Natural Computing (2023) 22:685–704
https://doi.org/10.1007/s11047-023-09967-w

Neighbourhood search for energy minimisation in flexible job shops under fuzziness

Pablo García Gómez[1] · Camino R. Vela[2] · Inés González-Rodríguez[1]

Accepted: 3 October 2023 / Published online: 18 October 2023

Abstract
Uncertainty pervades real life and supposes a challenge for all industrial processes as it makes it difficult to predict the outcome of otherwise risk-free activities. In particular, time deviation from projected objectives is one of the main sources of economic losses in manufacturing, not only for the delay in production but also for the energy consumed by the equipment during the additional unexpected time they have to work to complete their labour. In this work we deal with uncertainty in the flexible job shop, one of the foremost scheduling problems due to its practical applications. We show the importance of a good model to avoid introducing unwanted imprecision and producing artificially pessimistic solutions. In our model, the total energy is decomposed into the energy required by resources when they are actively processing an operation and the energy consumed by these resources simply for being switched on. We propose a set of metrics and carry out an extensive experimental analysis that compares our proposal with the more straightforward alternative that directly translates the deterministic model. We also define a local search neighbourhood and prove that it can reach an optimal solution starting from any other solution. Results show the superiority of the new model and the good performance of the new neighbourhood.

Keywords Flexible job shop scheduling · Energy consumption · Fuzzy numbers · Neighbourhood

1 Introduction

Scheduling consists in arranging the execution of a set of operations on a set of resources under some constraints to complete a set of objectives in the best way (Pinedo 2016; Błazewicz et al. 2019). It is a classical problem in manufacturing and industrial activities in general (Han et al. 2022; Bakon et al. 2022), but it is also fundamental in areas such as healthcare (Razali et al. 2022; Luo et al. 2020), computing infrastructures (Mansouri and Ghafari 2022; Ye et al. 2019), education (Imaran Hossain et al. 2019) or airport baggage transportation (Guo et al. 2020).

✉ Pablo García Gómez
pablo.garciagomez@unican.es

Camino R. Vela
crvela@uniovi.es

Inés González-Rodríguez
gonzalezri@unican.es

1 Universidad de Cantabria, Santander, Cantabria, Spain

2 Universidad de Oviedo, Oviedo, Asturias, Spain

One of the most prominent families of scheduling problems is job shop scheduling problems (Çalis and Bulkan 2015) because their different variants can be adapted to model many different real-life problems (Xiong et al. 2022). In particular, the flexible job shop, which not only solves the classic operation-sequencing problem but also deals with a resource-assignment subproblem, draws special attention due to its relevance for industrial problems (Gao et al. 2019).

Most scheduling problems, and the flexible job shop in particular, are NP-hard (Lenstra et al. 1977), so exact algorithms that guarantee finding the optimal solutions can only be applied to the simplest instances. For this reason, approximate search algorithms have been proposed with the objective of finding good-enough solutions using a reasonable amount of computational resources (Gendreau and Potvin 2019). In order to find new innovative algorithms, researchers have been taking inspiration from the natural world, developing strategies that emulate natural phenomena to search for solutions in more intelligent ways that lessen the computational costs. These methods have been successful in many real-world and shop scheduling

Table 1 List of symbols

Symbol	Description
$\mathbf{O}$	The set of operations
$\mathbf{R}$	The set of resources
m	The number of resources
$\mathbf{J}$	The set of jobs
n	The number of jobs
$\mathbf{R}(o)$	The subset of resources in which operation o can be executed
$\chi(o)$	The job to which operation o belongs to
$\eta(o)$	The position in which operation o has to be executed relative to its job
$\widehat{p}(o,r)$	The processing time of operation o in the resource r
$P_p(r)$	The passive power consumption of resource r
$P_a(o,r)$	The active power consumption of executing operation o in resource r
$\boldsymbol{\tau}_\phi$	The resource assignments for solution ϕ
$\mathbf{s}_\phi$	The starting time assignments for solution ϕ
$\boldsymbol{\sigma}_\phi$	The operation processing order for solution ϕ
$\tau_\phi(o)$	The resource assigned to operation o in solution ϕ
$\widehat{s_\phi}(o)$	The starting time assigned to operation o in solution ϕ
$\sigma_\phi(o)$	The position of operation o in $\boldsymbol{\sigma}_\phi$
$\widehat{c_\phi}(o)$	The completion time of operation o in solution ϕ
$\widehat{p_\phi}(o)$	The processing time of operation o in solution ϕ. Equivalent to $\widehat{p}(o,\tau_\phi(o))$
$\widehat{h_\phi}(o)$	The head of operation o in solution ϕ
$\widehat{q_\phi}(o)$	The tail of operation o in solution ϕ
$JP(o)$	The job predecessor of operation o
$JS(o)$	The job successor of operation o
$RP_\phi(o)$	The resource predecessor of operation o in solution ϕ
$RS_\phi(o)$	The resource successor of operation o in solution ϕ
$\widehat{C_{max}}(\phi)$	The makespan of solution ϕ
$\widehat{E_p}(\phi)$	The passive energy consumption in solution ϕ
$\widehat{E_a}(\phi)$	The active energy consumption in solution ϕ
$\widehat{E}(\phi)$	The total energy consumption in solution ϕ
$\mathbf{T}_{C_{max}}(\phi)$	The set of all makespan-critical operations in solution ϕ
$\mathbf{B}_{C_{max}}(\phi)$	The set of all makespan-critical blocks in solution ϕ
$\mathbb{E}(\widehat{a})$	The expected value of TFN $\widehat{a}$
$S(\widehat{a})$	The spread of TFN $\widehat{a}$
$MVP(\widehat{a})$	The modal value position of TFN $\widehat{a}$
$RDEV(\phi,\boldsymbol{\rho})$	The relative distance of a possible scenario $\boldsymbol{\rho}$ to the expected value of a solution ϕ
$UU(\phi,\mathbf{S})$	The used uncertainty of a set of possible scenarios $\mathbf{S}$ for a solution ϕ

problems (Palacios et al. 2014, 2019; Guo et al. 2020; Niu et al. 2021; Ye et al. 2019), including the flexible job shop (Palacios et al. 2015a; Chaudhry and Khan 2016; Zuo et al. 2017; Gao et al. 2019; Gen et al. 2021; García Gómez et al. 2022).

Many of these algorithms need to define neighbourhood functions that allow them to travel from one solution to another. Defining efficient neighbourhoods with theoretical properties such as allowing to reach optimal solutions from any other solution or generating only feasible solutions is a very important research area of job shop (Mattfeld 1995; Kuhpfahl and Bierwirth 2016). Nevertheless, using these neighbourhood functions alone, in so-called trajectory-based methods, is not usually enough and they can be hybridized with other search strategies. In particular, combining trajectory-based and population-based search algorithms results in a type of algorithms named memetic algorithms that have been proved to be very successful (Gong et al. 2020; Mencía et al. 2022; García Gómez et al. 2022) due to their balance between exploration of the search space and exploitation of the most promising zones (Osaba et al. 2022).

The objective of scheduling problems has been evolving from the earliest days of industrialisation to the present day. Historically, the most common objective was minimising the maximum completion time, usually known as the makespan. While it is still relevant, it has increasingly been replaced by other production-related measures such as tardiness, as a response to changes in supply-chains consequence of the relocation of factories to different countries. Nowadays the trend is changing again, and the reduction of energy consumption has been introduced as a new objective to adapt to newer laws and regulations that seek a reduction in the environmental footprint as a countermeasure to global warming. Moreover, due to highly-tensed geopolitical relations, energy costs are rising and the industry needs to make a more efficient use of energy to remain competitive. For these reasons, in recent years we have seen a significant increase in research in this area of scheduling (González et al. 2017; Li and Wang 2022; Liu et al. 2019; Gong et al. 2020; Han et al. 2022; Mansouri and Ghafari 2022).

To make models simpler and the search for solutions less computationally demanding, most researchers in scheduling limit the complexity by limiting themselves to a deterministic setting. However, real-life problems have uncertainty and working with it is in itself a challenge, so models cannot ignore it if they want to be useful (Bakon et al. 2022). One of the most successful ways of modelling uncertainty in scheduling is using fuzzy numbers, and in particular, triangular fuzzy numbers (Abdullah and Abdolrazzagh-Nezhad 2014; Palacios et al. 2016) because as pointed out in Dubois et al. (2003), they provide a good balance between ease of understanding, informativity and computing cost.

It is easy to find in the literature several works that make use of metaheuristics to optimise the more traditional objectives in a fuzzy job shop (Abdullah and Abdolrazzagh-Nezhad 2014; Palacios et al. 2019) and, more specifically, in the fuzzy flexible job shop (Sun et al. 2019; Li et al. 2022a). Works that incorporate both uncertainty and energy-consumption objectives are more scarce. Regarding the job shop without flexibility, in González-Rodríguez et al. (2020) the authors propose an evolutionary algorithm to reduce the non-processing energy and the total weighted tardiness, whereas in Afsar et al. (2022) a memetic algorithm is used to minimise the non-processing energy and the makespan. As for job shops with flexibility, in García Gómez et al. (2022, 2023) the authors use memetic algorithms to minimise total energy consumption; in Pan et al. (2022) a bi-population evolutionary algorithm with a feedback mechanism and enhanced local search is proposed with the goal of minimising both the fuzzy makespan together and the fuzzy total energy consumption and maximising flexible due-date satisfaction; and in Li et al. (2022b) a learning-based memetic algorithm is used to minimise makespan and energy consumption.

In this work, we will consider the flexible job shop with fuzzy processing times and the objective of minimizing the total energy consumption. We define the total energy consumption as the sum of the passive energy, the energy consumed by resources only by the fact of being switched on, and the active energy, consumed by resources when they are working in addition to the passive energy. This proposal builds on the conference paper from García Gómez et al. (2022) and the objective is to complete the ideas therein as follows:

- Propose a set of metrics to compare two fuzzy models of total energy for the flexible job shop. To do so we motivate some measures that compare the models based both on their fuzzy values and on the use of crisp samples to simulate the behaviour in a real setting.
- Compare the proposed energy model for the flexible job shop with the more straightforward alternative that directly translates the crisp model.
- Propose a new neighbourhood function that ensures feasibility and connectivity. We formally demonstrate that the neighbourhood allows to obtain an optimal solution starting from any other solution. This is a desirable property when the neighbourhood is used in a wide variety of local search algorithms.
- Integrate the proposed neighbourhood in a memetic algorithm and compare its performance with the state of the art. The results show that it is competitive and especially relevant in CPU-constrained applications.

The rest of the paper is organised as follows: Sect. 2 formally defines the problem and introduces the measures for comparing fuzzy schedules, Sect. 3 describes and studies the proposed neighbourhood structure and Sect. 4 reports and analyzes the experimental results to evaluate the potential of our proposal. Finally, Sect. 5 presents some conclusions.

2 Problem formulation

The flexible job shop scheduling problem consists in scheduling a set $\mathbf{O}$ of operations (also called tasks) in a set $\mathbf{R}$ of m resources (also called machines) subject to a set of constraints. Operations are organised in a set $\mathbf{J}$ of n jobs, so operations within a job must be sequentially scheduled. Given an operation $o \in \mathbf{O}$, the job to which it belongs is denoted by $\chi(o) \in \mathbf{J}$ and the position in which it has to be executed relative to this job is denoted by $\eta(o)$. There are also capacity constraints, by which every operation $o \in \mathbf{O}$ can only be processed in a resource from a given subset $\mathbf{R}(o) \subseteq \mathbf{R}$ with its processing time $p(o, r)$ depending on the resource $r \in \mathbf{R}(o)$ on which it is executed. Also, the operation requires the uninterrupted and exclusive use of the resource for its whole processing time.

In the energy-aware problem, the resources need energy to function, with the amount of energy depending on two overlapping states: on and active. Just for being switched on, each resource $r \in \mathbf{R}$ consumes a certain amount of energy per time unit. This constitutes the resource's passive power consumption $P_p(r)$. Additionally, when an operation $o \in \mathbf{O}$ is processed in a resource $r \in \mathbf{R}(o)$, this resource consumes an extra amount of energy per time unit, the active power $P_a(o, r)$.

2.1 Fuzzy processing times

We model processing times as triangular fuzzy numbers (TFNs), a particular type of fuzzy numbers, with an interval $[a^1, a^3]$ of possible values and a modal value a^2 (Dubois and Prade 1993). A TFN can be represented as a triplet $\widehat{a} = (a^1, a^2, a^3)$, so its membership function is given by:

$$\mu_{\widehat{a}}(x) = \begin{cases} \dfrac{x - a^1}{a^2 - a^1} & a^1 \le x \le a^2 \\ \dfrac{x - a^3}{a^2 - a^3} & a^2 \le x \le a^3 \\ 0 & \text{otherwise} \end{cases}$$

Arithmetic operations on TFNs are defined based on the Extension Principle, so for two TFNs $\widehat{a}$ and $\widehat{b}$, the sum is defined as:

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

and the product of a scalar c by a TFN $\widehat{a}$ is defined as:

$$c\widehat{a} = (ca^1, ca^2, ca^3). \tag{2}$$

Extending the maximum is not that straightforward because the set of TFNs is not closed under the operation obtained with the Extension Principle. For this reason, in fuzzy scheduling it is usual to approximate it by interpolation as follows:

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

If the result of the maximum is a TFN, it coincides with the approximated value and, in any case, the approximation always preserves the support and the modal value (cf. (Palacios et al. 2015b)).

