J Sched (2014) 17:5–15
DOI 10.1007/s10951-013-0349-6

Minimizing conditional-value-at-risk for stochastic scheduling problems

Subhash C. Sarin · Hanif D. Sherali · Lingrui Liao

Received: 15 February 2011 / Accepted: 18 September 2013 / Published online: 8 October 2013

Abstract This paper introduces the use of conditional-value-at-risk (CVaR) as a criterion for stochastic scheduling problems. This criterion has the tendency of simultaneously reducing both the expectation and variance of a performance measure, while retaining linearity whenever the expectation can be represented by a linear expression. In this regard, it offers an added advantage over traditional nonlinear expectation-variance-based approaches. We begin by formulating a scenario-based mixed-integer program formulation for minimizing CVaR for general scheduling problems. We then demonstrate its application for the single machine total weighted tardiness problem, for which we present both a specialized L-shaped algorithm and a dynamic programming-based heuristic procedure. Our numerical experimental results reveal the benefits and effectiveness of using the CVaR criterion. Likewise, we also exhibit the use and effectiveness of minimizing CVaR in the context of the parallel machine total weighted tardiness problem. We believe that minimizing CVaR is an effective approach and holds great promise for achieving risk-averse solutions for stochastic scheduling problems that arise in diverse practical applications.

Keywords Stochastic scheduling · Conditional-value-at-risk · Total weighted tardiness · Benders decomposition · Dynamic programming

S. C. Sarin (✉) · H. D. Sherali · L. Liao
Grado Department of Industrial and Systems Engineering,
Virginia Tech, Blacksburg, VA 24061, USA
e-mail: sarins@vt.edu

1 Introduction

Machine scheduling problems involve the processing of given sets of jobs on a finite number of machines. The performance measure that is typically used is a scalar function of job completion times. Deterministic versions of this problem have been studied extensively over several decades. However, as pointed out by McKay et al. (1988), the practical applicability of deterministic scheduling models is hampered due to its disregard of uncertainty factors, which are encountered frequently in practice. Consequently, various approaches have been developed to address uncertainty in scheduling. *Robust scheduling* strives to provide the best protection against worst-case scenarios (Daniels and Kouvelis 1995; Kouvelis et al. 2000). *Reactive scheduling* modifies a baseline schedule after uncertainties are revealed during its implementation (Sabuncuoglu and Bayiz 2000; Vieira et al. 2003). *Fuzzy scheduling* is suitable for situations in which available data are insufficient to construct viable probabilistic models, hence fuzzy numbers are used to represent uncertain parameters such as job processing times (Balasubramanian and Grossmann 2003). On the other hand, *stochastic scheduling* tries to find solutions that optimize a performance measure under the assumption that the uncertain parameters are random variables with known distributions. In this spirit, for cases when probability density functions of job processing times are known, Sarin et al. (2010) presented an expectation-variance analysis for a given job assignment/sequence solution. In this paper, we address stochastic scheduling problems where a reliable prior knowledge of random parameters exists.

Since the value of an objective function depends on problem parameters, randomness of these parameters implies that the objective value is also a random variable. To compare performances of different schedules, some distributional prop-

erty of the objective function is usually adopted as a scheduling criterion. A commonly used criterion is the expected value, which can be regarded as the long-run average performance of a schedule (see, for example, Pinedo 2001 and Skutella and Uetz 2005). As pointed out by Daniels and Kouvelis (1995), a critical disadvantage of using expectation as a performance measure is that it does not account for the risk-averse attitude of a decision-maker. De et al. (1992) used variance as a risk measure and determined expectation-variance-based efficient schedules. However, using variance as a risk measure has several drawbacks. First, except for some special cases (such as the single machine flow time problem discussed by De et al. 1992), it is difficult to derive analytical expressions for the variance of typical performance measures. Moreover, in case a scenario-based approach is adopted, the sample variance of any given performance measure involves a quadratic expression, which makes the optimization problem relatively hard to solve. Second, minimizing the variance of a random variable equally penalizes positive and negative deviations from its mean value. For example, suppose that two solutions yield identical expectation and variance values of the objective function, and that the distribution of the objective function value for the first solution has a longer tail on the right side of its probability density function. For a minimization problem, a risk-averse decision-maker would choose the second solution to avoid the risk of encountering large outcomes. However, based on the values of the expectation and variance alone, it is not possible to distinguish between these two solutions. To avoid the above drawbacks of variance, we use the conditional-value-at-risk (CVaR) as a criterion for stochastic scheduling. This criterion, which was introduced in the finance context by Rockafellar and Uryasev (2000), is advantageous due to its tendency of simultaneously minimizing both the variance and the expectation of a given performance measure, while maintaining linearity whenever the expectation objective function can be represented by a linear expression.

In this paper, we formulate a scenario-based mixed-integer program (MIP) to minimize CVaR for general scheduling problems. We propose a decomposition approach based on Benders algorithm (Benders 1962) for its solution, wherein we include certain additional valid inequalities to strengthen the relaxed master program. We demonstrate the effectiveness of minimizing CVaR through two classic scheduling problems: the single machine total weighted tardiness problem and the parallel machine total weighted tardiness problem. To solve large-sized problem instances, we also develop a dynamic programming-based heuristic procedure, which is shown to yield good solutions with significantly reduced computational effort.

The remainder of this paper is organized as follows. In Sect. 2, we introduce the concept of CVaR and formulate a general model for its minimization. In Sect. 3, we consider the application of the foregoing formulation to the classic single machine total weighted tardiness scheduling problem, and we develop an optimization approach for the solution of this problem, as well as a heuristic procedure for handling large-sized instances. Results of numerical experiments are provided to demonstrate the benefits and effectiveness of using the CVaR criterion. An extension of this approach to a parallel machine total weighted tardiness problem is addressed in Sect. 4. Finally, some concluding remarks are presented in Sect. 5.

2 Optimizing CVaR in a stochastic scheduling environment

2.1 Motivation

Definition 1 Given a random variable X and a probability level $\alpha \in (0, 1)$, the Value-at-Risk (VaR) and the Conditional-Value-at-Risk (CVaR) are, respectively, defined as:

$$\eta_X(\alpha) = \inf\{x : F_X(x) \geq \alpha\}$$

and

$$\phi_X(\alpha) = E[X|X \geq \eta_X(\alpha)] = \frac{1}{1-\alpha} \int_{\eta_X(\alpha)}^{+\infty} x \, \mathrm{d}F_X(x),$$

where $F_X(x)$ is the cumulative distribution function of X.

