ELSEVIER European Journal of Operational Research 113 (1999) 575–592

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

Theory and Methodology

Scheduling of design projects with uncertain number of iterations

Peter B. Luh [a], Feng Liu [a,*], Bryan Moser [b,1]

[a] Department of Electrical and Systems Engineering, University of Connecticut, Storrs, CT 06269-2157, USA
[b] United Technologies Research Center, University of Tokyo, Tokyo, Japan

Received 2 April 1997; accepted 3 December 1997

Abstract

A short product design cycle is critical to the success of companies in the era of time-based competition. The underlying design activities, however, are often interlinked and quite uncertain. For example, some activities may have to be iterated several times to meet the design criteria. Furthermore, time-critical projects suffer the risk of failure if they cannot meet established target dates. Generating good and robust schedules is thus critical, especially under the concurrent engineering paradigm where the delay of a single task may have a domino effect on subsequent tasks and on other projects sharing designers and/or resources. This paper studies the scheduling of design projects with uncertain number of iterations while managing design risks. A "separable" problem formulation that balances modeling accuracy and computation complexity is created with the goal to minimize project tardiness and risk penalties. An optimization-based methodology that combines Lagrangian relaxation, stochastic dynamic programming, and "ordinal optimization" is developed. Numerical results supported by simulation demonstrate that near optimal solutions are obtained, and uncertainties are effectively managed for problems of practical sizes.

Keywords: Scheduling theory; Design project management; Integer programming; Stochastic dynamic programming; Risk management

1. Introduction

1.1. Managing design projects with uncertain number of iterations

A short lead time for product development is critical in the era of time-based competition. Development time determines how responsive a company can be to competitive forces and how quickly the company receives the economic returns from the development efforts. The importance of time is well illustrated in a study that if a product suffers from a 50% over expenditure in product development, the loss of total recoverable profit is 4%. However, if the product is late to market by 6 months for a life cycle of 5 years, it can lose one third of its profit (Nichols, 1990). The completion of design projects is thus required to be on-time and predictable. The underlying design activities,

* Corresponding author. Fax: +1 860 486 5585; e-mail: liuf@brc.uconn.edu.
[1] E-mail: moserbr@utrc.utc.com.

0377-2217/99/$ – see front matter
PII: S 0 3 7 7 - 2 2 1 7 (9 8) 0 0 0 2 7 - 7

however, are often interlinked and quite uncertain. For example, iterations of a task (or a sequence of tasks) may occur when the results of the task(s) fail to meet specified criteria, or when new information from other tasks is obtained prompting changes on the design (Nukala et al., 1995). Furthermore, time-critical projects suffer the risk of failure if they cannot meet established target dates. Uncertain number of iterations and risks often introduces major uncertainties on the commitment of designers and resources and on project completion. With concurrent engineering principles widely used to cut short lead time resulting in tightly coupled tasks, the delay of a single task may have a domino effect on subsequent tasks and on other projects sharing designers and/or resources. Generating good and robust schedules and reducing the adverse consequences of risks are thus becoming critical. Effectively scheduling multiple projects and managing risks, however, have been proved to be extremely difficult because of the combinatorial nature of the problem and the presence of uncertainties.

1.2. Literature review

After the introduction of network scheduling techniques of PERT and CPM for project planning without the consideration of resource capacities (Program Evaluation and Review Technique (PERT) and Critical Path Method (CPM), see, e.g., Elmaghraby (1977)), many efforts have been on expanding these techniques to handle finite resource capacities. Early attempts concentrated on the formulation and resolution of mathematical (usually integer) optimization problems. Since it was discovered that these problems were a generalization of the well-known job-shop scheduling problems and as such is NP-hard (Blazewicz et al., 1983), major efforts shifted towards the development of heuristic procedures for obtaining "satisfying" solutions (Davis and Patterson, 1975; Patterson, 1984; Boctor, 1990). Efforts on the improvement of optimization methods for solving certain variants of problems continue, including branch-and-bound (Christofides et al., 1987; Demeulemeester and Herroelen, 1996). Most results reported were on the scheduling of a single project having less than 50 activities and using about 5 types of resources.

To deal with more general projects having probabilistic routings and repetition of activities via feedback loops, stochastic project networks have been introduced. They are generalized PERT networks (also called Graphic Evaluation and Review Technique (GERT) networks), and most problems with GERT precedence constraints are NP-hard except for a few specialized cases with a single-machine (Neumann, 1990).

For the management of design projects, design structure matrices (DSM) have often been used to analyze the technical relationships among design tasks. In a DSM, a task is assigned to a row and a column in the identical order. A row, corresponding to one task, is annotated with marks indicating those tasks (columns) on which it depends. The relationships represented by a DSM thus define the technical structure of a project, and can be used to find alternative sequences of tasks. The DSM is also useful for identifying where iteration is necessary, and to predict slow and rapid convergence of iterations (Eppinger et al., 1994; Smith and Eppinger, 1995). Most results using DSM were obtained ignoring resource capacities.

Risk analysis models and methods for project management have been presented in Cooper and Chapman (1987). Recently, IDEF3 models have been used for risk assessment in concurrent design (Larson and Kusiak, 1996). Because of the difficulty of scheduling problems, not many results have been obtained for simultaneously scheduling projects and managing risks.

Much progress has recently been made on deterministic manufacturing scheduling problems. In the work of Luh and Hoitomt (1993), a "separable" integer programming formulation for job shops was created, and an optimization-based methodology was developed by using Lagrangian relaxation (LR) to exploit the separability of the problem structure. Near optimal schedules are efficiently generated for practical size problems with quantifiable quality. Following this approach, a "forward" dynamic programming algorithm was embedded within the LR framework in Chen et al. (1995) and Wang and Luh (1996) to alleviate convergence difficulties as reported in Czerwinski and Luh (1994). However, not many optimization-based results have been reported for stochastic scheduling problems with more than two machines, since the problems are extremely difficult Pinedo (1982, 1995). To avoid the difficulties, an intuitive approach is to replace random variables by their means, consequently converting a problem into a deterministic one (Pinedo, 1995), and referred to as the "mean" method in this paper. Previous deterministic methods can then be applied, however, the performance of this approach may not be good (Federgruen and Mosheiov, 1997).

1.3. Scope of this paper

This paper considers the scheduling of design projects while managing design risks for an organization that pursues multiple projects concurrently with a finite number of shared designers and resources. It is assumed that a design project is divided into subprojects based on overall product design strategy. A subproject is further broken down into inter-related tasks, with the constitution and precedence relationship of tasks given, and risk factors identified. A task may simultaneously require multiple designers of distinct capabilities and resources of different types, and some tasks (or sequences of tasks) may have to go through an uncertain number of iterations for completion where the time needed to perform each iteration is assumed deterministic and given. Multiple tasks and subprojects within a project may also be performed in parallel, subject to precedence constraints as required by the technical structure to receive design information and/or materials from preceding tasks and/or subprojects. Our goal is to develop a good problem formulation and an efficient methodology to generate near-optimal schedules with quantifiable quality for problems of realistic sizes. The objective is to maximize expected on-time completion while minimizing project risk penalties, subject to precedence constraints and expected designer/resource capacity constraints. The separable problem formulation is presented in Section 2.

A combined Lagrangian relaxation (LR) and stochastic dynamic programming (SDP) method is developed in Section 3. The problem is first relaxed and decomposed into subproblems, one for each subproject. A subproblem is solved by using SDP, where a stage corresponds to one iteration of a task and state transitions governed by probabilities and scheduling decisions. The close-loop nature of SDP is fully exploited so that precedence constraints are satisfied for each possible number of iterations, and the complexity is only slightly higher than the one without uncertainty. A dual solution is selected by using "ordinal optimization" which saves simulation efforts dramatically (Ho, 1995; Chen, 1995), and schedules are dynamically constructed by using heuristics based on the dual solution selected and the realizations of random events.

Testing results reported in Section 4 demonstrate that by satisfying iteration and task precedence constraints for each possible number of iterations, uncertainties are effectively handled. The expected resource capacity constraints reduce computational requirements without much loss of scheduling performance, enabling the method to solve problems of practical sizes. Furthermore, project risk penalties are low, indicating that risks are well managed.

2. Problem formulation

General description. In this section, an integer optimization problem is formulated based on what was presented in Czerwinski and Luh (1994) for deterministic job-shop scheduling with the following new features: simultaneous use of multiple types of resources, uncertain number of iterations, and design risks. A list of symbols is provided in Appendix A for easy reference.

