Neurocomputing 148 (2015) 129–135

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A new probability model for insuring critical path problem with heuristic algorithm

Zhenhong Li, Yankui Liu *, Guoqing Yang

College of Mathematics & Computer Science, Hebei University, Baoding 071002, Hebei, China

ARTICLE INFO

Article history:
Received 4 April 2012
Received in revised form
1 June 2012
Accepted 4 July 2012
Available online 30 July 2014

Keywords:
Critical path
Probability model
Stochastic optimization
Minimum risk criterion
Hybrid algorithm

ABSTRACT

In order to obtain an adequate description of risk aversion for insuring critical path problem, this paper develops a new class of two-stage minimum risk problems. The first-stage objective function is to minimize the probability of total costs exceeding a predetermined threshold value, while the second-stage objective function is to maximize the insured task durations. For general task duration distributions, we adapt sample average approximation (SAA) method to probability objective function. The resulting SAA problem is a two-stage integer programming model, in which the analytical expression of second-stage value function is unavailable, we cannot solve it by conventional optimization algorithms. To avoid this difficulty, we design a new hybrid algorithm by combining dynamic programming method (DPM) and genotype-phenotype-neighborhood based binary particle swarm optimization (GPN-BPSO), where the DPM is employed to find the critical path in the second-stage programming problem. We conduct some numerical experiments via a critical path problem with 30 nodes and 42 arcs, and discuss the proposed risk averse model and the experimental results obtained by hybrid GPN-BPSO, hybrid genetic algorithm (GA) and hybrid BPSO. The computational results show that hybrid GPN-BPSO achieves the better performance than hybrid GA and hybrid BPSO, and the proposed critical path model is important for risk averse decision makers.

1. Introduction

In a complex project management problem, we often use a directed network graph to describe various tasks and the relationships among the tasks. In this framework, the arcs represent dependent tasks and the arc weights serve as associated task durations. Also, there exists a node 0 representing the start of the project and a node n representing its termination. A project can be considered completed if all its activities have been finished. An important theoretical result is that the minimum time to complete all the activities in the activity network equals to the length of the longest path from the source node to the destination node [1]. Thus, this path, called *critical path*, represents the sequence of activities, which will take the longest time to complete. Chen et al. [2] developed a polynomial time algorithm to find the critical path and analyzed the float of each arc in a time-constrained activity network. Guerriero and Talarico [3] proposed a general method to find the critical path in a deterministic activity-on-the-arc network, considering three different types of time constraints. Another area of research dealt with the stochastic nature of activity time. For example, Kelley [4,5] and Moehring [6] estimated the probability that a project would be completed by a given deadline if the duration for each activity is not known with certainty; Burt and Garman [7], Bowman [8] and Mitchell and Klastorin [9] treated mass uncertain information by heuristic-based and Monte Carlo simulation-based techniques, and Shen et al. [10] proposed expectation and chance-constrained models for insuring critical path problems and designed decomposition strategies to solve these models.

In this paper, we approach the insuring critical path problem from a new viewpoint. It is known that the appropriateness of expectation criterion for insuring critical path problem depends on the assumption that the insuring process can be repeated a great number of times, this implies by the law of large numbers that in the long run the average cost will be equal to the expected cost. But, this assumption will often not be justified and thus the expected cost may not be of much interest to risk averse decision makers. On the other hand, the optimal solution of the expected value problem may only assure the achievement of the corresponding expected cost with a relatively small probability. Consequently, the risk averse decision maker will not consider the solution of the expected value problem to be optimal. Instead, what may be desired is a solution ensuring a low probability of very large costs. These considerations lead us to adopt minimum risk criterion in insuring critical path problems. In the proposed

* Corresponding author. Tel./fax: +86 312 5066629.
E-mail addresses: lizhenhong6688@163.com (Z. Li), yliu@hbu.edu.cn (Y. Liu), ygqfq100@gmail.com (G. Yang).

http://dx.doi.org/10.1016/j.neucom.2012.07.061
0925-2312/

risk averse two-stage stochastic insuring critical path problem, the first-stage objective function is to minimize the probability of total costs exceeding a predetermined threshold value, while the second-stage objective function is to maximize the insured task durations. For general task duration distributions, we adapt the SAA method to probability objective function and turn the original insuring critical path problem into its associated SAA one. This approximation method for chance constrained model and expected value model have been discussed in [11–13]. Since the resulting SAA model belongs to the class of NP-hard problems, we cannot solve it by conventional optimization algorithms. In this paper, we will employ evolutionary algorithms (EAs) to solve the resulting SAA critical path problem.

