A genetic solution based on lexicographical goal programming for a multiobjective job shop with uncertainty

ARTICLE *in* JOURNAL OF INTELLIGENT MANUFACTURING · JANUARY 2010

Impact Factor: 1.14 · DOI: 10.1007/s10845-008-0161-x

3 AUTHORS:

Ines Gonzalez Rodriguez
Universidad de Cantabria
41 PUBLICATIONS **146** CITATIONS
SEE PROFILE

Camino Rodríguez Vela
University of Oviedo
62 PUBLICATIONS **287** CITATIONS
SEE PROFILE

Jorge Puente Peinador
University of Oviedo
50 PUBLICATIONS **193** CITATIONS
SEE PROFILE

Journal of Intelligent Manufacturing (2010), 21(1):65--73
DOI 10.1007/s10845-008-0161-x, Springer
Accepted author's version; the final publication is available at link.springer.com

Noname manuscript No.
(will be inserted by the editor)

A Genetic Solution based on Lexicographical Goal Programming for a Multiobjective Job Shop with Uncertainty

Inés González-Rodríguez · Camino R. Vela · Jorge Puente

Abstract In this work we consider a multiobjective job shop problem with uncertain durations and crisp due dates. Ill-known durations are modelled as fuzzy numbers. We take a fuzzy goal programming approach to propose a generic multiobjective model based on lexicographical minimisation of expected values. To solve the resulting problem, we propose a genetic algorithm searching in the space of possibly active schedules. Experimental results are presented for several problem instances, solved by the GA according to the proposed model, considering three objectives: makespan, tardiness and idleness. The results illustrate the potential of the proposed multiobjective model and genetic algorithm.

Keywords Job shop scheduling, Uncertain duration, Multiobjective optimisation

1 Introduction

Scheduling problems form an important body of research since the late fifties, with multiple applications in industry, finance and science (Brucker and Knust, 2006). Part of this research is devoted to fuzzy scheduling, in an attempt to model the uncertainty and vagueness pervading real-world situations. The approaches are diverse, ranging from representing incomplete or vague states of information to using fuzzy priority rules with linguistic qualifiers or preference modelling (Dubois et al., 2003a), (Słowiński and Hapke, 2000).

The complexity of scheduling problems such as job shop means that practical approaches to solving them usually involve heuristic strategies (Brucker and Knust, 2006). In the literature, we find some extensions of these strategies to job shop problems with uncertain durations represented as fuzzy numbers. For instance, genetic algorithms are used in (Sakawa and Kubota, 2000), (Fayad and Petrovic, 2005) and (González Rodríguez et al., 2008) for multiobjective problems while single-objective job shop is approached using simulated annealing in (Fortemps, 1997) and a memetic algorithm, combining a genetic algorithm with local search, in (González Rodríguez et al., 2007b). Far from being trivial, extending heuristic strategies usually requires a significant reformulation of both the problem and solving methods. For example, defining and computing critical paths remains an open question, only partially solved for simple problems (Dubois et al., 2003b).

In the sequel, we describe a job shop problem with uncertain durations and crisp due dates. We propose a generic multiobjective model where the objective function is defined in order to lexicographically minimise the expected values of several fuzzy goals (makespan, tardiness and idleness). In addition to the priority structure for the lexicographical minimisation, target levels for each objective are introduced, in order to balance possibly incompatible goals. The resulting problem is solved by means of a genetic algorithm (GA) based on permutations with repetitions that searches in the space of possibly active schedules. We analyse the performance of the multiobjective GA on a set of problem instances, both based on the expected values of each objective

Corresponding author. Dep. Mathematics, Statistics and Computing, University of Cantabria. Los Castros s/n, Santander, 39005, Spain. E-mail: ines.gonzalez@unican.es; Tel: +34 942202201; Fax: +34 942 201402

A.I. Centre and Dep. of Computer Science, University of Oviedo. Campus de Viesques s/n, Gijón, 33271, Spain. E-mail: crvela@uniovi.es; Tel: +34 985182134

A.I. Centre and Dep. of Computer Science, University of Oviedo. Campus de Viesques s/n, Gijón, 33271, Spain. E-mail: puente@uniovi.es; Tel: +34 985182479

Address(es) of author(s) should be given

and on the quality measures obtained from a semantics for fuzzy schedules presented in (González Rodríguez et al., 2008).

2 Uncertain Processing Times

In real-life applications, it is often the case that the exact duration of a task is not known in advance. However, based on previous experience, an expert may have some knowledge about the duration, thus being able to estimate, for instance, an interval for the possible processing time or its most typical value. In the literature, it is common to use fuzzy intervals to represent such processing times, as an alternative to probability distributions, which require a deeper knowledge of the problem and usually yield a complex calculus.

When there is little knowledge available, 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 a *triangular fuzzy number* or *TFN*, using only an interval $[a^1, a^3]$ of possible values and a single plausible 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:

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

2.1 Operations on Fuzzy Processing Times

Triangular fuzzy numbers and more generally fuzzy intervals have been extensively studied in the literature (cf. (Dubois and Prade, 1988)). A *fuzzy interval* Q is a fuzzy quantity (a fuzzy set on the reals) whose *α-cuts* $Q_\alpha = \{u \in \mathbb{R} : \mu_Q(u) \geq \alpha\}$, $\alpha \in (0, 1]$, are convex, i.e., they are intervals (bounded or not). The *core* of Q contains the elements with full membership $\mu_Q(u) = 1$, and any of its elements is called *modal value*. The *support* of Q is $Q_0 = \{u \in \mathbb{R} : \mu_Q(u) > 0\}$. A *fuzzy number* is a fuzzy quantity whose α-cuts are closed intervals, with compact (i.e. closed and bounded) support and unique modal value.