Another operation where we must be cautious when extending it is subtraction. For two TFNs $\widehat{a}$ and $\widehat{b}$, their difference is defined as:

$$\widehat{a} - \widehat{b} = (a^1 - b^3, a^2 - b^2, a^3 - b^1) \tag{4}$$

Notice however that the difference is not necessarily the inverse of the sum, that is, $\widehat{a} \neq (\widehat{a} + \widehat{b}) - \widehat{b}$. This is because underlying the definition is the assumption that $\widehat{a}$ and $\widehat{b}$ are non-interactive variables, i.e., they can be assigned values in their support independently of each other. Therefore, when the TFNs $\widehat{a}$ and $\widehat{b}$ are linked, using the difference may result in an "over-uncertain" or "pessimistic" quantity, which may be a problem in certain applications. It is, for instance, the main difficulty in critical-path analysis and backpropagation in activity networks with fuzzy durations (Dubois et al. 2003).

Another important issue when considering fuzzy processing times is the lack of a natural total order for TFNs. Some ranking method is thus needed in order to make comparisons. A ranking method widely used in the literature is the one based on the expected values. For a TFN $\widehat{a}$, its expected value is:

$$\mathbb{E}[\widehat{a}] = \frac{a^1 + 2a^2 + a^3}{4} \tag{5}$$

and this can be used to rank any two TFNs $\widehat{a}$ and $\widehat{b}$ as follows:

$$\widehat{a} \le_E \widehat{b} \text{ if and only if } \mathbb{E}[\widehat{a}] \le \mathbb{E}[\widehat{b}] \tag{6}$$

Accordingly, we will write $\widehat{a} <_E \widehat{b}$ when $\mathbb{E}[\widehat{a}] < \mathbb{E}[\widehat{b}]$.

In addition, we consider partial order relations based on precedence, so for any pair of two TFNs $\widehat{a}$ and $\widehat{b}$:

$$\widehat{a} \prec \widehat{b} \text{ if and only if } a^i < b^i \, \forall i \in \{1, 2, 3\} \tag{7}$$

with non-strict precedence $\widehat{a} \preceq \widehat{b}$ if $a^i \le b^i \; \forall i \in \{1, 2, 3\}$. We may also write $\widehat{b} \succ \widehat{a}$ (resp. $\widehat{b} \succeq \widehat{a}$) as an equivalent of $\widehat{a} \prec \widehat{b}$ (resp $\widehat{a} \preceq \widehat{b}$). Notice that if $\widehat{a} \prec \widehat{b}$ ($\widehat{a} \preceq \widehat{b}$), then $\widehat{a} <_E \widehat{b}$ ($\widehat{a} \le_E \widehat{b}$), but the inverse is not necessarily true.

2.2 Fuzzy schedules

A solution $\phi = (\boldsymbol{\tau}_\phi, \mathbf{s}_\phi)$ to the problem, commonly called schedule, consists of both a resource assignment $\boldsymbol{\tau}_\phi$ and starting time assignment $\mathbf{s}_\phi$ to all operations. This solution is feasible if and only if all precedence and capacity constraints hold. More formally, for an operation $o \in \mathbf{O}$ let $\tau_\phi(o)$ be the resource it is assigned to in ϕ and let $\widehat{s_\phi}(o)$ and $\widehat{c_\phi}(o) = \widehat{s_\phi}(o) + \widehat{p}(o, \tau_\phi(o))$ be, respectively, its starting and completion times in the solution ϕ. Then, solution ϕ is feasible if all constraints hold, that is, for any pair of operations $o, u \in \mathbf{O}$, precedence constraints hold if

$$\widehat{c_\phi}(o) \preceq \widehat{s_\phi}(u) \text{ when } \eta(o) < \eta(u), \chi(o) = \chi(u) \tag{8}$$

and capacity constraints hold if

$$\tau(o) \in \mathbf{R}(o) \tag{9}$$

$$\widehat{c_\phi}(o) \preceq \widehat{s_\phi}(u) \vee \widehat{c_\phi}(u) \preceq \widehat{s_\phi}(o) \text{ when } \tau_\phi(o) = \tau_\phi(u). \tag{10}$$

For a feasible solution ϕ, the makespan is the maximum completion time of all tasks:

$$\widehat{C_{max}}(\phi) = \max_{o \in \mathbf{O}} \widehat{c_\phi}(o) \tag{11}$$

Notice that a feasible starting time assignment $\mathbf{s}_\phi$ induces partial orderings for all the operations in the same job and all operations in the same resource. These can be represented by a global operation processing order $\boldsymbol{\sigma}_\phi$ where the position of operation o in $\boldsymbol{\sigma}_\phi$ is denoted by $\sigma_\phi(o)$ and where $\sigma_\phi(o) < \sigma_\phi(u)$ for any pair of operations $o, u \in \mathbf{O}$ in the same job $(\chi(o) = \chi(u))$ or in the same resource $(\tau_\phi(o) = \tau_\phi(u))$ such that o is completed before u starts $(\widehat{c_\phi}(o) \preceq \widehat{s_\phi}(u))$. Conversely, given a feasible resource assignment $\boldsymbol{\tau}$ and a global operation processing order $\boldsymbol{\sigma}$ compatible with job precedence constraints (that is, $\sigma(o) < \sigma(u)$ when $\eta(o) < \eta(u), \chi(o) = \chi(u)$), a feasible starting time assignment $\mathbf{s}$ can be obtained from $\boldsymbol{\sigma}$ by assigning to each operation the earliest possible starting time (in the sense of $\preceq$) satisfying precedence and resource capacity constraints according to this global operation order, that is, for every pair of operations $o, u \in \mathbf{O}$ in the same job or assigned to the same resource (either $\chi(o) = \chi(u)$ or $\tau_\phi(o) = \tau_\phi(u)$), if $\sigma(o) < \sigma(u)$, then $\widehat{c}(o) \preceq \widehat{s}(u)$.

In consequence, we can indistinctly refer to a solution as a pair $\phi = (\boldsymbol{\tau}_\phi, \mathbf{s}_\phi)$ or as a pair $\phi = (\boldsymbol{\tau}_\phi, \boldsymbol{\sigma}_\phi)$. When no confusion is possible, for the sake of a simpler notation, we may drop the subindex ϕ from the solution's components.

2.3 Total energy consumption

We consider that the total energy consumption in a solution ϕ stems from two different sources: passive and active energy consumption. On the one hand, for a given solution ϕ there is an intrinsic passive energy consumption $\widehat{E_p}(\phi)$ incurred as soon as the resources are turned on and until they are turned off. The amount of energy required by a resource $r \in \mathbf{R}$ depends on its passive power consumption $P_p(r)$ and is proportional to the time the resource remains switched on. Assuming that all resources are simultaneously turned on and off (respectively, at instant 0 and at the time the whole project is completed), the passive energy consumption in a solution ϕ is thus obtained as follows:

$$\widehat{E_p}(\phi) = \sum_{r \in \mathbf{R}} P_p(r) \widehat{C_{max}}(\phi) \tag{12}$$

On the other hand, when an operation $o \in \mathbf{O}$ is processed in a resource $r \in \mathbf{R}(o)$, this results in an additional energy cost, denoted $\widehat{E_a}(o, r)$, which depends on the active power $P_a(o, r)$ required to execute o in r and is proportional to the processing time $\widehat{p}(o, r)$:

$$\widehat{E_a}(o, r) = P_a(o, r)\widehat{p}(o, r) \tag{13}$$

Thus, given a solution ϕ, the active energy consumption will be:

$$\widehat{E_a}(\phi) = \sum_{o \in \mathbf{O}, \mathbf{r} = \tau_\phi(\mathbf{o})} \widehat{E_a}(o, r) \tag{14}$$

Our model is an additive model so the total energy consumption is the sum of the passive and the active energy consumption:

$$\widehat{E}(\phi) = \widehat{E_p}(\phi) + \widehat{E_a}(\phi) \tag{15}$$

2.3.1 Alternative energy models in the fuzzy context

The total energy consumption is modelled as the sum of the energy consumed when the resources are processing operations and the energy consumed only by the fact of being switched on.

Having said that, in the deterministic problem the usual way to model this kind of energy consumption is to assume that the possible states for the resources are disjoint, so there is a power consumption when they are active and a different power consumption when they are idle. In this way, the total energy is the sum of the processing energy (i.e., the energy consumed by resources when they are active) and the non-processing energy (i.e., the energy consumed when resources are idle). For each resource, its total idle time is obtained by adding all time gaps between the completion of one operation and the start of the next operation in that resource; these individual idle time gaps are easily obtained using subtraction.

If this model is directly translated to the fuzzy problem, idle gaps are computed as the difference between two fuzzy numbers (4), resulting in over-uncertain idle times, with a support including many values that are not attainable. In particular, the difference may include negative values, which are clearly impossible. Negative values can be avoided by truncating the difference, so if o is an operation and u is its successor in the resource, the idle time between o and u is computed as $\max((0, 0, 0), \widehat{s}_u - \widehat{c}_o)$. This certainly solves the problem of having negative values for idle times, but we still have an artificially-increased support for the idle time and, in consequence, some artificial uncertainty in the total non-processing energy. We will refer to the resulting model for total energy consumption in fuzzy flexible job shop as the "gaps model".

The total energy model adopted in this work was proposed in García Gómez et al. (2022) as an alternative translation of the deterministic approach to total energy consumption that avoids subtracting interactive fuzzy numbers by considering that resources can be in overlapping states. In this way, total energy consumption can be seen as the result of two stacked layers, one that corresponds to the energy consumed by resources simply by being on, and another layer that corresponds to the extra energy needed by resources when they are working. This alternative model for total energy consumption will be referred to as the "stack model".

Notice that, in the deterministic setting, the gaps and the stack approaches are equivalent. However, using overlapping states in the fuzzy setting avoids introducing artificial uncertainty due to subtraction. Let us illustrate this with an example. Consider a toy problem with only two operations o_1 and o_2 belonging to different jobs and sharing the same resource r. Let $p(o_1,r)=(1,2,3)$, $p(o_2,r)=(2,2,2)$, $P_p(r)=1$, $P_a(r,o_1)=2$ and $P_a(r,o_2)=2$. Regardless of the order in which operations are executed, the resource will start at (0, 0, 0) and finish at (3, 4, 5). Let us suppose that in a solution ϕ operation o_1 is scheduled before o_2, so (1, 2, 3) is both the completion time of o_1, $\widehat{c_\phi}(o_1)$, and the starting time of o_2, $\widehat{s_\phi}(o_2)$. The gap between these operations, and consequently the idle time of the resource r, would be computed with fuzzy subtraction as $\max((0,0,0),\widehat{s_\phi}(o_2)-\widehat{c_\phi}(o_1))=(0,0,2)$. In this way, the energy consumption according to the gap model will be $P_p(r)\widehat{idle_\phi}(r)+(P_p(r)+P_a(o_1,r))\widehat{p}(o_1,r)+(P_p(r)+P_a(o_2,r))\widehat{p}(o_2,r)=1(0,0,2)+3(1,2,3)+3(2,2,2)=(9,12,17)$. However, we know that the starting time of o_2 and the completion time of o_1 are not only linked, but identical, so a gap value other than 0 is impossible and, hence, the passive energy consumption will always be null. In fact, using the "stack model" the energy consumption will be $P_p(r)\widehat{C_{max}}(\phi)+P_a(o_1,r)\widehat{p}(o_1,r)+P_a(o_2,r)\widehat{p}(o_2,r)=1(3,4,5)+2(1,2,3)+2(2,2,2)=(9,12,15)$. We can see that with the "stack model" we have avoided introducing the passive energy consumption proportional to the gap (0, 0, 2), which we have already identified as artificial uncertainty due to a bad model choice.

Using resource overlapping states to avoid calculating the size of idle periods under uncertainty is also underlying the definition of fuzzy total energy proposed in Pan et al. (2022), although in this work it is assumed that resources can be turned off individually.

2.4 Measures for comparing fuzzy schedules

We will be interested in comparing and measuring the goodness of fuzzy schedules beyond the ranking provided by $\leq_E$. To this end, we now propose different measures. These have been selected with two objectives in mind. First, as our main motivation to introduce an alternative energy model has been to avoid artificial uncertainty, these metrics must give evidence of this effect. Second, they need to clearly show the performance of the models in real scenarios.

2.4.1 Measures based on fuzzy values

When comparing two schedules ϕ and ϕ' based on their total energy values, $\widehat{E}(\phi)$ and $\widehat{E}(\phi')$, which are two TFNs, we may consider the following features of TFNs:

- Expected Value ($\mathbb{E}$). As explained in Sect. 2.1, the expected value can be used to rank TFNs, so the smaller the expected value of the total energy, the better the solution.
- Spread (S). For a TFN $\widehat{a}$ its spread (Ghrayeb 2003) is defined as:

$$S(\widehat{a})=a^3-a^1 \tag{16}$$

The spread is a measure of the uncertainty of the TFN. Although uncertainty is present in the problem, between two solutions, that with an energy value having a smaller spread (uncertainty) should be preferred.

- Modal value position (MVP). Given a TFN $\widehat{a}$ its modal value position is defined as:

$$MVP(\widehat{a})=\frac{(a^2-a^1)-(a^3-a^2)}{a^3-a^1} \tag{17}$$

This is a number in $[-1,1]$, negative when the TFN's modal value is on the left of the support's midpoint, positive when the modal value is on the right of the midpoint, and zero when the TFN is symmetric. It could somehow be seen as a measure of the skewness of the TFN. Obviously, the MVP of the schedule's energy depends on the processing times. However, if two schedules for the same problem have total energy values with different modal value position, the one with a MVP closer to 0 seems preferable in the sense that it is less biased, compared to the other schedule, which could be seen as accumulating uncertainty to the left (which could be seen as over-optimistic) or to the right (which could be seen as over-pessimistic). This measure is inspired by the ranking defined in Chen (1985), although the authors use it for a different objective.

