Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 14, No. 4, pp. 595–619, 2018

SOLVING THE OPTIMAL RESOURCE ALLOCATION IN MULTIMODAL STOCHASTIC ACTIVITY NETWORKS USING AN OPTIMAL COMPUTING BUDGET ALLOCATION TECHNIQUE

Jen-Yen Lin, Ming-Jong Yao* and Yi-Hua Chu

Abstract: The problem of optimal Resource Allocation in Multimodal Stochastic Activity Networks (RAMSAN) has been studied for more than a decade. Many researchers proposed solution approaches, including dynamic programming and meta-heuristics, for solving the RAMSAN, and they commonly applied Monte Carlo Simulation (MCS) for evaluating the objective function values for the candidate solutions. However, since the computational load of MCS is very demanding, the solution approaches using MCS become impractical, even for only medium-size problems. In this study, we propose a Genetic Algorithm (GA) with an Optimal Computing Budget Allocation (OCBA) approach for solving the RAMSAN. We utilize the technique of OCBA to optimally allocate the computational budget among the candidate solutions in the evolutionary process of GA for evaluating their objective functions. Based on the benchmark instances in the literature, we demonstrate the proposed GA with OCBA is more effective than the other solution approaches in the literature and a GA without using OCBA.

Key words: *genetic algorithm; stochastic activity networks; resource allocation; Monte Carlo simulation; optimal computing budget allocation*

Mathematics Subject Classification: *65C05, 68U20, 90B15, 90C59*

1 Introduction

The focus of this study is to investigate the optimal resource allocation to the activities in Stochastic Activity Networks (SAN). In practice, project managers need to allocate the resources available to the activities so as to "optimally" complete the project. However, when the available resources are limited, it becomes a tough optimization problem for the project managers because of the stochastic characteristics (for instance, the completion time, etc.) of SAN. In this study, we propose a Genetic Algorithm (GA) with an Optimal Computing Budget Allocation (OCBA) technique as a decision-support tool for solving this stochastic optimization problem.

Usually, project managers have to pay close attentions to the randomness of the duration of the activities since it directly impacts the total costs and the completion time of the project (see [9], [21], [27]) On the other hand, there usually exists a trade-off relation between the resource allocation and the duration of the activities. That is, one may allocate more resources to shorten the duration of an activity. The objective of the time-cost trade-off

*Corresponding Author

resource allocation problem is to get the optimal performance of the network by maintaining both the resources and the duration of activities in proper realization (see [1]). In this paper, our concerned problem is to solve the problem of optimal Resource Allocation in Multimodal SAN (which will be abbreviated as the RAMSAN for the rest of this paper).

In the literature, researchers proposed many approaches for solving the RAMSAN. One group of the researchers employed mathematical programming approaches as their solution approaches. For example, Tereso et al. [23] proposed a dynamic programming approach for solving the RAMSAN. For an early review of relative literature, one may refer to [10]. Theoretically, these mathematical programming approaches can obtain the optimal solution for the RAMSAM. Elmaghraby et al. [11] proposed a 'policy iteration-type' procedure of dynamic programming for solving the problem posed, which they formulated it as a Continuous-Time Markov chain (CTMC) following [16]. They also introduced the 'phase type distributions' which played an important role in policy evaluation, and discussed its application. Recently, Klerides and Hadjiconstantinou [15] proposed a path-based two-stage stochastic integer programming approach though the resource consumed for the execution of each activity is not restricted by a lower and an upper bound (which is different from the scenario in this study). In their solution approach, they determine the execution modes in the first stage, and the second stage performs activity scheduling according to the realizations of activity durations. However, the above studies usually demand enormous computational efforts, and such a problem makes them impractical for the decision-makers when facing even just a medium-size activity network.

Other group of researchers employed simulation-based meta-heuristics for solving the RAMSAN. Feng et al. [12] and Chu et al. [8] proposed genetic algorithms (GA), and Tereso et al. [24] and Godinho and Branco [13] applied Electromagnetism Algorithms (EMA) for the RAMSAN. We note that both GA and EMA belong to the category of so-called population-based methodology that employs a population of sample points in the solution space, utilizes the solution-improvement mechanisms of exploration and exploitation to search through the space, and hopefully, converges to a close-to-optimal solution after iteratively implementing the solution-improvement mechanisms long enough. The aforementioned approaches employed Monte Carlo Simulation (MCS), see [20] and [19], for evaluating the objective function values of the solutions obtained during the search process. However, these simulation-based approaches could also become impractical, even for medium-size problems, since MCS is very demanding in computational load.