In order to work with fuzzy numbers, it is necessary to extend the usual arithmetic operations on real numbers. In general, if f is a function $f : \mathbb{R}^2 \to \mathbb{R}$ and Q_1, Q_2 are two fuzzy quantities, the fuzzy quantity $f(Q_1, Q_2)$ is calculated according to the *Extension Principle* as follows:

$$\forall u \in \mathbb{R}, \mu_{f(Q_1,Q_2)}(u) = \sup\{\min(\mu_{Q_1}(w_1), \mu_{Q_2}(w_2)) : f(w_1, w_2) = u\} \quad (2)$$

if $f^{-1}(u) \neq \emptyset$, being equal to 0 otherwise. Computing the above equation is cumbersome, if not intractable. It can be somewhat simplified if M and N are two fuzzy numbers, so the α-cuts M_α and N_α are closed bounded intervals of the form $[\underline{m}_\alpha, \overline{m}_\alpha]$ and $[\underline{n}_\alpha, \overline{n}_\alpha]$, and if f is a continuous isotonic mapping from $\mathbb{R}^2$ into $\mathbb{R}$, that is, if $u \geq u'$ and $v \geq v'$, then $f(u, v) \geq f(u', v')$. In this case, the α-cuts of the fuzzy quantity $f(M, N)$, obtained by applying the Extension Principle, are the images under f of the α-cuts of M and N:

$$\forall \alpha > 0, [f(M, N)]_\alpha = [f(\underline{m}_\alpha, \underline{n}_\alpha), f(\overline{m}_\alpha, \overline{n}_\alpha)] \quad (3)$$

which is a closed interval. Any fuzzy set can be expressed as the union of its α-cuts according to the First Decomposition Theorem, so the above property provides us with an alternative formula for $f(M, N)$:

$$f(M, N) = \cup_{\alpha \in (0,1]} [f(\underline{m}_\alpha, \underline{n}_\alpha), f(\overline{m}_\alpha, \overline{n}_\alpha)] \quad (4)$$

In the job shop, we essentially need two operations on fuzzy durations: the sum and maximum. Additionally, we may need the substraction.

In the case of TFNs, both the addition and substraction are fairly easy to compute, as they are reduced to operating on the three defining points, so for any pair of TFNs M and N, we have the following:

$$M + N = (m^1 + n^1, m^2 + n^2, m^3 + n^3) \quad (5)$$

$$M - N = (m^1 - n^3, m^2 - n^2, m^3 - n^1). \quad (6)$$

Unfortunately, for the maximum of TFNs there is no such simplified expresion. Being an isotonic function, we can use equation (4) above to compute the maximum of two fuzzy numbers. However, in general this still requires an infinite number of computations, because we have to evaluate two maxima for each value $\alpha \in (0, 1]$. For the sake of simplicity and tractability of numerical calculations, we follow Fortemps (Fortemps, 1997) and approximate all results of isotonic algebraic operations on TFNs by a TFN. Instead of evaluating the intervals corresponding to all α-cuts, we evaluate only those intervals corresponding to the support and $\alpha = 1$, which is equivalent to working only with the three defining points of each TFN.

Notice that if we approximate the sum (an isotonic function), the approximation coincides with the sum of TFNs given in (5). The same is not always true for the maximum $\vee$, which would be approximated as follows:

$$M \vee N \sim M \sqcup N = (m^1 \vee n^1, m^2 \vee n^2, m^3 \vee n^3). \quad (7)$$

However, it is possible to prove the following relationship between the maximum and its approximation: let M, N be two TFNs and let $F = N \vee M$ their maximum and $G = N \sqcup M$ its approximated value; it holds that:

$$\forall \alpha \in [0,1], \quad \underline{f}_\alpha \leq \underline{g}_\alpha, \overline{f}_\alpha \leq \overline{g}_\alpha. \tag{8}$$

In particular, F and G have identical support and modal value: $F_0 = G_0$ and $F_1 = G_1$. The approximated maximum can be trivially extended to $n > 2$ TFNs.

2.2 Expected Value of Fuzzy Processing Times

Possibility theory provides a framework to mathematically model scheduling problems with uncertainty (Dubois et al., 1996). For a fuzzy quantity Q, its membership function μ_Q can be interpreted as a possibility distribution on the real numbers, so the *possibility* and *necessity measure* that $Q \leq r$, where r is a real number, are given by:

$$\Pi(\xi \leq r) = \sup_{x \leq r} \mu(x) \quad \mathrm{N}(\xi \leq r) = 1 - \sup_{x > r} \mu(x) \tag{9}$$

Related to the dual measures of possibility and necessity is the *credibility measure* that $Q \leq r$ (Liu, 2006):

$$\mathrm{Cr}(Q \leq r) = \frac{1}{2}(\Pi(Q \leq r) + \mathrm{N}(Q \leq r)) \tag{10}$$

In this case, we have a self-dual measure, i.e. $Cr(Q \leq r) = 1 - Cr(Q > r)$.

The *expected value* of a fuzzy quantity Q is defined on the basis of the credibility measure in (Liu and Liu, 2002):

$$E[Q] = \int_0^\infty \mathrm{Cr}(Q \geq r)dr - \int_{-\infty}^0 \mathrm{Cr}(Q \leq r)dr \tag{11}$$

provided that at least one of the above two integrals is finite. It is easy to prove that the expected value of a TFN A is given by $E[A] = \frac{1}{4}(a^1 + 2a^2 + a^3)$.

The expected value induces a total ordering $\leq_E$ in the set of fuzzy intervals (Fortemps and Roubens, 1996), where for any two fuzzy intervals M, N $M \leq_E N$ if and only if $E[M] \leq E[N]$. Indeed, ranking fuzzy intervals is usually done via defuzzification methods, obtaining a scalar value from a given fuzzy quantity, so ranking fuzzy intervals becomes a matter of ranking their scalar substitutes. The expected value $E[M]$ coincides with the the *neutral scalar substitute* $s(M)$ of a fuzzy interval M (Yager, 1981):

$$s(M) = \frac{1}{2}\int_0^1 (\underline{m}_\alpha + \overline{m}_\alpha)d\alpha. \tag{12}$$

The neutral scalar substitute is cited in (Dubois et al., 2003a) as the most natural defuzzification procedure among those proposed in the literature. This definition can also be obtained using the area compensation method proposed in (Fortemps and Roubens, 1996). Considering the set of all probability functions dominated by the possibility function induced by the membership function μ_M, $E[M]$ is also the expectation of the probability distribution which lies at the centre of gravity of that set. An interesting property is its linearity. Trivially, for any two TFNs $A = (a^1, a^2, a^3)$ and $B = (b^1, b^2, b^3)$, if $\forall i, a^i \leq b^i$, then $A \leq_E B$, but the reverse does not hold.

