On optimizing a bi-objective flowshop scheduling problem in uncertain environment

Arnaud Liefooghe, Matthieu Basseur, Jérémie Humeau, Laetitia Jourdan, El-Ghazali Talbi

On Optimizing a Bi-objective Flowshop Scheduling Problem in Uncertain Environment

Arnaud Liefooghe[a,b,*], Matthieu Basseur[c], Jérémie Humeau[d], Laetitia Jourdan[a,b], El-Ghazali Talbi[a,b]

[a] *LIFL, Université Lille 1, UMR CNRS 8022, 59655 Villeneuve d'Ascq cedex, France*
[b] *INRIA Lille-Nord Europe, 40 av. Halley, 59650 Villeneuve d'Ascq, France*
[c] *LERIA, University of Angers, 2 bd. Lavoisier, 49045 Angers, France*
[d] *École des Mines de Douai, IA department, BP 10838, 59508 Douai, France*

Abstract

Existing models from scheduling often over-simplify the problems appearing in real-world industrial situations. The original application is often reduced to a single-objective one, where the presence of uncertainty is neglected. In this paper, we focus on multi-objective optimization in uncertain environments. A bi-objective flowshop scheduling problem with uncertain processing times is considered. An indicator-based evolutionary algorithm is proposed to handle these two difficulties (multiple objectives and uncertain environment) at the same time. Four different strategies, based on uncertainty-handling quality indicators, are proposed in the paper. Computational experiments are performed on a large set of instances by considering different scenarios with respect to uncertainty. We show that an uncertainty-handling strategy is a key issue to obtain good-quality solutions, and that the algorithm performance is strongly related to the level of uncertainty over the environmental parameters.

Keywords: Permutation flowshop scheduling, evolutionary algorithms, multi-objective combinatorial optimization, uncertain processing times.

*Corresponding author.

Email addresses: arnaud.liefooghe@univ-lille1.fr (Arnaud Liefooghe), basseur@info.univ-angers.fr (Matthieu Basseur), jeremie.humeau@mines-douai.fr (Jérémie Humeau), laetitia.jourdan@lifl.fr (Laetitia Jourdan), talbi@lifl.fr (El-Ghazali Talbi)

1. Introduction

Many real-world problems arising in combinatorial optimization have to face a lot of difficulties. They are often characterized by large and complex search spaces, multiple conflicting objective functions, and a host of uncertainties that have to be taken into account. This is the case, for instance, of most scheduling problems: They are clearly multi-objective [1] and they are subject to many uncertainties [2]. There exists a growing demand for solving such real-world applications. However, in practice, the original problem is usually modeled in a single-objective and deterministic way. In fact, a few adjustments in terms of resolution methods can be very useful to address such problems. This research area has received an increasing interest in recent years because of its difficulty. Evolutionary algorithms are natural candidates to tackle such problems and make them preferable to classical optimization approaches. Indeed, on the one hand, their aptitude has been recognized for one of the most challenging issue related to multi-objective optimization, that is to the identification of a Pareto set approximation [3, 4]. On the other hand, they are often investigated as proper candidates when solving optimization problems that are subject to uncertainties coming from many sources, whether on decision variables and environmental parameters, or directly on the objective function(s) [5]. Due to their inherent stochastic nature and to their ability to find multiple solutions in a single run, evolutionary algorithms with sufficient adaptations present interesting mechanisms for solving both multi-objective and uncertain problems. However, very few studies devoted to the consideration of multi-objective optimization under uncertainty as a whole exist to date, and they are generally confined to problems from continuous optimization.

In many cases, the basic concept of uncertainty-handling can be translated to the application of sets from the decision space to the objective space. Hence, taking uncertainty into account often results in the comparison of sets. From a multi-objective standpoint, a solution is projected into a sample set of objective vectors, whose shape is generally not known in advance. The main challenge raised by such uncertain multi-objective problems can be summarized by the pairwise comparison of objective vector sets, rather than a pairwise comparison of single objective vectors in the deterministic case. Resolution methods and performance assessment must then be adapted to deal with this specific issue, either through the selection of representative objective vectors or by adjusting their internal mechanisms. In this pa-

per, evolutionary approaches are proposed for multi-objective optimization in uncertain environments, together with their application to a scheduling problem. In particular, we investigate a bi-objective permutation flow-shop scheduling problem with uncertain processing times. To the best of our knowledge, this is the first time that such an uncertain scheduling problem is investigated in a multi-objective way. Then, a number of components to be used within an indicator-based evolutionary algorithm are presented, and fitted to the problem under consideration. Note that, in the deterministic case, indicator-based search appears to be very effective [6], in particular for the flowshop scheduling problem under study [7]. At last, experimental design is discussed in the context of uncertain multi-objective optimization, and a number of variants of the general algorithm are experimented on flowshop scheduling problem instances of different structure and size. The contributions of this work can be summarized as follows.

- We formulate a permutation flowshop scheduling problem which aims at concurrently minimizing the makespan and the total tardiness, and for which the processing times are subject to uncertainty.
- We propose a new evolutionary algorithm to deal with uncertain multi-objective optimization problems based on multiple scenarios, *i.e.* realizations of stochastic data. It extends indicator-based selection approaches proposed in previous work from Basseur and Zitzler [8]. Four variants are here investigated, based on (i) objective vector level or indicator level aggregations and (ii) average and worst-case scenarios.
- We lead an experimental analysis of the approaches introduced in the paper for the problem under investigation. The experiments are conducted on a large set of benchmark instances and reveal interesting results depending on the optimization scenario, and on the level of uncertainty arising on the problem instance.

Recently, we suggested an evolutionary algorithm based on an average-case objective vector for bi-objective flowshop scheduling under uncertainty [9]. In this paper, we investigate this approach in detail, we propose new indicator-based aggregations, and consider both average and worst-case scenarios. Moreover, we discuss the issue of performance assessment in multi-objective optimization under uncertainty, and we lead an in-depth experimental analysis, with a particular focus on the impact of the number of scenarios considered by the resolution approach with respect to the degree of uncertainty.

The paper is organized as follows. Section 2 introduces a bi-objective flowshop scheduling problem with uncertain processing times. Section 3 gives the necessary background on evolutionary algorithms for multi-objective and uncertain optimization. Section 4 proposes new indicator-based evolutionary algorithms for multi-objective optimization in uncertain environments. Section 5 presents an experimental analysis of the approaches proposed in the paper for the flowshop scheduling problem. The last section concludes the paper.

2. A Bi-objective Flowshop Scheduling Problem with Uncertain Processing Times

The Flowshop Scheduling Problem (FSP) is one of the most studied problem from scheduling [10]. The majority of works devoted to the FSP considers it on a deterministic single-objective form and mainly aims at minimizing the *makespan*, *i.e.* the total completion time. However, many objective functions, varying according to the particularities of the tackled problem, can be taken into account, and multi-objective approaches have also been proposed. The reader is referred to [1, 11, 12] for surveys about multi-objective scheduling. Following the formulation of a deterministic model for the bi-objective FSP, this section presents various sources of uncertainty that have to be taken into account and introduces different probability distributions to model the stochastic nature of processing times.

2.1. Bi-objective Flowshop Scheduling

The FSP consists in scheduling N jobs $\{J_1, J_2, \ldots, J_N\}$ on M machines $\{M_1, M_2, \ldots, M_M\}$. Machines are critical resources, *i.e.* two jobs cannot be assigned to a given machine at the same time. A job J_i is composed of M consecutive tasks $\{t_{i1}, t_{i2}, \ldots, t_{iM}\}$, where t_{ij} is the j^{th} task of the job J_i, requiring the machine M_j. A processing time p_{ij} is associated to each task t_{ij}; and a due date d_i is given to each job J_i (the deadline of the job). As illustrated in Figure 1, we here focus on the permutation FSP, where the operating sequences of the jobs are identical and unidirectional on every machine. Then, for a problem instance of N jobs, there exists $N!$ feasible solutions. In this study, we focus on minimizing both the makespan (C_{max}) and the total tardiness ($\overline{T}$), that are among the most widely investigated objective functions from the literature [12]. Let C_{ij} be the completion date

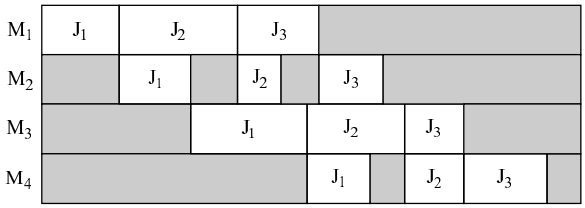

Figure 1: A feasible solution for the permutation flowshop scheduling problem where 3 jobs (J_1, J_2, J_3) are scheduled on 4 machines (M_1, M_2, M_3, M_4).

of task t_{ij}, the objective functions can be defined as follows.

$$C_{max} = \max_{i \in \{1,\ldots,N\}} \{C_{iM}\} \tag{1}$$

$$\overline{T} = \sum_{i=1}^{N} \Big\{ \max\{0, C_{iM} - d_i\} \Big\} \tag{2}$$

The single-objective FSP of minimizing the makespan is known to be NP-hard for three machines and more [13]. The objective of minimizing the total tardiness is already NP-hard for one machine [14]. Hence, medium and large-size bi-objective problem instances can generally not be solved to optimality. According to Graham et al. [15], this problem can be denoted by $F/perm, d_i/(C_{max}, \overline{T})$.