2.4.2 Measures based on crisp realisations

According to the semantics of fuzzy schedules proposed in González Rodríguez et al. (2008a), a fuzzy schedule ϕ is an a-priori solution providing a possibility distribution (in the form of TFNs) for the starting times of all operations and for the resulting total energy consumption. When operations are actually executed according to the resource assignment and the order provided by ϕ, uncertainty is no longer present For each operation o, its executed processing time $p_{\phi}(o)$ is a crisp value in the support of the fuzzy processing time $\widehat{p}(o, \tau_{\phi}(o))$; it is a possible realisation of the fuzzy processing time. Also, the resulting total energy is a crisp value within the support of the fuzzy total energy consumption of ϕ. Hence, the executed schedule can be seen as the a-posteriori realisation of the original fuzzy schedule ϕ. The vector of possible realisations of fuzzy processing times, $\boldsymbol{\rho}$ where $\rho(o) = p_{\phi}(o)$, is also called a possible scenario.

Under these semantics, it is interesting to measure the quality of the a-priori fuzzy schedule relative to a possible scenario or a-posteriori realisation of processing times. For ϕ a feasible solution, let $\widehat{a}$ denote the fuzzy value of the objective function for ϕ; we propose the following two measures:

- Relative distance to the expected value ($RDEV$). Let $a_{\boldsymbol{\rho}}$ denote the value of the crisp objective function when the solution is evaluated on a possible scenario $\boldsymbol{\rho}$; then:

$$RDEV(\phi, \boldsymbol{\rho}) = \frac{a_{\boldsymbol{\rho}} - \mathbb{E}[\widehat{a}]}{\mathbb{E}[\widehat{a}]} \tag{18}$$

  If this value is negative, that is, the total energy consumption in the executed schedule is smaller than expected, it means that the a-priori schedule objective function is pessimistic. On the contrary, if $RDEV$ is positive, the a-priori schedule could be seen as optimistic. Overall, the smaller the absolute value of $RDEV$, the closer the predicted energy consumption of the solution is to real scenarios.

- Used uncertainty (UU). Let $\mathbf{S}$ be a set of possible scenarios and let $a_{\boldsymbol{\rho}}$ denote the value of the objective function in a scenario $\boldsymbol{\rho} \in \mathbf{S}$. Then:

$$UU(\phi, \mathbf{S}) = \frac{\max_{\boldsymbol{\rho} \in \mathbf{S}} a_{\boldsymbol{\rho}} - \min_{\boldsymbol{\rho} \in \mathbf{S}} a_{\boldsymbol{\rho}}}{a^3 - a^1} \tag{19}$$

  This gives a value in [0, 1] measuring the proportion of the support of the predicted fuzzy objective function that is covered with the possible scenarios in $\mathbf{S}$. When $\mathbf{S}$ is representative enough of all the possible scenarios, a value of UU close to 1 means that the range of values considered as obtainable by the TFN is actually achievable in real situations. Small values of UU would on the other hand mean that the fuzzy schedule incorporates artificial uncertainty and, hence, is less informative.

3 Neighbourhood for energy minimisation

To solve the problem, a competitive memetic algorithm was proposed in García Gómez et al. (2022). This is a hybrid algorithm that combines an evolutionary algorithm with tabu search. The idea is that by incorporating the problem-domain knowledge provided by the tabu search into the evolution process, the memetic algorithm benefits from the synergy between both search methods (Gendreau and Potvin 2019, Chapter 9). At each iteration, tabu search applies local transformations (called moves) to the current solution to generate a set of neighbouring solutions, so the search continues by moving to one of these neighbours. Obviously, defining an adequate neighbourhood structure is one of the most critical steps in designing any tabu search procedure (Gendreau and Potvin 2019, Chapter 2).

In this work, we define a new neighbourhood structure for the energy-aware flexible fuzzy job shop scheduling problem, based on the one proposed in García Gómez et al. (2022). A formal study will show its good theoretical behaviour, guaranteeing both feasibility and the connectivity property, that is, the possibility of reaching an optimal solution starting from any solution in the search space and performing only moves from this neighbourhood.

Given a solution $\phi = (\boldsymbol{\tau}_{\phi}, \boldsymbol{\sigma}_{\phi})$, we shall consider two kinds of moves depending on the component of ϕ that is modified. The current resource assignment $\boldsymbol{\tau}_{\phi}$ can be modified by a reassignment move, whereby an operation o is reassigned to a different resource $r \neq \tau_{\phi}(o)$, so the resulting resource assignment is denoted $\boldsymbol{\tau}_{(o,r)}$. The processing order vector $\boldsymbol{\sigma}_{\phi}$ can be modified with an insertion move, whereby an operation o is inserted in the i-th position of $\boldsymbol{\sigma}_{\phi}$. The resulting processing order is denoted $\boldsymbol{\sigma}_{(o,i)}$. A special kind of insertion in ϕ is the one that results in exchanging the position of two consecutive operations in a resource; it will be called a swap.

The distance between two solutions $\phi = (\boldsymbol{\tau}, \boldsymbol{\sigma})$ and $\phi' = (\boldsymbol{\tau}_{\phi'}, \boldsymbol{\sigma}_{\phi'})$, denoted as $d(\phi, \phi')$, is defined as the minimum number of reassignments and swaps that need to be applied to go from one solution to the other.

Not all the reassignment and insertion moves lead to feasible solutions and not all feasible moves are interesting in the sense that they may lead to solutions improving on total energy consumption. We will propose in Sects. 3.1

and 3.2 intermediate neighbourhood structures, based on one type of move, ensuring feasibility and aimed at reducing either passive or active energy consumption. They will then be combined in a global neighbourhood in Sect. 3.3.

We start by introducing some preliminary concepts and notation. Given a solution $\phi = (\boldsymbol{\tau}_\phi, \boldsymbol{\sigma}_\phi)$, for an operation $o \in \mathbf{O}$, let $JP(o)$ (resp. $JS(o)$) denote its predecessor (resp. successor) in its job, $RP_\phi(o)$ (resp. $RS_\phi(o)$) its predecessor (resp. successor) in its resource and $\widehat{p_\phi}(o) = \widehat{p}(o, \tau_\phi(o))$ the operation's processing time in the resource it is assigned to.

The head $\widehat{h_\phi}(o)$ of operation o is its earliest starting time according to ϕ, that is:

$$\widehat{h_\phi}(o) = \max\{\widehat{h_\phi}(JP(o)) + \widehat{p_\phi}(JP(o)), \widehat{h_\phi}(RP_\phi(o)) + \widehat{p_\phi}(RP_\phi(o))\} \tag{20}$$

where $\widehat{h_\phi}(JP(o)) + \widehat{p_\phi}(JP(o))$ (resp. $\widehat{h_\phi}(RP_\phi(o)) + \widehat{p_\phi}(RP_\phi(o))$) is taken to be (0, 0, 0) if o is the first operation in its job (resp. its resource).

The tail $\widehat{q_\phi}(o)$ of operation o is the time left once o has been processed until all other operations are completed:

$$\widehat{q_\phi}(o) = \max\{\widehat{q_\phi}(JS(o)) + \widehat{p_\phi}(JS(o)), \widehat{q_\phi}(RS_\phi(o)) + \widehat{p_\phi}(RS_\phi(o))\} \tag{21}$$

where $\widehat{q_\phi}(JS(o)) + \widehat{p_\phi}(JS(o))$ (resp. $\widehat{q_\phi}(RS_\phi(o)) + \widehat{p_\phi}(RS_\phi(o))$) is taken to be (0, 0, 0) if o is the last operation in its job (resp. its resource).

An operation o is said to be makespan-critical in a solution ϕ if there exists a component $i \in \{1, 2, 3\}$ of the fuzzy makespan such that $C^i_{max}(\phi) = (h_\phi(o) + p_\phi(o) + q_\phi(o))^i$. A makespan-critical block is a maximal sequence $\mathbf{b}$ of operations all requiring the same resource, with no pair of consecutive operations belonging to the same job and such that all operations in the block are makespan-critical for the same component of the fuzzy makespan, that is, there exists $i \in \{1, 2, 3\}$ such that for every operation $o \in \mathbf{b}$, $C^i_{max}(\phi) = (h_\phi(o) + p_\phi(o) + q_\phi(o))^i$. The set of all makespan-critical operations is denoted $\mathbf{T}_{C_{max}}(\phi)$ and the set of all makespan-critical blocks is $\mathbf{B}_{C_{max}}(\phi)$

3.1 Neighbourhoods aimed at reducing passive energy

In order to reduce passive energy consumption (in our case, in terms of $\leq_E$) it is necessary to reduce the time span when resources are active, that is, the makespan.

Now, let ϕ be a solution and let ϕ' be a feasible neighbouring solution obtained by an insertion or a reassignment move. Then ϕ' can improve in terms of makespan, (in the sense that $\widehat{C_{max}}(\phi') \leq_E \widehat{C_{max}}(\phi)$) and, thus, in passive energy consumption (that is, $\widehat{E_p}(\phi') \leq_E \widehat{E_p}(\phi)$), only if the move has involved a makespan-critical operation; the proof is a trivial generalisation to the case with flexibility of Proposition 3 in González Rodríguez et al. (2008b). Also, if a solution ϕ' is obtained by an insertion in ϕ that results in exchanging the position of two consecutive operations in a resource (called a swap), then ϕ' is always guaranteed to be feasible; the proof is a generalisation to the case with flexibility of Theorem 1 in González Rodríguez et al. (2008b). Furthermore, ϕ' can only improve in terms of makespan if the operations lie at the extreme of a makespan-critical block; here, the proof is a direct consequence of the analogous property proved for the fuzzy job shop in Theorem 2 of González Rodríguez et al. (2009).

These properties motivate the definition of two neighbourhood structures for passive energy. The first one consists in feasible reassignment moves of critical operations, aimed at redistributing the resources' workload. It is based on the neighbourhood proposed in González et al. (2013) for the deterministic flexible job shop with setup times. The desired effect of this neighbourhood is to reduce (in terms of $\leq_E$) the fuzzy makespan and, in consequence, the passive energy consumption, although it could increase active energy in the process.

Definition 1 *Makespan-critical operation resource reassignment neighbourhood*

$$\mathbf{N_{MCORR}}(\phi) = \{(\boldsymbol{\tau}_{(o,r)}, \boldsymbol{\sigma}) : o \in \mathbf{T}_{C_{max}}(\phi), r \in \mathbf{R}(o) \setminus \{\tau(o)\}\}$$

Notice that neighbourhood $\mathbf{N_{MCORR}}(\phi)$ contains only feasible solutions. Indeed, the selected operation o is reassigned to a resource in $\mathbf{R}(o)$ and this move does not alter $\boldsymbol{\sigma}_\phi$ in any way, so precedence and resource restrictions still hold.

It is possible to reduce the distance to an optimal solution using $\mathbf{N_{MCORR}}(\phi)$.

Lemma 1 *Let $\phi_{opt} = (\boldsymbol{\tau}_{\phi_{opt}}, \boldsymbol{\sigma}_{\phi_{opt}})$ be an optimal solution and let $\phi = (\boldsymbol{\tau}, \boldsymbol{\sigma})$ a non-optimal solution such that $\widehat{E_p}(\phi_{opt}) <_E \widehat{E_p}(\phi)$. If there exists a makespan-critical operation $o \in \mathbf{T}_{C_{max}}(\phi)$ such that o is assigned to different resources in ϕ and ϕ_{opt}, that is, $\tau_{\phi_{opt}}(o) \neq \tau_\phi(o)$, then there is a neighbouring solution $\phi' \in \mathbf{N_{MCORR}}(\phi)$ which is closer to ϕ_{opt} in the sense that $d(\phi', \phi_{opt}) < d(\phi, \phi_{opt})$.*

Proof Let $\phi' = (\boldsymbol{\tau}_{(o,r)}, \boldsymbol{\sigma})$ be the solution obtained by reassigning o to the resource $r = \tau_{\phi_{opt}}(o)$. Clearly, $\phi' \in \mathbf{N_{MCORR}}(\phi)$ and $d(\phi', \phi_{opt}) < d(\phi, \phi_{opt})$.

The second neighbourhood aimed at reducing the makespan and, hence, the passive energy $\widehat{E_p}$ in terms of $\leq_E$ is based on performing insertion moves. It is inspired by the neighbourhoods proposed by Dell' Amico and Trubian (1993) and Zhang et al. (2007) for the deterministic job shop problem. Its objective is, for a fixed resource assignment, to find an operation processing order that improves in terms of makespan. It consists in moving an operation forwards and backwards in a critical block as much as possible provided that the result still constitutes a feasible solution.

Let $\phi = (\boldsymbol{\tau}, \boldsymbol{\sigma})$ be a solution, let $\mathbf{b} \in \mathbf{B}_{C_{max}(\phi)}$ be a makespan-critical block for ϕ and let $o \in \mathbf{b}$ be an operation in that block. We define the minimal feasible insertion position of o in $\mathbf{b}$ as the earliest position where o can be inserted in $\boldsymbol{\sigma}$ so o is still in the block and the resulting solution maintains feasibility, that is:

$$\mathrm{minBI}(\phi, o) = \min\{i : \sigma(o_f) \leq i < \sigma(o), (\boldsymbol{\tau}, \boldsymbol{\sigma}_{(o,i)}) \text{ is feasible}\} \tag{22}$$

where o_f denotes the first operation in block $\mathbf{b}$ according to $\boldsymbol{\sigma}$. Analogously, we can define the maximal feasible insertion position of o in $\mathbf{b}$ as the latest position where o can be inserted in $\boldsymbol{\sigma}$ so o is still in the block and the resulting solution maintains feasibility, that is:

$$\mathrm{maxBI}(\phi, o) = \max\{i : \sigma(o) < i \leq \sigma(o_l), (\boldsymbol{\tau}, \boldsymbol{\sigma}_{(o,i)}) \text{ is feasible}\} \tag{23}$$

where o_l denotes the last operation in block $\mathbf{b}$ according to $\boldsymbol{\sigma}$. Having defined these indices, the second neighbourhood structure $\mathbf{N_{MCOI}}$ is as follows:

Definition 2 *Makespan-critical operation insertion neighbourhood.*

$$\mathbf{N_{MCOI}}(\phi) = \{(\boldsymbol{\tau}, \boldsymbol{\sigma}_{(o,i)}) : o \in \mathbf{T}_{C_{max}}(\phi), \\ i \in \{\mathrm{minBI}(\phi, o), \mathrm{maxBI}(\phi, o)\}\} \tag{24}$$

Clearly, by definition, $\mathbf{N_{MCOI}}(\phi)$ only contains feasible solutions.