The expected value of a fuzzy interval will allow us to give an interpretation or model for the fuzzy job shop and it will provide a means of ranking objective values represented by fuzzy intervals, something necessary to select the best solution to the job shop with ill-known durations.

3 The Job Shop Scheduling Problem

The *job shop scheduling problem*, also denoted *JSP*, consists in scheduling a set of jobs $\{J_1, \ldots, J_n\}$ on a set of physical resources or machines $\{M_1, \ldots, M_m\}$, subject to a set of constraints. There are *precedence constraints*, so each job J_i, $i = 1, \ldots, n$, consists of m tasks $\{\theta_{i1}, \ldots, \theta_{im}\}$ to be sequentially scheduled. Also, there are *capacity constraints*, whereby each task θ_{ij} requires the uninterrupted and exclusive use of one of the machines for its whole processing time. In addition, there may be *due-date constraints*, where each job J_i, $i = 1, \ldots, n$, has a maximum completion time D_i and all its tasks must be scheduled to finish before this time. A solution to this problem is a schedule (an allocation of starting times for all tasks) which, besides being *feasible*, in the sense that precedence and capacity constraints hold, is optimal according to some criteria, for instance, that the makespan or the non-fulfillment of due dates are minimal.

3.1 Fuzzy Schedules from Crisp Task Orderings

A schedule s for a job shop problem of size $n \times m$ (n jobs and m machines) may be determined by a decision variable $\boldsymbol{x} = (x_1, \ldots, x_{nm})$ representing a task processing order, where $1 \leq x_l \leq n$ for $l = 1, \ldots, nm$ and $|\{x_l : x_l = i\}| = m$ for $i = 1, \ldots, n$. This is a permutation with repetition as proposed by Bierwirth (Bierwirth, 1995); a permutation of the set of tasks, where each task is represented by the number of its job. A job number appears in such decision variable as many times

as different tasks it has and the order of precedence among tasks requiring the same machine is given by the order in which they appear in the decision variable $\boldsymbol{x}$. Thus, the decision variable represents a task processing order that uniquely determines a feasible schedule. This permutation should be understood as expressing partial orderings for every set of tasks requiring the same machine.

Let us assume that the processing time p_{ij} of each task θ_{ij}, $i = 1, \ldots, n$, $j = 1, \ldots, m$ is a fuzzy variable (a particular case of which are TFNs). The problem may be represented by a matrix of fuzzy processing times $\boldsymbol{\xi}$ such that $\xi_{ij} = p_{ij}$ and a machine matrix $\boldsymbol{\nu}$ such that ν_{ij} is the machine required by task θ_{ij}. For a given task processing order $\boldsymbol{x}$, let $C_i(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})$ denote the completion time of job J_i and let $C_{ij}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})$ denote the completion time of task θ_{ij}, $i = 1, \ldots, n$ $j = 1, \ldots, m$. The completion time of a job is the completion time of its last task, that is, $C_i(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}) = C_{im}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}), \quad i = 1, \ldots, n$. The starting time $S_{ij}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})$ for task θ_{ij}, $i = 1, \ldots, n$, $j = 1, \ldots, m$ will be the maximum between the completion times of the tasks preceding θ_{ij} in its job and its machine. Hence, the starting and completion times of task θ_{ij} are given by:

$$S_{ij}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}) = C_{i(j-1)}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}) \sqcup C_{rs}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}) \tag{13}$$

$$C_{ij}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}) = S_{ij}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}) + p_{ij} \tag{14}$$

where θ_{rs} is the task preceding θ_{ij} in the machine according to the processing order given by $\boldsymbol{x}$. $C_{i0}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})$ is assumed to be zero and, analogously, $C_{rs}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})$ is taken to be zero if θ_{ij} is the first task to be processed in the corresponding machine.

For this fuzzy schedule, we may define the *fuzzy makespan* $C_{max}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})$, the *fuzzy maximum tardiness* (fuzzy tardiness for short) $T_{max}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})$ and the *fuzzy maximum idleness* (fuzzy idleness for short) $I_{max}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})$ as follows:

$$C_{max}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}) = \sqcup_{1 \leq i \leq n} \left(C_i(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})\right) \tag{15}$$

$$T_{max}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}) = \sqcup_{1 \leq i \leq n} \left(C_i(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}) - D_i\right) \vee 0 \tag{16}$$

$$I_{max}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}) = \sqcup_{1 \leq i \leq n} \left(C_{max}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu}) - C_{i_k j_k}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})\right) \tag{17}$$

where $C_{i_k j_k}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})$ is the completion time of the last task to be processed on machine M_k, $k = 1, \ldots, m$, according to the ordering provided by the decision variable $\boldsymbol{x}$.

Let us illustrate the previous definitions with an example. Consider a problem of 3 jobs and 2 machines with the following matrices for fuzzy processing times and machine allocation:

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

For instance, $\xi_{22} = (2,3,4)$ is the processing time of task 2 of job 2 θ_{22} and, given that $\nu_{22} = 1$, this task must be processed on machine 1. Figure 1 shows the Gantt chart (adapted to TFNs) of the schedule given by the decision variable $\boldsymbol{x}$=(1 2 3 2 3 1). It represents the partial schedules obtained from the decision variable for each machine. For machine 1, tasks are processed in the following order: $\theta_{11}, \theta_{22}, \theta_{32}$, and for machine 2, tasks are processed in the order $\theta_{21}, \theta_{31}, \theta_{12}$. Given these orderings and precedence constraints, the starting time for task θ_{22} will be the maximum of the completion times of θ_{21} and θ_{11}, the preceding tasks in the job and in the machine: $S_{22} = C_{21} \sqcup C_{11} = (4,5,6) \sqcup (3,4,7) = (4,5,7)$. Consequently, its completion time will be $C_{22} = S_{22} + \xi_{22} = (4,5,7) + (2,3,4) = (6,8,11)$.

