Heuristic Local Search for Fuzzy Open Shop Scheduling

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

Abstract— **We consider the fuzzy open shop scheduling problem, where task durations are assumed to be ill-known and modelled as triangular fuzzy numbers. We propose a neighbourhood structure for local search procedures, based on reversing critical arcs in the associated disjunctive graph. We provide a thorough theoretical study of the structure and, in particular, prove that feasibility and asymptotic convergence hold. We further illustrate its good behaviour with experimental results obtained by incorporating the local search procedure to an existing genetic algorithm from the literature and provide a new benchmark of problem instances.**

I. Introduction

Scheduling problems form an important body of research since the late fifties, with multiple applications in industry, finance and science [1]. In particular, the open shop scheduling problem has clear applications; consider for instance testing facilities, where units go through a series of diagnostic tests that need not be performed in a specified order and where different testing equipment is usually required for each test (so it is not possible to conduct any two tests concurrently). This situation is frequent in testing components of electronic systems and in general repair facilities when repairs can be performed in an arbitrary order, as well as certain medical diagnosis procedures. The open shop is actually obtained from the much better-known job shop by removing certain constraints, making it tempting to regard it as a simpler problem. However, the open shop scheduling problem is NP-complete for a number of machines $m \geq 3$ and, additionally, its solution space is significantly larger. Specific and efficient methods to solve the open shop are still scarce, despite their increasing presence in the recent literature [2],[3],[4].

To enhance the range of applications of scheduling, part of the research is devoted to model the uncertainty and vagueness pervading real-world situations, with great diversity of approaches [5]. In particular, fuzzy sets have been used in different manners, ranging from representing incomplete or vague states of information to using fuzzy priority rules with linguistic qualifiers or preference modelling [6],[7]. They are also emerging as an interesting tool for improving solution robustness, a much-desired property in real-life applications [8],[9].

In deterministic scheduling the complexity of problems such as shop problems means that practical approaches to solving them usually involve heuristic strategies: simulated annealing, genetic algorithms, local search, etc [10]. Some attempts have been made to extend these heuristic methods to the case where uncertain durations are modelled via fuzzy intervals, most commonly and successfully for the flow shop problem: among others, a genetic algorithm is used in [11] and a genetic algorithm is hybridised with a local search procedure in [12]. For the job shop, we find a neural approach [13], genetic algorithms [14],[15],[16], simulated annealing [17], genetic algorithms hybridised with local search [18] or particle swarm optimisation [19],[20]. To the best of our knowledge, the open shop problem has not yet been tackled in the fuzzy scheduling framework, with the exception of [21], where fuzzy sets are used to represent flexible job start and due dates, and [22], where a genetic algorithm is proposed to solve the open shop with fuzzy durations.

In this paper, we intend to advance in the study of metaheuristic methods to solve the fuzzy open shop problem with expected makespan minimisation, denoted $FuzO||E[C_{max}]$. We shall propose a neighbourhood structure for local search, studying its good behaviour from a theoretical point of view. Finally, we shall see how it can be combined to the genetic algorithm from [22] to improve the quality of the best solutions found so far.

II. The Fuzzy Open Shop Problem

The *open shop scheduling problem*, or *OSP* in short, consists in scheduling a set of n jobs $J_1, \ldots, J_n$ to be processed on a set of m physical resources or machines $M_1, \ldots, M_m$. Each job consists of m tasks or operations, each requiring the exclusive use of a different machine for its whole processing time without preemption, i.e. all operations must be processed without interruption. In total, there are mn operations, $\{O_k, 1 \leq k \leq mn\}$. A solution to this problem is a *schedule*–an allocation of starting times for all operations–which is *feasible*, in the sense that all constraints hold, and is also optimal according to some criterion. Here, we shall consider the objective of minimising the makespan C_{max}, that is, the time lag from the start of the first operation until the end of the last one, a problem often denoted $O||C_{max}$ in the literature.

A. Uncertain Durations as Triangular Fuzzy Numbers

In real-life applications, it is often the case that the exact time it takes to process a task is not known in advance. However, based on previous experience, an expert may have some knowledge (albeit uncertain) about the duration. The crudest representation for uncertain processing times would be a human-originated confidence interval. If some values appear to be more plausible than others, a natural extension is a fuzzy interval or fuzzy number. The simplest model is

Juan José Palacios, Camino R. Vela and Jorge Puente are with the Department of Computer Science, University of Oviedo, Campus de Viesques s/n, 33271, Gijón, Spain (email: {UO165228,crvela,puente}@uniovi.es).

Inés González-Rodríguez is with the Department of Mathematics, Statistics and Computing, University of Cantabria, Los Castros s/n, 39005, Santander, Spain (email: ines.gonzalez@unican.es).

a *triangular fuzzy number* or *TFN*, using an interval $[a^1, a^3]$ of possible values and a modal value a^2 in it. For a TFN A, denoted $A = (a^1, a^2, a^3)$, the membership function takes the following triangular shape:

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

Triangular fuzzy numbers and more generally fuzzy intervals have been extensively studied in the literature (cf. [23]). A *fuzzy interval* Q is a fuzzy quantity (a fuzzy set on the reals) whose α-cuts $Q_\alpha = \{r \in \mathbb{R} : \mu_Q(r) \geq \alpha\}$, $\alpha \in (0, 1]$, are intervals (bounded or not). The *support* of Q is $Q_0 = \{r \in \mathbb{R} : \mu_Q(r) > 0\}$. A *fuzzy number* M is a fuzzy quantity whose α-cuts are closed intervals, denoted $M_\alpha = [\underline{m}_\alpha, \overline{m}_\alpha]$, with compact support and unique modal value.