EAs are stochastic search methods that have been used in a variety of fields. Among existing EAs, GA [14] and PSO [15,16] are the well-known tools for solving complex optimization problems, and many modified and improved GA and PSO as well as their successful applications can be found in the literature. For example, Zeng et al. [17] proposed a dynamic chain-like agent GA for solving global numerical optimization problem; He and Tan [18] proposed a two-stage GA and applied it to automatic clustering; Lee et al. [19] modified and improved BPSO; Nanni and Lumini [20] proposed an efficient method based on PSO for finding a good set of prototypes; Qin and Liu [21,22] designed Monte Carlo simulation-based GAs to solve stochastic data envelopment analysis problems, and Liu et al. [23] solved stochastic portfolio selection problems by Monte Carlo simulation-based PSO algorithms. Motivated by the work mentioned above, this paper designs a new hybrid algorithm by combining DPM and GPN-BPSO, where DPM is employed to find the critical path in the second-stage programming problem. In our designed algorithm, we adopt the concept of genotype–phenotype in biology. The genotype means genetic messages carried by the individual's genes, and the phenotype denotes all the observable characteristics of an individual such as physical appearance and internal physiology. To demonstrate the effectiveness of the proposed method, we conduct some numerical experiments via a critical path problem with 30 nodes and 42 arcs. We first solve our critical path problem by hybrid GPN-BPSO, then compare its solution results with those obtained by hybrid GA and hybrid BPSO. We also discuss the difference between the proposed risk averse insuring critical path model and the traditional risk neural model via numerical experiments.

The remainder of this paper is organized as follows: Section 2 presents a new class of risk averse two-stage stochastic insuring critical path problems. In Section 3, we adapt the SAA method to probability objective function, and turn the original insuring critical path problem into its associated SAA one, which can be reformulated as a two-stage integer programming model by introducing additional binary variables. To solve the resulting SAA critical path problem, Section 4 designs a new hybrid algorithm by integrating DPM and GPN-BPSO. Section 5 provides one critical path problem with 30 nodes and 42 arcs and performs some numerical experiments to demonstrate the effectiveness of the designed hybrid GPN-BPSO. Section 6 gives detailed discussions about the proposed insuring critical path model and the experimental results. Finally, Section 7 gives the conclusions.

2. Formulation of risk averse two-stage stochastic insuring critical path problem

In this section, we will construct a risk averse two-stage stochastic optimization model for insuring critical path problem. For this purpose, we adopt the following notations to describe our problem.

Indices:

i: index of nodes, $i \in N$;
j: index of nodes, $j \in N$.

Parameters:

$N=\{0,1,\ldots,n\}$: the set of nodes in the network;
A: the set of arcs in the network, $A \subset N \times N$, where A is topologically ordered such that $(i,j) \in A$ only if $i<j$;
$G(N,A)$: the directed graph representing the tasks to be completed in a complex project;
$FS(i)=\{j|(i,j) \in A\}$: the set of nodes adjacent from node i, $\forall i \in N$;
$RS(i)=\{j|(j,i) \in A\}$: the set of nodes adjacent to node i, $\forall i \in N$;
Ω: the set of possible scenarios;
ω: a scenario of Ω;
c_{ij}: the cost of insuring arc $(i,j) \in A$;
d_{ij}^{ω}: an uninsured task duration of arc $(i,j) \in A$ in scenario ω;
g_{ij}^{ω}: an insured task duration of arc $(i,j) \in A$ in scenario ω;
Θ: a nondecreasing function of task completion time that penalizes the critical path length in the second stage for each scenario ω;
ξ: the random vector obtained by piecing together all random task durations in the network;
$\overline{\varphi}$: a predetermined maximum allowable cost.

Decision variables:

x_{ij}: 1 if arc (i,j) is insured, and 0 otherwise;
x: a decision vector (x_{ij}) in $\{0,1\}^{|A|}$ with $|A|$ being the number of arcs in the network;
y_{ij}^{ω}: 1 if arc (i,j) is part of one identified critical path in scenario ω, and 0 otherwise.

The second-stage objective function:

The second-stage objective function is to maximize the sum of the insured task durations:

$$\max \sum_{(i,j) \in A} (d_{ij}^{\omega}-(d_{ij}^{\omega}-g_{ij}^{\omega})x_{ij})y_{ij}^{\omega}.$$

The second-stage constraints:

The first constraint imposes a single-assignment rule:

$$\sum_{j \in FS(0)} y_{0j}^{\omega}=1.$$

The second constraint enforces flow-balance constraints for critical path contiguity:

$$\sum_{j \in FS(i)} y_{ij}^{\omega}-\sum_{l \in RS(i)} y_{li}^{\omega}=0, \forall i \in N\backslash\{0,n\}.$$

The third constraint bounds a binary decision variable:

$$y_{ij}^{\omega}=\begin{cases}1 & \text{if arc } (i,j) \text{ is part of an identified critical path in scenario } \omega\\ 0 & \text{otherwise.}\end{cases}$$

Hence, the second-stage programming problem can be built as follows:

$$\begin{cases}\max & \sum_{(i,j) \in A} (d_{ij}^{\omega}-(d_{ij}^{\omega}-g_{ij}^{\omega})x_{ij})y_{ij}^{\omega}\\ \text{subject to}: & \sum_{j \in FS(0)} y_{0j}^{\omega}=1\\ & \sum_{j \in FS(i)} y_{ij}^{\omega}-\sum_{l \in RS(i)} y_{li}^{\omega}=0, \forall i \in N\backslash\{0,n\}\\ & y_{ij}^{\omega} \in \{0,1\}, \quad \forall (i,j) \in A.\end{cases} \tag{1}$$