We now study under which circumstances it is possible to get closer to an optimal solution using this neighbourhood.

Lemma 2 *Let $\phi_{opt} = (\boldsymbol{\tau}_{\phi_{opt}}, \boldsymbol{\sigma}_{\phi_{opt}})$ be an optimal solution and let $\phi = (\boldsymbol{\tau}, \boldsymbol{\sigma})$ be a non-optimal solution such that $\widehat{E_p}(\phi_{opt}) <_E \widehat{E_p}(\phi)$. If the resource assigned to every makespan-critical operation in ϕ coincides with the resource assigned to that operation in the optimal solution, i.e. $\boldsymbol{\tau}_{\phi_{opt}}(o) = \boldsymbol{\tau}_{\phi}(o), \forall o \in \mathbf{T}_{C_{max}}(\phi)$, then there exists a neighbouring solution $\phi' \in \mathbf{N_{MCOI}}(\phi)$ that is closer to ϕ_{opt} in the sense that $d(\phi', \phi_{opt}) < d(\phi, \phi_{opt})$.*

Proof To start with, notice that $\exists \mathbf{b} \in \mathbf{B}_{C_{max}(\phi)}$ and $\exists o \in \mathbf{b}$ such that in the optimal solution $\sigma_{\phi_{opt}}(o) < \sigma_{\phi_{opt}}(o_f)$ or $\sigma_{\phi_{opt}}(o) > \sigma_{\phi_{opt}}(o_l)$, with o_f and o_l the first and last tasks in the block $\mathbf{b}$. Indeed, suppose by contradiction that for every makespan-critical block in ϕ it holds that every operation $o \in \mathbf{b}$ is still between the first o_f and the last o_l operations of the block in the optimal solution, that is, $\sigma_{\phi_{opt}}(o_f) \leq \sigma_{\phi_{opt}}(o_o) \leq \sigma_{\phi_{opt}}(o_l)$. Then, it is possible to obtain $\sigma_{\phi_{opt}}$ from σ_ϕ by exchanging operations inside the critical blocks. However, swaps inside critical blocks do not improve the makespan, and, in consequence, $\widehat{C_{max}}(\phi) \leq_E \widehat{C_{max}}(\phi_{opt})$. But this contradicts the assumption that $\widehat{E_p}(\phi_{opt}) <_E \widehat{E_p}(\phi)$.

Without loss of generality, let us assume that $\exists \mathbf{b} \in \mathbf{B}_{C_{max}(\phi)}$ and $\exists o \in \mathbf{b}$ such that $\sigma_{\phi_{opt}}(o) < \sigma_{\phi_{opt}}(o_f)$. Let o^* be the first operation in $\mathbf{b}$ such that $\sigma_{\phi_{opt}}(o^*) < \sigma_{\phi_{opt}}(o_f)$ (notice that $o^* \neq o_f$)

We know that there exists at least one feasible insertion position $i < \sigma_\phi(o^*)$ for o^* in $\boldsymbol{\sigma}$ such that o^* is still in the block, because it is always feasible to swap the positions of two adjacent critical tasks and $o^* \neq o_f$. Therefore, we can take the earliest of these feasible insertion positions $i^* = \mathrm{minBI}(\phi, o)$ and define $\phi' = (\boldsymbol{\tau}, \boldsymbol{\sigma}_{(o^*, i^*)})$.

To prove that the $d(\phi', \phi_{opt}) < d(\phi, \phi_{opt})$ we show that every operation u in the block that was originally between the insertion position i^* and the original position of o^*, i.e., $i^* \leq \sigma_\phi(u) < \sigma_\phi(o^*)$, u is scheduled in the optimal solution after o^*, i.e. $\sigma_{\phi_{opt}}(o^*) < \sigma_{\phi_{opt}}(u)$. By contradiction, if this were not true, there would exist an operation $u \in \mathbf{b}$ such that $\sigma_{\phi_{opt}}(u) \leq \sigma_{\phi_{opt}}(o^*)$ and $i^* \leq \sigma_\phi(u) < \sigma_\phi(o^*)$. Given that $\sigma_{\phi_{opt}}(o^*) < \sigma_{\phi_{opt}}(o_f)$, this would mean that $\sigma_{\phi_{opt}}(u) < \sigma_{\phi_{opt}}(o_f)$ and $i^* \leq \sigma_\phi(u) < \sigma_\phi(o^*)$, but this contradicts the fact that o^* is the first operation in $\mathbf{b}$ that can be scheduled before o_f.

In practice, checking feasibility for every possible insertion position can be time consuming. This is also the case for the neighbourhood for deterministic job shop based on insertion proposed in Dell' Amico and Trubian (1993). Their solution to improve computational efficiency is to rely on a sufficient condition that guarantees that no unfeasible neighbours are generated. Although this results in discarding some feasible neighbours, they are non-improving ones and this happens with very small probability. We now generalise the sufficient condition to the case where both recirculation due to flexibility and fuzzy uncertainty are present.

Lemma 3 *Let $\phi = (\boldsymbol{\tau}, \boldsymbol{\sigma})$ be a feasible solution, let $\mathbf{b} \in \mathbf{B}_{C_{max}(\phi)}$ be a critical block with more than one operation so $\mathbf{b} = (\mathbf{b}_1, \mathbf{b}_2, o, \mathbf{b}_3)$ where $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$ are subsequences of operations with $\mathbf{b}_1$ and $\mathbf{b}_3$ possibly empty and $\mathbf{b}_2$ containing at least one element, and let $i = \min_{u \in \mathbf{b}_2} \sigma_\phi(u)$. Then, $\phi' = (\boldsymbol{\tau}, \boldsymbol{\sigma}_{(o,i)})$ is feasible if:*

$$\forall u \in \mathbf{b}_2 \ \chi(o) \neq \chi(u) \tag{25}$$

$$\forall u \in \mathbf{b}_2 \ \widehat{h}_\phi(JP(o)) \not\preceq \widehat{c}_\phi(JS(u)) \tag{26}$$

Proof The solution ϕ' that results from inserting o right before $\mathbf{b}_2$ is unfeasible if any of the two following conditions hold:

1. o is scheduled before a predecessor in its job.
2. There exists an subsequence of operations in σ going from the successor in the job $JS(u)$ of some operation u in $\mathbf{b}_2$ to $JP(o)$, the predecessor in the job of o.

It is clear that (25) guarantees that the unfeasibility condition 1 does not occur, since it avoids moving o before any other operation in its job.

Now, if (25) holds, i is an unfeasible insertion position only if condition 2 holds. In other words, there exists $u \in \mathbf{b}_2$ such that $JS(u)$ is completed before $JP(o)$ starts being processed, that is, $\widehat{c}_\phi(JS(u)) \preceq \widehat{h}_\phi(JP(o))$. However, if (26) holds, this is not possible, so it cannot be the case that i is an unfeasible insertion position.

The sufficient conditions for feasibility in Lemma 3 suggest redefining the minimal and maximal insertion position for a more efficient neighbourhood in the following way. Let $\phi = (\boldsymbol{\tau}, \boldsymbol{\sigma})$ be a solution, let $\mathbf{b} \in \mathbf{B}_{C_{max}(\phi)}$ be a makespan-critical block for ϕ and let $o \in \mathbf{b}$ be an operation in that block. We define the minimal surely feasible insertion position of o in $\mathbf{b}$ as the earliest position where o can be inserted in $\boldsymbol{\sigma}$ so o is still in the block and the sufficient conditions for feasibility (25) and (26) hold, that is:

$$\begin{aligned}&\mathrm{minBI}_r(\phi, o) = \min\{i : \sigma(o_f) \leq i < \sigma(o),\\ &(25) \text{ and } (26) \text{ hold for } (\boldsymbol{\tau}, \boldsymbol{\sigma}_{(o,i)})\}\end{aligned} \tag{27}$$

where o_f and o_l denote, respectively, the first and last operations in block $\mathbf{b}$ according to $\boldsymbol{\sigma}$. Analogously, we can define the maximal surely feasible insertion position of o in $\mathbf{o}$:

$$\begin{aligned}&\mathrm{maxBI}_r(\phi, o) = \max\{i : \sigma(o) < i \leq \sigma(o_l),\\ &(25) \text{ and } (26) \text{ hold for } (\boldsymbol{\tau}, \boldsymbol{\sigma}_{(o,i)})\}\end{aligned} \tag{28}$$

These new indices allow us to define the reduced neighbourhood structure $\mathbf{N}_{\mathbf{MCOI_r}}$ as follows:

Definition 3 *Reduced makespan-critical operation insertion neighbourhood*

$$\begin{aligned}&\mathbf{N}_{\mathbf{MCOI_r}}(\phi) = \{(\boldsymbol{\tau}, \boldsymbol{\sigma}_{(o,i)}) : o \in \mathbf{T}_{\mathbf{C_{max}}}(\phi),\\ &i \in \{\mathrm{minBI}_r(\phi, o), \mathrm{maxBI}_r(\phi, o)\}\}\end{aligned} \tag{29}$$

The property from Lemma 2 still holds for this reduced variant.

Lemma 4 *Let $\phi_{opt} = (\boldsymbol{\tau}_{\phi_{opt}}, \boldsymbol{\sigma}_{\phi_{opt}})$ be an optimal solution and let $\phi = (\boldsymbol{\tau}, \boldsymbol{\sigma})$ be a non-optimal solution such that $\widehat{E_p}(\phi_{opt}) <_E \widehat{E_p}(\phi)$. If the resource assigned to every makespan-critical operation in ϕ coincides with the resource assigned to that operation in the optimal solution, i.e. $\boldsymbol{\tau}_{\phi_{opt}}(o) = \boldsymbol{\tau}_\phi(o), \forall o \in \mathbf{T}_{C_{max}}(\phi)$, then there exists a neighbouring solution $\phi' \in \mathbf{N}_{\mathbf{MCOI_r}}(\phi)$ that is closer to ϕ_{opt} in the sense that $d(\phi', \phi_{opt}) < d(\phi, \phi_{opt})$.*

Proof It suffices to show that the sufficient conditions for feasibility (25) and (26) always allow to swap two adjacent critical operations u and o in a critical block $\mathbf{b} \in \mathbf{B}_{\mathbf{C_{max}}}(\phi)$; then, the proof of Lemma 2 is applicable to the reduced neighbourhood.

The first condition (25) holds trivially for swaps because by definition a critical block cannot contain two consecutive operations of the same job.

Also, since u and o are consecutive and belong to the same block, we have that

$$\widehat{c}_\phi(JP(o)) \not\preceq \widehat{c}_\phi(u)$$

because, if this were not the case, u would not affect the starting time of o (contradicting the definition of critical block). This together with:

$$\widehat{h}_\phi(JP(o)) \preceq \widehat{c}_\phi(JP(o)) \text{ and } \widehat{c}_\phi(u) \preceq \widehat{c}_\phi(JS(u))$$

yields

$$\widehat{h}_\phi(JP(o)) \not\preceq \widehat{c}_\phi(JS(u))$$

so the sufficient condition (26) holds.

3.2 Neighbourhood aimed at reducing active energy

Active energy consumption can only be reduced in terms of $\leq_E$ by moving operations to a more energy-efficient resource. This is the rationale behind the last intermediate neighbourhood, which performs reassignment moves to relocate non-critical operations to resources where they incur in a lower active energy consumption.

Definition 4 *Operation power-efficient resource reassignment neighbourhood*

$$\mathbf{N_{OPERR}}(\phi) = \{(\boldsymbol{\tau}_{(o,r)}, \boldsymbol{\sigma}) : o \notin \mathbf{T}_{C_{max}}(\phi), r \in \mathbf{R}(o) \setminus \{\tau(o)\}, \widehat{E_a}(o,r) <_E \widehat{E_a}(o, \tau(o))\} \tag{30}$$

Notice that this neighbourhood contains only feasible solutions, since the only change with respect to the feasible solution ϕ is the reassignment of an operation o to a resource in $\mathbf{R}(o)$.

It is also possible to prove that, under certain circumstances, there is a move corresponding to $\mathbf{N_{OPERR}}(\phi)$ which brings ϕ closer to a given optimal solution.

Lemma 5 *Let $\phi_{opt} = (\boldsymbol{\tau}_{\phi_{opt}}, \boldsymbol{\sigma}_{\phi_{opt}})$ be an optimal solution and let $\phi = (\boldsymbol{\tau}, \boldsymbol{\sigma})$ be a non-optimal solution such that all makespan-critical operations are already assigned to the same resource as in ϕ_{opt}, that is, $\forall o \in \mathbf{T}_{C_{max}}(\phi)\ \tau_\phi(o) = \tau_{\phi_{opt}}(o)$, and such that $\widehat{E_a}(\phi_{opt}) <_E \widehat{E_a}(\phi)$. Then there exists a neighbouring solution $\phi' \in \mathbf{N_{OPERR}}(\phi)$ that is closer to ϕ_{opt} in the sense that $d(\phi', \phi_{opt}) < d(\phi, \phi_{opt})$.*

Proof First we notice that there exists at least a non-critical operation $o \notin \mathbf{T}_{C_{max}}(\phi)$ that is assigned to different resources in ϕ and ϕ_{opt}. Indeed, given that all critical operations are already in the same resource as in ϕ_{opt}, if all non-critical operations were also in the same resource as in ϕ_{opt} then it would not be possible that $\widehat{E_a}(\phi) <_E \widehat{E_a}(\phi_{opt})$.

