Proceedings of the 2021 Winter Simulation Conference
S. Kim, B. Feng, K. Smith, S. Masoud, Z. Zheng, C. Szabo, and M. Loper, eds.

SENSITIVITY ANALYSIS AND TIME-COST TRADEOFFS IN STOCHASTIC ACTIVITY NETWORKS

Peng Wan

Department of Mathematics
Institute for Systems Research
University of Maryland
College Park, MD 20742, USA

Michael C. Fu

Robert H. Smith School of Business
Institute for Systems Research
University of Maryland
College Park, MD 20742, USA

Steven I. Marcus

Department of Electrical and Computer Engineering
Institute for Systems Research
University of Maryland
College Park, MD 20742, USA

ABSTRACT

Using Monte Carlo simulation, this paper proposes a new estimator for the gradient of the first moment of project completion time. Combining the new stochastic gradient estimator with a Taylor series approximation, a functional estimation procedure for activity criticality and expected project completion time is proposed and applied to optimization problems involving time-cost tradeoffs.

1 INTRODUCTION

Stochastic activity networks have been widely used in project management and digital circuit design (Kim et al. 2007). A stochastic activity network (SAN) is a directed acyclic graph with arcs representing activities and the direction of arcs representing the precedence relationships between different activities. For example, all arcs whose tail node is i have to be completed in order for the directed arc with head node i to start. The lengths of arcs represent the length of the time for completing the activities and are assumed to be independent random variables with known cumulative distribution function $F_i(x)$ and distribution parameter θ_i. In a SAN, a path is a route starting from the source node and ending at the sink node. The length of a path is the sum of the lengths of all arcs on the path. The path with the longest length is called the critical path. The length of the critical path represents the project completion time, which is of interest to project managers and is to be minimized.

Much of existing research focuses on estimating the distribution and expectation of the total project completion time (Sigal et al. 1979) and their stochastic gradient estimators (Bowman 1994; Fu 2006), the criticality index of paths (Sigal et al. 1979; Bowman 1994), the criticality index of arcs (Bowman 1995) and their stochastic gradients (Wan and Fu 2020). Cho and Yum (2004) proposed applying logistic regression to have a functional fit of the arc criticality curve and showed that their approach underestimates the change of expected project completion time compared with direct Monte Carlo simulation. Bowman (1994) proposed a time-cost tradeoff optimization problem in PERT networks and provided a heuristic algorithm based on a stochastic gradient of the expectation of project completion time.

In this paper, we present a new Taylor series approximation approach to fit the arc criticality curve and illustrate using simulation experiments how our approach improves upon Cho and Yum (2004) and

978-1-6654-3311-2/21/$31.00 ©2021 IEEE

direct Monte Carlo simulation. Also, we will provide a new heuristic algorithm for the time-cost tradeoff optimization problem using the functional criticality curve estimation.

In section 2, we present the notation and problem setting. In section 3, the Threshold Arc Criticality (TAC) estimator and its higher order gradient estimators are presented. In section 4, the IPA gradient estimators for expected project completion time are presented. In section 5, applications of the gradient estimators to the time-cost tradeoff optimization problem are presented. Section 6 describes the simulation experiments and results. Section 7 contains conclusions and future research.

2 PROBLEM SETTING AND NOTATION

Assume the stochastic network has m arcs and n nodes. In the stochastic activity network, the completion times of different tasks are assumed to be independent with known distributions. We also assume throughout that the critical path denotes the path with the longest length. We define:

$X_i =$ length of arc i (with realizations x_i); $\{X_i\}_{i=1}^m$ are assumed independent,

$P^* =$ set of arcs on the critical path,

$T = T(X_1, ..., X_m) = \sum_{j \in P^*} X_j,$

$\Delta_i \mathbb{E}[T](\delta) = \mathbb{E}[T]\Big|_{\mu_i=\mu} - \mathbb{E}[T]\Big|_{\mu_i=\mu-\delta},$

$F_i(x) = \Pr(X_i \le x) =$ probability distribution function for the length of arc i,

$\theta_i =$ parameter in F_i; in this paper, only consider $\theta_i = \mu_i = \mathbb{E}[X_i]$ throughout,

$\bar{F}_i(x) = 1 - F_i(x),$

$f_i(x) =$ probability density function for arc i,

$C_a(i) = \Pr\{\text{arc } i \text{ is on the critical path}\} =$ criticality of arc i,

$C_a(i|\cdot) = \Pr\{\text{arc } i \text{ is on the critical path}|\cdot\},$

$C_i(x) = C_a(i)|_{\mu_i=x},$

$\mathscr{P} =$ set of all paths, $\mathscr{P}_i = \{P \in \mathscr{P} | i \in P\} =$ set of paths containing arc i,

$\mathscr{P}_{i-} = \{P \in \mathscr{P} | i \notin P\} =$ set of paths not containing arc i,

$\|\cdot\|$ is an operator that calculates the length of the longest path for a given set of paths,

$\|\cdot\|^i$ is an operator that calculates the length of the longest path for a given set of paths under the condition that arc i has length 0,

$|\mathscr{P}| =$ number of elements in set $\mathscr{P}$,

$\mathscr{R}_i = |\mathscr{P}_i|/|\mathscr{P}| =$ fraction of all paths containing arc i,