2.2. Uncertain Processing Times

In real-world scheduling situations, the uncertainty mainly occurs from environmental parameters and can then be classified into the second category given in [5], *i.e.* decision variables or environmental parameters are subject to perturbation, see Section 3.3. To our knowledge, no other study has been leaded on a multi-objective combinatorial optimization problem with uncertainty on the environmental parameters. However, solutions are sensitive to perturbations coming from many potential sources such as, for instance, release or due date variations, machine breakdowns, unexpected arrival or cancellation of orders, variable processing times, etc. It is obvious that no parameter can be regarded as an exact and precise data and that non-deterministic approaches are required to solve scheduling problems. In this paper, we adopt a proactive stochastic approach where processing times are regarded as uncertain parameters.

Widely investigated in a single-objective form, the stochastic FSP has, to our knowledge, never been investigated in a multi-objective way. Following an analysis, we propose four different general distributions a processing time may follow. Of course, a rigorous statistical analysis, based on real data, is imperative to determine the concrete and exact distribution associated to a given processing time p_{ij} of a real-world optimization problem. A review of existing work on single-objective flowshop scheduling with uncertain processing times can be found in [16].

- **Uniform Distribution.** A processing time p_{ij} can uniformly be included between two values a and b. Then, p_{ij} follows a uniform distribution over the interval $[a, b]$. This kind of distribution is used to provide a simplified model of real industrial cases. For instance, it has already been used in [16, 17].

- **Exponential Distribution.** A processing time p_{ij} may follow an exponential distribution $\mathcal{E}(\lambda, a)$. Exponential distributions are commonly used to model random events that may occur with uncertainty. This is typically the case when a machine is subject to unpredictable breakdowns. For example, processing times have been modeled by an exponential distribution in [18, 19] among others.

- **Normal Distribution.** A processing time p_{ij} may follow a normal distribution $\mathcal{N}(\mu, \sigma)$ where μ stands for the mean and σ stands for the standard deviation. This kind of distribution is especially usual when human factors are observed. A process may also depend on unknown or uncontrollable factors and some parameters can be described in a vague or ambiguous way by the analyst. Therefore, processing times vary according to a normal distribution [16, 20].

- **Log-normal Distribution.** A random variable X follows a log-normal distribution with parameters μ and σ if $\log X$ follows a normal distribution $\mathcal{N}(\mu, \sigma)$. The log-normal distribution is often used to model the influence of uncontrolled environmental variables. For instance, this modeling has already been used in [21, 22].

3. Background on Evolutionary Optimization for Multi-objective and Uncertain Problems

In this section, we give some definitions related to evolutionary multi-objective optimization. Then, we focus on indicator-based evolutionary optimization. At last, we review existing approaches for uncertainty-handling in evolutionary computation, with a particular interest on multi-objective applications.

3.1. Evolutionary Multi-objective Optimization

A Multi-objective Optimization Problem (MOP) can be defined by a set of $n \geq 2$ objective functions $f = (f_1, f_2, \ldots, f_n)$, and a set X of feasible solutions in the *decision space*. In the combinatorial case, X is a discrete set. Let Z denote the set of feasible points in the *objective space*. Without loss of generality, we here assume that $Z \subseteq \mathbb{R}^n$ and that all n objective functions are to be minimized. In the deterministic case, to each decision vector $x \in X$ is assigned exactly one objective vector $z \in Z$ on the basis of the vector function $f : X \to Z$ with $z = (z_1, z_2, \ldots, z_n) = f(x) = (f_1(x), f_2(x), \ldots, f_n(x))$. A decision vector $x \in X$ is said to be *efficient* (or *Pareto optimal*) if there does not exist any other decision vector $x^\star \in X$ such that (i) $f(x^\star)$ is component-wise smaller than or equal to $f(x)$ and (ii) $f(x) \neq f(x^\star)$. One of the most challenging issue in multi-objective optimization is to find the minimal set of efficient solutions (or *efficient set*). However, generating the entire efficient set is usually infeasible, due to, *e.g.*, the complexity of the underlying problem or the large number of optima. Therefore, in many applications, the overall goal is often to identify a good approximation of it. Evolutionary algorithms are commonly used to this end, as they are particularly well-suited to find multiple efficient solutions in a single simulation run. The reader can refer to [3, 4] for more details about evolutionary multi-objective optimization.

3.2. Indicator-Based Evolutionary Multi-objective Optimization

3.2.1. Main Principles

Different interpretations of what a good efficient set approximation is are possible, and the definition of approximation quality strongly depends on the decision-maker preferences. In [6], Zitzler and Künzli assume that the optimization goal is given in terms of a binary quality indicator $I_\Omega : \Omega \times \Omega \to \mathbb{R}$, where Ω stands for the set of all efficient set approximations. Thereby, a

value $I_\Omega(A, B)$ quantifies the difference in quality between two efficient set approximations $A, B \in \Omega$. So, if R denotes the optimal efficient set (or any other reference set), the overall optimization goal can be formulated as:

$$\arg\min_{A \in \Omega} I_\Omega(A, R) \tag{3}$$

As noticed by the original authors, R does not have to be known since it just serves the formalization of the optimization goal. Therefore, R being fixed, I_Ω can be seen as a unary function that assigns, to each approximation set, a value reflecting its quality with respect to the optimization goal. If I_Ω is dominance preserving [6], $I_\Omega(A, R)$ is minimum for $A = R$. The indicator, that can be chosen according to the decision-maker preferences, is thus directly used in the fitness assignment scheme of the so-called Indicator-Based Evolutionary Algorithm (IBEA) proposed in [6]. As a consequence, the fitness value F of a solution x belonging to a population P measures the usefulness of x according to the optimization goal:

$$F(x) = I_\Omega(P \setminus \{x\}, \{x\}) \tag{4}$$

An interesting property of indicator-based optimization is that no specific diversity preservation mechanism is generally required, according to the indicator being used.

3.2.2. Binary Quality Indicators

In addition to I_Ω, let us define two other kinds of binary quality indicators as follows.

- First, $I_Z : Z \times Z \to \mathbb{R}$ is a function whose purpose is to compare two objective vectors z and $z' \in Z$.
- Second, $I_X : X \times X \to \mathbb{R}$ is a function whose purpose is to compare two decision vectors x and $x' \in X$.

Thus, these three types of binary indicators all consist of mapping two elements to a real number by comparing their respective quality relatively to each other. Even if there might exist stand-alone versions for every kinds of indicators presented above, a possibility is to build an I_X-indicator from an I_Z-indicator. Let us take the example of the binary additive ϵ-indicator. First defined to compare two non-dominated set approximations [23], it can

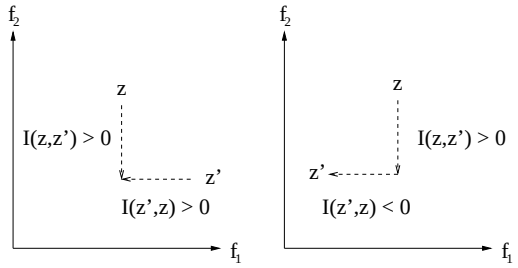

Figure 2: Illustration of the the binary additive ϵ-indicator ($I_{Z,\epsilon+}$).

naturally be used to compare two single objective vectors. In the latter case, it is defined as follows:

$$I_{Z,\epsilon+}(z, z') = \max_{i \in \{1,\ldots,n\}} (z_i - z'_i) \tag{5}$$

As illustrated in Figure 2, $I_{Z,\epsilon+}$ computes the minimum value by which an objective vector $z \in Z$ can or has to be translated in the objective space to weakly dominate another point $z' \in Z$. Thus, in the deterministic case, it is a common place to compare two decision vectors x and $x' \in X$ with the help of $I_{Z,\epsilon+}$, as defined in the following equation.

$$I_{X,\epsilon+}(x, x') = I_{Z,\epsilon+}(f(x), f(x')) \tag{6}$$

This will not be the case in the uncertain case, where a set of objective vectors can be associated to a single solution. Now, to evaluate the quality of a solution $x \in P$ in comparison to a whole population P (and then to compute the fitness value of x within IBEA), there exists several approaches. As presented in [6], a simple possibility is to sum up the I_X-values of each population item in comparison to the remainder of the population.

$$F(x) = \sum_{x' \in P \setminus \{x\}} I_X(x', x) \tag{7}$$

Note that other examples of binary quality indicators than can be used within indicator-based multi-objective search methods can be found in [6, 23].

