detailed values for different metrics useless.

5.4. Comparison with other solving methods

To our knowledge, this is the first time the biobjective FJSP with makespan and non-processing energy minimisation has been considered. Hence, there is no state-of-the-art solving method from the literature to compare MOEMA with. When this is the case, some authors retort to comparing their solving method with a commercial solver (see for instance [5]). Alternatively, the results of a multiobjective method can be evaluated taking as reference the state-of-the-art mono-objective methods for each of the objective functions, as done for instance in [49]. Therefore, faced with the impossibility of straightforward comparisons with other methods, we adopt here these two approaches to further assess the performance of MOEMA.

First, an experiment has been conducted in order to compare our proposal with the commercial solver CPLEX applied to the original multiobjective MIP formulation provided in Section 3. We have used the ϵ-constraint method as explained in Section 3.5, finding the Ideal and Nadir points first and then applying the solver to find the Pareto front. However, when attempting to find the Ideal and Nadir points, the solver is not able to reach a feasible solution in 24 hours, except for the two smallest instances, FT10 and FT20. Even for these two instances, the quality of the obtained feasible solutions is very low regarding both objectives, so far that the point obtained in this time limit as “ideal”

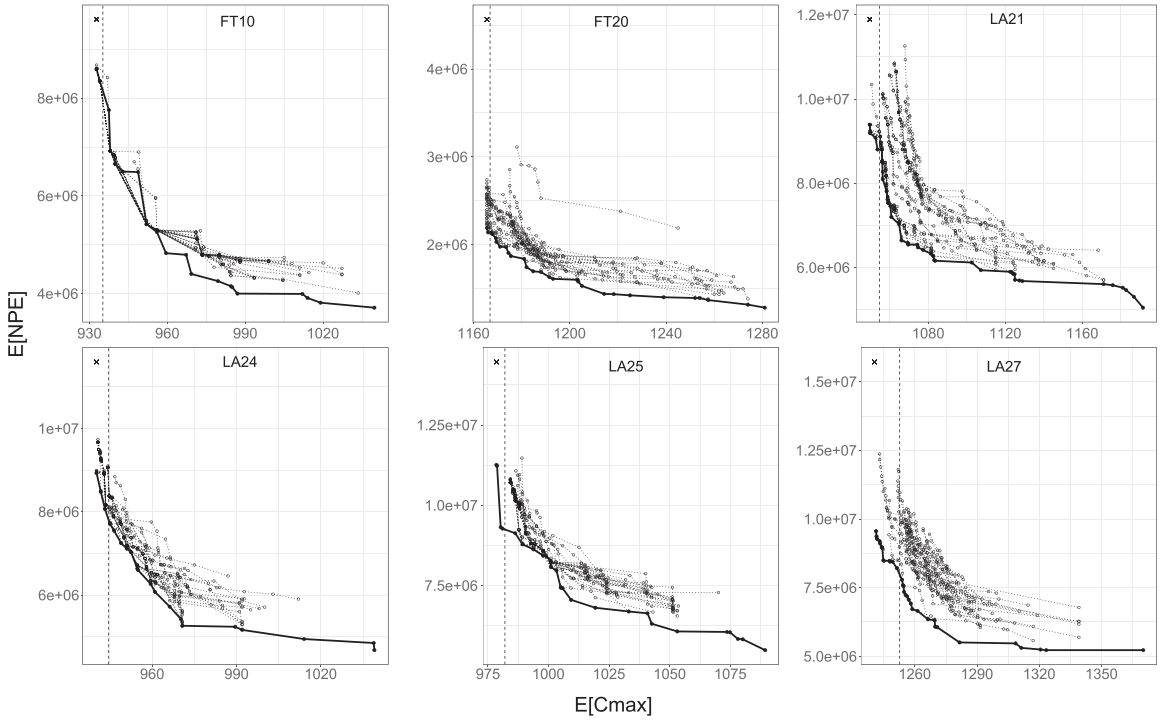

Fig. 7. Detailed performance of MOEMA and comparison with single-objective HTS from [15] for instances FT10, FT20, La21, 24, 25 and 27.

is dominated by the solutions found by MOEMA. This illustrates the difficulty of the selected instances and the need to tackle them using powerful metaheuristic search methods.