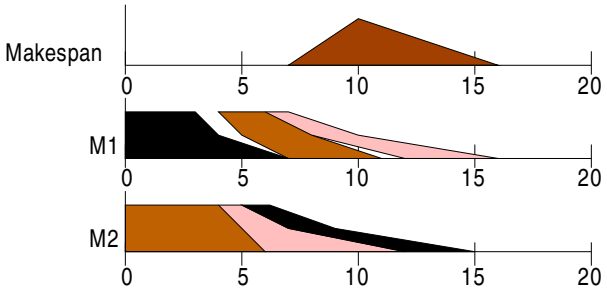

Fig. 1 Gantt chart of the schedule represented by the decision variable (1 2 3 2 3 1)

Notice that if uncertain durations are given as fuzzy intervals the schedule s will be a fuzzy schedule, in the sense that the starting and completion times of all tasks and the makespan are fuzzy intervals. These fuzzy intervals may be seen as possibility distributions on the values that these times may take. However, the task processing ordering represented by $\boldsymbol{x}$ that determines such schedule is crisp; there is no uncertainty regarding the order in which tasks are to be processed. In other words, we obtain a fuzzy schedule from a crisp task ordering. These ideas are essential for the semantics of fuzzy schedules proposed in (González Rodríguez et al., 2008) and described in Section 3.3.

3.2 Multiobjective Models

It is not trivial to optimise a schedule in terms of a fuzzy quantity, since neither the maximum $\vee$ nor its approximation $\sqcup$ define a total ordering. In the literature, this problem is tackled using some ranking method for fuzzy numbers, comparisons based on λ-cuts or defuzzification methods. Here the modelling philosophy is similar to that of stochastic scheduling and is inspired

in the work on expected value models from (Liu and Liu, 2002).

If we only consider the makespan, the expected value $E[C_{max}(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})]$ should be minimised, thus providing an *expected makespan model* for fuzzy job shop (González Rodríguez et al., 2007b). Similarly, we may define the *expected tardiness* and the *expected idleness* models or, in general, an expected model for any single fuzzy goal.

Alternatively, we may consider several objectives and formulate a multiobjective problem. Now, with multiple goals it is often the case that some are achievable only at the expense of others, hence the need of a hierarchy of importance among these possibly incompatible goals so as to satisfy as many as possible in the specified order. In general, for k objectives $f_1, \ldots, f_k$ such priority structure should be established by the decision maker (DM) and may be represented by a one-to-one mapping ρ from $\{f_1, \ldots, f_k\}$ onto $\{1, \ldots, k\}$, such that $\rho(f_i)$ is the priority level of f_i, $i = 1, \ldots, k$, where 1 represents the highest priority. For instance, if $f_1 = C_{max}, f_2 = T_{max}$ and $f_3 = I_{max}$ and the DM considers that the most prioritary objective is minimising the expected tardiness, then $\rho(f_2) = 1$. Without loss of generality, let us assume that the objective functions f_i $i = 1, \ldots, k$ are ordered according to their priority, that is, $\rho(f_i) = i$. Then, we may formulate the following *expected multiobjective model* for the fuzzy job shop problem (FJSP):

$$\begin{cases} \text{lexmin} & (E[f_1(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})], \ldots, E[f_k(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})]) \\ \text{subject to:} & 1 \leq x_l \leq n, \quad l = 1, \ldots, nm, \\ & |\{x_l : x_l = i\}| = m, \quad i = 1, \ldots, n, \\ & x_l \in \mathbb{Z}^+, \quad l = 1, \ldots, nm. \end{cases} \tag{18}$$

where lexmin denotes lexicographically minimising the objective vector.

Additionally, a goal programming model may be used to balance the multiple conflicting objectives, considering target levels established by the DM, so $E[f_i(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})]$ should not exceed a given target value b_i, $i = 1, \ldots, k$. This translates into the following goal constraints:

$$E[f_i(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})] + d_i^- - d_i^+ = b_i, \quad i = 1, \ldots, k \tag{19}$$

where d_i^+, the positive deviation from the target, should be minimised. We thus obtain the following *expected fuzzy goal multiobjective model* for the FJSP:

$$\begin{cases} \text{lexmin} & (d_1^+, \ldots, d_k^+) \\ \text{subject to:} & E[f_i(\boldsymbol{x}, \boldsymbol{\xi}, \boldsymbol{\nu})] + d_i^- - d_i^+ = b_i, \quad i = 1, \ldots, k, \\ & b_i, d_i^-, d_i^+ \geq 0, \quad i = 1, \ldots, k, \\ & 1 \leq x_l \leq n, \quad l = 1, \ldots, nm, \\ & |\{x_l : x_l = i\}| = m, \quad i = 1, \ldots, n, \\ & x_l \in \mathbb{Z}^+, \quad l = 1, \ldots, nm. \end{cases} \tag{20}$$

Notice that (18) is a particular case of (20). Indeed, this last formulation is general enough to comprise all possible fuzzy goals, priority structures and target levels established by the DM. Similar ideas for the fuzzy parallel machine scheduling problem with a fixed priority structure and three objectives can be found in (Peng and Liu, 2004).

3.3 A-Posteriori Semantics of Fuzzy Schedules

In (González Rodríguez et al., 2008), a semantics for the fuzzy schedules was proposed and used to measure the performance of such schedules. According to this semantics, solutions to the FJSP are interpreted as *a-priori solutions*, found when the duration of tasks is not exactly known. In this setting, it is impossible to predict what the exact time-schedule will be, because it depends on the realisation of the task's durations, which is not known yet. Each schedule corresponds to a crisp ordering of tasks and, it is not until tasks are executed according to this ordering that we know their real duration and, hence, obtain a real schedule, the *a-posteriori solution* with crisp job completion times and makespan.

Given this, the main interest of a solution to the FJSP lies in the ordering of tasks that it provides a priori, when information about the problem is incomplete. Ideally, this ordering should yield good schedules in the moment of its practical use, when tasks do have real durations. Hence, its behaviour should be evaluated on a family of N crisp job shop problems, generated from the fuzzy problem so that they can be interpreted as its realisations. Such possible realisations are simulated by generating exact durations for each task at random according to a probability distribution which is coherent with the possibility distribution given by the fuzzy duration.