3.2.3. Indicator-Based Evolutionary Algorithm

A detailed description of the Indicator-Based Evolutionary Algorithm (IBEA) introduced in [6] is reproduced in Figure 3. The selection scheme

IBEA

1. *Initialization.* Start with a user-given initial population P of size N, or generate it randomly.
2. *Fitness assignment.* Calculate fitness values of individuals in P; *i.e.* $F(x) = \sum_{x' \in P \setminus \{x\}} I_X(x', x)$.
3. *Environmental selection.* Iterate the following steps until the size of the population P does not exceed N:
   (a) Choose an individual $x^\star \in P$ with the smallest fitness value; *i.e.* $F(x^\star) \leq F(x)$ for all $x \in P$.
   (b) Remove $x^\star$ from P.
   (c) Update the fitness values of the remaining individuals; *i.e.* $F(x) = F(x) - I_X(x^\star, x)$ for all $x \in P$.
4. *Termination.* If a stopping condition is satisfied, then stop and return P.
5. *Mating selection.* Perform binary tournament selection with replacement on P in order to fill the temporary mating pool P'.
6. *Variation.* Apply recombination and mutation operators to the mating pool P' and add the resulting offspring to P. Go to Step 2.

Figure 3: Outline of the Indicator-Based Evolutionary Algorithm (IBEA), adapted from [6].

for reproduction is a deterministic tournament between two randomly chosen individuals. The replacement strategy consists in deleting, one-by-one, the worst individuals, and in updating the fitness values of the remaining solutions each time a solution is removed. This process is iterated until the required population size is reached.

3.3. Evolutionary Optimization in Uncertain Environments

In their review on evolutionary optimization approaches in the presence of uncertainty, Jin and Branke [5] classify uncertainties into four categories: (*i*) noisy objective function, (*ii*) robustness, (*iii*) approximated objective function or (*iv*) time-varying objective function. Note that the two last classes are not addressed in this paper. First, we assume that an approximated objective function is closely related to the problem to be solved,

whereas we attempt to be as generic as possible from an algorithmic point of view. Second, we believe that taking dynamic variations into account within a reactive approach is a complete different issue than dealing with other kinds of uncertainty. However, the first two classes are closely related the one to the other. Indeed, while dealing with a noisy objective function, an expected objective function is, in practice, often approximated by using a set of sample values. This is also the case when searching for *robust* solutions. In this second category, the authors include two sub-classes. The first one relates to problems where design variables are subject to perturbations or change after the search process, while the second one relates to variations on the environmental parameters, as it is the case for the FSP investigated in this paper. As noticed in [5], very few studies have already investigated the optimization of MOPs in uncertain environments.

Note that an existing class of search methods consists of applying multi-objective optimization approaches to find robust solutions for single-objective optimization problems. In this case, a robustness measure is generally defined and a corresponding objective function is added to the problem formulation. The problem is then converted into a MOP where performance and robustness are treated as separate goals. An example of such approach is given in [24]. Note that similar techniques can be applied to MOPs (see, for instance, [25]), and the issue of robustness in multi-objective optimization has already been addressed in [26] for problems where solutions are sensitive to decision variables perturbations.

To our knowledge, existing approaches for solving MOPs that are subject to uncertainties are the following ones. First, Teich [27] and Hughes [28] independently suggested to extend the concept of Pareto dominance for the stochastic case. They integrate a probabilistic dominance relation into the fitness assignment scheme of a multi-objective search method. But both studies make an assumption on the probability distribution the objective functions follow. In [29], another ranking scheme, based on the average value and on the variance of a set of objective vector samples, is presented. This ranking strategy is integrated into NSGA-II [30] to solve noisy MOPs. In [26], Deb and Gupta propose a similar strategy. To a feasible solution is associated an average value for each dimension of the objective space, determined over a given sample of objective vectors. A classical multi-objective search method, usually designed for deterministic MOPs, is then applied over these approximated objective vectors. More recently, Barrico and Antunes [31] introduced a process to quantify the degree of robustness of a solution that is inte-

grated into a multi-objective evolutionary algorithm. Additionally, Goh and Tan [32] studied the impact of noisy objective functions on the performance of a set of evolutionary multi-objective optimization algorithms. In the same paper, the authors propose some features to handle noise, including an experiential learning directed perturbation operator, a gene adaptation selection strategy and a "possibilistic" archiving methodology. Finally, the concept of indicator-based multi-objective optimization introduced in the previous section has been extended in order to take stochasticity into account in [8]. According to the classification given in [5], all presented methods deal either with noisy objective functions or with robustness on design variables, even if all of them can generally be applied in both cases.

To summarize, a small but increasing number of studies are devoted to the resolution of MOPs in uncertain environments. A first remark is that existing approaches generally assume specific characteristics for the probability distribution that is associated to a given objective function. Therefore, they exploit problem knowledge that may not be available in practice. A second remark is that the methods proposed in the literature have all been experimented on continuous optimization problems, and mainly where noisy objective functions are involved. A last remark is that a key issue while taking uncertainty into account for MOPs is performance assessment, as no suitable protocol has been designed up to now.

4. Indicator-Based Evolutionary Algorithms for Multi-objective Optimization in Uncertain Environments

In this section, a number of new approaches, based on indicator-based selection, are proposed to handle multi-objective optimization problems subject to uncertainty.

4.1. An Uncertainty-Handling Approach based on Scenarios

As explained in section 3.1, in deterministic multi-objective optimization, a single objective vector $z \in Z$ is assigned to every solution $x \in X$, based on the vector function f. Thus, $f(x)$ defines the 'true' evaluation of x, and f is assumed to represent a deterministic mapping from the decisional space X to the objective space Z. While taking uncertainty into account, each time a solution is evaluated, the resulting objective vector can possibly map to a different point in the objective space. We will then consider that f does not represent a deterministic mapping from X to Z, but that a (potentially

infinite) set of objective vectors is now associated to a given solution x. We assume that the 'true' evaluation of a solution is not known before the end of the search process. No hypothesis is done on any probability distribution associated to the objective function(s), the decision variable(s) or to the environmental parameter(s), because such distribution is generally not known in advance and may differ for any solution.

Thus, in the uncertain case, each solution $x \in X$ is assigned a sample of objective vectors. The higher the degree of uncertainty, the larger the variance among the objective vectors resulting from multiple and independent evaluations of x. It is then necessary to determine a satisfactory sample size, as the sampling step could be expensive in terms of computation time. Indeed, since evaluating a solution can be expensive, a good trade-off between a fine accuracy and a reasonable time consumption must be found. In addition, the potential number of evaluations for one solution is generally limited in practice. In this case, only a subset of these evaluations must be used during the search process, the remaining part being used to assess the performance of the algorithm(s). More formally, for each solution $x \in X$, we assume that a set of independent and equally probable evaluations are computed. Thus, a sample of objective vectors $\{z^{(i)}\}_{i=1}^{p}$ is now associated to each solution. Let us consider two arbitrary solutions. Two cases may arise. First, it may happen that one evaluation of a given solution is strictly independent of all the other evaluations performed until now. Then, all the objective vectors of all the evaluated solutions are independent with each other, and the size of the objective vector sample can differ from a solution to another. Another alternative is to consider a finite set of independent and equally probable scenarios $S = \{s_1, s_2, \ldots, s_p\}$, as in *robust optimization* [33]. $S(x) = \{z^{(1)}, z^{(2)}, \ldots, z^{(p)}\}$ is the sample of independent evaluations of a solution $x \in X$. The sample elements $z^{(i)}$ associated to x represents the value of the objective vector of x if scenario s_i occurs (see Figure 4). Herewith, given two solutions $x, x' \in X$, we assume that the corresponding objective vectors $\{z^{(i)}\}_{i=1}^{p}$ and $\{z'^{(j)}\}_{j=1}^{p'}$ are paired and of same size ($p = p'$). Thus, given a scenario s_i, the corresponding objective vectors $z^{(i)}$ and $z'^{(i)}$ are comparable with each other.

4.2. Uncertainty-Handling Quality Indicators

The proposed algorithms for multi-objective optimization in uncertain environments are based on the indicator-based fitness assignment strategy introduced in Section 3.2, and proposed in [6]. In order to take uncertainty

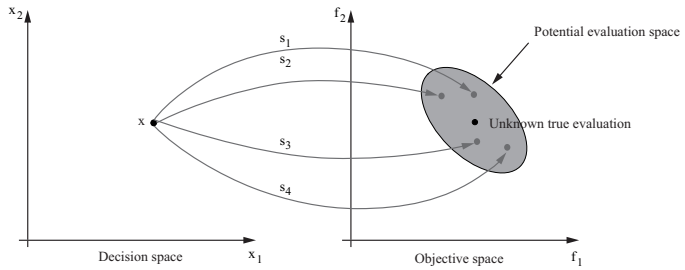

Figure 4: An example of multiple evaluations of a solution x considering four scenarios $S = \{s_1, s_2, s_3, s_4\}$.

into account, we introduce a set of four uncertainty-handling quality indicators that could be used within IBEA, or any other indicator-based metaheuristic. These indicators correspond to different strategies among which the decision-maker can choose according to his/her preferences or according to the kind of problem he/she has to face. Let us assume that a binary indicator $I_Z : Z \times Z \to \mathbb{R}$ dedicated to the comparison of two objective vector has been defined, for instance $I_{Z,\epsilon+}$ as defined in Section 3.2. In contrast with the deterministic case, the interpretation of this kind of indicator from the objective space to the decision space is no more obvious. Indeed, due to the use of scenarios, not a single objective vector is assigned to a solution, but rather a set of objective vectors corresponding to the collected sample.

$$I_X(x, x') \neq I_Z(f(x), f(x')) \tag{8}$$

As a consequence, the proposed approaches consist in defining different aggregation strategies of the information available in the two objective vector sets $\{z_k^{(1)}, z_k^{(2)}, \ldots, z_k^{(p)}\}$ and $\{z_k'^{(1)}, z_k'^{(2)}, \ldots, z_k'^{(p')}\}$, respectively associated to two solutions $x, x' \in X$, in a unique scalar I_X-value. In that sense, this work extends the contribution of [8], where such indicators have been proposed to deal with noisy objective functions for the particular case of the ϵ-indicator. The approaches presented in this paper are independent of the I_Z indicator under consideration. The obtained I_X-value could then take place in a fitness assignment scheme based on a binary quality indicator.

