J Intell Manuf
DOI 10.1007/s10845-014-1026-0

Effective multiobjective EDA for bi-criteria stochastic job-shop scheduling problem

Xinchang Hao · Mitsuo Gen · Lin Lin · Gursel A. Suer

Received: 2 November 2014 / Accepted: 15 December 2014

Abstract This paper proposes an effective multiobjective estimation of distribution algorithm (MoEDA) which solves the bi-criteria stochastic job-shop scheduling problem with the uncertainty of processing time. The MoEDA proposal minimizes the expected average makespan and the expected total tardiness within a reasonable amount of computational time. With the framework of proposed MoEDA, the probability model of the operation sequence is estimated firstly. For sampling the processing time of each operation with the Monte Carlo methods, allocation method is used to decide the operation sequence, and then the expected makespan and total tardiness of each sampling are evaluated. Subsequently, updating mechanism of the probability models is proposed according to the best solutions to obtain. Finally, for comparing with some existing algorithms by numerical experiments on the benchmark problems, we demonstrate the proposed effective estimation of distribution algorithm can obtain an acceptable solution in the aspects of schedule quality and computational efficiency.

X. Hao
Waseda University, Kitakyushu, Japan
e-mail: haoxc@ruri.waseda.jp

M. Gen (✉)
Tokyo University of Science, Tokyo, Japan
e-mail: gen@flsi.or.jp

M. Gen · L. Lin
Fuzzy Logic Systems Institute, Kitakyushu, Japan

L. Lin
Dalian University of Technology, Dalian, China
e-mail: lin@dlut.edu.cn

G. A. Suer
Ohio University, Athens, OH, USA
e-mail: suer@ohio.edu

Keywords Estimation of distribution algorithm (EDA) · Multiobjective optimization model · Manufacturing scheduling · Stochastic job-shop scheduling problem (S-JSP)

Introduction

Over the past 60 years, a great number of researches have been conducted on job-shop scheduling problem (JSP), which is one branch of the scheduling problem and highly popular in the manufacturing industry. JSP is one of the famous combinatorial optimization problems as NP-hard under the precedence and resource constraints (Lawler et al. 1993; Gen and Cheng 1997). For conventional JSP, there is often making the assumptions in traditional machine scheduling theory is that all time parameters are known exactly and in deterministic values. However, there are often uncertainties in manufacturing systems. These uncertainties may be caused by a number of possible sources described as the following: the processing time of the operation may be more or less time than originally estimated. Moreover, the resources may become unavailable. Due dates may have to be changed. New orders may have to be incorporated, etc (Bianchi et al. 2009; Ouelhadj and Petrovic 2009). For modeling uncertainness in the scheduling problem, the effective techniques such as stochastic programming, fuzzy programming, rough sets, grey programming and interval theory are carefully surveyed in Gu et al.'s work (Gu et al. 2009). Notably, the stochastic programming is one technique in the filed of mathematical optimization where the parameters are initially described in terms of probability distributions. It has extensive applications in a range of areas ranging from finance to energy optimization, and the problem in scheduling field is named as the stochastic scheduling (Kall and Wallace 1994).

Published online: 06 January 2015

In real-world problem, most of the JSPs is the stochastic scheduling problems. As one of the newest issues, more and more attention is spent on the problem with random processing time. As a result, in the last several decades, a significant amount of results have been achieved on stochastic job-shop scheduling problem (S-JSP). Meanwhile, there are some novel intelligent evolutionary computation methods are carried out.

Intelligent manufacturing scheduling based on evolutionary algorithms (EAs) in meta-heuristics such as genetic algorithm (GA), simulated annealing (SA), ant colony optimization (ACO) and particle swarm optimization (PSO), have become some of the common tools for finding satisfactory solutions (Gen et al. 2009; Gen and Lin 2014). Recently, there are growing interests in stochastic optimization methods called estimation of distribution algorithms (EDA) that build and sample explicit probabilistic model for the distribution of promising candidate solutions found so far and use the constructed model to guide further search behavior (Larrañaga and Lozano 2002). Therefore, it is uncomplicated to apply the machine learning techniques to improve the performance of EDA. Particular for the linkage problems in which where discrete stochastic variables have interaction behavior, EDA is known as an effective method for solving stochastic optimization problems.

In this paper, we propose an effective multiobjective estimation of distribution algorithm (MoEDA) for solving the bi-criteria stochastic job shop scheduling problem (BS-JSP) with the uncertainty of processing time. The proposal is to minimize the expected average makespan and expected total tardiness within a reasonable amount of calculation time. With the framework of estimation of distribution algorithm, the probability model of the operation sequence is formulated. By sampling the processing time of each operation with the Monte Carlo methods, we use allocation method to decide the operation sequence, and then the expected makespan and expected total tardiness of each sampling are calculated.

The remainder of this paper is organized as follows: "Literature review" section provides a review of the S-JSP and Fuzzy JSP models; "Bicriteria stochastic job-shop scheduling problem" section presents a mathematical programming model for the BS-JSP; "Effective multiobjectives estimation of distribution algorithm" section proposes effective MoEDA approach in the detail; "Experiments and discussion" section provides experimental comparisons that apply the MoEDA approach for analyzing and solving several BS-JSP and finally, "Conclusion" section concludes this study with a discussion of the finding and future research directions.

Literature review

Machine scheduling problems arise in diverse areas such as flexible manufacturing system, production planning, computer design, logistics, communication, etc. A common feature of many of these problems is that no efficient solution algorithm is known yet for solving it to optimality in polynomial time. The classical JSP is one of the most well-known machine scheduling problems (Gen et al. 1994). Informally, the problem can be described as follows: there is a set of jobs and a set of machines. Each job requires a number of operations, each of which needs to be processed during an uninterrupted processing time of a given length on a given machine. Each machine can process at most one operation at a time. A schedule is an allocation of the operations to time intervals on the machines. The problem is to find a schedule of minimum length.

JSP is one of the hardest combinatorial optimization problems. Because of its inherent complexity, heuristic procedures are an attractive alternative. Most conventional heuristic procedures use a priority rule, i.e., a rule for choosing an operation from a specified subset of as yet unscheduled operations. In last decades years, an interest in using probabilistic search methods to solve JSP has been growing rapidly, such as SA, tabu search (TS), and genetic algorithms (GA) as surveyed by Cheng et al. (1996, 1999).

However, in most real-world environment, there are some uncertain variations or disruptions on some features (machine breakdown, variation of the processing time and dual date) of unforeseen shop floor. One understandable approach is to eliminate the cause of disruptions after the scheduling is released to the floor. On the occurrence of a disruption, a given policy is developed to adjust the schedule to minimize penalty cost causing by disruption. These approaches presented in research work, Raheja and Subramaniam (2002) and Subramania et al. (2005), are named reactive scheduling. Recently, another classification of methods called robust scheduling attracts significant attention. A robust schedule is established according to given robust measure while the disruptions are taken into account. For the case, the right-shift policy is used to delay the unfinished jobs as necessary to accommodate the disruption.