As mentioned earlier, a task may simultaneously require designers of distinct capabilities *and* resources of different types. For simplicity of presentation, both designers and resources are considered as *generic resources* with given functionality. There are H resource types, each consisting of one or multiple units of identical functionality from the scheduling viewpoint. The number of type h resource $(1 \leqslant h \leqslant H)$ available at discrete time k $(0 \leqslant k \leqslant K-1)$ is assumed given and denoted as M_{kh}.

There are P design projects, Project p $(1 \leqslant p \leqslant P)$ has given discrete due date d_p and contains S_p subprojects. Subproject i of Project p consists of a *series* of tasks, and the set of these tasks is denoted as S_{pi}. Among all subprojects of Project p, there are total N_p tasks, and task j of Project p is denoted as (p, j). Task (p, j) simultaneously requires m_{pjh} units of type h resource for all h belonging to the "resource set" h_{pj} ("multiple resource constraints" in the scheduling literature, Gargeya and Deane, 1996). Sometimes it is possible for task (p, j) to be performed by a different resource set, and the set of all "eligible resource sets" is denoted as H_{pj}.

In a project, multiple tasks and subprojects can be performed in parallel, subject to precedence constraints. A particular task of a subproject may be required to be "assembled" with tasks in other subprojects, e.g., tasks $(p, 5)$ and $(p, 8)$ in Fig. 1. In this case, these two tasks must be completed before the subsequent task $(p, 9)$ can be started. Furthermore, the design result of a particular task, e.g., $(p, 3)$ in Fig. 1, may be needed by several tasks in other subprojects. This task $(p, 3)$ should thus be completed before the beginning of tasks $(p, 4)$ and $(p, 6)$. It is assumed without loss of generality that a project begins and terminates with a single subproject, subproject 1 and subproject S_p, respectively.

Uncertain number of design iterations. Some tasks (or sequences of tasks) may need to be repeated several times to satisfy the design criteria, and these tasks are called "uncertain tasks". For simplicity of presentation but without loss of generality, we shall consider the case where a task, rather than a sequence of tasks, may have to go through an uncertain number of iterations. To model the repetition of such a task, an independent random variable is used at the end of an iteration to determine whether to repeat the same task with a given probability, or to move on to the next task. The nth iteration of an uncertain task (p, j) is denoted as (p, j, n), and can be performed by an eligible resource set h^n_{pj} belonging to H_{pj} for a specific period of time t^n_{pjh}, where t^n_{pjh} is a given deterministic integer. At the end of (p, j, n), the probability of going to $(p, j, n+1)$ is denoted as P^n_{pj}. Since the number of iterations is usually small as limited by due date and cost considerations, the maximum allowable iteration number N_{pj} can be estimated and is given.

Tasks without uncertain number of iterations are called "deterministic tasks", and such a task requires a single iteration. For simplicity of presentation, deterministic task (p, j) will be denoted as iteration $(p, j, 1)$ with N_{pj} equal to 1.

Design risk. As mentioned earlier, some time-critical projects may fail and be dropped out of consideration if they cannot meet established target dates. Such a risk is captured when Project p cannot be completed by a given "absolute deadline" $\bar{d}_p$ (called "time risk"), or when a specific uncertain task (p, j) cannot meet the design criteria within N_{pj} iterations (called "iteration risk"). When Project p fails for either case, a "risk penalty" R_p is incurred. This penalty depends on the importance of the project (opportunity cost $\bar{R}_p$) and the status upon failure (cost foregone). It is assumed that the cost foregone consists of costs

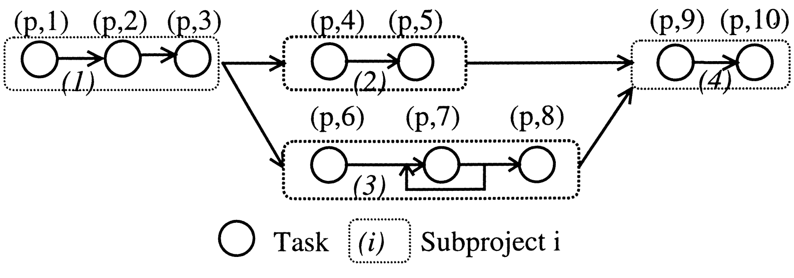


Fig. 1. A simplified design project.

incurred on subprojects, one for each subproject. Such a cost is denoted as R_{pj} if subproject i is at task (p,j) upon failure for $j \in S_{pi}$, or is zero if the subproject has not yet been started. The risk penalty R_p is thus the sum of $\bar{R}_p$ and these R_{pj}. In reality, $\bar{R}_p$ and R_{pj} can often be estimated, and are given here.

Precedence constraints. For an uncertain task (p,j), the iteration precedence constraints state that the $(n+1)$th iteration cannot be started before the completion of the nth iteration, i.e.,

$$c^n_{pj} + 1 \leqslant b^{n+1}_{pj}, \quad p = 1,\ldots,P;\ j = 1,\ldots,N_p; \quad n = 1,\ldots,n_{pj} - 1, \tag{1}$$

where c^n_{pj} is the completion time of (p,j,n), and b^{n+1}_{pj} the beginning time of $(p,j,n+1)$. The beginning time of task (p,j) is the beginning time of the first iteration, i.e., b^1_{pj}, and completion time c_{pj} is the completion time of the last iteration, i.e., $c^{n_{pj}}_{pj}$, if n_{pj} iterations occur.

Task and subproject precedence constraints state that a particular task (p,j) must be completed before the beginning of its immediately succeeding task (p,r), i.e.,

$$c_{pj} + 1 \leqslant b^1_{pr}, \quad j \in S_{pi},\ r \in S_{pq};\ (p,r) \in I_{pj}, \tag{2a}$$

where I_{pj} denotes the set of tasks of project p immediately following (p,j). Constraint (2a) is called subproject precedence constraint if tasks (p,j) and (p,r) belong to different subprojects, i.e., if $i \neq q$, and called task precedence constraint otherwise. Without loss of generality, it is assumed that $r > j$ if $q \geqslant i$, and specially $r = j + 1$ if $q = i$.

Iteration and task precedence constraints are required to be satisfied for every possible number of iterations to accurately model the uncertainties. Since tasks in different subprojects generally are less related than those within the same subproject, subproject constraints are required to be satisfied in the expected sense in the model to reduce solution complexity, and to reflect the common practice in coordinating less related uncertain activities, i.e.,

$$\mathrm{E}(c_{pj}) + 1 \leqslant \mathrm{E}(b^1_{pr}), \quad j \in S_{pi},\ r \in S_{pq},\ i \neq q; \quad (p,r) \in I_{pj}. \tag{2b}$$

Resource capacity constraints. Resource capacity constraints state that the total number of type h resource allocated to tasks at time k should not exceed M_{kh}, the number of the resource available at that time, i.e.,

$$\sum_{pjn} m_{pjh} \delta^n_{pjkh} \leqslant M_{kh}, \quad n = 1,\ldots,n_{pj}; \quad k = 0,\ldots,K-1;\ h \in H, \tag{3a}$$

where δ^n_{pjkh} equals 1 if type h resource is assigned to the nth iteration of task (p,j) at time k, and 0 otherwise. The simultaneous use of multiple resource types by (p,j) is captured through m_{pjh} by having $m_{pjh} > 0$ for multiple h's. The parameter m_{pjh} can be a fractional number to reflect the fact that the resource may not be needed 100% of time during processing. Since it is very difficult to handle resource capacity constraints mathematically for all possible numbers of iterations, constraints (3a) are also required to be satisfied in the expected sense, i.e.,

$$\mathrm{E}\left(\sum_{pjn} m_{pjh} \delta^n_{pjkh}\right) \leqslant M_{kh}, \quad n = 1,\ldots,n_{pj}; \quad k = 0,\ldots,K-1;\ h \in H. \tag{3b}$$

Processing time requirements. Processing time requirements state that the nth iteration of task (p,j) will be performed by an eligible resource set h^n_{pj} for a specific period of time, i.e.,

$$c^n_{pj} = b^n_{pj} + t^n_{pjh} - 1, \quad p = 1,\ldots,P; \quad j = 1,\ldots,N_p;\ n = 1,\ldots,n_{pj}. \tag{4}$$