Two kind of indicators, corresponding to two levels of aggregations, are proposed (Figure 5). First, indicators based on the objective values lay on the sample $\{z_k^{(1)}, z_k^{(2)}, \ldots, z_k^{(p)}\}$ associated to a solution $x \in X$. These approaches consist in transforming the sample associated to a solution in a

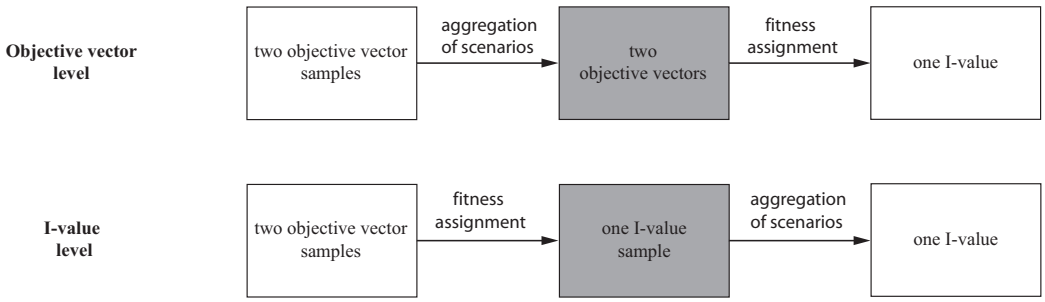

Figure 5: Two classes of uncertainty-handling multi-objective indicators: Objective vector level indicators and I-value level indicators.

single objective vector by means of a representative value for each objective function. Note that these objective vector level approaches are based on generic concepts that can be applied outside indicator-based search. In this case, a single representative vector can be seen as the determinist objective vector, and the problem can be handled as in the deterministic case. The second class of indicators, based at the I-value level, consists in computing a sample of I-values extracted from the two samples of objective vectors collected from a pair of solutions from the current population. For each class, two indicators are proposed. They correspond to an average-case and a worst-case strategy, respectively.

4.2.1. Objective Vector Level Approaches

The first set of quality indicators designed to deal with uncertainty rely at the objective vector level. They are based an the objective vector sample $\{z_k^{(1)}, z_k^{(2)}, \ldots, z_k^{(p)}\}$ associated with a solution $x \in X$. They consist in transforming the objective vector sample into a single point. Then, the resulting objective vectors associated with two solutions can be compared using an I_Z indicator, as in the deterministic case.

Worst-case Objective Vector Indicator. A *worst-case objective vector* (z^{worst}) can be settled as below. This results in a *pessimistic* approach, with a high risk aversion.

$$I_X^{z^{worst}}(x, x') = I_Z(z^{worst}, z'^{worst}) \tag{9}$$

such that $z_k^{worst} = \max_{i \in \{1,\ldots,p\}} z_k^{(i)}$, $k \in \{1, \ldots, n\}$. Let us remind that a minimization problem is here assumed. As a consequence, such indicator is somehow linked with the field of robust optimization [33].

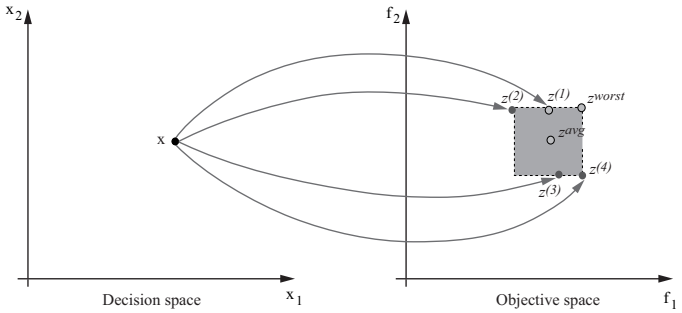

Figure 6: Illustration of the representative objective vectors considered in objective vector level approaches: the worst-case objective vector (z^{worst}) and the average-case objective vector (z^{avg}).

Average-case Objective Vector Indicator. When taking uncertainty into account, most approaches aim to find the 'optimal' expected value with respect to each objective function. Let us define an *average-case objective vector* (z^{avg}), together with the corresponding $I_X^{z^{avg}}$ indicator.

$$I_X^{z^{avg}}(x, x') = I_Z(z^{avg}, z'^{avg}) \tag{10}$$

such that $z_k^{avg} = \frac{1}{p}\sum_{i=1}^{p} z_k^{(i)}$, $k \in \{1, \ldots, n\}$. This idea is commonly used in the single-objective case [5], and has already been suggested in several studies from multi-objective optimization [26, 29].

The worst-case and the average-case objective vectors are illustrated in Figure 6.

4.2.2. I-value Level Approaches

Unlike previous strategies, both approaches presented below are based on the I-values obtained by means of the objective vector samples of two solutions. They are then specific to indicator-based approaches. Let $x, x' \in X$ be two arbitrarily chosen solutions from the current population, and let $\{z_k^{(1)}, z_k^{(2)}, \ldots, z_k^{(p)}\}$ and $\{z_k'^{(1)}, z_k'^{(2)}, \ldots, z_k'^{(p')}\}$ be their associated objective vector samples. For each of the indicator presented below, two cases may happen. If the scenario associated with the i^{th} evaluation of the two solutions is the same for all $i \in \{1, 2, \ldots, p\}$, then the objective vector samples of the two solutions are *paired*. Otherwise, if the scenarios are different or randomly chosen for each evaluation, then the objective vector samples are

independent. As proposed by Basseur and Zitzler [8], our approaches consist in defining an aggregation strategy from the sample of I_Z-values obtained from the pairwise comparison of the objective vectors of two solutions. In the case of paired samples, the strategy will be based on the following I_X-value samples of size p, with I_Z being an arbitrary chosen indicator.

$$I_Z(z^{(1)}, z'^{(1)}), \ldots, I_Z(z^{(p)}, z'^{(p)}) \tag{11}$$

In the case of independent objective vector samples, the considered sample will rather be the following one, of size $(p \times p')$.

$$I_Z(z^{(1)}, z'^{(1)}), \ldots, I_Z(z^{(i)}, z'^{(j)}), \ldots, I_Z(z^{(p)}, z'^{(p')}) \tag{12}$$

Thus, the idea of this second class of approaches consists in taking all the objective vector combinations into account. Once again, two indicator level strategies will be proposed, corresponding to the worst-case and the average-case, respectively. Both work on the pairwise comparison of the objective vector samples collected for two solutions, based on a I_Z indicator. Without loss of generality, we here assume that I_Z-values are to be minimized.

Worst-case I-value Level Indicator. Given an I_Z indicator and two solutions x, x' from the current population, a possible approach consists in considering the worst element of the I_Z-value sample. The resulting strategy is pessimistic, given that the best-performing objective vector associated with x with respect to the objective vector sample of x' is taken into account. A worst case indicator can be defined as follows for paired samples:

$$I_X^{worst}(x, x') = \max_{i \in \{1,\ldots,p\}} I_Z(z^{(i)}, z'^{(i)}) \tag{13}$$

For independent samples:

$$I_X^{worst}(x, x') = \max_{i \in \{1,\ldots,p\}, j \in \{1,\ldots,p'\}} I_Z(z^{(i)}, z'^{(j)}) \tag{14}$$

Average-case I-value Level Indicator. As well, let us define an average-case indicator where the mean of I-values is used in the fitness assignment scheme. For paired samples:

$$I_X^{avg}(x, x') = \frac{1}{p} \sum_{i=1}^{p} I_Z(z^{(i)}, z'^{(i)}) \tag{15}$$

For independent samples:

$$I_X^{avg}(x, x') = \frac{1}{p \cdot p'} \sum_{i=1}^{p} \sum_{j=1}^{p'} I_Z(z^{(i)}, z'^{(j)}) \tag{16}$$

4.2.3. Summary

The proposed uncertainty-handling indicators can now take place into the indicator-based fitness assignment scheme based on a binary quality indicator. They can then be used inside EMO algorithms like IBEA. They give birth to a set of evolutionary algorithms that are able to handle multi-objective optimization problems in uncertain environments. These strategies allow to specify different kinds of decision-maker preferences in terms of uncertainty-handling, by means of the definition of a simple indicator. Thus, only two levels must now be defined to instantiate an IBEA-like algorithm in order to take uncertainty into account: an indicator to compare two objective vectors (as in the deterministic case) and an additional uncertainty-handling indicator. Four possible uncertainty-handling indicators have been proposed in the paper.