The first-stage objective function:

For each arc $(i,j)\in A$, we represent the cost of insuring (i,j) by c_{ij}, and define the binary decision variable x_{ij}, where $x_{ij}=1$ if we insure arc (i,j) and $x_{ij}=0$ otherwise. Thus, the insuring cost in the first stage is $\sum_{(i,j)\in A}c_{ij}x_{ij}$. In our insuring critical path problem, we assume that the critical path lengths are carefully monitored since financial penalties accrue as a nondecreasing function of task completion time. More precisely, we define a nondecreasing function Θ that penalizes the critical path length in the second stage for each scenario and fixed first-stage decision $x=(x_{ij})$. Thus, the penalizing cost in the second stage is $\Theta(Q(x,\xi(\omega)))$ for scenario ω, where $Q(x,\xi(\omega))$ is the optimal value of problem (1), representing the critical path length in scenario ω.

As a consequence, the total costs incurred in the two stages can be represented as the following random function:

$$f(x,\omega)=\sum_{(i,j)\in A}c_{ij}x_{ij}+\Theta(Q(x,\xi(\omega))),$$

where $x=(x_{ij}),(i,j)\in A$. Traditional stochastic programming is risk neutral in the sense that it is concerned with the optimization of an expectation criterion for random cost $f(x,\omega)$, in which the sum of the first-stage cost and the expected value of second-stage cost is minimized. This leads to the following objective function:

$$\min\ \sum_{(i,j)\in A}c_{ij}x_{ij}+E[\Theta(Q(x,\xi(\omega)))].$$

The appropriateness of expectation criterion for insuring critical path problem depends on the assumption that the insuring process can be repeated a great number of times, this implies by the law of large numbers that in the long run the average cost will be equal to the expected cost. But, this assumption will often not be justified and thus the expected cost may not be of much interest to risk averse decision makers. In addition, the optimal solution of the expected value problem may only assure the achievement of the corresponding expected cost with a relatively small probability. These considerations imply that the risk averse decision makers will not consider the solution of the expected value problem to be optimal. Instead, what may be desired is a solution ensuring a low probability of very large costs. The minimum risk insuring critical path problem is the problem of minimizing the probability of total costs exceeding a predetermined threshold value $\overline{\varphi}$, which may be the level of bankruptcy or budget limit. Formally, the first-stage objective function reads

$$\min \Pr\left\{\sum_{(i,j)\in A}c_{ij}x_{ij}+\Theta(Q(x,\xi(\omega)))>\overline{\varphi}\right\},$$

and the first-stage programming problem can be built as

$$\begin{cases}\min & \Pr\left\{\sum_{(i,j)\in A}c_{ij}x_{ij}+\Theta(Q(x,\xi(\omega)))>\overline{\varphi}\right\}\\ \text{subject to}: & x_{ij}\in\{0,1\},\quad \forall(i,j)\in A.\end{cases}\tag{2}$$

Combining (1) and (2), we can formally build a risk averse two-stage stochastic insuring critical path programming model as follows:

$$\begin{cases}\min & \Pr\left\{\sum_{(i,j)\in A}c_{ij}x_{ij}+\Theta(Q(x,\xi(\omega)))>\overline{\varphi}\right\}\\ \text{subject to}: & x_{ij}\in\{0,1\},\quad \forall(i,j)\in A,\end{cases}\tag{3}$$

where $Q(x,\xi(\omega))$ is the optimal value of the following 0–1 programming problem:

$$\begin{cases}\max & \sum_{(i,j)\in A}(d_{ij}^{\omega}-(d_{ij}^{\omega}-g_{ij}^{\omega})x_{ij})y_{ij}^{\omega}\\ \text{subject to}: & \sum_{j\in FS(0)}y_{0j}^{\omega}=1\\ & \sum_{j\in FS(i)}y_{ij}^{\omega}-\sum_{l\in RS(i)}y_{li}^{\omega}=0,\ \forall i\in N\backslash\{0,n\}\\ & y_{ij}^{\omega}\in\{0,1\},\ \forall(i,j)\in A.\end{cases}\tag{4}$$

We next define the feasible region of two-stage minimum risk problem (3).

First, suppose that the first-stage decision variable $x=(x_{ij})$ has to satisfy the deterministic constraint

$$D_1=\{x\in\{0,1\}^{|A|}|x_{ij}\in\{0,1\},\forall(i,j)\in A\},\tag{5}$$

where $|A|$ is the number of arcs in the network.

Second, to define the feasible solution of problem (3), we introduce additional constraints on x. Let D_2 be the set of all those x vectors in $\{0,1\}^{|A|}$ for which problem (4) has a feasible solution $y=(y_{ij}^{\omega})$ for almost every possible realized value $\xi(\omega)$ of random vector ξ, that is the critical path can be found for almost every scenario ω. If we denote $Q(x,\xi(\omega))$ as the critical path length of problem (4), then we can express D_2 as follows:

$$D_2=\{x\in\{0,1\}^{|A|}|\Pr\{Q(x,\xi)<\infty\}=1\}.\tag{6}$$

Finally, the feasible region D of problem (3) is defined as $D=D_1\cap D_2$.