We can review some recent articles to classify these studies as shown in Table 1. From Table 1, various researchers considered different methodology classification for solving the problem that we can be divided into two parts that are a combined or hybrid metaheuristics method in job-shop scheduling problems and fuzzy problems during last decades. The hybrid meta-heuristics method helps performing to find the solution in job-shop scheduling problems such as Tavakkoli-Moghaddam et al. (2005) proposed a hybrid method using a neural network approach and a simulated annealing algorithm in 2 stages, in order to produce the optimal/near-optimal solution. Liu et al. (2005) proposed an approach named PSOSAHT, which is hybrid PSO for flow shop scheduling with SA and hypothesis test (HT) for stochastic flow shop scheduling with uncertain processing

Table 1 The published journal paper for solving JSP under uncertainty

Uncertainty	Modelling technique	Objectives	Authors	Methodology
Processing time and machine breakdown	Stochastic	Total weighted tardiness	Kutanoglu and Sabuncuoglu (2001)	Simulation methods
Processing time	Stochastic	Penalty costs	Golenko-Ginzburg and Gonik (2002)	Decision-making rules
Processing time	Stochastic	Makespan	Yoshitomi and Yamaguchi (2003)	Genetic algorithm and Monte Carlo method
Processing time	Stochastic	Makespan	Tavakkoli-Moghaddam et al. (2005)	Co-evolutionary quantum genetic algorithm
Processing time	Stochastic	Makespan	Liu et al. (2005)	Hybrid particle swarm optimization
Processing time	Stochastic	Expected makespan and expected total tardiness	Lei and Xiong (2007)	Multiobjective genetic algorithm
Processing time and dual date	Fuzzy rule	Expected makespan and expected total tardiness	Lei (2008)	Multiobjective particle swarm optimization
Lot-size and priority of the job	Fuzzy sets	Average weighted tardiness	Petrovic et al. (2008)	Multiobjective genetic algorithm
Machine unavailability and breakdown	Stochastic	Average makespan	Hasan et al. (2011)	Hybrid genetic algorithm
Processing time	Stochastic	Makespan	Azadeh et al. (2011)	Simulation-artificial neural network algorithm
Processing time	Stochastic	Expected total weighted tardiness	Zhang et al. (2013)	Hybrid particle swarm optimization
Processing time	Stochastic	Expected sum of earliness and tardiness	Yang et al. (2014)	Hybrid evolutionary strategy

time. Zhou et al. (2009) proposed an ACO with different levels of machine utilizations, processing time distributions, and performance measures. Gu et al. (2010) proposed a novel parallel quantum genetic algorithm (NPQGA) for the S-JSP with the objective of minimizing the expected value of makespan. Zhang and Wu (2011) presented an artificial bee colony algorithm for the job shop scheduling problem with random processing times with the objective of minimizing the maximum lateness.

Gholami and Zandieh (2009) integrated simulation into genetic algorithm to the dynamic scheduling of a flexible job-shop with the objectives of minimizing expected makespan and mean tardiness. Lei (2012) developed an efficient genetic algorithm to solve the problem of scheduling stochastic job shop subject to breakdown with minimizing the makespan. Horng et al. (2012) proposed an evolutionary algorithm ESOO as embedding evolutionary strategy (ES) in ordinal optimization (OO) to solve for a good enough schedule with the objective of minimizing the expected sum of storage expenses and tardiness penalties. Zhang et al. (2012) proposed a two-stage PSO algorithm for S-JSP with the objective of minimizing the expected total weighted tardiness. In the first-stage PSO, a performance estimate is used for quick evaluation of the solutions, and a local search procedure is embedded for accelerating the convergence to promising regions in the solution space. The second-stage PSO continues the search process, but applies a more accurate solution evaluation policy. Recently, Yang et al. (2014) developed hybrid evolutionary strategy in ordinal optimization (ESOO) which is embedded optimal computing budget allocation (OCBA) technique into the exploration stage of ESOO to optimize the performance evaluation process by controlling the allocation of simulation times. Lin et al. (2012) proposed a network modeling and evolutionary optimization for scheduling in manufacturing system. Also, Gen and Lin (2014) surveyed the recent multiobjective evolutionary algorithms for solving manufacturing scheduling problems.

In this paper, we extend the approach proposed in the reference (Hao et al. 2013a, b) to improve performance in terms of convergent speed and optimization quality for solving S-JSP with probabilistic processing time to minimize the expected makespan and total tardiness lacking classification of them in Table 1. This approach will describe in detail in "Effective multiobjectives estimation of distribution algorithm" section.

Bicriteria stochastic job-shop scheduling problem

In order to solve a job-shop scheduling problem in stochastic and static environments, it assumes that the probability

distribution of the processing time is known in advance. The realized outcome of a random processing time of operation only gets to be known at the completion of the processing. In this paper, we use a pure integer programming model to transmute the processing times in terms of stochastic variable. The S-JSP can be formulated as an extended version of JSP. Difference to the conventional JSP, Each operation o_{ij} is carried out under uncertain random disturbance with pregiven expected valued E[p_{ij}] and variance v_{ij} where p_{ij} is the processing time of o_{ij} on the machines. The distribution of the variance can be predicted from the experimental data such as normal distribution, uniform distribution and exponential distribution etc. Therefore, the processing time uncertainty can be constructed through the concept scenario ξ which corresponds to an assignment of reasonable processing time on o_{ij}.

The stochastic expected value model of bi-criteria S-JSP may be formulated as follows: the makespan is the maximum completion time of jobs and objective is to find a schedule that minimizes the expected value of makespan C_{max}, the Eq. (1) and the expected total tardiness, the Eq. (2) conflicting each other.

Indices:

i, k	the index of jobs; $i, k = 1 \ldots J$
j, h	the index of operations; $j, h = 1 \ldots N$
m	the index of machines; $m = 1 \ldots M$

Parameters:

J	the number of jobs
N	the number of operations
M	the number of machines
o_{ij}	the operation j of the job i
C_i	the completion-time of the job i
d_i	the due-date of the job i
ξp_{ij}	the random processing time of operation o_{ij} under scenario ξ.
L_i	$\max\{0, C_i - d_i\}$, the tardiness of job i.
w_i	the tardiness level (weight) of the job i
$\mathcal{N}$	a positive big number enough as a penalty factor.

Decision variables:

$x_{ikm} = 1$	if job i precedes job k on machine m; 0 otherwise.
$y_{ijm} = 1$	if it is available to process operation o_{ij} on machine m; 0 otherwise.
ξs_{ijm}	the starting time operation o_{ij} on machine m, a stochastic variable
ξc_{ijm}	the completion time operation o_{ij} on machine m, a stochastic variable

Bicriteria stochastic job-shop scheduling problem can be formulated as a nonlinear mixed integer programming model as follows:

$$\min \quad E[C_{max}] = E\left[\max_{i=1,\ldots,J}\left\{\max_{j=1,\ldots,N}\left\{\max_{m=1,\ldots,M}\xi c_{ijm}\right\}\right\}\right] \tag{1}$$

$$\min \quad ET_{tot} = \sum_{i=1}^{J} w_i E\left[max\{0, L_i\}\right] \tag{2}$$

$$\text{s. t.} \quad \sum_{m=1}^{M} y_{ijm} \leq 1, \forall i, j \tag{3}$$

$$\sum_{m=1}^{M} x_{ikm} y_{ijm} = 1, \quad \forall i, j, k \tag{4}$$

$$\sum_{m=1}^{M} \xi s_{ijm} + \xi p_{ij} y_{ijm} \leq \sum_{m=1}^{M} \xi s_{i(j+1)m}, \forall i, j \tag{5}$$

$$\sum_{i=1}^{J}\sum_{j=1}^{N}\left(\xi s_{ijm} + \xi p_{ijm} y_{ijm}\right) \leq \sum_{i=1}^{J}\sum_{k=1}^{J}\sum_{h=1}^{N}\left(\xi s_{khm} x_{ikm} + \mathcal{N}(1 - x_{ikm})\right), \forall m \tag{6}$$

$$\xi c_{khm} \geq 0, \xi s_{khm} \geq 0, x_{ikm}, y_{ijm} \in [0, 1] \ \forall i, j, k, h, m \tag{7}$$

where the Eq. (3) shows that only one operation can be in each sequence on a machine. The Eq. (4) guarantees that each operation for each job must be allocated to just one machine in a sequence. The Eq. (5) guarantees the operation precedence sequences for each job. The Eq. (6) shows that the processing time of each operation does not have any overlap with any other. The Eq. (7) represents the nonnegative restrictions.

