A Monte-Carlo Ant Colony System for Scheduling Multi-mode Projects with Uncertainties to Optimize Cash flows

Wei-Neng Chen, *Student Member, IEEE*, Jun Zhang, *Senior Member, IEEE*, Ou Liu and Hai-lin Liu

***Abstract*—Project scheduling under uncertainty is a challenging field of research that has attracted an increasing attention in recent years. While most existing studies only considered the classical single-mode project scheduling problem with makespan criterion under uncertainty, this paper aims to deal with a more realistic and complicated model called the stochastic multi-mode resource constrained project scheduling problem with discounted cash flows (S-MRCPSPDCF). In the model, uncertainty is sourced from activity durations and costs, which are given by random variables. The objective is to find an optimal baseline schedule so that the project's expected net present value (NPV) of cash flows is maximized. In order to solve this intractable problem, an ant colony system (ACS) algorithm is designed. The algorithm dispatches a group of ants to build baseline schedules iteratively based on pheromones and an expected discounted cost (EDC) heuristic. In addition, because it is impossible to evaluate the expected NPVs of baseline schedules directly due to the presence of random variables, the algorithm adopts Monte Carlo (MC) simulations to evaluate the performance of baseline schedules. Experimental results on 33 instances demonstrate the effectiveness of the proposed scheduling model and the ACS approach.**

Index Terms—project scheduling, optimization under uncertainty, cash flow, ant colony optimization (ACO), ant colony system (ACS)

I. Introduction

Project scheduling has long been a key factor in project management. The classic model for project scheduling, namely the resource constrained project scheduling problem (RCPSP) [1][2], involves scheduling the activities of a project with the objective to minimize makespan subject to precedence and resource constraints. The problem has been proven to be NP-hard [2]. A more general and realistic formulation of the RCPSP is the multi-mode RCPSP (MRCPSP) [4][5]. The MRCPSP considers a common situation in real-life projects that activities can be performed in alternative execution modes. These modes may consume different quantities of processing time, cost, and resources to fulfill the same activity. To handle the time-cost-resource tradeoffs between these modes, the MRCPSP has to deal with not only the schedule of activities but also the selections of execution modes. As a result, the computational complexity of the MRCPSP significantly increases. It has been demonstrated by Kolisch [6] that finding a feasible solution for the MRCPS P with more than one nonrenewable resource is already NP-complete, let alone the problem of finding the optimal feasible solution. While the classical RCPSP and MRCPSP use makespan as the criterion for measuring the performance of a schedule, recently, the net present value (NPV) of cash flows has been increasingly considered as a more meaningful criterion for evaluating the potential of a project [7][8]. NPV is defined as the difference between cash inflows and outflows during the processing course of a project, taking into account the time value of money by discounting the cash flows. The presence of the cash flow criterion results in a more complicated scheduling model called the MRCPSP with discounted cash flows (MRCPSPDCF) [9]-[11].

Due to the importance and difficulty of these scheduling problems, a considerable amount of research effort has been devoted in the literature to propose various scheduling models and algorithms during the last decades [1]-[12]. However, in most studies, the scheduling models are based on a premise that the parameters such as activity durations and costs are deterministic. In real world application, because of the occurrence of unexpected situations such as machine malfunction and fluctuation in resource cost, it is almost impossible to determine all parameters of the project before execution. The probability of processing a project exactly following the predefined plan is very low. Therefore, to be a practical scheduling model, uncertainty is an innegligible factor.

Scheduling under uncertainty is a challenging field of research that has attracted an increasing attention in recent years [13]-[16]. For the classical single-mode RCPSP with the makespan criterion, several types of scheduling models with the consideration of uncertainty have been proposed, for example, reactive models [17], stochastic models [18]-[22], fuzzy models [23], and robust models [24]. The most common model is the stochastic RCPSP [14], which aims at scheduling project activities with uncertain durations in order to minimize the expected project makespan or maximizing the probability of finishing the project before the due date. To solve the RCPSP with uncertainty, existing scheduling algorithms include simple scheduling polices [18], branch-and-bound [19], tabu search [20], genetic algorithms

This work was supported in part by the NSFC Joint Fund with Guangdong under Key Project U0835002 and the National High Technology Research and Development Program ("863"Program) of China (2009-2010) No.2009AA01Z208.

Wei-Neng Chen and Jun Zhang are with the Department of Computer Science, Sun Yat-sen University, Guangzhou, China. Jun Zhang is the corresponding author (email: junzhang@ieee.org)

Ou Liu is with the Hong Kong Polytechnic University, Hung Hom, Hong Kong.

Hai-lin Liu is with Guangdong University of Technology, Guangzhou, China.

978-1-4244-8126-2/10/$26.00 ©2010 IEEE

[21][22], and so on.

However, for the more practical scheduling models with the cash flow criterion, the literature that considers uncertainty is rather sparse [13]. The existing studies of cash flow optimization in projects with uncertainty only include the dynamic scheduling policy based on the backward stochastic dynamic programming recursion method proposed by Creemers et al. [25][26], the dynamic scheduling algorithm designed by Sobel et al. [27], and the branch-and-bound approach developed by Wolfram et al. [28]. In addition, in all these approaches, the resource constraints are disregarded and only a single execution mode is considered. In other words, these scheduling models are significantly simplified. To the best of our knowledge, the more realistic and complicated model with resource constraints and alternative execution modes is still not considered in the literature.

This paper intends to deal with a multi-mode resource constrained project scheduling problem under uncertainty with the objective to maximize the NPV of cash flows. We term the scheduling model the stochastic MRCPSPDCF (S-MRCPSPDCF). In the model, uncertainty is sourced from activity durations and costs. The duration and cost of every alternative execution modes are given by random variables of certain probability distributions. Because of the presence of random variables, the scheduling problem can be divided into two stages: the pre-execution stage before the actual realization of the random variables and the post-execution stage when all random variables have been revealed. In application, in order to effectively manage a project and evaluate the potential of a project, a baseline schedule is needed to build in the pre-execution stage. The baseline schedule is composed of a processing order of activities and a list of selected execution modes. After the realization of all random variables in the post-execution stage, the baseline schedule can be converted to an actual schedule for the real execution of the project. The objective of the S-MRCPSPDCF is actually to find a baseline schedule in the pre-execution state, so that for all possible realization of the random variables in the post-execution state, the expected NPV of cash flows is maximized.