Take o^* one of the non-critical operations assigned to different resources in ϕ and ϕ_{opt} and take $\phi' = (\boldsymbol{\tau}(o^*, r), \boldsymbol{\sigma})$ with $r = \tau_{\phi_{opt}}(o^*)$ the solution that results from ϕ by reassignment of o^* to the resource where it is processed in the optimal solution. Clearly, $\phi' \in \mathbf{N_{OPERR}}(\phi)$ and $d(\phi', \phi_{opt}) < d(\phi, \phi_{opt})$.

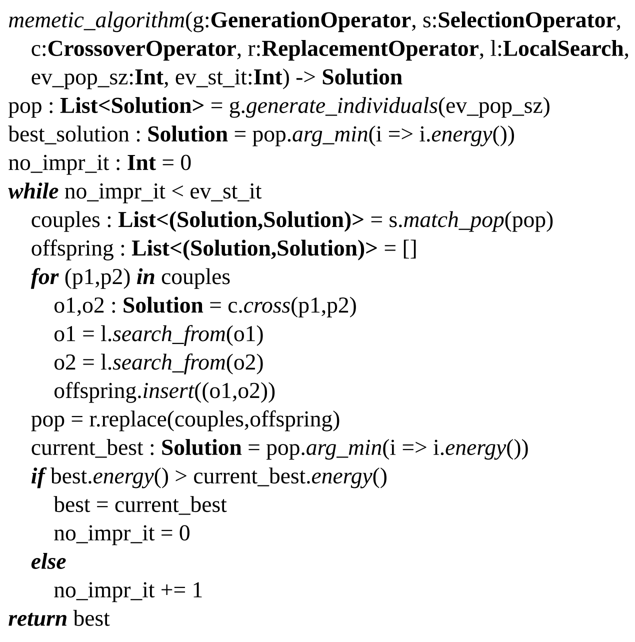

```
memetic_algorithm(g:GenerationOperator, s:SelectionOperator,
  c:CrossoverOperator, r:ReplacementOperator, l:LocalSearch,
  ev_pop_sz:Int, ev_st_it:Int) -> Solution
pop : List<Solution> = g.generate_individuals(ev_pop_sz)
best_solution : Solution = pop.arg_min(i => i.energy())
no_impr_it : Int = 0
while no_impr_it < ev_st_it
  couples : List<(Solution,Solution)> = s.match_pop(pop)
  offspring : List<(Solution,Solution)> = []
  for (p1,p2) in couples
    o1,o2 : Solution = c.cross(p1,p2)
    o1 = l.search_from(o1)
    o2 = l.search_from(o2)
    offspring.insert((o1,o2))
  pop = r.replace(couples,offspring)
  current_best : Solution = pop.arg_min(i => i.energy())
  if best.energy() > current_best.energy()
    best = current_best
    no_impr_it = 0
  else
    no_impr_it += 1
return best
```

Fig. 1 Memetic algorithm pseudocode

3.3 Neighbourhood aimed at reducing total energy consumption

We are now in condition to define a neighbourhood structure $\mathbf{N}(\phi)$ as the union of the three intermediate neighbourhoods $\mathbf{N_{MCOI_r}}(\phi)$, $\mathbf{N_{MCORR}}(\phi)$ and $\mathbf{N_{OPERR}}(\phi)$. Notice that, although $\mathbf{N_{MCORR}}(\phi)$ aims at reducing the passive energy E_p and $\mathbf{N_{OPERR}}(\phi)$, at reducing the active energy E_a, they can also affect the other type of energy because they are based on reassignment moves. On the other hand, $\mathbf{N_{MCOI_r}}(\phi)$ can only alter the passive energy consumption.

Definition 5 *Neighbourhood for Total Energy Consumption*

$$\mathbf{N}(\phi) = \mathbf{N_{MCORR}}(\phi) \cup \mathbf{N_{MCOI_r}}(\phi) \cup \mathbf{N_{OPERR}}(\phi) \tag{31}$$

Given that $\mathbf{N}(\phi)$ is the union of feasible neighbourhoods, it must also be feasible. Also, we can show that the connectivity property holds for this neighbourhood.

Theorem 1 *The neighbourhood for total energy consumption verifies the connectivity property, that is, starting from any non-optimal feasible solution ϕ it is possible to build a finite sequence of transitions of $\mathbf{N}$ leading to an optimal solution.*

Proof Let ϕ_{opt} be an arbitrary optimal solution. We need to find a finite sequence of solutions $\phi_0, \ldots, \phi_n$ such that $\phi_0 = \phi$, $\phi_{k+1} \in \mathbf{N}(\phi_k)$ for every $0 \leq k < n$ and $\mathbb{E}[\widehat{E}(\phi_{opt})] = \mathbb{E}[\widehat{E}(\phi_n)]$. To this end, for $k \geq 0$ we take $\phi_{k+1} \in \mathbf{N}(\phi_k)$ according to the following schema:

- If $\widehat{E_p}(\phi_k) >_E \widehat{E_p}(\phi_{opt})$ and $\forall o \in \mathbf{T}_{C_{max}}(\phi_k)$ $\tau_{\phi_{opt}}(o) \neq \tau_{\phi_k}(o)$, then, by Lemma 1 there exists $\phi' \in \mathbf{N_{MCORR}}(\phi_k)$ which is closer to ϕ_{opt}; take $\phi_{k+1} = \phi'$.
- Else if $\widehat{E_p}(\phi_k) >_E \widehat{E_p}(\phi_{opt})$, then, by Lemma 4 there exists $\phi' \in \mathbf{N_{MCOI_r}}(\phi_k)$ which is closer to ϕ_{opt}; take $\phi_{k+1} = \phi'$.
- Else if $\widehat{E_a}(\phi_k) >_E \widehat{E_a}(\phi_{opt})$, then by Lemma 5 there exists $\phi' \in \mathbf{N_{OPERR}}(\phi_k)$ which is closer to ϕ_{opt}; take $\phi_{k+1} = \phi'$.

First, notice that if none of the above conditions holds for a given ϕ_k, this means that we have reached an optimal solution. Indeed, if none of the above hold, it must be case that $\widehat{E_p}(\phi_k) \leq_E \widehat{E_p}(\phi_{opt})$ and $\widehat{E_a}(\phi_k) \leq_E \widehat{E_a}(\phi_{opt})$. Due to the linearity of the expected value and the definition of $\widehat{E}(\phi_k)$ as the sum of $\widehat{E}_p(\phi_k)$ and $\widehat{E}_a(\phi_k)$, this means that $\mathbb{E}[\widehat{E}(\phi_k)] \leq \mathbb{E}[\widehat{E}(\phi_{opt}]$. But ϕ_{opt} is an optimal solution according to $\leq_E$, so $\mathbb{E}[\widehat{E}(\phi_{opt}) \leq \mathbb{E}(\widehat{E}(\phi_k)]$. In consequence, $\mathbb{E}[\widehat{E}(\phi_k)] = \mathbb{E}[\widehat{E}(\phi_{opt}]$, that is, ϕ_k is an optimal solution (even if it may not be ϕ_{opt}).

Second, we can be sure that the sequence will finish in a finite number of steps. Indeed, the distance $d(\phi_0, \phi_{opt})$ between the initial and the optimal solution ϕ_{opt} is a finite number. Now, at step k, if none of the conditions to select ϕ_{k+1} hold, the sequence finishes and ϕ_k is an optimal solution. Otherwise, a new solution ϕ_{k+1} is chosen from $\mathbf{N}(\phi_k)$ according to one of Lemmas 1, 4 or 5, so it always holds that $d(\phi_{k+1}, \phi_{opt}) < d(\phi_k, \phi_{opt})$. That is, at each step, either we find an optimal solution or the distance between the next solution in the sequence and the optimal solution decreases. Hence, if an optimal solution has not been found earlier, after at most $n = d(\phi_0, \phi_{opt})$ steps it must be the case that $d(\phi_n, \phi_{opt}) = 0$, that is, $\phi_n = \phi_{opt}$ is optimal and the sequence finishes.

3.4 The memetic algorithm

As mentioned at the beginning of this section, the neighbourhood structure $\mathbf{N}(\phi)$ is used in a tabu search procedure which is combined with an evolutionary algorithm to yield the memetic algorithm from García Gómez et al. (2022). The pseudocode of the resulting method can be seen in Fig. 1

The evolutionary algorithm is composed of a population of solutions that, in each iteration, is replaced by a new one obtained by combining its individuals. To do so, individuals are randomly matched, giving everyone an equal chance to reproduce, and each pair is combined by means of a crossover operator that generates two offspring. The new population is generated by a tournament such that the best two individuals are chosen from each pair of parents and their two offspring. To encode the solutions, we use the tuple of sequences $(\boldsymbol{\tau}, \boldsymbol{\sigma})$ and to decode them, we assign to each operation the earliest starting time such that the order defined by $\boldsymbol{\sigma}$ is not altered. To ensure enough diversity the initial population is generated randomly. A key component here is the crossover operator. We use the extension of the Generalized Order Crossover (GOX) proposed in García Gómez et al. (2021). Because we use a tournament, this operator is applied unconditionally. We do not make use of any mutation operators because it is incorporated in the local search explained below.

The other component of the memetic algorithm is the local search. In our proposal, all offspring generated at each iteration of the evolutionary algorithm are improved using tabu search with $\mathbf{N}(\phi)$ before the tournament is applied. Tabu search is a local search algorithm that keeps a memory structure, called tabu list, where it stores a trace of the recently visited search space (Gendreau and Potvin 2019, Chapter 2). In particular, to avoid undoing recently made moves, we store in the tabu list the inverse of the moves performed to obtain the neighbours. Thus, at each iteration the tabu search selects the best neighbour in the neighbourhood that is not obtained with a tabu move. Our tabu list has a dynamic size, similar to the one introduced in Dell' Amico and Trubian (1993), so the size of the list can vary between a lower and an upper bound. When the selected neighbour is worse (resp. better) than the current solution and the upper (resp. lower) bound has not been reached, the list's size increases (resp. decreases) in one unit. If the selected neighbour is the best solution found so far, the list is cleared; this is similar to restarting the search from this solution. We also incorporate an aspiration criterion, so a tabu move can be executed if it improves the best solution found up to this moment. In the rare situation that all neighbours are tabu, we choose the best one, clear the tabu list and slightly change its bounds by picking a random number within a given range.

As neighbour evaluation is the most time-consuming part of the local search, we make use of a filtering mechanism to discard uninteresting solutions. This mechanism consists in evaluating the neighbours following the order defined by a lower bound, and stopping as soon as this lower bound is bigger than the exact value of any of the already evaluated solutions. Here, we adopt the lower bound for the total energy consumption proposed in García Gómez et al. (2023).

4 Experimental results

The objective of the experimental study is twofold. On the one hand, to empirically measure the benefits of the stack model over the gaps model for fuzzy total energy computation. On the other hand, to compare our algorithm with the state of the art to check the behaviour of the proposed neighbourhood.

As test bed we will use the 12 instances from the literature (García Gómez et al. 2023). Namely, 07a, 08a, 09a, 10a, 11a, and 12a with 15 jobs and 8 resources (denoted 15×8) and 13a, 14a, 15a, 16a, 17a and 18a sized 20×10. They have three increasing flexibility levels, being instances 07a, 10a, 13a and 16a those with low flexibility and 09a, 12a, 15a and 18a those with the highest flexibility.

Table 2 Parameters of the memetic algorithm: `ev_pop_sz` is the size of the population; `ev_st_it` represents the number of iterations without improvement after which the search will be halted; `tb_lst_lbr` and `tb_lst_ubr` correspond to the range of lower and upper bound values for the size of the tabu list; `ls_st_it` represents the number of iterations after which a run of the tabu search will be stopped

ev_pop_sz	ev_st_it	tb_lst_lbr	tb_lst_ubr	ls_st_it
mn	$mn/2$	$[mn/2, mn]$	$[2mn, 3mn]$	mn

In parameter tuning, we have defined the parameters of the algorithm in terms of the instances' characteristics. First, we consider the flexibility of the instances in the form:

$$flx = \frac{\sum_{o\in\mathbf{O}} |\mathbf{R}(o)|}{|\mathbf{O}||\mathbf{R}|} \tag{32}$$

We also take into account the ratio $|\mathbf{O}|/|\mathbf{R}|$ because having high flexibility but few resources will make it harder to find a good schedule. We combine both flexibility and ratio operations to resources in the following single value:

$$mn = 2\lfloor\frac{(1+flx)|\mathbf{O}|}{2|\mathbf{R}|}\rceil \tag{33}$$

where $\lfloor\rceil$ represents the nearest integer function. This value is used to parameterise the algorithm as summarised in Table 2.

All results have been obtained in a Linux machine with two Intel Xeon Gold 6132 processors without Hyper-Threading (14 cores/14 threads) and 128GB RAM using a parallelized implementation of the algorithm in Rust. The source code together with detailed results and benchmark instances can be found at https://pablogarciagomez.com/research.

4.1 Models comparison

In this section, we compare the stack model with the gaps model using the metrics defined in Sect. 2.4. We have claimed that the stack model is better because it does not add any extra uncertainty due to the avoidance of the subtraction operation between fuzzy numbers needed to compute the idle times (gaps) of resources. Although the reasoning behind this fact has already been detailed, this can mean nothing to the search process. It could even happen that a more uncertain model, when considering real and deterministic scenarios, had a better behaviour. For this reason, we propose two types of comparisons: a priori and a posteriori.

A priori comparisons are conducted on random solutions with the aim of measuring the advantages of the model without the possible influence of the solver. A posteriori comparisons, as the name suggests, compare the performance of the models after being used on a search algorithm, showing how good they are at guiding the search. It is important to highlight that a posteriori comparisons check the performance within a specific algorithm, so results may differ when using the models in another algorithm.

