Multi-objective genetic algorithm for single machine scheduling problem under fuzziness

Duenas, A. and Petrović, D

A New Approach to a Multi-objective Single Machine Scheduling Problem under Fuzziness

Alejandra Duenas, Dobrila Petrovic[*]

Control Theory and Applications Centre (CTAC), School of Mathematical and Information Sciences, Coventry University, Coventry, United Kingdom

Abstract

This paper presents a new approach to a single machine scheduling problem in the presence of uncertainty and multiple scheduling objectives. The uncertain parameters under consideration are duedates of jobs. They are modelled by fuzzy sets where membership degrees represent decision maker's satisfaction grades with respect to the jobs' completion times. The two objectives defined are to minimise the maximum and the average tardiness of the jobs. Due to fuzziness in the duedates, the two objectives become fuzzy too. In order to find a job schedule that maximises the aggregated satisfaction grades of the objectives, a hybrid algorithm that combines a multi-objective genetic algorithm with local search is developed. The algorithm is applied to solve a real-life problem of a manufacturing pottery company.

Keywords

Single machine scheduling, fuzzy sets, multi-objective optimisation, genetic algorithms, local search.

1. Introduction

Scheduling of jobs and control of their flow through a production process is of great importance to modern manufacturing organisations. The diversity of scheduling problems, large-scale dimensions and their dynamic nature make scheduling problems computationally very complex and difficult to solve. They require multi-disciplinary approaches, drawing on various concepts and techniques of operations managements, artificial intelligence, optimisation theory, etc. In real world production scheduling problems, quite often, parameters are characterised vaguely due to imprecise data or incomplete information available. Conventional concepts of

[*] Corresponding author. Tel.: +44-2476888766
E-mail address: d.petrovic@coventry.ac.uk

randomness and probability distributions have been traditionally applied to model uncertainty inherent in production scheduling, mainly uncertainty in processing times. However, in cases where there is no evidence recorded in the past, or where there is lack of evidence available, or simply the evidence do not exist, probability distributions cannot be derived with complete confidence.

The potential of using fuzzy sets theory in treating different sources of uncertainty, particularly when intuition and judgement play an important role, has been acknowledged in the literature [15, 16]. It has been successfully applied to model various manufacturing parameters, including for example, modelling of fuzzy customer demand [13], fuzzy processing times [10], fuzzy production due dates [5] or fuzzy job precedence relations [9].

Real-life production environments, in essence, involve multiple objectives to be considered simultaneously. The most frequently considered objective in production scheduling has been to minimise completion time of all jobs. In addition, other objectives have been of interest such as objectives related to due dates (e.g., to minimise the maximum tardiness or the maximum earliness), resource utilisation, cost incurred, throughput, etc [18].

Different approaches to multi-objective single machine problems with fuzzy parameters have been presented in the literature in the last decade. Ishii and Tada [9] considered a single machine scheduling problem with the objective to minimize the maximum lateness of jobs with fuzzy precedence relations. A fuzzy precedence relation relaxes the crisp precedence relation and represents the satisfaction grade with respect to precedence between two jobs. Therefore, an additional objective to maximise the minimum satisfaction level achieved regarding the fuzzy precedence relations was considered. An algorithm for determining nondominated solutions based on a graph representation of the precedence relations was proposed. Adamopoulos and Pappis [1] presented a fuzzy-linguistic approach to a multi-criteria sequencing problem. They considered a single machine where each job was characterised by fuzzy processing times. The objective was to determine the length of processing times, the common duedate and a sequence of the jobs for the machine through the search for a near-optimal value of a related cost function. Another approach to solving a single machine multi-criteria scheduling problem was presented by Lee et al. [11]. The proposed approach used linguistic values for the evaluation of each criterion (e.g. *very poor*, *poor*, *fair*, *good* and *very good*) and for the representation of its relative weight (e.g. *very unimportant*, *unimportant*, *medium important*, *important* and *very important*). A tabu-search was used as the optimisation technique to find the near-optimal solution with an aggregated fuzzy objective function. Ishibuchi and Murata [7] presented a flow shop scheduling problem with fuzzy parameters such as fuzzy duedates and fuzzy processing times, where the objectives were to minimise the maximum earliness and tardiness of all jobs, to minimise the makespan and to minimise the total flowtime. A multi-objective genetic algorithm that can handle these fuzzy scheduling objectives was developed.

In this paper, a new approach to solving a fuzzy multi-objective single machine scheduling problem is proposed. It brings together three areas, namely fuzzy reasoning, multi-criteria decision making and production scheduling in uncertain environments. Fuzzy parameters are jobs' duedates, modelled by fuzzy sets, where the corresponding membership functions represent decision maker satisfaction degrees with respect to jobs' completion times. Therefore, the objective functions under consideration become fuzzy. They are defined using membership functions and aggregated using a standard averaging operator. The aggregation operator allows the consideration of fuzzy multi-objectives simultaneously, and it is used to define a decision membership function. Finally, a hybrid model that combines a genetic algorithm (GA) with local search is applied to find a decision (schedule) that maximises the decision function, i.e., the satisfaction degree achieved for all the objectives. The new approach is applied to a real life-scheduling problem identified through the collaboration with a manufacturing pottery company.