Loosely speaking, $\phi_X(\alpha)$ is the average value of the $(1 - \alpha) \cdot 100$ % largest outcomes of X. As a risk measure, CVaR possesses several useful properties, which have enabled it to gain popularity in the fields of insurance and finance. First of all, CVaR is consistent with second degree stochastic dominance (SSD) (see Ogryczak and Ruszczynski 2002). Specifically, if a random variable X dominates a random variable Y under the SSD rule (denoted as $X \geq_{\mathrm{SSD}} Y$), then X is better than Y with respect to all risk-averse non-decreasing utility functions. CVaR is consistent with SSD in the sense that

$$X \geq_{\mathrm{SSD}} Y : \phi_X(\alpha) \leq \phi_Y(\alpha), \quad \forall \alpha \in (0, 1).$$

Interestingly, this property does not always hold for the variance (see Porter and Gaumnitz 1972). Secondly, CVaR is a coherent risk measure as defined by Artzner et al. (1999), in that it satisfies the four axioms of translation invariance, subadditivity, positive homogeneity, and monotonicity. The VaR, on the other hand, does not satisfy subadditivity. Also, note that, for any random variable X with finite values of $\phi_X(\alpha)$ and $E[X]$, we always have $\phi_X(\alpha) \geq E[X]$, $\forall \alpha \in (0, 1)$. Therefore, minimizing the CVaR of a random variable tends to reduce its expected value as well. Furthermore, it can be shown that (see Rockafellar and Uryasev 2000):

$$\phi_X(\alpha) = \min_{\eta \in \Re} \left\{ \eta + \frac{1}{1-\alpha} E\left[(X-\eta)^+\right] \right\} \quad (1)$$

where $(\xi)^+$ denotes $\max\{0, \xi\}$. This representation allows the incorporation of CVaR into an optimization framework. Wang and Ahmed (2008) have used CVaR to construct side-constraints while minimizing the expected objective value. In contrast, as discussed next, we investigate the formulation of a stochastic scheduling problem that directly minimizes CVaR as the objective function.

2.2 Problem formulation and a decomposition approach

Consider a general scheduling problem whose deterministic version can be formulated as the following MIP:

$$\begin{array}{lll} \mathbf{P_1}: & \text{Minimize} & \mathbf{h}^{\mathrm{T}}\mathbf{y} \\ & \text{subject to:} & \mathbf{x} \in \Gamma, \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & & \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{y} \geq \mathbf{g} \\ & & \mathbf{y} \geq 0, \end{array}$$

where the vectors $\mathbf{x}$ and $\mathbf{y}$ are the decision variables, and the set Γ denotes bound values and possibly integrality restrictions. Suppose now that due to the stochastic nature of the problem, once $\mathbf{x}$ is determined, the parameters $\mathbf{C}$ and $\mathbf{g}$ are subject to random variations. We assume that such randomness is captured by a finite set of scenarios, S, with corresponding parameter values $\mathbf{C}_s$ and $\mathbf{g}_s$, $\forall s \in S$. These scenarios are derived either from some discrete approximation of the underlying distributions of the problem parameters, or from some scenario generation procedure, with the probability value π_s being associated with a scenario s, $\forall s \in S$. The stochastic version of Problem $\mathbf{P_1}$ can then be formulated as a two-stage stochastic program as follows:

$$\begin{array}{lll} \mathbf{P_2}: & \text{Minimize} & \sum_{s\in S} \pi_s \mathbf{h}^{\mathrm{T}}\mathbf{y}_s \\ & \text{subject to:} & \mathbf{x} \in \Gamma, \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & & \mathbf{C}_s\mathbf{x} + \mathbf{D}\mathbf{y}_s \geq \mathbf{g}_s, \quad \forall s \in S \\ & & \mathbf{y}_s \geq 0, \qquad\quad \forall s \in S, \end{array}$$

where the objective function considers the traditional minimization of the expected value. In this context, $\mathbf{x}$ is the vector of first-stage decision variables, which need to be determined before the values of the random parameters are observed as alluded above, and $\mathbf{y} \equiv \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_S\}$ is the vector of second stage recourse variables with the sub-vector $\mathbf{y}_s$ denoting the decision under scenario s, $\forall s \in S$. Furthermore, note that we assume a fixed recourse situation, so that the matrix $\mathbf{D}$ is constant across all scenarios.

As discussed in the introduction, our interest lies in minimizing CVaR for the scheduling criterion $\mathbf{h}^{\mathrm{T}}\mathbf{y}$, instead of minimzing the common expected value objective as in Problem $\mathbf{P_2}$. Adopting the derivation (1) of Rockafellar and Uryasev (2000), it can be shown that the following MIP provides an equivalent formulation for the minimization of CVaR:

$$\mathbf{MP}: \text{Minimize} \quad \phi = \eta + \frac{1}{1-\alpha} \sum_{s\in S} \pi_s \mu_s$$

$$\begin{array}{lll} \text{subject to:} & \mathbf{x} \in \Gamma, \mathbf{A}\mathbf{x} \geq \mathbf{b} & (2) \\ & \eta + \mu_s \geq \mathbf{h}^{\mathrm{T}}\mathbf{y}_s, \qquad \forall s \in S & (3) \\ & \mathbf{C}_s\mathbf{x} + \mathbf{D}\mathbf{y}_s \geq \mathbf{g}_s, \qquad \forall s \in S & (4) \\ & \mu_s \geq 0 \text{ and } \mathbf{y}_s \geq 0, \qquad \forall s \in S. & (5) \end{array}$$

Note that, by constraints (3) and (5), μ_s represents $(\mathbf{h}^{\mathrm{T}}\mathbf{y}_s - \eta)^+$. Because of the size and structure of Problem $\mathbf{MP}$, particularly for a large number of scenarios, it is attractive to consider Benders decomposition to solve this problem (Benders 1962). For each scenario $s \in S$, we define the second-stage problem as follows:

$$\mathbf{MP_{II}}(s): \quad Q_s(\mathbf{x}) \equiv \min_{\mathbf{y}_s \geq 0} \{\mathbf{h}^{\mathrm{T}}\mathbf{y}_s : \mathbf{D}\mathbf{y}_s \geq \mathbf{g}_s - \mathbf{C}_s\mathbf{x}\}.$$

We further assume relatively complete recourse, so that Problem $\mathbf{MP_{II}}(s)$ is feasible for all relevant values of $\mathbf{x}$ feasible to (2), and that $\mathbf{MP_{II}}(s)$ is bounded. Observe that, consequently, the first stage problem seeks to minimize ϕ subject to (2), $\mu_s \geq 0$ for $s \in S$, and that $\eta + \mu_s \geq Q_s(x)$, $\forall s \in S$. Hence, the first-stage problem can be written as follows:

$$\mathbf{MP_I}: \text{Minimize} \quad \phi = \eta + \frac{1}{1-\alpha} \sum_{s\in S} \pi_s \mu_s$$

$$\begin{array}{ll} \text{subject to:} & \mathbf{x} \in \Gamma, \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \eta + \mu_s \geq (\mathbf{g}_s - \mathbf{C}_s\mathbf{x})^{\mathrm{T}}\psi, \qquad \forall s \in S, \psi \in \Psi \\ & \mu_s \geq 0, \qquad\qquad\qquad\qquad \forall s \in S, \end{array} \quad (6)$$

where Ψ denotes the set of extreme point feasible solutions to the dual of $\mathbf{MP_{II}}(s)$ for any $s \in S$, and where the latter is given as follows:

$$Q_s(\mathbf{x}) \equiv \max_{\psi \geq 0} \{(\mathbf{g}_s - \mathbf{C}_s\mathbf{x})^{\mathrm{T}}\psi : \mathbf{D}^{\mathrm{T}}\psi \leq \mathbf{h}\}.$$