To solve this problem, this paper proposes an ant colony optimization (ACO) approach. ACO is a population-based optimization algorithm proposed by Dorigo [29][30] in the early 1990s in the light of how ants manage to discover the shortest path from their nest to food source. It has been successfully applied to many complicated scheduling problems [11][31]. Recently, Gutjahr [32][33] and Balaprakash [34] have developed a simulation ACO algorithm to tackle the probabilistic traveling salesman problem (PTSP), demonstrating the great potential of ACO for solving optimization problems under uncertainty. Inspired by these studies, this paper develops an ant colony system (ACS) algorithm for the S-MRCPSPDCF. ACS is by now one of the best-performed ACO variants. In the algorithm, artificial ants build baseline schedules iteratively according to pheromones and an expected discounted cost based (EDC) heuristic. The EDC heuristic is defined based on the expected values of activity durations and costs. Because the model has uncertainties, it is impossible to evaluate the expected NPV of a baseline schedule through a deterministic expression. In this situation, the proposed algorithm adopts the Monte Carlo (MC) simulations. Thus the algorithm is termed MC-ACS. Different from other ACO variants, ACS only uses the best-so-far solution to update pheromones during each iteration. As such, the algorithm is able to use a rough simulation by a small number of scenarios to distinguish the best-so-far solution. In addition, a special technique that increases the sample size linearly [33] is applied. These design schemes have the function of reducing computational cost for the algorithm. The proposed algorithm is tested on 33 randomly-generated instances. Experimental results demonstrate the effectiveness of the proposed scheduling model and the ACS approach.

The rest of this paper is organized as follows. Section II formulates the considered S-MRCPSPDCF model. Section III presents the proposed ACS approach. Experimental studies are shown in Section IV, and the conclusions are finally summarized in Section V.

II. FORMULATION OF THE S-MRCPSPDCF MODEL

A. Basic Features

The S-MRCPSPDCF considered in this paper is a problem of scheduling a multi-mode project with uncertain durations and costs, subject to precedence and resource constraints, with the objective to maximize the expected NPV of cash flows of the project. The model has the following features:

1) Precedence Constraint: In the model, a project is described by an activity-on-arc (AoA) network $G=(E,A)$. The node set $E=\{e_1,e_2,\ldots,e_m\}$ in the network corresponds to the set of events in the project, where m is the number of events. The arc set $A=\{a_1,a_2,\ldots,a_n\}$ corresponds to the set of activities, where n is the number of activities. The AoA network defines the precedence relations between the activities. An example of the AoA network is shown in Fig. 1.

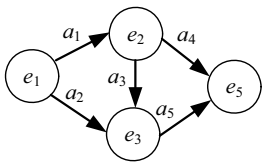

Fig. 1. An example of the AoA network

2) Resource Constraint: Renewable resource constraint is considered in the model. Suppose the project uses R types of renewable resources, for each time period, the consumption of the k^{th} (k=1,2,…,R) renewable resource is limited to R^k units.

3) Multi-mode: In real world applications, it is common that the activity can be processed in alternative modes. To

formulate this multi-mode characteristic, in the model, each activity a_i (i=1,2,…,n) is associated with a set of execution modes $M_i = \{m_{i1}, m_{i2}, \cdots, m_{i|M_i|}\}$, where m_{ij} is the j^{th} mode for the execution of a_i and $|M_i|$ is the total number of available modes for a_i. The execution modes are non-preemption, i.e., once a mode m_{ij} is executed, it has to be complete without any interruption. The duration and cost of m_{ij} are denoted as d_{ij} and c_{ij}, respectively. The mode m_{ij} also consumes r_{ij}^k units of the k^{th} renewable resource. For the process of a_i, different modes $m_{i1}, m_{i2}, \cdots, m_{i|M_i|}$ may consume different quantities of processing time, cost, and resources. In this case, there are time-resource-cost tradeoffs between different modes.

4) Uncertain Durations and Costs: Different from the traditional deterministic scheduling models, the S-MRCPSPDCF takes account of uncertainties. In the model, the duration d_{ij} and cost c_{ij} of the mode m_{ij} are given by random variables instead of deterministic numbers. The random variables are of some pre-given probability distributions (e.g., beta, uniform and normal distributions). For the sake of convenience, we suppose that the durations are positive integer valued random variables and the costs are non-negative random variables in the form of either integer or real numbers. In addition, a lower and upper bound are defined to control the domain of durations and costs. The durations and costs of all modes compose a random vector:

$$\omega = \begin{pmatrix} d_{11}, c_{11}, d_{12}, c_{12}, \cdots, d_{1|M_1|}, c_{1|M_1|} \\ d_{21}, c_{21}, d_{22}, c_{22}, \cdots, d_{2|M_2|}, c_{2|M_2|} \\ \vdots \\ d_{n1}, c_{n1}, d_{n2}, c_{n2}, \cdots, d_{n|M_n|}, c_{n|M_n|} \end{pmatrix} \tag{1}$$

B. From Baseline Schedules to Actual Schedules

In practice, in order to manage a project efficiently and estimate the potential of a project accurately, it is necessary to build an efficient baseline schedule before executing the project. A baseline schedule $\boldsymbol{S}$ for the S-MRCPSPDCF is in the form of

$$\boldsymbol{S} = \begin{bmatrix} (a_{p_1}, a_{p_2}, \cdots, a_{p_n}) \\ (k_{p_1}, k_{p_2}, \cdots, k_{p_n}) \end{bmatrix} \tag{2}$$

where $p_1, p_2, \cdots, p_n$ is a permutation of the numbers from 1 to n, $(a_{p_1}, a_{p_2}, \cdots, a_{p_n})$ gives the processing order of activities, and $(k_{p_1}, k_{p_2}, \cdots, k_{p_n})$ gives the selections of execution modes. Here k_{p_i} means that the activity a_{p_i} is mapped to the $k_{p_i}^{\text{th}}$ mode of a_{p_i} (i.e., the mode $m_{p_i k_{p_i}}$) to process in the schedule $\boldsymbol{S}$. To guarantee the satisfaction of precedence constraint, if a_i is a predecessor activity of a_j in the AoA network, a_i must appear earlier than a_j in the processing order $(a_{p_1}, a_{p_2}, \cdots, a_{p_n})$.

