

Multi-objective evolutionary algorithm for solving energy-aware fuzzy job shop problems

Inés González-Rodríguez · Jorge Puente · Juan José Palacios · Camino R. Vela

Received: date / Accepted: date

Abstract A growing concern about the environmental impact of manufacturing processes and in particular, the associated energy consumption, has recently driven some researchers within the scheduling community to consider energy costs in addition to more traditional performance-related measures, such as satisfaction of due-date commitments. Recent research is also devoted to narrowing the gap between real-world applications and academic problems by handling uncertainty in some input data. In this paper we address the job shop scheduling problem, a well-known hard problem with many applications, using fuzzy sets to model uncertainty in processing times and with the target of finding solutions that perform well with respect to both

due-date fulfilment and energy efficiency. The resulting multiobjective problem is solved using an evolutionary algorithm based on the NSGA-II procedure, where the decoding operator incorporates a new heuristic procedure in order to improve the solutions' energy consumption. This heuristic is based on a theoretical analysis of the changes in energy consumption when a solution is subject to slight changes, referred to as local right shifts. The experimental results support the theoretical study and show the potential of the proposal.

Keywords job shop scheduling · fuzzy durations · multiobjective · due dates · energy efficiency · genetic algorithm

This research has been financially supported by the Spanish Government under research grant TIN2016-79190-R and by the Principality of Asturias Government under grant IDI/2018/000176.

Inés González-Rodríguez
Dep. of Maths, Stats and Computing
University of Cantabria
Santander, Spain
E-mail: gonzalezri@unican.es

Jorge Puente
Dep. of Computer Science
University of Oviedo
Gijón, Spain
E-mail: puente@uniovi.es

Juan José Palacios
Dep. of Computer Science
University of Oviedo
Gijón, Spain
E-mail: palaciosjuan@uniovi.es

Camino R. Vela
Dep. of Computer Science
University of Oviedo
Gijón, Spain
E-mail: crvela@uniovi.es

1 Introduction

Scheduling problems appear in a growing number of domains, including 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 (e.g. wafer fabs in the semiconductor industry often function as job shops) (Jain and Meeran 1999; Pinedo 2016). It also poses a challenge to the research community due to its complexity (Garey and Johnson 1979).

The most common objective in the literature consists in finding solutions minimising the execution time span of the project, known as makespan. However, due-date satisfaction has also occupied researchers (Koullamas 1994) and its interest is growing in recent years as on-time fulfilment gains importance in modern pull-oriented supply chain systems, and keeping job due dates is a prerequisite for serving customers within the promised delivery time and avoiding out-of-stocks costs

(González et al. 2012; González and Vela 2015; Kuhpfahl and Bierwirth 2016). The use of due-date satisfaction measures such as the total weighted tardiness (TWT) in production scheduling helps the companies in increasing their logistic service and creating competitive advantage.

More recently, the growing environmental awareness translates into concerns about the carbon footprint of every industrial process. In this new context, production scheduling can play a key role in reducing the energy consumption of factories and industrial plants or the use of fossile fuels in hybrid production systems by incorporating energy efficiency to the range of objectives under consideration (Tian et al. 2015; Zhang and Chiong 2016; Paolucci et al. 2017). 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 the 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 (Liu et al. 2014).

Mainstream approaches usually assume that scheduling problems are deterministic; in particular, it is assumed 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 design variables are subject to perturbations or changes, causing optimal solutions to the original problem to be of little or no use in practice (Herroelen and Leus 2005; Aytung et al. 2005). 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 (Dubois et al. 2003). 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 (Abdullah and Abdolrazzagh-Nezhad 2014; Palacios et al. 2016).

In the following, we consider a multiobjective fuzzy job shop problem with two different (possibly conflicting) objectives: the total weighted tardiness, related to due-date satisfaction, and the total non-processing energy, related to energy costs. The simultaneous minimisation of both objectives has been the goal of several recent contributions (Firas and Denis 2015; González et al. 2017; Liu et al. 2016; Zhang and Chiong 2016). Most of these propose successful solving methods based on the well-known multiobjective genetic algorithm template NSGA-II (Deb 2014). However, ideal conditions are assumed and no uncertainty is taken into account. We thus propose to advance in this line of research by incorporating uncertainty in processing times. To this

end, we first extend the existing definition of the total non-processing energy, to the fuzzy context. Then, we propose a new powerful decoding algorithm to be used in the NSGA-II specific for this problem. The proposal is based on a theoretical study of the behaviour of schedules in terms of fuzzy total non-processing energy consumption when they are subject to slight changes in activity starting times.

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, with a novel definition of the non-processing energy function adapted to the fuzzy context. Section 4 presents a theoretical analysis of how tasks in a given schedule can be right-shifted without increasing the non-processing energy. This provides the basis for the solving method proposed in Section 5, a multiobjective evolutionary algorithm that incorporates a novel heuristic strategy to reduce the non-processing energy in a schedule without affecting the total weighted tardiness. Finally, Section 6 reports experimental results to empirically evaluate the contribution of the proposed heuristic and provides results on a new set of benchmark instances for future reference. Finally, some remarks and conclusions are drawn in Section 7.

2 Preliminaries

2.1 Multi-objective optimisation

A *multi-objective optimisation problem* is an optimisation problem considering multiple objective functions f_1, \dots, f_G , $G \geq 2$. Besides the usual *decision variable space* X of feasible solutions there exists an additional space called the *objective space* Z , so every solution $\mathbf{x} \in X$ is associated to an objective vector $\mathbf{z} = \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_G(\mathbf{x})) \in Z$. It is usually understood that $Z \subseteq \mathbb{R}^G$, that is, f_k associates to each decision variable \mathbf{x} a value $f_k(\mathbf{x}) \in \mathbb{R}$. It suffices however that $f_k(\mathbf{x})$ 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 by $\mathbf{x} \leq_k \mathbf{y}$ if and only if $f_k(\mathbf{x}) \leq_L f_k(\mathbf{y})$. Given this mapping $\mathbf{f} : X \rightarrow Z$, solutions in X may be compared in terms of some ordering relation in the objective space Z , most commonly, a Pareto order.

In a minimisation problem, the objective is to minimise $\mathbf{f}(\mathbf{x})$ subject to $\mathbf{x} \in X$. Let $\mathbf{x}, \mathbf{y} \in X$ be two feasible solutions; in a minimisation context, \mathbf{x} is said to *dominate* \mathbf{y} , denoted $\mathbf{x} \prec \mathbf{y}$, iff $\forall k = 1, \dots, G, f_k(\mathbf{x}) \leq f_k(\mathbf{y})$ and there exists at least one objective k^* , $1 \leq k^* \leq G$, such that $f_{k^*}(\mathbf{x}) < f_{k^*}(\mathbf{y})$. In other words, \mathbf{x} is no worse than \mathbf{y} in all objectives and is strictly better

in at least one of them. We also say that \mathbf{x} is *preferred* to \mathbf{y} or *non-dominated* by \mathbf{y} and that \mathbf{y} is *dominated* by \mathbf{x} . A solution $\mathbf{x}^* \in X$ is *Pareto-optimal* if it is non-dominated by any other solution, that is, there exists no other solution $\mathbf{x} \in X$ such that $\mathbf{x} \prec \mathbf{x}^*$. 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*, or *JSP* in short, consists in scheduling a set of jobs $\{J_1, \dots, J_n\}$ on a set of physical resources or machines $\{M_1, \dots, M_m\}$, subject to a set of constraints. There are *precedence constraints*, so each job J_j , $j = 1, \dots, n$, consists of $m_j \leq m$ tasks $(o(j, 1), \dots, o(j, m_j))$ to be sequentially scheduled. There are also *resource constraints*, whereby each task $o(j, l)$ requires the uninterrupted and exclusive use of a machine $\nu_{o(j,l)} \in M$ for its whole processing time $p_{o(j,l)}$. We assume w.l.o.g. that tasks are indexed from 1 to $N = \sum_{j=1}^n m_j$, so we can refer to a task $o(j, l)$ by its index $o = \sum_{i=1}^{j-1} m_i + l$ and simply write ν_o, p_o to refer to its machine and processing time. The set of all tasks is denoted $O = \{0, \dots, N\}$, where 0 is an initial dummy task with zero processing time that precedes the first task of each job (i.e. $o(j, 0) = 0, \forall j = 1, \dots, n$).