Since the set Ψ is in general exponentially sized, we solve Problem $\mathbf{MP_I}$ via a constraint relaxation and cut generation procedure. Specifically, given a solution $\mathbf{x}$ to some relaxation of $\mathbf{MP_I}$ that only has a subset of restrictions in (6) for some selected $\psi \in \Psi_s \subset \Psi$, $\forall s \in S$, we compute $Q_s(\mathbf{x})$ by solving $\mathbf{MP_{II}}(s)$, $\forall s \in S$. In the case where the inequality $\eta + \mu_s \geq Q_s(\mathbf{x})$ is violated for some $s \in S$, we generate an optimality cut of the form $\eta + \mu_s \geq (\mathbf{g}_s - \mathbf{C}_s\mathbf{x})^{\mathrm{T}}\psi^*$, where ψ^* denotes an optimal dual solution to $\mathbf{MP_{II}}(s)$, and we augment $\Psi_s \leftarrow \Psi_s \cup \{\psi^*\}$. Also note that as described by Sherali and Lunday (2010), we can generate maximally non-dominated Benders cuts by perturbing the right-hand side of $\mathbf{MP_{II}}(s)$ to $\mathbf{g}_s(1+\lambda) - \mathbf{C}_s(\mathbf{x} + \lambda\mathbf{e})$, where $\mathbf{e}$ is a conformable vector of ones and λ is a suitable (small) perturbation parameter. Adding such cuts for all $s \in S$ as necessary to the relaxed $\mathbf{MP_I}$, we re-solve the augmented first stage

problem and repeat the above procedure until we obtain a first stage solution for which $\eta + \mu_s \geq Q_s(\mathbf{x})$, $\forall s \in S$. To reduce the number of added cuts, we aggregate, using equal weights, the optimality cuts that are generated within the same iteration. We also embed this cut generation procedure in a branch-and-bound (B&B) framework whenever Γ contains integrality restrictions, so that the B&B tree is explored only once. The resulting algorithm is a specialized implementation of the integer L-shaped method (Birge and Louveaux 1997, Chap. 8). In addition, the following valid inequality further improves the (initial) relaxation of $\mathbf{MP_I}$.

Theorem 1 *For any given set $S' \subset S$ such that $\pi_{S'} \triangleq \sum_{s\in S'} \pi_s \geq 1-\alpha$, the following inequality is valid for Problem* **MP***:*

$$\phi \geq \frac{1}{\pi_{S'}} \sum_{s\in S'} \pi_s \mathbf{h}^{\mathrm{T}} \mathbf{y}_s. \tag{7}$$

Proof By aggregating the inequalities (3) using weights of π_s, $\forall s \in S'$, we have

$$\begin{aligned} &\sum_{s\in S'} \pi_s \eta + \sum_{s\in S'} \pi_s \mu_s \geq \sum_{s\in S'} \pi_s \mathbf{h}^{\mathrm{T}} \mathbf{y}_s \\ &\quad \Rightarrow \eta + \frac{1}{\pi_{S'}} \sum_{s\in S'} \pi_s \mu_s \geq \frac{1}{\pi_{S'}} \sum_{s\in S'} \pi_s \mathbf{h}^{\mathrm{T}} \mathbf{y}_s. \end{aligned} \tag{8}$$

Since $\pi_{S'} \geq 1-\alpha$, we further have

$$\begin{aligned} \phi &= \eta + \frac{1}{1-\alpha} \sum_{s\in S} \pi_s \mu_s \\ &\geq \eta + \frac{1}{\pi_{S'}} \sum_{s\in S} \pi_s \mu_s \geq \eta + \frac{1}{\pi_{S'}} \sum_{s\in S'} \pi_s \mu_s. \end{aligned} \tag{9}$$

Using (8) in (9) thus validates (7). □

The idea here is to select some reasonable set of scenarios $S' \subset S$ that satisfy the condition of Theorem 1 and then to explicitly incorporate the restrictions (7) along with the constraints (4) and (5) pertaining to $s \in S'$ directly within $\mathbf{MP_I}$. We denote these restrictions as:

$$(\phi, \mathbf{x}, \mathbf{y}) \in Z. \tag{10}$$

In particular, this set Z provides a lower bound for CVaR based on the outcomes of a selected set of scenarios, and induces a tighter relaxation for the first-stage problem $\mathbf{MP_I}$ without complicating its formulation too much. Ideally, the selected scenarios for S' should yield relatively large values of $\mathbf{h}^T\mathbf{y}_s$ at optimality. Note that with this modification, ϕ is regarded both as a decision variable and as the objective function of $\mathbf{MP_I}$, where we accommodate within the constraints the additional inequality: $\phi \geq \eta + \frac{1}{1-\alpha} \sum_{s\in S} \pi_s \mu_s$.

Alternatively, we can incorporate the constraints (3), (4), and (5) pertaining to $s \in S'$ directly within $\mathbf{MP_I}$. We denote these restrictions as:

$$(\eta, \boldsymbol{\mu}, \mathbf{x}, \mathbf{y}) \in Z', \tag{11}$$

which can be regarded as a disaggregated version of the restrictions in (10).

3 Application to a single machine scheduling problem

To demonstrate the viability and benefit of optimizing CVaR, we first apply the above methodology to the single machine total weighted tardiness (TWT) problem. We assume the processing times to be the only random elements in this problem. Consider the following notation:

Parameters:

J: Set of jobs;
w_j: Weight of job j, $\forall j \in J$;
d_j: Due date of job j, $\forall j \in J$;
π_s: Probability of scenario s, $\forall s \in S$.
p_j^s: Processing time of job j under scenario s, $\forall j \in J, s \in S$.

Decision variables:

c_j^s: Completion time of job j under scenario s, $\forall j \in J, s \in S$;
t_j^s: Tardiness of job j under scenario s, $\forall j \in J, s \in S$;
η: A threshold value (equal to the value-at-risk when an optimal solution is obtained);
μ_s: Amount of TWT exceeding the threshold value η under scenario s, $\forall s \in S$;
$z_{i,j}$: Job precedence binary variable, where, $z_{i,j} = 1$ if job i is processed before job j, $z_{i,j} = 0$ otherwise, $\forall i \neq j \in J$.

