A Study of Schedule Robustness for Job Shop with Uncertainty

Inés González-Rodríguez[1], Jorge Puente[2], Ramiro Varela[2], and Camino R. Vela[2]

[1] Department of Mathematics, Statistics and Computing, University of Cantabria, (Spain) ines.gonzalez@unican.es
[2] A.I. Centre and Department of Computer Science, University of Oviedo, (Spain) {puente,ramiro,crvela}@uniovi.es, http://www.aic.uniovi.es/Tc

Abstract. We consider a job shop problem with uncertain processing times modelled as triangular fuzzy numbers and propose a methodology to study solution robustness with respect to different perturbations in the durations. This methodology is applied to obtain experimental results for several problem instances, using a hybrid genetic algorithm that minimises the expected makespan. We conclude that taking into account the uncertainty information provided by fuzzy numbers produces proactive solutions, coping well with posterior changes in processing times.

1 Introduction

Scheduling problems form an important body of research since the late fifties, with multiple applications in industry, finance and science [1]. Traditionally, scheduling has been treated as a deterministic problem that assumes precise knowledge of all data. However, modelling real-world problems usually involves processing uncertainty and flexibility. In the literature we find different proposals for dealing with uncertainty in scheduling [2], either finding solutions which adapt dynamically to changes or incorporating available knowledge about possible changes to the solution. Fuzzy scheduling comprises diverse approaches to dealing with both uncertainty and flexibility, ranging from representing incomplete or vague states of information to using fuzzy priority rules with linguistic qualifiers or preference modelling [3],[4].

The complexity of scheduling problems such as job shop means that practical approaches to solving them usually involve heuristic strategies [1]. Extending these strategies to problems with uncertain durations represented as fuzzy numbers usually requires a significant reformulation of both the problem and solving methods. Proposals from the literature include genetic algorithms [5],[6],[7], simulated annealing [8] and genetic algorithms hybridised with local search [9].

In the sequel, we concentrate on a job shop problem where uncertain task durations are given as fuzzy numbers. Based on a semantics of fuzzy schedules from the literature, we propose a new methodology to test the robustness of the obtained schedule with respect to posterior changes in task durations. Following this methodology, experimental results are obtained for modified benchmark problems using a hybrid algorithm, illustrating how the use of fuzzy numbers allows for robust proactive solutions.

2 Job Shop Scheduling with Uncertain Durations

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. 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 is minimal.

2.1 Uncertain Durations

In real-life applications, it is often the case that the exact duration of a task, i.e. the time it takes to be processed, is not known in advance, and only some uncertain knowledge is available. Such knowledge can be modelled using a *triangular fuzzy number* or TFN, given by an interval $[n^1, n^3]$ of possible values and a modal value n^2 in it. For a TFN N, denoted $N = (n^1, n^2, n^3)$, the membership function takes the following triangular shape:

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

In the job shop, we essentially need two operations on fuzzy numbers, the sum and the maximum. These are obtained by extending the corresponding operations on real numbers using the *Extension Principle*. However, computing the resulting expression is cumbersome, if not intractable. For the sake of simplicity and tractability of numerical calculations, we follow [8] and approximate the results of these operations, evaluating the operation only on the three defining points of each TFN. It turns out that for any pair of TFNs M and N, the approximated sum $M + N \approx (m^1 + n^1, m^2 + n^2, m^3 + n^3)$ coincides with the actual sum of TFNs; this may not be the case for the maximum $M \vee N \approx (m^1 \vee n^1, m^2 \vee n^2, m^3 \vee n^3)$, although they have identical support and modal value.

The membership function of a fuzzy number can be interpreted as a possibility distribution on the real numbers. This allows to define its expected value [10], given for a TFN N by $E[N] = \frac{1}{4}(n^1 + 2n^2 + n^3)$. It coincides with the neutral scalar substitute of a fuzzy interval and the centre of gravity of its mean value [3]. It induces a total ordering $\leq_E$ in the set of fuzzy numbers [8], where for any two fuzzy numbers M, N $M \leq_E N$ if and only if $E[M] \leq E[N]$.