Objective function. The objective of scheduling is to meet on-time project completion while discouraging starting earlier than necessary, and to reduce project failures. The problem is thus to minimize the sum of expected weighted tardiness, earliness, and risk penalties by selecting appropriate task or iteration beginning times b^n_{pj} and resource sets h^n_{pj}, i.e.,

$$\min_{\{b^n_{pj}, h^n_{pj}\}} J, \quad \text{with}$$

$$J = \sum_{p=1}^{P} \mathrm{E}[\bar{\Delta}_p(w_p T_p^2 + \beta_p E_p^2) + (1 - \bar{\Delta}_p) R_p], \tag{5}$$

subject to constraints (1), (2a), (2b), (3b) and (4). In the above, $\bar{\Delta}_p$ is one when Project p does not fail, and zero otherwise. Tardiness T_p is the amount of time Project p overdue, i.e., $\max(0, c_p - d_p)$, with $c_p = c_{pN_p}$ being the project completion time. For a given project due date d_p, a desired project start date $\bar{b}_p$ can be roughly estimated based on the project critical path (Czerwinski and Luh, 1994). Earliness, E_p, is then defined as the amount the project beginning time, $b_p = b_{p1}^1$, leads the desired start date $\bar{b}_p$, i.e., $\max(0, \bar{b}_p - b_p)$. The square on tardiness reflects the fact that a project becomes more critical with each time unit after passing its due date (Hoitomt et al., 1993), and similarly for the square on earliness. Parameters w_p and β_p are given weights associated with tardiness and earliness penalties, accounting for the importance of meeting on-time completion and start. Since on-time completion is the foremost criterion in Eq. (5), β_p is an order of magnitude smaller than w_p. Risk penalty R_p represents the lost opportunity and the cost incurred as explained earlier. The expectation is taken with respect to the random number of iterations and random decision variables.

A schedule satisfying Eqs. (1), (2a), (2b)–(4) is called "model feasible", and a schedule satisfying Eqs. (1)–(4) is called "implementable". A model feasible schedule is usually not implementable since constraints (2b) and (3b) are satisfied in the expected rather than exact sense.

Since Eqs. (1), (2a) and (4) are linear, and Eqs. (2b), (3b) and (5) are additive in terms of decision variables b_{pj}^n and h_{pj}^n, the formulation is "separable" down to the subproject level. Lagrangian relaxation technique can then be effectively applied.

3. Solution methodology

Similar to pricing concept of a market economy, the LR method replaces "hard" coupling constraints (subproject precedence constraints and resource capacity constraints) by "soft" prices (Lagrange multipliers) for the violation of precedence constraints and the use of resources at each time. The original problem can thus be decomposed into smaller and easier subproject subproblems. The individual subproblems are solved by using SDP. Those prices or multipliers are then iteratively adjusted based on the degree of constraint violations following again the market economy mechanism. Subproblems thus are resolved using the new set of multipliers. In mathematical terms, a "dual function" is maximized in this multiplier updating process. Since a good dual solution may not necessarily be associated with a good implementable schedule, a dual solution is selected by using "ordinal optimization". At the termination of the multiplier updating iterations, on-line heuristics are applied to adjust the dual solution selected to remove any infeasibilities and dynamically construct an implementable schedule based on the realizations of random events.

3.1. The Lagrangian relaxation framework

By using Lagrange multipliers η_{pjr} to relax subproject precedence constraints (2b) and using multipliers π_{kh} to relax resource capacity constraints (3b), the following relaxed problem is obtained:

$$\min_{\{b_{pj}^n, h_{pj}^n\}} J, \quad \text{with}$$

$$\begin{aligned} L = \mathrm{E}\Bigg[&\sum_p \Bigg(\bar{\Delta}_p(w_p T_p^2 + \beta_p E_p^2) + (1 - \bar{\Delta}_p)R_p \\ &+ \sum_{jnkh} m_{pjh}\pi_{kh}\delta_{pjkh}^n\Bigg) - \sum_{kh}\pi_{kh}M_{kh}\Bigg] \\ &+ \mathrm{E}\Bigg[\sum_{p,j,(p,r)\in I_{pj}} \eta_{pjr}\Big(c_{pj} + 1 - b_{pr}^1\Big)\Bigg], \end{aligned} \tag{6}$$

subject to iteration and task precedence constraints (1) and (2a), and processing time requirements Eq. (4). By regrouping relevant terms, the relaxed problem can be decomposed into the following subproject subproblems:

$$\min_{\{b^n_{pj}, h^n_{pj}\}} L_{pi}, \quad \text{with}$$

$$\begin{aligned} L_{pi} = \mathrm{E}\Bigg[& \bar{\Delta}_p(w_p T_p^2 \Delta_{pS_p} + \beta_p E_p^2 \Delta_{p1}) \\ & + (1-\bar{\Delta}_p)(\Delta_{p1}\bar{R}_p + \bar{\Delta}_{pj} R_{pj}) \\ & + \sum_{j \in S_{pi}} \sum_{n=1}^{n_{pj}} \sum_{h \in H_{pj}} m_{pjh}\pi_{kh}\delta^n_{pjkh} \\ & + \sum_{j,(p,r)\in I_{pj}} \eta_{pjr}(c_{pj}+1) - \sum_{r:(p,j)\in I_{pr}} \eta_{prj} b^1_{pj} \Bigg], \quad j \in S_{pi}, \end{aligned} \tag{7}$$

subject to Eqs. (1), (2a) and (4). In the above, Δ_{p1} is an integer variable equal to one if subproject i is the first subproject of p, i.e., $i = 1$, and zero otherwise, and Δ_{pS_p} is similarly defined for the ending subproject S_p of p. The integer variable $\bar{\Delta}_{pj}$ equals to one if subproject i is at (p, j) upon failure, and zero otherwise.

Let L^*_{pi} denote the resulting minimal subproblem cost. The high level dual problem is then obtained as

$$\max_{\pi,\eta \geqslant 0} L(\pi,\eta), \quad \text{with}$$

$$L(\pi,\eta) = \sum_{pi} L^*_{pi} - \sum_{kh} \pi_{kh} M_{kh}. \tag{8}$$

3.2. Dynamic programming for solving subproblems

The forward dynamic programming algorithm presented by Chen et al. (1995) can be used to solve the decomposed subproblems in the deterministic case. In this paper, a *backward* SDP is developed to solve subproject subproblems Eq. (7) to manage uncertainties. In the procedure, an SDP stage corresponds to one iteration of a task, and at each stage, the states (or nodes) are the possible iteration beginning times. To clearly illustrate the SDP procedure for a subproject, we shall assume that Project p has only one subproject, there is no project earliness penalty, and each task can only be performed by a single eligible resource set. The SDP algorithm starts by calculating terminal costs at the last stage, then moving backwards to recursively compute optimal cumulative costs at preceding stages, and finally obtaining subproblem solution at the first stage. Project risk penalties are imbedded within SDP cost to manage risks.

Calculating terminal cost. Without loss of generality, it is assumed that the last task (p, N_p) is deterministic. The DP algorithm starts with the last stage having the following terminal cost:

$$V^1_{pN_p}(b^1_{pN_p}) = w_p T_p^2 + \sum_{k=b^1_{pN_p}}^{c^1_{pN_p}} \sum_{h \in H_{pN_p}} m_{pN_p h}\pi_{kh}. \tag{9}$$

Calculating optimal cumulative cost and managing iteration risks. When moving backwards, the optimal cumulative cost V^n_{pj} of stage (p, j, n) at state b^n_{pj} is obtained recursively subject to iteration and task precedence constraints. Deterministic and uncertain tasks are separately considered in the following.

Stage representing a deterministic task. At such a stage $(p, j, 1)$, the optimal cumulative cost at state b^1_{pj} is obtained as

$$\begin{aligned} V^1_{pj}(b^1_{pj}) &= \min_{b^1_{p,j+1}} \left[\sum_{k=b^1_{pj}}^{c^1_{pj}} \sum_{h \in H_{pj}} m_{pjh}\pi_{kh} + V^1_{p,j+1}(b^1_{p,j+1}) \right] \\ &= \sum_{k=b^1_{pj}}^{c^1_{pj}} \sum_{h \in H_{pj}} m_{pjh}\pi_{kh} + \min_{b^1_{p,j+1}} V^1_{p,j+1}(b^1_{p,j+1}), \\ & \quad 1 \leqslant j \leqslant N_p - 1, \end{aligned} \tag{10}$$

where $b^1_{p,j+1}$ is the decision variable. The second equality in Eq. (10) is obtained since the first term is a fixed value for the given b^1_{pj}.