The model formulation of the single machine total weighted tardiness problem SM-TWTP is as follows:

SM − TWTP :

$$\text{Minimize} \quad \phi = \eta + \frac{1}{1-\alpha} \sum_{s\in S} \pi_s \mu_s$$

$$\text{subject to:} \quad \eta + \mu_s \geq \sum_{j\in J} w_j t_j^s, \qquad \forall s \in S \tag{12}$$

$$\sum_{i\in J\setminus\{j\}} p_i^s z_{i,j} + p_j^s \leq c_j^s, \qquad \forall s \in S, j \in J \tag{13}$$

$$t_j^s + d_j \geq c_j^s, \qquad \forall s \in S, j \in J \tag{14}$$

$$z_{i,j} + z_{j,i} = 1, \qquad \forall i \neq j \in J \tag{15}$$

Table 1 Parameters of an example problem

Job j	1	2	3	4	5	6	7	8
$E[p_j]$	64	98	93	21	12	55	14	19
$Var[p_j]$	42	7	8	36	20	400	5	100
w_j	4	3	1	3	2	5	2	4
d_j	137	157	147	150	168	141	162	142

$$z_{i,j} + z_{j,k} + z_{k,i} \leq 2, \qquad \forall i \neq j \neq k \in J \tag{16}$$

$$c_j^s \geq 0, t_j^s \geq 0, \mu_s \geq 0, \qquad \forall s \in S, j \in J \tag{17}$$

$$z_{i,j} \in \{0, 1\}, \qquad \forall i \neq j \in J.$$

For each scenario s, constraint (12) (along with (17)) determines μ_s as the amount of TWT that exceeds the threshold value of η (if at all). Constraint (13) bounds the job completion times according to job sequencing relationships. Constraints (14) and (17) determine the tardiness of jobs. Constraints (15) and (16) ensure feasibility of the job sequence by eliminating cyclic sequences (see Sarin et al. 2005). Note that the variables $\{z_{i,j}\}$ and $\{c_j^s, t_j^s\}$ correspond to the variables $\mathbf{x}$ and $\mathbf{y}_s$, respectively, in the general formulation **MP**.

As an illustration, consider an 8-job problem with 100 scenarios. The job processing times are assumed to follow left-truncated normal distributions (truncated at zero to ensure nonnegativity). The job parameter values are summarized in Table 1. Scenario-wise values of p_j were generated via Monte Carlo sampling. This problem was solved to optimality for each of the criteria of minimizing the expected value as well as minimizing CVaR (with $\alpha = 0.8$), which resulted in different optimal sequences: sequence $\delta_E = \langle 1, 6, 8, 4, 5, 7, 2, 3\rangle$ for the expected value criterion, and sequence $\delta_C = \langle 6, 8, 4, 7, 5, 1, 2, 3\rangle$ for the CVaR criterion. The empirical cumulative distribution functions of total weighted tardiness under both sequences are plotted in Fig. 1. Note that δ_C, the sequence that minimizes CVaR, does not achieve the minimal expected value given by the optimal sequence δ_E. However, for a slight increment in the expected value (about 7 %), δ_C results in almost a 50 % reduction in the value of variance. As a result, the cumulative distribution function of the TWT corresponding to δ_C reaches probability 1 at the TWT value of 1179.5, while the TWT corresponding to δ_E has a substantial associated probability of exceeding this value. This example illustrates the risk-averse nature of CVaR and its effectiveness in reducing variability.

To further illustrate the above observation, we randomly generated 30 sample problems with 8 jobs and with uniformly distributed processing times. The processing time distribution for job j was taken as

$$U(\rho_j(1 - r_j), \rho_j(1 + r_j)),$$

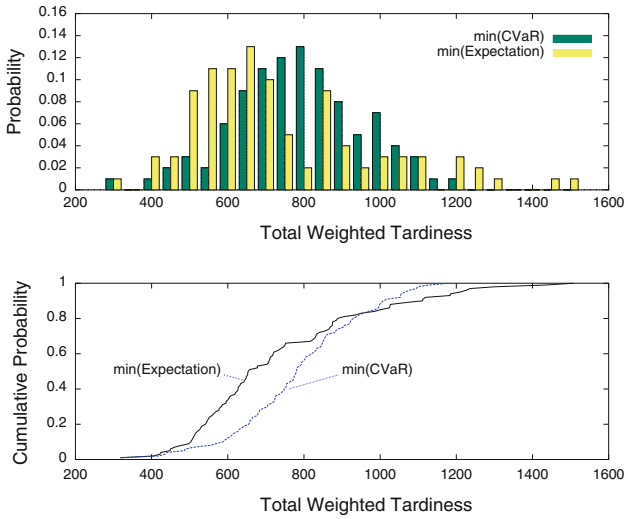

Fig. 1 Cumulative distribution function and histogram of TWT for the expected value and CVaR criteria

where the values of ρ_j and r_j were generated from uniform distributions $U(20, 100)$ and $U(0.1, 0.25)$, respectively. The job due dates were generated according to

$$U\left(1 - TF - \frac{RDD}{2}, 1 - TF + \frac{RDD}{2}\right) \cdot \sum_{j \in J} E[p_j],$$

with a tardiness factor TF of 0.6, and a relative due date RDD of 0.4. One hundred scenarios were generated for each sample problem. After the optimal sequences (δ_E and δ_C) were obtained, the expectation and variance of the total weighted tardiness were evaluated for both δ_E and δ_C. The results obtained are plotted in Fig. 2. The horizontal axis represents the ratio of expectations corresponding to the optimal sequences δ_C and δ_E, which indicates the relative increment in expectation due to choosing δ_C over δ_E. The vertical axis corresponds to the ratio of the corresponding variances. The smaller the ratio, the greater is the reduction in variability of the TWT value. Note that for 15 out of the 30 problem instances used, the resulting distributions of TWT are different for δ_E and δ_C. In these cases, choosing δ_C over δ_E leads to a significant reduction in the variability of TWT, with only a slight increment in its expected value.