$|P| =$ length of path P,

$P^* = \arg\max_{P \in \mathscr{P}} |P| =$ set of arcs on the longest path corresponding to path set $\mathscr{P}$,

To illustrate the notation, the example in Figure 1 depicts a stochastic network with 4 nodes and 5 arcs, where each node represents an activity in a project and each arc represents the length of the time required

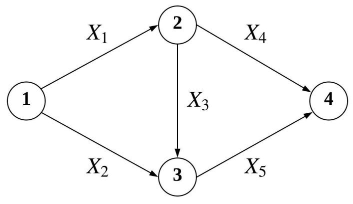

Figure 1: A small stochastic network.

to complete an activity. Then,

$$\mathscr{P} = \{(1,3,5),(2,5),(1,4)\}$$
$$\mathscr{R}_2 = 1/3$$

3 STOCHASTIC GRADIENTS OF ARC CRITICALITIES

3.1 Threshold Arc Criticalities

Wan and Fu (2020) proposed a new arc criticality estimator called the Threshold Arc Criticality (TAC) estimator, by conditioning on all arc lengths except for the arc of interest.

The TAC estimator is derived by conditioning:

$$\begin{aligned} C_a(i) &= \mathbb{E}[\mathbb{E}[I\{\text{arc } i \text{ is on critical path}\}|X_j, j \neq i]] \\ &= \int\cdots\int_{\mathbb{R}^{m-1}} C_a(i|X_j = x_j, 1 \le j \le m, j \neq i) \times \prod_{\substack{j=1 \\ j\neq i}}^{m} f_j(x_j)\,dx_j \\ &= \int\cdots\int_{\mathbb{R}^{m-1}} \Pr(X_i \ge m_i) \times \prod_{\substack{j=1 \\ j\neq i}}^{m} f_j(x_j)\,dx_j \end{aligned} \tag{1}$$

where m_i is given by:

$$m_i = max(\|\mathscr{P}_{i-}\| - \|\mathscr{P}_i\|^i, 0)$$

and the TAC estimator is given by:

$$\Pr(X_i \ge m_i).$$

3.2 Stochastic Gradient of TAC

Assuming the distribution function of the length of activity i is n-times differentiable and the regularity conditions to exchange derivative with integration are satisfied, then we have:

$$\frac{\partial^n C_a(i)}{\partial \theta_i^n} = \int\cdots\int_{\mathbb{R}^{m-1}} \frac{d^n \Pr(X_i \ge m_i)}{d\theta_i^n} \times \prod_{\substack{j=1 \\ j\neq i}}^{m} f_j(x_j)\,dx_j,$$

and the IPA stochastic gradient estimator for $\frac{\partial^n C_a(i)}{\partial \theta_i^n}$ is given by:

$$\frac{d^n \Pr(X_i \ge m_i)}{d\theta_i^n}. \tag{2}$$

4 STOCHASTIC GRADIENTS OF PROJECT COMPLETION TIMES

4.1 Stochastic Gradient of First Moment of Project Completion Time

In this section, our goal is to estimate $\frac{\partial \mathbb{E}[T]}{\partial \theta_i}$. Bowman (1994) proposed a conditional IPA stochastic gradient estimator for $\frac{\partial \mathbb{E}[T]}{\partial \theta_i}$, conditioning on node release times. Here, we present a new IPA estimator conditioned on all arc lengths except for arc i (Fu 2006):

$$\begin{aligned}\frac{\partial \mathbb{E}[T]}{\partial \theta_i} &= \mathbb{E}\Big[I(i \in P^*)\frac{\partial X_i(\theta_i)}{\partial \theta_i}\Big] \\ &= \mathbb{E}\Big[\mathbb{E}\Big[I(i \in P^*)\frac{\partial X_i(\theta_i)}{\partial \theta_i}\Big| X_j, j \neq i\Big]\Big].\end{aligned}$$

From Fu (2006) we have for θ being a location parameter $\frac{\partial X_i(\theta_i)}{\partial \theta_i} = 1$ and for θ being a scale parameter $\frac{\partial X_i(\theta_i)}{\partial \theta_i} = \frac{X_i}{\theta_i}$. Thus we have the conditional IPA estimator for $\frac{\partial \mathbb{E}[T]}{\partial \mu_i}$ given by,

$$E\Big[I\{i \in P^*\}\frac{\partial X_i}{\partial \theta_i}\Big| X_j, j \neq i\Big] = \begin{cases}\Pr(X_i \geq m_i), \theta_i \text{ is a location parameter} \\ \frac{1}{\theta_i}\int_{m_i}^{+\infty} x f_i(x)\,dx, \theta_i \text{ is a scale parameter}\end{cases}$$

For most commonly used distributions, we can derive an analytical form of $\frac{1}{\theta_i}\int_{m_i}^{+\infty} x f_i(x)\,dx$. For example, if arc i's length is exponentially distributed with mean duration μ_i, then we have:

$$\frac{1}{\mu_i}\int_{m_i}^{+\infty} x f_i(x)\,dx = \frac{1}{\mu_i}(1+m_i)e^{-\frac{m_i}{\mu}},$$

which is the conditional IPA estimator for $\frac{\partial \mathbb{E}[T]}{\partial \mu_i}$.