Although the processing order of activities and the selections of execution modes have been defined in the baseline schedule, it is still not enough to determine an actual schedule for the execution of the project because of the presence of random variables. Only after the realization of random variables, the actual schedule can be formed. Suppose

$$\omega' = \begin{pmatrix} d'_{11}, c'_{11}, d'_{12}, c'_{12}, \cdots, d'_{1|M_1|}, c_{1|M_1|} \\ d'_{21}, c'_{21}, d'_{22}, c'_{22}, \cdots, d'_{2|M_2|}, c_{2|M_2|} \\ \vdots \\ d'_{n1}, c'_{n1}, d'_{n2}, c'_{n2}, \cdots, d'_{n|M_n|}, c_{n|M_n|} \end{pmatrix} \tag{3}$$

is a realization of ω (where d'_{ij} and c'_{ij} are the realizations of d_{ij} and c_{ij}, respectively), a baseline schedule $\boldsymbol{S}$ can be converted into an actual schedule $\boldsymbol{S}(\omega')$ under the realization ω' based on a serial schedule generation scheme (SSGS) [35]. The basic idea of SSGS is to schedule the modes to the earliest possible start time to execute, following the processing order $(a_{p_1}, a_{p_2}, \cdots, a_{p_n})$ and subject to precedence and resource constraints. The pseudo code of the SSGS is shown in Fig. 2. Here if activity a_i is executed in mode m_{ik_i} in $\boldsymbol{S}$, we denote the start time of m_{ik_i} in $\boldsymbol{S}(\omega')$ as $\boldsymbol{S}(\omega').ST_{ik_i}$ and the finish time of $\boldsymbol{S}(\omega')$ as $\boldsymbol{S}(\omega').FT$.

Procedure SSGS($\boldsymbol{S}, \omega'$)
1 Initialize; /*all units of resources are set available
2 **for** i=1 **to** n+1
3 Select the activity a_{p_i} from the list $(a_{p_1}, a_{p_2}, \cdots, a_{p_n})$ in $\boldsymbol{S}$;
4 Map a_{p_i} to the execution mode $m_{p_i k_{p_i}}$;
5 Schedule $m_{p_i k_{p_i}}$ to the earliest possible start time t to execute subject to precedence and resource constraints;
6 $\boldsymbol{S}(\omega').ST_{ik_i} \leftarrow t$;
7 Update resources;
8 **end for**
9 Set $\boldsymbol{S}(\omega').FT$ to the time when the all execution modes have finished;
end procedure

Fig. 2. Pseudo code of the SSGS

C. Evaluation of the NPV of Cash Flows

After converting the baseline schedule $\boldsymbol{S}$ into an actual schedule $\boldsymbol{S}(\omega')$ under the realization ω', we can determine the actual NPV of cash flows of the schedule $\boldsymbol{S}(\omega')$. We denote the NPV of $\boldsymbol{S}(\omega')$ as $\boldsymbol{S}(\omega').NPV$. It can be computed by

$$\boldsymbol{S}(\omega').NPV = \boldsymbol{S}(\omega').In - \boldsymbol{S}(\omega').Out + \boldsymbol{S}(\omega').BP \tag{4}$$

where $\boldsymbol{S}(\omega').In$, $\boldsymbol{S}(\omega').Out$ and $\boldsymbol{S}(\omega').BP$ represent the cash inflows, cash outflows, and bonus or penalty, respectively.

1) Cash Inflows

Cash inflows occur when the contractor pays for the execution of the project. We formulate the cash inflows based on the payments at event occurrences (PEO) model, in which payments are made at some predefined milestone events. At the beginning, a proportion of the total value of the project is prepaid as the startup cash flows for the project. Suppose U is the total value of the project and λ is the prepayment rate, the startup cash inflows $\boldsymbol{S}(\omega').SIn$ is given by

$$\boldsymbol{S}(\omega').SIn = \lambda U \tag{5}$$

During the course of the project, the contractor pays for the finishing activities at some predefined milestone events. The cash flows have to be discounted according to a discounted rate α to reflect the time value of money. If W_i (i=1,2,…,n) is the net worth of each activity, θ is the milestone payment rate, and the payment for a_i under $\boldsymbol{S}(\omega')$ occurs at $\boldsymbol{S}(\omega').PT_i$, then the milestone inflows $\boldsymbol{S}(\omega').MIn$ is given by

$$\boldsymbol{S}(\omega').MIn = \sum_{i=1}^{n} \theta W_i \cdot \exp[-\alpha \cdot S(\omega').PT_i] \tag{6}$$

At the end, the contractor pays the rest unpaid values for the project. The final inflows $\boldsymbol{S}(\omega').FIn$ is thus computed by

$$\boldsymbol{S}(\omega').FIn = (U - \lambda U - \sum_{i=1}^{n} \theta W_i) \cdot \exp[-\alpha \cdot \boldsymbol{S}(\omega').ET] \tag{7}$$

So the present value of cash inflows can be computed by summarizing the startup inflows, milestone inflows and final inflows as follows.

$$\boldsymbol{S}(\omega').In = \boldsymbol{S}(\omega').SIn + \boldsymbol{S}(\omega').MIn + \boldsymbol{S}(\omega').FIn \tag{8}$$

2) Cash Outflows

Cash outflows involve the costs of executing the project. If activity a_i is executed in mode m_{ik_i} in the schedule $\boldsymbol{S}$, under the realization ω', the actual duration and cost of m_{ik_i} are d'_{ik_i} and c'_{ik_i}, respectively. So the discounted cash outflows can be evaluated as follows.

$$\boldsymbol{S}(\omega').Out = \sum_{i=1}^{n} c'_{ik_i} \cdot \exp(-\alpha \cdot [\boldsymbol{S}(\omega').ST_{ik_i} + d'_{ik_i}]) \tag{9}$$