In the open shop, we essentially need two operations on fuzzy quantities, the sum and the maximum. These are obtained by extending the corresponding operations on real numbers using the *Extension Principle*. If f is a bivariate continuous isotonic function and M and N are two fuzzy numbers, it can be proved that the result of applying f is a fuzzy number $F = f(M, N)$ such that $F_\alpha = [f(\underline{m}_\alpha), f(\underline{n}_\alpha)), f(\overline{m}_\alpha), f(\overline{n}_\alpha))]$, that is, computing the function is equivalent to computing it on every α-cut of the fuzzy numbers. Since both the addition and the maximum are continuous isotonic functions, we can use this equality to compute the sum or maximum of two fuzzy numbers. However, this is cumbersome if not intractable in general, because it requires evaluating two sums or two maxima for every value $\alpha \in [0, 1]$. For the sake of simplicity and tractability of numerical calculations, we follow [17] and approximate the results of these operations by a linear interpolation evaluating only the operation on the three defining points of each TFN (an approach also taken, for instance, in [19],[24]). The approximated sum coincides with the actual sum, so for any pair of TFNs M and N:

$$M + N = (m^1 + n^1, m^2 + n^2, m^3 + n^3)$$

Regarding the maximum, for any two TFNs M, N, if $F = \max(M, N)$ denotes their maximum and $G = (\max\{m^1, n^1\}, \max\{m^2, n^2\}, \max\{m^3, n^3\})$ the approximated value, it holds that:

$$\forall \alpha \in [0, 1], \quad \underline{f}_\alpha \leq \underline{g}_\alpha, \overline{f}_\alpha \leq \overline{g}_\alpha.$$

The approximated maximum G is thus a TFN which artificially increases the value of the actual maximum F, but maintaining the support and modal value, that is, $F_0 = G_0$ and $F_1 = G_1$. This approximation can be trivially extended to the case of more than two TFNs.

The membership function μ_Q of a fuzzy quantity Q can be interpreted as a possibility distribution on the real numbers; this allows to define the *expected value* of a fuzzy quantity [25], given for a TFN A by

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

The expected value coincides with the *neutral scalar substitute* of a fuzzy interval and can also be obtained as the centre of gravity of its *mean value* or using the *area compensation* method [6]. It induces a total ordering $\leq_E$ in the set of fuzzy intervals [17], where for any two fuzzy intervals M, N $M \leq_E N$ if and only if $E[M] \leq E[N]$. Clearly, for any two TFNs A and B, if $\forall i, a^i \leq b^i$, then $A \leq_E B$.

B. The Disjunctive Graph Model Representation

An open shop problem instance may be represented by a directed graph $G = (V, E)$ with $E = A \cup D$. Each node in the set V represents a task of the problem, with the exception of the dummy nodes *start* or 0 and *end* or $nm + 1$, representing tasks with null processing times. Task θ_{ij}, $1 \leq i \leq n$, $1 \leq j \leq m$, is represented by node $x = m(i - 1) + j$. Arcs in E represent job-precedence and resource constraints: A contains two arcs (x, y) and (y, x) for each pair x, y of tasks in the same job as well as arcs $(0, x)$ and $(x, nm + 1)$ for each task x, and D includes two arcs (x, y) and (x, y) for every pair x, y of tasks requiring the same machine. $A = \cup_{i=1,\dots,n} A_i$ and $D = \cup_{j=1,\dots,m} D_j$, where A_i corresponds to job J_i and D_j corresponds to machine M_j. Each arc is weighted with the processing time of the task at the source node (a TFN in our case). A feasible processing order of tasks π corresponds to an acyclic subgraph $G(\pi) = (V, E(\pi))$ of G, where $A(\pi) \cup D(\pi))$, $A(\pi) = \cup_{i=1\dots m} A_i(\pi)$ and $D(\pi) = \cup_{j=1\dots m} D_j(\pi)$, $A_i(\pi)$ and $D_j(\pi)$ being a hamiltonian selection of A_i and D_j respectively. Using forward propagation in $G(\pi)$, we can obtain the starting and completion times for all tasks and, therefore, the makespan $C_{max}(\pi)$.

Since task processing times are fuzzy intervals, the addition and maximum operations used to propagate constraints are taken to be the corresponding operations on fuzzy intervals, approximated for the particular case of TFNs as explained above. The obtained schedule will be a fuzzy schedule in the sense that the starting and completion times of all tasks and the makespan are fuzzy intervals, interpreted as possibility distributions on the values that the times may take. However, the task processing ordering π that determines the schedule is crisp; there is no uncertainty regarding the order in which tasks are to be processed.

To illustrate these ideas, consider a problem of $n = 3$ jobs and $m = 3$ machines with the following matrices for fuzzy processing times and machine allocation:

$$\boldsymbol{p} = \begin{pmatrix} (2,3,6) & (4,5,6) & (7,8,8) \\ (6,6,6) & (4,5,6) & (5,6,7) \\ (6,8,8) & (6,7,9) & (1,3,5) \end{pmatrix} \boldsymbol{\nu} = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}$$

A feasible task processing order for this problem is given by $\pi =$(1 5 9 2 6 7 3 4 8); its corresponding solution graph can be seen in Figure 1 (for the sake of clarity, the operations in the same job have been displayed in the order in which they are to be performed according to π). If S_x and C_x denote, respectively, the starting and completion times of a task x, it is easy to check that, for instance for task 6, $S_6 = \max\{C_5, C_9\} = (4, 5, 6)$ and $C_6 = S_6 + p_6 = (9, 11, 13)$.

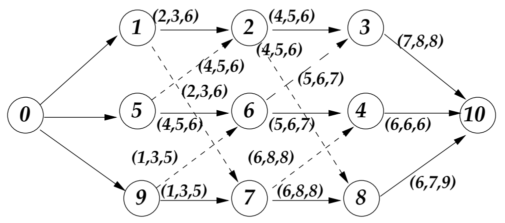


Fig. 1. Solution graph $G(\pi)$ for the processing order π =(1 5 9 2 6 7 3 4 8). $C_{max}(\pi) = (16, 19, 23)$

In the crisp case, a *critical path* in a solution graph is defined as the longest path from node 0 to node $nm + 1$ and a *critical arc* or *critical activity* is an arc or activity in a critical path. It is not trivial to extend these concepts and related algorithms to the problem with uncertain durations (cf. [26],[6],[27],[28]). For the fuzzy open shop as considered herein it may even be the case that the makespan (a TFN) does not coincide with the completion time of one job (unlike the crisp case).

C. Fuzzy Open Shop Scheduling

In analogy to the original problem, our objective is to find a fuzzy schedule with optimal makespan. An important issue with fuzzy times is to decide on the precise meaning of "optimal makespan". This is not trivial, since neither the maximum nor its approximation define a total ordering in the set of TFNs. Using ideas similar to stochastic scheduling, we follow the approach taken for the fuzzy job shop in [18] and use the total ordering provided by the expected value and consider that the objective is to minimise the expected makespan $E[C_{max}]$. The resulting problem will be denoted $FuzO||E[C_{max}]$, following the $\alpha|\beta|\gamma$ notation.