A second indirect comparison is with the state-of-the-art for FJSP with makespan minimisation. For the arithmetic introduced in Section 2.3 (in particular, for the approximated maximum and the ranking based on expected values), the best makespan results are obtained by a hybrid tabu search (HTS) method from [15].

Figures 7 and 8 show for each instance the solutions obtained by MOEMA and HTS in the objective space, where the x-axis corresponds to $E[\widehat{C}_{\max}]$ while the y-axis corresponds to $E[\widehat{\text{NPE}}]$. The 30 runs of MOEMA produce 30 sets of non-dominated solutions, represented as grey circles joined by dotted lines; a black bold line connects the approximation of the Pareto front that results from taking the non-dominated elements in the union of all sets of solutions obtained along the experimental study. Additionally, we can see the best and average $E[\widehat{C}_{\max}]$ (a cross and a vertical dotted line) obtained by HTS (the $E[\widehat{\text{NPE}}]$ value of the best solution from HTS has been calculated from the task ordering induced by the solution).

As could be expected, the solutions obtained by the single-objective method HTS are very good in terms of $E[\widehat{C}_{\max}]$ but they are pretty bad in terms of $E[\widehat{\text{NPE}}]$. Perhaps more surprisingly, the approximation of the Pareto front by MOEMA contains a solution with equal or better $E[\widehat{C}_{\max}]$ value than the solution from HTS in 9 out of the 12 instances, so there is always a solution from MOEMA that either dominates or is equal to the best solution found by HTS. Furthermore, in the remaining 3 instances, the $E[\widehat{\text{NPE}}]$ values of the best solutions produced by HTS are so poor that these solutions never dominate any of the solutions produced by MOEMA.

For every instance, Table 4 contains the average $E[\widehat{C}_{\max}]$ in the solutions obtained by HTS across all runs and the corresponding average value of the MOEMA solutions located at the extreme of the approximation of the Pareto front corresponding to this objective. In 6 of the 12 instances, MOEMA obtains a slightly better average $E[\widehat{C}_{\max}]$ while in the other 6 instances differences are below 0.4%. Furthermore a paired t-test indicates that there are not significant differences between both

Table 4
Average $E[\widehat{C}_{\max}]$ of solutions from HTS and MOEMA, with the best value in bold..

Inst.	Avg. $E[\widehat{C}_{\max}]$		Inst.	Avg. $E[\widehat{C}_{\max}]$	
	HTS	MOEMA		HTS	MOEMA
FT10	935.2	**934.8**	La29	**1181.2**	1183.5
FT20	**1166.8**	1167.4	La38	1213.6	**1206.2**
La21	**1054.5**	1056.5	La40	1233.7	**1233.0**
La24	944.6	**942.3**	ABZ7	676.1	**673.6**
La25	**982.0**	985.4	ABZ8	687.1	**681.3**
La27	**1252.6**	1254.4	ABZ9	**700.4**	701.1

methods as far as $E[\widehat{C}_{\max}]$ is concerned. However, as it is clearly seen in the figures, no such similarity between solutions from both methods exists when considering the energy objective. This is not surprising, since MOEMA is a multiobjective algorithm considering $E[\widehat{\text{NPE}}]$ while HTS only optimises $E[\widehat{C}_{\max}]$. What is perhaps less expected is that the multiobjective MOEMA is also competitive with HTS in terms of its single objective $E[\widehat{C}_{\max}]$.