The paper is organised as follows. The description of the fuzzy multi-objective scheduling problem and a new approach to solving it are presented in Section 2. In Section 3, the GA in conjunction with the tabu-search used as a local search optimisation technique is presented. Section 4 contains a brief description of a real-life scheduling problem. Additionally in Section 5, the results obtained by applying the new multi-objective hybrid algorithm to the real-life problem are described and analysed using different GA and tabu-search parameter specifications. Finally, a paper summary is presented in Section 6 with some issues for future research.

2. Multi-objective single machine scheduling with fuzzy duedates

Consider a problem of K jobs, $J_{1,} J_{2,\ldots,} J_K$, to be scheduled on a single machine with the following assumptions:

- No job pre-emption is allowed.
- The machine can only process one job at a time.
- All jobs are available in time zero.
- The machine set-up times are negligible.
- All jobs have identical and deterministic processing times.

- Duedates are fuzzy.

The schedule to be generated is represented as a permutation of K jobs denoted as $\Pi(\pi_1, \pi_2,\ldots, \pi_K)$. A new approach to defining and solving the single machine scheduling problem as a fuzzy multi-objective problem is proposed as depicted in Figure 1.

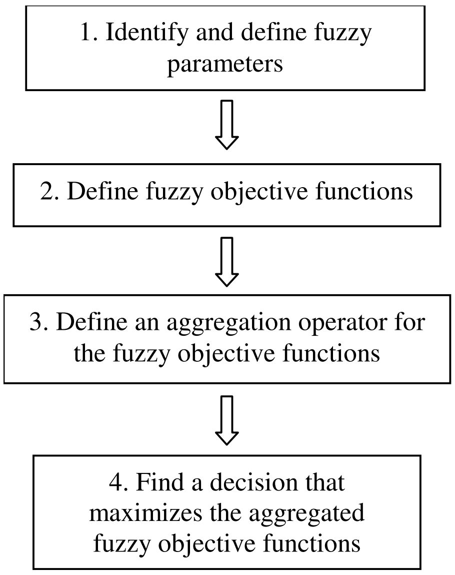
1. Identify and define fuzzy parameters
2. Define fuzzy objective functions
3. Define an aggregation operator for the fuzzy objective functions
4. Find a decision that maximizes the aggregated fuzzy objective functions

Figure 1: An approach to fuzzy multi-objective optimisation

The steps included in the fuzzy multi-objective approach to scheduling are the following:

<u>Step 1.</u> First, uncertain scheduling parameters are identified. They are modelled using fuzzy sets where membership degrees represent a decision maker (DM) (in this case a production manager) preference profile, i.e., satisfaction grades with respect to the different parameter values. The membership functions can have different shapes such as semi-linear (e.g. triangular, trapezoidal) and nonlinear (e.g. Gaussian function).

In this paper, the fuzzy parameters considered are jobs' duedates. A fuzzy duedate is defined as follows [7]. Let C_k be the completion time of job k. The membership function of fuzzy duedate d_k, $k = 1,\ldots,K$ is defined in (1) and represented in Figure 2:

$$\mu_{d_k}(C_k) = \begin{cases} 1 & \text{if } C_k \le d_k^L \\ 1 - \dfrac{C_k - d_k^L}{d_k^U - d_k^L} & \text{if } d_k^L < C_k < d_k^U, \quad k = 1,\ldots,K \\ 0 & \text{if } d_k^U \le C_k \end{cases} \tag{1}$$

where the two boundaries: a lower bound d_k^L and an upper bound d_k^U are defined. If job k is completed before or at time d_k^L the DM is completely satisfied i.e., the satisfaction degree is 1, if the job is completed between d_k^L and d_k^U a linear shape represents the DM's satisfaction degrees, and finally if the job is completed at or after d_k^U the DM is completely unsatisfied and the satisfaction degree is 0.

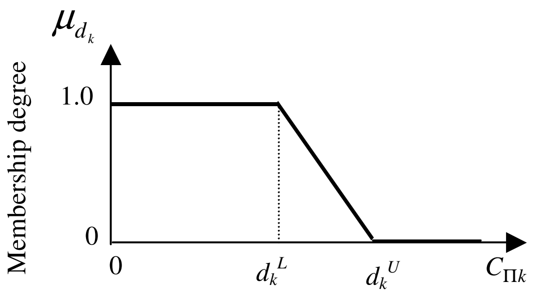

Figure 2: Fuzzy duedate

Step 2. This step involves identification and definition of the objectives of interest. In this paper, the scheduling problem under consideration is restricted to two objectives, however, the fuzzy multi-objective approach developed can handle more than two objectives. The two identified objectives are to find a permutation of the jobs in such a way as (a) to minimise the maximum tardiness of all the jobs and (b) to minimise the average tardiness.