D. Local Search

Part of the interest of critical paths stems from the fact that they may be used to define neighbourhood structures for local search. Roughly speaking, a typical local search schema starts from a given solution, calculates its neighbourhood and then neighbours are evaluated in the search of an improving solution. In simple hill-climbing, the first improving neighbour found will replace the original solution, so local search starts again from that improving neighbour. The procedure finishes when no neighbour satisfies the acceptation criterion.

Clearly, a central element in any local search procedure is the definition of neighbourhood. For the crisp job shop, a well-known neighbourhood, which relies on the concept of critical path was proposed in [29]. For a given solution graph, a *move* is defined as a transformation of this graph, based on reversing a critical arc, which yields a new solution graph. The neighbourhood structure is then defined as the set of solutions obtained from the original one after applying all possible moves. It presents interesting features: all neighbours represent feasible solutions, additional moves based on reversing non-critical arcs can never reduce the makespan and there exists a finite sequence of transitions leading from any given element to some globally optimal element, usually referred to as connectivity property. Feasibility means that feasibility checks or repair procedures are not necessary, with the consequent computational gain and better coverage of the solution space. The second property means that the neighbourhood reduces the search space without any loss in the potential quality of solutions. Finally, connectivity ensures asymptotical convergence in probability to a globally optimal solution, a property which does not always hold for neighbourhood structures.

Here, we shall extend this neighbourhood in a twofold manner: from the job shop to the more general open shop and from the deterministic to the fuzzy framework. We shall also prove that the proposed extension maintains the aforementioned desirable properties.

III. Main Results

The definition of the neighbourhood structure will be based on reversing critical arcs. We thus start by providing a definition of criticality for the open shop scheduling problem, as an extension of the deterministic case.

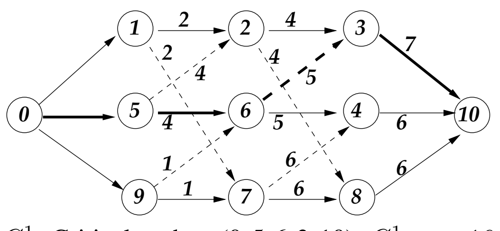
A. Criticality for the Fuzzy Open Shop

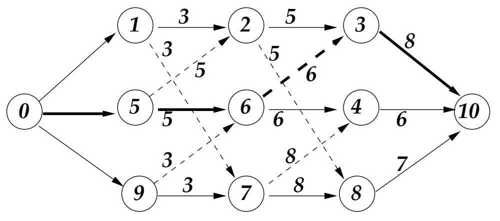
All arithmetic operations used to propagate constraints in the graph are performed on the three defining points or components of the TFNs. This makes it possible to decompose the solution graph with fuzzy durations into three parallel graphs with crisp durations. Similar ideas for the fuzzy job shop have been proposed in [30]. Let Σ denote the space of feasible solutions for the $FuzO||E[C_{max}]$.

Definition 1: Let $\pi \in \Sigma$ be a feasible task processing order and let $G(\pi) = (V, E(\pi))$ be the corresponding solution graph, where the cost of any arc $(x, y) \in E(\pi)$ is a TFN representing the processing time p_x of task x. From $G(\pi)$, we define the *parallel solution graphs* $G^i(\pi)$, $i = 1, 2, 3$, which are identical to $G(\pi)$ except for the cost of arc $(x, y) \in E(\pi)$, which for graph $G^i(\pi)$ will be the i-th defining point of p_x, that is, p^i_x.
Each of the parallel graphs $G^i(\pi)$ is a solution graph identical to those for crisp OSP. Therefore, in each of them a critical path is the longest path from node *start* to node *end*. Notice that it is not necessarily unique; for instance, Figure 2, shows the three parallel graphs generated from the graph in Figure 1 and it is easy to see that, if $p_4 = (7, 7, 7)$, then (0 5 6 4 10) would also be critical in $G^1(\pi)$.

Using the parallel graph representation, we may extend the notion of criticality to the fuzzy open shop as follows:

Definition 2: A path P in $G(\pi)$ is an *i-critical path* if and only if P is critical in some $G^i(\pi)$. Nodes and arcs in a i-critical path are termed *i-critical*.
According to this definition, the sets of i-critical paths, arcs and tasks are respectively the union of critical paths, arcs and tasks in the parallel solution graphs. Unlike for the crisp OSP, we may not state that the makespan of the schedule is the cost of a i-critical path. However, it holds that each component of the makespan $C^k_{max}(\pi)$ is the cost of a critical path in the corresponding solution parallel graph $G^k(\pi)$ (coinciding with the k-th component of the cost of an i-critical path).


G^1. Critical path = (0 5 6 3 10). $C^1_{max} = 16$


G^2. Critical path = (0 5 6 3 10). $C^2_{max} = 19$

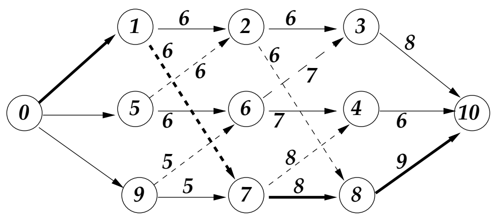

G^3. Critical path = (0 1 7 8 10). $C^3_{max} = 23$

Fig. 2. Parallel graphs corresponding to the graph in Figure 1, with i-critical paths in bold.