Stages representing iterations of an uncertain task. For a stage $(p, j, n), n < N_{pj}$, two cases need to be considered: move to $(p, j, n+1)$ of the same task with probability P^n_{pj}, or move to the next task $(p, j+1, 1)$ with probability $1 - P^n_{pj}$. The optimal expected cumulative cost at state b^n_{pj} is thus calculated as

$$\begin{aligned} V^n_{pj}(b^n_{pj}) = & \sum_{k=b^n_{pj}}^{c^n_{pj}} \sum_{h \in H_{pj}} m_{pjh}\pi_{kh} + P^n_{pj} \min_{b^{n+1}_{pj}} V^{n+1}_{pj}(b^{n+1}_{pj}) \\ & + (1 - P^n_{pj}) \min_{b^1_{p,j+1}} V^1_{p,j+1}(b^1_{p,j+1}), \\ & 1 \leqslant j \leqslant N_p - 1; \ 1 \leqslant n \leqslant N_{pj} - 1. \end{aligned} \tag{11}$$

Minimum $V_{pj}^{n+1}(b_{pj}^{n+1})$ and minimum $V_{p,j+1}^{1}(b_{p,j+1}^{1})$ can be obtained by selecting b_{pj}^{n+1} and $b_{p,j+1}^{1}$, respectively, as in Eq. (10).

At stage (p, j, N_{pj}), two different cases need to be considered: the project fails with probability $P_{pj}^{N_{pj}}$ as the specified number of iterations has been exceeded (iteration risk), or moves to $(p, j+1, 1)$ of the next task with probability $1 - P_{pj}^{N_{pj}}$. The optimal expected cumulative cost is thus calculated as

$$V_{pj}^{N_{pj}}(b_{pj}^{N_{pj}}) = \sum_{k=b_{pj}^{N_{pj}}}^{c_{pj}^{N_{pj}}} \sum_{h \in H_{pN_{pj}}} m_{pjh}\pi_{kh} + P_{pj}^{N_{pj}}(\bar{R}_p + R_{pj}) + (1 - P_{pj}^{N_{pj}}) \min_{b_{p,j+1}^{1}} V_{p,j+1}^{1}(b_{p,j+1}^{1}), \quad 1 \leqslant j \leqslant N-1, \tag{12}$$

where $(\bar{R}_p + R_{pj})$ is the risk penalty. When $P_{pj}^{N_{pj}}$ is zero, i.e., no iteration risk, the right-hand side of Eq. (12) is similar to that of Eq. (10).

Managing time risks. A time-critical Project p fails at iteration (p, j, n) of either a deterministic or an uncertain task if this iteration cannot be completed by $\bar{d}_{pj}^{n}$, the "absolute iteration deadline" derived from $\bar{d}_p$ based on project critical paths. For states with c_{pj}^{n} greater than $\bar{d}_{pj}^{n}$, optimal cumulative costs are thus given by

$$V_{pj}^{n}(b_{pj}^{n}) = \bar{R}_p + R_{pj}, \quad b_{pj}^{n} + t_{pjh}^{n} - 1 > \bar{d}_{pj}^{n}. \tag{13}$$

For other states, $V_{pj}^{n}(b_{pj}^{n})$ is calculated by using one of the above formulas (10), (11), or (12).

From Eqs. (9)–(13), it can be seen that risks are managed within SDP by appropriately trading off risk penalties vs. resource utilization costs and project tardiness penalties.

Subproblem solution. The optimal L_{pi}^{*} is obtained as the minimum optimal cumulative cost at the first stage. Finally, optimal iteration and task beginning times and corresponding resource sets can be obtained by tracing forwards the stages.

Example (SDP Procedure for solving a subproblem). In this example, there are three different resources available over a planning horizon of 7 time units. Project p has only one subproject with due date 5 and without risks. Task $(p, 2)$ may need only one iteration (50%), or may require a second iteration (50%). Other data are shown in Table 1, with multipliers π_{kh} given in Table 2. The state transition diagram for the SDP procedure is shown in Fig. 2.

To satisfy precedence constraints ((1) and (2a)) and processing time requirements (4), only the shaded nodes in Fig. 2 need to be considered. At stage 4, the terminal costs for nodes 3, 4 and 5 can be calculated by Eq. (9), and are shown next to the nodes. At stage 3, consider node 3 for example. Since constraints (2a) must be satisfied, only nodes 4 and 5 can be selected at stage 4, with the smaller terminal cost 7.6 at node 4. The optimal cumulative cost of node 3 at stage 3 thus is obtained as 17.6 by Eq. (12). Similarly, the optimal cumulative cost for node 4 at stage 3 can be calculated. At stage 2, consider node 1 for example. The node may go to stage 3 (50%) subject to constraints (1) or may go to stage 4 (50%) subject to constraints (2a), and the optimal expected cumulative cost 15.05 is obtained from Eq. (11) by selecting nodes 4 at stage 3 and stage 4, respectively. At stage 1, the optimal cumulative costs for node 0 and 1 are calculated by Eq. (10), and are shown in Fig. 2. The optimal L_{p1}^{*} is selected as the minimum optimal cumulative cost 26.05 at stage 1, and the op-

Table 1
Data of the SDP example

Task	Resource needed	t_{pjh}^{n}
(p,1)	Resource 1 and resource 3	1
(p,2,1)	Resource 1	2
(p,2,2)	Resource 1	1
(p,3)	Resource 2 and 50% resource 3	2

Table 2
Multiplier π_{kh} of the SDP example

π_{10}	π_{11}	π_{12}	π_{13}	π_{14}	π_{15}	π_{16}
2.0	2.0	2.0	10.0	5.0	1.0	0.0
π_{20}	π_{21}	π_{22}	π_{23}	π_{24}	π_{25}	π_{26}
−1.0	0.0	2.0	3.1	3.1	5.0	4.0
π_{30}	π_{31}	π_{32}	π_{33}	π_{34}	π_{35}	π_{36}
9.0	10.0	9.0	3.0	0.0	−1.0	−1.0

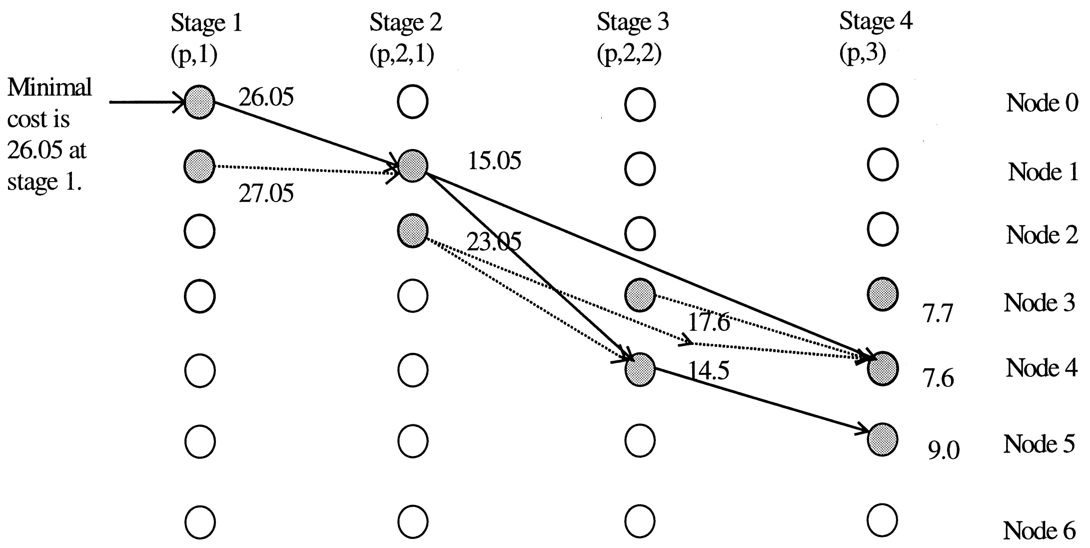


Fig. 2. SDP for subproject 1 of project p with uncertain number of iterations. Solid lines – optimal SDP paths; dotted lines – SDP paths.

timal b_{p1}^{1*} is thus 0. Finally, $b_{p2}^{1*} = 1$ and $b_{p3}^{1*} = 4$ when task $(p, 2)$ needs only one iteration; and $b_{p2}^{1*} = 1$, $b_{p2}^{2*} = 4$, $b_{p3}^{1*} = 5$ if $(p, 2)$ requires the second iteration.