Having the permutation $\Pi(\pi_1, \pi_2, \ldots, \pi_K)$, the job k completion time $C_{\Pi k}$ is defined as:

$$\begin{aligned} C_{\Pi 1} &= p_1 \\ C_{\Pi k} &= C_{\Pi\, k-1} + p_k; \quad k = 2, \ldots, K \end{aligned} \tag{2}$$

where p_k is the processing time of job k in permutation Π.

Due to fuzziness of duedates, the tardiness of each job becomes fuzzy too. The corresponding membership functions associated with the objectives represent the DMs satisfaction grades with respect to achieving the objectives. As previously defined, the first objective is to minimise the maximum tardiness of the jobs. According to Ishibuchi and Murata [7], the problem of minimising the maximum tardiness can be transformed into the problem of maximising the minimum of all the satisfaction grades achieved with respect to the fuzzy duedates. Therefore, the first objective is to find a permutation of jobs $\Pi(\pi_1, \pi_2, \ldots, \pi_K)$ that maximises membership function μ_{f_1} associated with the first objective:

$$\begin{aligned}&\text{maximise } \mu_{f_1},\\ &\text{where } \mu_{f_1} = \min_{k=1,\ldots,K}(\mu_{d_k}(C_{\Pi k}))\end{aligned} \tag{3}$$

The second objective is the minimisation of the average tardiness [2]. The average tardiness in the crisp case is defined as $\overline{\mathrm{T}} = \frac{\sum_{k=1}^{K} T_k}{K}$, where T_k is the tardiness of job J_k. In the presence of uncertainty in dudates, the new fuzzy objective is defined as to maximise the average satisfaction grade with respect to the fuzzy duedates; i.e., find a permutation of jobs $\Pi(\pi_1, \pi_2, \ldots, \pi_K)$ that maximises membership function μ_{f_2} :

$$\begin{aligned}&\text{maximise } \mu_{f_2},\\ &\text{where } \mu_{f_2} = \frac{\sum_{k=1}^{K}\mu_{d_k}(C_{\Pi k})}{K}\end{aligned} \tag{4}$$

<u>*Step* 3.</u> The aggregation operation is an essential fuzzy set operation that combines elements from a number of fuzzy sets into a single fuzzy set [12]. Various operators such as triangular norms (*t-norms*), triangular conorms (*t-conorms*), averaging, and ordered weighted averaging (OWA) have been defined in fuzzy sets theory to model fuzzy sets aggregation. The first two operators, t-norms and t-conorms, are used to model the intersection and union of fuzzy sets [20]. The averaging operation is defined as a generalised mean [3] and it has been most commonly used in decision making, specifically utility theory and multi-criteria decision theory. The OWA operators are special cases of the mean operators, defined as a weighted sum with ordered arguments [19]. The OWA operators have been developed to aggregate criteria in multi-criteria decision making.

In this paper, a standard averaging operator, *arithmetic mean*, is used to aggregate the objective membership functions defined in *Step* 2 as follows:

$$A(\mu_{f_1}, \mu_{f_2}) = \frac{1}{2}(\mu_{f_1} + \mu_{f_2}) \tag{5}$$

Step 4. Once the aggregation operator has been selected, a decision membership function μ_D is defined as follows:

$$\mu_D = A(\mu_{f_1}, \mu_{f_2}) \tag{6}$$

where μ_{f_1} is the membership function of Objective 1, μ_{f_2} is the membership function of Objective 2 and D is a fuzzy set of all job permutations where the membership degrees represent aggregated satisfaction grades achieved with respect to the given objectives.

Once the decision membership function has been defined, a decision that maximises the grade of satisfaction with respect to all the objectives has to be found. In this way, an initial fuzzy multi-objective problem is mapped into the following problem:

$$\text{find a permutation of jobs that maximises } \mu_D \tag{7}$$

In order to find this job permutation, a search algorithm must be applied. The algorithm used is a multi-objective genetic algorithm with local search defined in the following section.

3. Multi-objective genetic algorithm with local search

A general multi-objective optimisation problem (MOP) is defined as follows [6]:

$$\begin{gathered}\text{Minimise } \mathbf{z} = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_N(\mathbf{x})) \\ \text{subject to } \mathbf{x} \in \mathbf{X}\end{gathered} \tag{8}$$

where $\mathbf{z}$ is the objective vector, $\mathbf{x}$ is the decision variable vector, N is the number of objectives and $\mathbf{X}$ is the feasible region in the decision space.