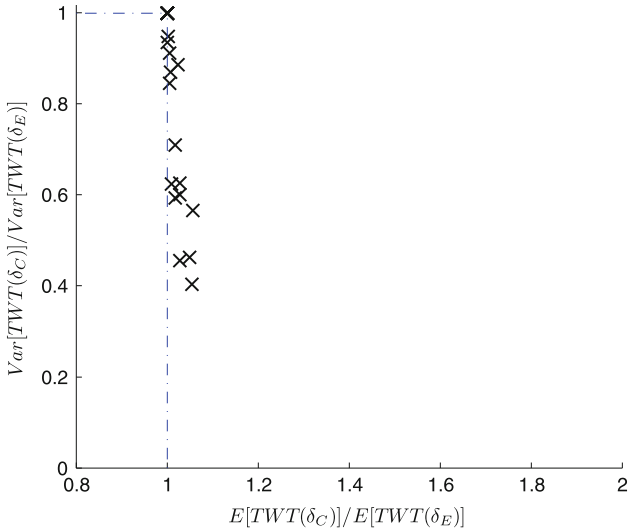

Fig. 2 Expectation-variance comparison for the sequences δ_C and δ_E for the 8-job single machine problems

3.1 Solution approach for SM-TWTP

To solve the SM-TWTP model more efficiently, we implemented the integer L-shaped method after enhancing its first-stage model with the valid inequalities (10) or (11). In particular, the selected set of scenarios, S', was determined according to the total weighted processing times. A scenario having a higher value of the total weighted processing time tends to have larger tardiness values as well. Hence, to induce a tight bound, we constructed the set S' incrementally, starting with the empty set, and then adding scenarios in nonincreasing order of the total weighted processing time, until $\pi_{S'}$ reaches or exceeds $1-\alpha$. In addition, since a job sequence (given by $\{z_{i,j}\}$) is fixed by the first-stage problem $\mathbf{MP_I}$, the subproblem has the following simple structure, which leads to its straightforward solution in closed-form, where $\mathcal{C}_j^s \equiv \sum_{i\in J\setminus\{j\}} p_i^s z_{i,j} + p_j^s$, $\forall j \in J, s \in S$:

$\mathbf{MP_{II}}(s)$:

Minimize	$\sum_{j\in J} w_j t_j^s$	
subject to:	$c_j^s \geq \mathcal{C}_j^s,$	$\forall j \in J$
	$t_j^s - c_j^s \geq -d_j,$	$\forall j \in J$
	$c_j^s \geq 0, t_j^s \geq 0,$	$\forall j \in J.$

Dual of $\mathbf{MP_{II}}(s)$:

Maximize	$\sum_{j\in J}(\mathcal{C}_j^s u_j - d_j v_j)$	
subject to:	$u_j - v_j \leq 0,$	$\forall j \in J$
	$v_j \leq w_j,$	$\forall j \in J$
	$u_j \geq 0, v_j \geq 0,$	$\forall j \in J.$

Note that an optimal dual solution is given by $u_j = v_j = w_j$, if $\mathcal{C}_j^s \geq d_j$; and $u_j = v_j = 0$ otherwise. It is easy to verify that this produces maximally nondominated Benders cuts (see Sherali and Lunday 2010).

To evaluate the performance of our decomposition approach, we benchmarked it against a method that directly uses CPLEX 10.1 to solve the formulation MP. To further reduce the size of the problem formulation, we applied the following substitution for both methods, which is implied by constraint (15):

$$z_{i,j} = 1 - z_{j,i}, \quad \forall i > j.$$

Experiments were conducted on a workstation with a 3.6 GHz CPU and 3 GBytes of RAM. We used $\alpha = 0.8$. In addition to the 8-job problems previously considered, we further generated 30 sample problems having 15 jobs and 400 scenarios. To begin with, we applied the integer L-shaped method with and without the additional inequalities defined by Z given by (10) or Z' given by (11), in order to determine the effectiveness of these alternative formulations. A maximum time limit of 3,600 s was applied. The average results are summarized in Table 2.

Table 2 Average results obtained with L-shaped method, with and without the set Z and Z' inequalities

$\|J\|$	$\|S\|$	Valid ineq.	Initial LP bound	No. of nodes	CPU time (s)
8	100	N/A	0.00	2,400.57	6.49
		Z	1,091.77	140.90	0.92
		Z'	1,091.77	86.00	0.56
15	400	N/A	0.00	110,310.00	3,600.00
		Z	3,003.53	10,077.20	571.71
		Z'	3,003.53	7,793.90	362.37

Note that, the inclusion of the inequalities of Z or Z' greatly enhances the performance of the L-shaped method by inducing a good lower bound for the first stage problem. Between Z and Z', the disaggregated set Z' generated fewer nodes, and hence required less CPU time. However, both sets of restrictions generated identical initial lower bound values. We also experimented with McDaniel and Devine (1977)'s approach of generating an initial set of optimality cuts based on the LP relaxation of $\mathbf{MP_I}$. However, it was not found to improve the computational efficiency of the integer L-shaped method.

With the addition of the Z'-inequalities to $\mathbf{MP_I}$, we further compared the performance of the L-shaped method with the direct solution of Problem **MP** by CPLEX. The results are summarized in Table 3. Although the L-shaped method generated more B&B nodes, since its first-stage problem is smaller in size than the original problem **MP**, it consumed significantly smaller CPU times, reducing the average computational effort required for the 8-job and 15-job instances by 61.6 and 81.0 %, respectively.

Table 3 Comparison of the proposed L-shaped method and a direct solution by CPLEX

$\|J\|$	$\|S\|$	Approach	Initial LP bound	No. of nodes	CPU time (s)
8	100	Directly by CPLEX	1,100.05	64.63	1.46
		L-shaped method	1,091.77	86.00	0.56
15	400	Directly by CPLEX	3,021.33	5,910.70	2,012.37
		L-shaped method	3,003.53	7,793.90	362.37

3.2 Solution of large-sized problems

Although the L-shaped method provides a good alternative approach to solving Problem **MP** directly by CPLEX, yet it cannot be used for large-sized problem instances because of excessive memory requirements. Therefore, to handle large-sized problem instances, we adapted the deterministic Dynasearch method proposed by Congram et al. (2002) and further extended by Grosso et al. (2004) in order to address the stochasticity present in our problem. Dynasearch is a local search heuristic method. Given a complete job sequence, δ, as the current solution, Dynasearch searches all solutions that can be reached from δ by non-overlapping pairwise interchanges of jobs (see Congram et al. 2002, for details). If a neighboring solution is found to be better than δ, it is designated as the starting point for the next iteration of the local search process. Although the local search neighborhood has a size that grows exponentially with the number of jobs, Dynasearch affords an efficient determination of a local optimal solution by using a dynamic programming approach. For this approach, the stages are defined by the number of jobs already fixed in the leading part of a job sequence. At each stage j, Dynasearch considers the application of non-overlapping pairwise interchanges over the first j jobs of δ. The minimum TWT of these j jobs is regarded as the value function $f(j)$. The value of $f(j+1)$ can then be determined recursively by considering the following two options: keeping the $(j+1)$th job at its original position in δ, or by exchanging it with the ith job, $\forall i < j$. Dynasearch has been shown to be quite competitive in solving single machine TWT problems. Grosso et al. (2004) generalized the idea of pairwise interchanges to also include forward- and backward-shifted reinsertions, calling this procedure generalized pairwise interchanges (GPI). Our method is built upon this GPI-version of Dynasearch to exploit its full strength.