SDP for the general case and complexity. For the general case where a project has earliness penalty and multiple subprojects, and a task can be performed by multiple eligible resource sets, the SDP procedure presented before can be extended as follows. At the last stage, terminal cost will be calculated for each eligible resource set. Moving backwards to a preceding stage, the optimal cumulative cost for using an eligible resource set at a state will be computed by selecting the beginning time and resource set of the next iteration and/or next task. For the first stages of projects' first subproject and the stages with subproject precedence constraints, the earliness penalty and the cost for violating those subproject precedence constraints will be added to the SDP cost, respectively.

The complexity of this SDP algorithm is $O(K \sum_{j \in S_{pi}} N_{pj} |H_{pj}|)$ for subproject i which has $\sum_{j=1}^{|S_{pi}|} N_{pj}$ stages, where $|S_{pi}|$ and $|H_{pj}|$ are the cardinalities of S_{pi} and H_{pj}, respectively. This complexity is slightly higher than that for the deterministic case, i.e., $O(K \sum_{j \in S_{pi}} |H_{pj}|)$ (Chen et al., 1995).

3.3. Solving the dual problem

Among exiting methods for solving the high-level dual problem, the subgradient method is commonly used to iteratively update the Lagrange multipliers. Since the subgradient method requires minimizing all subproblems before each update of multipliers, solving subproblems becomes very time consuming for large problems. To overcome this, an interleaved subgradient method (ISG) was developed by Kaskavelis and Caramanis (1995) with the proof of convergence provided by Zhao et al. (1997). The ISG method updates multipliers after solving each subproblem, and converges faster than the subgradient method especially for large problems. In this paper, the ISG method is used to solve the dual problem (8). To update the multipliers, their subgradients $[(\mathrm{E} \sum_{pjn} m_{pjh} \delta_{pjkh}^{n}) - M_{kh}]_{K \times H}$ and $[\mathrm{E}(c_{pj}) - \mathrm{E}(b_{pr}^{1})]_{Q \times 1}$ are needed, where Q is number of η_{pjr} multipliers. These subgradients can be calculated from the optimal SDP paths.

The overall solution is of *semi-close-loop* nature since the "close-loop" solution of the stochastic subproblems are obtained by using SDP with given (*static*) Lagrangian multipliers.

3.4. Selecting a dual solution and implementing the schedule on line

The iterative multiplier updating process is stopped after a fixed amount of computation time or after a fixed number of multiplier updating iterations have been executed. Since capacity constraints (3b) and subproject precedence constraints (2b) are relaxed when solving subproblems, subproblem solutions may not be model feasible when put together. Furthermore, a model feasible schedule may not be implementable since the constraints (2b) and (3b) are approximations. A heuristic procedure is developed to obtain an implementable schedule on line based on subproblem solutions and the realizations of random events. How to select a good dual solution is presented first.

Selecting dual solution. In view of the heuristic procedure to construct feasible schedules, a dual solution with a high dual cost may not necessarily associate with a good feasible schedule. One therefore has to try out several candidate dual solutions having high dual costs to find which one generates a good feasible schedule. In the stochastic setting, each dual or feasible solution is in fact a policy, indicating what to do under which circumstances. The tryout of a single dual solution thus involves simulation, and is very time consuming. The idea of ordinal optimization (Ho, 1995; Chen, 1995) has been employed to perform short simulation runs on selected candidate dual solutions to determine the "order" or "ranking" of their objective function values. The winner(s) of the short tryout are then the dual solutions(s) to be selected to generate implementable schedules, and rigorous simulation runs are then performed to obtain performance statistics.

Schedule implementation. The list scheduling heuristics used here is a modified version of what was presented in Hoitomt et al. (1993). The difference is that the schedule is dynamically constructed here based on the realizations of random events by exploiting the close-loop nature of SDP solutions. A list U of "assignable" tasks that can be started without violating precedence constraints is created at time 0, and updated at subsequent time units based on the realizations of random events and SDP solutions. Tasks in U are sorted in the ascending order of their beginning times, where an uncertain task is considered one iteration at a time. The required types of resources are then assigned to tasks according to list U as resources become available. If at a particular time k there are not enough resources for tasks, a greedy heuristic determines which tasks should begin at that time and which ones are to be delayed by one time unit.

If iteration (p, j, n) is completed and it has a successor, $(p, j, n+1)$ or $(p, j+1, 1)$, the successor's beginning time and the corresponding resource set are obtained from the SDP solutions. Task list U is then updated by inserting the successor in the ascending order of the beginning times. If Project p fails, all tasks of p will be deleted from further consideration. The heuristic is terminated after all tasks are scheduled. Otherwise, above procedure is repeated for the next time unit.

Because of the semi close-loop nature of the overall solution, rescheduling is needed periodically or after a major random event occurrence. Rescheduling can achieve better results without much computational requirement if the multipliers are initialized at their previous values.

3.5. Performance evaluation

Simulation. For a very small problem, algorithm performance can be evaluated by enumerating all possible events and determining their associated possibilities. To analyze results for large problems, a simulation shell has been developed. Random numbers are generated according to their discrete distributions, and Monte Carlo simulation is performed based on the dual solution selected and the realizations of random events. After N Monte Carlo runs, tardiness T_{pn}, earliness E_{pn}, and sample cost J_n of each Project p are obtained for each run n. We are interested in the expected cost J and the weighted sum of project tardiness and earliness variances, $\sigma^2 = \sum_p [w_p \sigma_{tp}^2 + \beta_p \sigma_{ep}^2]$, where σ_{tp}^2 and σ_{ep}^2 are Project p's tardiness and earliness variances, respectively. Small σ^2 indicates predictable start and completion of projects, weighed by the projects' priorities. The values of J and σ^2 can be estimated as

$$\bar{J} = \frac{1}{N} \sum_{n=1}^{N} J_n, \tag{14}$$

$$\overline{\sigma^2} = \sum_p \left\{ W_p \left[\frac{1}{N-1} \sum_{n=1}^{N} \left(T_{pn} - \frac{1}{N} \sum_{n=1}^{N} T_{pn} \right)^2 \right] + \beta_p \left[\frac{1}{N-1} \sum_{n=1}^{N} \left(E_{pn} - \frac{1}{N} \sum_{n=1}^{N} E_{pn} \right)^2 \right] \right\}. \tag{15}$$

The accuracy of the above estimates can be statistically evaluated by using the confidence regions for a given probability of error α. Furthermore, confidence regions can be used as a simple way to compare the performance, e.g., J, of two algorithms using the same set of random variables. If the confidence regions of two methods do not overlap, the one having smaller $\bar{J}$ is better with confidence $1-\alpha$. Otherwise, a so-called "optimal" comparison technique can be used based on hypothesis testing (Bar-Shalom and Li, 1993).

Evaluating the solution via duality gap. Although simulation can be used to obtain statistics of schedules, it cannot tell how close the schedules are to the optimal. It has been proved (Luh et al., 1997) that any dual cost D is a lower bound to the optimal expected cost J^* for the stochastic case under consideration. The relative duality gap $(J-D)/D$ thus provides a measure about the quality of schedules obtained. The optimal schedule can be detected when the duality gap becomes zero, and the iterative multiplier updating process will then be stopped.

4. Numerical results

The method has been implemented in C++, and testing has been performed on a Pentium Pro200 personal computer. In the results to be presented, Example 1 shows the advantage of including risk management in scheduling, and Example 2 shows the solutions in detail and presents valuable insights. Examples 3 and 4 draw data from Delta Industries, an engine part manufacturer in Connecticut, demonstrating that our method (LR/SDP) can effectively handle uncertainties for problems of practical sizes. In Examples 2, 3, and 4, the performance of our method is compared with that of the "mean method" where all uncertain numbers of iterations are replaced by their means, and the converted deterministic problem is solved by using the approach of Wang et al. (1997). The advantage of ordinal optimization for selecting a good dual solution is then illustrated in Example 4.