6. Conclusions

In this paper we have addressed the job shop scheduling problem under uncertainty in task processing times with two conflicting objectives: reducing the energy consumption, measured as the fuzzy total non-processing energy $\widehat{\text{NPE}}$, and simultaneously reducing production times, measured as the fuzzy makespan $\widehat{C}_{\max}$. Being the first time this variant of the job shop problem is tackled in the literature, we have rigorously defined the problem, including a MIP model thereof. As solving method we have proposed a multiobjective enhanced memetic algorithm MOEMA concerned with exploiting all available knowledge about the problem. The basis of this memetic algorithm is the well-known NSGA-II template for a dominance-based genetic algorithm, endowed with problem-specific coding, decoding and recombination operators. Taking into account the different nature of the objective functions, the genetic algo-

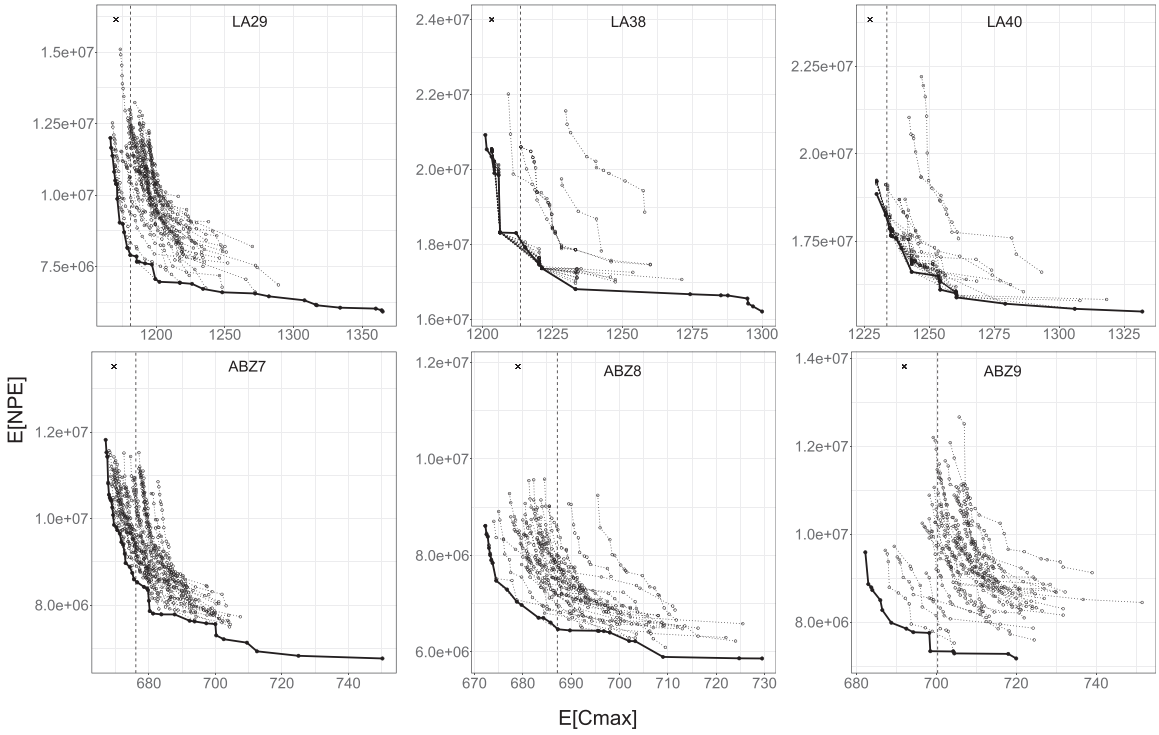

Fig. 8. Detailed performance ofMOEMA and comparison with single-objective HTS from [15] for instances La29, 38, 40, ABZ7, 8 and 9.

rithm incorporates two single-objective intensification procedures during the evolution. First, a tabu search method for makespan minimisation using a neighbourhood structure and speed-up procedures from the literature. Second, a heuristic for energy reduction, HER, first proposed in this paper, based on previous theoretical results regarding the fuzzy total non-processing energy. Both the TS and HER are applied to every individual in the population and a Lamarckian learning model is used to improve the quality of the successive generations of the multiobjective genetic algorithm. Finally, the resulting memetic algorithm is enhanced with a post-process of the resulting set of non-dominated solutions, where a commercial solver is applied to an LP formulation of every solution to obtain optimal non-processing energy values for the given job orders. We have presented a suite of experimental results that avail our proposal, showing a clear synergy effect among the different components of the memetic algorithm and the potential of the multiobjective method, by indirect comparisons with a commercial solver and the state-of-the-art in single-objective makespan optimisation. The problem instances as well as the approximations to the Pareto Front and detailed results have been made available, as a reference for future research, as electronic supplementary material to this work.