Effective multiobjectives estimation of distribution algorithm

The multiple objective optimization problems have been receiving growing interest from researchers with various backgrounds since early 1960. There are a number of scholars who have made significant contributions to the problem. Among the studies, Pareto is perhaps one of the most recognized pioneers in this field. Recently, EAs have been received considerable attention as a novel approach to multiobjective optimization problems, resulting in a fresh body of research and applications known as evolutionary multiobjective optimization (Gen and Lin 2014). The inherent characteristics of EAs demonstrate why evolutionary search is possibly well suited to the multiple objective optimization problems. The basic feature of EAs is the multiple directional and global

Hybrid moEDA routine

Input:
X: $X = (X_1, X_2, \ldots, X_n)$, vector of the decision variables.
$popSize$: number of the population solutions kept by moEDA.
gen : number of the generations (iterations).
$prRate$: rate of the promising solutions kept by moEDA.
$elimRate$: elimination rate of the keeping promising solutions.
Output:
S_{best}: the best Pareto solution.

begin
  Initialization:
    Step 1: $t \leftarrow 0$.
    Step 2: Initialize the population $Pop(0)$ with the size $popSize$ solutions randomly.
    Step 3:Calculate objectives $f_k(Pop), k = 1, \ldots, q$ by decoding routine.
    Step 4 Create Pareto optimal solutions $E(Pop)$ by nondominated routine.
    Step 5 Calculate fitness $eval(Pop)$ by HSS-EA routine and keep best Pareto solution S_{best}.
  Optimization:
  while *terminating criteria are not met* **do**
    Step 6: Select the promising data set D from $Pop(t)$ consisting of $popSize \times prRate$ solutions.
    Step 7: Estimate the probabilities $p(x_i)$ for each variable X_i over the set D, and sample $candidates$ based on the $p(x_i)$.
    Step 8: Perform a problem-specific local search $DoLocalSearch(candinates)$.
    Step 9:Calculate objectives $f_k(Pop), k = 1, \ldots, q$ by decoding routine.
    Step 10:update Pareto optimal solutions $E(Pop)$ by nondominated routine.
    Step 11 Calculate fitness $eval(Pop)$ by HSS-EA routine and keep best Pareto solution S_{best}.
    Step 12: Replace $popSize \times elimRate$ worse items of the keeping population solutions with the best solutions $candidates$.
    Step 13: $t \leftarrow t + 1$.
  end
end

Fig. 1 Pseudo-code for h-MoEDA

searches by maintaining a population of potential solutions from generation to generation. The population-to-population approach is hopeful to explore all Pareto solutions.

In the paper, we present a scheme of hybrid multiobjective estimation of distribution algorithm (h-MoEDA). In h-MoEDA, MoEDA takes care of exploration that tries to identify the most promising search space regions, and modeling the distribution of the stochastic variables. For NP-hard combinatorial optimization problems, it became necessary that the global search dynamics of EDAs ought to be complemented with local search refinement (Michiels et al. 2010). Variable neighborhood search (VNS) is proposed to improve the solution quality, and the simple principle for driving this improvement is based on the systematic change of neighborhood within a possibly randomized local search. Figure 1 presents the scheme of h-MoEDA. In Fig. 1, h-MoEDA starts by generating the solutions kept in the population randomly. The set of solutions of high fitness, which refers to promising solutions, is selected from the population using a selection method, and the promising solutions are used to learn the probability model. Thereafter, probabilities defined by the probability model are estimated, and the new candidate solutions are sampled according to the given sampling method. For a candidate solution, problem specific local search algorithm is used to improve each candidate solution to reach a local optimum. Finally, the new solutions are evaluated, and these with high fitness are incorporated into a solution pool, which keeps these individuals contributing to the makeup of promising solutions. The iteration will continue until the predefined termination criteria are met.

The following subparagraph presents the vital components of the proposed h-MoEDA in detail. Firstly, the representation of a chromosome for an individual is described, and transition probability model is presented. Then, fitness assignment mechanisms are discussed. Finally, local search algorithm of h-MoEDA is summarized.

Solution representation

The representation of the operation sequence uses job-based encoding in Gen and Cheng (2000), and the length of the chromosome equals the total number of operations. The job number denotes the operation of each job, and the l-th occurrence of a job number refers to the l-th operation in the

Table 2 Example of a 3-job 3-machine problem

Processing time				Machine sequence			
Jobs	Operations			Jobs	Operations		
	1	2	3		1	2	3
J_1	3	3	2	J_1	M_1	M_2	M_3
J_2	1	5	2	J_2	M_1	M_3	M_2
J_3	3	2	3	J_3	M_2	M_1	M_3

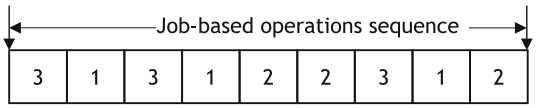

Fig. 2 Illustration of the representation of a solution for S-JSP

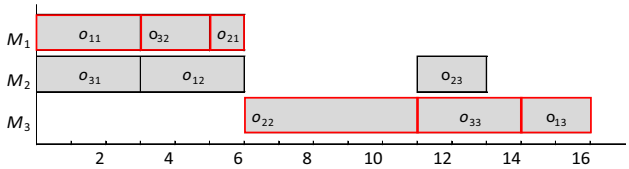

Fig. 3 Gantt chart of the schedule encoded by the operation sequence shown in Fig. 2

sequence of this job. Table 2 presents an example considering with a 3 jobs and 3 machines For a job-based operations sequence vector $v_1 = [3, 1, 3, 1, 2, 2, 3, 1, 2]$ (shown in Fig. 2), the operations sequence can be interrupted as follows: (3, 1), (1, 1), (3, 2), (1, 2), (2, 1), (2, 2), (3, 3), (1, 3), (2, 3). When operation sequences are determined in terms of aforementioned algorithms, Giffler–Thompson based heuristic algorithm (Giffler and Thompson 1960) based decoding procedure is applied to generate a feasible schedule. This produces a timetable with the start and end of the processing period, and the makespan time. The Gantt chart of this solution is illustrated in Fig. 3.

Transition probability

For traditional EAs, the representation of a chromosome for an individual is generated by mapping the decision space into the search space or directly encoding the decision variables. Each position in a probability vector indicates the distribution of probability regarding each variable. When prior knowledge of distribution is not assumed, the domain of discrete variable X is a set of predefined values (x). The distribution of random variable X has the same equal probability; the initialization is as following:

$$P_{t=0}(X) = \frac{1}{|X|} \tag{8}$$

where $|X|$ denotes the number of values in the set of domain X.