To the best of the authors' knowledge, most of the existing studies on the RAMSAN assume that the duration of any activity conforms to an exponential function; e.g., [23], [24], [26], [25]. Note that, in real cases, the duration of any activity may conform to any type of probability density function (PDF), for instance, a normal distribution function, a gamma distribution function, or a uniform distribution, $\cdots$, etc. Therefore, it shall match the RAMSAN much closer with the managers' decision-making scenario in the real world allowing a general-type of PDF for the duration of any activity. Recently, Godinho and Branco [13] studied a scenario that is similar to [23] and they allowed a general-type PDF for each activity. However, they assumed that several distinct modes may be used to undertake each activity, and the choice of mode may be adapted to the way the project is developing. Also, the scheduling policies are based on a set of thresholds, and the starting time of the activity is compared with those thresholds in order to define the execution mode. Therefore, the scenario is significantly different from what we are interested in this study.

In this paper, we propose a GA for solving the RAMSAN, in which the duration of any activity may conform to a general-type of PDF. In the proposed GA, we employed the technique of OCBA to improve the overall simulation efficiency by optimally allocating the computational efforts of MCS among the solutions in the population.

The rest of this paper is organized as follows. Section 2 discusses the problem definition and formulates the mathematical model for the problem of optimal Resource Allocation in Multimodal SAN (namely, the RAMSAN). Section 3 provides the details of the proposed GA with MCS and OCBA, which is abbreviated as the GAwOCBA for the rest of the paper. Section 4 presents our numerical experiments that compare the proposed GAwOCBA with the other solution approaches proposed in [23, 24] and a GA without using OCBA.

2 The Mathematical Model

In this section, we present a mathematical model of the problem of optimal Resource Allocation in Multimodal SAN (namely, the RAMSAN).

First, we present a summary of the decision-making scenario of the RAMSAN. One may refer to [23] and [11] for further details. We are given a *multi-modal* activity network (that is, each activity can be performed at any number of levels of resource intensity applied to it, with resulting shorter or longer duration), with stochastic work content $\{W_a\}$ for each activity $a \in A$, where A is the set of activities, and $|A| = n$ (we adopt the Activity-on-Arc mode of representation of the project). We assume that W_a's are stochastically independent. In this study, the activity's work content could be *any probability distribution* (it is different from the cases in [23, 24] in which they assumed that activity's work content conforms to exponential distribution). We consider only one resource in this study, and denote the intensity-level of resource allocated to each activity $a \in A$ by x_a, which indicates that a total of x_a units of the resource will be consumed for the execution of activity a. The value of x_a is supposed to lie within a specified interval:

$$0 < l_a \leq x_a \leq u_a < \infty, \forall a \in A \tag{2.1}$$

where l_a is the lower bound of the resource allocation and u_a is the upper bound. In this study, the vector of resource allocations $\mathbf{x} \triangleq (x_a | \forall a \in A)$ contains the decision variables of the RAMSAN. Therefore, the duration of an activity a, denoted by Y_a, depends on its work content and the amount of resources allocated to it; that is,

$$Y_a = W_a / x_a \tag{2.2}$$

In such a case, the duration of the activity would be the *consequence* of the resources allocated to the activity, rather than the *cause* of the uncertainty. For simplicity, we assume that such resource allocation incurs a "resource usage cost" that is proportional to the square of the allocation for the duration of the activity (this assumption can be easily relaxed by taking different cost functions following the practical cases in the real-world). The resource usage cost for activity a is given by

$$C_a^U (x_a) = U_a \cdot x_a^2 \cdot Y_a = U_a \cdot x_a \cdot W_a, \tag{2.3}$$

in which U_a is the proportional constant (without loss of generality, we set $U_a = 1$ in this study). Observe that $C_a^U (x_a)$ and Y_a are also random variables.