Specifically, for a priori comparisons we use 30 random solutions, obtained with the same procedure employed to generate the initial population of the memetic algorithm, probably the naivest algorithm to generate random solutions because it only supposes that operations are started as soon as possible. Conversely, in a posteriori comparisons, we use two different sets of solutions, obtained by independently using each model in our algorithm. We can expect that the algorithm obtains solutions "adapted" to the inner workings of the models so results may differ with respect to a priori comparisons.

Inspired by the metrics, we start comparing the models by means of the fuzzy values of the objective function, and then we compare the performance of the solutions when they are tested on crisp scenarios. For the latter comparisons, we will use a unique set of 1000 crisp possible scenarios $\mathbf{S}$ derived from the fuzzy instances for both models. These samples are generated by taking a random value for the processing time of operations in resources using a uniform distribution within the range of the TFNs. Arguments for choosing such uniform distribution can be found in Palacios et al. (2017) and references therein.

To illustrate the study, we mainly use charts and figures. Nevertheless, we provide as supplementary material detailed tables with all the data obtained in the experiments.

4.1.1 Comparisons based on fuzzy values

Figure 2 illustrates the better performance of the stack model compared to the gaps model in terms of the expected value $\mathbb{E}[E]$ of the fuzzy total energy consumption, both in the a priori (31% lower on average) and a posteriori (24% lower) comparisons. In Fig. 2b, we can also appreciate some dependency with the flexibility of instances. This can be better seen in Fig. 3 where the improvement of a posteriori mean $\mathbb{E}[E]$ values w.r.t. a priori ones is depicted for both models. It is clear that, the bigger flexibility the

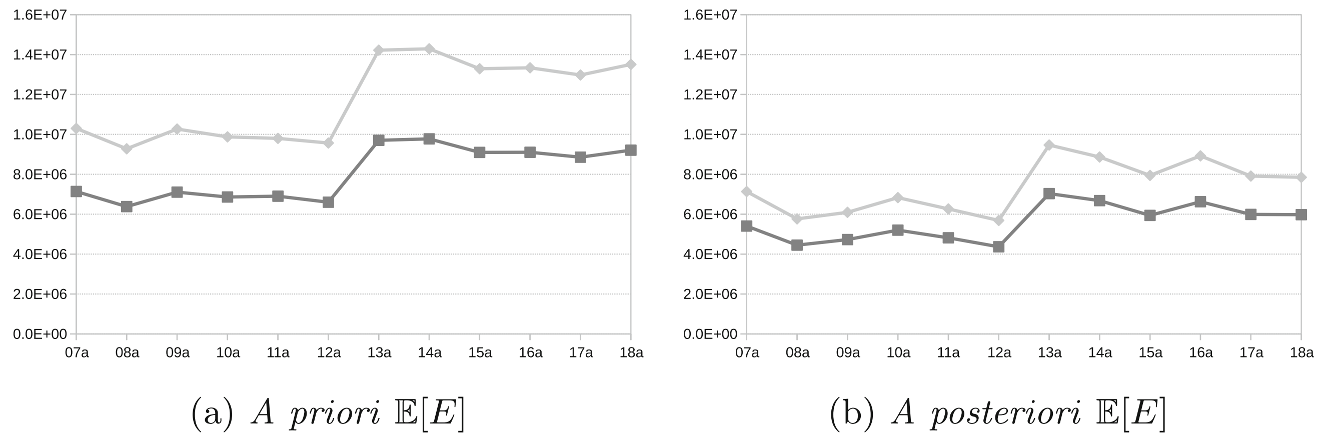


Fig. 2 $\mathbb{E}[E]$ comparisons. The dark grey values represent the stack model and the light grey values, the gaps model

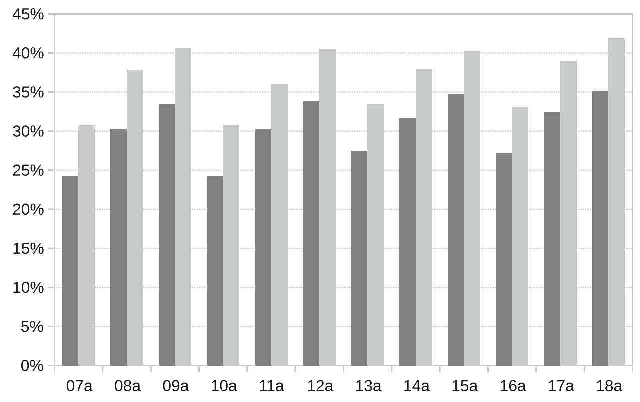


Fig. 3 Comparison between a priori and a posteriori expected values. The dark grey corresponds to the stack model and the light grey to the gaps model

greater the improvement and that this behaviour is produced by the algorithm, because in a priori comparisons the expected value is similar across instances with the same size. This shows that the neighbourhoods designed to cope with flexibility are doing their work. Moreover, comparing Fig. 2a with Fig. 2b we can appreciate that the gaps model is having more improvement. Although it may be seen as an advantage in favour of the gaps model, it is not, because in Fig. 2b its expected value is still higher. The explanation could be that in random solutions the results are much worse for the gaps model, and so when embedding the model in an algorithm there is much more margin for improvement.

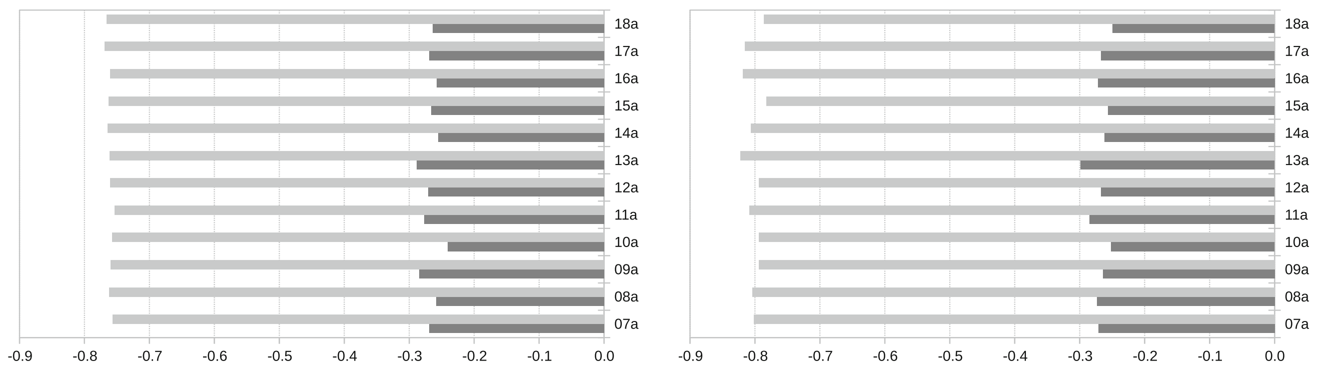
Regarding *MVP*, negative values in Fig. 4 indicate that in both models and both comparisons, the modal value is deviated to the left side of the TFNs, that is, the uncertainty is larger in upper values. Little changes can be observed on *MVP* values in the a posteriori evaluation. In the gaps model, the larger uncertainty can be explained by the truncation of negative values needed to avoid negative, so inconsistent, values in the fuzzy subtraction used to compute idle times of resources. It seems that this effect remains throughout the search.

Regarding the spread S, the TFNs obtained across a priori solutions with the gaps model are on average 11.5 times wider than the TFNs obtained with the stack model (with a 0.33 standard deviation). The difference in the a posteriori comparisons is reduced, so the gaps model results in TFNs which are on average 6.5 times wider than the ones found with the stack model (with 0.58 standard deviation). Even being large, the reduction of the spread is significant; this reflects that our algorithm has been able to overcome the shortcomings of the gaps model to obtain solutions with a more reasonable spread. This huge difference and the reduction of the spread are depicted in Fig. 5 for instance 18a, with the other instances behaving similarly. Be aware of the differences in scale in the axis. As in both models it is guaranteed that the energy values of crisp scenarios are within the interval defined by the TFNs, having wider TNFs means that the model is providing a larger spread than necessary.

4.1.2 Comparisons based on crisp scenarios

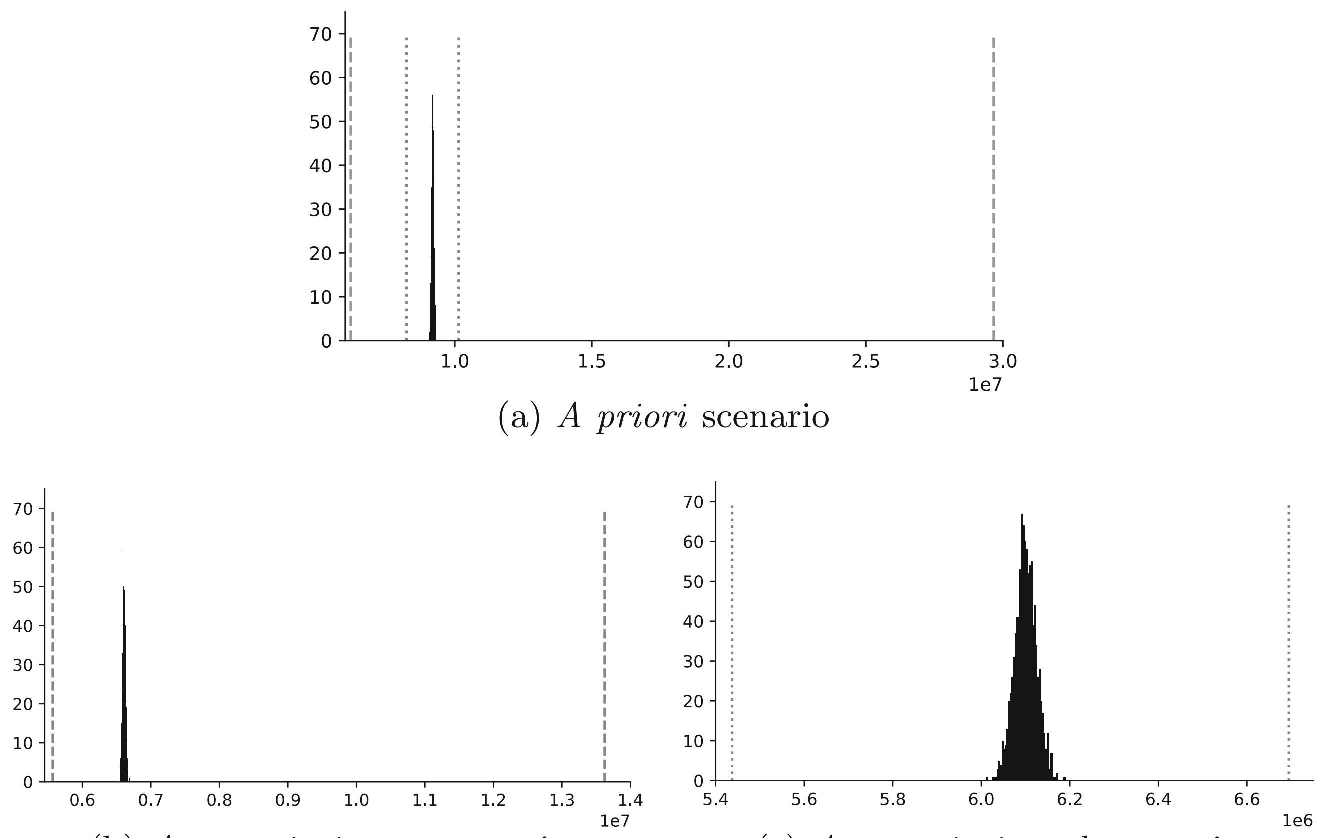
Regarding the used uncertainty, although the better performance of the stack model was expected by the previous results, *UU* values depicted in Fig. 6 illustrate that only 1.5% (a priori) and 3.6% (a posteriori) of the spread are mostly used for the gaps model solutions. This value is near 23% in the solutions a posteriori with the stack model. Notice that *UU* values are highly dependent on the way crisp scenarios are generated. Both models improve in this area in a posteriori results with respect to the a priori ones. We can also appreciate a clear increase in *UU* values when the flexibility on the instances also increases.

Figure 5 also allows a better display of the used uncertainty. There is only one figure for the a priori comparison because the set of random solutions used for a



(a) *A priori* (b) *A posteriori*

Fig. 4 *MVP* comparison. The dark grey corresponds to the stack model and the light grey to the gaps model



(a) *A priori* scenario

(b) *A posteriori* gaps scenario (c) *A posteriori* stack scenario

Fig. 5 Comparison of the spread and the used uncertainty in instance 18a. The dark grey dotted vertical lines represent the support of the fuzzy energy consumption in the stack model and the light grey dashed lines represent the same for the gaps model

priori comparisons is the same for both models and they are equivalent in the deterministic setting. It is very clear how the possible scenarios are inside the support of the TFN both for the gaps and the stack model but the stack model supposes a major reduction in uncertainty, especially in the upper range. Moreover, we can appreciate how the possible scenarios are centred inside the support in the case of the stack model but this does not happen for the gaps model. The conclusions are the same in the a posteriori situation, but we can see more clearly the differences between the models.