Declaration of Competing Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Declaration of Competing Interest

The authors declare no conflict of interest.

CRediT authorship contribution statement

Sezin Afsar: Conceptualization, Methodology, Software, Formal analysis, Investigation. **Juan José Palacios:** Conceptualization, Methodology, Formal analysis, Investigation, Visualization. **Jorge Puente:** Methodology, Software, Formal analysis, Investigation, Visualization. **Camino R. Vela:** Supervision, Writing – original draft, Conceptualization, Methodology. **Inés González-Rodríguez:** Conceptualization, Writing – review & editing, Methodology, Writing – original draft.

Acknowledgements

This research has been supported by the Spanish Government under research grants TIN2016-79190-R and PID2019-106263RB-I00, and by the Principality of Asturias Government under grant IDI/2018/000176.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.swevo.2021.101016

References

[1] M.L. Pinedo, Scheduling. Theory, Algorithms, and Systems, 5th, Springer, 2016. 10.1007/978-1-4614-2361-4
[2] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, 1979.
[3] M. Paolucci, D. Anguinolfi, F. Tonelli, Facing energy-aware scheduling: a multi-objective extension of a scheduling support system for improving energy efficiency in a moulding industry, Soft comput 21 (2017) 3687–3698, doi:10.1007/s00500-015-1987-8.
[4] D. Yuksel, M. Tasgetiren, L. Kandiller, L. Gao, An energy-efficient biobjective no-wait permutation flowshop scheduling problem to minimize total tardiness and total energy consumption, Computers & Industrial Engineering (2020), doi:10.1016/j.cie.2020.106431.
[5] H. Zhang, Y. Wu, R. Pan, G. Xu, Two-stage parallel speed-scaling machine scheduling under time-of-use tariffs, J Intell Manuf 32 (2021) 91–112, doi:10.1007/s10845-020-01561-6.
[6] S. Mansouri, E. Aktas, U. Besikci, Green scheduling of a two-machine flowshop: trade-off between makespan and energy consumption, Eur J Oper Res 248 (2016) 772–788, doi:10.1016/j.ejor.2015.08.064.
[7] H. Tian, X. Yuan, Y. Huang, An improved gravitational search algorithm for solving short-term economic/environmental hydrothermal scheduling, Soft comput 19 (2015) 2783–2797, doi:10.1007/s00500-014-1441-3.
[8] R. Zhang, R. Chiong, 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, J Clean Prod 112 (4) (2016) 3361–3375, doi:10.1016/j.jclepro.2015.09.097.
[9] Y. Liu, H. Dong, N. Lohse, S. Petrovic, N. Gindy, An investigation into minimising total energy consumption and total weighted tardiness in job shops, J Clean Prod 65 (2014) 87–96, doi:10.1016/j.jclepro.2013.07.060.
[10] W. Herroelen, R. Leus, Project scheduling under uncertainty: survey and research potentials, Eur J Oper Res 165 (2005) 289–306, doi:10.1016/j.ejor.2004.04.002.
[11] H. Aytung, M.A. Lawley, K. McKay, M. Shantha, R. Uzsoy, Executing production schedules in the face of uncertainties: a review and some future directions, Eur J Oper Res 161 (2005) 86–110, doi:10.1016/j.ejor.2003.08.027.
[12] D. Dubois, H. Fargier, P. Fortemps, Fuzzy scheduling: modelling flexible constraints vs. coping with incomplete knowledge, Eur J Oper Res 147 (2003) 231–252, doi:10.1016/S0377-2217(02)00558-1.
[13] S. Abdullah, M. Abdolrazzagh-Nezhad, Fuzzy job-shop scheduling problems: a review, Inf Sci (Ny) 278 (2014) 380–407, doi:10.1016/j.ins.2014.03.060.