After the initialization of EDA, probability sample for new alternative solutions and the new solutions are evaluated according to a specific system objective. EDA collects all new alternative solutions and replaces the inferior solutions in the promising data. The probability distribution of X can be estimated as follows:

$$B_t(X = x) = \frac{N(X = x) + 1/|X|}{prSize + 1/|X|} \tag{9}$$

where $N(X = x)$ denotes the number of instances in promising solutions with variable $X = x$, and represents the low bound to the probability of X.

The distribution probability of X in the probability vector is learned toward the estimated distribution of promising data, as follows:

$$P_{t+1}(X = x) = (1 - \alpha)P_t(X = x) + \alpha B_t(X = x) \tag{10}$$

where α denotes the learning rate from the current promising solutions; in particular, for $\alpha = 1$, the probability distribution is completely reconstructed by the current promising solutions.

To maintain the diversity of sampling, the distribution probability of X is updated toward the estimation distribution. The distribution can be tuned with probability p_{m} of the mutation, and the mutation is performed using the following definition:

$$P_{t+1}(X = x) = \frac{P_t(X = x) + \lambda_m}{\sum_{x' \in X \setminus \{x\}} max\left(P_t(x) - \lambda_m/(|X| - 1), \varepsilon\right) + (P_t(X = x) + \lambda_m)} \tag{11}$$

where λ_m is the mutation shift that controls the amount for mutation operation, and ε is a small probability value to avoid the negative probability value.

For S-JSP, The decision-making space is operation sequence. Therefore, the first probability model is to model the priority (importance) and the likelihood of adjacent operations in operation sequence. Let η_{jl} be the number of times that job i appears before or in position l in the promising data D. It denotes the importance of the order of jobs. μ_{il} is the number of times that job i appears immediately after job i' when job i' is in position $l - 1$. μ_{il} indicates the importance of the similar blocks of jobs in the promising data D. Then, the probability for positioning job i in the l-th position of the offspring for generation t is determined by

$$p_t(i, l) = \beta M(\eta_{il}) + (1 - \beta)M(\mu_{il}) \tag{12}$$

where

$$M(\eta_{il}) = 1 - \exp\left(-\eta_{ij} \Big/ \sum_{i' \in D} \eta_{i'j}\right) \Big/ \left(\sum_{i'' \in D} 1 - \exp\left(-\eta_{i''j} \Big/ \sum_{i' \in D} \eta_{i'j}\right)\right)$$

$$M(\mu_{il}) = 1 - \exp\left(-\mu_{ij} \Big/ \sum_{i' \in D} \mu_{i'j}\right) \Big/ \left(\sum_{i'' \in D} 1 - \exp\left(-\mu_{i''j} \Big/ \sum_{i' \in D} \mu_{i'j}\right)\right)$$

$M(\eta_{il})$ moreover, $M(\mu_{il})$ are completeness measures for importance of the order jobs and similar blocks respectively. β is behavior coefficient, which is used to adjust the preference on which covering percentage we want to have a good discrimination.

Since Giffler–Thompson-based decoding algorithm (described in "Transition probability" section) does not consider the internal idle time, the solution is always not the optimal. For improving the objective, neighbourhood search is employed to improve the objectives In h-MoEDA. The priority (importance) of jobs on each machine is modeled as statistical inference. Similar to η_{jl}, let η_{jlm} be the number of times that job i appears before or in position l on machine m in the promising data D, and μ_{ilm} indicates the importance of the similar blocks of jobs on machine m in the promising data D. Similarly, the probability for positioning job i in the l-th position on machine m of the offspring for generation t is determined by

$$p_t(i, l|m) = \beta M(\eta_{ilm}) + (1 - \beta) M(\mu_{ilm}) \tag{13}$$

$M(\eta_{il\mathrm{m}})$ moreover, $M(\mu_{il\mathrm{m}})$ are completeness measures for importance of the order jobs and similar blocks on machine m, respectively.

Fitness assignment mechanisms

The multiple objective optimization problems have been receiving growing interest from researchers with various backgrounds since early 1960. There are a number of scholars who have made significant contributions to the problem. Among them, Pareto is perhaps one of the most recognized pioneers in this field and recently, EAs have been received considerable attention as a novel approach to multiobjective optimization problems, resulting in a fresh body of research and applications known as evolutionary multiobjective optimization (Gen and Lin 2014). The inherent characteristics of EAs demonstrate why evolutionary search is possibly well suited to the multiple objective optimization problems. The basic feature of EAs is the multiple directional and global searches by maintaining a population of potential solutions from generation to generation. The population-to-population approach is hopeful to explore all Pareto solutions.

Because the multiobjective optimization problems are the natural extensions of constrained and combinatorial optimization problems, so many useful methods based on EAs developed during the past two decades. One of special issues in the multiobjective optimization problems is fitness assignment mechanism. Since the 1980s, several fitness assignment mechanisms have been proposed and applied in multiobjective optimization problems (Gen et al. 2008). Although most fitness assignment mechanisms are just different approach and suitable to different cases of multiobjective optimization problems, in order to understanding the development of multiobjective EAs (MOEAs), we classify algorithms according to proposed years of different approaches (Gen and Lin 2014):

- Vector evaluated genetic algorithm (VEGA; Schaffer 1985)
- Multiobjective genetic algorithm (MOGA; Fonseca and Fleming 1995)
- Non-dominated sorting genetic algorithm (NSGA; Srinivas and Deb 1994)
- Random-weight genetic algorithm (RWGA; Ishibuchi and Murata 1998)
- Adaptive-weight genetic algorithm (AWGA; Gen and Cheng 2000)
- Strength Pareto evolutionary algorithm II (SPEA 2; Zitzler and Thiele 2001)
- Non-dominated sorting genetic algorithm II (NSGA II; Deb et al. 2000)
- Interactive adaptive-weight genetic algorithm (i-AWGA; Gen et al. 2008)
- Hybrid sampling strategy-based EA (HSS-EA; Zhang et al. 2013)

Many multiobjective genetic algorithms differ mainly in the fitness assignment strategy which is known as an important issue in solving multiple objectives optimization problem (Gen et al. 2008). Zhang et al. proposed a hybrid sampling strategy-based evolutionary algorithm (HSS-EA) in which a Pareto dominating and dominated relationship-based fitness function (PDDR-FF) is used to evaluate the individuals. The PDDR-FF of an individual S_i is calculated by the following fitness assignment function (14):

$$\mathrm{eval}(S_i) = q(S_i) + 1/(p(S_i) + 1), \quad i = 1, 2, \ldots, popSize \tag{14}$$

where q(·) is the number of individuals which can dominate the individual S_i. p(·) is the number of individuals which can be dominated by the individual S_i. The PDDR-FF can set the obvious difference values between the nondominated

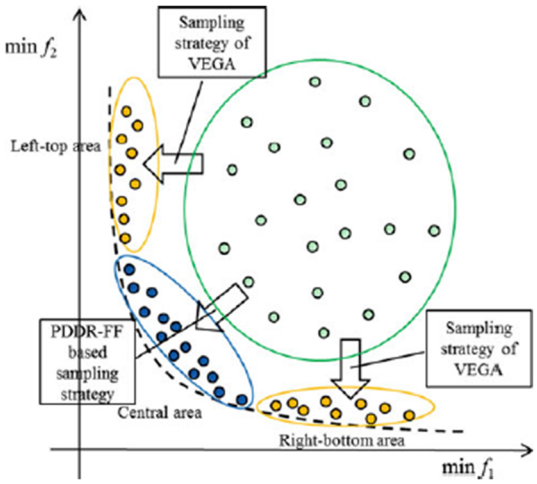

Fig. 4 The description of HSS-EA

and dominated individuals as shown in Fig. 4 (Zhang et al. 2013).