2.2 Fuzzy Job Shop Scheduling

For a job shop problem instance of size $n \times m$ (n jobs and m machines), let $\boldsymbol{p}$ be a duration matrix and let $\boldsymbol{\nu}$ be a machine matrix such that p_{ij} is the processing time of task θ_{ij} and ν_{ij} is the machine required by θ_{ij}, $i = 1, \ldots, n$, $j = 1, \ldots, m$. Let σ be a feasible task processing order, i.e., a lineal ordering of tasks which is compatible with a processing order of tasks that may be carried out so that all constraints hold. A feasible schedule may be derived from σ using a *semi-active schedule builder*: if $S_{ij}(\sigma, \boldsymbol{p}, \boldsymbol{\nu})$ and $C_{ij}(\sigma, \boldsymbol{p}, \boldsymbol{\nu})$ denote respectively the starting and completion times of task θ_{ij}, these times are given by:

$$S_{ij}(\sigma, \boldsymbol{p}, \boldsymbol{\nu}) = C_{i(j-1)}(\sigma, \boldsymbol{p}, \boldsymbol{\nu}) \vee C_{rs}(\sigma, \boldsymbol{p}, \boldsymbol{\nu}) \quad (2)$$

$$C_{ij}(\sigma, \boldsymbol{p}, \boldsymbol{\nu}) = S_{ij}(\sigma, \boldsymbol{p}, \boldsymbol{\nu}) + p_{ij} \quad (3)$$

where θ_{rs} is the task preceding θ_{ij} in the machine according to the processing order σ, $C_{i0}(\sigma, \boldsymbol{p}, \boldsymbol{\nu})$ is assumed to be zero and, analogously, $C_{rs}(\sigma, \boldsymbol{p}, \boldsymbol{\nu})$ is taken to be zero if θ_{ij} is the first task to be processed in the corresponding machine. The completion time of job J_i will then be $C_i(\sigma, \boldsymbol{p}, \boldsymbol{\nu}) = C_{im}(\sigma, \boldsymbol{p}, \boldsymbol{\nu})$ and the *makespan* $C_{max}(\sigma, \boldsymbol{p}, \boldsymbol{\nu})$ is the maximum completion time of all jobs:

$$C_{max}(\sigma, \boldsymbol{p}, \boldsymbol{\nu}) = \vee_{1 \leq i \leq n} \left(C_i(\sigma, \boldsymbol{p}, \boldsymbol{\nu}) \right) \quad (4)$$

For the sake of a simpler notation, we may write $C_{max}(\sigma)$ when the problem (hence $\boldsymbol{p}$ and $\boldsymbol{\nu}$) is fixed or even C_{max} when no confusion is possible.

If task processing times are TFNs, the resulting schedule is fuzzy in the sense that starting and completion times and the makespan are TFNs. Each TFN can be seen as a possibility distributions on the values that the time may take. Notice however that there is no uncertainty regarding the task processing ordering σ that determines the schedule. To illustrate these ideas, consider a problem of 3 jobs and 2 machines with the following matrices for fuzzy processing times and machine allocation:

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

Figure 1 shows the Gantt chart (adapted to TFNs following [8]) of the schedule given by the task order $\theta_{11}, \theta_{21}, \theta_{31}, \theta_{22}, \theta_{12}, \theta_{32}$. For instance, for task θ_{22} we have $S_{22} = C_{21} \vee C_{11} = (4,5,7)$ and $C_{22} = S_{22} + p_{22} = (6,8,11)$, meaning that θ_{22} completion time will be between 6 and 11, with 8 as most pausible value.

Since we may build a feasible schedule from a feasible task processing order, we restate the goal of the job shop problem as that of finding an optimal task processing order, in the sense that the makespan for the derived schedule is optimal. It is not trivial to optimise a fuzzy makespan, since neither the maximum nor its approximation define a total ordering in the set of TFNs. Following [9], we use the total ordering provided by the expected value and consider that the objective is to minimise the expected makespan $E[C_{max}(\sigma, \boldsymbol{p}, \boldsymbol{\nu})]$.