A nondominated solution (Pareto-optimal solution) of the MOP is the solution in which it is not possible to improve one objective without increasing the other objectives. In a formal way, $\mathbf{x}^*$ is a nondominated solution if and only if there is not any $\mathbf{x} \in \mathbf{X}$ such that $f_i(\mathbf{x}) \leq f_i(\mathbf{x}^*)$ for all $i = 1,\ldots,$ N, and $f_j(\mathbf{x}) < f_j(\mathbf{x}^*)$ for at least one j.

Genetic algorithms (GA) have been used to solve single objective and MOPs [4]. Generally, GAs are stochastic algorithms based on natural evolution principles, that perform a search iteratively starting from an initial population of candidate solutions (strings) and apply certain genetic operators to find a near optimal solution. GAs used for MOPs are called multi-objective GAs and their main purpose is to find as many nondominated solutions as possible [4].

Ishibuchi et al. [8] proposed a multi-objective genetic algorithm with local search (MOGLS), for determining a solution of a flowshop scheduling problem. The local search technique is introduced in order to improve convergence time of the GA. The MOP is solved using a scalar fitness function *f*(x), to be minimised, where all the objectives were combined as follows:

$$f(\mathrm{x}) = w_1 f_1(\mathrm{x}) + w_2 f_2(\mathrm{x}) + \ldots + w_n f_N(\mathrm{x}) \tag{9}$$

where $w_i \geq 0$, $i = 1, 2, \ldots, N$, $\sum_{i=1}^{N} w_i = 1$ and w_i is specified randomly. The MOGLS algorithm is the hybridisation of a GA and a local search procedure applied in order to improve the convergence time to the Pareto-front, where the Pareto-front is a curve or surface in an n-dimensional space composed by all nondominated or Pareto-optimal solutions, and, given the constraints of the model, no solutions exist beyond the Pareto-front. One of the main characteristics of this algorithm is that it uses two populations, an initial population defined as in any typical GA and a secondary population where all the nondominated solutions obtained during the GA's performance are stored. This secondary population has two main functions: (a) to store the nondominated solutions generated during the GA's performance and (b) to input nondominated solutions into the elitist strategy. The elitist strategy is incorporated in order to ensure that good solutions are considered in each of the algorithm iterations.

In the new approach proposed in this paper, the minimisation of the scalar fitness function is replaced by the maximisation of the grade of satisfaction of all the objectives given by (6) and (7). The steps included are described below:

1. Initialisation. Generate randomly an initial solution population of size N_p.
2. Evaluation. Calculate the N objectives for each string in the solution population and store the nondominated solutions in the secondary population.

3. Selection. Use tournament selection in order to select (N_p - N_elite) pairs of parents for crossover and mutation operators, where N_elite is the number of solutions selected for the elitist strategy.
4. Crossover and mutation application. Define probability of crossover p_c and probability of mutation p_m. Apply crossover operator to each of the selected pairs of parents and mutation operator to the offspring generated after applying the crossover operator.
5. Elitist strategy utilisation. Randomly select N_elite solutions from the secondary population and add the selected solutions to the population.
6. Local search application. Select an element from N_p using tournament selection with a tournament size bigger than 3 and local search probability p_S. If local search is applied to the selected element, the new solution is included in the next population, otherwise the selected element is copied directly to the next population. The local search procedure proposed for this step is tabu-search and it is defined in Subsection 3.1.
7. Termination. If the number of solutions evaluated is met, stop the algorithm, else return to *Step* 2.

3.1 Local search procedure

Tabu-search is an "improvement type" local search procedure, that can be applied in order to find a better solution through the manipulation of a solution arbitrarily selected [14]. For instance, in case of a scheduling problem, the tabu-search procedure can be used to obtain a better schedule through the modification of the current schedule. The main characteristic of a local search procedure is that it cannot be guaranteed that an optimal solution will be found, because the best solution within the neighbourhood of the current solution only is searched. The solutions found within a certain neighbourhood, called neighbours, are candidates to become the next solution to move to. In order to determine whether a neighbour solution should be accepted or rejected, an acceptance-rejection criterion has to be defined. A tabu-list with a fixed number of entries is kept, where the modifications that the procedure is not allowed to do, are stored. If the tabu-list is full and a new modification (mutation) has to be entered, the modification that was first entered into the list is deleted and the list is pushed down for one position.

Tabu-search can be applied either to single objective or multi-objective problems [14]. In the case of single objective problems, the search stores in the memory the best solution found so far, whilst in the multi-objective case all schedules that are nondominated are stored in the memory.

In this paper, the tabu-search procedure is used as part of the MOGLS algorithm to perform the local search. The acceptance-rejection criterion is the same as the one defined for the corresponding GA, i.e., maximisation of the grade of satisfaction of all the objectives. Therefore, if a neighbour has a smaller grade of satisfaction it is rejected, otherwise is accepted. Having selected a specific schedule $\prod_s$, a new schedule is generated through an interchange of a pair of jobs in the schedule i.e., by applying the mutation operator and the new schedule is considered as a part of the neighbourhood of $\prod_s$. Every time a mutation is performed in $\prod_s$, the reverse pair is entered in the tabu-list.