The sampling strategy of VEGA prefers the edge rather than centre regions of Pareto front that it causes VEGA cannot achieve better distribution performance. So it is natural, reasonable and possible to combine PDDR-FF and VEGA methods to improve the overall performance and reduce the computational time of the algorithm proposed.

Local search

In this paper, variable neighborhood search (VNS) is intended to improve the solution quality. The simple principle for driving this improvement is based on the systematic change of neighborhood within a possibly randomized local search (Gao et al. 2008). VNS plays local search method under the framework of the proposed h-MoEDA, and adopts the method that varies the operation in the critical path represented with a disjunctive graph.

(a) *Disjunctive graph* The feasible schedules of S-JSP can be represented with a disjunctive graph. The disjunctive graph is a specific directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A}, \mathcal{E})$, with node set $\mathcal{G}$, ordinary arc set $\mathcal{A}$, and disjunctive arc set $\mathcal{E}$. The nodes of $\mathcal{G}$ correspond to operations, the real arcs $\mathcal{A}$ to immediate precedence relations, and the dashed arc $\mathcal{E}$ to immediate implementation sequence of operations to be performed on the same machine. Figure 3 shows an illustration disjunctive graph and relative Gantt chart of a feasible schedule with three jobs. In Fig. 5, S and T are dummy starting and terminating nodes respectively. The pair of numbers associated to each node represents the assigned machine and the processing time of that operation.

The job predecessor $PJ(r)$ of an operation r is the operation preceding r in the operation sequence of the job to which r belongs. The machine predecessor $PM(r)$ of an operation r is the operation preceding r in the operation sequence on the machine that r is processed on. If node r is a job predecessor of v, then node v is a job successor of r, $SJ(r)$. If node r is a machine predecessor of v, then node v is a machine successor of r, $SM(r)$. In Fig. 5, if r sets o_{12}, the $PJ(r)$ is o_{11}, and the $SJ(r)$ is o_{13}. The machine predecessor $PM(r)$ and successor $SM(r)$ of o_{12} on the machine 2 are o_{31} and o_{23} respectively.

The makespan cannot be reduced any further while maintaining the current critical paths. The mission of local search is to identify and break the existent critical paths one by one in order to get a new schedule with shorter makespan. If an operation r is critical, then at least one of the job predecessor $PJ(r)$ and machine predecessor $PM(r)$ must be critical, if they exist. In this study, if both a job predecessor and a machine predecessor of a critical operation are critical, then choose the job predecessor as the critical path. In order to reduce the computation workload, we consider only a single randomly selected critical path P of disjunctive graph $\mathcal{G}$. For the schedule shown in Fig. 3, the operations that are marked with solid lines in Fig. 3 present one critical path and the critical path consists of the operations sequence $\{o_{12}, o_{32}, o_{21}, o_{22}, o_{33}, o_{13}\}$.

(b) *Variable neighbourhood algorithm:* The idea of the neighbourhood based moving algorithm VNS is straightforward: when the local optimum of moving one critical operation, the solution may be further improved by moving two operations. Simultaneously, at least one operation is critical.

Moving one operation Firstly, moving an operation on the critical path is considered in VNS. The purpose of moving an operation r is to delete the operation from the current position and to insert it at another feasible position of the disjunctive graph $\mathcal{G}$. Operation r is deleted from $\mathcal{G}$ by removing the disjunctive arc from and to r. Then, it is to connect $PM(r)$ to $SM(r)$ with a dashed arc and set the processing time of node r equals to '0'. Let G^- be the disjunctive graph obtained from $\mathcal{G}$ when an operation is removed (the superscript '$-$' refers to the situation after the removal of an operation). Let $C_M(\mathcal{G})$ be the makespan of the solution graph $\mathcal{G}$. Since $\mathcal{G}^-$ is obtained by deleting one operation from $\mathcal{G}$, it is obvious that the makespan of $\mathcal{G}^-$ is no larger than $C_M(\mathcal{G})$. The basic scheme for local search of moving one critical operation is presented in Fig. 6.

Moving two operations When the local optimum of moving one operation is found, there is a critical path P that cannot be broken by moving one of the operations. That is; no assignable time interval can be found for any operation

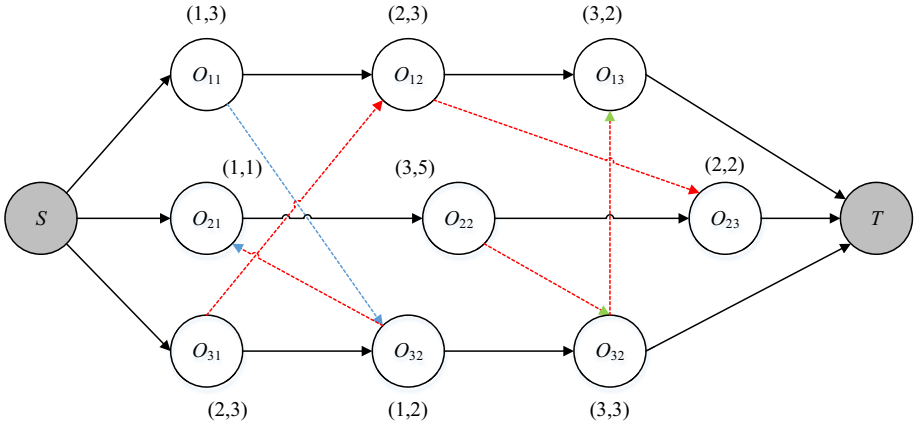

Fig. 5 The disjunctive graph and Gantt chart of a feasible solution shown in Fig. 3

Fig. 6 Pseudo-code for local search of moving one operation

VNS:Moving one operation

Input:
S: the feasible schedule of S-JSP.
begin
  Step L1: Identify a critical path P for a given incumbent solution S.
  Step L2: Set r to be the first operation of P.
  repeat
    Step L3: Delete r from disjunctive graph $\mathcal{G}$ of S to get $\mathcal{G}^-$;
    Step L4: Search for an assignable time that is assignable for r in $\mathcal{G}^-$;
    Step L5: If no assignable interval for r is found, set r to be the next operation of P;
  until (an assignable interval is found, allocate r in the interval. Otherwise, S is a local optimum of moving one operation);
end

$r \in P$ in the disjunctive graph $\mathcal{G}^-$. Deleting an additional operation from $\mathcal{G}^-$ brings more idle time, and may create time interval assignable for r, Fig. 7 shows the basic scheme for VNS based local search of moving two operations.

In the local search of moving two operations, operations r (r is a critical operation) and v are deleted simultaneously from $\mathcal{G}$ to get $\mathcal{G}^-$. We then search for an assignable time interval for r in $\mathcal{G}^-$. When the assignable time interval is found, and r is inserted to get solution $\mathcal{G}^{-*}$, we search for an assignable time interval for v in $\mathcal{G}^{-*}$.

Experiments and discussion

Test problems

To examine the practical viability and efficiency of the proposed MoEDA, we designed a numerical study to compare MoEDA with efficient algorithms from previous studies. The proposed MoEDA was compared with adaptive weight genetic algorithm (awGA) proposed by Gen and Cheng (2000), Non-dominated Sorting Genetic Algorithm II (NSGA-II) by Deb et al. (2002) and Strength Pareto Evolutionary Algorithm 2 (SPEA2) by Zitzler et al. (2001) for a set of simulation data of testing standard benchmark problems.