Congram et al. (2002) and Grosso et al. (2004) considered the deterministic version of the single machine TWT problem. In our stochastic setting, multiple scenarios introduce additional complications. First, the value function of the dynamic program needs to be modified to account for scenario-wise outcomes. Second, the CVaR objective function precludes the separability required by dynamic programming. In particular, given a partial sequence $\langle \xi, \zeta \rangle$ that minimizes CVaR for the considered $|\xi| + |\zeta|$ jobs, it is not necessarily true that ξ is optimal with respect to CVaR for the jobs in ξ. We show this by an example: Let $|S| = 5$, $\alpha = 0.8$, $|\xi| = 2$, and $|\zeta| = 1$, and $\pi_s = 0.2, \forall s \in S$. For the two jobs in ξ, there exist two alternative sequences: ξ and $\xi' \equiv \text{reverse}(\xi)$. Suppose that the scenario-wise TWT values of these two sequences are $\{1, 2, 2, 5, 2\}$ for ξ, and $\{1, 2, 3, 3, 4\}$ for ξ'. The corresponding CVaR values are 5 and 4, respectively. If the weighted tardiness values of the third job, ζ, are $\{4, 3, 2, 1, 3\}$, the total weighted tardiness values of the three jobs are $\{5, 5, 4, 6, 5\}$ for the sequence $\langle \xi, \zeta \rangle$, and $\{5, 5, 5, 4, 7\}$ for the sequence $\langle \xi', \zeta \rangle$. Note that although $\langle \xi, \zeta \rangle$ yields a smaller CVaR value (=6) for the three job problem, ξ' gives a better value of CVaR for the first two jobs than ξ does. This outcome follows from the fact that $\max\{a, b\} \geq \max\{c, d\} \not\Rightarrow \max\{a + e, b + f\} \geq \max\{c + e, d + f\}$ for all real e, f.

To overcome these difficulties, our proposed method comprises two types of steps: Selection step (S-step) and Optimization step (O-step). Hence, we call our procedure SO-Dynasearch. The S-step chooses a set of worst scenarios, $\hat{S}$, which yield the highest TWT-values for the current solution. This set is defined such that the cumulative probability value, $\pi_{\hat{S}} \equiv \sum_{s \in \hat{S}} \pi_s$, is no less than $1 - \alpha$, but removing any scenario from it would violate this condition. The O-step then uses dynamic programming to search for the sequence that optimizes the average value (weighted by probabilities) of the total weighted tardiness over the scenarios in $\hat{S}$. Note that, the weighted average over $\hat{S}$ gives a lower bound on CVaR by Theorem 1. The definition of stages is the same as that in the Dynasearch method. At each stage j, we consider all non-overlapping combinations of GPI operations over the jobs $\delta_1, \ldots, \delta_j$. The optimal value of $\frac{1}{\pi_{\hat{S}}} \sum_{s \in \hat{S}} \pi_s \sum_{k=1}^{j} w_{[k]} t^s_{[k]}$ [1] is regarded as the value function $f(j)$, and is computed according to the following recursive equation:

$$f(j) = \min \left\{ \begin{array}{l} f(j-1) + \frac{1}{\pi_{\hat{S}}} \sum_{s \in \hat{S}} \pi_s w_{[j]} t^s_{[j]}, \\ \min\limits_{1 \leq i \leq j-1; \gamma} \left\{ \begin{array}{l} f(i-1) \\ + \frac{1}{\pi_{\hat{S}}} \sum_{s \in \hat{S}} \pi_s I^{\gamma}_s(i, j) \end{array} \right\} \end{array} \right\},$$

where $I^{\gamma}_s(i, j)$ is the total weighted tardiness of jobs $\delta_i, \ldots, \delta_j$ in scenario s, subject to the application of some

[1] $w_{[k]}$ and $t^s_{[k]}$ denote the weight and tardiness of the job in the kth position, respectively.

GPI operator $\gamma(i, j)$. In view of these two types of steps, our SO-Dynasearch method can be described as follows:

Control parameters:

N_0: Maximum number of S-steps after each random start;
N_1: Total number of random starts.

Variables:

ϕ^*: The smallest value of CVaR found so far;
δ^*: The best incumbent solution (job sequence) found so far;
δ: The starting solution of the next local search step;
i_s: Number of S-steps allowed before the next random start;
i_r: Remaining number of random starts.

Function:

TWT(δ, s): The total weighted tardiness resulting from solution δ under scenario s.

SO-Dynasearch method:

Step 1. Let $\phi^* = \infty$, $i_s = N_0$, and $i_r = N_1$. Randomly generate a job sequence and assign it to δ.

Step 2. Use δ to calculate the values of TWT under all scenarios in S. Determine the set of worst scenarios $\hat{S}$ by sorting the TWT values in ascending order and picking the minimal set of top scenarios such that $\pi(\hat{S}) \geq (1-\alpha)$.(S-step)

Step 3. Perform iterated local search starting from δ (O-step):

(a) Use dynamic programming to find a sequence δ', in the GPI neighborhood of δ, which minimizes the average value of TWT across all scenarios in $\hat{S}$.

(b) If δ' is better than δ, let $\delta = \delta'$, and go to Step 3a; otherwise, continue.

Step 4. Use δ to calculate total weighted tardiness values under all scenarios in S. Determine ϕ as the resulting CVaR value. If $\phi < \phi^*$, let $\phi^* = \phi$ and $\delta^* = \delta$.

Step 5. If $i_s > 0$ and $\phi > \frac{1}{\pi_{\hat{S}}} \sum_{s\in\hat{S}} \pi_s \text{TWT}(\delta, s)$, let $i_s \leftarrow i_s - 1$, and go to Step 2; otherwise, continue.

Step 6. If $i_r = 0$, stop; otherwise, let $i_r \leftarrow i_r - 1$ and $i_s = N_0$.

Step 7. Permute δ^* by applying $\frac{|J|}{2}$ random pairwise interchanges sequentially, and denote the resulting sequence as δ. Go to Step 2.

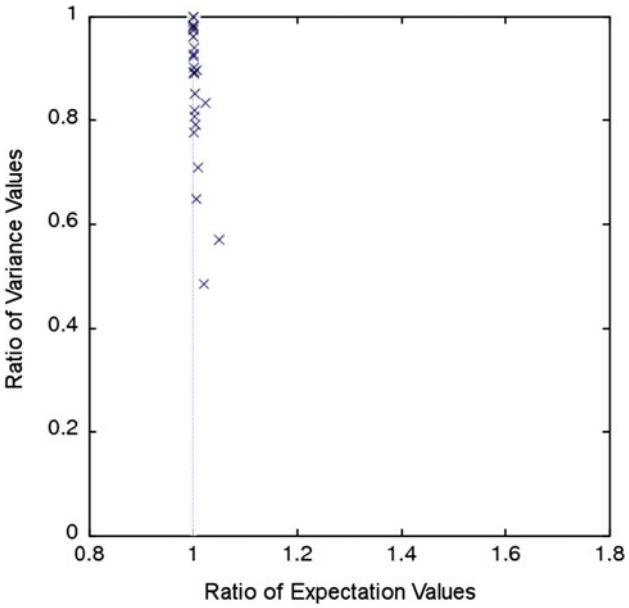

Fig. 3 Expectation-variance comparison of sequences δ_C over δ_E for the 100-job single machine problems

Note that in Step 5, the estimated objective value $\frac{1}{\pi_{\hat{S}}} \sum_{s\in\hat{S}} \pi_s TWT(\delta, s)$ is compared with the true corresponding value of CVaR. If these two values are not the same, the current set $\hat{S}$ does not constitute the worst scenarios under the sequence δ. Therefore, the S- and O-step are repeated to seek a further improvement. The parameter N_0 is used to control the number of repetitions and to avoid an infinite loop. Also, note that SO-Dynasearch can be used to optimize E[TWT] by letting $\alpha = 0$, in which case, the O-step would be executed only once after each random start.