In the testing, all the multipliers are initialized at zero. For simplicity of presentation, solving all the subproblems once, i.e., updating multipliers $\sum_p S_p$ times, is called an "iteration" for the interleaved subgradient method (ISG). Based on the dual solutions selected, our method and the mean method use the same heuristics to allocate resources to tasks without re-optimization. Numerical results are summarized by the dual cost (D), expected cost (J), duality gap (Gap), CPU time in seconds (S, including ordinal optimization time but not rigorous Monte Carlo simulation time), and weighted sum of tardiness and earliness variances (σ^2).

Example 1. This example is to show the effects of risk management. There are two resource types each with a single unit, and these resources are available over a planning horizon of 10 time units. Three equally weighted projects with 6 tasks in total are to be scheduled as summarized in Table 3. Project 2 fails if it cannot be completed before time 5, or if task (2, 1) cannot meet design criteria after the third iteration. Project 3 fails if it cannot be completed before time 5. The problem is solved by using LR/SDP method in two ways, managing risks (Case 1) or ignoring risk penalties (Case 2) in scheduling, and testing results are summarized in Table 4 and Fig. 3. It can be shown by exhaustive search that $J = 30.5$ of Case 1 is the optimal expected cost. In Case 2, since the dual cost is obtained based on a simplified model, the

Table 3
Data of Example 1

Project	Due date	Task/iteration	Resource needed	t^n_{pjh}	P^n_{pjh}	$\bar{R}_p$	R_{pj}	w_p	β_p
1	2	(1,1)	1	2				1	0.1
		(1,2)	2	1					
2	4	(2,1,1)	1	1	0.15	50	50	1	0.1
		(2,1,2)	1	1	0.2		50		
		(2,1,3)	1	1	0.1		50		
		(2,2)	2	2			55		
3	3	(3,1)	1	3		50	55	1	0.1
		(3,2)	2	1			55		

duality gap is calculated using the dual cost of Case 1 which is the actual lower bound of J.

In the schedules obtained, the schedule for resource 1 is (2, 1), (3, 1), and (1, 1) in Case 1, and (1, 1), (2, 1), and (3, 1) in Case 2. In Case 1, a tradeoff between the tardiness of Project 1 and the failures of Projects 2 and 3 has been made so that Project 2 is started first, and the optimal expected cost 30.5 obtained. In Case 2, since risk penalties are ignored in the simplified scheduling problem, Project 1 is started earlier than Projects 2 and 3, resulting in a much higher expected cost J than that of Case 1 because of the higher failure prob-

Table 4
Results of Example 1

Case	J	D	Gap (%)	Failure probability	
				Project 2 (%)	Project 3 (%)
1 (Managing risks)	30.5	30.2	1.0	0.3	15
2 (Ignoring risks)	115	10.53	280.8	15	100

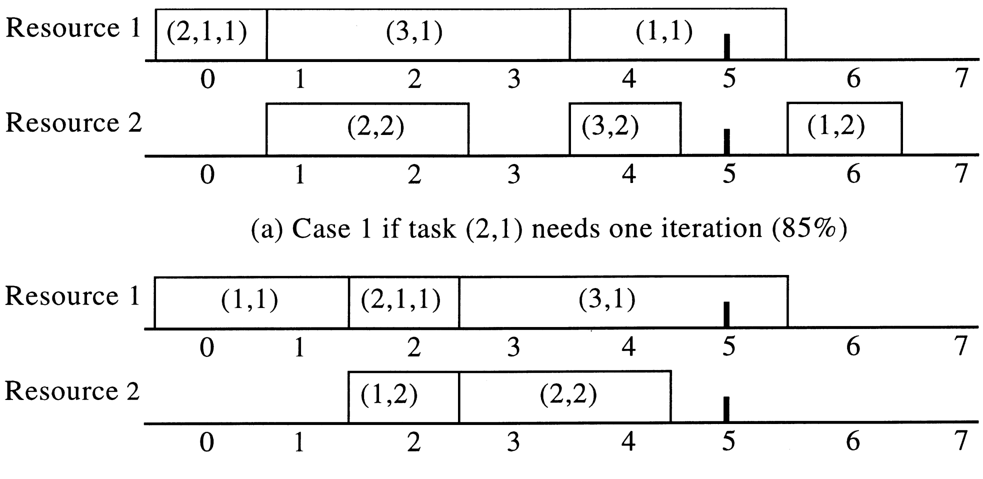


Fig. 3. Gantt charts of Example 1.

abilities of Projects 2 and 3 as shown in Table 4. The results thus illustrate that managing risks in scheduling can achieve better solutions with lower expected costs and lower project failure probabilities, implying reliable on-time completion.

Example 2. There are three resource types each with a single unit, with Type 1 unit not available in time periods 2, 3, and 7–10. Three unequally weighted projects (8 tasks in total) are to be scheduled over a planning horizon of 50 time units as summarized in Table 5. Each task can be performed by one eligible resource type except that task (3, 2) can be performed either by resource Type 1 or Type 2. Task 2 of Project 1 may be completed in one iteration (85%), or needs a second iteration (15%). Results are summarized in Table 6 and Fig. 4. It can be shown by exhaustive search that the schedule obtained by the LR/SDP method has the optimal expected cost. In the mean method, the mean number of iterations of task (1, 2) is 1.3, and is rounded to 1. At the same time, the round-off error is compensated by enlarging processing time t^1_{12h} with 15% of t^2_{12h} to 15.95, which is rounded to 16. Similar to Case 2 of Example 1, the duality gap of the mean method is calculated based on the actual lower bound of J, the D obtained by the LR/SDP method.

In the mean method, the uncertain task (1, 2) is treated as having a single iteration, and Project 1 is started before Project 2 in Fig. 4(c). This sequence, however, fails to consider that both Projects 1 and 2 will have large tardiness when the second iteration of task (1, 2) is needed (Fig. 4(d)). In our method, Project 1 is started after Project 2, and only Project 1 has large tardiness in Fig. 4(b). Our method thus has lower expected cost and lower variance than those of the mean method, implying better on-time project completion with higher predictability. The reason is that the mean method handles uncertainties based on their mean values, and cannot adequately manage the realizations of individual events although the same heuristics is used.

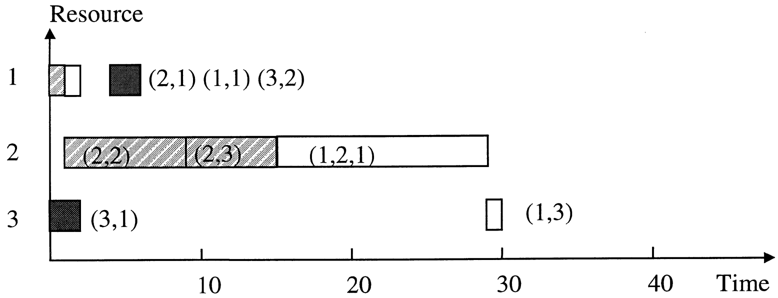
Example 3. This example draws data from Delta Industries with contrived uncertainties, and is to show the effects of various levels of uncertainties on algorithm performance. Eighteen resource types of 35 units are available over a planning horizon of 150 time units. Twenty projects are scheduled, with 23 subprojects decomposed into 123 tasks. A task may be performed by an eligible

Table 5
Data of Example 2

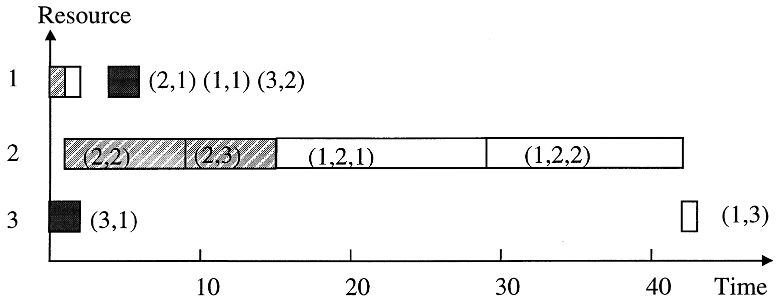
Project	Due date	Task	Resource needed	t^n_{pjh}	w_p	β_p
1	1	(1,1)	1	1	2	0.1
		(1,2,1)	2	14		
		(1,2,2)	2	13		
		(1,3)	3	1		
2	1	(2,1)	1	1	1.9	0.1
		(2,2)	2	8		
		(2,3)	2	6		
3	3	(3,1)	3	2	1	0.1
		(3,2)	1	2		
			2	2		