In order to ensure the fairness of comparison, three large-scale instances LA29 (10×20), LA35 (30×10) and LA38 (15×15) of proposed in Lawrence (1984) are adopted in this paper, where due dates and the tightness level of the due dates are designed according to the design policy of Essafi et al. (2008). According to practical observation for which 20 % of customers are very important, 60 % are of average importance and the remaining 20 % are of less importance, 20 % of jobs are assigned a weight of 4, 60 % of jobs are assigned a weight of 2 and weight of 1 sets to the remainder of jobs.

For the processing time uncertainty, the processing time of operation o_{ij} may equally take any real value from the uniform distribution $U(\underline{t_{ij}}, \overline{t_{ij}})$ where $\underline{t_{ij}}$ and $\overline{t_{ij}}$ are the given lower and upper bounds respectively. The expected processing time t_{ij}^E of an operation o_{ij} is to be equal to processing time of that operation in original instances. The upper and lower processing time bounds of an operation affected by uniform variation in its processing time is calculated by:

$$\left[\underline{t_{ij}}, \overline{t_{ij}}\right] = t_{ij}^E \times \left[1, (1+\gamma)\right] \tag{15}$$

Fig. 7 Pseudo-code for local search of moving two operations

VNS: Moving two operations

Input:
S: the feasible schedule of S-JSP.
begin
  Step L1: Identify a critical path P for a given incumbent solution S.
  Step L2: Set r to be the first operation of P.
  repeat
    repeat
      Step L3: Set v to be an operation in solution S.
      Step L4: Delete r and the v from disjunctive graph $\mathcal{G}$ of S to get $\mathcal{G}^-$.
      Step L5: Search for an interval assignable time for r in $\mathcal{G}^-$.
      Step L6: If the assignable interval for r is found, insert r in the interval to get $\mathcal{G}^{-*}$; Otherwise, go to step 9.
      Step L7: Search for assignable time intervals for v in $\mathcal{G}^{-*}$.
      Step L8: If only an assignable interval for v is found, insert v in the interval to get $\mathcal{G}^*$. Otherwise, select one with respect to probabilistic model estimated in $p_t(i, l|m)$.
      Step L9: If either the assignable interval for r or v is not found, set v to be the next operation.
    until (the assignable intervals for both r and v are found or v is the last operation);
    Step L10: If the assignable interval for either r or v is not found, set r to be the next critical operation of P.
  until (both the assignable intervals for r and v are found; or r is the dummy terminating node);
end

30 percent of operations is under uncertainty and disturbance ratio γ is assigned to 0.3. For measuring the uncertainty, a set of 400 scenarios Ξ is sampled as the input of S-JSP. The due date of the job depend on the expected processing times of its operations, and it is calculated using the following formula $d_i = \left\lfloor f*(\sum_{j=1}^{M} \sum_{\xi \in \Xi} \xi p_{ij} / |\Xi| \right\rfloor$ where f is due date tightness factor. The value of f is considered in the experiment: 1.3.

All of the above algorithms were implemented using JAVA under the Eclipse environment, and the simulation experiments were conducted on Intel Core i5 (2.3 GHz clock) with 4G memory. Data were collated from 30 test runs for each algorithm. In order to compare the performance of these algorithms fairly and under the same environment, the strategies of related algorithms and their respective parameters are presented in Table 3.

Results and discussion

In order to evaluate the performance of a given algorithm for S-JSP, the following two performance measures are considered in this paper. Let S_j be a solution set for each solution method ($j = 1, 2, 3, 4$). S^* is a known set of the Pareto-optimal set.

Coverage C(S_1, S_2) is the percent of the individuals in S_2 who are weakly dominated by S_1. The value $C(S_1, S_2) = 1$ means that all individuals in S_2 are weakly dominated by S_1.

Table 3 The parameters of NSGA-II, SPEA2, AWGA and h-MoEDA for S-JSP

	NSGA-II, SPEA2 and awGA	MoEDA
Iteration	1,000	1,000
Population	200	200
Selection	Roulette	Tournament (k)
Operators	Crossover (P_c)	Sampling
	Mutation (P_m)	Improvement (P_i)
Parameters	$P_m = 0.20$	promisingRate = 0.7
	$P_c = 0.80$	$\alpha = 0.02$ $\beta = 0.02$
	$w_1 = 0.4$, $w_2 = 0.6$	$P_m = 0.4$ k = 2

On the contrary, $C(S_1, S_2) = 0$ denotes that none of the individuals in S_2 is weakly dominated by S_1 (Zitzler and Thiele 1999). The larger $C(S_1, S_2)$ is; the better S_1 outperforms S_2 in C.

Spacing SP(S_j) is the standard deviation of the closest distances of individuals by S_j. Smaller *SP* means better distribution performance. The C is used to verify convergence performance while *SP* is used to check the distribution performance.

(a) *Convergence* Table 4 shows a comparison of coverage by MoEDA, awGA, NSGA-II and SPEA2. Figure 8 presents the distribution range of coverage conducted on LA35. From Table 4 and Fig. 8, it is easy to see that the

Table 4 Comparison of coverage measure by h-MoEDA, AWGA, NSGA-II, SPEA2 and AWGA

Problem	Algorithm (A)	Mean		Improved (C(h-MoEDA, A)-C(A, h-MoEDA)) (%)	SD	
		C(h-MoEDA, A)	C(A, h-MoEDA)		C(h-MoEDA, A)	C(A, h-MoEDA)
LA29	NSGA-II	0.4669	0.3524	11.45	0.0842	0.0709
	SPEA2	0.4965	0.3224	17.41	0.0696	0.0603
	awGA	0.5731	0.2541	31.90	0.0451	0.0553
LA35	NSGA-II	0.6486	0.2824	36.63	0.0777	0.0794
	SPEA2	0.6606	0.2618	39.89	0.0860	0.0631
	awGA	0.6844	0.2171	46.73	0.0672	0.0460
LA38	NSGA-II	0.4543	0.3425	11.19	0.0799	0.0754
	SPEA2	0.5063	0.3412	16.52	0.0696	0.0603
	awGA	0.5625	0.2767	28.58	0.0572	0.0619

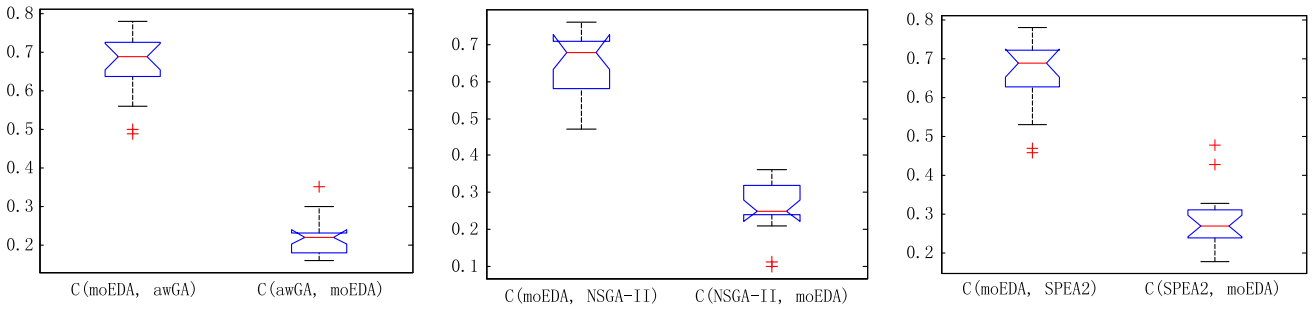

Fig. 8 Box plot of coverage measure conducted on LA35

h-MoEDA is better than MSGA-II, SPEA2 and awWA on C measure. Such better convergence should mainly attribute to the hybrid sampling strategy of VEGA's preference for the edge region of the Pareto front and PDDR-FF's tendency converging toward the center area of the Pareto front. They preserve better performances both in efficacy and efficiency. Especially, h-MoEDA can also keep diversity evenly without special distribution mechanisms like NSGA-II and SPEA2.