Given a solution to the FJSP, we consider the ordering of tasks it provides, represented by the deci-

sion variable $\boldsymbol{x}$. For a crisp version of the FJSP, let $\boldsymbol{\eta}$ be the matrix of crisp durations, such that η_{ij}, the a-posteriori duration of task θ_{ij} is coherent with the possibility distribution defined by the fuzzy duration ξ_{ij}. Then, the ordering $\boldsymbol{x}$ can be used by an algorithm of semiactive schedule building to obtain a crisp time-schedule, as presented in Section 3.1 but using real durations instead of fuzzy ones. For such crisp schedule, the *Relative Makespan Error*, ME, is defined as the relative difference in time units between the obtained crisp makespan $C_{max}(\boldsymbol{x}, \boldsymbol{\eta}, \boldsymbol{\nu})$ and a given lower bound for the makespan $LB(\boldsymbol{\eta}, \boldsymbol{\nu})$, that is:

$$ME(\boldsymbol{x}, \boldsymbol{\eta}, \boldsymbol{\nu}) = \frac{C_{max}(\boldsymbol{x}, \boldsymbol{\eta}, \boldsymbol{\nu}) - LB(\boldsymbol{\eta}, \boldsymbol{\nu})}{LB(\boldsymbol{\eta}, \boldsymbol{\nu})} \tag{21}$$

This lower bound may be obtained with some of the existing methods from the literature. We also define the *Feasibility Error*, $F(\boldsymbol{x}, \boldsymbol{\eta}, \boldsymbol{\nu})$, as the proportion of due-date constraints that do not hold for a given ordering $\boldsymbol{x}$ for a given a-posteriori realisation $\boldsymbol{\eta}$ of task durations.

If instead of a single crisp instance we consider the whole family of N crisp problems, each with a duration matrix $\boldsymbol{\eta}_l$, we obtain N values of ME, denoted $ME_l = ME(\boldsymbol{x}, \boldsymbol{\eta}_l, \boldsymbol{\nu})$, and N values of F, denoted $F_l = F(\boldsymbol{x}, \boldsymbol{\eta}_l, \boldsymbol{\nu})$, $l = 1, \ldots, N$. The overall performance of the fuzzy solution across the family of N crisp problems is then measured by the following average values:

$$ME(\boldsymbol{x}) = \frac{\sum_{l=1}^{N} ME_l}{N}, \quad F(\boldsymbol{x}) = \frac{\sum_{l=1}^{N} F_l}{N} \tag{22}$$

We may now compare different solutions to the FJSP based on due-date satisfaction (using $F(\boldsymbol{x})$), based on makespan (using $ME(\boldsymbol{x})$) or even based on the overall achievement of both objectives (using some combination of $F(\boldsymbol{x})$ and $ME(\boldsymbol{x})$). In any case, we should bear in mind the quality of a given ordering $\boldsymbol{x}$ is measured on a family of problems which may be quite diverse. In fact, the greater the uncertainty in the FJSP, the greater the variety of possible crisp realisations and hence, the diversity within the family of associated crisp JSSPs.

4 Using Genetic Algorithms to Solve FJSP

The crisp job shop problem is a paradigm of constraint satisfaction problem and has been approached using many heuristic techniques. In particular, genetic algorithms (GAs) have proved to be a promising solving method (Bierwirth, 1995), (Mattfeld, 1995), (Varela et al., 2003).

The structure of a GA for the FJSP is described in Algorithm 2. First, the initial population is generated and evaluated. Then the GA iterates for a number of generations. In each iteration, a new population is built from the previous one by applying the genetic operators of selection, recombination and acceptation.

Require: an instance of fuzzy JSP, P
Ensure: a schedule s for P
1. Generate the initial population;
2. Evaluate the population;
while No termination criterion is satisfied **do**
3. Select chromosomes from the current population;
4. Apply recombination to the selected chromosomes to generate new ones;
5. Evaluate the chromosomes generated at step 4;
6. Apply the acceptance criterion to the set of chromosomes selected at step 3 together with the chromosomes generated at step 4;
return the schedule from the best chromosome evaluated so far;

Fig. 2 Genetic Algorithm

To codify chromosomes we use the decision variable $\boldsymbol{x}$, a permutation with repetition, which presents a number of interesting characteristics (Varela et al., 2005). The quality of a chromosome is evaluated by the fitness function, which is taken to be the objective function $\text{lexmin}(d_1^+, \ldots, d_k^+)$ as defined in (20).

In the selection phase, chromosomes are grouped into pairs at random. Each of these pairs is mated to obtain two offsprings and acceptance consists in selecting the best individuals from the set formed by the pair of parents and their offsprings. For chromosome mating we consider the *Job Order Crossover* (JOX) (Bierwirth, 1995). Given two parents, JOX selects a random subset of jobs, copies their genes to the offspring in the same positions as they appear in the first parent, and the remaining genes are taken from the second parent so as to maintain their relative ordering. This operator has an implicit mutation effect. Therefore, no explicit mutation operator is actually necessary and parameter setting is considerably simplified, as crossover probability is 1 and mutation probability need not be specified.

From a given decision variable $\boldsymbol{x}$ as explained in Section 3 we may obtain a *semi-active* schedule, meaning that for any operation to start earlier, the relative ordering of at least two tasks must be swapped. However, other possibilities may be considered. For the crisp job shop, it is common to use the G&T algorithm (Giffler and Thomson, 1960), which is an active schedule builder. A schedule is *active* if one task must be delayed for any other one to start earlier. Active schedules are good in average and, most importantly, the space of active schedules contains at least an optimal one, that is, the set of active schedules is *dominant* (cf. (Brucker and

Require: a chromosome $\boldsymbol{x}$ and a fuzzy JSP P
Ensure: the schedule s given by chromosome $\boldsymbol{x}$ for problem P
1: $A = \{\theta_{i1},\ i = 1, \ldots, n\}$; /*set of first tasks of all jobs*/
2: **while** $A \neq \emptyset$ **do**
3: Determine the task $\theta' \in A$ with minimum earliest completion time $C^1_{\theta'}$ if scheduled in the current state;
4: Let M' be the machine required by θ' and $B \subseteq A$ the subset of tasks requiring machine M';
5: Remove from B any task θ that starts later than $C_{\theta'}$: $C^i_{\theta'} \leq S^i_{\theta}$, $i = 1, 2, 3$;
6: Select $\theta^\star \in B$ such that it is the leftmost operation in the sequence $\boldsymbol{x}$;
7: Schedule θ^* as early as possible to build a partial schedule;
8: Remove $\theta^\star$ from A and insert in A the task following $\theta^\star$ in the job if $\theta^\star$ is not the last task of its job;
9: **return** the built schedule;

Fig. 3 Extended G&T for triangular fuzzy times

Knust, 2006)). For these reasons it is worth to restrict the search to this space. Moreover, the G&T algorithm is complete for the job shop problem.