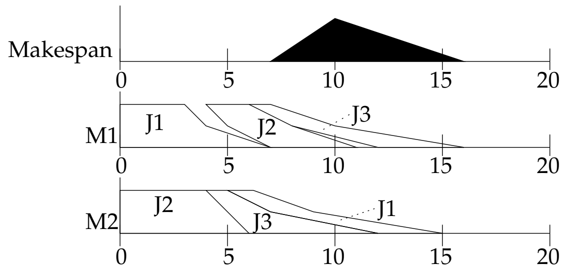

Fig. 1. Gantt chart of the schedule represented by $\theta_{11}, \theta_{21}, \theta_{31}, \theta_{22}, \theta_{12}, \theta_{32}$

2.3 A Hybrid Algorithm

Evolutionary strategies have been shown to perform well in presence of uncertainty [11]. In particular, hybrid methods combining a genetic algorithm (GA) with local search (LS) have proved successful for the fuzzy flow shop problem [12] and the fuzzy job shop problem [9]. For the latter, experimental results show a clear synergy between the GA and the LS, with the hybrid method also comparing favourably with other heuristic methods from the literature [8], [5].

The *genetic algorithm* proposed in [9] uses permutations with repetition as chromosomes. These are decodified using an extension of G&T algorithm to fuzzy durations, with the fitness function being the expected makespan of the obtained schedule. Pairs of chromosomes are selected using tournament and mated using job order crossover (JOX) to obtain two offsprings; acceptance consists in selecting the best individuals from both parents and their offsprings. JOX operator has an implicit mutation effect and therefore, no explicit mutation operator is actually introduced, with the consequent simplification of parameter setting, as crossover and mutation probabilities being 1 and 0 respectively.

The *local search procedure* is applied to every chromosome immediately after its generation. Starting from this given processing order, its neighbourhood is calculated and neighbours are evaluated using the aforementioned semiactive schedule builder in the search of an improving solution. The selection criterion is *steepest descent hill-climbing*, i.e. all neighbours are evaluated, selecting the best one, which replaces the original solution if it is an improving neighbour (with a smaller $E[C_{max}]$). Local search starts again from that improving neighbour, so the procedure finishes when no neighbour satisfies the acceptance criterion. In [9] it is proposed that the neighbourhood structure be an extension of that defined for the crisp job shop in [13]. Here, a *move* for a feasible task processing order σ is defined as the change in the order of a pair of consecutive tasks (x, y) in a critical block. The *neighbourhood* of σ, denoted $H(\sigma)$, is defined as the set of processing orders obtained from σ after applying all possible moves.

A key aspect then for the neighbourhood is the definition of critical block in the fuzzy case. In general, a job shop problem instance may be represented by a directed graph $G = (V, A \cup D)$, where each node in the set V represents a task of the problem, except for the dummy nodes *start* and *end*, representing tasks with null processing times. Arcs in A, or *conjunctive arcs*, represent precedence constraints (including arcs from node *start* to the first task of each job and arcs form the last task of each job to node *end*). Arcs in D, called *disjunctive arcs*, represent capacity constraints: set D is partitioned into subsets D_i, where D_i corresponds to machine M_i and includes an arc for each pair of tasks requiring that machine. Each arc is weighted with the processing time of the task at the source node (a TFN in our case).

A feasible task processing order σ is represented by a *solution graph*, an acyclic subgraph of G, $G(\sigma) = (V, A \cup R(\sigma))$, where $R(\sigma) = \cup_{i=1 \dots m} R_i(\sigma)$, $R_i(\sigma)$ being a hamiltonian selection of D_i. In the fuzzy case, from $G(\sigma)$ we obtain the *parallel solution graphs* $G^i(\sigma)$, $i = 1, 2, 3$, which are identical to $G(\sigma)$ except for the cost of arc $(x, y) \in A \cup R(\sigma)$, which for graph $G^i(\sigma)$ will be the i-th defining point of p_x, that is, p_x^i. Each parallel solution graph $G^i(\sigma)$ is a disjunctive graph with crisp arc weights, so in each of them a critical path is the longest path from node *start* to node *end*. For the fuzzy solution graph $G(\sigma)$, a path will be considered to be *critical* if and only if it is critical in some $G^i(\sigma)$. Nodes and arcs in a critical path are termed critical and a critical path is naturally decomposed into critical blocks $B_1, \dots, B_r$, where a *critical block* is a maximal subsequence of operations of a critical path requiring the same machine.