For a solution graph $G(\pi)$ and a task x, let S_x and C_x denote respectively the starting and completion times of x, let $P\nu_x$ and $S\nu_x$ denote the predecessor and successor nodes of x on the machine sequence (predecessor and successor in $D(\pi)$), and let PJ_x and SJ_x denote respectively the predecessor and successor nodes of x on the job sequence (predecessor and successor in $A(\pi)$). We now extend some well-known notions for the crisp case and define the *head* r_x of task x as the starting time of x i.e. S_x, in our context, a TFN $r_x = (r_x^1, r_x^2, r_x^3)$ given by:

$$r_x = \max(r_{PJ_x} + p_{PJ_x}, r_{P\nu_x} + p_{P\nu_x}),$$

and the *tail* q_x of task x as:

$$q_x = \max(q_{SJ_x} + p_{SJ_x}, q_{S\nu_x} + p_{S\nu_x}),$$

Clearly, $C_x = r_x + p_x$. For each parallel graph $G^i(\pi)$, r^i_x is the length of the longest path from node *start* to node x and $q^i_x + p^i_x$ is the length of the longest path from node x to node *end*. Hence, $r^i_x + p^i_x + q^i_x$ is the length of the longest path from node *start* to node *end* through node x in $G^i(\pi)$; it is a lower bound of the i-th component of the makespan, being equal to $C^i_{max}(\pi)$ if node x belongs to a critical path in $G^i(\pi)$. For two TFNs A and B let $A \simeq B$ denote that $\exists i, a^i = b^i$. The next proposition states some properties of i-critical arcs and tasks which follow trivially from the above definition:

Proposition 1: If an arc (x, y) is i-critical, then $r_x + p_x \simeq r_y$. A task x is i-critical if and only if $r_x + p_x + q_x \simeq C_{max}$.

B. Neighbourhood Structure

For the proposed definition of i-criticality, we may extend the neighbourhood structure given for the job shop problem in [29] to the fuzzy open shop as follows:

Definition 3: Given an arc $v = (x, y) \in E(\pi)$, let $\pi_{(v)}$ denote the processing order obtained from π after an exchange in the processing order of tasks in arc v. Then, the *neighbourhood structure* obtained from π is given by $H(\pi) = \{\pi_{(v)} : v \in E(\pi) \text{ is } i\text{-critical}\}$.

Given a task processing order π and an i-critical arc (x, y) in $G(\pi)$, the reversal of that arc produces a new processing order $\sigma = \pi_{(x,y)}$ with solution graph $G(\sigma)$. This situation is illustrated in Figure 3 for the case where (x, y) is a resource arc (being the other case analogous). The makespan after the

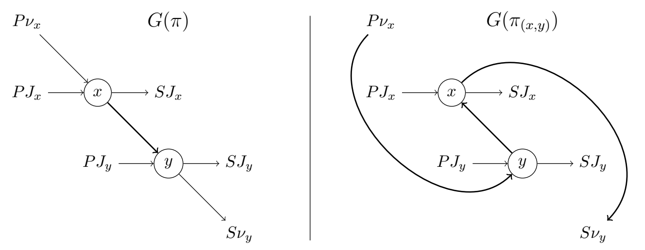


Fig. 3. Situation before (π) and after $(\sigma = \pi_{(x,y)})$ the reversal of an i-critical arc $(x, y) \in D(\pi)$.

move may be calculated as for any solution, using forward propagation in the graph $G(\sigma)$ from 0. Alternatively, the evaluation of the makespan of neighbouring solutions may be done very quickly (in time $O(N)$) using heads and tails. Without loss of generality, let us suppose the reversed arc was a resource arc, $(x, y) \in D(\pi)$, let r and q denote the heads and tails in $G(\pi)$ (before the move) and let r' and q' denote the heads and tails in $G(\sigma)$ (after the move). For every task a previous to x in π, $r_a = r'_a$ and for every task b posterior to y in π, $q_b = q'_b$. The heads and tails for x and y after the move (see Figure 3) are given by the following:

$$\begin{aligned} r'_y &= \max\{r_{PJ_y} + p_{PJ_y}, r_{P\nu_x} + p_{P\nu_x}\}, \\ r'_x &= \max\{r_{PJ_x} + p_{PJ_x}, r'_y + p_y\} \\ q'_x &= \max\{q_{SJ_x} + p_{SJ_x}, q_{S\nu_y} + p_{S\nu_y}\} \\ q'_y &= \max\{q_{SJ_y} + p_{SJ_y}, q'_x + p_x\} \end{aligned}$$

The formulae for the case where $(x, y) \in A(\pi)$ are analogous. To calculate the makespan $C_{max}(\sigma)$, we need only re-calculate the heads of tasks from x onwards in the graph $G(\sigma)$. This way of evaluating neighbours in H, albeit simple, may prove a considerable reduction in computational load for the local search procedure.

C. Comparison to Previous Approaches

In [17], we find a first proposal to extend the neighbourhood structure in [29] to the fuzzy job shop. An arc (x, y) in the solution graph for the job shop problem with fuzzy durations is taken to be a *critical arc* if and only if $C_x \simeq S_y$, that is, if there is no slack time in one of its components. This definition, being quite natural, yields some counterintuitive results. For instance, in the graph from Figure 1, arc $(7, 4)$ would be considered to be critical (in fact, all arcs except $(9, 6)$, $(2, 3)$ and $(4, 10)$ would be critical).

In [18], this definition was slightly modified by incorporating backpropagation: a *critical path* is taken to be a path in the solution graph from node 0 to node $nm + 1$ where $C_z \simeq C_{max}$ and such that for all arcs (x, y) in the path it holds $C_x \simeq S_y$. A *critical arc* (respectively a *critical task*) is an arc (respectively task) belonging to a critical path.

Let us denote as critical' an arc which is critical according to this last definition. All arcs considered to be critical according to [17] will still be critical', but the reverse does not hold, avoiding some of the counterintuitive examples. For instance, in Figure 1 the critical' arcs are (5,6), (6,3), (5,2), (2,8), (1,2), (1,7), (7,8) and (9,7). Clearly, it is possible to use this definition to extend the neighbourhood structure proposed in [29] to the fuzzy open shop: for a given feasible task processing order π, let $H'(\pi) = \{\pi_{(v)} : v \in E(\pi) \text{ is critical'}\}$ denote the resulting neighbourhood. A first remark about H' is that, even for a small problem such as the one used as example throughout this paper, the neighbourhood would be of considerable size. The next proposition relating H and H' follows trivially from the above definitions:

Proposition 2: The set of i-critical paths is a subset of the critical' ones. Consequently, neighbourhood H is contained in H': $\forall \pi \in \Sigma, H(\pi) \subseteq H'(\pi)$.
Notice that the set of i-critical paths may be a proper subset of the set of critical' paths. For instance, in our example, only (5,6), (6,3), (1,7) and (7,8) are also i-critical arcs (half of the critical' ones). Moreover, the second condition in Proposition 1 may not hold for a critical' task. For instance, in the example, arc $(5, 2)$ is critical' but is not i-critical and task 2 is critical' but $\forall i, r_1^i + p_1^i + q_1^i \neq C_{max}^i$.