To evaluate the quality of solutions generated by SO-Dynasearch and its computational performance, we implemented it for the same data set used to produce the results of Table 3, for both types of objectives (CVaR $\equiv \phi_{\text{TWT}}(0.8)$ and E[TWT]). The resulting solutions were observed to be optimal for all the problem instances considered. Moreover, the average CPU time required by SO-Dynasearch for the 15-job instances, in particular, was within 25 s, which yields more than a 14- and 80-fold improvement over the L-shaped method and the direct solution of Problem **MP** by CPLEX, respectively.

Next, we applied SO-Dynasearch to large-sized problems. We were able to solve problem instances involving upto 100 jobs and 800 scenarios within a reasonable amount of CPU time. We present the results for 25 problem instances involving 100 jobs and 800 scenarios in Fig. 3. These problem instances are randomized versions of the first 25 of 125 instances from the OR-Library Beasley (2012) used in Congram et al. (2002), and have RDD = 0.2 and TF = {0.2, 0.4, 0.6, 0.8, 1.0}, with 5 instances for each of the 5 TF values. For each problem instance, the processing time of job

Table 4 Computational performance of the enhanced SO-Dynasearch method for the 100-job single machine problems

Obj.	TF	CPU time (s)		No. of visited neighborhoods	
		Average	Max	Average	Max
CVaR	0.2	4060.2	4410.4	220	238
	0.4	6020.6	7332.7	327	398
	0.6	11751.8	13721.4	650	759
	0.8	10572.9	12517.0	583	689
	1.0	11906.4	13605.0	653	750
$E[TWT]$	0.2	19781.6	22747.7	218	251
	0.4	28787.4	29749.5	317	328
	0.6	35051.6	36445.5	386	401
	0.8	37298.1	38919.0	407	421
	1.0	35677.0	37280.5	396	413

j is uniformly distributed as per $U(\rho_j(1-r_j), \rho_j(1+r_j))$, where ρ_j is the processing time of job j in the deterministic version from the OR-Library and r_j is a sample from $U(0.2, 0.5)$. To deal with these large instances, we further enhanced the SO-Dynasearch by applying Elimination rule 1(a) of Grosso et al. (2004) to the selected scenarios in each optimization step of SO-Dynasearch. With this enhancement, SO-Dynasearch was able to solve these problems with an average CPU time of 8862.4 s for minimizing CVaR $\equiv \phi_{TWT}(0.8)$ and 31319.1 s for minimizing $E[\text{TWT}]$. Based on the solutions obtained, the distribution of TWT was determined for each of the 25 problems. Figure 3 depicts the results obtained. Note that, the optimization of CVaR tends to reduce variance while only slightly increasing the expected value of TWT. For completeness, we have also presented average and maximum values of CPU times and numbers of Dynasearch neigbhorhoods visited for these problem instances in Table 4.

4 Application to a parallel machine scheduling problem

We extend the above discussion to the problem of minimizing CVaR for the total weighted tardiness objective in an identical parallel machine environment. As in the single machine case, we assume that the only uncertain elements are the random processing times. Consider the following notation in addition to that for the single machine case:

Parameters:

M: Set of machines;

Decision variables:

$x_{i,j}$: Job assignment; $x_{i,j} = 1$, if job j is assigned to machine i, and $x_{i,j} = 0$, otherwise, $\forall i \in M, j \in J$;

$z_{j,k}$: Sequencing priority when job j and job k are assigned to the same machine; $z_{j,k} = 1$, if job j precedes job k, and $z_{j,k} = 0$, otherwise, $\forall j \neq k \in J$;

$\gamma^i_{k,j}$: Linearization variable defined as $\gamma^i_{k,j} \equiv z_{k,j}x_{i,k}$.

PM − TWTP :

$$\text{Minimize} \quad \phi = \eta + \frac{1}{1-\alpha}\sum_{s\in S}\pi_s\mu_s$$

$$\text{subject to:} \quad \eta + \mu_s \geq \sum_{j\in J} w_j t^s_j, \qquad \forall s \in S \tag{18}$$

$$\sum_{k\in J\setminus\{j\}} p^s_k \gamma^i_{k,j} + p^s_j x_{i,j} \leq d_j + t^s_j + (\Lambda - d_j)(1 - x_{i,j}), \forall s \in S, i \in M, j \in J \tag{19}$$

$$\sum_{i\in M} x_{i,j} = 1, \quad \forall j \in J \tag{20}$$

$$\sum_{j\in J} j x_{(i-1),j} \geq \sum_{j\in J} j x_{i,j}, \forall i \in M, i > 1 \tag{21}$$

$$z_{j,k} + z_{k,j} = 1, \qquad \forall j \neq k \in J \tag{22}$$

$$z_{j,k} + z_{k,l} + z_{l,j} \leq 2, \qquad \forall j \neq k \neq l \in J \tag{23}$$

$$\sum_{i\in M} i x_{i,k} - \sum_{i\in M} i x_{i,j} \leq (|M| - 1)z_{j,k}, \qquad \forall j \neq k \in J \tag{24}$$

$$\gamma^i_{k,j} + 1 \geq z_{k,j} + x_{i,k}, \qquad \forall i \in M, j \neq k \in J \tag{25}$$

$$x_{i,j} \in \{0,1\}, z_{j,k} \in \{0,1\}, \quad \gamma^i_{k,j} \in [0,1], \qquad \forall i \in M, j \neq k \in J$$

$$t^s_j \geq 0, \mu_s \geq 0, \qquad \forall s \in S, j \in J \tag{26}$$

Constraints (18) and (26) enforce the definitional role for μ_s. Constraints (19) and (26) determine the tardiness of job j when it is assigned to machine i. The parameter Λ is a sufficiently large number to ensure the validity of constraint (19) when $x_{i,j} = 0$; in our implementation, we took Λ as a pre-calculated upper bound value on the makespan. Constraint (20) assigns each job j to a machine in M. Since all machines are identical, constraint (21) serves as symmetry-