Similarly, assuming $F_i(x) \in C^n$ (n-times continuously differentiable), we have the conditional IPA estimator for $\frac{\partial^n \mathbb{E}[T]}{\partial \mu_i^n}$ given by,

$$\begin{cases}\frac{d^{n-1}\Pr(X_i \geq m_i)}{d\theta_i^{n-1}}, \theta_i \text{ is a location parameter} \\ \frac{1}{\theta_i}\frac{d^{n-1}\int_{m_i}^{+\infty} x f_i(x)\,dx}{d\theta_i^{n-1}}, \theta_i \text{ is a scale parameter}\end{cases}$$

5 APPLICATIONS

5.1 Estimating the Change of Expected Project Completion Time

Cho and Yum (2004) claimed that $\frac{\partial \mathbb{E}(T)}{\partial \mu_i}$ cannot serve as an accurate estimate for change in $\mathbb{E}(T)$ due to a discrete change in the mean duration of activity i, μ_i. For example, in Figure 1, if the original value of activity 2's mean duration μ_2 is 25, then using $\frac{\partial \mathbb{E}(T)}{\partial \mu_2}$ to estimate the amount of change of $\mathbb{E}(T)$ when μ_2 is decreased by 5, keeping all the other activity mean durations unchanged, is not accurate.

The goal is to estimate:

$$\Delta_i \mathbb{E}[T](\delta) = \mathbb{E}[T]\Big|_{\mu_i=\mu} - \mathbb{E}[T]\Big|_{\mu_i=\mu-\delta}, \text{ for } \delta \geq 0.$$

Cho and Yum (2004) consider:

$$\Delta_i \mathbb{E}[T](\delta) = \int_{\mu-\delta}^{\mu} C_i(x)dx. \qquad (3)$$

where $C_i(x)$ is a function with input of arc i's mean duration and output of criticality index of arc i. In Cho and Yum (2004), it is assumed that all activities are normally distributed and the parameter of interest is the mean duration of activity i. An explanation of equation (3) is given by:

$$\begin{aligned}\Delta_i \mathbb{E}[T](\delta) &= \mathbb{E}[T]\Big|_{\mu_i=\mu} - \mathbb{E}[T]\Big|_{\mu_i=\mu-\delta} \\ &= \int_{\mu-\delta}^{\mu} \frac{\partial \mathbb{E}[T]}{\partial \mu_i} dx = \int_{\mu-\delta}^{\mu} C_i(x) dx. \end{aligned} \tag{4}$$

Equation (4) follows from the mean duration μ_i being a location parameter for normal distributions, and Bowman (1994) proved that for location parameters, $\frac{\partial \mathbb{E}[T]}{\partial \mu_i} = C_i(x)$. Then, to find a functional approximation of $C_i(x)$, Cho and Yum (2004) proposed using logistic regression to fit $C_i(x)$, because $C_i(x)$ is a S shaped curve. Numerical experiments indicated that their logit fitting approach underestimates $\Delta_i \mathbb{E}[T](\delta)$ when σ_i/μ_i is large, where μ_i and σ_i are the mean and standard deviation of normally distributed X_i.

We propose using a Taylor series approximation to fit $C_i(x)$ locally, assuming $F_i(x)$ is N-times continuously differentiable,

$$C_i(x) \approx \sum_{k=0}^{N} \frac{1}{k!} C_i^{(k)}(\mu)(x-\mu)^k. \tag{5}$$

where $C_i^{(k)}(\mu)$ is the k^{th} order derivative of function $C_i(x)$ at activity i's original mean duration μ, which can be estimated using the IPA estimator in Equation (2). As for the case θ_i is a scale parameter rather than a location parameter, we have the expectation version of Equation (5) given by:

$$\frac{\partial \mathbb{E}[T]}{\partial \theta_i} \approx \sum_{k=0}^{N} \frac{1}{k!} \frac{\partial^k \mathbb{E}[T]}{\partial \theta_i^k}(\theta)(x-\theta)^k. \tag{6}$$

In Cho and Yum (2004), they also claimed that using direct Monte Carlo simulation (MCS) to fit a logistic regression requires several thousands of runs to estimate $C_i(x)$ at different x values. Hence, they proposed using Taguchi Orthogonal Array experiment to reduce the number of simulation replications. Using the TAC estimator, we can solve this issue with an easy and efficient approach. Since m_i in Equation (1) does not depend on X_i, and therefore not on μ_i, no new simulation replications are needed for estimating $C_i(x)$ at a new x value.

5.2 Estimating the Criticality Curve

As mentioned in section 5.1, in order to fit a logistic regression model for $C_i(x)$, we need to estimate the $C_a(i)$ value when the mean duration of activity i takes different discrete values while the parameters of other arc length remain unchanged. For estimating arc criticality using the TAC estimator, calculating the threshold m_i is important. The following algorithm presents an efficient way of calculating the TAC threshold for one activity of interest.

```
1. *Initialization* : Specify the arc index $i$ of interest, distribution parameters of all arc lengths $\{\theta_i\}$.
2. Simulate all arc lengths.
3. Select the activity $j$ that has the largest simulated arc length $x_j$ and $D \leftarrow mx_j$, ($m$ is the number of
   arcs in the network).
4. $\mu_i \leftarrow -D$.
5. Calculate the project completion time $T$.
6. $T0 \leftarrow T$.
7. $\mu_i \leftarrow D$.
8. Calculate the project completion time $T$.
```

9. $T1 \leftarrow T$.
10. Calculate the threshold M_i, $M_i \leftarrow \max(T0-(T1-D),0)$.