Next, we assume that the project has a specified due date T_z, and a tardiness penalty that is a function of the actual completion time of the project, denoted by $C_P (\Upsilon_n (\mathbf{x}) - T_z)$, where $\Upsilon_n (\mathbf{x})$ is the realization time of node n, which is also the completion time of the project, with $\mathbf{x}$ being the vector $\{x_a\}_{a \in A}$ and A is the set of activities. And, Υ_n obviously is also a random variable. Again, for simplicity, we take a simple expression for the penalty function that is proportional to the tardiness, with the proportionality constant c_L (cost of tardiness per unit time).

$$C_P\left(\Upsilon_n\left(\mathbf{x}\right) - T_z\right) = c_L \cdot \max\left\{0, \Upsilon_n\left(\mathbf{x}\right) - T_z\right\} \tag{2.4}$$

We are now ready to state the RAMSAN in a precise fashion. The objective of the RAMSAN is to determine the resource allocation vector $\mathbf{x}$ so as to minimize the total expected cost $E\left[C_T\left(\mathbf{x}\right)\right]$ (of undertaking the activities and of being penalized for tardiness) as expressed in (2.5).

$$\min_{\mathbf{x}\in S} E\left[C_T\left(\mathbf{x}\right)\right] = E\left\{\sum_{a\in A} C_a^U\left(x_a\right) + C_P(\Upsilon_n\left(\mathbf{x}\right) - T_z)\right\} \tag{2.5}$$

with the feasible set S being given by

$$S = \left\{\mathbf{x}\in\mathbb{R}^n | 0 < l_a \le x_a \le u_a < \infty, a\in A\right\}. \tag{2.6}$$

$\boxed{3}$ The Proposed Solution Approach

In this section, we propose a Genetic Algorithm (GA) with an Optimal Computing Budget Allocation (OCBA) technique, which is abbreviated as "GAwOCBA". Section 3.1 presents the details of the Genetic Algorithm, and Section 3.2 introduces the technique of OCBA. Finally, we summarize the proposed GAwOCBA in Section 3.3.

$\boxed{3.1}$ Scheme of the Genetic Algorithm

In this subsection, we state the details of GA by describing its six core components, namely, the representation of chromosome, initialization of population, evaluation of fitness, selection mechanism, crossover, and mutation.

We employ binary encoding for the representation of chromosome as follows. For every activity $a \in A$, the interval $[l_a, u_a]$ is discretized into m_a equal-distance solutions $\left\{x_a^i\right\}_{i=0}^{m_a}$ where $x_a^i = l_a + \frac{i(u_a - l_a)}{m_a}$. We shall need a binary string of β_a bits for encoding the mapping where β_a is the smallest integer such that $2^{\beta_a} \ge m_a$. Then, we may use eq. (3.1) to express the mapping between the binary string and the value of i as follows.

$$\langle b_1\ b_2\ \cdots\ b_{\beta_a - 1}\ b_{\beta_a}\rangle \equiv i = \left(\sum_{j=1}^{\beta_a} b_j 2^{j-1}\right)_{10} \tag{3.1}$$

In case the value of the mapped i is larger than u_a, we flip all bits in the binary string (equivalently, using compliment computation) to assure that it is no larger than u_a. So, we may guarantee the feasibility of our chromosome representation. For example, if $m = 15$, then we shall need $\lceil\log_2(m+1)\rceil = \lceil\log_2(16)\rceil = 4$ bits to store the value of decision variable x_a. Moreover, if $x_a = x_a^5$, then we store it by $<\underline{1010}>$. Hence, if there are $n = 5$ activities, every chromosome is stored in 20×1 array. If a chromosome is chosen as $<\underline{0010}\quad\underline{1000}\quad\underline{0100}\quad\underline{0001}\quad\underline{0000}>$, then $x_1 = x_1^4$, $x_2 = x_2^1$, $x_3 = x_3^2$, $x_4 = x_4^8$, $x_5 = x_5^0$.

Suppose that the population size PS is given. We randomly generated the initial population following the binary-encoding representation above.

On the evaluation of fitness, we first estimate the expected total cost $E\left[C_T\left(\mathbf{x}\right)\right]$ for every chromosome by Monte Carlo Simulation (MCS). The number of MCS runs will be allocated uniformly or by the technique of OCBA, which will be discussed in Section 3.2. After estimating $E\left[C_T\left(\mathbf{x}\right)\right]$ by MCS, we sort the chromosomes in the population and obtain a ranking in ascending order. With such a ranking, we propose to perform *fitness normalization* in our GA where fitness normalization is a process of converting row fitness values to ones that behave better (Refer to [14]) and give high probability for selecting good solutions in new generations, while maintaining some chance of survival to poor solutions. (Refer to [3])

Fitness normalization can be carried out in three forms: (i) inversion normalization, (ii) linear ranking normalization, and (iii) nonlinear normalization. In our study, we tested two types of fitness normalization: inversion normalization and linear ranking normalization and chose linear ranking normalization in our GA since linear ranking normalization was slightly better than inversion normalization according to our experiments. The details discussions of the linear ranking normalization can be referred to [18].