[14] J.J. Palacios, J. Puente, C.R. Vela, I. González-Rodríguez, Benchmarks for fuzzy job shop problems, Inf Sci (Ny) 329 (2016) 736–752, doi:10.1016/j.ins.2015.09.042.
[15] J.J. Palacios, J. Puente, I. González-Rodríguez, C.R. Vela, Hybrid tabu search for fuzzy job shop, in: J.M.F. Vicente, J.R.A. Sánchez, F.P. López, F.T. Moreo (Eds.), Natural and Artificial Models in Computation and Biology, Vol. 7930 of Lecture Notes in Computer Science, Springer, 2013, pp. 376–385. 10.1007/978-3-642-38637-4_39
[16] L.P. Cota, F.G. Guimarães, R.G. Ribeiro, I.R. Meneghini, F.B. de Oliveira, M.J. Souza, P. Siarry, An adaptive multi-objective algorithm based on decomposition and large neighborhood search for a green machine scheduling problem, Swarm Evol Comput 51 (2019) 100601, doi:10.1016/j.swevo.2019.100601.
[17] A. Oddi, R. Rasconi, M.A. González, Energy-aware multiple state machine scheduling for multiobjective optimization, in: R. Goebel, Y. Tanaka, W. Wahlster (Eds.), AI*IA 2018 - Advances in Artificial Intelligence, Vol. 11298 of Lecture Notes in Computer Science, Springer Nature, 2018, pp. 474–486. 10.1007/978-3-030-03840-3_35
[18] S. Wang, X. Wang, J. Yu, S. Ma, M. Liu, Bi-objective identical parallel machine scheduling to minimize total energy consumption and makespan, J Clean Prod 193 (2018) 424–440, doi:10.1016/j.jclepro.2018.05.056. 424e440 193 (2018)
[19] X. Wu, A. Che, A memetic differential evolution algorithm for energy-efficient parallel machine scheduling, Omega (Westport) 82 (2019) 155–165, doi:10.1016/j.omega.2018.01.001.
[20] K. Fang, N. Uhan, F. Zhao, J.W. Sutherland, A new approach to scheduling in manufacturing for power consumption and carbon footprint reduction, J. Manuf. Syst. 30 (2011) 234–240, doi:10.1016/j.jmsy.2011.08.004.
[21] H. Luo, B. Du, G.Q. Huang, H. Chen, X. Li, Hybrid flow shop scheduling considering machine electricity consumption cost, Int. J.Production Economics 146 (2013) 423–439, doi:10.1016/j.ijpe.2013.01.028.
[22] Y. Han, J. Li, H. Sang, Y. Liu, K. Gao, Q. Pan, Discrete evolutionary multi-objective optimization for energy-efficient blocking flow shop scheduling with setup time, Appl Soft Comput (2020), doi:10.1016/j.asoc.2020.106343.
[23] H. Öztop, M.F. Tasgetiren, L. Kandiller, D.T. Eliiyi, L. Gao, Ensemble of metaheuristics for energy-efficient hybrid flowshops: makespan versus total energy consumption, Swarm Evol Comput 54 (2020) 100660, doi:10.1016/j.swevo.2020.100660.
[24] G. Wang, L. Gao, X. Li, P. Li, M.F. Tasgetiren, Energy-efficient distributed permutation flow shop scheduling problem using a multi-objective whale swarm algorithm, Swarm Evol Comput 57 (2020) 100716, doi:10.1016/j.swevo.2020.100716.
[25] H. Wei, S. Li, H. Quan, D. Liu, S. Rao, C. Li, J. Hu, Unified multi-objective genetic algorithm for energy efficient job shop scheduling, IEEE Access 9 (2021) 54542–54557, doi:10.1109/ACCESS.2021.3070981.
[26] G. Gong, R. Chiong, Q. Deng, Q. Luo, A memetic algorithm for multi-objective distributed production scheduling: minimizing the makespan and total energy consumption, J Intell Manuf 31 (2020) 1443–1466, doi:10.1007/s10845-019-01521-9.
[27] I. González-Rodríguez, J. Puente, J.J. Palacios, C.R. Vela, Multi-objective evolutionary algorithm for solving energy-aware fuzzy job shop problems, Soft comput 24 (2020) 16291–16302, doi:10.1007/s00500-020-04940-6.