According to Proposition 2, the number of neighbours is reduced when using H instead of H'. The following result ensures that this neighbourhood reduction does not affect the potential quality of solutions found by the local search:

Proposition 3: Let $\pi \in \Sigma$ be a feasible processing order and let $\sigma \in \Sigma$ be a feasible processing order obtained by the reversal an arc which is not i-critical in $G(\sigma)$. Then $\forall i, \quad C_{max}^i(\pi) \leq C_{max}^i(\sigma)$ and therefore

$$C_{max}(\pi) \leq_E C_{max}(\sigma)$$

Proof: Let σ be obtained from π by the reversal of an arc $v = (x, y)$ where v is not an i-critical arc, that is, it is not critical in any of the parallel graphs $G^i(\pi)$. Clearly, for all $i = 1, 2, 3$ the arcs inside critical paths of $G^i(\pi)$ remain unchanged after the move in $G^i(\sigma)$, and therefore $C_{max}^i(\pi) \leq C_{max}^i(\sigma)$ for all i. ■

In particular, neighbours in H' which do not belong to H can never improve the makespan:

Corollary 1: $\forall \pi \in \Sigma, \forall \sigma \in (H'(\pi) - H(\pi)) \bigcap \Sigma$, $C_{max}(\pi) \leq_E C_{max}(\sigma)$

We may conclude that using H instead of H' in the local search procedure leads to a reduction in the number of evaluated neighbours without missing any improving neighbours, i.e. with no loss in the quality of the solution obtained by the local search. Notice that if H' were to denote the neighbourhood proposed in [17], the above results would still hold.

D. Neighbourhood Properties

Now we further study the new neighbourhood H, in order to prove that it has two highly desirable properties: feasibility and connectivity.

Theorem 1: Let $\pi \in \Sigma$ be a feasible task processing order; the reversal of an i-critical arc $v = (x, y) \in E(\pi)$ produces a feasible processing order, i.e., $\pi_{(v)} \in \Sigma$. In consequence, $H(\pi) \subset \Sigma$.

Proof: If $v = (x, y)$ is i-critical, by definition, $\exists i, r_y^i = r_x^i + p_x^i$. Suppose by contradiction that $G(\pi_{(v)})$ has a cycle. Since $G(\pi)$ has no cycles and the only change in $\pi_{(v)}$ w.r.t. π has been the processing order between x and y, having a cycle in $G(\pi_{(v)})$ means that there must exist an alternative path from x to y in $G(\pi)$, as shown in Figure 4.

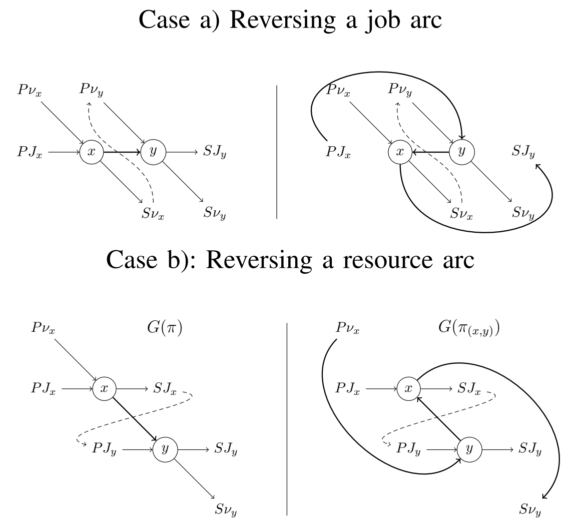


Fig. 4. Reversing an arc in a critical block with an alternative path

From the expression defining the head r_y, $\forall i$ we have that

$$r_y^i \geq r_{PJ_y}^i + p_{PJ_y}^i, \quad r_y^i \geq r_{P\nu_y}^i + p_{P\nu_y}^i.$$

Given the existence of an alternative path from x to y, there are two possibilities:

- If $(x, y) \in A(\pi)$ is a job arc,

  $$r_y^i \geq r_{S\nu_x}^i + p_{S\nu_x}^i + p_{P\nu_y}^i \geq r_x^i + p_x^i + p_{S\nu_x}^i + p_{P\nu_y}^i$$

  But being critical, there is at least one component k where $r_y^k = r_x^k + p_x^k$, and for this k we have:

  $$r_x^k + p_x^k \geq r_x^k + p_x^k + p_{S\nu_x}^k + p_{P\nu_y}^k$$

  which is contradiction with all task durations being strictly positive.
- If $(x, y) \in R(\pi)$ is a resource arc,

  $$r_y^i \geq r_{SJ_x}^i + p_{SJ_x}^i + p_{PJ_y}^i \geq r_x^i + p_x^i + p_{SJ_x}^i + p_{PJ_y}^i$$

and similarly, we obtain that for some component k, $p^k_{SJ_x} + p^k_{PJ_y} = 0$, which again is a contradiction.

Therefore, there can be no cycles in $G(\pi_{(v)})$, that is, $\pi_{(v)}$ is a feasible task processing order. ■

Notice that feasibility means that local search is automatically limited to the subspace of feasible task orders. It has the additional advantage of avoiding feasibility checks for the neighbours, hence increasing the efficiency of the local search procedure (reducing computational load) and avoiding the loss of feasible solutions that is usually encountered for feasibility checking procedures (cf. [31]).

A second property of the proposed neighbourhood is connectivity. In order to prove it, we first state the following partial result:

Proposition 4: Let $\pi \in \Sigma$ be a feasible task processing order, $G(\pi) = (V, E(\pi))$ its disjunctive graph and π_0 an optimal processing order. Let

$$V_\pi(\pi_0) = \{v = (x,y) \in E(\pi) : v \text{ is } i\text{-critical },\ (y,x) \in \overline{A(\pi_0)} \cup \overline{D(\pi_0)}\}$$

where $\overline{A(\pi)}$ ($\overline{D(\pi)}$) denotes the transitive closure of $A(\pi)$ ($D(\pi)$); i.e., $V_\pi(\pi_0)$ is the set of i-critical arcs (x,y) in $E(\pi)$ such that there exists a path from y to x in $E(\pi_0)$. If π is not optimal, then $V_\pi(\pi_0) \neq \emptyset$ or, equivalently, if $V_\pi(\pi_0) = \emptyset$ then π is optimal.