3) Bonus or Penalty

The cash flows of bonus (or penalty) arise when the project finishes earlier (or later) than a predetermined interval of due dates $[T_{LOW}, T_{UP}]$. If the finish time $\boldsymbol{S}(\omega').FT$ is earlier than T_{LOW}, the company will gain a bonus payment. Otherwise, if $\boldsymbol{S}(\omega').FT$ comes later than T_{UP}, the company will lose money due to the penalty of delay. If γ and δ are the bonus and penalty rates, the cash flows of bonus or penalty is given by

$$\boldsymbol{S}(\omega').BP = \begin{cases} \gamma U(T_{LOW} - \boldsymbol{S}(\omega').FT) \cdot \exp[-\alpha \cdot \boldsymbol{S}(\omega').FT], \\ \quad \text{if } \boldsymbol{S}(\omega').FT < T_{LOW} \\ 0, \text{if } T_{LOW} \le \boldsymbol{S}(\omega').FT \le T_{UP} \\ -\delta U(\boldsymbol{S}(\omega').FT - T_{UP}) \cdot \exp[-\alpha \cdot \boldsymbol{S}(\omega').FT], \\ \quad \text{if } T_{\mathrm{UP}} < \boldsymbol{S}(\omega').FT \end{cases} \tag{10}$$

Because a baseline schedule is not an actual schedule for the execution of the project, to evaluate the performance of a baseline schedule, the S-MRCPSPDCF model adopts the expected NPV of cash flows under all possible realizations as the criterion. If the random variables in ω are of discrete values (i.e., both durations and costs are integer valued random variables), given a baseline schedule $\boldsymbol{S}$, its the expected NPV is defined as

$$E(NPV(\boldsymbol{S})) = \sum_{\omega' \in \Omega} [\boldsymbol{S}(\omega').NPV \cdot p(\omega = \omega')] \tag{11}$$

where Ω is the domain of ω and $p(\omega = \omega')$ is the probability of obtaining ω' from ω.

If ω contains continuous random variables (i.e., the costs are real valued random variables), the expected NPV of $\boldsymbol{S}$ is defined as

$$E(NPV(\boldsymbol{S})) = \int_{\Omega} \boldsymbol{S}(\omega').NPV \cdot p(\omega')d\omega' \tag{12}$$

where $p(\omega')$ is the probability density function of ω.

The objective of the S-MRCPSPDCF is actually to find a baseline schedule $\boldsymbol{S}$ that maximizes $E(NPV(\boldsymbol{S}))$. However, according to the above definitions, directly computing the value of $E(NPV(\boldsymbol{S}))$ through (11) or (12) within an acceptable time is always an impossible task. This feature makes the scheduling problem even more intractable.

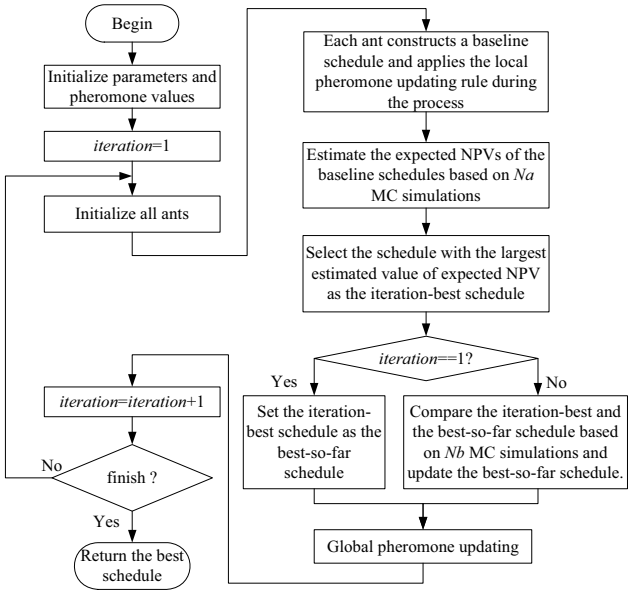

Fig. 3 Flowchart of the MC-ACS Algorithm

III. The Monte Carlo ant colony system

In order to solve this intractable scheduling problem, this paper proposes an MC-ACS algorithm. The basic idea of the algorithm is simulate the pheromone depositing and following behavior of real ants to build baseline schedules for the problem. As the objective functions defined in (11) and (12) cannot be computed directly, the MC simulations are adopted by the algorithm to compare the quality of the baseline schedules built by ants. In general, the MC-ACS algorithm is composed of the following procedures:

- *Solution Construction*: During each iteration, a group of M ants is dispatched to build baseline schedules for the problem.
- *Evaluation and Comparison*: After all ants have completed their solutions at the end of each iteration, the algorithm runs MC simulations to evaluate and compare the performance of

the schedules generated by ants.

- *Pheromone Updating*: Pheromone values are updated by two updating rules: the local pheromone updating rule and the global pheromone updating rule.

These three procedures run iteratively to search for an optimal baseline schedule for the S-MRCPSPDCF. The overall flowchart of MC-ACS is shown in Fig. 3. The algorithm is described in detail as follows.

A. Solution Construction

Let us denote the baseline schedule generated by the l^{th} artificial ant as

$$\boldsymbol{S}^{(l)} = \begin{bmatrix} (a_{p1}^{(l)}, a_{p2}^{(l)}, \cdots, a_{pn}^{(l)}) \\ (k_{p1}^{(l)}, k_{p1}^{(l)}, \cdots, k_{p1}^{(l)}) \end{bmatrix} \tag{13}$$

The schedule defines an order of activities $(a_{p1}^{(l)}, a_{p2}^{(l)}, \cdots, a_{pn}^{(l)})$ and the selections of the execution modes $(k_{p1}^{(l)}, k_{p1}^{(l)}, \cdots, k_{p1}^{(l)})$. It can be actually mapped to a list of execution modes

$$\boldsymbol{S}^{(l)} = \left[m_{p1k_{p1}^{(l)}}, m_{p2k_{p2}^{(l)}}, \cdots, m_{pnk_{pn}^{(l)}} \right] \tag{14}$$

Therefore, to build a baseline schedule, the ants in MC-ACS add execution modes to the list step by step using pheromone and heuristic values until the schedule has been completed.