5. Experimental Analysis

This section presents an experimental analysis of the approaches proposed in the paper applied to the uncertain and multi-objective FSP introduced in Section 2. The issue of performance assessment is first discussed in the context of multi-objective optimization in uncertain environments, and the evolutionary algorithms presented in the previous section are experimented on a number of FSP instances of different structure and size.

5.1. Performance Assessment

In the deterministic case, approximating the efficient set is itself a bi-objective problem, as the aim is to find a Pareto set approximation with both high convergence and diversity properties. In recent years, the performance assessment of EMO algorithms has been widely investigated in the literature [23]. Nevertheless, to the best of our knowledge, this issue has not been satisfactorily addressed so far for multi-objective optimization in uncertainty environments.

5.1.1. Set-based Quality Indicators and Statistics

The following performance assessment approach applies to deterministic multi-objective optimization problems. The adaptations required to handle the uncertain nature of the problem are discussed afterwards. In this study, a set of 20 runs *per* instance and *per* algorithm is performed. In order to evaluate the quality of the Pareto set approximations for every instance we experimented, we follow the protocol proposed by Knowles et al. [34]. For a given instance, we first compute a reference set $Z_N^\star$ containing the whole set of non-dominated objective vectors we obtained during all our experiments. Second, we define $z^{min} = (z_1^{min}, \ldots, z_n^{min})$ and $z^{max} = (z_1^{max}, \ldots, z_n^{max})$, where z_k^{min} (resp. z_k^{max}) denotes the lower (resp. upper) bound with respect to the k^{th} objective function for all the solutions we obtained. In order to give a roughly equal range to all the objective functions, values are normalized with respect to z^{min} and z^{max}.

Let us consider an efficient set approximation A. In order to measure the quality of A in comparison to $Z_N^\star$, we compute the difference between these two sets by using the unary hypervolume metric [23], z^{max} being the reference point. The hypervolume difference indicator ($\mathrm{I}^-_{\Omega,H}$) computes the portion of the objective space that is weakly-dominated by $Z_N^\star$ and not by A. Furthermore, we also consider the additive ϵ-indicator [23] presented in Section 3.2. The unary additive ϵ-indicator ($\mathrm{I}^1_{\Omega,\epsilon}$) gives the minimum factor by which an approximation A can or has to be translated in the objective space to weakly-dominate the reference set $Z_N^\star$.

As a consequence, for each test instance, we obtain 20 $\mathrm{I}^-_{\Omega,H}$-values and 20 $\mathrm{I}^1_{\Omega,\epsilon}$-values, corresponding to the 20 runs, *per* algorithm. Once all these values are computed, we perform a statistical analysis for a pairwise comparison of methods. To this end, we use the Wilcoxon signed rank test. Details for this statistical testing procedure are given in [34]. Hence, for a given test instance, and with respect to the indicator under consideration, this statistical test reveals if the sample of approximation sets obtained by a given search method is significantly better than the one of another search method, or if there is no significant difference between them. For the sake of conciseness, we only report how many algorithms obtained statistically better results than the corresponding algorithm for the instance under consideration. In other words, a value of *zero* means that no other method generated significantly better results. Note that all the performance assessment procedures have been achieved using the performance assessment tools provided in PISA [35].

5.1.2. General Comments

A common practice consists in converting the uncertain problem into a deterministic one be means of a particular strategy, and then to assess the performance of the approximation sets with regards to this deterministic formulation. For instance, the mean or the expected value over a sample of objective vectors are used in [36, 27, 37]. Another way to do so consists in considering the deterministic model from which the uncertain problem has been derived from as the 'true' single scenario, which is used to evaluate the outputs [38, 32, 39]. However, as pointed out earlier, in real-world situations, there does not exist an single (average-case or 'true') evaluation associated with a given solution, or at least it is usually not known in advance. Thus, no realization of stochastic data can be considered more reliable than another, whatever it is issued from deterministic data or from an arbitrary realization. As a consequence, we pay a particular attention to the following issues.

- At the end of the search process, we take a sample of possible evaluations into account rather than a unique evaluation in order to assess the performance of the approximation obtained by a given algorithm.
- We also re-evaluate the solutions by means of uncertain data that differ from the ones used during the search process. This allows to avoid the influence of the search process, and then provides a more fair and unbiased comparison between the algorithms.

5.1.3. Methodological Approaches

Two methodological approaches are proposed to assess the performance of algorithms for multi-objective optimization in uncertain environments. They are based on similar ideas than for the resolution approaches, presented in Section 4, and follow the same reasoning as illustrated in Figure 4. The first class is based at the objective vector level whereas the second class is based on I-values.

Objective Vector Level Approach. Let A be a set of solutions found by a given algorithm. To each solution $x \in A$ is then associated a set of objective vectors $\{z_k^{(1)}, z_k^{(2)}, \ldots, z_k^{(q)}\}$ of size q, where q is the number of scenarios considered for performance assessment. The first approach consists in computing, for each solution, a single objective vector corresponding to the worst-case (z^{worst}) and to the average-case (z^{avg}), as explained in Section 4.2.1. As a consequence, each solution is now associated to a single objective vector aggregating the

information related to uncertainty according to the worst- or the average-case.

I-value Level Approach. The second approach is based on the indicator values obtained by the approximation sets found by the algorithms. Let A and B be two solution sets obtained by two different algorithms on the same problem instance. To each solution $x \in A$ is associated a set of objective vectors $\{z_k^{(1)}, z_k^{(2)}, \ldots, z_k^{(q)}\}$ of size q, and to each solution $x' \in B$ is associated a set of objective vectors $\{z_k'^{(1)}, z_k'^{(2)}, \ldots, z_k'^{(q')}\}$ of size q'. Once again, two cases may arise: either the objective samples are paired (then $q = q'$), or they are independent. In the following, we assume that they are paired, even if the reasoning can easily be adapted to the case of independent samples. We assume that an algorithm A has obtained r approximations for a given instance, issued from r independent executions. If we consider a single scenario $s \in S$, a classical analysis can be performed, with respect to a given indicator I. As a consequence, we obtain a set of r scalar values $\{I(A_1), I(A_2), \ldots, I(A_r)\}$, where $I(A_i)$ is the value obtained by the ith execution of the algorithm A with respect to indicator I and scenario s. Now, if we consider q scenarios simultaneously, we obtain the following set of I-values:

	exec. 1	exec. 2	$\cdots$	exec. r
scenario s_1	$I(A_1^1)$	$I(A_2^1)$	$\cdots$	$I(A_r^1)$
scenario s_2	$I(A_1^2)$	$I(A_2^2)$	$\cdots$	$I(A_r^2)$
$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$
scenario s_q	$I(A_1^q)$	$I(A_2^q)$	$\cdots$	$I(A_r^q)$

Then, for a given execution $j \in r$, we compute the corresponding I-value with respect to the worst-case and to the average-case of the following set: $\{I(A_j^1), I(A_j^2), \ldots, I(A_j^r)\}$. This is performed for all the algorithms. Therefore, a single (scalar) I-value is obtained *per* algorithm, *per* instance and *per* scenario. At last, we use a statistical test to determine if a given algorithm obtains significantly better results than another algorithm with respect to indicator I, and to worst- or average-case preferences.

5.2. Experimental Design

5.2.1. Benchmark Instances

In order to experiment the different approaches introduced in the paper, we propose a set of benchmark instances[1] built from Taillard's instances for the single-objective FSP [40]. In the original instances, the processing times are generated randomly, according to a uniform distribution $p_{ij} \in \mathcal{U}(0, 99)$. These instances are extended here, first for the multi-objective case, then for the uncertain case. These instances contain uncertain processing times and due dates for different problem sizes.

Benchmark Instances for the Multi-objective FSP. First, we need to extend the instances for the multi-objective case by adding a due date for every job. These dates were fixed using a random value chosen between $\overline{p} \times M$ and $\overline{p} \times (N + M - 1)$, where N stands for the number of jobs, M for the number of machines and $\overline{p}$ for the average value of previously generated processing times for the instance under consideration. Thus, a due date d_i roughly lies between the average completion date of the first scheduled job and the average completion date of the last scheduled job. An instance denoted by $N \times M \times i$ represents the i^{th} instance made of N jobs and M machines.

Benchmark Instances for the FSP with Uncertain Processing Times. To generate uncertain data, we apply the four probability distributions introduced in Section 2 over initial deterministic processing times by means of a configuration file. We choose to allow to configure this uncertainty over the machines only, by specifying, for each machine, a probability distribution associated with its parameters or some proportions depending on its central tendency. Thus, there exists a correlation between all the processing times of a given machine. Moreover, as in real-world problems, each time a randomness is applied on an initial deterministic test instance using the same configuration file, the processing times obtained in the resulting instances are different.