Proof: First notice that if π is not optimal, there is at least an i-critical arc in $A(\pi)$ and another in $D(\pi)$. Indeed, w.l.o.g. assume that there are no i-critical arcs in $D(\pi)$. That means that all i-critical arcs in $G(\pi)$ belong to $A(\pi)$. Therefore, for all i, all critical paths in $G^i(\pi)$ belong to $A(\pi)$. Hence, in each $G^i(\pi)$ there exists a critical path where all arcs belong to $A(\pi)$ and such path is optimal in $G^i(\pi)$ (for every i, a path where all arcs belong to the same job is a lower bound of C^i_{max}). Therefore, π is optimal.

Let us now asume that all i-critical arcs $v = (x,y) \in E(\pi)$ verify that $(x,y) \in \overline{A(\pi_0)} \cup \overline{D(\pi_0)}$. This means that for every component i, the critical arcs in $A^i(\pi) \cup D^i(\pi)$ are in the transitive closure $\overline{A(\pi_0)} \cup \overline{D(\pi_0)}$. Hence, for every i, a critical path P^i in $G^i(\pi)$ is also a path in $\overline{G^i(\pi_0)} = (V, \overline{A^i(\pi_0)} \cup \overline{D^i(\pi_0)})$. Let R^i denote an arbitrary critical path in $\overline{G^i(\pi_0)}$ and let $\|R^i\|$ denote its length. Notice that the length of a longest path in $\overline{G^i(\pi_0)}$ is also the length of a longest path in $G^i(\pi_0) = (V, A^i(\pi_0) \cup D^i(\pi_0))$, that is, $\|R^i\| = C^i_{max}(\pi_0)$. Since P^i is critical in $G^i(\pi)$ and a path in $\overline{G^i(\pi_0)}$, it holds that

$$\forall i, C^i_{max}(\pi) = \|P^i\| \leq \|R^i\| = C^i_{max}(\pi_0)$$

and therefore

$$E[C_{max}(\pi)] \leq E[C_{max}(\pi_0)].$$

But π_0 is optimal, which means $E[C_{max}(\pi)] = E[C_{max}(\pi_0)]$, that is, π is optimal. ■

Theorem 2: H verifies the connectivity property: given a globally optimal processing order π_0, it is possible to build a finite sequence of transitions of H starting from any non-optimal task processing order π and leading to π_0.

Proof: Let π_0 be any optimal processing order and let the sequence $\{\lambda_k\}_{k\geq 0}$ of processing orders be given by the following recursive definition:

$$\lambda_0 = \pi$$

λ_{k+1} is obtained from λ_k by reversal of an arc $v \in V_{\lambda_k}(\pi_0)$

Notice that the reversal of an arc in $V_{\lambda_k}(\pi_0)$ is a move from H so, by Theorem 1, $\forall k\ \lambda_k \in \Sigma$ (i.e., all task processing orders in the sequence are feasible solutions). Let us prove that the above sequence is finite. For any processing order $\pi \in \Sigma$, we define the following sets:

$$M_\pi(\pi_0) = \{v = (x,y) \in E(\pi) : (y,x) \in \overline{E(\pi_0)}\}$$
$$\overline{M_\pi(\pi_0)} = \{v = (x,y) \in \overline{E(\pi)} : (y,x) \in \overline{E(\pi_0)}\}$$

Clearly, $V_\pi(\pi_0) \subset M_\pi(\pi_0)$ and $M_\pi(\pi_0) \subset \overline{M_\pi(\pi_0)}$. Let $\|M_\pi(\pi_0)\|$ and $\|\overline{M_\pi(\pi_0)}\|$ denote their cardinals. By definition of λ_k, if $\|\overline{M_{\lambda_k}(\pi_0)}\| > 0$ then

$$\|\overline{M_{\lambda_{k+1}}(\pi_0)}\| = \|\overline{M_{\lambda_k}(\pi_0)}\| - 1.$$

Therefore, for $k^\star = \|\overline{M_\pi(\pi_0)}\|$, we have that $\|\overline{M_{\lambda_{k^\star}}(\pi_0)}\| = 0$. Given that $V_\pi(\pi_0) \subseteq M_\pi(\pi_0) \subseteq \overline{M_\pi(\pi_0)}$, this implies that $V_{\lambda_{k^\star}}(\pi_0) = \emptyset$ so, by Proposition 4, $\lambda_{k^\star}$ is optimal. ■

As mentioned above, connectivity is an important property for any neighbourhood used in local search. It ensures the non-existence of starting points from which local search cannot reach a global optimum as well as asymptotic convergence in probability to a globally optimal order. Additionally, connectivity would allow for using the neighbourhood structure in the design an exact solving method for fuzzy open shop, of the style of Branch and Bound.

IV. Experimental Results

Having analysed the neighbourhood from a theoretical point of view, the purpose of this section is to provide an experimental evaluation of its behaviour on a varied set of fuzzy open shop problem instances.

Plain hill-climbing algorithms cannot be expected to perform very well on complex problems such as open shop. However, hybrid methods combining a genetic algorithm (GA) with local search (LS) generally improve the quality of results obtained when these methods are used independently (see for instance [12], [18], [32]). The usual approach is to apply local search to every chromosome right after this chromosome has been generated. The resulting algorithm is called a *memetic algorithm* (MA). In [22] we find a genetic algorithm to solve the fuzzy open shop (to our knowledge, the first and only heuristic algorithm of this kind in the literature). Thus, to obtain experimental results for the proposed neighbourhood H, we shall use a memetic algorithm that results from combining this genetic algorithm with simple hill-climbing using H and compare the obtained results with those of the genetic algorithm alone.