Table 6
Results of Example 2

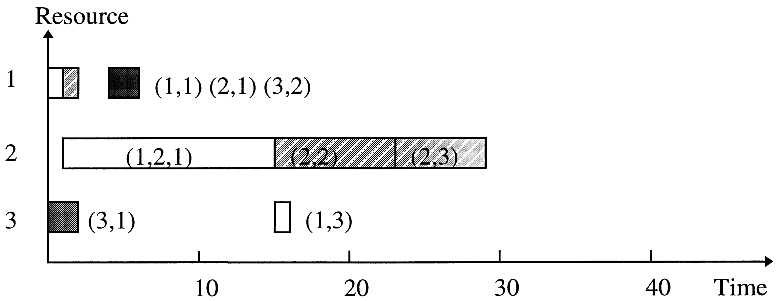
Method	J	D	Gap (%)	σ^2	S.
LR/SDP	2162.2	2159.63	0.12	43.10	0.8
Mean	2189.2	2107.30	1.37	84.04	0.6



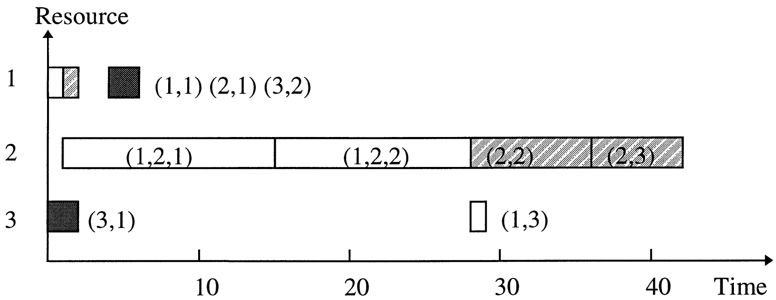
(4.a) LR/SDP method when task (1,2) needs one iteration



(4.b) LR/SDP method when task (1,2) needs two iterations



(4.c) Mean method when task (1,2) needs one iteration



(4.d) Mean method when task (1,1,2) needs two iterations

Fig. 4. Gantt charts of Example 2.

resource set, with the number of eligible resource sets ranging from one to three. There are 2703 multipliers. Four cases are considered, having 0%, 10%, 30%, and 55% projects with uncertainties, respectively. The uncertain numbers of iterations are randomly generated. For each case, a dual solution is selected after 100 ISG iterations, and 1000 Monte Carlo runs are then conducted. Testing results are summarized in Tables 7 and 8, and confidence regions are obtained with error probability $\alpha = 0.05$. Another stopping criteria for the LR/SDP method is examined where the multiplier updating process is terminated when the duality gap is reduced to less than 5%, and the corresponding results are presented in Table 9.

The two methods generate identical results for the deterministic case (0% uncertainties). For cases with 30% and 55% uncertainties, our method is better than the mean method by having lower expected costs and lower variances with non-overlapping confidence regions. For the case with 10% uncertainties, the confidence regions of the two methods overlap, and the result of "optimal" comparison technique is also inconclusive. This implies that the two methods tend to the same result when uncertainties are low. The CPU time needed for our method increases with the levels of uncertainties, but is only slightly higher than that of the mean method.

Example 4. This example also draws data from Delta Industries with contrived uncertainties. It is to show the improvement in schedule quality as the number of multiplier updating iterations increases, and to demonstrate that ordinal optimization can select a good dual solution in a timely fashion. In this example, 47 projects are scheduled on 18 resource types of 35 units over a planning horizon of 160 time units, with 24 projects having uncertain number of iterations. The projects consists of 55 subprojects that are decomposed into 292 tasks, with a total of 2888 multipliers. A task may be performed by an eligible resource set, with the number of eligible resource sets ranging from

Table 7
Results of the LR/SDP method after 100 ISG iterations

Percentage of uncertainties (%)	$\bar{J}$, [confidence region]	$\overline{\sigma^2}$, [confidence region]	D	Gap	S
0	5301.6, [5301.6, 5301.6]	0, [0, 0]	4926.6	7.6	20.1
10	5591.5, [5493.4, 5689.6]	115.1, [98.1, 132.1]	5201.6	7.5	22.3
30	6476.8, [6395.8, 6557.8]	180.5, [170.7, 190.3]	5900.3	9.8	28.3
55	7600.1, [7500.8, 7699.4]	226.3, [211.4, 241.2]	6871.8	10.6	39.8

Table 8
Results of the mean method after 100 ISG iterations

Percentage of Uncertainties (%)	$\bar{J}$, [confidence region]	$\overline{\sigma^2}$, [confidence region]	S
0	5301.6, [5301.6, 5301.6]	0, [0, 0]	19.0
10	5650.9, [5549.7, 5752.1]	102.5, [94.0, 111.0]	20.2
30	6712.1, [6612.1, 6812.1]	212.0, [196.6, 227.4]	26.5
55	9020.5, [8843.6, 9197.4]	1214.0, [1122.9, 1305.1]	35.0

Table 9
Results of the LR/SDP method when duality gap less than 5%

Percentage of uncertainties (%)	$\bar{J}$	$\overline{\sigma^2}$	D	Gap (%)	S
0	5151.0	0	4962.4	3.8	39.0
10	5499.5	100.9	5282.9	4.1	45.9
30	6266.9	177.7	6037.5	3.8	102.0
55	7321.7	195.7	6986.4	4.8	230.2

one to four. Four cases are considered, with 50, 100, 200, and 500 ISG iterations, respectively. Ordinal optimization conducts 30 simulation runs for each dual solution of the last 5% ISG iterations, and selects the best one. Based on 1000 Monte Carlo runs, testing results are summarized in Tables 10 and 11, where the confidence regions are obtained with error probability $\alpha = 0.05$.

From the testing, it can be seen that our method is better than the mean method for a fixed number of ISG iterations. As the number of ISG iterations increases, better dual costs are obtained for both methods at the increase of computation time. Good expected costs, however, can generally be obtained within a reasonable number of ISG iterations (200 here).

To examine the advantage of using ordinal optimization in the LR/SDP method, 1000 Monte Carlo runs are conducted using the dual solution with the highest dual cost for the four cases, and the resulting average duality gap is 10.3% larger than that of Table 10. On the other hand, 1000 Monte Carlo runs are conducted on every dual solution of Case 4 to find the best implementable schedule. The minimum expected cost 7511.9 with a 4.1% duality gap is obtained in 1000 min, only slightly smaller than 7606.1 with a 5.4% gap of Table 10. Above comparison demonstrates that ordinal optimization can select a good dual solution while saving significant amount of simulation efforts.

5. Concluding remarks

A new problem formulation and a novel solution methodology that synergistically combines Lagrangian relaxation, stochastic dynamic programming, ordinal optimization, and heuristics are presented for scheduling design projects with uncertain number of iterations while managing design risks. Uncertainties are appropriately handled by satisfying iteration and task precedence constraints for each possible number of iterations. The expected capacity constraints and expected subproject precedence constraints maintain the computational complexity at a manageable level without much loss of modeling accuracy and scheduling performance, enabling the method to solve problems of practical sizes. Testing results supported by simulation demonstrate that near optimal schedules are obtained in a computationally efficient manner with low expected tardiness and earliness penalties and low project failure probabilities, implying reliable completion with short lead times.

Although only uncertain number of iterations is considered, the formulation and method can be extended to handle other kinds of uncertainties,

Table 10
Results of Example 4 (LR/SDP method)

ISG iteration no.	$\bar{J}$, [confidence region]	$\overline{\sigma^2}$, [confidence region]	D	Gap (%)	S
50	9878.8, [9266.5, 10491.1]	1313.6, [1243.5, 1383.7]	6410.4	54.1	36.2
100	8661.0, [8489.1, 8832.9]	688.0, [647.7, 728.3]	6769.4	27.9	72.1
200	7675.0, [7476.9, 7873.1]	441.3, [400.5, 482.1]	6938.8	10.6	157.4
500	7606.1, [7508.0, 7704.2]	202.9, [161.6, 244.2]	7216.0	5.4	400.0

Table 11
Results of Example 4 (mean method)

ISG iteration no.	$\bar{J}$, [confidence region]	$\overline{\sigma^2}$, [confidence region]	S
50	10511.2, [10199.7, 10822.7]	2011.0, [1910.9, 2111.1]	35.0
100	9354.6, [9053.4, 9655.8]	1832.0, [1726.3, 1937.7]	67.3
200	9034.9, [8735.4, 9334.4]	1704.1, [1598.9, 1809.3]	144.1
500	8975.2, [8734.6, 9215.8]	1690.4, [1579.9, 1800.9]	377.0

e.g., random processing time, uncertain project importance and due date (Luh et al., 1997).