4. Real-world scheduling problem

In this paper, a real-world problem has been defined through collaboration with Denby Pottery Company Ltd. This company has been involved in the pottery industry for almost 200 years and manufactures a wide range of ceramic tableware products. Hence, scheduling is of great interest in many areas of their manufacturing processes such as biscuit making and glazing where kilns play one of the most important roles in the production. Since the data obtained from the company is confidential some of the data used in this paper are hypothetical.

Pottery industry usually uses two types of kilns: tunnel and intermittent kilns. Tunnel kilns are continuously fired and work 24 hours a day, 7 days a week, while intermittent kilns are fired once or twice a day. The kiln considered in this paper is an intermittent kiln that has a 12 hours cycle and processes 6 cars every time it is fired. The kiln is fired once a day, five days a week, and therefore the number of jobs processed in one week is equal to 30. Moreover, the schedule time horizon under consideration is one week. The problem is defined as a single machine scheduling problem, with the assumption that each car represents a job to be scheduled. This assumption can be made as each car is loaded with different products. Therefore, the problem under consideration is to sequence 30 jobs with respect to the objectives defined below.

As the kiln cycle time is fixed, it is assumed that all the jobs have an identical processing time of 12 hours. The machine set-up times are negligible. Figure 3 shows how the problem is visualised as a single machine scheduling problem.

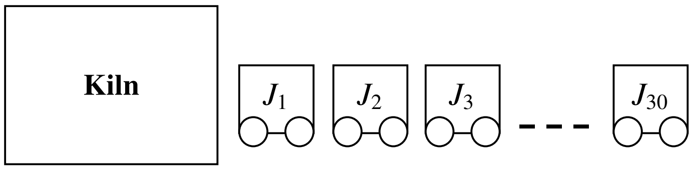

Figure 3: Kiln scheduling

Following the steps defined in Section 2 (Figure 1), the scheduling problem is defined and solved as follows.

Step 1. At present, it is considered that a DM is completely satisfied if at the end of the week (time horizon) all products specified in the production plan have been produced. The question that arises is what difference it will make in the jobs' sequence if the DM starts considering the completion time of each job in the scheduling process. Since some products have to go to different processes after being in the kiln, having a higher priority than other products, defining and meeting adequate duedates become an important element of the kiln scheduling. Moreover, the products inventory costs can be another important aspect in the consideration of job's duedates. Hence, it is desirable to represent different grades of DM's satisfaction with respect to jobs completion times and, therefore, a more detailed schedule (daily instead of weekly) has to be generated. For this reason, it has been decided to specify a fuzzy duedate for each job with a semi-linear shape membership function as shown in Figure 2.

The boundaries (d_k^L and d_k^U) of each fuzzy duedate expressed in days are specified as follows:

d_k^L is a random integer from the set $\{1,2,3,4,5\}$

d_k^U is a random integer from the set $\{1,2,3,4,5\}$

where the set $\{1,2,3,4,5\}$ represents the possible days in which the job can be completed and $d_k^U \geq d_k^L$.

Step 2. The objectives to be optimised are those defined in Section 2; namely

1. to minimise the maximum tardiness of all the jobs, i.e., to maximise the minimum degree of satisfaction with respect to the fuzzy duedates:

$$\text{maximise } \mu_{f_1}, \quad \text{where } \mu_{f_1} = \min_{k=1,\ldots,30} (\mu_{d_k}(C_{\Pi k})) \tag{10}$$

2. to minimise the average tardiness, i.e., to maximise the average degree of satisfaction with respect to the fuzzy duedates:

$$\text{maximise } \mu_{f_2}, \quad \text{where } \mu_{f_2} = \frac{\sum_{k=1}^{30} \mu_{d_k}(C_{\Pi k})}{30} \tag{11}$$

<u>*Step* 3.</u> In this problem, a standard averaging operator given in (5) is used to aggregate the two objective membership functions and to define a decision membership function μ_D.

<u>*Step* 4.</u> Finally, the decision that maximises the grade of satisfaction of both objectives, i.e., maximises μ_D is found using the multi-objective genetic algorithm with local search.

In order to apply the multi-objective genetic algorithm with local search the selection, crossover and mutation operators are defined as follows. The tournament selection is defined with a tournament size of 2; the crossover operator used is the position-based crossover proposed by Syswerda [17] and finally, the mutation operator selected is the insertion (shift) mutation. It is important to note that the insertion mutation operator is applied to the whole selected parent (string).

The initial parameter specifications of the multi-objective genetic algorithm with local search are the following:

Population size (N_p): 60,

Crossover probability (p_c): 0.8,

Mutation probability per string (p_m): 0.6,

Tournament size: 2,

Number of elite solutions (N_{elite}): 10,

Local search probability (p_s): 0.8,

Tournament size in the selection of solutions for the local search: 4, Tabu-list size: 5.