Note that in problem (4), the objective function is to maximize the sum of task durations, the first constraint imposes the single-assignment rule, the second constraint enforces the flow-balance constraints for critical path contiguity, and the last constraint bounds the y variables between 0 and 1. Then the critical path of problem (4) can always be found for every scenario and first-stage decision. That is, problem (3) is a complete recourse two-stage stochastic integer programming problem (see, [24]). Therefore, $D_2=\{0,1\}^{|A|}$, and the feasible region of problem (3) is

$$D=\{x\in\{0,1\}^{|A|}|x_{ij}\in\{0,1\},\forall(i,j)\in A\}.\tag{7}$$

Before ending this section, we highlight the difficulty about the solution of problem (3). One of the reasons is that for a given first-stage decisions (x_{ij}), the quantity of probability objective function

$$\Pr\left\{\sum_{(i,j)\in A}c_{ij}x_{ij}\ +\ \Theta(Q(x,\xi(\omega)))>\overline{\varphi}\right\}$$

is hard to evaluate since it requires the computation of multi-dimensional integration. In the next section, we discuss the approximation approach to evaluating this probability objective function based on the technique of Monte Carlo simulation.

3. The SAA insuring critical path problem

In this section, we adapt the SAA method to probability objective function. As a result, we can turn the original insuring critical path problem (3) into its associated SAA model.

More precisely, let K be the number of the sample size, and $\xi^1,\xi^2,\ldots,\xi^K$ the independent identically distributed sample of K realizations of random vector ξ. Then the original probability objective function becomes the following SAA probability function:

$$\Pr\left\{\sum_{(i,j)\in A}c_{ij}x_{ij}\ +\ \Theta(Q(x,\xi^k))>\overline{\varphi}\right\}.$$

Let z be a binary vector whose components $z_k,k=1,2,\ldots,K$, take 1 if the corresponding set of constraints has to be satisfied and 0 otherwise. In addition, we introduce a large enough positive number M such that the following inequalities hold

$$\sum_{(i,j)\in A}c_{ij}x_{ij}\ +\ \Theta(Q(x,\xi^k))-M(1-z_k)>\overline{\varphi},k=1,2,\ldots,K.$$

According to the law of large numbers, the SAA probability function can be equivalent represented by

$$\frac{1}{K}\sum_{k=1}^{K}z_k.$$

As a consequence, the original insuring critical path problem can be replaced by the following SAA model

$$\begin{cases} \min & \frac{1}{K}\sum_{k=1}^{K} z_k \\ \text{subject to :} & \sum_{(i,j)\in A} c_{ij}x_{ij} + \Theta(Q(x,\xi^k)) - M(1-z_k) > \overline{\varphi}, k=1,2,\ldots,K \\ & x_{ij}\in\{0,1\}, \forall (i,j)\in A \\ & z_k \in \{0,1\}, k=1,2,\ldots,K, \end{cases} \tag{8}$$

where $Q(x,\xi^k)$ is the optimal value of the following 0–1 programming problem:

$$\begin{cases} \max & \sum_{(i,j)\in A} (d_{ij}^k - (d_{ij}^k - g_{ij}^k)x_{ij})y_{ij}^k \\ \text{subject to :} & \sum_{j\in FS(0)} y_{0j}^k = 1 \\ & \sum_{j\in FS(i)} y_{ij}^k - \sum_{l\in RS(i)} y_{li}^k = 0, \forall i \in N\backslash\{0,n\} \\ & y_{ij}^k \in \{0,1\}, \forall (i,j)\in A. \end{cases} \tag{9}$$

Problem (8) is a two-stage integer programming problem, where the analytical expression of second-stage value function $Q(x,\xi^k)$ is unavailable. Thus, problem (8) cannot be solved by conventional numerical algorithms, its heuristic solution methods are designed in the next section.

4. Heuristic solution methods

In this section, we first employ DPM to find the critical path in second-stage programming problem (9), then design a hybrid algorithm by combining DPM and GPN-BPSO to solve SAA problem (8).

4.1. Finding critical path

In order to solve problem (8), it is required to find critical path effectively in the second-stage programming problem. By the structural characteristics of critical path network and the optimality principle of dynamic programming (see, [25]), we employ DPM to update the longest path in the network. The computational formula is as follows:

$$f(j) = \max_{(i,j)\in A}\{f(i) + (d[i][j] - (d[i][j] - g[i][j])x[i][j])\}, \tag{10}$$

where $f(i)$ represents the longest path of node i from the start of the project, $d[i][j]$ represents an uninsured task duration of arc $(i,j)\in A$, $g[i][j]$ represents an insured task duration of arc $(i,j)\in A$, and $x[i][j]$ is 1 if arc (i,j) is insured, and 0 otherwise.

4.2. Hybrid GPN-BPSO

In this subsection, we design a new hybrid algorithm for the solution of problem (8), in which DPM is embedded into a GPN-BPSO. In this hybrid algorithm, the DPM is utilized to find the critical path in the second-stage programming problem, and the GPN-BPSO is used to search for the optimal location of insuring critical path problem.