Additionally, each job J_j has a due date d_j , that is, a time by which it is desirable that the job be completed and a weight w_j representing its relevance. Finally, in an energy-aware context, we follow (Liu et al. 2014) and assume that the energy consumption per time unit of a machine M_k while it is idle is given by its idle power level P_k^{idle} .

A solution to this problem is a *schedule*, i.e. an allocation of starting times for each task, which, besides being *feasible* (in the sense that all precedence and resource constraints hold), is *optimal* according to some criteria, in our case, reducing due-date violation and energy consumption.

2.3 Fuzzy durations

In real-life applications, the exact processing time of a task is rarely known in advance. It is reasonable however to have some (uncertain) knowledge about the duration, possibly based on previous experience. The crudest representation of such uncertain knowledge would be a human-originated confidence interval and, 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*

or *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:

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

Triangular fuzzy numbers (or, more generally, fuzzy intervals) are widely used in scheduling as a model for uncertain processing times (Abdullah and Abdolrazzaghe-Nezhad 2014; Dubois et al. 2003; Palacios et al. 2016).

The job shop problem requires four operations on TFNs: addition, subtraction, 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:

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

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

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

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:

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

Besides its extended use, several arguments can be given in favour of this approximation (cf. (Palacios et al. 2016)).

The membership function $\mu_{\hat{a}}$ of a fuzzy time \hat{a} may be interpreted as a possibility distribution on the real numbers (Dubois and Prade 2000), 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} (Heilpern 1992):

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

The expected value is linear and it induces a total ordering \leq_E in the set of TFNs (Fortemps and Roubens 1996), so for any two TFNs \widehat{a}, \widehat{b} , $\widehat{a} \leq_E \widehat{b}$ if and only if $E[\widehat{a}] \leq E[\widehat{b}]$. Clearly, if $\forall i \in 1, 2, 3, a^i \leq b^i$, then $\widehat{a} \leq_E \widehat{b}$.

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, respects due dates as much as possible. Hence, we consider two (perhaps conflicting) objectives: minimise due-date violation and minimise energy-consumption.

A schedule \mathbf{s} can be determined by a partial task processing order on all machines, which may be represented by a linear processing order σ . Such schedule (starting and, consequently, completion times of all tasks) may be easily computed based on σ using a semi-active schedule builder (Palacios et al. 2014). For every task $o \neq 0$ with processing time \widehat{p}_o , let $PM_o(\sigma)$ denote the task preceding o in the machine sequence provided by σ and let PJ_o denote its predecessor in the job sequence (for $o = o(j, l)$, $PJ_o = o(j, l - 1)$). The earliest feasible starting time of o , $\widehat{s}_o(\sigma)$, and corresponding completion time $\widehat{c}_o(\sigma)$ are TFNs given by:

$$\widehat{s}_o(\sigma) = \max(\widehat{c}_{PJ_o}, \widehat{c}_{PM_o(\sigma)}) \quad (7)$$

$$\widehat{c}_o(\sigma) = \widehat{s}_o(\sigma) + \widehat{p}_o \quad (8)$$

with $\widehat{s}_o(\sigma) = \widehat{c}_o(\sigma) = (0, 0, 0)$ if $o = 0$.

The completion time of each job J_j in the resulting schedule \mathbf{s} is the completion time of the last task in that job, given by:

$$\widehat{C}_j(\mathbf{s}) = \widehat{c}_{o(j, m_j)}(\sigma). \quad (9)$$

The resulting schedule \mathbf{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 the task processing ordering σ from which the schedule derives is deterministic; there is no uncertainty regarding the order in which tasks are to be processed.

3.1 Total weighted tardiness

To measure the performance of a schedule \mathbf{s} in terms of due-date violation, it is possible to compute the *tardiness* of each job J_j w.r.t its due date d_j , denoted $\widehat{T}_j(\mathbf{s}, d_j)$, measuring the extra time taken to complete the job after its due date, that is:

$$\widehat{T}_j(\mathbf{s}, d_j) = \max\{\widehat{0}, \widehat{C}_j(\mathbf{s}) - \widehat{d}_j\}. \quad (10)$$

The *total weighted tardiness* for the schedule across all jobs is then defined as:

$$\widehat{TWT}(\mathbf{s}) = \sum_{j=1}^n w_j \widehat{T}_j(\mathbf{s}, d_j). \quad (11)$$

3.2 Energy efficiency

Regarding energy efficiency, we adopt an energy consumption model from Liu et al. (2014), also used in González et al. (2017). 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 non-processing energy (NPE), i.e. the energy consumed when machines are idle (i.e. on, but not processing any job).

When processing times are deterministic, the NPE depends on the starting and completion times of the first and last tasks to be processed in each machine respectively (cf. (González et al. 2017)). In contrast, when durations are fuzzy, different non-processing energy consumptions may be obtained depending not only on which are the first and last tasks to be processed in each machine but on the order in which tasks are to be processed on each machine. A trivial extension of the original expression for the NPE to the fuzzy framework is therefore not possible. Instead, we propose a new definition for the fuzzy case in the following.

Let n_k be the number of tasks scheduled on machine M_k and let $\sigma_k(\mathbf{s}) = (0, \sigma(1, k), \dots, \sigma(n_k, k))$ be the processing order of tasks in machine k for a schedule \mathbf{s} . The *idle time* between the q -th and the $(q+1)$ -th tasks processed on M_k according to $\sigma_k(\mathbf{s})$ is a fuzzy quantity $\widehat{I}_k(\mathbf{s}, q, q+1)$ measuring the gap between the completion of task $\sigma(q, k)$ and the start of task $\sigma(q+1, k)$. Considering that in a feasible schedule the minimum possible gap has to be zero:

$$\widehat{I}_k(\mathbf{s}, q, q+1) = \max\{\widehat{0}, \widehat{s}_{\sigma(q+1, k)} - \widehat{c}_{\sigma(q, k)}\}. \quad (12)$$

Notice that for the schedule \mathbf{s} to be feasible, the two consecutive tasks $\sigma(q, k)$ and $\sigma(q+1, k)$ in machine M_k cannot overlap. Given the fuzzy arithmetic introduced in Section 2.3, this means $c_{\sigma(q, k)}^i \leq s_{\sigma(q+1, k)}^i$ for $i = 1, 2, 3$. Therefore, we can express the fuzzy idle time $\widehat{I}_k(\mathbf{s}, q, q+1)$ component-wise as follows:

$$\begin{aligned} \widehat{I}_k^1(\mathbf{s}, q, q+1) &= \max(0, s_{\sigma(q+1, k)}^1 - c_{\sigma(q, k)}^3), \\ \widehat{I}_k^2(\mathbf{s}, q, q+1) &= s_{\sigma(q+1, k)}^2 - c_{\sigma(q, k)}^2, \\ \widehat{I}_k^3(\mathbf{s}, q, q+1) &= s_{\sigma(q+1, k)}^3 - c_{\sigma(q, k)}^1. \end{aligned} \quad (13)$$

Clearly, the total idle time for a machine M_k is the sum of all the idle times between every two consecutive tasks on that machine and the *total non-processing energy* is a TFN defined as:

$$\widehat{NPE}(\mathbf{s}) = \sum_{k=1}^m \left(P_k^{idle} \sum_{q=1}^{n_k-1} \widehat{I}_k(\mathbf{s}, q, q+1) \right) \quad (14)$$