(b) *Distribution performance* The experiment results of dispersion performances, *SP* are shown in Table 5 and Fig. 9 presents the distribution range of space conducted on LA35. It indicates that MoEDA is obviously better than awGA, while better than NSGA-II and SPEA2. Without special mechanism to preserve the diversity evenly, h-MoEDA can also achieve satisfactory dispersion performance.

(c) *Computation costs* The computation costs of evolutionary-based algorithms mainly depend on the number of fitness evaluations. We need to compare h-MoEDA with other methods, and the termination criterion is to reach the maximum number of iterations (generations). The mean computation cost of h-MoEDA is 217.78 s, NSGA-II and SPEA2 are 245.54, 390.24 s respectively. Compared with SPEA2, SPEA2 needs to calculate all the distances of each individual to all the other individuals first. Then SPEA2 gets the k-th nearest distance for each individual after sorting its distances and the time complexity of SPEA2 is $O(mN^2 \log N)$ where m is the number of objectives. It is clear that the NSGA-II achieved better computation costs. Without special dispersion preservation mechanism, the time complexity of fitness calculation (PDDR-FF) in h-MoEDA is O(0). It is slightly better than NSGA-II while h-MoEDA needs to estimate the parameters of probability model using the promising solutions.

Conclusion

This paper presents an effective h-MoEDA, which solves the bicriteria S-JSP with the uncertainty of processing time. It minimized the expected average makespan and expected total tardiness within a reasonable amount of calculation time. With the framework of the proposed EDA, the explicit probability model of the operation sequence is estimated on the distribution of good solutions found so far and use the constructed model to guide further search behavior. The sampling operator based on the probability model achieves better convergence and stability than the conventional operator such

Table 5 Comparison of the spacing measure by h-MoEDA, NSGA-II, and SPEA2 and AWGA

Problem	Algorithm (A)	Mean		Reduced (SP(h-MoEDA)-SP(A))/SP(A) (%)	SD	
		SP(h-MoEDA)	SP(A)		SP(h-MoEDA)	SP(A)
LA29	NSGA-II	16.3568	20.1712	−18.91	4.4081	6.4981
	SPEA2	16.3568	22.2868	−26.61	4.4081	6.7942
	awGA	16.3568	25.8421	−36.70	4.4081	7.2521
LA35	NSGA-II	19.0001	23.8413	−20.31	4.9569	6.2297
	SPEA2	19.0001	24.9510	−23.85	4.9569	7.0526
	awGA	19.0001	33.2410	−42.84	4.9569	7.1686
LA38	NSGA-II	17.1736	21.4569	−19.96	5.2566	6.9774
	SPEA2	17.1736	24.0161	−28.49	5.2566	7.1136
	awGA	17.1736	27.0155	−36.43	5.2566	7.9576

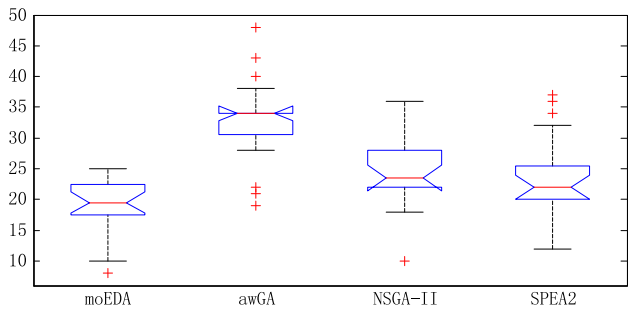

Fig. 9 Box plot of space measure conducted on LA35

crossover and mutation. In our future work, further experiments will be conducted to determine the accuracy of the proposed EDA in response to variations among the parameters. Furthermore, we will extend MoEDA to adapt to real case study based on multiobjective stochastic flexible job-shop models (moS-JSP).

Acknowledgments This work is partly supported by the Japan Society of Promotion of Science (JSPS): Grant-in-Aid for Scientific Research (C) (No. 24510219.0001), the Fundamental Research Funds (Software+X) of Dalian University of Technology (Nos. DUT12JR05, DUT12JR12), and supported by New Teacher Fund of Ministry of Education of China (No. 20120041120053).

References

Azadeh, A., Negahban, A., & Moghaddam, M. (2011). A hybrid computer simulation-artificial neural network algorithm for optimisation of dispatching rule selection in stochastic job shop scheduling problems. *International Journal of Production Research, 50*(2), 551–566.

Bianchi, L., Dorigo, M., Gambardella, L. M., & Gutjahr, W. J. (2009). A survey on metaheuristics for stochastic combinatorial optimization. *Natural Computing: An International Journal*, *8*, 239–287. doi:10.1007/s11047-008-9098-4.

Cheng, R., Gen, M., & Tsujimura, Y. (1996). A tutorial survey of job-shop scheduling problems using genetic algorithms, part I: Representation. *Computers & Industrial Engineering*, *30*, 983–97.

Cheng, R., Gen, M., & Tsujimura, Y. (1999). A tutorial survey of job-shop scheduling problems using genetic algorithms, part II: Hybrid genetic search strategies. *Computers & Industrial Engineering*, *36*, 343–364.

Deb, K, Agrawal, S., Pratap A., & Meyarivan T. (2000). A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. *Lecture notes in computer science, 1917*, 849–858.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*, 182–197.

Essafi, I., Mati, Y., & Dauzère-Pérès, S. (2008). A genetic local search algorithm for minimizing total weighted tardiness in the job-shop scheduling problem. *Computers & Operations Research*, *35*(8), 2599–2616.

Fonseca, C. M., & Fleming, P. J. (1995). An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, *3*(1), 1–16.

Gao, J., Sun, L., & Gen, M. (2008). A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. *Computers & Operations Research*, *35*(9), 2892–2907.

Gen, M., & Cheng, R. (1997). *Genetic algorithms and engineering design* (432 pp). New York: Wiley.

Gen, M., Lin, L., & Zhang, H. (2009). Evolutionary techniques for optimization problems in integrated manufacturing system: State-of-the-art-survey". *Computers & Industrial Engineering, 56*(3), 779–808.

Gen, M., & Lin, L. (2014). Multiobjective evolutionary algorithm for manufacturing scheduling problems: State-of-the-art survey. *Journal of Intelligent Manufacturing, 25*(5), 849–866.

Gen, M., Tsujimura, Y., & Kubota, E. (1994). Solving job-shop scheduling problem by genetic algorithm. In *Proceedings of IEEE international conference on systems, man, and cybernetics* (pp. 1577–1582).

Gen, M., & Cheng, R. (2000). *Genetic algorithms and engineering optimization* (512 pp). New York: Wiley.

Gen, M., Cheng, R., & Lin, L. (2008). *Network models and optimization: Multiobjective genetic algorithm approach*. Berlin: Springer.