1) Definition of Pheromone and Heuristic Values

ACO algorithms are characterized by using pheromone and heuristic values to build solutions. Pheromone is a kind of memory of the previous search experiences, and heuristic is some problem-based information. In MC-ACS, pheromone values are defined on the connections between execution modes, and heuristic values are defined on execution modes. Suppose an ant has chosen a mode m_{uv} in the list, the pheromone value for choosing the mode m_{ij} as the next execution mode in the list is given by $\tau(uv,ij)$. The heuristic value for choosing the mode m_{ij} is given by $\eta(ij)$. The heuristic value is defined based on the expected discounted costs of execution modes. Thus we term it the EDC heuristic. Assuming $\bar{d}_{ij}$ and $\bar{c}_{ij}$ are the expected values of d_{ij} and c_{ij}, $\eta(ij)$ can be computed by

$$\eta(ij) = \frac{1}{\bar{c}_{ij} \cdot \exp(-\alpha \cdot \bar{d}_{ij})} \tag{15}$$

In terms of this definition, the EDC heuristic favors the execution modes with low expected discounted cost.

2) Eligible Set

A feasible baseline schedule has to satisfy the precedence constraint of the project. In order to guarantee the feasibility of the schedules built by ants, MC-ACS applies the technique of eligible sets.

In the algorithm, each ant maintains an eligible set of feasible modes that satisfy the precedence constraints. We denote the eligible set of the l^{th} ant as $eligible_l$. For a mode m_{ij}, if its corresponding activity a_i has not been selected in the baseline schedule and all direct predecessor activities of a_i have been selected in the schedule, then m_{ij} is considered to be eligible. During the construction process, only the execution modes in the eligible set can be chosen. In this way, the schedules generated by ants can always satisfy the precedence constraint.

3) The Construction Process

Let m_{00} be a dummy mode that represents the beginning of the project. At the beginning of the construction process, every ant in the algorithm starts with the dummy mode m_{00}.

Suppose that in the $(t\text{-}1)^{\text{th}}$ step, the l^{th} ant has selected the execution mode m_{uv}, then in the t^{th} step, the selection rule for the ant to choose the next execution mode in the schedule is given by (16) and (17).

$$m_{ij} = \begin{cases} \arg\max_{m_{ij} \in eligible_l} \{\tau(uv,ij) \cdot [\eta(ij)]^{\beta}\}, & \text{if } q \le q_0 \\ \text{implement RWS}, & \text{otherwise} \end{cases} \tag{16}$$

$$p(m_{ij}) = \begin{cases} \dfrac{\tau(uv,ij) \cdot [\eta(ij)]^{\beta}}{\sum_{m_{xy} \in eligible_l} \tau(uv,xy) \cdot [\eta(xy)]^{\beta}}, & \text{if } m_{ij} \in eligible_l \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

In this selection rule, a random number $q \in [0,1]$ is generated and compared with a parameter q_0. If $q \le q_0$, the mode m_{ij} that has the largest value of $\tau(uv,ij) \cdot [\eta(ij)]^{\beta}$ among all the modes in the eligible set is selected. Here β is a parameter to weigh the influence of heuristic values. Otherwise, if $q > q_0$, the roulette wheel selection (RWS) scheme is adopted. In RWS, the probability of choosing m_{ij} is in direct proportion to the value of $\tau(uv,ij) \cdot [\eta(ij)]^{\beta}$.

To summarize, the pseudo code of how ants build baseline schedules is shown in Fig. 4.

B. Evaluation and Comparison

Because there are random durations and costs in the S-MRCPSPDCF model, it is impossible to evaluate the expected NPV of cash flows for a baseline schedule $\boldsymbol{S}$ directly. In order to evaluate and compare the performance of the schedules built by ants, the MC-ACS algorithm has to use MC simulations to estimate the expected NPVs.

```
Procedure Solution_Construction
1    Each ant selects the dummy mode m00 as the starting mode;
2    Initialize the eligible set of all ants;
3    for t=1 to n
4        for l=1 to M
5            The l^th ant selects a mode m_ij from its eligible set
                 based on pheromone and heuristic values;
6            a_pt^(l) ← a_i   ;
7            k_pt^(l) ← j   ;
8            Update the eligible set of the l^th ant based on
                 precedence relations;
9            Local pheromone updating;
10       end for
11   end for
end procedure
```

Fig. 4 The pseudo code of constructing baseline schedules

Procedure MC_Simulation($\boldsymbol{S}$)
1 Randomly sample N scenarios $\omega^{(1)},\omega^{(2)},\cdots,\omega^{(N)}$ from ω;
2 $E(NPV(\boldsymbol{S})) \leftarrow 0$;
3 **for** i=1 **to** N
4 SSGS($\boldsymbol{S},\omega^{(i)}$);
5 Evaluate $\boldsymbol{S}(\omega^{(i)}).NPV$;
6 $E(NPV(\boldsymbol{S})) \leftarrow E(NPV(\boldsymbol{S})) + \boldsymbol{S}(\omega^{(i)}).NPV$;
7 **end for**
8 $E(NPV(\boldsymbol{S})) \leftarrow E(NPV(\boldsymbol{S}))/N$;
end procedure

Fig. 5 The pseudo code of evaluating a baseline schedule $\boldsymbol{S}$ using N simulations

Suppose the sample size in the simulation is N, to estimate the expected NPV of a baseline schedule $\boldsymbol{S}$, the algorithm first randomly sample N scenarios $\omega^{(1)},\omega^{(2)},\cdots,\omega^{(Nb)}$ from ω. Then for the i^{th} scenario, it is possible to compute the NPV of the actual schedule $\boldsymbol{S}(\omega^{(i)})$ according to Part B and C of Section II. So the expected NPV of $\boldsymbol{S}$ can be estimated by

$$E(NPV(\boldsymbol{S})) \approx \frac{\sum_{i=1}^{N} \boldsymbol{S}(\omega^{(i)}).NPV}{N} \tag{18}$$

The pseudo code of the MC simulation is given in Fig. 5.