In [22], the authors follow [17] and generate a set of fuzzy problem instances from well-known benchmark problems from [33]. Given a crisp problem instance, each crisp processing time t is transformed into a symmetric fuzzy processing time $p(t)$ such that its modal value is $p^2 = t$ and p^1, p^3 are random values, symmetric w.r.t. p^2 and generated so the TFN's maximum range of fuzziness is 30% of p^2. By doing this, the optimal solution to the crisp problem provides a lower bound for the expected makespan of the fuzzified version [17]. Now, the original open shop problem instances are considered to be 'easy' and the results reported in [22] indicate that this is also the case for the fuzzy versions. In [34], a new benchmark set of harder open shop instances was proposed; for our experimental study, we have generated fuzzy versions of these benchmarks, following the same methodology as in [17] and [22]. The original problem instances consist of 6 families, denoted J3, J4,..., J8, of sizes 3×3, 4×4,...,8×8, containing 8 or 9 instances each. From each crisp problem instance 10 fuzzy versions were generated, so in total there are 520 problem instances. The obtained benchmarks for the fuzzy open shop are available at `http://www.aic.uniovi.es/tc/spanish/repository.htm`.

To decide on the parameter values for the GA (and the MA) on this new set of problem instances, a parametric analysis was performed (although it is not reported due to space restrictions), following that of [22]. The chosen configuration for both the GA and MA was: population size, 100; fitness, active scheduling; crossover, pmx with probability 0.70; mutation, insertion with probability 0.05; selection, random pairs; replacement, tournament without repetition (for further detail on these operators, we refer the interested reader to [22] and the references therein). To ensure convergence of the MA, the number of generations was 40, 40, 150, 500, 600 and 700 for sizes 3×3, 4×4,..., 8×8 respectively; the number of iterations of the GA was increased so as to obtain equivalent running times to the MA.

With the above configuration, both algorithms were run 30 times on each problem instance, recording the best, average and worst expected makespan values across the 30 runs. Table I shows a summary of the results, with average values across 30 executions on each the 80–90 instances of the same size (detailed results for each problem would require 520 rows per algorithm). It contains a column for each of the following values: BoB, Best of the Best values, AoB, Average of the Best values, AoA, Average of the Average values, AoW, Average of the Worst values, and WoW, Worst of the Worst values. A lower bound for the expected makespan of all problem instances is 1000, meaning that both the GA and the MA perform equally well on the small (3×3, 4×4) problems, probably without any space for improvement. As the problem size increases, also does increase the difference in solution quality between the MA and the GA, with the former obtaining better results thanks to the local search procedure. This is also illustrated in Figure 5, where the difference in average values of the best, average and worst expected makespan across all executions and problems of the same size are depicted, showing the increasing trend in the difference. It is noticeable both from Table I and Figure 5 that the difference between the MA and the GA also increases from the average of best values to the average of worst values. This suggests that not only does the MA obtain better solutions, but it is also more "reliable" in the sense that there is less variability in quality solution across different executions.

TABLE I
COMPARISON BETWEEN MA AND GA

Problem-	$E[C_{max}]$					Time (s)
Method	BoB	AoB	AoA	AoW	WoW	AoA
J3-AG	1061.4	1063.1	1063.1	1063.1	1066.1	0.154
J3-AM	1061.4	1063.1	1063.1	1063.1	1066.1	0.132
J4-AG	1043.9	1048.8	1050.6	1062.7	1082.5	0.251
J4-AM	1043.9	1048.8	1050.4	1062.1	1075.0	0.234
J5-AG	1024.3	1031.8	1048.1	1070.4	1082.3	1.313
J5-AM	1023.9	1030.8	1044.9	1062.3	1073.7	1.290
J6-AG	1029.4	1038.3	1059.8	1085.2	1095.8	7.691
J6-AM	1026.8	1033.9	1052.1	1074.1	1084.5	7.572
J7-AG	1040.8	1050.7	1077.7	1106.3	1119.0	14.552
J7-AM	1034.9	1044.1	1067.5	1092.2	1105.2	14.458
J8-AG	1040.8	1052.9	1080.9	1110.4	1120.8	26.248
J8-AM	1036.4	1045.5	1068.2	1092.9	1103.1	26.147

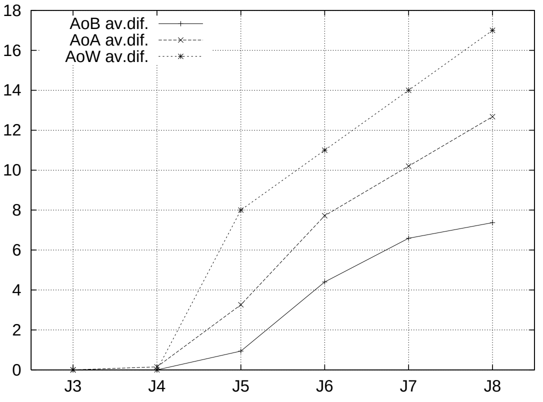


Fig. 5. Difference in average expected makespan between MA and GA.

V. Conclusions

The open shop scheduling problem is a very interesting but also challenging problem, for which few solving methods exist but which is recently gaining the attention of researchers. In order to increase its applicability, it seems necessary to contemplate the existence of ill-known task durations. We assume that this uncertainty is modelled using triangular fuzzy numbers and propose a neighbourhood structure for local search algorithms. This neighbourhood structure is shown to have some interesting properties from the theoretical point of view. Additionally, experimental results illustrate how it provides competitive solutions when combined with a genetic algorithm. The results are obtained on a new set of benchmark instances for the fuzzy open shop, more difficult than the previously existing ones. This should provide a starting point for further work on heuristic methods for open shop and, more generally, scheduling problems with uncertainty, thus narrowing the gap between theoretical work and practical applications.

Acknowledgments

This work is supported by the Spanish Government under grants no. MEC-FEDER Grant TIN2007-67466-C02-01 and and MTM2007-62799.

References

[1] M. L. Pinedo, *Scheduling. Theory, Algorithms, and Systems.*, 3rd ed. Springer, 2008.

[2] C. Guéret and C. Prins, "Classical and new heuristics for the open-shop problem: A computational evaluation." *European Journal of Operational Research*, vol. 107, pp. 306–314, 1998.

[3] C.-F. Liaw, "A hybrid genetic algorithm for the open shop scheduling problem." *European Journal of Operational Research*, vol. 124, pp. 28–42, 2000.

[4] D. Y. Sha and H. Cheng-Yu, "A new particle swarm optimization for the open shop scheduling problem," *Computers & Operations Research*, vol. 35, pp. 3243–3261, 2008.