In Algorithm 1 we propose an extension of G&T to the case of fuzzy processing times. It should be noted nonetheless that with uncertain durations we cannot guarantee that the produced schedule will indeed be active when it is actually performed (and tasks have exact durations). We may only say that the obtained fuzzy schedule is *possibly active*.

It often happens that a sequence of tasks is not compatible in order to obtain an active schedule, so the decoding algorithm in Algorithm 1 has to exchange the order of some tasks. This new order is translated to the chromosome, for it to be passed onto subsequent offsprings, thus GA exploiting the so-called lamarckian evolution. Again, an implicit mutation effect is obtained.

The GA described above has been successfully used in (González Rodríguez et al., 2007b) for a single objective job shop to minimise the expected makespan using semi-active schedules, comparing favourably to a simulated annealing algorithm from the literature (Fortemps, 1997). Also the GA combined with the extended G&T improves the expected makespan results obtained by a niche-based GA that follows the scheme proposed in (Sakawa and Kubota, 2000) with matrices of completion times as chromosomes and recombination operators based on fuzzy G&T.

5 Experimental Results

For the experimental results, we follow (Fortemps, 1997) and generate a set of fuzzy problem instances from well-known benchmark problems: FT06 of size 6×6 and LA11, LA12, LA13 and LA14 of size 20×5. This will allow for comparisons with the simulated annealing (SA) algorithm proposed in that paper. From a given crisp processing time x, a symmetric fuzzy processing time $p(x)$ is generated as follows: the modal value is $p^2 = x$ 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. To generate due dates, we use a method proposed in (González Rodríguez et al., 2006). For job J_i, let $\iota_i = \sum_{j=1}^{m} p^2_{ij}$ be the sum of most typical durations across all its tasks, for a given task θ_{ij}, let $\rho_{ij} = \sum_{r \neq i, s \neq j : \nu_{rs} = \nu_{ij}} p^2_{r,s}$ be the sum of most typical durations of all other tasks requiring the same machine as θ_{ij}, and let $\rho_i = \max_{j=1,\ldots,m} \rho_{ij}$ be the maximum of such values across all tasks in job J_i. Then, the due date D_i is a random value from $[\iota_i + 0.5\rho_i, \iota_i + \rho_i]$. In total, 10 instances of fuzzy job shop are generated from each original benchmark problem.

Given the three fuzzy goals $f_1 = C_{max}, f_2 = T_{max}$ and $f_3 = I_{max}$, we consider five objective functions: three single-objective functions given by the expected values $E[f_1], E[f_2]$ and $E[f_3]$ and two multiobjective functions that result from incorporating two different priority structures in expression (20). The first multiobjective function l_{123} corresponds to the priority structure defined by $\rho(i) = i$, that is, the most prioritary goal is the makespan f_1, then the tardiness f_2 and, finally, the idleness f_3. The second objective function l_{213} corresponds to $\rho(f_1) = 2, \rho(f_2) = 1, \rho(f_3) = 3$, i.e. the most prioritary goal is to minimise tardiness, and the makespan becomes the second goal. These hierarchies correspond to probably the most common objectives in the job shop literature, namely minimise makespan or maximise due-date satisfaction.

For each problem instance and objective function, the GA is run 30 times with population size 100 for 200 generations. To fix the target value b_1 for the expected makespan, we use the experience gained using $E[f_1]$ as single objective and set b_1 equal to the average value of $E[f_1]$ across 30 runs of the GA. Target values for expected tardiness and idleness are in all cases $b_2 = b_3 = 0$. Table 1 shows a summary of the results: for each fitness function we measure $E[f_1]$, $E[f_2]$ and $E[f_3]$ for the obtained schedule and compute the best, average and worst of these values across the 30 executions of the GA and the 10 problem instances generated from the same original problem. The optimal makespan value for the original crisp problem is also shown between brackets, as it provides a lower bound for the expected makespan of the fuzzified version (Fortemps, 1997).

From the results in Table 1, it is clear that the multiobjective versions with l_{123} and l_{213} behave similarly to the corresponding single-objective ones, $E[f_1]$