In our designed algorithm, we adopt the concept of genotype-phenotype, which is widely used in biology. Generally, the genotype means genetic messages carried by the individual's genes, and the phenotype denotes all the observable characteristics of an individual such as physical appearance and internal physiology. The phenotype is determined partly by genes, the environment and the way of life. Thus, the phenotype is partly subject to the genotype, while some genetic messages are behaved by the phenotype, just like the relationship between the generation particle and velocity for determining the particle position in BPSO. Under this consideration, we employ the genotype and phenotype to represent the velocity and the binary position parameters, respectively.

On the other hand, in the original BPSO, the velocity is updated according to the direction of the personal best position and the global best position. In this paper, we consider the updating process in its neighborhoods, and adjust the velocity in the direction of the best particles in the neighborhoods and the global best particle.

Based on the considerations above, we denote the modified BPSO with genotype-phenotype-neighborhood as GPN-BPSO. Incorporating DPM into the GPN-BPSO, we can design a new hybrid GPN-BPSO for the solution of insuring critical path problem.

Solution representation: A particle is denoted by a binary integer vector $x=(x_{ij})\in\{0,1\}^{|A|}$ to represent a feasible solution of problem (8), where $|A|$ is the number of the arcs in the network. Each gene x_{ij} is either 0 or 1, where 1 represents the arc (i,j) is insured, and 0 represents the arc (i,j) is not insured.

Initialization: We randomly generate a binary phenotype position vector $x_p=(x_{p,1},x_{p,2},\ldots,x_{p,|A|})$ from the $\{0,1\}^{|A|}$, and initialize the genotype position vector as $x_g=x_p$. Repeating this process P_{size} times, we can produce P_{size} pairs of initial phenotype and genotype particles $(x_{p,1},x_{g,1}),(x_{p,2},x_{g,2}),\ldots,(x_P,P_{\text{size}},x_g,P_{size})$.

Evaluation: Let $Fit(\cdot)$ be the fitness function. Its function value is computed by

$$Fit(\cdot) = -\frac{1}{K}\sum_{k=1}^{K} z_k.$$

Neighborhood-based updating velocity, genotype and phenotype particle: In these operations, we first need to find the $P_{best,i}$ with the highest previous fitness for each phenotype particle $x_{p,i}$, and to determine the global best particle G_{best} with the highest fitness in the entire phenotype population. Then, for each i, the velocity vector $v_{i,d}$, the genotype particle $x_{g,i}$, and the phenotype particles $x_{p,i}$ are updated by the following formulas:

$$v_{i,d} = w\cdot v_{i,d} + c_1\cdot \text{rand}()\cdot D(x_{p,i}) + c_2\cdot\text{rand}()\cdot(G_{best,i} - x_{p,i}),$$

$$x_{g,i} = x_{g,i} + v_{i,d},$$

and

$$x_{p,i,j} = \begin{cases} 1 & \text{if rand}() < S(x_{g,i,j}) \\ 0 & \text{if rand}() \geq S(x_{g,i,j}), \end{cases}$$

where w is a coefficient, c_1 and c_2 are rates, rand() is randomly generated from the interval $(0,1)$, $S(x)=1/(1+e^{-x})$ is the sigmoid function, $x_{p,i,j}$ is the component of vector $x_{p,i}$, $x_{g,i,j}$ is the component of vector $x_{g,i}$ and $D(x_{p,i})$'s are the average distances from $x_{p,i}$ to the best positions in its neighbors:

$$D(x_{p,1}) = \sum_{k=1}^{2} \text{rand}()\cdot\frac{P_{best,k}-x_{p,1}}{2},$$

$$D(x_{p,i}) = \sum_{k=i-1}^{i+1} \text{rand}()\cdot\frac{P_{best,k}-x_{p,i}}{3},\ i=2,3,\ldots,P_{size}-1,$$

and

$$D(x_{p,P_{size}}) = \sum_{k=P_{size}-1}^{P_{size}} \text{rand}()\cdot\frac{P_{best,k}-x_{p,P_{size}}}{2}.$$

From the updating process above, we can obtain a new generation of phenotype particles $x'_{p,1},x'_{p,2},\ldots,x'_{p,P_{size}}$.

Hybrid algorithm procedure: Combining DPM and GPN-BPSO, we now design a new hybrid GPN-BPSO for solving critical path problem (8), and its solution process is as follows.

Algorithm 1. Hybrid GPN-BPSO.

Step 1: Set parameters w, c_1, c_2, v_{max} and P_{size}.
Step 2: Initialize randomly a population of phenotype particles.
Step 3: Update all the genotype particles and phenotype particles.
Step 4: Find the critical path in the second-stage programming problem (9) according to formula (10).
Step 5: Calculate the fitness of all the phenotype particles.
Step 6: Update the $P_{best,i}$ for each phenotype particle, and the G_{best} for the phenotype population.
Step 7: Repeat Step 3 to Step 6 for a given number of cycles.
Step 8: Report the particle G_{best} as an optimal solution.