The above algorithm efficiently estimates the criticality index of a fixed arc. If we are interested in estimating the criticality index of all arcs in the network at the same time, the above algorithm is not computationally efficient. Instead, we need to calculate the lengths of all paths at each simulation replication. In the case of estimating the criticality curve using TAC, the above algorithm is helpful. Suppose we are interested in estimating the criticality curve at 10 different values. The following algorithm explains the advantage of the TAC estimator:

1. Simulate all arc lengths.
2. Calculate threshold m_i.
3. Calculate estimator for $C_i(x)$, $EST \leftarrow \bar{F}_i(m_i)$.

When estimating $C_i(x)$ at different x values, steps 1 and 2 only need to be run once, and for different x values, redo step 3 with the distribution parameter of $F_i(x)$ changed. This property is called Sample Performance Invariance (SPI), because estimating the gradient at different parameter values does not depend on simulated samples. This property is advantageous for all measure-based gradient estimation methods, such as the Likelihood Ratio (LR) method and the weak derivatives method. Sample path-based gradient estimation methods such as IPA do not generally possess this property, but for the TAC estimator, the IPA estimator also possesses the SPI property, so we can estimate the function of $C_i(x)$ more efficiently.

5.3 Optimization of Time-Cost Tradeoffs

Bowman (1994) formulated a nonlinear programming problem called the time-cost tradeoff optimization problem, where the objective function is the expected completion time of a stochastic activity network. The cost of reducing one unit of mean duration of activity i is a_i, and the total budget is B. The upper and lower bounds of the mean duration of activity i are given by u_i and l_i, i.e., $l_i \le \mu_i \le u_i$. Assume $\sum_i a_i(u_i-l_i) > B$ and there are m arcs in the activity network. Then we have an optimization problem with nonlinear objective function and linear constraints given by:

$$\begin{aligned} \min \quad & \mathbb{E}(T) \\ \text{s.t.} \quad & \sum_{i=1}^{m} a_i Y_i \le B \\ & \qquad Y_i \le u_i - l_i, \quad \forall i \in \{1,\dots,m\} \\ & \quad -Y_i \le 0, \quad \forall i \in \{1,\dots,m\} \end{aligned}$$

where Y_i is the amount of decreasing of μ_i and T is the project total completion time. In Bowman (1994), they derived a heuristic algorithm for finding the local optimum solution satisfying the local KKT condition (Boyd and Vandenberghe 2004). The heuristic algorithm proposed in Bowman (1994) has two stages: in stage 1, at each step, the algorithm decreases the mean duration time of the activity that has the largest $\frac{\partial \mathbb{E}(T)}{\partial \mu_i}/a_i$ by $\beta B/a_i$, where β is the fraction of budget to be used, e.g., $\beta = 0.01$; in stage 2, the algorithm redistributes the budget so that the solution is close to the KKT condition solution. Bowman (1994) also claimed that the necessary and sufficient condition for a solution to be local optimal is that for all activities such that $l_i < \mu_i < u_i$, $\frac{\partial \mathbb{E}(T)}{\partial \mu_i}/a_i$ are all equal.

Using the proposed Taylor series approximation method in section 5.1, we propose a new heuristic algorithm called the Knapsack Ratio (KR) algorithm for solving the time-cost tradeoff optimization problem. In the following algorithm, μ_i^0 stands for the original mean duration of activity i. The KR algorithm is given below:

1. Input: N, α, t.
2. Evaluate the criticality index of all activities based on N simulation replications. Find the set of activities $\mathbf{G}$ such that all arcs in $\mathbf{G}$ have estimated criticality index $C_a(i) \geq t$.
3. Set $\delta_i = \alpha * (\mu_i^0 - l_i)$, evaluate the value $\frac{\Delta_i \mathbb{E}(T)(\delta_i)}{\delta_i}/a_i$ for all activities in set $\mathbf{G}$ based on N additional simulation replications.
4. Select the activity i that maximizes $\frac{\Delta_i \mathbb{E}(T)(\delta_i)}{\delta_i}/a_i$.
5. $\theta_i \leftarrow \theta_i - \min(a_i\delta_i, B)/a_i$.
6. $B \leftarrow B - \min(a_i\delta_i, B)$.
7. **If** $B > 0$, **go to** step 3. **Else**, stop.

Our algorithm assumes that at each step, a chosen activity's mean duration will be decreased by a fixed amount. The optimal solution of the optimization problem is approximated by the optimal solution of a knapsack integer programming problem when α is small enough, e.g., $\alpha = 0.01$. $\Delta_i \mathbb{E}(T)(\delta_i)$ is estimated using the Taylor series approximation method proposed in section 5.1, and direct Monte Carlo simulation with common random numbers (DMCCR) is applied for the estimation in step 3.

6 SIMULATION EXPERIMENTS

In this section, stochastic activity networks with fixed numbers of nodes and arcs are randomly generated using the algorithm presented in Demeulemeester et al. (1993). All arc lengths in the randomly generated network are either normally distributed or exponentially distributed. For normally distributed activities, their mean durations are generated uniformly between 0.5 and 50, and their standard deviations are generated uniformly between 0.1 and 1 times their mean durations. For exponentially distributed activities, their mean durations are generated uniformly between 0.5 and 50.