The normalized fitness values of chromosome i_{temp} (denoted by $eval_{i_{temp}}$) are calculated as follows:

$$eval_{i_{temp}} = 2 - SP + \frac{2(SP-1)(i_{temp}-1)}{PS-1} \tag{3.2}$$

where PS is the population size, and we set $SP = 1.5$ in our GA (Note: Selection Pressure (SP) represents the ratio of the probability of selecting the best individual to the average probability of selecting all individuals, and SP usually takes values in the range of $[1.0, 2.0]$; see [18]).

After normalizing fitness, we used a *roulette wheel mechanism* for selecting chromosomes for reproduction. The reproduction probability of each chromosome is proportional to its normalized fitness (relative to the sum of the normalized fitness value of all the chromosomes) as expressed in eq. (3.3) as follows:

$$P_{i_{temp}} = \frac{eval_{i_{temp}}}{\sum\limits_{i=1}^{PS} eval_i} \tag{3.3}$$

As one can observe, the larger the probability (corresponding to better fitness) for a chromosome, the higher the chance it will be reproduced in the next generation. Those chromosomes, that survive the selection, undergo the alternation by two genetic operators, crossover and mutation, to generate the chromosomes in the next generation.

In this study, we test single-point crossover, two-point crossover and uniform crossover in our GA for its crossover operations since uniform crossover, like multi-point crossover, has been claimed to reduce the bias associated with the length of the binary representation used and the particular coding for a given parameter set (see [18]).

The single-point (two-point) crossover operator randomly selects one (two) cut point(s) for each chosen pair of chromosomes, then, exchanging the genes between the (two) cut point(s) to create two new offspring. For the uniform crossover, we first create a crossover mask, which is the same length as the chromosome structure, at random. The parity of the bits in the mask indicates which parent will supply the offspring with which bits. Following our numerical experiments, the three crossover operators are indifferent in their performance.

After performing crossover operator, we apply the mutation operator to the population that just experienced the crossover operator. The mutation operator randomly chooses the ones among the genes of all chromosomes in the population with a fixed mutation rate. Then, the mutation operator flips the chosen genes.

$\boxed{3.2}$ The Technique of OCBA

One may consider the problem of optimal Resource Allocation in Multimodal SAN as a simulation optimization problem. Many simulation optimization approaches were explored to improve the efficiency of selecting the best design of a stochastic system (which corresponds to the optimal resource allocation in this study). Chen and Lee [5] indicated that approaches to address this problem fall under the well-established branch of statistics known as ranking and selection or multiple comparison procedures; see [2]. Chen [4] and Chen et al. [7, 6] proposed the technique of Optimal Computing Budget Allocation (OCBA) that replies upon the usage of both the sample means and variances in the allocation procedures to maximize the probability of correct selection (that refers to choosing a design or configuration of the input variables optimizing the objective function).

Before introducing the procedure of OCBA, we provide a brief review of the problem definition (one may refer to [5] for details). After performing N_i replications for design i, the expected value of the performance of design i (denoted as J_i) is estimated using its sample mean $\overline{J}_i$ based on simulation output. Since we intend to find a design with minimum mean, it makes sense to choose the design with minimum sample mean, i.e.,

$$b = \underset{i}{\operatorname{argmin}} \overline{J}_i \tag{3.4}$$

Design b is usually called an "observed best design". Since the sample mean estimator has some variability, design b is not necessarily the one with the smallest unknown mean performance. Define the probability of correct selection, denoted as $P(CS)$, as the probability that design b is actually the best design (i.e., the smallest mean, hence the true best design), namely,

$$P(CS) = P\{\text{design } b \text{ is actually the best design}\} \tag{3.5}$$

One needs to evaluate many designs during the process of simulation optimization, which is our situation when applying GA for solving the RAMSAN. Recall that we need to evaluate the objective function for each of the PS chromosomes in the population for each generation of GA. Assume that there are a total of k different designs among the PS chromosomes where $k \leq PS$. The technique of OCBA is proposed to maximize the probability of correct selection as shown in (3.6).

$$\begin{array}{cl} \displaystyle\max_{N_1,N_2,\ldots,N_k} & P(CS) \\ s.t. & N_1 + N_2 + \ldots + N_k = T \text{ and } N_i \geq 0 : \text{integer}, \forall i. \end{array} \tag{3.6}$$

Next, we present a cost-effective sequential approach based on OCBA to select the best design from k alternatives with a given computing budget. Initially, n_0 simulation replications for each of k designs are conducted to get some information about the performance of each design at the first stage. As simulation proceeds, the sample means and sample variances of all designs are computed from the data already collected up to that stage. According to this collected simulation output, an incremental computing budget, Δ, should be determined at each stage. Ideally, each new replication should bring us closer to the optimal solution. The algorithm of OCBA is summarized as follows (see also [5] for details).