3 Robustness of Processing Orders

According to a classification of the most representative situations of uncertainty in optimisation problems given in [11], the uncertainty addressed in the fuzzy job shop problem lies in the robustness category, where design variables are subject to perturbations or changes after the optimal solution has been determined. A common requirement in this case is that a solution should still work satisfactorily when the design variables change slightly, for instance, due to manufacturing tolerances. Solutions fulfilling this requirement are termed *robust solutions*. Several methods have been proposed to deal with duration uncertainty in the framework of scheduling problems ([4], [8], [5]) but less effort has been dedicated to analyse the robustness of solutions ([14], [6]). Here, we propose a new method to study robustness for the fuzzy job shop problem, based on a numerical analysis and inspired by the semantics for fuzzy schedules from [7].

3.1 Semantics of Fuzzy Schedules

In [7] solutions to the fuzzy job shop 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 fuzzy 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, know the exact schedule, the *a-posteriori solution* with crisp job completion times and makespan. The practical interest of a solution to the fuzzy job shop would lie in the ordering of tasks that it provides a priori using the available incomplete information, which should yield good schedules in the moment of its practical use. Its behaviour could therefore be evaluated on a family of N crisp job shop problems, representing a posteriori realisations of the fuzzy problem. Such possible realisations are simulated by generating an exact duration for each task at random according to a probability distribution which is coherent with the fuzzy duration (namely the possibility distribution it provides normalised so the additivity axiom holds).

Given a solution to the fuzzy job shop, consider the task processing order it provides σ_F. For a crisp version of the problem, 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 constraint imposed by the fuzzy duration p_{ij}. The ordering σ_F can be used by an algorithm of semiactive schedule building as presented above, using the exact durations $\boldsymbol{\eta}$ instead of fuzzy ones, to obtain a time-schedule with a crisp makespan $C_{max}(\sigma_F, \boldsymbol{\eta}, \boldsymbol{\nu})$. If instead of a single crisp instance we consider the whole family of N crisp problems, each with a duration matrix, we obtain N makespan values, and its average gives a measure of the overall performance of the fuzzy solution across the family of N crisp problems.

3.2 A Methodology for Robustness Analysis

In [2], the authors distinguish between five approaches to dealing with uncertainty in a scheduling environment where the evolution structure of the precedence network is deterministic: reactive scheduling, stochastic scheduling, scheduling under fuzziness, proactive (robust) scheduling, and sensitivity analysis. The fuzzy job shop approach falls clearly into the third category. However, we shall argue that it also falls in the proactive or robust scheduling category. *Proactive scheduling* constructs a predictive schedule that accounts for statistical knowledge of uncertainty, used to make the predictive schedule more robust, i.e., insensitive to disruptions. Even if the information about the uncertainty is not of stochastic nature, the fuzzy job shop approach is still proactive in the sense that the built schedule (the obtained task processing order) also accounts for the uncertain knowledge available and should therefore be less sensitive to perturbations in task durations. Hence we propose to test such robustness in comparison to a simpler approach which does not take into account the available albeit uncertain knowledge.

The uncertainty in processing times is modelled using TFNs with a single modal value. Provided that the membership function is symmetric, this value coincides with the TFN's expected value. This suggests reducing the fuzzy problem to a crisp one where task durations are given by their most typical value. This approach, based on defuzzification, is pretty standard and has the advantage of reducing the problem to a less complex and better known one. It also has the potential disadvantage of losing some information, which may reflect in a loss of robustness.