Let us illustrate with an example how, unlike the crisp case, the NPE value is heavily dependent on the task processing order and the uncertainty present in the starting and completion times of every task scheduled in the machine. Consider a problem with two jobs J_1 , J_2 and only one task per job, $o(1,1)$ and $o(2,1)$, both requiring the same machine and with processing times $\widehat{p}_{o(1,1)} = (3, 3, 3)$ and $\widehat{p}_{o(2,1)} = (1, 4, 8)$. Independently of the order in which we schedule the tasks, the machine will start processing tasks at $(0,0,0)$ and it will finish at $(4,7,11)$. However, if task $o(1,1)$ is scheduled before task $o(2,1)$, then:

$$\begin{aligned} \widehat{s}_{o(1,1)} &= (0, 0, 0), \\ \widehat{c}_{o(1,1)} &= \widehat{s}_{o(2,1)} = (3, 3, 3), \text{ and} \\ \widehat{c}_{o(2,1)} &= (4, 7, 11). \end{aligned}$$

and therefore

$$\begin{aligned} \widehat{NPE}(\mathbf{s}) &= P^{idle} \max\{(0, 0, 0), \widehat{s}_{o(2,1)} - \widehat{c}_{o(1,1)}\} \\ &= P^{idle} (0, 0, 0) = (0, 0, 0). \end{aligned}$$

On the other hand, if task $o(2,1)$ is scheduled first, then:

$$\begin{aligned} \widehat{s}_{o(2,1)} &= (0, 0, 0), \\ \widehat{c}_{o(2,1)} &= \widehat{s}_{o(1,1)} = (1, 4, 8), \text{ and} \\ \widehat{c}_{o(1,1)} &= (4, 7, 11). \end{aligned}$$

and therefore

$$\begin{aligned} \widehat{NPE}(\mathbf{s}) &= P^{idle} \max\{(0, 0, 0), \widehat{s}_{o(1,1)} - \widehat{c}_{o(2,1)}\} \\ &= P^{idle} (0, 0, 7) = (0, 0, 7P^{idle}). \end{aligned}$$

In the first case, there is no uncertainty regarding the completion time of the first task and, consequently, the starting time of the next one, so the gap can be exactly known (in the example, it is zero). On the contrary, in the second case there is uncertainty regarding when the first task will finish and also about the starting time of its successor. Therefore, uncertainty is present in the idle time between both tasks.

For the sake of simplicity, when no possible confusion is possible regarding the processing order σ or the schedule s , notation may be simplified by writing \widehat{s}_o , \widehat{C}_j , \widehat{T}_j and so on.

3.3 Resulting problem

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

$$\left(\widehat{TWT}(\mathbf{s}), \widehat{NPE}(\mathbf{s}) \right).$$

The problem can be denoted $J|\widehat{p}_o| \left(\widehat{TWT}, \widehat{NPE} \right)$ according to the usual $\alpha|\beta|\gamma$ notation introduced in Graham et al. (1979).

Notice that each of the objective values $\widehat{TWT}(\mathbf{s})$ and $\widehat{NPE}(\mathbf{s})$ belongs to the set of TFNs, which is totally ordered under the relation \leq_E , as explained in Section 2.3. Even if the 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.

4 Heuristic reduction of energy consumption

A good knowledge and understanding of subspaces of schedules is of key importance when designing search methods, their properties and how to obtain schedules pertaining to a particular subspace, since it allows to reduce the search space without risking the possibility of finding good solutions. Of particular interest is the subspace of *semi-active schedules* (Pinedo 2016). These are feasible schedules such that no task may start earlier without changing the processing order on any of the machines. It is common practice 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*, that is, any measure that is non-decreasing w.r.t. job completion times. In particular, the TWT is a regular performance measure. In consequence, to minimise TWT it suffices to consider semi-active schedules. This is still the case for semi-active fuzzy schedules (Palacios et al. 2014), which can be obtained from a task processing order as explained in Section 3.

On the contrary, the NPE (both the deterministic and fuzzy version) is a non-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. In (González et al. 2017) it is proposed to use a post-optimisation procedure to improve the energy-consumption of semi-active schedules in a deterministic setting. This procedure is not directly applicable to the fuzzy problem, but it inspires the following study on

how idle times and the associated \widehat{NPE} value can be reduced in a fuzzy schedule. To this end, we introduce the concept of local right shift, intuitively, a change consisting in “moving a task block to the right on the Gantt chart while preserving the task sequence”. It is a move inspired in the left-shift used by Sprecher et al. (1995) to formally define semi-active schedules for the deterministic RCSP, later adapted to the job shop with setup times in Artigues et al. (2005) and to the fuzzy job shop in Palacios et al. (2014).

Definition 1 Let \mathbf{s} be a feasible schedule. A *one-period local right shift* of a task o in \mathbf{s} is a move giving another feasible schedule \mathbf{s}' such that

$$\begin{aligned} \exists i \in \{1, 2, 3\} : s_o^i &= s_o^i + 1, s_o^j = s_o^j \forall j \neq i \\ s'_u &= s_u \forall u \in O - \{o\}. \end{aligned} \quad (15)$$

A *local right shift* of a task o is one or more successfully applied one-period local right shifts of task o .

Intuitively, a one-period right shift of o consists in delaying in one unit one of the components of o 's fuzzy starting time \widehat{s}_o without changing the other components of \widehat{s}_o nor changing the starting time of the remaining tasks. A local right shift of o consists in delaying at least one of (and possibly all) the components of \widehat{s}_o without changing the starting times of any of the other tasks nor changing their relative order in each machine. Notice that within a local right shift each intermediate derived schedule has to be feasible by definition.

Let us study how a one-period right shift of a task affects the \widehat{NPE} value. In the following, let \mathbf{s} be a feasible schedule and let $\sigma_k(\mathbf{s}) = (0, \sigma(1, k), \dots, \sigma(n_k, k))$ be the processing order of tasks in machine M_k according to \mathbf{s} . Let \mathbf{s}' be the schedule that results from a one-period right shift of task $\sigma(q, k)$ with $q \in \{1, \dots, n_k\}$ and let $i \in \{1, 2, 3\}$ be the component such that $s_{\sigma(q, k)}^i = s_{\sigma(q, k)}^i + 1$.

Lemma 1 *If $q \in \{2, \dots, n_k - 1\}$, that is, the task is neither the first nor the last one in its machine, then:*

- if $i = 1$ and $c_{\sigma(q-1, k)}^3 > s_{\sigma(q, k)}^1$, the expected value of \widehat{NPE} decreases:

$$E[\widehat{NPE}(\mathbf{s}')] = E[\widehat{NPE}(\mathbf{s})] - \frac{1}{4}P_k^{idle};$$

- if $i = 3$ and $c_{\sigma(q, k)}^3 \geq s_{\sigma(q+1, k)}^1$, the expected value of \widehat{NPE} increases:

$$E[\widehat{NPE}(\mathbf{s}')] = E[\widehat{NPE}(\mathbf{s})] + \frac{1}{4}P_k^{idle};$$

- in all other cases, $E[\widehat{NPE}]$ does not change.

Proof Notice that, from all the elements contributing to the total non-processing energy in (14), only $\widehat{I}(\mathbf{s}, q-1, q)$ and $\widehat{I}(\mathbf{s}, q, q+1)$, the idle gaps between $\sigma(q, k)$ and its preceding and succeeding tasks in the machine may change after the one-period right shift of $\sigma(q, k)$. Let us see, according to (13), how these idle gaps change depending on the component where the one-period right shift takes place.