Gen, M., & Lin, L. (2014). Multiobjective evolutionary algorithm for manufacturing scheduling problems: State-of-the-art survey. *Journal of Intelligent Manufacturing*, *25*(5), 849–866.

Gholami, M., & Zandieh, M. (2009). Integrating simulation and genetic algorithm to schedule a dynamic flexible job shop. *Journal of Intelligent Manufacturing*, *20*, 481–498.

Giffler, B., & Thompson, G. L. (1960). Algorithms for solving production-scheduling problems. *Operations Research*, *8*(4), 487–503.

Golenko-Ginzburg, D., & Gonik, A (2002). Optimal job-shop scheduling with random operations and cost objectives. *International Journal of Production Economics, 76*(2), 147–157.

Gu, J., Gu, M., Cao, C., & Gu, X. (2010). A novel competitive co-evolutionary quantum genetic algorithm for stochastic job shop scheduling problem. *Computers & Operations Research, 37*(5), 927–937.

Gu, J., Gu, X., & Gu, M. (2009). A novel parallel quantum genetic algorithm for stochastic job shop scheduling. *Journal of Mathematical Analysis and Applications*, *355*(1), 63–81.

Hao, X. C., Lin, L., Gen, M., & Suer, G. (2013). Hybrid evolutionary algorithms and uncertainty in manufacturing & logistics systems III: Effective EDA for multiobjectives stochastic job-shop scheduling problem. In *Proceedings of the 43rd international conference on computers & industrial engineering*, Hong Kong.

Hao, X. C., Lin, L., Gen, M., & Ohno, K. (2013). Effective estimation of distribution algorithm for stochastic job shop scheduling problem. *Procedia Computer Science*, *20*, 102–107.

Hasan, S.M.K., Sarker, R., & Essam, D. (2011). Genetic algorithm for job-shop scheduling with machine unavailability and breakdowns. *International Journal of Production Research, 49*(2), 4999–5015.

Horng, S.-C., Lin, S.-S., & Yang, F.-Y. (2012). Evolutionary algorithm for stochastic job shop scheduling with random processing time. *Expert Systems with Applications*, *39*, 3603–3610.

Ishibuchi, H., & Murata, T. (1998). 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), 392–403.

Kall, P., & Wallace, S. W. (1994). *Stochastic programming*. New York: Wiley.

Kutanoglu, E., & Sabuncuoglu, I. (2001). Experimental investigation of iterative simulation-based scheduling in a dynamic and stochastic job shop. *Journal of Manufacturing Systems, 20*(4), 264–279.

Larrañaga, P., & Lozano, J. A. (2002). *Estimation of distribution algorithms: A new tool for evolutionary computation*. Berlin: Springer.

Lawler, E. L., Lenstra, J. K., Kan, A. R., & Shmoys, D. B. (1993). Sequencing and scheduling: Algorithms and complexity. *Handbooks of Operations Research & Management Science*, *4*, 445–522.

Lawrence, S. (1984). *Resource constrained project scheduling: An experimental investigation of heuristic scheduling techniques (supplement)*. Pittsburgh, Pennsylvania: Graduate School of Industrial Administration, Carnegie-Mellon University.

Lei, D. (2008). Pareto archive particle swarm optimization for multi-objective fuzzy job shop scheduling problems. *International Journal Advanced Manufacturing Technology, 37*(1), 157–165.

Lei, D. (2012). Minimizing makespan for scheduling stochastic job shop with random breakdown. *Applied Mathematics and Computation*, *218*(24), 11851–11858.

Lei, D-M., & Xiong H-J. (2007). An Efficient Evolutionary Algorithm for Multi-Objective Stochastic Job Shop Scheduling. *International Conference on Machine Learning and Cybernetics*, 867–872.

Lin, L., Hao, X.-C., Gen, M., & Jo, J.-B. (2012). Network modeling and evolutionary optimization for scheduling in manufacturing. *Journal of Intelligent Manufacturing*, *23*, 2237–2253.

Liu, B., Wang, L., & Jin, Y. (2005). Hybrid particle swarm optimization for flow shop scheduling with stochastic processing time. In *Computer intelligent security* (pp. 630–637). Berlin: Springer.

Michiels, W., Aarts, E., & Korst, J. (2010). *Theoretical aspects of local search* (1st ed.). Berlin: Springer.

Ouelhadj, D., & Petrovic, S. (2009). A survey of dynamic scheduling in manufacturing systems. *Journal of Scheduling*, *12*(4), 417–431.

Petrovic, S., Fayad, C., Petrovic, D., Burke, E., & Kendall, G. (2008). Fuzzy job shop scheduling with lot-sizing. *Annals of Operations Research, 159*(1), 275–292.

Raheja, A.S., & Subramaniam, V. (2002). Reactive recovery of job shop schedules–a review. *The International Journal of Advanced Manufacturing Technology, 19*(10), 756–763.

Schaffer, J. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st international conference on genetic algorithms* (pp. 93–100). Hillsdale, NJ: L. Erlbaum Associates Inc.

Srinivas, N., & Deb, K. (1994). Multiobjective optimization using non-dominated sorting in genetic algorithms. *Evolutionary computation, 2*(3), 221–248.

Subramaniam, V, Raheja, A.S., & Rama Bhupal Reddy, K. (2005). Reactive repair tool for job shop schedules. *International Journal of Production Research, 43*(1), 1–23

Tavakkoli-Moghaddam, R., Jolai, F., Vaziri, F., Ahmed, P. K., & Azaron, A. (2005). A hybrid method for solving stochastic job shop scheduling problems. *Applied Mathematics and Computation*, *170*, 185–206.

Yang, H., Lv, Y., Xia, C., Sun, S., & Wang, H. (2014). Optimal computing budget allocation for ordinal optimization in solving stochastic job shop scheduling problems. *Mathematical Problems in Engineering*, e619254. doi:10.1155/2014/619254

Yoshitomi, Y., & Yamaguchi, R. (2003). A genetic algorithm and the Monte Carlo method for stochastic job-shop scheduling. *International Transactions in Operational Research, 10*(6), 577.

Zhang, R., & Wu, C. (2011). An artificial bee colony algorithm for the job shop scheduling problem with random processing times. *Entropy*, *13*(9), 1708–1729. doi:10.3390/e13091708

Zhang, R., Song, S., & Wu, C. (2012). A two-stage hybrid particle swarm optimization algorithm for the stochastic job shop scheduling problem. *Knowledge-Based Systems*, *27*, 393–406.

Zhang, W., Xu, W., & Gen, M. (2013). Multi-objective evolutionary algorithm with strong convergence of multi-area for assembly line balancing problem with worker capability. *Procedia Computer Science*, *20*, 83–89.

Zhang, W., Gen, M. & Jo, J.B. (2014). Hybrid sampling strategy-based multiobjective evolutionary algorithm for process planning and scheduling problem. *Journal of Intelligent Manufacturing, 25*(5), 881–897

Zhou, R., Nee, A. Y. C., & Lee, H. P. (2009). Performance of an ant colony optimisation algorithm in dynamic job shop scheduling problems. *International Journal of Production Research*, *47*, 2903–2920.

Zitzler, E., Laumanns, M., & Thiele, L. (2001). SPEA2: Improving the strength Pareto evolutionary algorithm. In *Evolutionary methods for design, optimization and control*. Barcelona: CIMNE.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strengthen Pareto approach. *IEEE Transaction on Evolutionary Computation*, *4*(3), 257–271.