These parameters were defined as a starting point but varied in different analyses performed. The algorithm was run 1000 iterations for each analysis. Some of the results obtained are discussed in the following section.

5. Results analyses

Three different result analyses were performed in order to evaluate the performance of the algorithm proposed, including (a) analysis of nondominated solutions, (b) analysis of multi-objective genetic algorithm performance, and (c) analysis of local search performance.

5.1 Analysis of nondominated solutions

The first analysis was divided into two different stages, where the first stage consisted of running the GA with no local search and considering the candidate solutions stored in the secondary population (N_{elite}) after 1000 iterations. These candidate solutions represent the nondominated solutions found in the last iteration. In the second stage the candidate solutions from the primary population obtained every 100 iterations were considered while the algorithm was run 1000 iterations. The objective of this stage of the analysis was to observe the number of nondominated solutions obtained from the total of candidate solutions.

Figure 4 shows the results obtained when the problem was solved using the multi-objective GA with the elitist procedure but without performing the tabu-search. The number of iterations that the algorithm was run was 1000 and only 7 nondominated solutions were found.

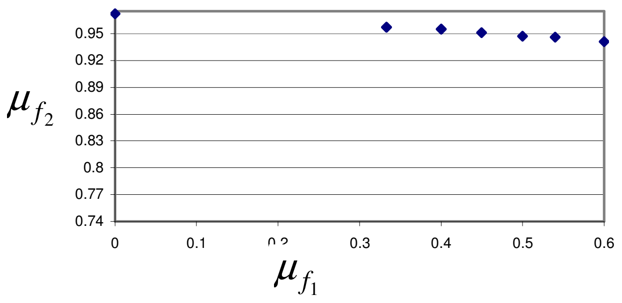

Figure 4: Nondominated solutions where N_{elite} = 10 obtained by running the GA without tabu-search

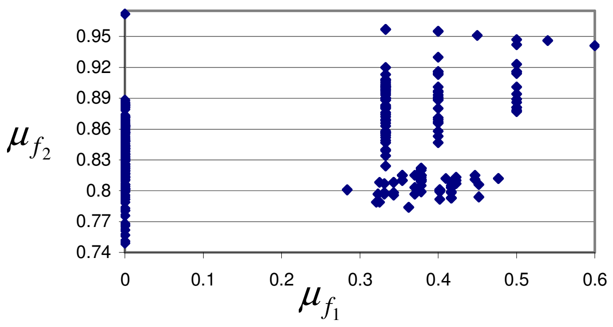

Figure 5: Results obtained every 100 iterations when the algorithm was run 1000 iterations

Figure 5 shows the best results obtained every 100 iterations when the algorithm was run 1000 iterations. The results in Figure 5 reveal that not all the solutions in the new population after 1000 iterations were nondominated (only 10%). Hence, in the best solution found the satisfaction grades were μ_{f_1} = 0.400 and $\mu_{f_2} = 0.919$. Moreover, 83% of the jobs scheduled (i.e., 25 jobs) were finished on time; in other words, the corresponding satisfaction grade with respect to the fuzzy duedate was $\mu_{d_k} = 1.0$, while the other 17% of the jobs had a satisfaction grade μ_{d_k} in the range from 0.400 to 0.667.

5.2 Analysis of multi-objective genetic algorithm performance

The second analysis concerned the performance of the multi-objective genetic algorithm with local search using different combinations of crossover and mutation probabilities, in order to find the values of p_c and p_m that yielded the best results. The algorithm was run 1000 and 5000 iterations, and the results are shown in Table 1. All other parameters were as defined in Section 4 (N = 60, p_s = 0.8, N_{elite} = 10). Some interesting observations are outlined here. When the probability of crossover p_c and the probability of mutation p_m were 0, the multi-objective genetic algorithm with local search acted as a tabu-search with a selection mechanism. From Table 1, it can be seen that the maximum values of μ_D obtained when $p_c = 0$ and $p_m = 0$ are higher than the values of μ_D when $p_c = 0.5$ and $p_m = 0$. In other words, the algorithm performs better when it is acting as a standard tabu-search than when it is incorporated with the GA because the mutation operator is not applied ($p_m = 0$) and the probability of crossover is not high enough.

p_c	p_m	**Max μ_D 1000 iterations**	**Max μ_D 5000 iterations**
0	0	0.451	0.451
0.5	0	0.448	0.448
0.5	0.2	0.723	0.803
0.5	0.4	0.718	0.724
0.5	0.6	0.719	0.721
0.5	1	0.676	0.711
0.8	0	0.724	0.724
0.8	0.2	0.779	0.801
0.8	0.4	0.712	0.725
0.8	0.6	0.631	0.697
0.8	1	0.612	0.659
1	0	0.716	0.716
1	0.2	0.812	0.819
1	0.4	0.721	0.766
1	0.6	0.677	0.713
1	1	0.606	0.643

Table 1: Results achieved with different crossover and mutation probabilities

Furthermore, for the cases when $p_c = 0.8$ and $p_m = 0$, and $p_c = 1$ and $p_m = 0$, the same maximum μ_D were achieved when the algorithm was run 1000 iterations and 5000 iterations. Consequently, it can be concluded that the mutation operator is necessary in order to maintain diversity in the population before the local search procedure is applied.