Case $i = 1$.

$$I^3(\mathbf{s}', q, q+1) = I^3(\mathbf{s}, q, q+1) - 1,$$

while the other components of this gap do not change. Hence,

$$E[\widehat{I}(\mathbf{s}', q, q+1)] = E[\widehat{I}(\mathbf{s}, q, q+1)] - \frac{1}{4}.$$

If $c_{\sigma(q-1, k)}^3 > s_{\sigma(q, k)}^1$, then $c_{\sigma(q-1, k)}^3 \geq s_{\sigma(q, k)}^1$ and it follows that

$$I^1(\mathbf{s}', q-1, q) = I^1(\mathbf{s}, q-1, q) = 0$$

while the other components remain unchanged. Hence,

$$E[\widehat{I}(\mathbf{s}', q-1, q)] = E[\widehat{I}(\mathbf{s}, q-1, q)] = 0$$

and, thanks to the linearity of the expected value

$$E[\widehat{NPE}(\mathbf{s}')] = E[\widehat{NPE}(\mathbf{s})] - \frac{1}{4}P_k^{idle}.$$

If $c_{\sigma(q-1, k)}^3 = s_{\sigma(q, k)}^1$, then $c_{\sigma(q-1, k)}^3 < s_{\sigma(q, k)}^1$, so $I^1(\mathbf{s}, q-1, q) = 0$ and $I^1(\mathbf{s}', q-1, q) = 1$ while the other components remain unchanged. Hence,

$$E[\widehat{I}(\mathbf{s}', q-1, q)] = E[\widehat{I}(\mathbf{s}, q-1, q)] + \frac{1}{4}$$

and

$$E[\widehat{NPE}(\mathbf{s}')] = E[\widehat{NPE}(\mathbf{s})].$$

Finally, if $c_{\sigma(q-1, k)}^3 < s_{\sigma(q, k)}^1$, it is also the case that $c_{\sigma(q-1, k)}^3 < s_{\sigma(q, k)}^1$, so

$$I^1(\mathbf{s}', q-1, q) = I^1(\mathbf{s}, q-1, q) + 1,$$

while the other components do not change and, following the same reasoning as above,

$$E[\widehat{NPE}(\mathbf{s}')] = E[\widehat{NPE}(\mathbf{s})].$$

Case $i = 2$. In this case

$$I^2(\mathbf{s}', q, q+1) = I^2(\mathbf{s}, q, q+1) - 1$$

and

$$I^2(\mathbf{s}', q-1, q) = I^2(\mathbf{s}, q-1, q) + 1,$$

while the other components do not change, so

$$E[\widehat{NPE}(\mathbf{s}')] = E[\widehat{NPE}(\mathbf{s})].$$

Case $i = 3$.

$$I^3(\mathbf{s}', q-1, q) = I^3(\mathbf{s}, q-1, q) + 1,$$

while the other components of this gap do not change.

If $c_{\sigma(q,k)}^3 < s_{\sigma(q+1,k)}^1$, we have

$$I^1(\mathbf{s}', q, q+1) = I^1(\mathbf{s}, q, q+1) - 1$$

and the other components of the gap do not change.

Thus

$$E[\widehat{NPE}(\mathbf{s}')] = E[\widehat{NPE}(\mathbf{s})].$$

If $c_{\sigma(q,k)}^3 \geq s_{\sigma(q+1,k)}^1$, we have

$$I^1(\mathbf{s}, q, q+1) = I^1(\mathbf{s}', q, q+1) = 0$$

and its other components do not change, so

$$E[\widehat{NPE}(\mathbf{s}')] = E[\widehat{NPE}(\mathbf{s})] + \frac{1}{4}P_k^{idle}.$$

□

Lemma 2 A 1-period right shift of the first task on a machine ($q = 1$) always decreases the expected NPE

$$E[\widehat{NPE}(\mathbf{s}')] = E[\widehat{NPE}(\mathbf{s})] - \frac{1}{4}P_k^{idle}$$

except when $i = 3$ and $c_{\sigma(1,k)}^3 \geq s_{\sigma(2,k)}^1$, in which case it remains unchanged.

A 1-period right shift of the last task ($q = n_k$) always increases the expected NPE

$$E[\widehat{NPE}(\mathbf{s}')] = E[\widehat{NPE}(\mathbf{s})] + \frac{1}{4}P_k^{idle}$$

except when $i = 1$ and $c_{\sigma(n_k-1,k)}^3 > s_{\sigma(n_k,k)}^1$, in which case it remains unchanged.

Proof For the first task, $q = 1$, it suffices to realise that only $\widehat{I}(\mathbf{s}, q, q+1)$ changes in the NPE expression (14) and proceed as in the proof of Lemma 1.

For the last task, $q = n_k$, $\widehat{I}(\mathbf{s}, q-1, q)$ is the only idle gap that changes. Reasoning as in Lemma 1, we obtain the result. □

The above indicates which is the largest local right shift of a task that does not increase $E[\widehat{NPE}]$ and is not infeasible in the sense that it does not produce overlaps with the successor in the machine or the job.

Proposition 1 Let $SJ_{\sigma(q,k)}$ denote the successor of task $\sigma(q, k)$ in its job and let

$$lsM_{\sigma(q,k)}^i = s_{\sigma(q+1,k)}^i - p_{\sigma(q,k)}^i, i = 1, 2, 3 \quad (16)$$

be the i th component of the latest possible starting time of $\sigma(q, k)$ satisfying machine feasibility and

$$lsJ_{\sigma(q,k)}^i = s_{SJ_{\sigma(q,k)}}^i - p_{\sigma(q,k)}^i, i = 1, 2, 3 \quad (17)$$

be the i th component of the latest possible starting time of $\sigma(q, k)$ satisfying job precedence constraints. Then, the latest possible starting time for $\sigma(q, k)$ obtained with a local right-shift of itself that is non-increasing in $E[\widehat{NPE}]$ is given by the following:

– For $q = 1$:

$$\begin{aligned} s_{\sigma(1,k)}^{i3} &= \min\{lsM_{\sigma(1,k)}^3, lsJ_{\sigma(1,k)}^3\} \\ s_{\sigma(1,k)}^{i2} &= \min\{s_{\sigma(1,k)}^{i3}, lsM_{\sigma(1,k)}^2, lsJ_{\sigma(1,k)}^2\} \\ s_{\sigma(1,k)}^{i1} &= \min\{s_{\sigma(1,k)}^{i2}, lsM_{\sigma(1,k)}^1, lsJ_{\sigma(1,k)}^1\} \end{aligned} \quad (18)$$

– For $q \in \{2, \dots, n_k - 1\}$:

$$\begin{aligned} s_{\sigma(q,k)}^{i3} &= \min\{\max\{s_{\sigma(q,k)}^3, s_{\sigma(q+1,k)}^1 - p_{\sigma(q,k)}^3\}, \\ &\quad lsJ_{\sigma(q,k)}^3\} \\ s_{\sigma(q,k)}^{i2} &= \min\{s_{\sigma(q,k)}^{i3}, lsM_{\sigma(q,k)}^2, lsJ_{\sigma(q,k)}^2\} \\ s_{\sigma(q,k)}^{i1} &= \min\{s_{\sigma(q,k)}^{i2}, lsM_{\sigma(q,k)}^1, lsJ_{\sigma(q,k)}^1\} \end{aligned} \quad (19)$$

– For $q = n_k$:

$$\begin{aligned} s_{\sigma(n_k,k)}^{i3} &= s_{\sigma(n_k,k)}^3 \\ s_{\sigma(n_k,k)}^{i2} &= s_{\sigma(n_k,k)}^2 \\ s_{\sigma(n_k,k)}^{i1} &= \max\{s_{\sigma(n_k,k)}^1, \min\{c_{\sigma(n_k-1,k)}^3, s_{\sigma(n_k,k)}^2\}\} \end{aligned} \quad (20)$$

Proof For a local right shift of $\sigma(q, k)$ to be feasible, it must hold:

$$\begin{aligned} s_{\sigma(q,k)}^{ti} &\leq lsM_{\sigma(q,k)}^i, \forall i \in \{1, 2, 3\} \\ s_{\sigma(q,k)}^{ti} &\leq lsJ_{\sigma(q,k)}^i, \forall i \in \{1, 2, 3\} \\ s_{\sigma(q,k)}^{i1} &\leq s_{\sigma(q,k)}^{i2} \leq s_{\sigma(q,k)}^{i3} \end{aligned}$$

The first inequality preserves feasibility in the machine, the second one in the job and the third one ensures that the resulting starting time is a TFN. Applying Lemmas 1 and 2, we get the expressions for the latest possible starting time obtained with a local right-shift not increasing $E[\widehat{NPE}]$. □

5 Multi-objective genetic algorithm

The most extended and well-known method for solving multi-objective problems is the procedure known as non-dominated sorting genetic algorithm NSGA-II (Deb et al. 2002). Roughly speaking, this is a genetic algorithm (GA) emphasising the non-dominated solutions using an elitist principle together with an explicit diversity preserving mechanism.