Table 1 Results obtained by the GA

Problem	Fitness	$E[f_1]$			$E[f_2]$			$E[f_3]$		
		Best	Avg	Worst	Best	Avg	Worst	Best	Avg	Worst
FT06 (55)	$E[f_1]$	55.05	55.05	55.05	3.60	4.02	4.33	24.60	25.51	25.85
	$E[f_2]$	58.73	62.40	67.55	0	0	0	17.08	27.52	35.33
	$E[f_3]$	63.18	64.80	70.55	5.83	9.48	15.68	7.05	7.30	10.70
	l_{123}	55.05	55.39	56.28	0.55	1.69	3	22.78	24.66	25.43
	l_{213}	56.90	58.03	58.90	0	0	0	18.10	20.30	27.15
LA11 (1222)	$E[f_1]$	1222	1222	1222	165.05	261.10	342.18	62.83	111.74	148.08
	$E[f_2]$	1257.08	1314.69	1366.08	3.95	5.23	12	109.68	177.53	248.38
	$E[f_3]$	1223.30	1244.71	1294.98	208.78	308.99	408.85	3.98	7.95	13.85
	l_{123}	1222	1222	1222	66.73	72.15	92.78	23.13	50.02	79.33
	l_{213}	1260.40	1300.45	1344.80	5.60	7.94	16.55	82.15	119.22	172.80
LA12 (1039)	$E[f_1]$	1040.13	1040.13	1040.13	140.55	240.54	316.23	41.95	82.80	129.95
	$E[f_2]$	1080.08	1140.30	1192.80	1.98	6.97	17.43	79.85	155.52	216.88
	$E[f_3]$	1041.23	1068.73	1149.08	141.23	286.85	441.30	3.10	7.09	10.70
	l_{123}	1040.13	1040.13	1040.13	31.38	41.51	56.73	16.95	31.94	61
	l_{213}	1081.55	1117.50	1183.80	4.80	12.89	28.40	55.35	98.80	176.35
LA13 (1150)	$E[f_1]$	1150	1150	1150	183.15	252.10	325.50	40.70	84.05	119.45
	$E[f_2]$	1189.55	1240.74	1303.83	0	1.35	2.58	101.40	179.02	253.48
	$E[f_3]$	1153.55	1181.28	1225.05	236.15	321.04	400.65	3.50	5.75	7.40
	l_{123}	1150	1150	1150	57.78	83.48	137.18	28.05	50.12	92.35
	l_{213}	1183	1191.63	1204.65	0	2.42	5.20	73.70	96.03	143.65
LA14 (1292)	$E[f_1]$	1292	1292	1292	230.95	328.89	404.20	81.55	150.73	235.90
	$E[f_2]$	1295.80	1339.04	1402.30	7.25	17.67	31.95	95.60	194.73	273
	$E[f_3]$	1292.65	1310.67	1350.75	249.35	365.62	446.15	4.30	8.35	14.55
	l_{123}	1292	1292	1292	39.4	49.55	78.35	45.96	77.09	126.75
	l_{213}	1297.80	1308.28	1360.20	3.65	10.17	29.2	51.05	86.51	170.10

and $E[f_2]$, regarding their most prioritary goal. Besides, they improve considerably on the remaining goals. Indeed, l_{123} and $E[f_1]$ obtain identical makespan values in all problem instances except those stemming from FT06, where the relative difference with respect to the expected makespan lower bound (55) is less than 1% in average. The expected tardiness values with l_{123} are better than with $E[f_1]$ in all cases. Clearly, minimising the makespan does not always imply minimising tardiness. If we consider the relative values of $E[f_2]$ with respect to the lower bound of the expected makespan (as a means of comparing tardiness values across different problem instances) we see that l_{123} obtains an average reduction of 4.24% in FT06 instances and of 17.73% in LA problem instances. Regarding expected idleness, l_{123} improves in average 1.55% for FT06 instances and 4.65% for LA instances (again, relative to the lower bound for the expected makespan). If we compare l_{213} to $E[f_2]$, expected tardiness is equal for FT06 instances and only 0.8% worse in average for LA instances, while expected makespan improves in average 7.94% and 2.5% for FT06 and LA instances respectively. Expected idleness also improves in both families, with values 13.13% and 6.46% better in average. This illustrates that, despite being the last goal, I_{max} is indeed taken into consideration in the optimisation process when l_{123} and l_{213} are used. Of course, being the last prioritary goal in both cases, it is natural that the expected idleness values for l_{123} and l_{213} are not as good as those obtained with $E[f_3]$.

Notice that the expected tardiness improvement for l_{123} is greater in LA problems than in FT06 instances. This is not surprising since tardiness values obtained with $E[f_1]$ for FT06 are already close to target values. This is not the case for LA instances, where there is greater room for improvement. The same explanation holds for makespan improvement when using l_{213} instead of $E[f_2]$, which is greater for FT06 instances than for LA ones. Notice as well that comparisons between different multiobjective functions do not make sense, as they model different priority requirements.

Let us now compare the GA using l_{123} with the single-objective SA algorithm from (Fortemps, 1997). In that work, 10 problem instances were also gener-

Table 2 Comparison of results for $E[C_{max}]$

Problem	$E[f_1]$ and SA			l_{123} and GA		
	Best	Avg	Worst	Best	Avg	Worst
FT06	55.02	55.2	56.01	55	55.05	55.25
LA11	1222	1222	1222	1222	1222	1222
LA12	1041	1046.81	1056.35	1039	1040.13	1043.25
LA13	1150	1155.07	1181.76	1150	1150	1150
LA14	1292	1292	1292	1292	1292	1292

ated from the same original benchmark problems with the same method but using 6-point fuzzy intervals, a particular case of which are TFNs. Table 2 contains expected makespan results for both methods. It shows the best, average and worst solutions obtained by the GA with l_{123} across the 10 instances generated from the same crisp problem, together with the results reported in (Fortemps, 1997). In Section 4 we already mentioned that the GA optimising only $E[C_{max}]$ compared favourably with the SA algorithm. Table 2 shows that this is also the case for the multiobjective function l_{123} with makespan as its most prioritary goal.

Table 3 Results for the a-posteriori semantics

Problem		$E[f_1]$	l_{123}	$E[f_2]$	l_{213}
FT06	$ME\%$	0.95	1.53	15.29	5.68
	$F\%$	0.00	0.00	0.00	0.00
LA11	$ME\%$	0.03	0.12	6.28	5.16
	$F\%$	4.62	0.00	0.41	0.00
LA12	$ME\%$	0.07	0.23	9.72	6.18
	$F\%$	3.48	0.00	0.78	0.00
LA13	$ME\%$	0.07	0.15	7.40	3.61
	$F\%$	2.68	0.30	0.85	0.00
LA14	$ME\%$	0.02	0.10	3.14	1.97
	$F\%$	3.40	0.00	1.37	0.00

Finally, Table 3 presents the obtained values of the performance measures ME and F based on the a-posteriori semantics presented in Section 3.3. They are average values across the 10 problems of a same family, rescaled as percentage values, obtained with different objective functions: two single-objective functions corresponding to makespan and tardiness and the two multiobjective functions l_{123} and l_{213} where the most prioritary goal is, respectively, the makespan and the tardiness. The results for the a-posteriori semantics, i.e., the behaviour of the task processing order on possible realisations of task durations, coincide with the results for the expected objective values in Table 1 and further support the corresponding analysis: the multiobjective versions with l_{123} and l_{213} behave similarly to the corresponding single-objective ones, $E[f_1]$ and $E[f_2]$, regarding their most prioritary goal, whilst improving on the secondary goal. If we compare the multiobjective function l_{123} to $E[f_1]$, we see that the ME increases in average less than 0.2%, whilst F is considerably reduced. In fact, F becomes null in all cases except LA13, where it goes from 2.68% al 0.3%. Comparing l_{213} to $E[f_2]$, the multiobjective version clearly outperforms the single objective one: not only do relative makespan errors ME improve considerably (up to 10%), but due-date fulfilment is also better or equal in all cases. In fact, in all cases but one the a-posteriori schedules obtained with multiobjective optimisation fully satisfy the due dates. There seems to be a clear synergy effect among different goals in the multiobjective approach.