[5] W. Herroelen and R. Leus, "Project scheduling under uncertainty: Survey and research potentials," *European Journal of Operational Research*, vol. 165, pp. 289–306, 2005.

[6] D. Dubois, H. Fargier, and P. Fortemps, "Fuzzy scheduling: Modelling flexible constraints vs. coping with incomplete knowledge," *European Journal of Operational Research*, vol. 147, pp. 231–252, 2003.

[7] R. Słowiński and M. Hapke, Eds., *Scheduling Under Fuzziness*, ser. Studies in Fuzziness and Soft Computing. Physica-Verlag, 2000, vol. 37.

[8] J. Wang, "A fuzzy robust scheduling approach for product development projects," *European Journal of Operational Research*, vol. 152, pp. 180–194, 2004.

[9] A. Kasperski and M. Kule, "Choosing robust solutions in discrete optimization problems with fuzzy costs," *Fuzzy Sets and Systems*, vol. 160, pp. 667–682, 2009.

[10] P. Brucker and S. Knust, *Complex Scheduling*. Springer, 2006.

[11] G. Celano, A. Costa, and S. Fichera, "An evolutionary algorithm for pure fuzzy flowshop scheduling problems," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 11, pp. 655–669, 2003.

[12] H. Ishibuchi and T. Murata, "A multi-objective genetic local search algorithm and its application to flowshop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics–Part C: Applications and Reviews*, vol. 67, no. 3, pp. 392–403, 1998.

[13] R. Tavakkoli-Moghaddam, N. Safei, and M. Kah, "Accessing feasible space in a generalized job shop scheduling problem with the fuzzy processing times: a fuzzy-neural approach," *Journal of the Operational Research Society*, vol. 59, pp. 431–442, 2008.

[14] M. Sakawa and R. Kubota, "Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy duedate through genetic algorithms," *European Journal of Operational Research*, vol. 120, pp. 393–407, 2000.

[15] S. Petrovic, C. Fayad, D. Petrovic, E. Burke, and G. Kendall, "Fuzzy job shop scheduling with lot-sizing," *Annals of Operations Research*, vol. 159, pp. 275–292, 2008.

[16] I. González Rodríguez, J. Puente, C. R. Vela, and R. Varela, "Semantics of schedules for the fuzzy job shop problem," *IEEE Transactions on Systems, Man and Cybernetics, Part A*, vol. 38, no. 3, pp. 655–666, 2008.

[17] P. Fortemps, "Jobshop scheduling with imprecise durations: a fuzzy approach," *IEEE Transactions of Fuzzy Systems*, vol. 7, pp. 557–569, 1997.

[18] I. González Rodríguez, C. R. Vela, and J. Puente, "A memetic approach to fuzzy job shop based on expectation model," in *Proceedings of IEEE International Conference on Fuzzy Systems, FUZZ-IEEE2007*. London: IEEE, 2007, pp. 692–697.

[19] Q. Niu, B. Jiao, and X. Gu, "Particle swarm optimization combined with genetic operators for job shop scheduling problem with fuzzy processing time," *Applied Mathematics and Computation*, vol. 205, pp. 148–158, 2008.

[20] D. Lei, "Pareto archive particle swarm optimization for multi-objective fuzzy job shop scheduling problems," *International Journal of Advanced Manufacturing Technology*, vol. 37, pp. 157–165, 2008.

[21] T. Konno and H. Ishii, "An open shop scheduling problem with fuzzy allowable time and fuzzy resource constraint," *Fuzzy Sets and Systems*, vol. 109, pp. 141–147, 2000.

[22] J. J. Palacios, J. Puente, C. R. Vela, and I. González-Rodríguez, "A genetic algorithm for the open shop problem with uncertain durations," in *Proceedings of IWINAC 2009, Part I*, ser. Lecture Notes in Computer Science, vol. 5601. Springer, 2009, pp. 255–264.

[23] D. Dubois and H. Prade, *Possibility Theory: An Approach to Computerized Processing of Uncertainty*. New York (USA): Plenum Press, 1986.

[24] S.-M. Chen and T.-H. chang, "Finding multiple possible critical paths using fuzzy PERT," *IEEE Transactions on Systems, Man, and Cybernetics–Part B:*, vol. 31, no. 6, pp. 930–937, 2001.

[25] B. Liu and Y. K. Liu, "Expected value of fuzzy variable and fuzzy expected value models," *IEEE Transactions on Fuzzy Systems*, vol. 10, pp. 445–450, 2002.

[26] S. Chanas and P. Zieliński, "Critical path analysis in the network with fuzzy activity times," *Fuzzy Sets and Systems*, vol. 122, pp. 195–204, 2001.

[27] P. Zieliński, "On computing the latest starting times and floats of activities in a network with imprecise durations," *Fuzzy Sets and Systems*, vol. 150, pp. 53–76, 2005.

[28] S.-P. Chen, "Analysis of critical paths in a project network with fuzzy activity times," *European Journal of Operational Research*, vol. 183, pp. 442–459, 2007.

[29] P. Van Laarhoven, E. Aarts, and K. Lenstra, "Job shop scheduling by simulated annealing," *Operations Research*, vol. 40, pp. 113–125, 1992.

[30] I. González Rodríguez, C. R. Vela, J. Puente, and R. Varela, "A new local search for the job shop problem with uncertain durations," in *Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling (ICAPS-2008)*. Sidney: AAAI Press, 2008, pp. 124–131.

[31] M. Dell' Amico and M. Trubian, "Applying tabu search to the job-shop scheduling problem," *Annals of Operational Research*, vol. 41, pp. 231–252, 1993.

[32] C. R. Vela, R. Varela, and M. A. González, "Local search and genetic algorithm for the job shop scheduling problem with sequence dependent setup times," *Journal of Heuristics*, vol. DOI 10.1007/s10732-008-9094-y, 2009.

[33] E. Taillard, "Benchmarks for basic scheduling problems," *European Journal of Operational Research*, vol. 64, pp. 278–285, 1993.

[34] P. Brucker, J. Hunrink, B. Jurisch, and B. Wöstmann, "A branch & bound algorithm for the open-shop problem," *Discrete Applied Mathematics*, vol. 76, pp. 43–59, 1997.