5.2.2. Implementation Issues

The uncertainty-handling indicators introduced in Section 4 and the related indicator-based evolutionary algorithms have all been implemented un-

[1]Benchmark instances for both the deterministic and the stochastic cases are all available at the following URL: `http://www.lifl.fr/~liefooga/benchmarks/`.

der the ParadisEO-MOEO software framework[2] [41]. ParadisEO-MOEO is a white-box object-oriented C++ library dedicated to the reusable design, implementation and analysis of metaheuristics for multi-objective optimization. Note that the algorithms share the same base components for a fair comparison between them.

The problem-related components designed for the FSP addressed in this paper are given below. The solution representation is based on permutations of size N, where N stands for the number of jobs for the instance under consideration. The population is initialized with randomly generated solutions. The crossover operator consists of a two-point crossover, and the mutation operator consists of a shift (or insertion) mutation. Both variation operators are described in [42].

5.2.3. Scenarios

In our experiments, we consider a set of $(p+q)$ equally probable scenarios: p scenarios devoted to the search process and $q = 20$ scenarios for the performance evaluation process. Two p-values are here considered: $p = 10$ and $p = 20$. These values correspond to an objective vector sample of size 10 and 20, respectively, associated with each solution. Every scenario corresponds to a realization of the uncertain environmental parameters, *i.e.* processing times. To create these scenarios, we use the stochastic models defined in Section 2. Then, for a given instance, we generate $p + q$ independent scenarios, for which the processing times follow different probability distributions in the following way, where p_{ij} denotes the processing time of Job N_i on Machine M_j.

- Uniform distribution: $p_{ij} \sim \mathcal{U}(a = (1 - \alpha) \times p_{ij},\, b = (1 + \alpha) \times p_{ij})$;
- Normal distribution: $p_{ij} \sim \mathcal{N}(\mu = p_{ij},\, \sigma = \alpha \times p_{ij})$;
- Exponential distribution: $p_{ij} \sim \mathcal{E}(a = p_{ij},\, \lambda = \frac{1}{\alpha \times p_{ij}})$;
- Log-normal distribution: $p_{ij} \sim \mathit{log}\text{-}\mathcal{N}(\mu = \log p_{ij},\, \sigma = \alpha \times \log p_{ij})$.

In the following, we consider various distributions, *i.e.* the probability distribution of the processing times is different on every machine. In any case, the

[2]ParadisEO is available for download at the following URL: `http://paradiseo.gforge.inria.fr`.

Table 1: Stopping condition: number of evaluations.

Instance	Number of evaluations
$20 \times 05 \times 01$	500000
$20 \times 05 \times 02$	500000
$20 \times 10 \times 01$	1000000
$20 \times 10 \times 02$	1000000
$20 \times 20 \times 01$	2000000
$50 \times 05 \times 01$	5000000
$50 \times 10 \times 01$	10000000
$50 \times 20 \times 01$	20000000

central tendency of the distribution always corresponds to the deterministic processing time p_{ij} for the instance under consideration. The α parameter allows to tune the degree of deviation of the processing times. In the following, two α-values will be considered $\alpha \in \{0.10, 0.20\}$. For instance, they correspond to a deviation of $\pm 10\%$ and $\pm 20\%$ for a uniform distribution, respectively.

5.2.4. Resolution Approaches under Study

The algorithms under consideration for the experiments correspond to the four proposed indicators which are integrated in the IBEA algorithm. Furthermore, we will also consider a naive approach based on a single scenario, denoted by z^1. The objective vector sample size associated with a solution is then 1, and the resulting algorithm behaves like in the deterministic case. Furthermore, we consider the approach proposed by Basseur and Zitzler [8], which is based on an estimation of the expected I_X-value associated with a solution for the particular case of the ϵ-indicator. Furthermore, let us note that the general concept of the objective vector level approach based on the average-case is frequently encountered in the single-objective case [5], and has been extended to multi-objective optimization in [26, 29], among others.

5.2.5. Parameter Setting

The remaining parameters are set as follows. First, the stopping condition is based on a maximum number of evaluations, given in Table 1. As a consequence, the higher the sample size, the lower the number of iterations of the algorithm. This will allow us to evaluate the impact of the sample size on the overall algorithm performance. The population size is set to 100 solutions. The crossover and mutation rates are set to 0.25 and 1.0,

Table 2: Parameter setting used in the paper for the experimental analysis.

Description	Parameter	Value(s)
Algorithm		
Solution representation		Permutation
Population size		100
Crossover operator		Two-point operator
Crossover rate		0.25
Mutation operator		Shift (insertion) operator
Mutation rate		1.0
Objective vector level indicator	I_Z	$I_{Z,\epsilon}$
Uncertainty-handling solution level indicator	I_X	$\{I_X^{z^1}, I_X^{z^{worst}}, I_X^{z^{avg}}, I_X^{worst}, I_X^{avg}\}$
Stopping condition (number of evaluations)		(see Table 1)
Instances		
Number of jobs	N	$\{20, 50\}$
Number of machines	M	$\{5, 10, 20\}$
Deviation level	α	$\{0.10, 0.20\}$
Number of scenarios (search process)	p	$\{10, 20\}$
Number of scenarios (performance assessment)	q	20

respectively. A summary of all the parameters used in the paper is given in Table 2.

5.3. Experimental Results

First of all, notice that only the results related to a different distribution for each machine are presented. We also experimented our approaches for each probability distribution separately (*i.e.* uniform distribution only, exponential distribution only, normal distribution only, and log-normal distribution only), but the results did not bring to light significant differences with a different distribution over each machine. The worst-case and average-case approaches are respectively compared with each other according to the two performance assessment process presented in the paper. For instance, the comparison with respect to the average-case objective vector values is performed only on the approaches z^{avg} and I^{avg}. First, we compare our approaches to the one proposed by Basseur and Zitzler [8]. The latter method is based on an estimation of the expected value of the $I_{Z,\epsilon}$-values associated with a solution. The corresponding algorithm is much more time-consuming than our proposed algorithms. Furthermore, its performance is not compet-

itive with respect to the performance assessment used in the paper. This is the reason why numerical results for this approach are omitted in the paper.

Table 3 and Table 4 provide computational results with respect to the objective vector level performance assessment approach for the worst-case and the average-case, respectively. Table 5 and Table 6 are the counterpart for the I-value level performance assessment. For each instance and each algorithm, the values reported in the tables number of algorithms that obtain significantly better results for the instance under consideration. A value of '**0**' means that no other algorithm performs significantly better than the one under consideration.

First, let us remark that uncertainty-handling evolutionary algorithms obtained an overall better performance than z^1, where a single scenario is considered. The only instance where z^1 performs significantly better than the other algorithms is $20 \times 10 \times 01$, for worst-case objective vector values. This proves that deterministic approaches cannot compete, even with very basic uncertainty-handling approaches. With respect to the representative objective vector, there is no significant difference between indicator level and objective vector level approaches for high deviations on the processing times ($\alpha = 0.20$), but the former generally outperforms the latter for a small deviation ($\alpha = 0.10$). We can also remark the very good performance of the average-case objective vector approach when considering an average-case performance assessment protocol. With respect to the I-value level performance assessment, there is generally no significant difference between indicator level and objective vector level approaches for $\alpha = 0.10$, even if the latter seems to be slightly better. On the contrary, for a higher deviation ($\alpha = 0.20$), it appears that indicator level approaches perform better most of the time. At last, with regards to the influence of the sample size associated with a solution (the number of scenarios), it seems that it is related to the level of deviation of the processing times. Indeed, in most cases, for a given resolution approach, a sample of size of $p = 10$ achieves better results for a deviation degree of $\alpha = 0.10$, whereas a sample of size $p = 20$ performs better for a deviation degree of $\alpha = 0.20$.

6. Discussion

This paper deals with the modeling of multi-objective optimization problems in uncertain environments, and with the design and analysis of evolutionary algorithms to tackle them. We argued that many real-world applica-

Table 3: Algorithm comparison according to the worst-case objective vector values.