The results obtained, when $p_m = 1.0$ i.e., the mutation operator was applied to the whole population, were very poor. Moreover, the worst result was found when $p_c = 1.0$ and $p_m = 1.0$ (the crossover and mutation operators were applied to the whole population).

On the other hand, the best results were achieved in the cases when $p_c = 0.5$ and $p_m = 0.2$, $p_c = 0.8$ and $p_m = 0.2$, and $p_c = 1.0$ and $p_m = 0.2$. It can be concluded that in the cases when $p_c = 0.5$, $p_c = 0.8$ and $p_c = 1.0$ the best results were obtained when $p_m = 0.2$. Finally, as expected, the results were better when the algorithm was run 5000 iterations, except in the cases mentioned above, when $p_m = 0$.

5.3 Analysis of local search performance

Having found the GA parameter values, p_c and p_m, that yielded the best results, the local search effects on the algorithm's search ability was analysed by considering different local search probabilities p_s. Although, the best result for the GA was found when $p_c = 1.0$ and $p_m = 0.2$, it was decided to use also $p_c = 0.5$ and $p_m = 0.2$, and $p_c = 0.8$ and $p_m = 0.2$ in order to analyse the effects of different probabilities of crossover and a variable local search probability.

Table 2 shows the results obtained with different local search probabilities and the parameters specified as: $N = 60$, $p_c = 0.5$, $p_m = 0.2$, $N_{elite} = 10$. When $p_s = 1$ the tabu-search was applied to every candidate solution in the algorithm.

p_s	Max μ_D 5000 iterations
0	0.644
0.1	0.711
0.2	0.721
0.3	0.718
0.4	0.775
0.5	0.714
0.6	0.716
0.7	0.722
0.8	0.803
0.9	0.725
1	0.721

Table 2: Results achieved with different p_s and $p_c = 0.5$ and $p_m = 0.2$

The results obtained when $p_s = 0$, i.e., when the local search was not performed in conjunction with the GA, were not satisfactory. When the p_s value was increased, the algorithm yielded better results and the best result was achieved when $p_s = 0.8$. It is important to note that the results obtained when $p_s = 1$, i.e., when the local search was applied to all the strings in the primary population, were not as good as when $p_s = 0.8$. Consequently, it is possible to conclude that although the addition of the local search to the genetic search enhanced the algorithm's performance, the selection of the right local search probability value is important for the good operation of the multi-objective genetic algorithm with local search.

Table 3 shows the results obtained with different local search probabilities considered and the GA parameters specified as previously, but with $p_c = 0.8$, $p_m = 0.2$. The best result was achieved when $p_s = 0.6$ and the worst result was obtained when $p_s = 1$, showing the importance of using the right local search probability.

p_s	**Max μ_D** **5000 iterations**
0	0.610
0.1	0.672
0.2	0.730
0.3	0.729
0.4	0.772
0.5	0.730
0.6	0.809
0.7	0.804
0.8	0.801
0.9	0.728
1	0.645

Table 3: Results achieved with different p_s and $p_c = 0.8$ and $p_m = 0.2$

Additionally, the local search effect on the algorithm's search ability was analysed by considering the GA parameters that yielded the best solution, specified as: $N = 60$, $p_c = 1.0$, $p_m = 0.2$, $N_{elite} = 10$. Table 4 shows the results obtained considering different search probabilities. The best result was achieved when $p_s = 0.8$, showing the importance of selecting the right probabilities not only for the crossover and mutation operators but also for the local search procedure.

p_s	**Max μ_D** **5000 iterations**
0	0.657
0.1	0.704
0.2	0.703
0.3	0.711
0.4	0.717
0.5	0.721
0.6	0.807
0.7	0.814
0.8	0.819
0.9	0.734
1	0.710

Table 4: Results achieved with different p_s and $p_c = 1.0$ and $p_m = 0.2$

6. Summary

A new approach to single machine scheduling is developed. A single machine scheduling is considered as a multi-objective problem with fuzzy due dates. In the presence of the fuzzy dates, the objectives become fuzzy too. They are considered simultaneously using the arithmetic mean as an aggregation operator of all the satisfaction grades achieved with respect to the given objectives. A GA combined with tabu-search is applied to find a scheduling solution that maximises the overall satisfaction grade.