The algorithm starts from a random initial population P_0 . This population is then evaluated and the algorithm iterates over a number of generations, keeping the set of non-dominated solutions found so far,

Require: An FJSP instance
Ensure: A schedule
 $Best \leftarrow \emptyset$
Generate a pool P_0 of random solutions.
Evaluate P_0
 $i \leftarrow 0$
while $i < \max_{\text{MOEA}}$ **do**
 $\text{Off}(P_i) \leftarrow \emptyset$
 while $|\text{Off}(P_i)| < |P_i|$ **do**
 $\{z_1, z_2\} \leftarrow$ Select two individuals from P_i
 $\{z'_1, z'_2\} \leftarrow$ Apply recombination to $\{z_1, z_2\}$
 $\text{Off}(P_i) \leftarrow \text{Off}(P_i) \cup \{z'_1, z'_2\}$
 Evaluate $\text{Off}(P_i)$;
 $P_{i+1} \leftarrow$ Apply crowded tournament to $P_i \cup \text{Off}(P_i)$;
 Update $Best$ with the non-dominated solutions in P_{i+1}
 if $Best$ remains unchanged **then**
 $i \leftarrow i + 1$;
 else
 $i \leftarrow 0$;
return $Best$

Fig. 1: Main steps of the multiobjective evolutionary algorithm

which approximates the Pareto set. At each iteration i , a new population $\text{Off}(P_i)$ is built from the current one P_i by applying the genetic operators of selection and recombination, and such that $|\text{Off}(P_i)| = |P_i|$. In this work, the selection is performed with random pairs and the GOX crossover and inversion mutation operators are used for recombination, which have proved to perform very well in multiobjective FJSP problems (Palacios et al. 2017). The next generation P_{i+1} is obtained from $\text{Off}(P_i) \cup P_i$ using the crowded tournament selection operator from (Deb et al. 2002) as follows. After initialising $P_{i+1} = \emptyset$, the non-dominated solutions from $\text{Off}(P_i) \cup P_i$ are added to P_{i+1} . If they are less than $|P_i|$, these solutions are removed from $\text{Off}(P_i) \cup P_i$ and the process is repeated with the remaining ones until $|P_{i+1}| = |P_i|$. If the size of 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} . Finally, the algorithm stops if the approach to the Pareto set remains constant for \max_{MOEA} consecutive iterations. A pseudocode description of the multiobjective evolutionary algorithm can be found in Fig. 1

The problem-specific parts of the NSGA-II procedure are the coding and the decoding and evaluation. For our problem, the genotype codifies a solution as a permutation with repetitions, as introduced in (Bierwirth 1995). This is a permutation of the set of tasks, where each task $o(j, l)$ is represented by its job number j . For example, for a problem with 3 jobs:

$$\begin{aligned} J_1 &= \{o(1, 1), o(1, 2)\}, \\ J_2 &= \{o(2, 1), o(2, 2), o(2, 3)\}, \\ J_3 &= \{o(3, 1), o(3, 2), o(3, 3)\}, \end{aligned}$$

a topological order of tasks $\sigma = (o(2, 1), o(1, 1), o(2, 2), o(3, 1), o(2, 3), o(3, 2), o(3, 3), o(1, 2))$ is codified by the sequence $\mathbf{x}_\sigma = (2, 1, 2, 3, 2, 3, 3, 1)$. This coding scheme covers all feasible schedules and no unfeasible one.

During the evaluation phase, every chromosome is decoded by generating an associated schedule and then the vector of fitness values $(\widehat{TWT}, \widehat{NPE})$ is computed. Based on the results from Section 4 we propose a decoding strategy consisting of two steps. First, a semi-active schedule is built from the task order encoded in the chromosome \mathbf{x} as explained in Section 3. Thus we search in the subspace of semi-active schedules, which is dominant for \widehat{TWT} . Second, local right shifts that are non-increasing in $E[\widehat{NPE}]$ are applied to tasks in the resulting schedule taking care not to deteriorate the other performance measure (i.e. \widehat{TWT}). To this end, the last task of each job is not right-shifted if that leads to an increase on the total weighted tardiness. The remaining tasks are right-shifted in inverse topological order following Proposition 1. By doing this, $E[\widehat{NPE}]$ is likely to decrease (and will never increase) and, at the same time, \widehat{TWT} will not change. We shall refer to this strategy as Heuristic Reduction of Energy Consumption, or HREC for short. The detailed heuristic decoding procedure can be seen in Fig. 2. Here, SemiactiveSGS refers to the procedure that produces a fuzzy semi-active schedule from a task order as explained in Section 3 and SM_o denotes the successor of task o in the machine sequence.

Require: a fuzzy JSP P , a chromosome \mathbf{x}
Ensure: a schedule s' for problem P
 $\mathbf{s} = \text{SemiactiveSGS}(P, \mathbf{x})$
 θ is the array of tasks codified by \mathbf{s}
while θ is not empty **do**
 remove o the last task in θ
 if o is the last task scheduled in a machine **then**
 $\hat{s}'_o = \hat{s}_o$
 else {not the last task in a machine}
 if o is the last task of a job j **then**
 $tJ_o^i = \max\{d_j - p_o^i, s_o^i\}$, $i = 1, 2, 3$
 else {not the last task of a job}
 $tJ_o^i = lsJ_o^i$, $i = 1, 2, 3$
 if o is the first task scheduled in a machine **then**
 $s_o^3 = \min\{lsM_o^3, tJ_o^3\}$
 else {not the first task in a machine}
 $s_o^3 = \min\{\max\{s_o^3, s_{SM_o}^1 - p_o^3\}, tJ_o^3\}$
 $s_o^2 = \min\{s_o^3, lsM_o^2, tJ_o^2\}$
 $s_o^1 = \min\{s_o^2, lsM_o^1, tJ_o^1\}$
 return the schedule s' ;

Fig. 2: Decoding procedure HREC to generate a schedule from a chromosome

6 Experimental study

The main purpose of provide an empirical assessment of the proposed decoding algorithm. To this end, we shall compare two versions of NSGA-II, one using HREC to decode chromosomes and the other one using simply SemiactiveSGS to obtain semi-active fuzzy schedules. In the sequel, we shall refer to these variants of NSGA-II as MO-HREC and MO-SA respectively.

6.1 Problem instances

Similar works for deterministic biobjective job shop with TWT and NPE such us (Liu et al. 2014; González et al. 2017) conduct experiments on the well-known instance FT10. We follow these works using FT10 but we also consider the other two instances FT06 (6 jobs and 6 machines) and FT20 (20 jobs and 5 machines) in the set. This should provide us with a more thorough and varied set of experimental results. Obviously, the original instances need to be modified to incorporate fuzzy durations, machine idle power consumption values, job weights and due dates.