For following experiments, we will use direct Monte Carlo simulation with common random numbers (DMCCR). Suppose we use Monte Carlo simulation with N simulation replications to estimate $\mathbb{E}(T)$ or $C_a(i)$ when a stochastic network and all activities' distribution parameters are given. Then one of the activity's distribution parameter θ_i is changed and we need to re-estimate performance functions like $\mathbb{E}(T)$ and $C_a(i)$. Instead of simulating all activities' lengths again, DMCCR only re-simulates activity i's length N times and replaces the old simulated X_i values while keeping all other simulated $X_j, j \neq i$, values unchanged. DMCCR can save computational time and reduce variance when only a small set of activities' parameters are changed.

6.1 Estimation of Criticality Curves

With the number of arcs and number of nodes fixed, a random network with all arcs independently distributed is generated. The criticality index of a given activity i is mainly affected by two factors: the number of paths that includes activity i and the mean duration of activity i. In extreme cases, if activity i is on all paths, then $C_i(x) = 1, \forall x$, and if the mean duration of activity i is sufficiently large, then $C_a(i)$ is very close to 1. For activities with very small criticality index values, we are less interested in fitting their criticality curves, because decreasing their mean duration has negligible effect on the expected project completion time. For activities that are on most of the paths, we are also less interested in fitting their criticality curves, because changes in their mean duration has negligible effect on their criticality indexes. Therefore, for each SAN, we randomly choose an activity i such that $C_a(i) > 0.5$ and $\mathscr{R}_i < 0.6$.

After the network structure and distribution parameters are randomly generated, and the arc of interest is chosen, for a chosen arc i with original mean duration μ, 30 mean duration values ranging from 0.1μ to 1.5μ with stepsize $1.4\mu/30$ are considered. For each mean duration value x, $N = 1000$ simulation replications are run for estimating $C_i(x)$ and its corresponding sample standard deviation is computed. The mean of the sample standard deviations across the 30 different x values is also calculated, called the Mean

Standard Deviation (MSD). To compare the TAC and CAC estimators (Bowman 1995), ratios of their MSD and computation time are computed.

The algorithms in section 5.2 are used for the TAC estimator, and DMCCR is applied for estimating the CAC estimator. From Table 1, we can see that the variance performance of the TAC estimator and the CAC estimator in estimating the criticality curve for both normally distributed and exponentially distributed activity times are close to each other. From Table 2, we can see that the computational time for the CAC estimator is about 100 times that of using the TAC estimator. Thus, considering variance performance and computing speed, the TAC estimator is preferred over the CAC estimator in estimating the criticality curve.

Table 1: Criticality Curve Estimation MSD Ratio of CAC/TAC

	Network Size					
	30 Nodes		50 Nodes		100 Nodes	
Arc Distributions	60 Arcs	90 Arcs	100 Arcs	150 Arcs	200 Arcs	300 Arcs
Normal	1.4	1.15	0.89	1.39	1.16	1.42
Exponential	0.93	0.85	0.87	0.86	0.9	1.02

Table 2: Criticality Curve Estimation Time Ratio of CAC/TAC

	Network Size					
	30 Nodes		50 Nodes		100 Nodes	
Arc Distributions	60 Arcs	90 Arcs	100 Arcs	150 Arcs	200 Arcs	300 Arcs
Normal	73.8	93.1	95.9	123.7	108.0	126.7
Exponential	88.2	102.6	85.9	96.9	86.2	90.1

Figure 2 depicts a representative criticality curve plot of a chosen activity i in a randomly generated SAN with normally distributed activities of size 20 nodes and 50 arcs, where 30 different criticality values are obtained using the same method as in the experiments in Tables 1 - 2. For logit curve fitting, we first do a logistic regression on 30 sample points estimated by the TAC estimator and have the estimation for coefficients of the Logit function, then plot the curve of the Logit function. For TAC and CAC, we first estimate 30 different criticality values, then join the 30 points with a smooth curve. Figure 2 indicates that the Logit curve fit of the criticality curve deviates from the other two, especially for lower values of μ_i.

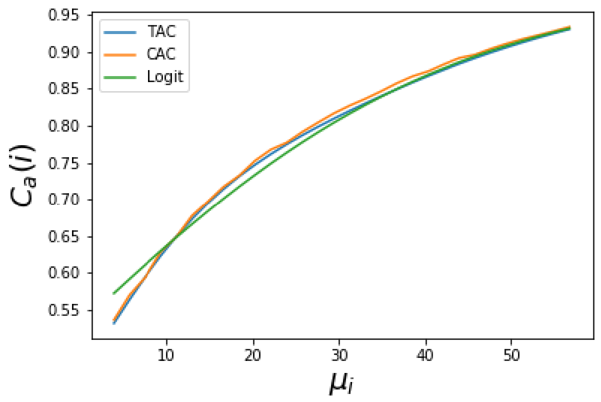

Figure 2: Criticality Curve of Normally Distributed SAN with 20 Nodes and 50 Arcs.