5. Computational results

In this section, we perform some numerical experiments to demonstrate the effectiveness of hybrid GPN-BPSO. The algorithm is coded in C++ programming language and the numerical experiments are carried out on a personal computer (Lenovo with Intel Pentium(R) Dual-Core E5700 3.00GHZ CPU and RAM 2.00GB), using the Microsoft Windows 7 operating system.

Consider an insuring critical path problem with 30 nodes and 42 arcs: (0,1), (0,7), (0,15), (0,21), (1,2), (1,3), (2,4), (2,5), (3,4), (3,9), (4,6), (5,6), (6,11), (7,8), (8,9), (8,12), (9,10), (10,11), (10,14), (11,29), (12,13), (12,26), (13,14), (14,29), (15,16), (15,17), (15,18), (16,19), (17,19), (18,19), (19,20), (20,26), (21,22), (22,23), (22,24), (23,25), (24,27), (25,26), (25,28), (26,29), (27,28), (28,29), where (0,1) represents the arc from node 0 to node 1, and the rests can be explained similarly.

For each arc $(i,j) \in A$, assume its task duration follows uniform distribution over the interval [10,300]. According to practical situation, a task duration is sometimes likely to be delayed or sometimes ahead of task duration. Hence, we generate d_{ij}^k by multiplying the task duration for $(i,j) \in A$ and a random number from a uniform distribution over the interval [0.9,1.5]. For each arc $(i,j) \in A$, we randomly generate g_{ij}^k from the uniformly distributed $[0.5d_{ij}^k, 0.7d_{ij}^k]$, and generate the cost c_{ij} to insure arc $(i,j) \in A$ from a uniform distribution over the interval [25,50]. Finally, we round the values of d_{ij}^k, g_{ij}^k, and c_{ij} to their nearest integer values. In this insuring critical path problem, we set $M = 10^5$, and assume that the penalty function is given by

$$\Theta(t) = \begin{cases} 0, & 0 < t \le 800 \\ 50 + (t-800)^2/400, & 800 < t \le 900 \\ 100 + \sqrt{t-900}/50, & 900 < t \le 1000 \\ 150 + (t-1000)^2/200, & t > 1000. \end{cases}$$

We now solve the insuring critical path problem by hybrid GPN-BPSO. During the solution process, we set the population size $P_{size} = 30$, the maximum velocity $v_{max} = 2$, the learning rates $c_1 = c_2 = 2$, and the weight w decreases linearly from 0.9 to 0.4 in consecutive iterations by the following formula:

$$w = 0.5 \times \frac{GEN - gen}{GEN} + 0.4,$$

where GEN and gen represent the maximum number of iterations and the current number of iterations, respectively.

Table 1 summarizes the computational results obtained by hybrid GPN-BPSO, where K in the first column is the number of sample size; $\overline{\varphi}$ in the second column represents the threshold value of maximum allowable cost; the best insured arcs

Table 1
The results of hybrid GPN-BPSO algorithm with $GEN = 300$.

K	$\overline{\varphi}$	Best solution	Objective value	CPU (s)
1000	600	(1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.153000	19.078000
	625	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.093000	19.735000
	650	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.044000	19.485000
2000	600	(1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.150500	37.407000
	625	(1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.080000	39.188000
	650	(1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.039500	40.000000
4000	600	(1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.149250	74.391000
	625	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.081000	78.156000
	650	(1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.041750	78.468000
8000	600	(1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.149625	148.828000
	625	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.081250	159.657000
	650	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	0.043750	156.437000

Table 2
Comparison results of different approaches with $GEN = 300$ and $K = 4000$.

$\overline{\varphi}$	600	625	650
Hybrid GPN-BPSO			
Best solution	(1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0)	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0)	(1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0)
Objective value	0.149250	0.081000	0.041750
CPU (s)	74.391000	78.156000	78.468000
Hybrid GA			
Best solution	(0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0)	(0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0)	(0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0)
Objective value	0.671300	0.635000	0.578700
CPU (s)	73.125000	74.812000	75.281000
Hybrid BPSO			
Best solution	(1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1)	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1)	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1)
Objective value	0.244750	0.199500	0.206000
CPU (s)	72.891000	74.906000	74.844000

corresponding to various threshold values are listed in the third column, and their best probability values are provided in the fourth column; the fifth column reports the average CPU times.

6. Discussions

In this section, we will give detailed discussions about the proposed risk averse insuring critical path model and the experimental results.

Firstly, we discuss the experimental results. Toward this end, we will show that some other heuristic algorithms like GA may be alternatively utilized to form different hybrid solution approaches to the proposed critical path problem. To further assess the performance of the hybrid GPN-BPSO, we now compare its solution results with those obtained by hybrid GA and BPSO.