Acknowledgements

The work was supported in part by the United Technologies Research Centre, the National Science Foundation under Grant DMI-9500037, and the Advanced Technology Center for Precision Manufacturing, the University of Connecticut. The authors would like to thank Dr. Ralph Wood of the United Technologies Research Center, Mr. Dong Chen, Mr. Xing Zhao, and Mr. Jihua Wang of the University of Connecticut for their valuable help.

Appendix A. A list of symbols

b^1_{pj}	beginning time of task (p, j)
b^n_{pj}	beginning time of iteration (p, j, n)
c_{pj}	completion time of task (p, j)
c^n_{pj}	completion time of iteration (p, j, n)
d_p	due date of Project p
$\bar{d}_p$	absolute deadline of Project p
E_p	earliness of Project p
H	number of resource types
H_{pj}	the set of all eligible resource sets that are capable of performing (p, j)
h^n_{pj}	an eligible resource set to perform (p, j, n), $1 \leqslant n \leqslant N_{pj}, h^n_{pj} \in H_{pj}$
I_{pj}	set of immediate succeeding tasks of (p, j)
J	objective function, also called (expected) cost
K	planning horizon of scheduling
k	discrete time index, $0 \leqslant k \leqslant K-1$
L_{pi}	subproblem of subproject i of Project p
L^*_{pi}	minimal cost for subproblem L_{pi}
M_{kh}	number of type h resource available at time $k, h \in H$
m_{pjh}	number of type h resource units in h_{pj}
n_{pj}	number of actually happening iterations of uncertain task (p, j)
N_p	number of tasks in Project p
N_{pj}	allowable (maximum) number of iterations of uncertain task (p, j)
P	total number of projects to be scheduled
P^n_{pj}	probability to need another iteration after finishing the nth iteration of uncertain task (p, j)
(p, j)	the jth task of Project p
(p, j, n)	the nth iteration of uncertain task (p, j), a deterministic (p, j) also denoted as $(p, j, 1)$
R_p	risk penalty of Project p
$\bar{R}_p$	opportunity cost when Project p fails
R_{pj}	incurred cost on subproject i which is at task (p, j) upon failure
S_p	number of subprojects in Project p
S_{pi}	set of serial tasks of subproject i of Project p
T_p	tardiness of Project p
t^n_{pjh}	deterministic integer processing time of (p, j, n) for resource set h_{pj}
w_p	weight of tardiness penalty for Project p
$\bar{\Delta}_p$ 0-1	integer variable identifying if Project p does not fail
β_p	weight of earliness penalty for Project p
δ^n_{pjkh} 0-1	integer variable identifying that type h resource is assigned on (p, j, n) at time k
η_{pjr}	Lagrange multiplier of subproject precedence constraint between (p, i, j) and (p, q, r), $i \neq q$
π_{kh}	Lagrange multiplier of resource type h at time k

References

Bar-Shalom, Y., Li, X.R., 1993. Estimation and Tracking: Principles, Techniques, and Software, Artech House. pp. 75–77.

Blazewicz, J., Lenstra, J.K., Rinnooy Kan, A.H.G., 1983. Scheduling projects to resource constraints: Classification and complexity. Discrete Applied Mathematics 5, 11–24.

Boctor, F., 1990. Some efficient multi-heuristic procedures for resource-constrained project scheduling. European Journal of Operational Research 49 (1), 3–13.

Chen, C.H., 1995. A effective approach to smartly allocate computing budget for discrete event simulation. Proceedings of the 34th IEEE Conference on Decision and Control, 2598–2605.

Chen, H., Chu, C., Proth, J.M., 1995. A more effective Lagrangian relaxation approach. Proceedings of IEEE International Conference on Robotics and Automation, 496–501.

Christofides, N., Alvarez-Valdez, R., Tamarit, J.M., 1987. Project scheduling with resource constraints: A branch and bound approach. European Journal of Operational Research 29 (3), 262–273.

Cooper, D., Chapman, C., 1987. Risk Analysis for Large Projects – Models, Methods and Cases. Wiley, New York .

Czerwinski, C.S., Luh, P.B., 1994. Scheduling products with bills of materials using an improved Lagrangian relaxation technique. IEEE Transactions on Robotics and Automation 10 (2), 99–111.

Davis, E.W., Patterson, J.H., 1975. A comparison of heuristic and optimum solutions in resource-constrained project scheduling. Management Science 21 (8), 944–955.

Demeulemeester, E., Herroelen, W., 1996. A efficient optimal solution procedure for the preemptive resource-constrained project scheduling problem. European Journal of Operational Research 90, 1803–1818.

Elmaghraby, S.E., 1977. Activity Networks: Project Planning and Control by Network Models. Wiley, New York.

Eppinger, S.D., Whitney, D.E., Smith, R.P., 1994. A model-based method for organizing tasks in product development. Research in Engineering Design 6, 1–13.

Federgruen, A., Mosheiov, G., 1997. Single machine scheduling problems with general breakdowns, earliness and tardiness costs. Operations Research 45 (1), 66–71.

Gargeya, V.B., Deane, R.H., 1996. Scheduling research in multiple resource constrained job shops: A review and critique. International Journal of Production Research 34 (8), 2077–2097.

Ho, Y.C., 1995. A new paradigm for stochastic optimization and parallel simulation. Discrete Event Systems, Manufacturing Systems, Communication Networks. Springer, Berlin, pp. 59–72.

Hoitomt, D., Luh, P.B., Pattipat, K.R., 1993. A practical approach to job-shop scheduling problems. IEEE Transactions on Robotics and Automation 9 (1), 1–13.

Kaskavelis, C.A., Caramanis, M., 1995. Efficient Lagrangian relaxation algorithms for real-life-size job-shop scheduling problems, Working paper. Department of Manufacturing Engineering, Boston University; also personal communications.

Larson, N., Kusiak, A., 1996. Managing design process: A risk assessment approach. IEEE Transactions on System, Man and Cybernetics 26 (6), 749–759.

Luh, P.B., Hoitomt, D.J., 1993. Scheduling of manufacturing systems using the Lagrangian relaxation technique. IEEE Transactions on Automatic Control 38 (7), 1066–1079.

Luh, P.B., Chen, D., Thakur, T.S., 1997. Modeling uncertainty in jobshop scheduling. First International Conference on Operations and Quantitative Management India, pp. 490–497.

Neumann, K., 1990. Stochastic Project Networks – Temporal Analysis, Scheduling and Cost Minimization. Springer, Berlin, 1990.

Nichols, G., 1990. Getting engineering changes under control. Journal of Engineering Design 1 (1) .

Nukala, M.V., Eppinger, S.D., Whitney, Daniel E., 1995. Generalized model of design iteration using signal flow graphs. DE-vol. 83, 1995 Design Engineering Technical Conferences, vol. 2. ASME, pp. 413–422.

Patterson, J.H., 1984. A comparison of exact approaches for solving the multiple constrained resource, project scheduling problem. Management Science 30 (7), 854–867.

Pinedo, M., Schrage, L., 1982. Stochastic shop scheduling: A survey. Deterministic and Stochastic Scheduling. Reidel, Dordrecht, pp. 181–196.

Pinedo, M., 1995. Scheduling – Theory, Algorithm and Systems. Prentice Hall, Englewood Cliffs, NJ.

Smith, R.P., Eppinger, S.D., 1995. Identifying controlling features of engineering design iteration, Working Paper No. WP #3348-91-MS. MIT Sloan School of Management, Cambridge, MA.

Wang, J., Luh, P.B., 1996. A combined Lagrangian relaxation and dynamic programming algorithm for job shop scheduling. In: Proceedings of Rensselaer's International Conference on Computer Integrated manufacturing, France, pp. 3–8.

Wang, J., Luh, P.B., Zhao, X., Wang, J.L., 1997. An optimization-based algorithm for job shop scheduling. Sādhanā, Journal of Indian Academy of Sciences 22 (2), 241–256.

Zhao, X., Luh, P.B., Wang, J., 1997. The surrogate gradient algorithm for Lagrangian relaxation method. Proceedings of the 36th IEEE Conference on Decision and Control .