Regarding fuzzy durations, we use the fuzzy versions of these instances FT_F06, FT_F10 and FT_F20 from (Palacios et al. 2016). They were are obtained following a fuzzification process, originally proposed by Fortemps (1997) for trapezoidal fuzzy numbers, so the original crisp processing time p_o of a task o coincides with the modal value p_o^2 of the fuzzy processing time \hat{p}_o as well as with its expected value $E[\hat{p}_o]$.

Machine idle power consumption levels P_k^{idle} are those proposed by Liu et al. (2014) and used in González et al. (2017). This is also the case for job weights for FT_F10. For the remaining instances, we follow a standard procedure from the literature (Singer and Pinedo 1998): the first 20% of jobs in the problem instance get a weight of 4, the next 60% get a weight of 2, and the final 20% get a weight of 1.

To obtain due dates, we also follow Singer and Pinedo (1998), with a slight modification to account for the fact that processing times are fuzzy, so $d_j = k \sum_{l=1}^{m_j} p_{o(j,l)}^2$ where k , taking values in $\{1.5, 1.6, 1.7, 1.8\}$ is a tightness factor. This expression, used for the first time by Palacios and Derbel (2015), actually yields the same due date values as in the deterministic setting (since the modal value for the fuzzy duration coincides with the original processing time) and is pretty natural. In a deterministic setting, the sum of the processing times of all operations in a job provides a lower bound of the job's completion time, so the due date is obtained by multiplying this lower bound by a tightness factor. In the fuzzy setting, since $p_{ij}^2 = E[\hat{p}_{ij}]$, by linearity of the

expected value, $\sum_{l=1}^{m_j} p_{o(j,l)}^2 = E[\sum_{l=1}^{m_j} \hat{p}_{o(j,l)}] \leq E[\widehat{C}_j]$ also provides a lower bound for the expected job completion time and d_j is simply the result of multiplying this lower bound by a given tightness factor k . In this way, 4 different sets of due dates for the jobs (decreasingly tight) are obtained, so a total of 12 instances is obtained from the original three.

All experiments reported in this section have been implemented in C++ and run on a PC Intel core i5-2400 (3,1 Ghz, 6Mb cache).

6.2 Comparison between MO-HREC and MO-SA

The comparison between the two multi-objective algorithms is made in terms of hypervolume (HV) (Fleischer 2003; Zitzler and Thiele 1998) and the unary additive ϵ indicator ($I_{\epsilon+}$) (Zitzler et al. 2003). In absence of the optimal Pareto front, PO^* , we approximate the reference set RF by the non-dominated elements of the union of all sets of solutions reached along all the experiments (Talbi 2009). When using these metrics, it is advised to normalise the values of the objective functions. Let $f_i(s)$ be the value of the i -th objective function for a solution $s \in S$, its normalised value is the following:

$$f_i(s) = \frac{f_i(s) - f_i^-(S)}{f_i^+(S) - f_i^-(S)}, \quad (21)$$

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

Table 1 summarises the results obtained after 10 runs of both MO-HREC and MO-SA. For each instance, it shows the average hypervolume (HV) and ϵ -indicator ($I_{\epsilon+}$) across the 10 sets of non-dominated solutions obtained by each algorithm, with standard deviation values between brackets. For each instance, a Kolmogorov-Smirnov test for normality is run over the results of each algorithm. In case of normality, algorithms are compared by means of an unpaired t -test, whereas a Mann-Whitney-U test is used otherwise. In each row, and for each metric, we highlight in bold those values that are significantly better than their counterparts according to these tests.

We can see that incorporating the HREC strategy as decoding operator clearly improves the performance of NSGA-II both in terms of HV and $I_{\epsilon+}$. In fact, MO-HREC outperforms MO-SA on every instance, with statistically significant differences in all cases for HV and in all but two cases for the $I_{\epsilon+}$ indicator.

Inst.	k	HV		$I_{\epsilon+}$	
		MO-SA	MO-HREC	MO-SA	MO-HREC
FT _F 06	1.5	0.895 (0.000)	0.913 (0.001)	0.027 (0.000)	0.002 (0.001)
	1.6	0.908 (0.002)	0.924 (0.001)	0.020 (0.000)	0.004 (0.001)
	1.7	0.908 (0.000)	0.927 (0.000)	0.021 (0.001)	0.001 (0.001)
	1.8	0.913 (0.000)	0.930 (0.000)	0.018 (0.000)	0.000 (0.000)
FT _F 10	1.5	0.767 (0.014)	0.808 (0.016)	0.049 (0.009)	0.034 (0.013)
	1.6	0.783 (0.016)	0.812 (0.016)	0.047 (0.010)	0.039 (0.015)
	1.7	0.786 (0.011)	0.829 (0.017)	0.052 (0.008)	0.034 (0.013)
	1.8	0.826 (0.018)	0.846 (0.019)	0.046 (0.010)	0.036 (0.015)
FT _F 20	1.5	0.686 (0.015)	0.724 (0.011)	0.061 (0.009)	0.030 (0.009)
	1.6	0.709 (0.011)	0.734 (0.015)	0.049 (0.008)	0.024 (0.008)
	1.7	0.715 (0.011)	0.739 (0.009)	0.052 (0.006)	0.033 (0.009)
	1.8	0.712 (0.015)	0.749 (0.011)	0.062 (0.011)	0.029 (0.006)

Table 1: Comparison, in terms of HV and the $I_{\epsilon+}$ values, between MO-SA and MO-HREC.

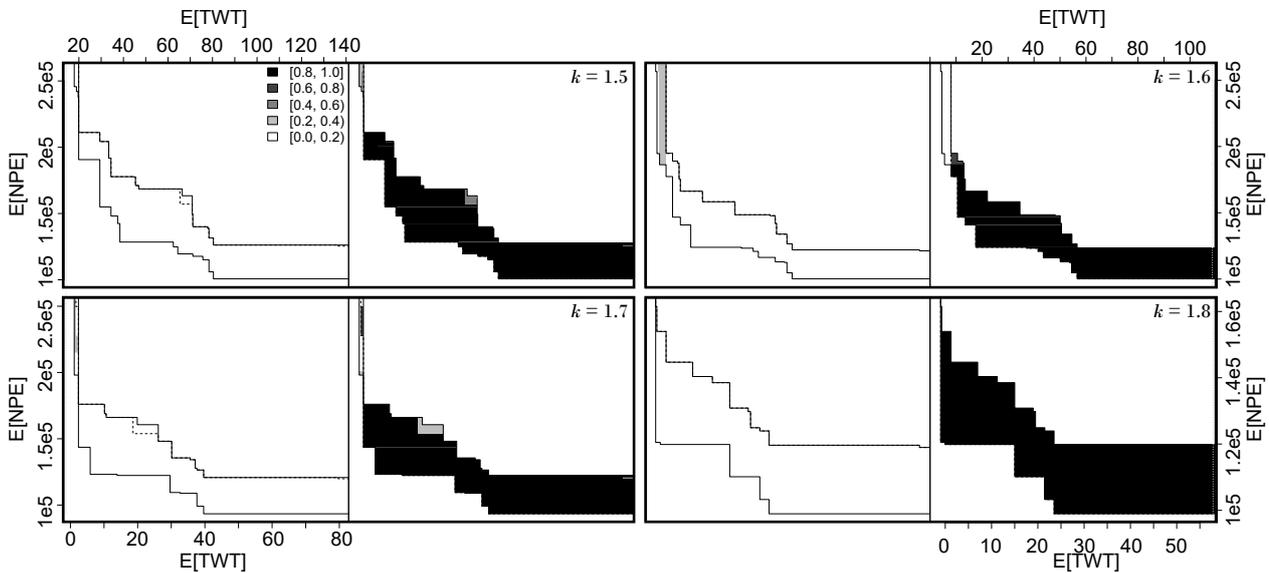


Fig. 3: EAF plots comparing NSGA-II using HREC and SA on instance FT_F06 (MO-SA - MO-HREC on the left and MO-HREC - MO-SA on the right).