In general, a larger sample size in the simulation will lead to a more accurate estimation of the expected NPV. However, the computational cost of simulation will also increase in this case. Therefore, how to balance the tradeoff between estimation accuracy and computational cost is a key concern for the algorithm.

Different from other ACO variants, the ACS algorithm [30] has a special characteristic that only the information of the best-so-far solution is used to update pheromone values. Therefore, for the rest worse solutions, it is not necessary to estimate their expected NPVs accurately.

Based on this feature of ACS, inspired by the simulation ACO algorithm proposed by Gutjahr for PTSP [32], this paper applies a two-phase simulation method to evaluate and compare the baseline schedules built by ants in the MC-ACS algorithm. In the first phase, a small number *Na* of random scenarios are realized from ω. The expected NPVs of all the baseline schedules built in this iteration are roughly estimated based on the simulations on these *Na* scenarios. The schedule that has the maximal estimated expected NPV is chosen as the iteration-best schedule. In the second phase, a relative larger number *Nb* of random scenarios are realized. The iteration-best schedule and the current best-so-far schedule then undergo a more accurate comparison based on the simulations on these *Nb* scenarios. The winner of the comparison becomes the new best-so-far schedule found by the algorithm.

C. Pheromone Updating

At the beginning of the algorithm, all pheromone values on the connections between modes are initialized to τ_0=1.

During the course of the MC-ACS algorithm, pheromone values are updated by two rules: the local pheromone updating rule and the global updating rule.

The local updating rule is applied immediately after the selection of modes (line 9 of the pseudo code in Fig. 4). If an ant selects the mode m_{uv} in the $(t\text{-}1)^{\text{th}}$ step and selects the mode m_{ij} in the t^{th} step, the pheromone value on the connection between m_{uv} and m_{ij} is updated by

$$\tau(uv,ij) \leftarrow (1-\xi)\tau(uv,ij) + \xi\tau_0 \tag{19}$$

where $\xi \in [0,1]$ is a parameter. The function of local pheromone updating is actually to reduce the pheromone values on the selected connections so that the following ants will have larger probabilities to explore other unselected connections.

After distinguishing the best-so-far schedule at the end of each iteration, the global updating rule is applied to reinforce the connections used by the best-so-far schedule. If m_{uv} is followed by m_{ij} in the best-so-far schedule, the pheromone value on the connection between m_{uv} and m_{ij} is updated by

$$\tau(uv,ij) \leftarrow (1-\xi)\tau(uv,ij) + \xi\Delta^{bs} \tag{20}$$

where Δ^{bs} is defined based on the estimated expected NPV of the best-so-far schedule $\boldsymbol{S}^{bs}$ as follows

$$\Delta^{bs} = \left(\frac{\widetilde{E}(NPV(\boldsymbol{S}^{bs})) - firstWorst}{firstBest - firstWorst} + 1\right)\cdot\psi \tag{21}$$

Here the estimated expected NPV $\widetilde{E}(NPV(\boldsymbol{S}^{bs}))$ is calculated in the second-phase simulation based on *Nb* random scenarios, *firstBest* is the estimated expected NPV of the iteration-best schedule in the first iteration, *firstWorst* is the smallest estimated expected NPV obtained in the first iteration, and $\psi(\psi \geq 1)$ is a parameter to control the scale of pheromone amount.

IV. Experimental Results

A. Test Instances

In the experiment, 33 instances are randomly generated. The instances are of eleven different sizes, and each size corresponds to three instances. These three instances belong to three groups: group a, group b and group c. Taking the three instances with 13 activities for example, we name them Ins13_a, Ins13_b, and Ins13_c, respectively. The other instances are named in the same way. In the instances, each activity is randomly associated with one to five alternative execution modes. The resources consumed by modes are randomly generated. The duration d_{ij} of m_{ij} is assumed to be integer valued random variable of uniform distribution within an interval $[(1-\lambda)\bar{d}_{ij},(1+\lambda)\bar{d}_{ij}]$, where $\bar{d}_{ij}$ is the expectation of d_{ij}. The cost c_{ij} of m_{ij} is assumed to be real valued random variable of normal distribution $N(\bar{c}_{ij},\sigma^2)$, where $\bar{c}_{ij}$ is the expectation of c_{ij}. The expectations of durations and costs are randomly generated. For the instances belonging to group a, group b, and group c, we set (λ=5%, $\sigma=\bar{c}_{ij}$), (λ=8%, $\sigma=5\bar{c}_{ij}$), and (λ=10%, $\sigma=10\bar{c}_{ij}$), respectively. In other

words, the instances in group c have higher uncertain degree than the ones in group a.

TABLE I PERFORAMNCE OF MC-ACS UNDER DIFFERENT SAMPLE SIZES

	MAX	*Na*=1 *Nb*=1	*Na*=1 *Nb*=5	*Na*=1 *Nb*=10	*Na*=1 *Nb*=20	*Na*=2 *Nb*=5	*Na*=1 *Nb*(I)
ins13_a	10000	2333	2344	2312	2292	2188	**2359**
ins13_b	10000	2430	2530	2543	2520	2507	**2569**
ins13_c	10000	2315	**2513**	2482	2404	2442	2466
ins16_a	10000	2957	**2987**	2966	2920	2931	2972
ins16_b	10000	1102	1172	1130	1009	1110	**1199**
ins16_c	10000	3642	3725	3694	3673	3699	**3730**
ins18_a	10000	2485	2511	**2514**	2491	2398	2508
ins18_b	10000	3827	3862	3756	3637	3589	**3947**
ins18_c	10000	3273	3434	3401	3361	3288	**3447**
Ins28_a	10000	**3388**	3193	2957	2825	2826	3386
Ins28_b	10000	579	741	515	355	223	**818**
Ins28_c	10000	5668	5933	6036	5982	5798	**6098**
Ins36_a	20000	**3097**	2790	2299	2213	2330	2978
Ins36_b	20000	2717	2699	2473	2379	2401	**3017**
Ins36_c	20000	1381	1834	1479	980	881	**2206**
Ins40_a	20000	8420	**9347**	9149	8891	8719	9279
Ins40_b	20000	-1242	-1569	-1718	-2274	-2410	**-754**
Ins40_c	20000	-2921	21	-28	-507	-1030	**451**
Ins48_a	20000	**1**	-958	-1452	-2859	-2691	-582
Ins48_b	20000	**1828**	853	860	-27	-130	1417
Ins48_c	20000	5642	6799	6499	6148	5710	**7147**
Ins58_a	30000	7408	9304	9405	9286	9001	**9545**
Ins58_b	30000	4844	8198	7919	6947	6130	**8544**
Ins58_c	30000	7236	9223	**9382**	9023	9210	9289
Ins60_a	30000	-199	-356	-658	-1947	-2134	**448**
Ins60_b	30000	2800	5305	498	3899	3139	**5467**
Ins60_c	30000	-291	5276	4833	3888	3801	**5499**
Ins80_a	40000	-3147	-1215	-2321	-3086	-2810	**-73**
Ins80_b	40000	3499	5342	4713	3038	2751	**6101**
Ins80_c	40000	-3800	**4231**	2140	666	49	3810
Ins98_a	50000	67	2562	2546	360	-239	**3469**
Ins98_b	50000	-6643	-2065	-3409	-4012	-5012	**-1116**
Ins98_c	50000	-7122	314	-515	-1759	-2310	**1594**