Regarding the relative distance to the expected value, *RDEV*, Fig. 7 clearly shows that, in absolute values of *RDEV*, in the stack model the deterministic values are

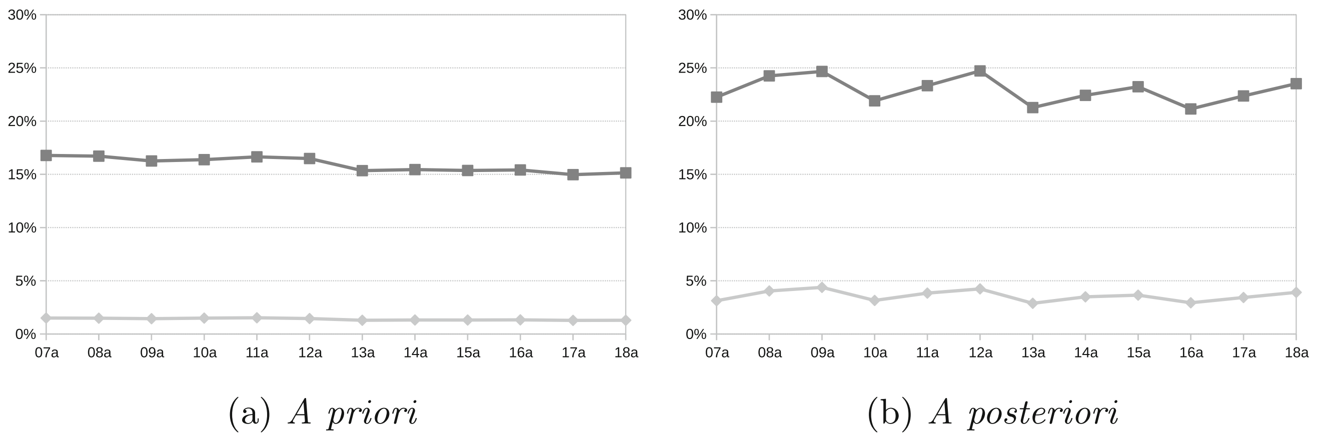


Fig. 6 *UU* comparison. The dark grey corresponds to the stack model and the light grey to the gaps model

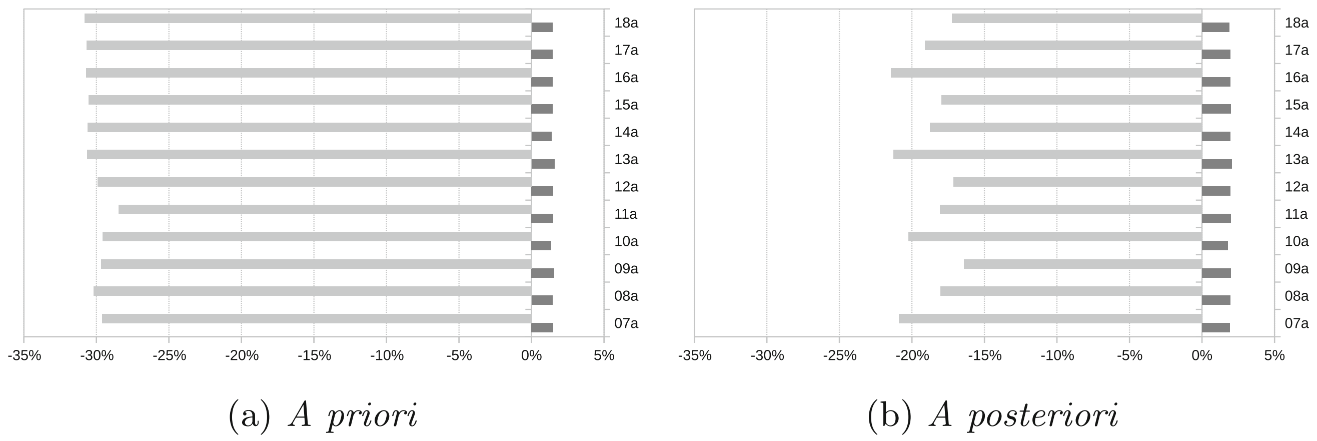


Fig. 7 *RDEV* comparison. The dark grey corresponds to the stack model and the light grey to the gaps model

much closer to the expected value: 30% versus 1.5% in a priori and 18.9% versus 1.9% in a posteriori. Negative values of the gaps model are again a consequence of the truncation introduced in the gaps model, which only shrinks the left side of the triangles deviating its expected value.

In addition to the above measures, we also check which model is better considering the average value for the total energy consumption in the crisp scenarios. This analysis could not be done a priori because the set of randomly generated solutions is the same for both methods and as they are equivalent in a crisp setting, values are the same. Results are shown in Fig. 8. It is clear that the stack model obtains better results, as its boxplots are significantly below the gaps model ones. Moreover, the distance between models is increased with the flexibility of the instances. It is also clear that the gaps model is more stable because its boxes are narrower. The conclusion is that the stack model helps the algorithm to guide the search more effectively.

4.1.3 Comparisons based on CPU times

Moreover, the stack model has another major advantage, its speed, as can be checked in Fig. 9 that compares the time consumed by the algorithm with each model. As we explained before, neighbour evaluation in the local search is very time-consuming and for this reason we use a lower bound as a filter to discard bad solutions without evaluating them. However, in the case of the gaps model there is no such lower bound in the literature and it is not trivial to find one so it is making use of the more expensive exact evaluation and this directly translated into a higher use of CPU time.

4.2 Comparison with the state of the art

To compare our algorithm with the state of the art, the experiments have been made using the same model of fuzzy total energy, the same base implementation and have been run on the same machine, so they are comparable both in terms of solution quality and in time.

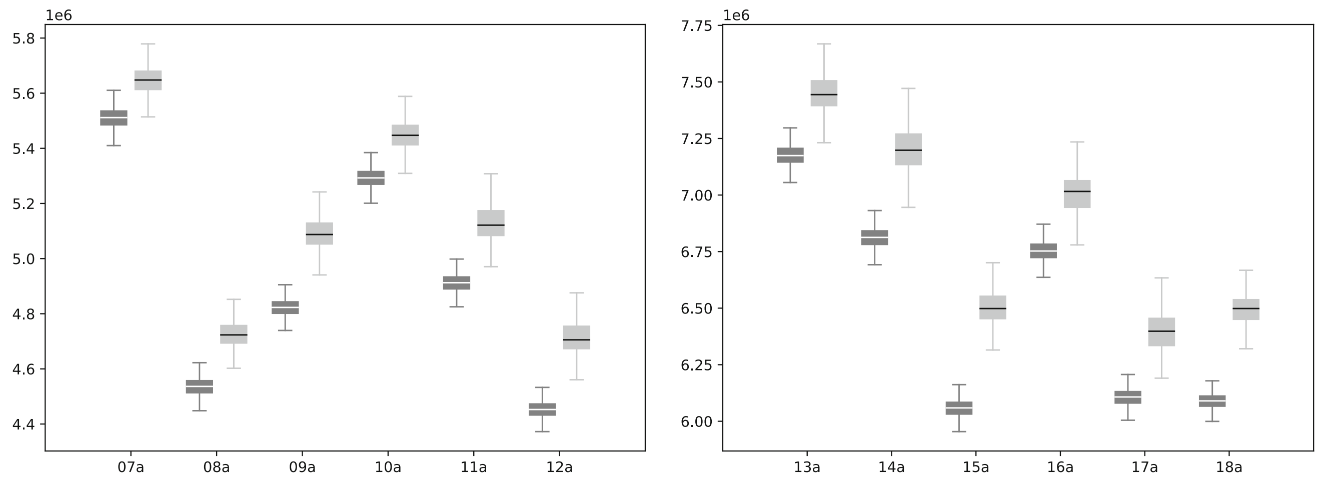


Fig. 8 Comparison of the total energy consumption in crisp scenarios. The dark grey corresponds to the stack model and the light grey to the gaps model

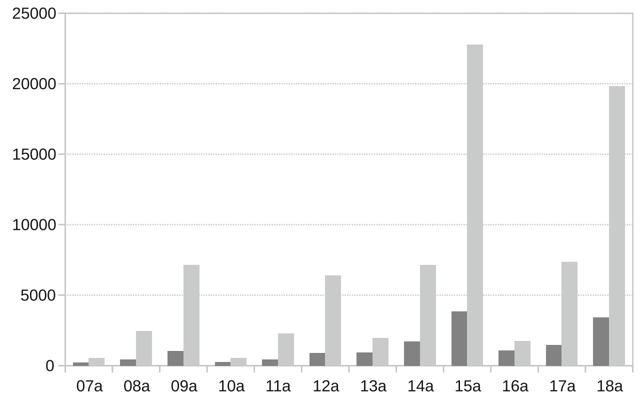


Fig. 9 CPU time (s). The dark grey corresponds to the stack model and the light grey to the gaps model.

In table Table 3 we can see a comparison between our algorithm (named RMA from now on), SMA from García Gómez et al. (2022) that proposes an algorithm very similar to ours but that makes use of a non-connective neighbourhood, and EMA from García Gómez et al. (2023) which, to the best of our knowledge, represents the state of the art. The first column contains the instance name. The group of next three columns contain the mean and best values over 30 executions for the total energy consumption and the CPU time employed on average per run of our proposal RMA. The next group of three columns contains the relative improvement of RMA with respect to SMA for the mean and best values for the energy and the CPU time. Finally, the last group of tree columns contain the same value with respect to EMA. In Fig. 10 we show boxplots for the values of the energy consumption in the 30 executions that better illustrate the differences between the algorithms.

Compared to SMA our algorithm gives results that quality-wise are not significantly different according to paired t-test (results are normally distributed according to Shapiro-Wilk test). Differences between the two algorithms concentrate on instances with the lower flexibility (in which SMA seems to work better) and those with higher flexibility (in which RMA gives better results). However, our algorithm uses 46% less CPU time to obtain the solutions, in consequence RMA can be considered better than SMA. Compared to EMA, it is behind in almost all cases in what quality of solutions is concerned, however, results are in any case very close as shown in the boxplots. Nevertheless, again, we have one major advantage over it, a reduction of CPU time of almost a 30%. Combining both results RMA can be considered competitive. It is worth noting that EMA is much more sophisticated than RMA as it has mechanisms to better balance the exploration and exploitation as well as adaptive parameters, so surpassing it in the quality of solutions was not expected. As conclusion, our proposal seems to be better suited to CPU-constrained applications.

5 Conclusions

In this paper we have worked on a model to minimise the energy consumption in a flexible job shop in an uncertain setting. This is a very important scheduling problem due to its relevance in manufacturing and engineering problems, in special when considering uncertainty, as it constitutes one of the most important sources of disruptions when applying theoretical models in real life. Our modelling of the processing times using fuzzy numbers allows the model to be easy to understand, while at the same time gives algorithms the right amount of information to achieve good solutions in a fair amount of time.

Finding a good model is not an easy task, and cannot be done without analysing its advantages and possible

Table 3 Comparison with the state of the art

Instance	RMA			Imp. w.r.t. SMA(%)			Imp. w.r.t. EMA(%)		
	Mean	Best	Time	Mean	Best	Time	Mean	Best	Time
07a	5406215.23	5342610.25	234.55	−1.82	−1.13	71.11	−0.66	−0.42	66.10
08a	4448751.69	4411737.00	436.93	−0.03	0.24	63.40	−0.12	−0.09	25.21
09a	4728223.43	4693336.75	1034.42	0.61	0.49	61.53	−0.25	−0.34	28.72
10a	5200786.87	5153733.75	251.85	−1.69	−1.33	62.46	−0.61	−1.03	66.26
11a	4816362.64	4764942.75	440.91	−0.05	0.33	64.07	−0.25	−0.22	43.75
12a	4367695.83	4328756.75	912.57	0.64	0.71	58.71	−0.24	0.02	5.79
13a	7032932.70	6976623.75	943.23	−0.73	−0.37	28.78	−0.25	−0.14	46.38
14a	6681278.33	6614463.50	1713.58	0.52	0.83	35.10	−0.19	−0.04	19.07
15a	5940752.25	5887899.00	3855.72	1.54	1.52	27.97	−0.13	−0.01	−13.82
16a	6625853.63	6581874.50	1064.52	−0.77	−0.44	21.93	−0.34	−0.31	47.23
17a	5989073.42	5931685.50	1480.84	0.27	0.55	40.17	−0.29	−0.34	31.47
18a	5975974.31	5917202.25	3433.81	1.51	1.60	25.74	−0.14	0.12	−10.90
average values				0.00	0.25	46.75	−0.29	−0.23	29.60

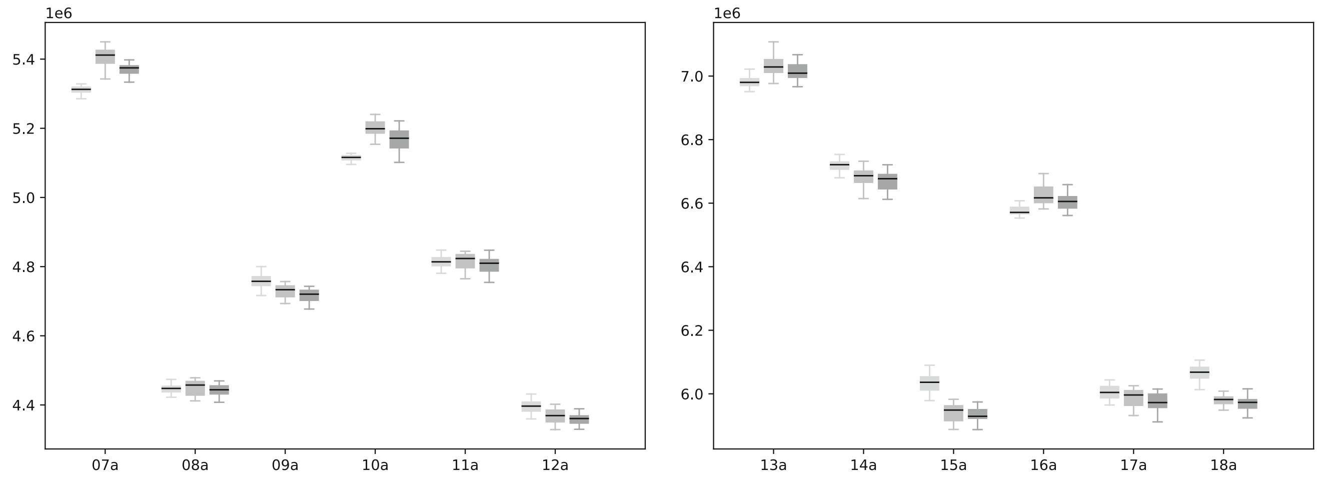


Fig. 10 Comparison with the state of the art. The light grey corresponds to SMA, the middle grey corresponds to RMA and the dark grey to EMA

drawbacks. For this reason, a set of metrics has been proposed. Using these metrics, we have compared our proposal to the direct translation of the crisp model for the total energy consumption, allowing us to evaluate their characteristics and conclude the superiority of the one we propose, not only for its higher performance in fuzzy measures but also because its better behaviour when considering real scenarios. The main defining characteristic of this model is its ability to keep uncertainty under control, allowing algorithms not only to obtain better results, but also results that are useful and do not introduce extra noise.