6.2 Estimation of Change of Mean Completion Time

We consider three approaches for estimating $\Delta_i\mathbb{E}[T](\delta)$: direct Monte Carlo simulation with common random numbers (DMCCR); Logit model approximation (LGT); Taylor series approximation (TSA). The experiment first generates a random network with given numbers of arcs and nodes, and then chooses an activity as in section 6.1, i.e., an activity i such that $C_a(i) > 0.5$ and $\mathscr{R}_i < 0.6$. In the following experiments, $\delta = \alpha\mu$, where μ is the original mean duration of the activity of interest and α takes two values: $\alpha = 10\%$ and $\alpha = 20\%$. For the TSA method, $N = 3$ in Equation (5). For each method, $N = 1000$ simulation replications are run for estimating $\Delta_i\mathbb{E}[T](\delta)$ once. For each methods and network, $\Delta_i\mathbb{E}[T](\delta)$ is estimated 100 times to compute the sample mean and sample standard error.

In Tables 3 - 6, the first three rows are the 95% confidence intervals (C.I.) of the three methods on the 100 macro replications. And the last three rows are the total computation time of the three methods. For each column, an activity network with given number of nodes, number of arcs, and arc distributions is first generated. From Tables 3 - 6 we can conclude that both TSA and LGT method have better variance performance than the DMCCR method. The variance performance of TSA and LGT are indistinguishable. In Tables 3 - 4, all three methods have close estimated mean values. But in Tables 5 - 6, LGT underestimates $\Delta_i\mathbb{E}(T)(\delta)$, as is mentioned in Cho and Yum (2004). As for computing speed, TSA and DMCCR are faster than LGT, and TSA and DMCCR have similar computing speed. In conclusion, in terms of variance performance and computing speed, the TSA method is the best among all three different methods.

Table 3: Normal Distribution Mean Completion Time with 10% Change (95% C.I. based on 100 Macro Replications)

	Network Size					
	30 Nodes		50 Nodes		100 Nodes	
	60 Arcs	90 Arcs	100 Arcs	150 Arcs	200 Arcs	300 Arcs
TSA C.I.	2.22 ± 0.006	3.85 ± 0.006	4.15 ± 0.004	3.39 ± 0.004	4.77 ± 0.004	3.52 ± 0.006
LGT C.I.	2.22 ± 0.006	3.86 ± 0.006	4.15 ± 0.004	3.4 ± 0.004	4.78 ± 0.004	3.52 ± 0.006
DMCCR C.I.	2.19 ± 0.19	3.91 ± 0.22	4.25 ± 0.18	3.35 ± 0.27	4.82 ± 0.16	3.68 ± 0.24
TSA Time	132	133	287	287	921	930
LGT Time	320	320	469	471	1087	1097
DMCCR Time	121	122	276	274	906	909

Table 4: Normal Distribution Mean Completion Time with 20% Change (95% C.I. based on 100 Macro Replications)

	Network Size					
	30 Nodes		50 Nodes		100 Nodes	
	60 Arcs	90 Arcs	100 Arcs	150 Arcs	200 Arcs	300 Arcs
TSA C.I.	4.25 ± 0.01	7.6 ± 0.014	8.27 ± 0.01	6.74 ± 0.006	9.5 ± 0.008	6.98 ± 0.01
LGT C.I.	4.24 ± 0.01	7.63 ± 0.014	8.28 ± 0.01	6.75 ± 0.006	9.52 ± 0.008	6.99 ± 0.01
DMCCR C.I.	4.36 ± 0.19	7.78 ± 0.24	8.31 ± 0.18	6.77 ± 0.31	9.5 ± 0.16	7.11 ± 0.19
TSA Time	127	125	276	280	900	904
LGT Time	304	302	451	461	1068	1073
DMCCR Time	115	115	266	268	889	893

6.3 Optimization of Time-Cost Tradeoffs

In this section, the DMCCR is not applied, i.e., whenever μ_i is changed, all arc lengths are re-simulated. The DMCCR is not applied here for two reasons: (1) the number of simulation replications $N = 1000$ is

Table 5: Exponential Distribution Mean Completion Time with 10% Change (95% C.I. based on 100 Macro Replications)

	Network Size					
	30 Nodes		50 Nodes		100 Nodes	
	60 Arcs	90 Arcs	100 Arcs	150 Arcs	200 Arcs	300 Arcs
TSA C.I.	2.42 $\pm$ 0.008	3.73 $\pm$ 0.008	4.05 $\pm$ 0.006	2.88 $\pm$ 0.008	4.63 $\pm$ 0.006	3.0 $\pm$ 0.004
LGT C.I.	1.28 $\pm$ 0.008	3.19 $\pm$ 0.01	3.81 $\pm$ 0.008	2.41 $\pm$ 0.008	4.3 $\pm$ 0.008	2.46 $\pm$ 0.01
DMCCR C.I.	2.6 $\pm$ 0.26	3.66 $\pm$ 0.34	3.95 $\pm$ 0.36	2.96 $\pm$ 0.29	4.34 $\pm$ 0.34	3.09 $\pm$ 0.27
TSA Time	119	121	273	276	908	915
LGT Time	126	127	280	283	914	918
DMCCR Time	117	119	271	273	905	916

Table 6: Exponential Distribution Mean Completion Time with 20% Change (95% C.I. based on 100 Macro Replications)