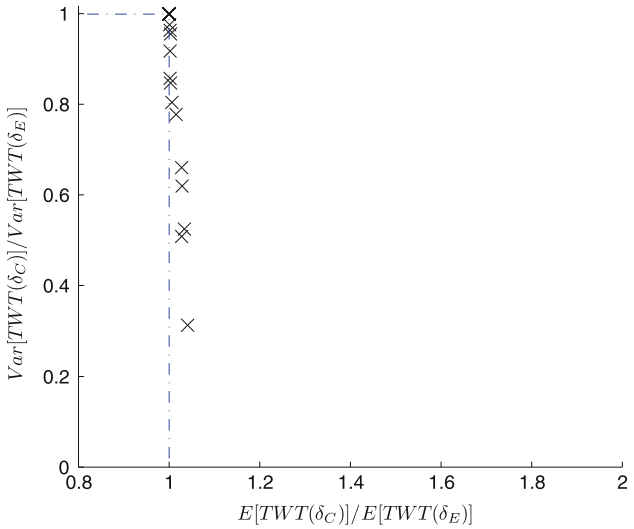

Fig. 4 Expectation-variance comparison of sequences δ_C and δ_E for the 8-job parallel machine problems

breaking constraints, requiring the summation of job indices on a lower-numbered machine to be no less than that on a higher-numbered machine. Constraints (22) and (23) are, respectively, two- and three-city subtour elimination constraints for the sequencing priority variables (see Sarin et al. 2005). Constraint (24) reduces the symmetry caused by the inconsequence of the variables $z_{j,k}$ when job j and job k are assigned to different machines, by requiring a job assigned to a lower-indexed machine to precede the other job (in light of (22)). Constraint (25) enforces the definitional role for $\gamma^i_{k,j}$.

We adapted the 30 instances of the 8-job single machine problem that were solved to obtain part of the results in Table 2, in order to conduct a computational study for the parallel machine case. We assumed two identical machines in parallel, where the job due dates were reduced by half accordingly. For the sake of illustrating the nature of solutions obtained, we simply used CPLEX to obtain optimal solutions for minimizing $\phi_{\text{TWT}}(0.8)$ and $E[\text{TWT}]$. The expectation and variance of TWT were calculated for both resulting solutions, and the related statistics are depicted in Fig. 4. Note that our previous observations with respect to minimizing CVaR versus $E[\text{TWT}]$ carry over to the parallel machine case as well; optimizing CVaR has the potential to reduce variability significantly, with only a slight increase in the expected value of TWT.

5 Conclusions and future research

The use of CVaR as an objective criterion appears to be quite promising to simultaneously control both the expected value and the variability of different performance measures in a stochastic scheduling environment. We have demonstrated the effectiveness of this strategy for the single machine and parallel machine total weighted tardiness (TWT) problems, where the variance in TWT was shown to be significantly reduced with only a marginal increase in the expected value in comparison with minimizing $E[\text{TWT}]$. The application of this concept and approach for other performance measures and in alternative scheduling environments are important topics for future study.

Acknowledgments This Research has been supported by the National Science Foundation under Grant CMMI-0856270.

References

Artzner, P., Delbaen, F., Eber, J., & Heath, D. (1999). Coherent measures of risk. *Mathematical Finance*, *9*, 203–228.

Balasubramanian, J., & Grossmann, I. E. (2003). Scheduling optimization under uncertainty—an alternative approach. *Computers & Chemical Engineering*, *27*(4), 469–490.

Beasley, J. E. (2012). OR-Library: Weighted tardiness. http://people.brunel.ac.uk/mastjjb/jeb/orlib/wtinfo.html. Accessed 3 June 2012.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, *4*(1), 238–252.

Birge, J. R., & Louveaux, F. (1997). *Introduction to stochastic programming*. New York: Springer.

Congram, R. K., Potts, C. N., & van de Velde, S. L. (2002). An iterated dynasearch algorithm for the single-machine total weighted tardiness scheduling problem. *INFORMS Journal on Computing*, *14*(1), 52–67.

Daniels, R. L., & Kouvelis, P. (1995). Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management Science*, *41*(2), 363–376.

De, P., Ghosh, J. B., & Wells, C. E. (1992). Expectation-variance analyss of job sequences under processing time uncertainty. *International Journal of Production Economics*, *28*(3), 289–297.

Grosso, A., Croce, F. D., & Tadei, R. (2004). An enhanced dynasearch neighborhood for the single-machine total weighted tardiness scheduling problem. *Operations Research Letters*, *32*(1), 68–72.

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

McDaniel, D., & Devine, M. (1977). A modified Benders' partitioning algorithm for mixed integer programming. *Management Science*, *24*(3), 312–319.

McKay, K. N., Safayeni, F. R., & Buzacott, J. A. (1988). Job-Shop scheduling theory: What is relevant? *Interfaces*, *18*(4), 84–90.

Ogryczak, W., & Ruszczynski, A. (2002). Dual stochastic dominance and related mean-risk models. *SIAM Journal on Optimization*, *13*(1), 60–78.

Pinedo, M. (2001). *Scheduling: Theory, algorithms, and systems* (2nd ed.). Upper Saddle, NJ: Prentice Hall.

Porter, R. B., & Gaumnitz, J. E. (1972). Stochastic dominance vs. mean-variance portfolio analysis: An empirical evaluation. *The American Economic Review*, *62*(3), 438–446.

Rockafellar, R. T., & Uryasev, S. (2000). Optimization of conditional value-at-risk. *The Journal of Risk*, *2*(3), 21–41.

Sabuncuoglu, I., & Bayiz, M. (2000). Analysis of reactive scheduling problems in a job shop environment. *European Journal of Operational Research*, *126*(3), 567–586.

Sarin, S., Sherali, H., & Bhootra, A. (2005). New tighter polynomial length formulations for the asymmetric traveling salesman problem with and without precedence constraints. *Operations Research Letters*, *33*(1), 62–70.

Sarin, S. C., Nagarajan, B., & Liao, L. (2010). *Stochastic scheduling: Expectation-variance analysis of a schedule* (1st ed.). New York: Cambridge University Press.

Sherali, H., Lunday, B. (2010). On generating maximal nondominated Benders cuts. Annals of Operations Research (in press).

Skutella, M., & Uetz, M. (2005). Stochastic machine scheduling with precedence constraints. *SIAM Journal on Computing*, *34*(4), 788.

Vieira, G. E., Herrmann, J. W., & Lin, E. (2003). Rescheduling manufacturing systems: A framework of strategies, policies, and methods. *Journal of Scheduling*, *6*(1), 39–62.

Wang, W., & Ahmed, S. (2008). Sample average approximation of expected value constrained stochastic programs. *Operations Research Letters*, *36*(5), 515–519.