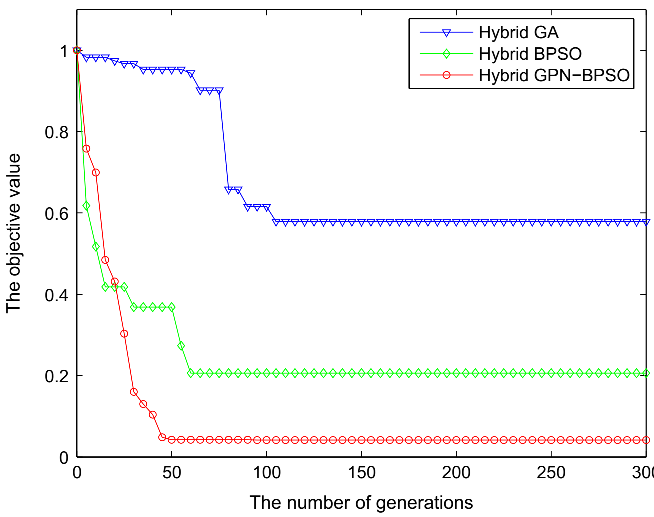


Fig. 1. Performance comparison with $\overline{\varphi}=650$, $GEN=300$ and $K=4000$.

In hybrid BPSO, we use the same data as in hybrid GPN-BPSO; in hybrid GA, the parameters are set as follows: $pop\_size=30$, number of generation $GEN=300$, crossover probability $P_c=0.3$, and mutation probability $P_m=0.2$.

With various threshold values of allowable costs, Table 2 reports the best solutions and their corresponding probability values obtained by three hybrid algorithms. The comparisons about the convergent performance of three hybrid algorithms are plotted in Fig. 1.

From Table 2 and Fig. 1, we can see that hybrid GPN-BPSO can always find the best solutions among three hybrid algorithms. On the other hand, from Table 2, hybrid GPN-BPSO needs much time among three hybrid algorithms. The reason is that hybrid GPN-BPSO includes the genotype–phenotype mechanism, which needs to operate a double update process at each generation. However, it is evident that the relative errors about the times consumed by three hybrid algorithms are very small.

From the above discussions, we conclude that hybrid GPN-BPSO achieves the better performance than hybrid GA and hybrid BPSO for our insuring critical path problem.

Secondly, we discuss the proposed risk averse insuring critical path model. Since traditional two-stage stochastic critical path problem is risk neural, it considers the expectation as a preference criterion while comparing random costs to identify the best decisions. In the current development, we consider a risk averse two-stage critical path problem, where we specify the excess probability as a risk measure. In other words, we may desire a solution ensuring a low probability of very large costs. To demonstrate our new modeling idea for insuring critical path problem, we formulate the critical path problem in Section 5 as a risk neural two-stage programming model; that is, we adopt expectation objective function instead of probability objective function. To compare the best decisions, we also solve the expected value model by hybrid GPN-BPSO, GA and BPSO respectively, and their solution results are reported in Table 3.

From Tables 2 and 3, we can find that the best solutions of expected value model are different from those of minimum risk problem. Consequently, risk averse decision makers will not

Table 3
Comparison results of risk-neural model with $GEN=300$.

K	1000	2000	4000	8000
Hybrid GPN-BPSO				
Best solution	(1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0)	(1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0)	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0)	(1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0)
Objective value	534.552164	529.849057	528.721097	529.913219
CPU (s)	19.438000	38.313000	77.219000	152.391000
Hybrid GA				
Best solution	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0)	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0)	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0)	(1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0)
Objective value	542.571200	540.262400	538.943700	530.470800
CPU (s)	19.188000	39.344000	75.922000	154.875000
Hybrid BPSO				
Best solution	(1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1)	(1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0)	(1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0)	(1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0)
Objective value	588.916990	590.815390	586.730401	590.677004
CPU (s)	19.093000	37.157000	75.969000	151.187000

consider the solutions of the expected value model to be optimal. Instead, they may consider the solutions of the proposed minimum risk model as their desired optimal solutions.

7. Conclusions

In this paper, we have studied stochastic insuring critical path problem, in which the task durations are uncertain and characterized by continuous random variables. The major new results include the following four aspects.

(i) A new risk averse two-stage stochastic insuring critical path problem was introduced, in which we adopted the minimum risk criterion in the first-stage objective function. The task durations in the second-stage programming problem are characterized by continuous random variables.
(ii) We adapted the SAA method to probability objective function, and turned the original stochastic insuring critical path problem into its approximation one. To solve the resulting SAA insuring critical path problem, we designed a new hybrid GPN-BPSO by embedding DPM into a GPN-BPSO, where DPM is used to find the critical paths in the second-stage programming problem.
(iii) We discussed the experimental results obtained by three hybrid algorithms via a critical path problem with 30 nodes and 42 arcs. We first solved our critical path problem by hybrid GPN-BPSO, then compared its solution results with those obtained by hybrid GA and hybrid BPSO. The computational results showed that hybrid GPN-BPSO achieves the better performance than hybrid GA and hybrid BPSO.
(iv) We discussed the importance of the proposed risk averse critical path model via numerical experiments. Computational results demonstrated that risk averse decision makers will not consider the solutions of the expected value model to be optimal. Instead, they may consider the solutions of the proposed minimum risk model as their desired optimal solutions.

Acknowledgments

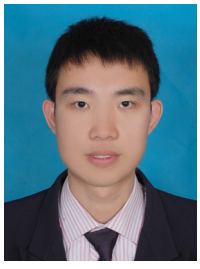
The authors wish to thank Guest Editors and anonymous referees, whose valuable comments led to an improved version of this paper. This work was supported by the National Natural Science Foundation of China (Nos. 61374184, 60974134), and the Natural Science Foundation of Hebei Province (No. A2011201007).