	Network Size					
	30 Nodes		50 Nodes		100 Nodes	
	60 Arcs	90 Arcs	100 Arcs	150 Arcs	200 Arcs	300 Arcs
TSA C.I.	4.68 $\pm$ 0.018	7.4 $\pm$ 0.018	8.07 $\pm$ 0.012	5.71 $\pm$ 0.014	9.21 $\pm$ 0.012	5.91 $\pm$ 0.016
LGT C.I.	2.42 $\pm$ 0.018	6.33 $\pm$ 0.024	7.6 $\pm$ 0.018	4.79 $\pm$ 0.018	8.58 $\pm$ 0.016	4.85 $\pm$ 0.02
DMCCR C.I.	4.75 $\pm$ 0.23	7.31 $\pm$ 0.31	8.24 $\pm$ 0.35	5.58 $\pm$ 0.25	9.15 $\pm$ 0.38	5.78 $\pm$ 0.27
TSA Time	116	116	264	268	887	887
LGT Time	123	123	270	275	895	895
DMCCR Time	114	115	263	266	885	887

too small to get an accurate estimate using DMCCR for the case when several μ_is are changed one by one; (2) for large enough N, e.g., $N = 10000$, the time complexity of evaluating the ratio is relatively large compared to that of extra simulation runs without using the DMCCR. For each randomly generated SAN, the KR algorithm and Bowman's algorithm (Bowman 1994) are compared for finding the optimal solution of time-cost tradeoff problems described in 5.3. The parameter settings are: μ_i^0 is the original mean duration of activity i, costs a_i are uniformly generated between 1 and 20, $u_i = \mu_i^0, l_i = 0.2 * \mu_i^0$, criticality lower bound $t = 0.001$, budget $B = 0.2\sum a_i(u_i - l_i)$. The parameter settings for the KR algorithm are: number of simulation replications $N = 1000$, decreasing step $\alpha = 0.2$, and criticality lower bound $t = 0.001$, Taylor series approach is used in step 4 of the KR algorithm with $N = 2$ in Equation (6). The parameter setting for Bowman's algorithm are: simulation replication for phase 1 and phase 2 are $N1 = N2 = 1000$, step of decreasing for phase 1 and phase 2 are $A1 = A2 = 0.1$*Budget, upper limit of iteration times for phase 2 is $M = 20$. After finding the solutions using the two different algorithms, 10,000 simulation replications are run for estimating the mean and standard deviation of project completion time under the parameter settings obtained through two algorithms. In Tables 7 - 8, the first two rows are the 95% confidence interval (C.I.) of project completion time under the optimized parameters obtained by the KR algorithm and Bowman's algorithm. The third row is the 95% C.I. for the project completion time before decreasing any mean duration. Rows 4 and 5 are the computation times for the optimization using the two methods.

From Tables 7 - 8, we can see that the KR algorithm is better than Bowman's algorithm in terms of computing speed, especially for large complex networks. However, in some cases, the KR algorithm's solution has a worse (larger) objective value compared with Bowman's algorithm's solution, although in most cases the difference is relatively small compared to the original objective function value. By decreasing α and t, it is believed that the KR algorithm will converge to the global optimal solution.

Table 7: Normal Distribution Optimal Project Completion Time Estimation (95% C.I. based on 10,000 Independent Replications)

	Network Size					
	30 Nodes		50 Nodes		100 Nodes	
	60 Arcs	90 Arcs	100 Arcs	150 Arcs	200 Arcs	300 Arcs
KR	181.7 ± 0.74	239 ± 0.78	211.1 ± 0.8	232.3 ± 0.71	237.8 ± 0.74	253.4 ± 0.74
Bowman	183.1 ± 0.74	239.5 ± 0.78	210.7 ± 0.82	233 ± 0.68	238.1 ± 0.76	257 ± 0.74
Original	301.7 ± 0.84	391.7 ± 0.92	409.3 ± 1.02	405.9 ± 0.91	427.2 ± 0.98	474.8 ± 0.84
KR Time	65	199	127	371	692	2806
Bowman Time	128	286	543	795	1949	3942

Table 8: Exponential Distribution Optimal Project Completion Time Estimation (95% C.I. based on 10,000 Independent Replications)

	Network Size					
	30 Nodes		50 Nodes		100 Nodes	
	60 Arcs	90 Arcs	100 Arcs	150 Arcs	200 Arcs	300 Arcs
KR	209.8 ± 1.08	278.3 ± 1.32	235.9 ± 1.19	291.1 ± 1.25	287.8 ± 1.2	318.9 ± 1.08
Bowman	208.6 ± 1.07	275.8 ± 1.29	232.3 ± 1.13	291.4 ± 1.22	288.4 ± 1.18	320.1 ± 1.07
Original	347.3 ± 1.7	460.8 ± 2.03	458.8 ± 2.02	499.9 ± 2.09	500.6 ± 1.95	595.9 ± 1.7
KR Time	22	177	82	248	567	2760
Bowman Time	70	209	315	450	2549	5148

7 CONCLUSIONS AND FUTURE RESEARCH

When applied to higher-order gradient estimation, the TAC estimator for estimating criticality curves and change of expected project completion time has faster computing speed and lower sample variance compared to existing methods. Applying the change of expected project completion time estimator to time-cost tradeoff optimization problems in PERT networks, a new heuristic algorithm is provided in this paper, and simulation experiments show that our new algorithm is considerably faster for large complex activity networks compared to Bowman's gradient-based algorithm.