Let σ_F and σ_C denote feasible task processing orders obtained respectively for the fuzzy job shop problem and the crisp one where each task is allocated its most typical duration. We propose to compare both orderings in terms of robustness, when durations are subject to perturbations, independently of the method of resolution used. The comparison will take place under four different situations. The first one corresponds to the above semantics, testing their performance on a set T_1 of N crisp instances (possible realisations) obtained as a random sample of the probability distributions derived from the TFNs. The remaining situations correspond to "extreme" crisp instances, modelling situations where, either the expert has been very conservative when estimating the most typical duration, so the actual processing time is in general lower, or the expert has been too optimistic, so there are significant delays w.r.t. the most typical duration. First, we generate a set T_2 of N crisp instances where, for each task with fuzzy duration (p^1, p^2, p^3), its realisation (crisp duration) is selected at random from the interval $[p^1, p^1 + 0.25(p^3 - p^1)]$ (i.e. the first "quarter" of all possible durations). Analogously, another set T_3 of N crisp instances is generated so, for each task, its possible crisp duration is selected at random from $[p^1 + 0.75(p^3 - p^1), p^3]$. A final set T_4 of N crisp instances is obtained as a mixture of conservative and optimistic estimations, with each task duration selected at random from $[p^1, p^1 + 0.25(p^3 - p^1)] \cup [p^1 + 0.75(p^3 - p^1), p^3]$. For every crisp instance, both task orderings σ_F and σ_C can be applied to obtain a makespan value; the average and standard deviation of these values across each set of N crisp instances will provide a means of comparing both orderings. If the use of TFNs throughout the solving process is an adequate approach to taking into account uncertain information, then solution σ_F should behave better when faced with these perturbations than σ_C and, hence, it would be considered robust.

4 Experimental Results

For the experimental results, we consider 12 well-known benchmark problems for job shop: FT10 (size 10×10), FT20 (20×5), La21, La24 and La25 (15×10), La27 and La29 (20×10), La38 and La40 (15×15), and ABZ7, ABZ8 and ABZ9 (20×15). Fuzzy versions of each benchmark are generated following [8] and [9], so task durations become symmetric TFNs where the modal value is the original duration, ensuring that the optimal solution to the crisp problem provides a lower bound for the fuzzified version. The hybrid genetic algorithm is run 30 times for each of the 12 fuzzy job shop instances with population size 100 and for 200 generations. Another 30 runs are performed for each of the 12 crisp job shop instances, with the same population size and an extended number of generations to obtain equivalent CPU times and allow for comparisons.

Table 1 presents a first comparison between both approaches, showing the fuzzy makespan values $C_{max}(\sigma^*_F, \boldsymbol{p})$ and $C_{max}(\sigma^*_C, \boldsymbol{p})$ obtained with both orderings on the problem with fuzzy durations. It illustrates the convergence of the hybrid algorithm minimising the expected makespan, since the solution found with this method for the fuzzy problem (σ^*_F) does obtain lower expected makespan than a different ordering σ^*_C, even if in some cases the most typical makespan value C^2_{max} is greater for σ^*_F than for σ^*_C.

Table 1. Initial comparison between σ_F^* and σ_C^*

Problem	$C_{max}(\sigma_F^*, \boldsymbol{p})$	$E[C_{max}(\sigma_F^*, \boldsymbol{p})]$	$C_{max}(\sigma_C^*, \boldsymbol{p})$	$E[C_{max}(\sigma_C^*, \boldsymbol{p})]$
FT10	(874, 935, 1003)	936.7	(893, 930, 999)	938.0
FT20	(1090,1165,1241)	1165.2	(1094,1165,1243)	1166.7
La21	(988,1052,1130)	1055.5	(992,1053,1135)	1058.2
La24	(872,939,1012)	940.5	(894,938,1007)	944.2
La25	(919,977,1059)	983	(923,977,1076)	988.2
La27	(1171,1249,1340)	1252.2	(1176,1254,1362)	1261.5
La29	(1107,1183,1262)	1183.7	(1112,1175,1264)	1181.5
La38	(1135,1215,1303)	1217	(1148,1215,1312)	1222.5
La40	(1145,1233,1324)	1233.7	(1160,1228,1328)	1236
ABZ7	(645, 675, 715)	677.5	(646, 680, 732)	684.5
ABZ8	(652,684,728)	687	(656,688,731)	690.7
ABZ9	(672,704,742)	705.5	(669,704,749)	706.5

Following the proposed methodology, for each fuzzy problem instance and each execution of the algorithm, the obtained task processing order σ_F is tested on the four sets T_1 to T_4 of perturbations, with $N = 1000$. We proceed analogously, for the ordering σ_C for the corresponding crisp problem. Figure 2 presents a boxplot of the makespan values obtained using both σ_F and σ_C on each set of 1000 perturbations for instance ABZ9, averaged across the 30 runs of the hybrid algorithm. It shows that σ_F compares favourably to σ_C. Similar plots are obtained for all large problems.