[28] M.A. González, A. Oddi, R. Rasconi, Multi-objective optimization in a job shop with energy costs through hybrid evolutionary techniques, in: Proceedings of the 27th International Conference on Automated Planning and Scheduling (ICAPS-2017), 2017, pp. 140–148.
[29] H. Ishibuchi, N. Tsukamoto, Y. Nojima, Evolutionary many-objective optimization: a short review, in: Genetic and Evolving Systems, 2008. GEFS 2008. 3rd International Workshop on, 2008, pp. 47–52. 10.1109/GEFS.2008.4484566
[30] S.-W. Lin, K.C. Ying, Optimization of makespan for no-wait flowshop scheduling problems using efficient matheuristics, Omega (Westport) 64 (2016) 115–125, doi:10.1016/j.omega.2015.12.002.
[31] L. Fanjul-Peyro, F. Perea, R. Ruiz, Models and matheuristics for the unrelated parallel machine scheduling problem with additional resources, Eur J Oper Res 260 (2017) 482–493, doi:10.1016/j.ejor.2017.01.002.
[32] P. Moscato, C. Cotta, An accelerated introduction to memetic algorithms, in: M. Gendreau, J.Y. Potvin (Eds.), Handbook of Metaheuristics, Springer, 2019, pp. 275–309. 10.1007/978-3-319-91086-4_9
[33] H. Ishibuchi, Y. Hitotsuyanagi, N. Tsukamoto, Y. Nojima, Use of heuristic local search for single-objective optimization in multiobjective memetic algorithms, in: Proceedings of Parallel Problem Solving from Nature (LNCS 5199), 2008, pp. 743–752. 10.1007/978-3-540-87700-4_74
[34] IBM, IBM CPLEX Optimizer, 2020, https://www.ibm.com/analytics/cplex-optimizer.
[35] M.T. Emmerich, A.H. Deutz, A tutorial on multiobjective optimization: fundamentals and evolutionary methods, Nat Comput 17 (3) (2018) 1567–7818, doi:10.1007/s11047-018-9685-y.
[36] D. Dubois, H. Prade (Eds.), Fundamentals of Fuzzy Sets, the Handbooks of Fuzzy Sets, Kluwer Academic Publishers, Boston/London/Dordrecht, 2000.
[37] S. Heilpern, The expected value of a fuzzy number, Fuzzy Sets Syst. 47 (1992) 81–86, doi:10.1016/0165-0114(92)90062-9.
[38] P. Fortemps, M. Roubens, Ranking and defuzzification methods based on area compensation, Fuzzy Sets Syst. 82 (1996) 319–330, doi:10.1016/0165-0114(95)00273-1.
[39] J.J. Palacios, C.R. Vela, I. González-Rodríguez, J. Puente, Schedule generation schemes for job shop problems with fuzziness, in: T. Schaub, G. Friedrich, B. O'Sullivan (Eds.), Proceedings of ECAI 2014, Vol. 263 of Frontiers in Artificial Intelligence and Applications, IOS Press, 2014, pp. 687–692. 10.3233/978-1-61499-419-0-687
[40] R. Graham, E. Lawler, J. Lenstra, A.R. Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey, Annals of Discrete Mathematics 4 (1979) 287–326, doi:10.1016/S0167-5060(08)70356-X.
[41] M. Sakawa, T. Mori, An efficient genetic algorithm for job-shop scheduling problems with fuzzy processing time and fuzzy duedate, Computers & Industrial Engineering 36 (1999) 325–341, doi:10.1016/S0360-8352(99)00135-7.
[42] Y. Haimes, On a bicriterion formulation of the problems of integrated system identification and system optimization, IEEE transactions on systems, man and cybernetics 1 (3) (1971) 296–297.
[43] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197, doi:10.1109/4235.996017.
[44] Q. Zhang, H. Manier, M.A. Manier, A genetic algorithm with tabu search procedure for flexible job shop scheduling with transportation constraints and bounded processing times, Computers & Operations Research 39 (2012) 1713–1723, doi:10.1016/j.cor.2011.10.007.