Our future research includes: improving the algorithm in section 5.3 by decreasing the mean duration in batches (instead of decreasing the largest one, decreasing the top m mean values at the same time), finding an algorithm for tuning the parameters for the KR algorithm and Bowman's algorithm for specific activity networks, deriving a new gradient estimator for the second moment of project completion time and applying it to optimize both mean and variance of the project completion time under budget constraints, and combining Monte Carlo simulation and PERT approaches to develop a new algorithm that finds the optimal solution more efficiently when the network is very large and complex.

ACKNOWLEDGMENTS

This work was supported in part by the Air Force Office of Scientific Research (AFOSR) under Grant FA95502010211.

REFERENCES

Bowman, R. 1994. "Stochastic Gradient-based Time-cost Tradeoffs in PERT Networks using Simulation". *Annals of Operations Research* 53(1):533–551.

Bowman, R. 1995. "Efficient Estimation of Arc Criticalities in Stochastic Activity Networks". *Management Science* 41(1):58–67.

Boyd, S. P., and L. Vandenberghe. 2004. *Convex Optimization*. Cambridge University Press.

Cho, J., and B.-J. Yum. 2004. "Functional Estimation of Activity Criticality Indices and Sensitivity Analysis of Expected Project Completion Time". *Journal of the Operational Research Society* 55(8):850–859.

Demeulemeester, E., B. Dodin, and W. Herroelen. 1993. "A Random Activity Network Generator". *Operations Research* 41(5):972–980.

Fu, M. C. 2006. "Sensitivity Analysis in Monte Carlo Simulation of Stochastic Activity Networks". In *Perspectives in Operations Research*, edited by F. B. Alt, M. C. Fu, and B. L. Golden, 351–366. Springer.

Kim, S.-J., S. P. Boyd, S. Yun, D. D. Patil, and M. A. Horowitz. 2007. "A Heuristic for Optimizing Stochastic Activity Networks with Applications to Statistical Digital Circuit Sizing". *Optimization and Engineering* 8(4):397–430.

Sigal, C., A. Pritsker, and J. Solberg. 1979. "The Use of Cutsets in Monte Carlo Analysis of Stochastic Networks". *Mathematics and Computers in Simulation* 21(4):376–384.

Wan, P., and M. C. Fu. 2020. "Sensitivity Analysis of ARC Criticalities in Stochastic Activity Networks". In *Proceedings of the 2020 Winter Simulation Conference*, edited by K. G. Bae, B. Feng, S. Kim, S. Lazarova-Molnar, Z. Zheng, T. Roeder, and R. Thiesing, 2911–2922. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHIES

PENG WAN is a Ph.D. candidate in Applied Mathematics, Statistics and Scientific Computation at the Department of Mathematics at the University of Maryland, College Park. His research interests include stochastic optimization, stochastic activity networks, and applied probability. His e-mail address is pwan@umd.edu.

MICHAEL C. FU holds the Smith Chair of Management Science in the Robert H. Smith School of Business, with a joint appointment in the Institute for Systems Research and affiliate faculty appointment in the Department of Electrical and Computer Engineering, A. James Clark School of Engineering, all at the University of Maryland, College Park, where he has been since 1989. He is the co-author of the books, *Conditional Monte Carlo: Gradient Estimation and Optimization Applications*, which received the INFORMS Simulation Society's 1998 Outstanding Publication Award, and *Simulation-Based Algorithms for Markov Decision Processes*, and has also edited/co-edited four volumes: *Perspectives in Operations Research*, *Advances in Mathematical Finance*, *Encyclopedia of Operations Research and Management Science* (3rd edition), and *Handbook of Simulation Optimization*. He attended his first WSC in 1988 and served as Program Chair for the 2011 WSC. He has served as Program Director for the NSF Operations Research Program (2010–2012, 2015), *Management Science* Stochastic Models and Simulation Department Editor, and *Operations Research* Simulation Area Editor. He received the INFORMS Simulation Society's Distinguished Service Award in 2018. He is a Fellow of INFORMS and IEEE. His e-mail address is mfu@umd.edu, and his Web page is https://www.rhsmith.umd.edu/directory/michael-fu.

STEVEN I. MARCUS is a Professor in the Department of Electrical and Computer Engineering and the Institute for Systems Research, University of Maryland. He received the B.A. degree in electrical engineering and mathematics from Rice University in 1971, and the S.M. and Ph.D. degrees in electrical engineering from MIT in 1972 and 1975, respectively. From 1975 to 1991, he was with the Department of Electrical and Computer Engineering, University of Texas, Austin, where he was the L.B. (Preach) Meaders Professor in Engineering. At the University of Maryland, he has served as Director of the Institute for Systems Research from 1991-1996, and as Chair of the Electrical and Computer Engineering Department from 2000-2005. He is former Editor-in-Chief of the *SIAM Journal on Control and Optimization*. Currently, his research is focused on stochastic control and estimation, Markov decision processes, and hybrid systems, with applications in manufacturing and telecommunication networks. He is a Fellow of IEEE and SIAM. His email address is marcus@umd.edu.