The algorithm is successfully applied to a real-world problem of kiln scheduling, that involved two objectives. The use of fuzzy duedates proved to be beneficial when subjective judgement is appropriate to use. The results obtained showed that the algorithm performs better when the genetic search and local search are combined. The algorithm developed is flexible in the sense that it can be applied to a problem with any number of fuzzy objectives.

After discussing the preliminary results with the collaborating company, some directions for further research are outlined.

Other parameters of importance for kiln scheduling will be included, such as earliness and tardiness penalties. Different membership functions for the jobs' fuzzy duedates such as triangular or trapezoidal shapes will be used.

Other kind of t-norms (such as product), as well as the use of other aggregation operators such as t-conorm and ordered weighted aggregating (OWA), will be analysed.

Additional constraints related to car loading will be included into the model. The car loading is an indispensable element in the kiln scheduling since the total size and weight of products to be loaded are restricted. The effect that an appropriate car loading can have on energy consumption will be investigated.

Acknowledgements

This research is supported by Engineering and Physical Sciences Research Council (EPSRC), grant no. GR/R95326/01. This support is gratefully acknowledged. We also acknowledge the support of the industrial collaborator the Denby Pottery Company Ltd.

References

[1] G. I. Adamopoulos and C.P Pappis, A fuzzy-linguistic approach to a multi-criteria sequencing problem, European Journal of Operational Research 92, No. 3 (Aug. 1996) 628-636.

[2] C-L. Chen and R.L. Bulfin, Scheduling Unit Processing Time Jobs on a Single Machine with Multiple Criteria, Computers and Operations Research 17, No. 1 (1990) 1-7.

[3] H. Dyckhoff and W. Pedrycz, Generalized Means as a Model of Compensative Connectives, Fuzzy Sets and Systems 14, No. 2 (Nov. 1984) 143-154.

[4] D.E. Goldberg, Genetic Algorithms in Search, Optimisation and Machine Learning (Addison-Wesley, 1989).

[5] T. Hong and T. Chuang, New triangular fuzzy Johnson algorithm, Computers and Industrial Engineering 36, No. 1, (Jan. 1999) 179-200.

[6] 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 28, No. 3, (Aug. 1998) 392-403.

[7] H. Ishibuchi and T. Murata, Flowshop scheduling with fuzzy duedate and fuzzy processing time, in: R. Slowiński and M. Hapke, Eds., Scheduling Under Fuzziness (Physica-Verlag, Heidelberg, 2000).

[8] H. Ishibuchi, T. Yoshida and T. Murata, Balance between Genetic Search and Local Search in Memetic algorithms for Multiobjective Permutation Flowshop Scheduling, IEEE Transaction on Evolutionary Computation 7, No. 2 (Apr. 2003) 204-223.

[9] H. Ishii and M. Tada, Single machine scheduling problem with fuzzy precedence relation, European Journal of Operational Research 87, No. 2 (Dec. 1995) 284-288.

[10] M. Kuroda and Z. Wang, Fuzzy job shop scheduling, International Journal of Production Economics 44, No. 1-2 (June 1996) 45-51.

[11] H.T. Lee, S.H. Chen and H.Y. Kang, Multicriteria scheduling using fuzzy theory and tabu search, International Journal of Production Research 40, No. 5. (Mar. 2002) 1221-1234.

[12] W. Pedrycz and F. Gomide, An Introduction to Fuzzy Sets: Analysis and Design, Bradford Book (MIT Press, 1998).

[13] R. Petrovic and D. Petrovic, Multicriteria Ranking of Inventory Replenishment Policies in the Presence of Uncertainty in Customer Demand, International Journal of Production Economics 71, No. 1-3 (May, 2001) 439-446.

[14] M. Pinedo and X. Chao, Operation Scheduling with applications in manufacturing and services (McGraw-Hill International Editions, Computer Science Series, 1999).

[15] E. Ruspini, P. Bonissone and W. Pedrycz, Eds., Handbook of Fuzzy Computation, (Institute of Physics Publishing, Bristol and Philadelphia, 1998).

[16] R. Slowiński and M. Hapke, Scheduling Under Fuzziness (Physica-Verlag, Heidelberg, 2000).

[17] G. Syswerda, Uniform Crossover in Genetic Algorithms, Proceedings of the 3rd International Conference on Genetic Algorithms and Their Applications, J.D. Schaffer, Ed., (Morgan Kaufmann Publishers, San Mateo, CA., 1989).

[18] V. T'kindt and J.-C. Billaut, Multicriteria Scheduling: Theory, Models and Algorithms (Springer-Verlag, Berlin Heidelberg, 2002)

[19] R. Yager, On Ordered Weighted Averaging Aggregation Operations in Multicriteria Decision Making, IEEE Transactions on Systems, Man and Cybernetics 18, No. 1, (Jan.-Feb. 1988) 183-190.

[20] H.-J. Zimmermann, Fuzzy Set Theory and Its Applications (Kluwer Academic Publishers, Massachusetts, 2001)