		$I^-_{\Omega,H}$					$I^1_{\Omega,\epsilon+}$			
	z^1	z^{worst}		I^{worst}		z^1	z^{worst}		I^{worst}	
		10	20	10	20		10	20	10	20
	$\alpha = 0.10$									
$20 \times 05 \times 01$	4	**0**	2	**0**	2	4	**0**	2	**0**	2
$20 \times 05 \times 02$	4	**0**	**0**	**0**	1	4	**0**	**0**	**0**	**0**
$20 \times 10 \times 01$	2	**0**	**0**	1	3	1	**0**	**0**	3	3
$20 \times 10 \times 02$	4	1	**0**	**0**	1	4	1	2	**0**	**0**
$20 \times 20 \times 01$	4	2	1	**0**	**0**	4	2	**0**	**0**	**0**
$50 \times 05 \times 01$	3	**0**	**0**	**0**	1	2	**0**	**0**	2	2
$50 \times 10 \times 01$	4	**0**	1	**0**	**0**	4	**0**	1	**0**	**0**
$50 \times 20 \times 01$	4	1	**0**	**0**	**0**	4	2	1	1	**0**
	$\alpha = 0.20$									
$20 \times 05 \times 01$	4	3	**0**	**0**	1	4	2	**0**	**0**	**0**
$20 \times 05 \times 02$	4	1	3	**0**	1	4	2	3	1	**0**
$20 \times 10 \times 01$	3	**0**	**0**	2	**0**	4	1	**0**	2	**0**
$20 \times 10 \times 02$	**0**	3	2	1	1	**0**	3	2	**0**	**0**
$20 \times 20 \times 01$	4	2	1	**0**	1	4	1	1	**0**	1
$50 \times 05 \times 01$	3	3	1	2	**0**	3	3	1	1	**0**
$50 \times 10 \times 01$	4	2	3	**0**	1	4	2	3	**0**	1
$50 \times 20 \times 01$	4	3	**0**	1	**0**	4	3	**0**	**0**	**0**

Table 4: Algorithm comparison according to the average-case objective vector values.

		$I^-_{\Omega,H}$					$I^1_{\Omega,\epsilon+}$			
	z^1	z^{worst}		I^{worst}		z^1	z^{worst}		I^{worst}	
		10	20	10	20		10	20	10	20
	$\alpha = 0.10$									
$20 \times 05 \times 01$	1	**0**	3	**0**	3	1	**0**	3	**0**	3
$20 \times 05 \times 02$	3	**0**	1	2	2	2	**0**	**0**	2	2
$20 \times 10 \times 01$	1	**0**	1	1	3	1	**0**	1	3	3
$20 \times 10 \times 02$	**0**	**0**	**0**	**0**	2	1	**0**	**0**	2	3
$20 \times 20 \times 01$	4	**0**	**0**	**0**	3	4	**0**	**0**	**0**	3
$50 \times 05 \times 01$	2	**0**	**0**	3	3	1	**0**	**0**	3	3
$50 \times 10 \times 01$	3	**0**	1	1	3	3	**0**	1	2	3
$50 \times 20 \times 01$	4	**0**	**0**	2	1	4	**0**	**0**	2	2
	$\alpha = 0.20$									
$20 \times 05 \times 01$	4	**0**	**0**	**0**	1	4	**0**	**0**	**0**	2
$20 \times 05 \times 02$	4	**0**	**0**	**0**	3	4	**0**	**0**	**0**	3
$20 \times 10 \times 01$	4	**0**	**0**	3	2	3	**0**	**0**	3	2
$20 \times 10 \times 02$	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
$20 \times 20 \times 01$	4	1	**0**	2	**0**	4	1	**0**	1	**0**
$50 \times 05 \times 01$	4	**0**	1	2	3	3	**0**	1	2	3
$50 \times 10 \times 01$	4	**0**	**0**	3	**0**	4	**0**	**0**	3	2
$50 \times 20 \times 01$	4	1	**0**	3	1	4	1	**0**	1	1

Table 5: Algorithm comparison according to the worst-case indicator values.

	$I^-_{\Omega,H}$					$I^1_{\Omega,\epsilon+}$				
	z^1	z^{worst}		I^{worst}		z^1	z^{worst}		I^{worst}	
		10	20	10	20		10	20	10	20
	$\alpha = 0.10$									
$20 \times 05 \times 01$	4	**0**	2	**0**	**0**	4	**0**	**0**	**0**	**0**
$20 \times 05 \times 02$	3	2	2	**0**	**0**	2	2	1	**0**	**0**
$20 \times 10 \times 01$	3	**0**	1	2	2	1	**0**	**0**	2	2
$20 \times 10 \times 02$	2	2	3	**0**	**0**	3	**0**	2	**0**	**0**
$20 \times 20 \times 01$	4	1	1	**0**	**0**	4	**0**	**0**	**0**	**0**
$50 \times 05 \times 01$	1	**0**	2	**0**	2	**0**	**0**	**0**	1	3
$50 \times 10 \times 01$	4	1	2	**0**	1	4	**0**	**0**	**0**	2
$50 \times 20 \times 01$	4	2	2	**0**	**0**	4	1	1	1	**0**
	$\alpha = 0.20$									
$20 \times 05 \times 01$	4	3	1	**0**	**0**	4	2	2	**0**	**0**
$20 \times 05 \times 02$	4	2	2	**0**	1	4	2	2	**0**	**0**
$20 \times 10 \times 01$	3	3	1	1	**0**	2	2	1	2	**0**
$20 \times 10 \times 02$	2	2	2	**0**	**0**	2	2	2	**0**	**0**
$20 \times 20 \times 01$	2	2	2	**0**	**0**	2	2	2	1	**0**
$50 \times 05 \times 01$	3	2	2	**0**	1	2	2	2	**0**	**0**
$50 \times 10 \times 01$	3	2	2	**0**	1	4	2	2	**0**	**0**
$50 \times 20 \times 01$	4	2	2	**0**	**0**	3	2	2	**0**	**0**

Table 6: Algorithm comparison according to the average-case indicator values.

	$I^-_{\Omega,H}$					$I^1_{\Omega,\epsilon+}$				
	z^1	z^{worst}		I^{worst}		z^1	z^{worst}		I^{worst}	
		10	20	10	20		10	20	10	20
	$\alpha = 0.10$									
$20 \times 05 \times 01$	2	**0**	2	**0**	2	4	**0**	2	**0**	2
$20 \times 05 \times 02$	4	**0**	**0**	**0**	1	4	**0**	**0**	2	**0**
$20 \times 10 \times 01$	2	**0**	1	2	2	2	**0**	**0**	3	2
$20 \times 10 \times 02$	2	**0**	**0**	**0**	2	1	**0**	**0**	1	2
$20 \times 20 \times 01$	4	**0**	**0**	**0**	**0**	4	**0**	**0**	**0**	2
$50 \times 05 \times 01$	2	**0**	1	2	3	2	**0**	**0**	3	3
$50 \times 10 \times 01$	4	**0**	1	1	3	3	**0**	1	2	3
$50 \times 20 \times 01$	4	1	**0**	3	1	4	1	**0**	3	2
	$\alpha = 0.20$									
$20 \times 05 \times 01$	4	**0**	**0**	**0**	**0**	4	**0**	**0**	**0**	**0**
$20 \times 05 \times 02$	4	2	1	**0**	**0**	4	2	1	**0**	**0**
$20 \times 10 \times 01$	4	2	**0**	3	1	3	1	**0**	3	1
$20 \times 10 \times 02$	4	1	2	**0**	**0**	4	1	2	**0**	**0**
$20 \times 20 \times 01$	4	3	1	2	**0**	4	2	1	2	**0**
$50 \times 05 \times 01$	4	**0**	1	1	1	4	**0**	1	2	2
$50 \times 10 \times 01$	4	1	1	**0**	**0**	4	1	**0**	**0**	**0**
$50 \times 20 \times 01$	4	3	1	1	**0**	4	3	1	1	**0**

tions, especially in scheduling, involve some sort of uncertainty, and multiple objective functions. This uncertainty may come from many different sources, such as the objective function(s), the decision variables or the environmental parameters. We formulated a bi-objective flowshop scheduling problem with uncertain processing times. Even if uncertainty-handling optimization is reasonably studied in the single-objective case, this research area is very limited in multi-objective optimization. However, with a reasonable effort, evolutionary algorithms can be adapted to such problems. When dealing with uncertainty, we show that the basic concept often results on the comparison of sample sets, from the decision space to the objective space. Then, we proposed new methodological approaches, based on evolutionary computation, for multi-objective optimization under uncertainty. They are based on an additional level to be defined within an indicator-based fitness assignment. Some of them are based on the aggregation of the objective vector sample, other are applied at a more fine-grained level, through the pairwise comparison of solutions by means of a binary quality indicator applying to objective vectors. We discussed the issue of performance assessment in such context. Then, we applied and experimented our evolutionary algorithms to the multi-objective scheduling problem under uncertainty investigated in the paper. Our results show that taking uncertainty into account is a crucial issue to obtain good-quality solutions subject to uncertain environmental parameters. Moreover, we show that the sample size (number of scenarios) considered during the resolution process has a large impact on the performance of the algorithms. In particular, the higher the deviations over processing times, the larger the number of scenarios to be taken into account. At last, objective vector level approaches appear to slightly outperform indicator level approaches for small noise over processing times, while the latter outperforms the former for a higher noise level.

In future works, we plan to study more deeply the impact of the remaining parameters over the performance of our algorithms. Moreover, many issues remain open with regard to the modeling, the resolution and the performance assessment of multi-objective optimization in uncertain environments. In particular, during our experiments, we found that taking uncertainty into account proved to be very expensive in terms of computational time, due to the high number of evaluations required *per* solution, and to the increase on the cardinality of sets to be compared during the search process. Therefore, we believe that parallel computing has a major role to play in solving such problems, especially for real-world applications where the evaluation step

often requires significant computational resources.