B. Performance Studies

The parameters of the proposed algorithm are set as follows: M=10, ξ=0.1, β=1, q_0=0.9 and ψ=20. These configurations are based on our previous studies of ACO for solving the deterministic project scheduling problem without uncertainties [11]. We find that these configurations still contribute to good performance with respect to the S-MRCPSPDCF.

Here we first study the performance of the algorithm under different sample sizes *Na* and *Nb*. The results of six configurations for *Na* and *Nb* on 33 instances are reported in Table I. In the table, *Nb*(I) means a special scheme to increase *Nb* linearly from 1 to 10 with the growing iteration number [33]. In the experiment, the algorithm with each configuration is run for 30 times. Each run is terminated when the total number of sampled scenarios has reached a predefined maximum value *MAX*. In addition, to compare the performance of different runs, 10000 scenarios are randomly sampled before executing the algorithm. For every schedule outputted by the algorithm, its NPV expectation is estimated based on the simulation on these 10000 scenarios. The table reports the averaged results of each configuration on 30 runs.

From the table, it can be seen that with *Na*=*Nb*=1, the simulation is too rough to reflect the real quality of schedules. Thus the performance is very unstable. On the other hand, if $Na \geq 2$ or $Nb \geq 10$, although the simulation is more accurate, the computational cost is also increased, so that the search speed of the algorithm becomes slow. Overall, the configuration with *Na*=1 and increasing *Nb* is able to achieve the best performance. Note that the algorithm only needs a single random scenario in the first-phase simulation and not larger than ten scenarios in the second-phase simulation, which is not computational intensive.

TABLE II COMPARISON WITH THE DETERMINISTIC APPROACH

Size	group a		group b		group c	
	ACS	MC-ACS	ACS	MC-ACS	ACS	MC-ACS
13	2215	**2359#**	**2577**	2569	2420	**2466**
16	2942	**2972**	1137	**1199**	3670	**3730**
18	**2518**	2508	3353	**3947#**	3360	**3447#**
28	2663	**3386#**	488	**818#**	5420	**6098#**
36	2415	**2978#**	1715	**3017#**	562	**2206#**
40	7601	**9279#**	-2582	**-754#**	-1661	**451#**
48	-1228	**-582**	956	**1417**	5529	**7147#**
58	8547	**9545#**	5059	**8544#**	7129	**9289#**
60	-1479	**448#**	3842	**5467#**	2865	**5499#**
80	-3397	**-73#**	1602	**6101#**	-628	**3810#**
98	1278	**3469#**	-5584	**-1116#**	-1830	**1594#**

The difference between the results of ACS and MC-ACS is significant suggested by t-test at α=0.05

In addition, we also compare the MC-ACS algorithm with a deterministic ACS algorithm. To approximate the expected NPV, the deterministic algorithm is characterized by using the NPV under a scenario where every random variable is realized to its expected value. In the comparison, the maximal number of scenarios sampled by MC-ACS and the maximal number of NPV evaluations in the deterministic ACS are all bounded by the corresponding "*MAX*" given in Table I.

According to the results given in Table II, it is apparent that MC-ACS outperforms the deterministic ACS. Out of the 33 instances, MC-ACS achieves significantly better results suggested by t-test on 25 instances. In particular, for the instances with more than 58 activities, MC-ACS obtains significantly better results on all cases. These results demonstrate that the proposed algorithm is effective.

V. CONCLUSION

An MC-ACS algorithm has been proposed to solve the S-MRCPSPDCF. As the considered scheduling model contains random variables, MC-ACS uses MC simulations to estimate the expected NPV of cash flows. In the experiment, it is found that only a rough simulation based on a small number of random scenarios is already enough for balancing the simulation accuracy and computational cost of the algorithm. Therefore, the algorithm is effective to find promising baseline schedules. For future research, designing

local search methods and developing adaptive schemes to tune the sample size will be some potential ways to further enhancing the performance of the algorithm.

REFERENCES

[1] W. Herroelen, B. De Reyck, E. Demeulemeester, "Resource-constrained project scheduling, a survey of recent developments," *Computers Operations Research*, vol. 13, no. 4, pp. 279-302, 1998.

[2] P. Brucker, A. Drexl, R. Mohring, K. Neumann, E. Pesch, "Resource-constrained project scheduling: notation, classification, models and methods", *European Journal of Operational Research*, 112, pp. 3–41, 1999.

[3] J. Blazewicz, J.K. Lenstra, A.H.G. Rinnooy Kan, "Scheduling subject to resource constraints," *Discrete Applied Mathematics*, vol. 5, pp. 11-24, 1983.

[4] L. Özdamar, "A genetic algorithm approach to a general category project scheduling problem," *IEEE Transactions on Systems, Man, and Cybernetics – Part C*, vol. 29, no. 1, pp. 44-59, 1999.

[5] L.-Y. Tseng and S.-C. Chen, "Two phase genetic local search algorithm for the multimode resource-constrained project scheduling problem," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 848-857, 2009.