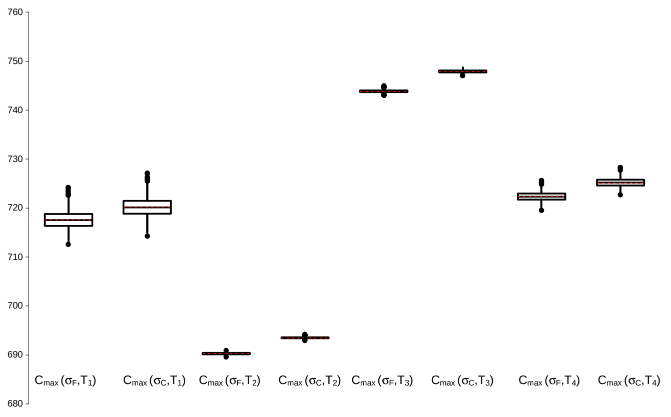

Fig. 2. Boxplot of crisp makespan values obtained with σ_F and σ_C on the four sets of possible perturbations of ABZ9.

Table 2 contains a summary of results obtained with both processing orders σ_F and σ_C for each of the twelve problems with the average and standard deviation of makespan values across each set of 1000 perturbations, also averaged

Table 2. Results for the robustness analysis

		T_1	T_2	T_3	T_4
FT10	σ_F	943.8±5.34	891.7±0.41	960.8±1.37	951.0±2.21
	σ_C	941.6±4.70	901.3±0.39	988.5±0.60	949.5±2.00
FT20	σ_F	1178.9±6.29	1117.3±0.50	1198.8±1.40	1185.4±2.38
	σ_C	1179.4±5.84	1117.9±0.50	1234.3±0.56	1186.6±2.38
La21	σ_F	1064.0±6.22	1003.3±0.51	1114.8±0.57	1073.9±2.30
	σ_C	1064.6±6.15	1008.5±0.43	1118.8±0.66	1075.3±2.31
La24	σ_F	951.4±4.47	901.2±0.45	967.7±1.16	959.0±1.86
	σ_C	955.7±3.99	906.9±0.41	1000.0±0.52	964.0±1.84
La25	σ_F	990.1±5.32	941.2±0.47	1038.7±0.60	999.8±2.23
	σ_C	991.6±6.06	945.9±0.43	1045.8±0.69	1000.6±2.30
La27	σ_F	1269.6±3.37	1193.8±0.58	1337.2±0.67	1280.9±2.36
	σ_C	1274.4±5.48	1203.0±0.48	1345.1±0.72	1286.3±2.26
La29	σ_F	1205.7±5.13	1145.6±0.46	1261.3±0.58	1215.8±2.11
	σ_C	1211.6±5.27	1153.3±0.42	1272.1±0.69	1222.6±2.14
La38	σ_F	1235.1±6.19	1170.6±0.56	1293.4±0.39	1245.8±2.56
	σ_C	1239.2±5.44	1177.4±0.50	1301.3±0.52	1251.6±2.53
La40	σ_F	1246.0±5.68	1173.5±0.53	1306.4±0.59	1256.9±2.19
	σ_C	1247.9±5.09	1179.7±0.52	1313.6±0.71	1260.1±2.26
ABZ7	σ_F	687.6±2.35	655.4±0.23	717.6±0.27	692.2±1.00
	σ_C	690.2±2.06	661.3±0.20	721.3±0.28	695.2±0.98
ABZ8	σ_F	701.4±2.17	671.9±0.23	729.3±0.33	706.4±1.01
	σ_C	704.1±2.12	677.0±0.20	733.6±0.33	709.9±0.97
ABZ9	σ_F	717.6±1.82	690.3±0.22	743.8±0.25	722.3±0.91
	σ_C	720.2±1.93	693.5±0.21	747.9±0.28	725.2±0.92

across the 30 runs of the hybrid algorithm. Overall, it is clear that the task ordering produced using fuzzy durations, σ_F, performs better than the task ordering σ_C found when only the most typical duration was considered.