Figures 3, 4 and 5 further illustrate the comparison between both algorithms. They depict for all instances the difference between the empirical attainment functions or EAFs (Grunert da Fonseca et al. 2001) obtained by both algorithms using the visualisation tool proposed in Lopez-Ibañez et al. (2010). The EAF plots show that MO-HREC clearly dominates MO-SA in the region with lower (better) energy consumption in all instances with probability close to 1 in some areas for all instances. Only on some instances (e.g. FT_F06 with $k = 1.6$, FT_F10 with $k = 1.8$ or FT_F20 with $k = 1.7$) there is some probability (between 0.2 and 0.4) that MO-SA dominates MO-HREC in a small region of the objective space, corresponding to the lowest $E[\widehat{TWT}]$ values but highest energy costs. On small instances stemming from FT06 the sets of solutions obtained by

MO-SA are dominated by the solutions obtained by MO-HREC with a probability very close to 1 in almost every region. This supports our theoretical analysis, showing the potential of the heuristic reduction of the energy without a significant loss of performance in tardiness.

7 Conclusions

In this paper we have considered the problem of reducing energy consumption in job shop scheduling while keeping a good level of satisfaction in job delivery times. Since energy awareness is a real concern in many real projects, we have tried to reduce the gap between academic and real-world by considering uncertainty in task

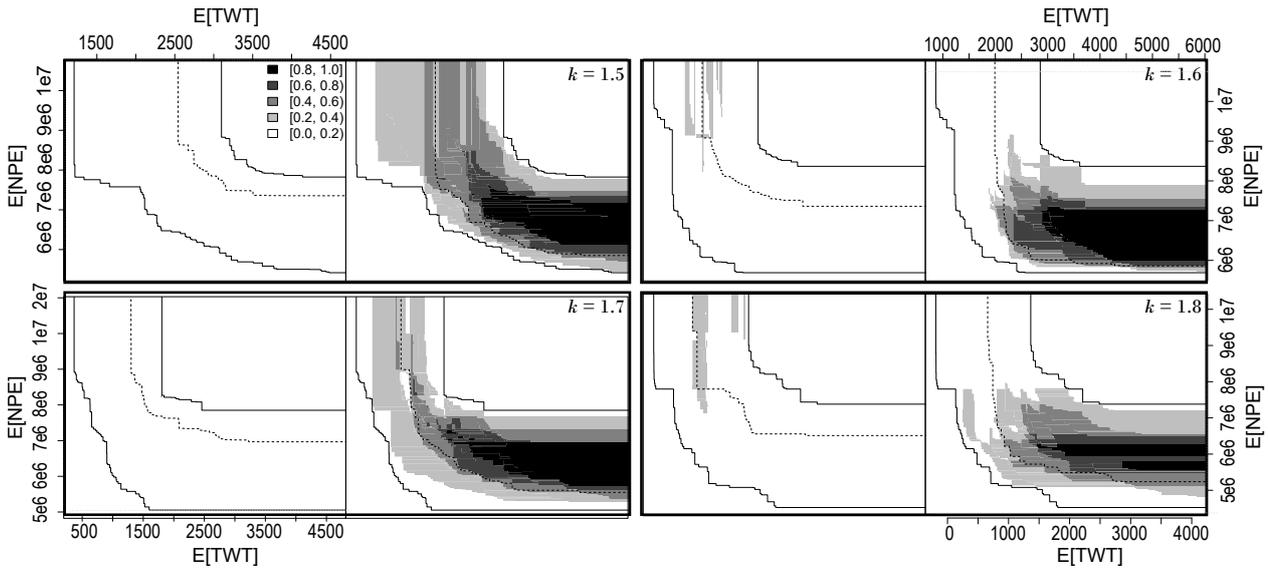


Fig. 4: EAF plots comparing NSGA-II using HREC and SA on instance FT_{F10} (MO-SA - MO-HREC on the left and MO-HREC - MO-SA on the right).

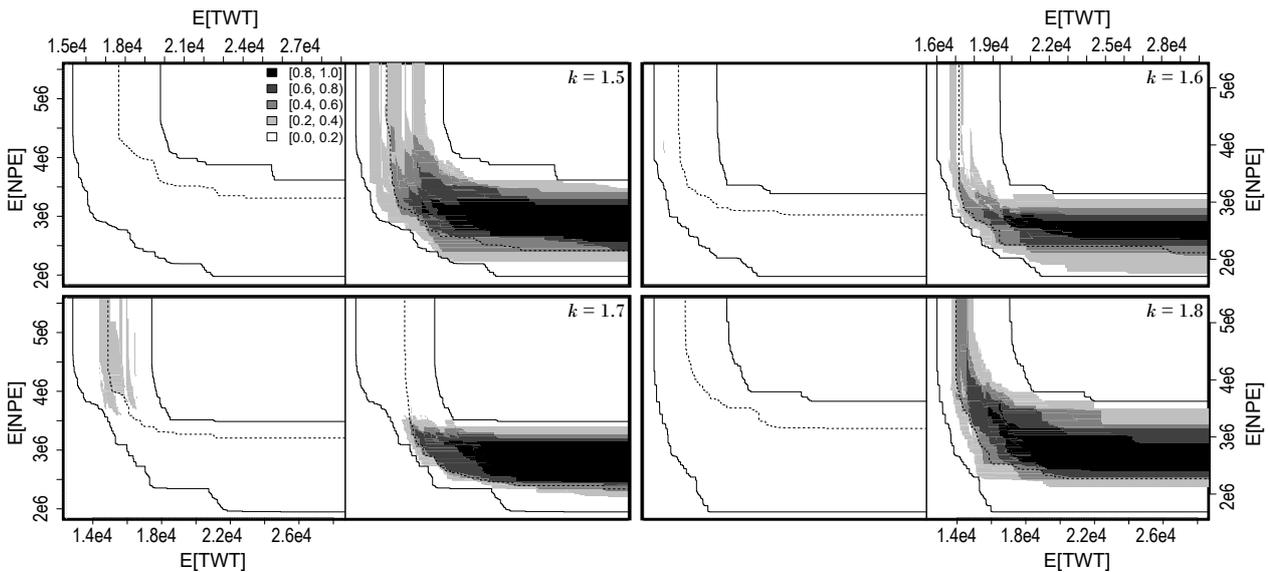


Fig. 5: EAF plots comparing NSGA-II using HREC and SA on instance FT_{F20} (MO-SA - MO-HREC on the left and MO-HREC - MO-SA on the right).

processing times, modelled with triangular fuzzy numbers. This results in uncertainty both in energy consumption, measured as the Total Non-processing Electricity Consumption (NPE), and in due-date satisfaction, measured as the Total Weighted Tardiness (TWT). We have provided a first definition and theoretical study on the properties of fuzzy NPE. This has allowed us to propose a schedule generation scheme including a heuristic reduction of energy consumption (HREC). Experimental results have shown that including this heuris-

tic strategy in a well-known multi-objective evolutionary algorithm such as NSGA-II improves its performance with respect of simply using a decoder producing semi-active schedules. The empirical attainment functions obtained by both versions of NSGA-II indicate that the sets of solutions obtained using HREC tend to yield lower energy costs than those obtained using a fuzzy semi-active scheduler, without significant loss in quality in terms of due-date satisfaction.