Moreover, to integrate the model into search algorithms, we have proposed a neighbourhood that guarantees feasibility and connectivity. This claim has been mathematically demonstrated and, in addition, it has been tested in a memetic algorithm, achieving results comparable to the state of the art, even using much simpler search strategies.

As future work, we intend to combine the energy consumption minimization objective with more production-related measures with the aim of finding a balance between productivity and sustainability. In addition, in the industry it is common to have further constraints on how energy can be consumed, for example, a power threshold may exist that cannot be surpassed or there may be a maximum energy budget. The current model can be extended to consider such variants and allow its applicability to different scenarios.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11047-023-09967-w.

Acknowledgements This research has been supported by the Spanish Government under research Grants PID2019-106263RB-I00 and TED2021-131938B-I00 and by Universidad de Cantabria and the Government of Cantabria under Grant Concepción Arenal UC-20-20.

Author Contributions Authors are listed following the signing order. Conceptualization: Pablo García Gómez, Inés González-Rodríguez; Methodology: Pablo García Gómez, Inés González-Rodríguez, Camino R. Vela; Formal analysis and investigation: Pablo García Gómez, Inés González-Rodríguez, Camino R. Vela; Writing: Pablo García Gómez, Inés González-Rodríguez, Camino R. Vela; Visualization: Pablo García Gómez, Camino R. Vela; Software: Pablo García Gómez; Funding acquisition: Ines González-Rodríguez, Camino R. Vela; Resources: Ines González-Rodríguez, Camino R. Vela; Supervision: Camino R. Vela, Inés González-Rodríguez

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This research has been supported by the Spanish Government under research Grants PID2019-106263RB-I00 and TED2021-131938B-I00 and by Universidad de Cantabria and the Government of Cantabria under Grant Concepción Arenal UC-20-20.

Data availability All instances used in this study are taken from the literature and therefore available to the scientific community. Detailed data is published as supplementary material. The source code and more detailed information is published in the corresponding author website https://pablogarciagomez.com/research.

Declarations

Conflict of interest The authors declare that they have no financial or non-financial conflicts of interest.

Ethics approval Not applicable.

References

Abdullah S, Abdolrazzagh-Nezhad M (2014) Fuzzy job-shop scheduling problems: a review. Inf Sci 278:380–407. https://doi.org/10.1016/j.ins.2014.03.060

Afsar S, Palacios JJ, Puente J et al (2022) Multi-objective enhanced memetic algorithm for green job shop scheduling with uncertain times. Swarm Evol Comput 68(101):016. https://doi.org/10.1016/j.swevo.2021.101016

Bakon K, Holczinger T, Sule Z et al (2022) Scheduling under uncertainty for industry 4.0 and 5.0. IEEE Access 10:74,977-75,017. https://doi.org/10.1109/ACCESS.2022.319142

Błazewicz J, Ecker KH, Pesch E et al (2019) Handbook on scheduling: from theory to practice. International handbooks on information systems, 2nd edn. Springer, Berlin

Çalis B, Bulkan S (2015) A research survey: review of AI solution strategies of job shop scheduling problem. J Intell Manuf 26(5):961–973

Chaudhry IA, Khan AA (2016) A research survey: review of flexible job shop scheduling techniques. Int Trans Oper Res 23(3):551–591. https://doi.org/10.1111/itor.12199

Chen SH (1985) Ranking fuzzy numbers with maximizing set and minimizing set. Fuzzy Sets Syst 17(2):113–129

Dell' Amico M, Trubian M (1993) Applying Tabu search to the job-shop scheduling problem. Ann Oper Res 41:231–252

Dubois D, Prade H (1993) Fuzzy numbers: an overview. In: Dubois D, Prade H, Yager RR (eds) Readings in fuzzy sets for intelligent systems. Morgan Kaufmann, Cambridge, pp 112–148. https://doi.org/10.1016/b978-1-4832-1450-4.50015-8

Dubois D, Fargier H, Fortemps P (2003) Fuzzy scheduling: modelling flexible constraints vs. coping with incomplete knowledge. Eur J Oper Res 147:231–252. https://doi.org/10.1016/S0377-2217(02)00558-1

Gao K, Cao Z, Zhang L et al (2019) A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems. IEEE/CAA J Autom Sin 6(4):904–916. https://doi.org/10.1109/JAS.2019.1911540

García Gómez P, Vela CR, González-Rodríguez I (2021) A memetic algorithm to minimize the total weighted tardiness in the fuzzy flexible job shop. In: Proceedings of the 19th conference of the Spanish Association for artificial intelligence, CAEPIA 2020/2021, Málaga, Spain, Sept 22–24, 2021

García Gómez P, González-Rodríguez I, Vela CR (2022) Reducing energy consumption in fuzzy flexible job shops using memetic search. In: Proceedings of the 9th international work-conference on the interplay between natural and artificial computation, IWINAC 2022, Puerto de la Cruz, Spain, pp 140–150. https://doi.org/10.1007/978-3-031-06527-9_14

García Gómez P, González-Rodríguez I, Vela CR (2023) Enhanced memetic search for reducing energy consumption in fuzzy flexible job shops. Integr Comput Aided Eng **(accepted)**

Gen M, Lin L et al (2021) Advances in hybrid evolutionary algorithms for fuzzy flexible job-shop scheduling: State-of-the-art survey. In: Proceedings of the 13th international conference on agents and artificial intelligence (ICAART 2021), vol 1. SciTePress, pp 562–573. https://doi.org/10.5220/0010429605620573

Gendreau M, Potvin JY (eds) (2019) Handbook of metaheuristics, vol 272, 3rd edn. International series in operations research & management science. Springer, Berlin. https://doi.org/10.1007/978-3-319-91086-4

Ghrayeb OA (2003) A bi-criteria optimization: minimizing the integral value and spread of the fuzzy makespan of job shop scheduling problems. Appl Soft Comput 2(3):197–210. https://doi.org/10.1016/S1568-4946(02)00069-8

Gong G, Chiong R, Deng Q et al (2020) A memetic algorithm for multi-objective distributed production scheduling: minimizing the makespan and total energy consumption. J Intell Manuf 31:1443–1466. https://doi.org/10.1007/s10845-019-01521-9

González M, Vela CR, Varela R (2013) An efficient memetic algorithm for the flexible job shop with setup times. In: Proceedings of the 23th international conference on automated planning and scheduling (ICAPS-2013), pp 91–99

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

González Rodríguez I, Puente J, Vela CR et al (2008a) Semantics of schedules for the fuzzy job shop problem. IEEE Trans Syst Man Cybern A 38(3):655–666

González Rodríguez I, Vela CR, Puente J et al (2008b) A new local search for the job shop problem with uncertain durations. In: Proceedings of the 18th international conference on automated planning and scheduling (ICAPS-2008). AAAI Press, Sidney, pp 124–131

González Rodríguez I, Vela CR, Hernández-Arauzo A et al (2009) Improved local search for job shop scheduling with uncertain durations. In: Proceedings of the 19th international conference on automated planning and scheduling (ICAPS-2009). AAAI Press, Thesaloniki, pp 154–161

González-Rodríguez I, Puente J, Palacios JJ et al (2020) Multi-objective evolutionary algorithm for solving energy-aware fuzzy job shop problems. Soft Comput 24:16,291-16,302. https://doi.org/10.1007/s00500-020-04940-6

Guo W, Xu P, Zhao Z et al (2020) Scheduling for airport baggage transport vehicles based on diversity enhancement genetic algorithm. Natural Comput. https://doi.org/10.1007/s11047-018-9703-0

Han Z, Zhang X, Zhang H et al (2022) A hybrid granular-evolutionary computing method for cooperative scheduling optimization on integrated energy system in steel industry. Swarm Evol Comput 73(101):123. https://doi.org/10.1016/j.swevo.2022.101123

Imaran Hossain S, Akhand M, Bhuvo M et al (2019) Optimization of university course scheduling problem using particle swarm optimization with selective search. Expert Syst Appl 127:9–24. https://doi.org/10.1016/j.eswa.2019.02.026

Kuhpfahl J, Bierwirth C (2016) A study on local search neighbourhoods for the job shop scheduling problem with total weighted tardiness objective. Comput Oper Res 261:44–57

Lenstra J, Rinnooy Kan A, Brucker P (1977) Complexity of machine scheduling problems. Ann Discrete Math 1:343–362

Li M, Wang GG (2022) A review of green shop scheduling problem. Inf Sci 589:478–496. https://doi.org/10.1016/j.ins.2021.12.122

Li R, Gong W, Lu C (2022) Self-adaptive multi-objective evolutionary algorithm for flexible job shop scheduling with fuzzy processing time. Comput Ind Engi 168(108):099. https://doi.org/10.1016/j.cie.2022.108099

Li R, Gong W, Lu C et al (2022) A learning-based memetic algorithm for energy-efficient flexible job shop scheduling with type-2 fuzzy processing time. IEEE Trans Evol Comput. https://doi.org/10.1109/TEVC.2022.3175832

Liu Z, Guo S, Wang L (2019) Integrated green scheduling optimization of flexible job shop and crane transportation considering comprehensive energy consumption. J Clean Prod 211:765–786. https://doi.org/10.1016/j.jclepro.2018.11.231

Luo H, Dridi M, Grunder O (2020) A green routing and scheduling problem in home health care. IFAC PapersOnLine 52(2):11119–11124

Mansouri N, Ghafari R (2022) Cost efficient task scheduling algorithm for reducing energy consumption and makespan of cloud computing. J Comput Knowl Eng 5(1):1–10. https://doi.org/10.22067/cke.2022.70000.1008

Mattfeld DC (1995) Evolutionary search and the job shop investigations on genetic algorithms for production scheduling. Springer-Verlag

Mencía R, Mencía C, Varela R (2022) A memetic algorithm for restoring feasibility in scheduling with limited makespan. Nat Comput 21(4):553–563. https://doi.org/10.3233/ICA-160519

Niu B, Liu Q, Wang Z et al (2021) Multi-objective bacterial colony optimization algorithm for integrated container terminal scheduling problem. Nat Comput 20:89–104. https://doi.org/10.1007/s11047-019-09781-3

Osaba E, Ser JD, Cotta C et al (2022) Memetic computing: Accelerating optimization heuristics with problem-dependent local search methods (editorial). Swarm Evol Comput 70(101):047. https://doi.org/10.1016/j.swevo.2022.101047

Palacios JJ, González-Rodríguez I, Vela CR et al (2014) Robust swarm optimisation for fuzzy open shop scheduling. Nat Comput 13(2):145–156

Palacios JJ, González MA, Vela CR et al (2015a) Genetic tabu search for the fuzzy flexible job shop problem. Comput Oper Res 54:74–89. https://doi.org/10.1016/j.cor.2014.08.023

Palacios JJ, González-Rodríguez I, Vela CR et al (2015b) Coevolutionary makespan optimisation through different ranking methods for the fuzzy flexible job shop. Fuzzy Sets Syst 278:81–97. https://doi.org/10.1016/j.fss.2014.12.003

Palacios JJ, Puente J, Vela CR et al (2016) Benchmarks for fuzzy job shop problems. Inf Sci 329:736–752. https://doi.org/10.1016/j.ins.2015.09.042

Palacios JJ, González-Rodríguez I, Vela CR et al (2017) Robust multiobjective optimisation for fuzzy job shop problems. Appl Soft Comput 56:604–616. https://doi.org/10.1016/j.asoc.2016.07.004

Palacios JJ, González-Rodríguez I, Vela CR et al (2019) Satisfying flexible due dates in fuzzy job shop by means of hybrid evolutionary algorithms. Integr Comput Aided Eng 26:65–84. https://doi.org/10.3233/ICA-180583

Pan Z, Lei D, Wang L (2022) A bi-population evolutionary algorithm with feedback for energy-efficient fuzzy flexible job shop scheduling. IEEE Trans Syst Man Cybern Syst 52(8):5295–5307. https://doi.org/10.1109/TSMC.2021.3120702

Pinedo ML (2016) Scheduling. Theory, Algorithms, and Systems, 5th edn. Springer, Berlin. https://doi.org/10.1007/978-1-4614-2361-4

Razali M, Abd Rahman A, Ayob M et al (2022) Research trends in the optimization of the master surgery scheduling problem. IEEE Access 10:91,466-91,480

Sun L, Lin L, Gen M et al (2019) A hybrid cooperative coevolution algorithm for fuzzy flexible job shop scheduling. IEEE Trans Fuzzy Syst 27(5):1008–1022. https://doi.org/10.1109/TFUZZ.2019.2895562

Xiong H, Shi S, Ren D et al (2022) A survey of job shop scheduling problem: the types and models. Comput Oper Res 142(105):731. https://doi.org/10.1016/j.cor.2022.105731

Ye X, Li J, Liu S et al (2019) A hybrid instance-intensive workflow scheduling method in private cloud environment. Nat Comput 18(4):735–746. https://doi.org/10.1007/S11047-016-9600-3

Zhang C, Li P, Guan Z et al (2007) A tabu search algorithm with a new neighborhood structure for the job shop scheduling problem. Comput Oper Res 34(11):3229–3242. https://doi.org/10.1016/j.cor.2005.12.002

Zuo Y, Gong M, Jiao L (2017) Adaptive multimeme algorithm for flexible job shop scheduling problem. Nat Comput 16:677–698