[45] J.J. Palacios, M.A. González, C.R. Vela, I. González-Rodríguez, J. Puente, Genetic tabu search for the fuzzy flexible job shop problem, Computers & Operations Research 54 (2015) 74–89, doi:10.1016/j.cor.2014.08.023.
[46] C. Bierwirth, A generalized permutation approach to jobshop scheduling with genetic algorithms, OR Spectrum 17 (1995) 87–92, doi:10.1007/BF01719250.
[47] C. Bierwirth, D.C. Mattfeld, H. Kopfer, On permutation representations for scheduling problems, in: PPSN IV: Proceedings of the 4th International Conference on Parallel Problem Solving from Nature, Springer-Verlag, London, UK, 1996, pp. 310–318.
[48] E.G. Talbi, Metaheuristics. From Design to Implementation, Wiley, 2009.
[49] J.J. Palacios, I. González-Rodríguez, C.R. Vela, J. Puente, Robust multiobjective optimisation for fuzzy job shop problems, Appl Soft Comput 56 (2017) 604–616, doi:10.1016/j.asoc.2016.07.004.
[50] F. Glover, Tabu search–Part I, ORSA Journal on Computing 1 (3) (1989) 190–206, doi:10.1287/ijoc.1.3.190.
[51] F. Glover, Tabu search–part II, ORSA Journal on Computing 2 (1) (1990) 4–32, doi:10.1287/ijoc.2.1.4.
[52] I. González-Rodríguez, C.R. Vela, J. Puente, R. Varela, A new local search for the job shop problem with uncertain durations, in: Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling (ICAPS-2008), AAAI Press, Sidney, 2008, pp. 124–131.
[53] I. González-Rodríguez, C.R. Vela, A. Hernández-Arauzo, J. Puente, Improved local search for job shop scheduling with uncertain durations, in: Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS-2009), AAAI Press, Thesaloniki, 2009, pp. 154–161.
[54] C. Cotta, L. Mathieson, P. Moscato, Memetic algorithms, in: R. Martí, P. Panos, M.G.C. Resende (Eds.), Handbook of Heuristics, Springer International Publishing, 2016, pp. 1–32. 10.1007/978-3-319-07153-4_29-1
[55] C.R. Vela, S. Afsar, J.J. Palacios, I. González-Rodríguez, J. Puente, Evolutionary tabu search for flexible due-date satisfaction in fuzzy job shop scheduling, Computers & Operations Research 119 (2020) 104931, doi:10.1016/j.cor.2020.104931.
[56] P. Fattahi, M.S. Mehrabad, F. Jolai, Mathematical modeling and heuristic approaches to flexible job shop scheduling problems, Journal of Intelligent Manfacturing 18 (2007) 331–342, doi:10.1007/s10845-007-0026-8.
[57] W.T. Lunardi, E.G. Birgin, P. Laborie, D.P. Ronconi, H. Voos, Mixed integer linear programming and constraint programming models for the online printing shop scheduling problem, Computers and Operations Research 123 (2020), doi:10.1016/j.cor.2020.105020.
[58] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, V.G.d. Fonseca, Performance assessment of multiobjective optimizers: an analysis and review, IEEE Trans. Evol. Comput. 7 (2) (2003) 117–132, doi:10.1109/TEVC.2003.810758.
[59] V. Grunert da Fonseca, C.M. Fonseca, A.O. Hall, Inferential performance assessment of stochastic optimisers and the attainment function, in: E. Zitzler, L. Thiele, K. Deb, C.A. Coello Coello, D. Corne (Eds.), Evolutionary Multi-Criterion Optimization, Springer, 2001, pp. 213–225. 10.1007/3-540-44719-9_15
[60] M. Lopez-Ibañez, L. Paquete, T. Stützle, Exploratory analysis of stochastic local search algorithms in biobjective optimization, in: Experimental Methods for the Analysis of Optimization Algorithms, Springer, 2010, pp. 209–222. Ch. 9, 10.1007/978-3-642-02538-9_9
[61] M.D. Amico, M. Trubian, Applying tabu search to the job-shop scheduling problem, Annals of Operational Research 41 (1993) 231–252.