8 Compliance with Ethical Standards

- **Funding:** This study was funded by the Spanish Government under research grant TIN2016-79190-R and by the Principality of Asturias Government under grant IDI/2018/000176.
- **Conflict of Interest:** Author Inés González-Rodríguez declares that she has no conflict of interest. Author Jorge Puente declares that he has no conflict of interest. Author Juan José Palacios declares that he has no conflict of interest. Author Camino R. Vela declares that she has no conflict of interest.
- **Ethical approval:** This article does not contain any studies with human participants or animals performed by any of the authors.

References

- Abdullah S, Abdolrazzagah-Nezhad M (2014) Fuzzy job-shop scheduling problems: A review. *Information Sciences* 278:380–407, DOI 10.1016/j.ins.2014.03.060
- Artigues C, Lopez P, Ayache P (2005) Schedule generation schemes for the job shop problem with sequence-dependent setup times: Dominance properties and computational analysis. *Annals of Operations Research* 138:21–52
- Aytung H, Lawley MA, McKay K, Shantha M, Uzsoy R (2005) Executing production schedules in the face of uncertainties: A review and some future directions. *European Journal of Operational Research* 161:86–110
- Bierwirth C (1995) A generalized permutation approach to jobshop scheduling with genetic algorithms. *OR Spectrum* 17:87–92
- Deb K (2014) Multi-objective optimization. In: Burke EK, Kendall G (eds) *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, Springer, pp 403–449, DOI 10.1007/978-1-4614-6940-715
- Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6(2):182–197
- Dubois D, Prade H (eds) (2000) *Fundamentals of Fuzzy Sets. The Handbooks of Fuzzy Sets*, Kluwer Academic Publishers, Boston/London/Dordrecht
- Dubois D, Fargier H, Fortemps P (2003) Fuzzy scheduling: Modelling flexible constraints vs. coping with incomplete knowledge. *European Journal of Operational Research* 147:231–252
- Firas AQ, Denis G (2015) A multi-objective genetic method minimizing tardiness and energy consumption during idle times. *International Federation of Automatic Control (IFAC)– PapersOnLine* 48(3):1216–1223, DOI 10.1016/j.ifacol.2015.06.250
- Fleischer M (2003) The measure of Pareto optima. applications to multi-objective metaheuristics. In: Fonseca CM, Fleming PJ, Zitzler E, Thiele L, Deb K (eds) *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, vol 2632, Springer, pp 519–533, DOI 10.1007/3-540-36970-837
- Fortemps P (1997) Jobshop scheduling with imprecise durations: a fuzzy approach. *IEEE Transactions of Fuzzy Systems* 7:557–569
- Fortemps P, Roubens M (1996) Ranking and defuzzification methods based on area compensation. *Fuzzy Sets and Systems* 82:319–330
- Garey MR, Johnson DS (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman
- González MA, Vela CR (2015) An efficient memetic algorithm for total weighted tardiness minimization in a single machine with setups. *Applied Soft Computing* 37:506–518
- González MA, González-Rodríguez I, Vela C, Varela R (2012) An efficient hybrid evolutionary algorithm for scheduling with setup times and weighted tardiness minimization. *Soft Computing* 16:2097–2113
- González MA, Oddi A, Rasconi R (2017) 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)*, pp 140–148
- Graham R, Lawler E, Lenstra J, Rinnooy Kan A (1979) Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics* 4:287–326
- Grunert da Fonseca V, Fonseca CM, Hall AO (2001) Inferential performance assessment of stochastic optimisers and the attainment function. In: Zitzler E, Thiele L, Deb K, Coello Coello CA, Corne D (eds) *Evolutionary Multi-Criterion Optimization*, Springer, pp 213–225, DOI 10.1007/3-540-44719-9_15
- Heilpern S (1992) The expected value of a fuzzy number. *Fuzzy Sets and Systems* 47:81–86
- Herroelen W, Leus R (2005) Project scheduling under uncertainty: Survey and research potentials. *European Journal of Operational Research* 165:289–306
- Jain AS, Meeran S (1999) Deterministic job-shop scheduling: Past, present and future. *European Journal of Operational Research* 113:390–434
- Koulamas C (1994) The total tardiness problem: review and extensions. *Operations Research* 42(6):1025–1041, DOI 10.1287/opre.42.6.1025
- Kuhpfahl J, Bierwirth C (2016) A study on local search neighbourhoods for the job shop scheduling problem with total weighted tardiness objective. *Computers & Operations Research* 261:44–57
- Liu Y, Dong H, Lohse N, Petrovic S, Gindy N (2014) An investigation into minimising total energy consumption and total weighted tardiness in job shops. *Journal of Cleaner Production* 65:87–96, DOI 10.1016/j.jclepro.2013.07.060
- Liu Y, Dong H, Lohse N, Petrovic S (2016) A multi-objective genetic algorithm for optimisation of energy consumption and shop floor production performance. *International Journal of Production Economics* 179:259–272, DOI 10.1016/j.ijpe.2016.06.019
- Lopez-Ibañez M, Paquete L, Stützle T (2010) Exploratory analysis of stochastic local search algorithms in biobjective optimization. In: *Experimental Methods for the Analysis of Optimization Algorithms*, Springer, chap 9, pp 209–222
- Palacios JJ, Derbel B (2015) On maintaining diversity in MOEA/D: Application to a biobjective combinatorial FJSP. In: *GECCO '15 Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, ACM, pp 719–726, DOI 10.1145/2739480.2754774
- Palacios JJ, Vela CR, González-Rodríguez I, Puente J (2014) Schedule generation schemes for job shop problems with fuzziness. In: Schaub T, Friedrich G, O’Sullivan B (eds) *Proceedings of ECAI 2014*, IOS Press, *Frontiers in Arti-*

- cial Intelligence and Applications, vol 263, pp 687–692, DOI 10.3233/978-1-61499-419-0-687
- Palacios JJ, Puente J, Vela CR, González-Rodríguez I (2016) Benchmarks for fuzzy job shop problems. *Information Sciences* 329:736–752, DOI 10.1016/j.ins.2015.09.042
- Palacios JJ, González-Rodríguez I, Vela CR, Puente J (2017) Robust multiobjective optimisation for fuzzy job shop problems. *Applied Soft Computing* 56:604–616, DOI 10.1016/j.asoc.2016.07.004
- Paolucci M, Anguinolfi D, Tonelli F (2017) Facing energy-aware scheduling: a multi-objective extension of a scheduling support system for improving energy efficiency in a moulding industry. *Soft Computing* 21:3687–3698, DOI 10.1007/s00500-015-1987-8
- Pinedo ML (2016) *Scheduling. Theory, Algorithms, and Systems.*, 5th edn. Springer
- Singer M, Pinedo M (1998) A computational study of branch and bound techniques for minimizing the total weighted tardiness in job shops. *IIE Transactions* 30(2):109–118, DOI 10.1023/A:1007405915227
- Sprecher A, Kolisch R, Drexel A (1995) Semi-active, active, and non-delay schedules for the resource-constrained project scheduling problem. *European Journal of Operational Research* 80:94–102
- Talbi EG (2009) *Metaheuristics. From Design to Implementation.* Wiley
- Tian H, Yuan X, Huang Y (2015) An improved gravitational search algorithm for solving short-term economic/environmental hydrothermal scheduling. *Soft Computing* 19:2783–2797
- Zhang R, Chiong R (2016) Solving the energy-efficient job shop scheduling problem: a multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. *Journal of Cleaner Production* 112(4):3361–3375, DOI 10.1016/j.jclepro.2015.09.097
- Zitzler E, Thiele L (1998) Multiobjective optimization using evolutionary algorithms—a comparative case study. In: Eiben AE, Bäck T, Schoenauer M, Schwefel HP (eds) *International conference on parallel problem solving from nature*, pp 292–301, DOI 10.1007/BFb0056872
- Zitzler E, Thiele L, Laumanns M, Fonseca CM, da Fonseca VG (2003) Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation* 7(2):117–132, DOI 10.1109/TEVC.2003.810758