References

[1] J.J. Moder, C.R. Phillips, E.W. Davis, Project Management with CPM, 3rd ed., PERT and Precedence Diagramming, Van Nostrand Reinhold, New York, 1983.
[2] Y.L. Chen, D. Rinks, K. Tang, Critical path in an activity network with time constraints, Eur. J. Oper. Res. 100 (1997) 122–133.
[3] F. Guerriero, L. Talarico, A solution approach to find the critical path in a time-constrained activity network, Comput. Oper. Res. 37 (2010) 1557–1569.
[4] J.E. Kelley, Critical-path planning and scheduling: mathematical basis, Oper. Res. 9 (1961) 296–320.
[5] J.E. Kelley, The critical path method: resource planning and scheduling, Ind. Sched. 2 (1963) 347–365.
[6] R.H. Moehring, Minimizing costs of resource requirements in project networks subject to a fixed completion time, Oper. Res. 32 (1984) 89–120.
[7] J.M. Burt, M.B. Garman, Conditional Monte Carlo: a simulation technique for stochastic network analysis, Manag. Sci. 18 (1971) 207–217.
[8] R.A. Bowman, Efficient estimation of arc criticalities in stochastic activity networks, Manag. Sci. 41 (1995) 58–67.
[9] G. Mitchell, T. Klastorin, An effective methodology for the stochastic project compression problem, IIE Trans. 39 (2007) 957–969.
[10] S. Shen, J.C. Smith, S. Ahmed, Expectation and chance-constrained models and algorithms for insuring critical paths, Manag. Sci. 56 (2010) 1794–1814.
[11] A. Shapiro, T. Homem-de-Mello, A simulation-based approach to two-stage stochastic programming with recourse, Math. Program. 81 (1998) 301–325.
[12] J. Luedtke, S. Ahmed, A sample approximation approach for optimization with probabilistic constraints, SIAM J. Optim. 19 (2008) 674–699.
[13] P. Kall, J. Mayer, Stochastic Linear Programming: Models Theory and Computation, 2nd ed., Springer Science+Business Media, LLC, 2011.
[14] J.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, MI, 1975.
[15] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference on Neural Networks, 1995, pp. 1942–1948.
[16] J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm algorithm, in: Proceedings of the Conference on Systems, Man, and Cybernetics, 1997, pp. 4104–4109.
[17] X.P. Zeng, Y.M. Li, J. Qin, A dynamic chain-like agent genetic algorithm for global numerical optimization and feature selection, Neurocomputing 72 (2009) 1214–1228.
[18] H. He, Y. Tan, A two-stage genetic algorithm for automatic clustering, Neurocomputing 81 (2012) 49–59.
[19] S. Lee, S. Soak, S. Oh, W. Pedrycz, M. Jeon, Modified binary particle swarm optimization, Prog. Nat. Sci. 18 (2008) 1161–1166.
[20] L. Nanni, A. Lumini, Particle swarm optimization for prototype reduction, Neurocomputing 72 (2009) 1092–1097.
[21] R. Qin, Y. Liu, Modeling data envelopment analysis by chance method in hybrid uncertain environments, Math. Comput. Simul. 80 (2010) 922–950.
[22] R. Qin, Y. Liu, A new data envelopment analysis model with fuzzy random inputs and outputs, J. Appl. Math. Comput. 33 (2010) 327–356.
[23] Y. Liu, X. Wu, F. Hao, A new chance-variance optimization criterion for portfolio selection in uncertain decision systems, Expert Syst. Appl. 39 (2012) 6514–6526.
[24] J.R. Birge, F. Louveaux, Introduction to Stochastic Programming, 2nd ed., Springer Science+Business Media, LLC, 2011.
[25] R.E. Bellman, Dynamic Programming, Princeton University Press, New Jersey, 1957.

Zhenhong Li was born in Hebei, China in 1986. She received her B.S. degree from the Department of Mathematics and Information Technology, Tangshan Normal College in 2010. After receiving her M.S. degree in Operations Research from Hebei University in 2013, she has been a Ph.D. candidate in Management Science and Engineering at Tianjin University. Her research interests include stochastic optimization theory and its application in engineering problems.

Yankui Liu was born in Hebei, China in 1964. He received his Ph.D. degree in computational mathematics from Department of Mathematical Science, Tsinghua University in 2002. He is now a Professor and Ph.D. Supervisor at College of Mathematics and Computer Science, Hebei University. His research interests include optimization under uncertainty and its applications to engineering and management problems. His interests concentrate on these and related issues of modeling and computation, at both theoretical and applied levels. Dr. Liu is the Editor-in-Chief of Journal of Uncertain Systems.



Guoqing Yang was born in Hebei, China in 1986. He received his B.S. degree from College of Mathematics and Computer Science, Hebei University in 2009. After receiving his M.S. degree in Operations Research from Hebei University in 2013, he has been a Ph.D. candidate in Management Science and Engineering at Tianjin University. His research interests include supply chain network design and intelligent algorithms.