6 Conclusions and Future Work

We have considered a job shop problem with uncertain durations. Such uncertainty is modelled using TFNs and the goal is to find a task processing order that yields a feasible schedule optimising several objectives, for instance, fuzzy makespan, fuzzy tardiness and fuzzy idleness. We have proposed to formulate the multiobjective problem as a fuzzy goal programming model according to a generic priority structure and target levels established by the decision maker, using the expected value of the fuzzy quantities. As solving method, a GA with codification based on permutations with repetitions has been described. Experimental results on fuzzy versions of well-known crisp problem instances illustrate the potential of both the proposed multiobjective formulation and the GA. This is further illustrated with experimental results that incorporate the semantics of fuzzy schedules proposed in (González Rodríguez et al., 2008)

In the future, the multiobjective approach will be further analysed using a more varied set of problem instances. This wider set of problems should also enable a thorough parametric analysis of the target values established by the decision maker. Finally, the GA may be hybridised with other heuristic techniques, such as local

search, to increase its potential. This leads to further studying task criticality for fuzzy durations.

Acknowledgements

All authors are supported by MEC-FEDER Grant TIN2007-67466-C02-01. A preliminary version of this work was presented at the Workshop on Planning, Scheduling and Constraint Satisfaction held in conjunction with CAEPIA 2007 (González Rodríguez et al., 2007a).

References

Bierwirth, C. (1995). A generalized permutation approach to jobshop scheduling with genetic algorithms. *OR Spectrum*, 17:87–92.

Brucker, P. and Knust, S. (2006). *Complex Scheduling*. Springer.

Dubois, D., Fargier, H., and Fortemps, P. (2003a). Fuzzy scheduling: Modelling flexible constraints vs. coping with incomplete knowledge. *European Journal of Operational Research*, 147:231–252.

Dubois, D., Fargier, H., and Galvagnon, V. (2003b). On latest starting times and floats in activity networks with ill-known durations. *European Journal of Operational Research*, 147:266–280.

Dubois, D., Fargier, H., and Prade, H. (1996). Possibility theory in constraint satisfaction problems: Handling priority, preference and uncertainty. *Applied Intelligence*, 6:287–309.

Dubois, D. and Prade, H. (1988). *Possibility Theory: An Approach to Computerized Processing of Uncertainty*. Plenum Press, New York (USA).

Fayad, C. and Petrovic, S. (2005). A fuzzy genetic algorithm for real-world job-shop scheduling. *Innovations in Applied Artificial Intelligence, Lecture Notes in Computer Science*, 3533:524–533.

Fortemps, P. (1997). Jobshop scheduling with imprecise durations: a fuzzy approach. *IEEE Transactions of Fuzzy Systems*, 7:557–569.

Fortemps, P. and Roubens, M. (1996). Ranking and defuzzification methods based on area compensation. *Fuzzy Sets and Systems*, 82:319–330.

Giffler, B. and Thomson, G. L. (1960). Algorithms for solving production scheduling problems. *Operations Research*, 8:487–503.

González Rodríguez, I., Puente, J., Vela, C. R., and Varela, R. (2008). Semantics of schedules for the fuzzy job shop problem. *IEEE Transactions on Systems, Man and Cybernetics, Part A*. Accepted for publication.

González Rodríguez, I., Vela, C. R., and Puente, J. (2006). Study of objective functions in fuzzy job-shop problem. *ICAISC 2006, Lecture Notes in Artificial Intelligence*, 4029:360–369.

González Rodríguez, I., Vela, C. R., and Puente, J. (2007a). A genetic solution for multiobjective fuzzy job shop based on lexicographical goal programming. In Salido, M. A. and Fdez-Olivares, J., editors, *Proc. of the Workshop on Planning, Scheduling and Constraint Satisfaction*, pages 93–104, Salamanca (Spain).

González Rodríguez, I., Vela, C. R., and Puente, J. (2007b). A memetic approach to fuzzy job shop based on expectation model. In *Proceedings of IEEE International Conference on Fuzzy Systems, FUZZ-IEEE2007*, pages 692–697, London.

Liu, B. (2006). A survey of credibility theory. *Fuzzy Optimization and Decision Making*, 5:387–408.

Liu, B. and Liu, Y. K. (2002). Expected value of fuzzy variable and fuzzy expected value models. *IEEE Transactions on Fuzzy Systems*, 10:445–450.

Mattfeld, D. C. (1995). *Evolutionary Search and the Job Shop Investigations on Genetic Algorithms for Production Scheduling*. Springer-Verlag.

Peng, J. and Liu, B. (2004). Parallel machine scheduling models with fuzzy processing times. *Information Sciences*, 166:49–66.

Sakawa, M. and Kubota, R. (2000). Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy duedate through genetic algorithms. *European Journal of Operational Research*, 120:393–407.

Słowiński, R. and Hapke, M., editors (2000). *Scheduling Under Fuzziness*, volume 37 of *Studies in Fuzziness and Soft Computing*. Physica-Verlag.

Varela, R., Serrano, D., and Sierra, M. (2005). New codification schemas for scheduling with genetic algorithms. *Lecture Notes in Computer Science*, 3562:11–20.

Varela, R., Vela, C. R., Puente, J., and Gómez, A. (2003). A knowledge-based evolutionary strategy for scheduling problems with bottlenecks. *European Journal of Operational Research*, 145:57–71.

Yager, R. R. (1981). A procedure for ordering fuzzy subsets of the unit interval. *Information Sciences*, 24:143–161.