By using TFNs, the hybrid algorithm, makes a joint search effort, trying to optimise makespan values across the three solution graphs $G^i(\sigma)$ in parallel. In some problem instances (such as ABZ or La29), the consequence is that the value of $E[C_{max}(\sigma_F)]$ is lower than the crisp makespan $C_{max}(\sigma_C)$ obtained with the defuzzified problem. In other instances (FT20, La21, La38), this is not the case; it could be thought that optimising three components in parallel penalises the global solution when compared to optimising only the most likely duration. In the first case, the improvement in terms of robustness of the solution when TFNs are used instead of crisp defuzzified durations is clear. Most interestingly, in the second case the experimental results in Table 2 show that the effort in maintaining an equilibrium among the three components of the TFNs has translated in greater robustness compared to the solution offered by the defuzzified problem.

5 Conclusions

We have considered a job shop problem with uncertain durations modelled as TFNs and have proposed to analyse the robustness of the solution based on its sensitivity to posterior perturbations in task durations. We have proposed a methodology based on a numerical analysis of the performance of the solution subject to different types of changes in task durations. This methodology has been applied to obtain experimental results on twelve problem instances. The results show that using fuzzy numbers to represent task durations and taking into account this information about uncertainty in the solving process produces proactive solutions, robust to possible changes in task processing times.

Acknowledgements All authors are supported by MEC-FEDER Grant TIN2007-67466-C02-01.

References

1. Brucker, P., Knust, S.: Complex Scheduling. Springer (2006)
2. Herroelen, W., Leus, R.: Project scheduling under uncertainty: Survey and research potentials. European Journal of Operational Research **165** (2005) 289–306
3. Dubois, D., Fargier, H., Fortemps, P.: Fuzzy scheduling: Modelling flexible constraints vs. coping with incomplete knowledge. European Journal of Operational Research **147** (2003) 231–252
4. Słowiński, R., Hapke, M., eds.: Scheduling Under Fuzziness. Volume 37 of Studies in Fuzziness and Soft Computing. Physica-Verlag (2000)
5. Sakawa, M., Kubota, R.: Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy duedate through genetic algorithms. European Journal of Operational Research **120** (2000) 393–407
6. Petrovic, S., Fayad, S., Petrovic, D.: Sensitivity analysis of a fuzzy multiobjective scheduling problem. Int. Journal of Production Research **46**(12) (2007)3327–3344
7. González Rodríguez, I., Puente, J., Vela, C.R., Varela, R.: Semantics of schedules for the fuzzy job shop problem. IEEE Transactions on Systems, Man and Cybernetics, Part A **38(3)** (2008) 655–666
8. Fortemps, P.: Jobshop scheduling with imprecise durations: a fuzzy approach. IEEE Transactions of Fuzzy Systems **7** (1997) 557–569
9. González Rodríguez, I., Vela, C.R., Puente, J.: A memetic approach to fuzzy job shop based on expectation model. In: Proceedings of IEEE International Conference on Fuzzy Systems, FUZZ-IEEE2007, London (2007) 692–697
10. Liu, B., Liu, Y.K.: Expected value of fuzzy variable and fuzzy expected value models. IEEE Transactions on Fuzzy Systems **10** (2002) 445–450
11. Jin, Y., Branke, J.: Evolutionay optimization in uncertain environments–a survey. IEEE Transactions on Evolutionary Computation **9** (2005) 303–317
12. Ishibuchi, H., Murata, T.: A multi-objective genetic local search algorithm and its application to flowshop scheduling. IEEE Transactions on Systems, Man, and Cybernetics–Part C **67**(3) (1998) 392–403
13. Van Laarhoven, P., Aarts, E., Lenstra, K.: Job shop scheduling by simulated annealing. Operations Research **40** (1992) 113–125
14. Branke, J., Mattfeld, D.: Anticipation and flexibility in dynamic scheduling. International Journal of Production Research **43** (2005) 3103–3129