Acknowledgments. We would like to acknowledge the anonymous reviewers for their valuable comments and suggestions.

References

[1] V. T'Kindt, J.-C. Billaut, Multicriteria Scheduling: Theory, Models and Algorithms, Springer-Verlag, Berlin, Germany, 2002.

[2] J.-C. Billaut, A. Moukrim, E. Sanlaville (Eds.), Flexibility and robustness in scheduling, Control systems, robotics and manufacturing series, ISTE-WILEY, 2008.

[3] C. A. Coello Coello, G. B. Lamont, D. A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems (second edition), Genetic and Evolutionary Computation Series, Springer, New York, USA, 2007.

[4] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, Chichester, UK, 2001.

[5] Y. Jin, J. Branke, Evolutionary optimization in uncertain environments - a survey, IEEE Transactions on Evolutionary Computation 9 (2005) 303–317.

[6] E. Zitzler, S. Künzli, Indicator-based selection in multiobjective search, in: Conference on Parallel Problem Solving from Nature (PPSN VIII), Vol. 3242 of Lecture Notes in Computer Science, Springer-Verlag, Birmingham, UK, 2004, pp. 832–842.

[7] M. Basseur, A. Liefooghe, K. Le, E. Burke, The efficiency of indicator-based local search for multi-objective combinatorial optimisation problems, Journal of Heuristics(to appear). doi:10.1007/s10732-011-9178-y.

[8] M. Basseur, E. Zitzler, Handling uncertainty in indicator-based multi-objective optimization, International Journal of Computational Intelligence Research 2 (3) (2006) 255–272.

[9] A. Liefooghe, M. Basseur, L. Jourdan, E.-G. Talbi, Combinatorial optimization of stochastic multi-objective problems: an application to the flow-shop scheduling problem, in: Evolutionary Multi-criterion Optimization (EMO 2007), Vol. 4403 of Lecture Notes in Computer Science, Springer-Verlag, Matsushima, Japan, 2007, pp. 457–471.

[10] R. A. Dudek, S. S. Panwalkar, M. L. Smith, The lessons of flowshop scheduling research, Operations Research 40 (1) (1992) 7–13.

[11] A. Nagar, J. Haddock, S. Heragu, Multiple and bicriteria scheduling: A literature survey, European Journal of Operational Research 81 (1995) 88–104.

[12] J. D. Landa Silva, E. Burke, S. Petrovic, An introduction to multiobjective metaheuristics for scheduling and timetabling, in: Metaheuristics for Multiobjective Optimisation, Vol. 535 of Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, Berlin, Germany, 2004, pp. 91–129.

[13] J. K. Lenstra, A. H. G. Rinnooy Kan, P. Brucker, Complexity of machine scheduling problems, Annals of Discrete Mathematics 1 (1977) 343–362.

[14] J. Du, J. Y.-T. Leung, Minimizing total tardiness on one machine is NP-hard, Mathematics of Operations Research 15 (1990) 483–495.

[15] R. L. Graham, E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: A survey, Annals of Discrete Mathematics 5 (1979) 287–326.

[16] M. Gourgand, N. Grangeon, S. Norre, Metaheuristics and performance evaluation models for the stochastic permutation flow-shop scheduling problem, in: Flexibility and Robustness in Scheduling, ISTE-WILEY, 2008, Ch. 5, pp. 143–170.

[17] P. Kouvelis, R. L. Daniels, G. Vairaktarakis, Robust scheduling of a two-machine flow shop with uncertain processing times, IIE Transactions 32 (5) (2000) 421–432.

[18] A. A. Cunningham, S. K. Dutta, Scheduling jobs with exponentially distributed processing times on two machines of a flow shop, Naval Research Logistics Quarterly 16 (1973) 69–81.

[19] P. S. Ku, S. C. Niu, On johnson's two-machine flow shop with random processing times, Operations Research 34 (1986) 130–136.

[20] L. Wang, L. Zhang, D.-Z. Zheng, A class of hypothesis-test based genetic algorithm for flow shop scheduling with stochastic processing time, The International Journal of Advanced Manufacturing Technology 25 (11-12) (2005) 1157–1163.

[21] M. J. Maddox, J. R. Birge, Using second moment information in stochastic scheduling, in: Recent advances in control and optimization of manufacturing systems, Vol. 214 of Lecture Notes in Control and Information Sciences, Springer, London, UK, 1996, pp. 99–120.

[22] S. Dauzère-Pérès, P. Castagliola, C. Lahlou, Service level in scheduling, in: Flexibility and Robustness in Scheduling, ISTE-WILEY, 2008, Ch. 7, pp. 99–121.

[23] E. Zitzler, L. Thiele, M. Laumanns, C. M. Foneseca, V. Grunert da Fonseca, Performance assessment of multiobjective optimizers: An analysis and review, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 117–132.

[24] Y. Jin, B. Sendhoff, Trade-off between optimality and robustness: An evolutionary multiobjective approach, in: Second International Conference on Evolutionary Multi-Criterion Optimization (EMO 2003), Vol. 2632 of Lecture Notes in Computer Science, Springer-Verlag, Faro, Portugal, 2003, pp. 237–251.

[25] C. K. Goh, K. C. Tan, Evolving the tradeoffs between Pareto-optimality and robustness in multi-objective evolutionary algorithms, in: Evolutionary Computation in Dynamic and Uncertain Environments, Vol. 51 of Studies in Computational Intelligence, Springer-Verlag, 2007, Ch. 20, pp. 457–478.

[26] K. Deb, H. Gupta, Introducing robustness in multi-objective optimization, Evolutionary Computation 14 (4) (2006) 463–494.

[27] J. Teich, Pareto-front exploration with uncertain objectives, in: First International Conference on Evolutionary Multi-criterion Optimization (EMO 2001), Vol. 1993 of Lecture Notes in Computer Science, Springer-Verlag, London, UK, 2001, pp. 314–328.

[28] E. J. Hughes, Evolutionary multi-objective ranking with uncertainty and noise, in: First International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001), Vol. 1993 of Lecture Notes in Computer Science, Springer-Verlag, London, UK, 2001, pp. 329–343.

[29] M. Babbar, A. Lakshmikantha, D. E. Goldberg, A modified NSGA-II to solve noisy multiobjective problems, in: Genetic and Evolutionary Computation Conference (GECCO 2003), Vol. 2723 of Lecture Notes in Computer Science, Springer-Verlag, Chicago, Illinois, USA, 2003, pp. 21–27.

[30] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197.

[31] C. Barrico, C. H. Antunes, An evolutionary approach for assessing the degree of robustness of solutions to multi-objective models, in: Evolutionary Computation in Dynamic and Uncertain Environments, Vol. 51 of Studies in Computational Intelligence, Springer-Verlag, 2007, Ch. 25, pp. 565–582.

[32] C. K. Goh, K. C. Tan, An investigation on noisy environments in evolutionary multiobjective optimization, IEEE Transactions on Evolutionary Computation 11 (3) (2007) 354–381.

[33] P. Kouvelis, G. Yu, Robust discrete optimization and its applications, Kluwer Academic, 1997.

[34] J. Knowles, L. Thiele, E. Zitzler, A tutorial on the performance assessment of stochastic multiobjective optimizers, TIK Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Zurich, Switzerland, (revised version) (2006).

[35] S. Bleuler, M. Laumanns, L. Thiele, E. Zitzler, PISA — a platform and programming language independent interface for search algorithms, in: Second International Conference on Evolutionary Multi-Criterion Optimization (EMO 2003), Vol. 2632 of Lecture Notes in Computer Science, Springer-Verlag, Faro, Portugal, 2003, pp. 494–508.

[36] K. C. Tan, C. Y. Cheong, C. K. Goh, Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation, European Journal of Operational Research 177 (2) (2007) 813–839.

[37] H. Eskandari, C. Geiger, Evolutionary multiobjective optimization in noisy problem environments, Journal of Heuristics 15 (16) (2009) 559–595.

[38] J. E. Fieldsend, R. M. Everson, Multi-objective optimisation in the presence of uncertainty, in: IEEE Congress on Evolutionary Computation (CEC 2005), IEEE Press, Edinburgh, UK, 2005, pp. 243–250.

[39] C. K. Goh, K. C. Tan, C. Y. Cheong, Y. S. Ong, Noise-induced features in robust multi-objective optimization problems, in: IEEE Congress on Evolutionary Computation (CEC 2007), IEEE Press, Singapore, 2007, pp. 568–575.

[40] E. D. Taillard, Benchmarks for basic scheduling problems, European Journal of Operational Research 64 (1993) 278–285.

[41] A. Liefooghe, L. Jourdan, E.-G. Talbi, A software framework based on a conceptual unified model for evolutionary multiobjective optimization: ParadisEO-MOEO, European Journal of Operational Research 209 (2) (2011) 104–112.

[42] H. Ishibuchi, 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 (3) (1998) 392–403.