[6] R. Kolisch, "Project scheduling under resource constraints, efficient heuristics for several problem classes," *Physica*, Heidelberg, 1995.

[7] A.H. Russell, "Cash flows in networks," *Management Science*, vol. 16, pp. 357–373, 1970.

[8] W. Herroelen, B. De Reyck, E. Demeulemeester, "Project network models with discounted cash flows: a guided tour through recent developments," *European Journal of Operational Research,* vol. 100, pp. 97-121, 1997.

[9] G. Ulusoy, "Four payment models for the multi-mode resource constrained project scheduling problem with discounted cash flows", *Annals of Operations Research*, vol. 102, pp. 237-261, 2001.

[10] M. Mika, G. Waligóra, J. Węglarz, "Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models," *European Journal of Operational Research,* vol. 164, no. 3, pp. 639-668, August, 2005.

[11] W.-N. Chen, J. Zhang, H. S.-H. Chung, R.-Z. Huang and O. Liu. "Optimizing discounted cash flows in project scheduling – an ant colony optimization approach," *IEEE Transactions on Systems, Man, and Cybernetics – Part C*, vol. 40, no. 1, pp. 64-77, 2010.

[12] R. Kolisch and R. Padman, "An integrated survey of deterministic project scheduling," *The International Journal of Management Science*, vol. 29, pp. 249-272, 2001.

[13] W. Herroelen and R. Leus, "Project scheduling under uncertainty: survey and research potentials," *European Journal of Operational Research*, vol. 165, no. 2, pp. 289-306, 2005.

[14] Z. Li and M. Ierapetritou, "Process scheduling under uncertainty: review and challenges," *Computers and Chemical Engineering*, vol. 32, pp. 715-727, 2008.

[15] A.J. Davenport and J.C. Beck, "A survey of techniques for scheduling with uncertainty," unpublished manuscript available in http://www.eil.utoronto.ca/profiles/chris/gz/uncertainty-survey.ps, 2002.

[16] N.V. Sahinidis, "Optimization under uncertainty: state-of-the-art and opportunities," *Computers and Chemical Engineering*, vol. 28, pp. 971-983, 2004.

[17] G.E. Vieira, J.W. Herrmann, E. Lin., "Rescheduling manufacturing systems: A framework of strategies, policies and methods," *Journal of Scheduling*, vol. 6, pp. 39 – 62. 2003.

[18] R.H. Möhring, and F. Stork, F, "Linear preselective policies for stochastic project scheduling," *Mathematical Methods of Operations Research*, vol. 52, pp. 501 – 515, 2000.

[19] F. Stork, "Branch-and-bound algorithms for stochastic resource-constrained project scheduling," Research Report No. 702/2000, Technische Universität Berlin, 2000.

[20] Y.-W. Tsai, D. Gemmill, "Using tabu search to schedule activities of stochastic resource-constrained projects," *European Journal of Operational Research*, vol. 111, no. 1, pp. 129-141, 1998.

[21] H. Ke and B. Liu, "Project scheduling problem with stochastic activity duration times," *Applied Mathematics and Computation*, vol. 168, no. 1, pp. 342-353, 2005.

[22] Z.X. Guo, W.X. Wong, S.Y.S Leung, J.T. Fan, S.F. Chan, "Genetic optimization of order scheduling with multiple uncertainties," *Expert Systems with Applications*, vol. 35, no. 4, pp. 1788-1801, 2008.

[23] L.D. Long, A. Ohsato, "Fuzzy critical chain method for project scheduling under resource constraints and uncertainty," *International Journal of Project Management*, vol. 26, no. 6, pp. 688-698, 2008.

[24] R.L. Daniels, J.E. Carrillo, "*β*-robust scheduling for single-machine systems with uncertain processing times," *IIE Transactions*, vol. 29, pp. 977 – 985, 1997.

[25] S. Creemers, R. Leus, B.De Reyck, M. Lambrecht, "Project scheduling for maximum NPV with variable activity durations and uncertain activity outcomes," *IEEE International Conference on Industrial Engineering and Engineering Management*, 2008.

[26] S. Creemers, R. Leus, and M. Lambrechts, "Scheduling Markovian PERT networks with maximum-NPV objective," Department of Decision Sciences and Information Management, FBE, KULeuven, Leuven, Belgium, Tech. Rep. 0811, 2008.

[27] M.J. Sobel, J.G. Szmetekovsky, V. Tilson. "Scheduling projects with stochastic activity duration to maximize expected net present value," *European Journal of Operational Research*, vol. 198, no. 3, pp. 697-705, 2009.

[28] W. Wiesemann, D. Kuhn, B. Rustem, "Maximizing the net present value of a project under uncertainty," *European Journal of Operational Research*, vol. 202, no. 2, pp. 356-367, 2010.

[29] M. Dorigo, V. Maniezzo, A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems Man, and Cybernetics - part B,* vol. 26, pp. 29-41, 1996.

[30] M. Dorigo, L. M. Gambardella, "Ant colony system: A cooperative learning approach to TSP", *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 53–66, 1997.

[31] D. Merkle, M. Middendorf, H. Schmeck, "Ant colony optimization for resource-constrained project scheduling," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 333-346, 2002.

[32] W.J. Gutjahr, "S-ACO: An ant based approach to combinatorial optimization under uncertainty," *ANTS-IV, LNCS,* vol. 3172, pp. 238–249, Berlin, Germany, Springer-Verlag, 2004.

[33] W. J. Gutjahr, "A converging ACO algorithm for stochastic combinatorial optimization," *In Proceedings of the 2nd Symposium on Stochastic Algorithms, Foundations and Applications (SAGA 2003)*, *LNCS*, vol. 2827, pp. 10-25, Springer, Berlin, Germany, 2003.

[34] P. Balaprakash, "Ant colony optimization under uncertainty," IRIDIA Technical Report No. TR/IRIDIA/2005-028, 2005.

[35] R. Kolisch and S. Hartmann, "Heuristic algorithms for solving the resource-constrained project scheduling problem: Classification and computational analysis," in *Handbook on Recent Advances in Project Scheduling*, J. Weglarz, Ed. Amsterdam, The Netherlands: Kluwer, pp. 197-212, 1999.