OCBA Procedure (Maximizing $P(CS)$)

INPUT k, T, Δ, η_0;

INITIALIZE Set $l = 0$. Perform a total of η_0 simulation replications for all designs; $N_1^l = N_2^l = \ldots = N_k^l = n_0 = \eta_0/k$, i.e., we conduct $n_0 = \eta_0/k$ simulation replications for each design where k is the total number of designs.

LOOP While $\sum_{i=1}^{k} N_i^l < T$ do

[**Update**] Calculate sample means $\overline{J_i}$, and sample standard deviations s_i, $i = 1, 2, \cdots, k$ using the new simulation outputs; find $b = \text{argmin}_i\, \overline{J_i}$.

[**Allocate**] Increase the computing budget by Δ and calculate the new budget allocation,

$$N_1^{l+1}, N_2^{l+1}, ..., N_k^{l+1},$$

according to

$$(1) \quad \frac{N_i^{l+1}}{N_j^{l+1}} = \left(\frac{s_i \left(\overline{J_b} - \overline{J_j} \right)}{s_j \left(\overline{J_b} - \overline{J_i} \right)} \right)^2, \text{for all } i \neq j \neq b, \text{and} \tag{3.7}$$

$$(2) \quad N_b^{l+1} = s_b \sqrt{\sum_{i=1, i \neq b}^{k} \left(\frac{N_i^{l+1}}{s_i} \right)^2} \tag{3.8}$$

[**Simulate**] Perform additional $\max \left(N_i^{l+1} - N_i^l, 0 \right)$ simulations for design $i = 1, \cdots, k$; set $l = l + 1$.

END OF LOOP

3.3 GAwOCBA Algorithm

We first discuss the parameter settings employed in the proposed GAwOCBA algorithm. Since the obtained solutions from GA have some variability, we introduce an elite set that keeps updating a set of "best" solutions on hand during the evolutionary process of GA. At last, we present a summary of the proposed GAwOCBA algorithm.

3.3.1 Parameter settings

We will first discuss the parameter settings in our GA, and present that of OCBA next in this section.

We know a set of good parameter settings assists the GA to obtain a close-to-optimal solution within a reasonable run time. Our GA needs four parameters, namely, the number of generations, the crossover rate, the mutation rate and the population size. Based on our numerical experiments, we suggest that the population size (denoted by PS) of the GA is $5 \cdot \lceil \sqrt{n} \rceil$ (where n is the number of activities in the activity network). This study has tested many settings of the crossover rate (denoted by P_c) and the mutation rate (denoted by P_m) for the GA. We tested several combinations of P_c and P_m in the range of $P_c \in [0.2, 0.6]$ and $P_m \in [0.01, 0.10]$, and observed that the setting of $P_c = 0.45$ and $P_m = 0.05$ usually leads to better performance following our numerical experiments. The GA ends the evolutionary process when the best-on-hand solution shows no improvement during the last 20 generations (denoted as the threshold being $\Psi = 20$ in the algorithm) or the number of generations reaches a designated threshold, e.g., 150 generations. For the former, we need an invariant counter ϕ that counts the number of generations the "best" solution did not change, and it resets, i.e., $\phi = 1$, as the "best" solution on hand updates during the evolutionary process. Our numerical experiments tested both termination conditions, and we will have further discussions in Section 4.

As applying the technique of OCBA, we need to set the following parameters: initial budget η_0 (i.e., the total number of initial simulation replications for all chromosomes), one-time incremental computing budget Δ, and total budget T. Law and Kelton [17] and Bechhofer et al. [2] suggested that a good choice of n_0 is usually somewhere between 5 and 20 with $n_0 = \eta_0/k$ where k is the total number of designs. However, we observed that the estimates of the mean and variance gave poor results when n_0 was set between 5 and 20 for the concerned problem. Therefore, we increase the number of MCS runs significantly and set $\eta_0 = 1,000$ for $n \leq 15$; $\eta_0 = 1,500$ for $15 < n \leq 30$; $\eta_0 = 2,030$ for $30 < n \leq 45$; or $\eta_0 = 3,015$ for $n > 45$ where n is the number of activities in the activity network. For example, if our GA has a population of 25 chromosomes (i.e., $k = 25$), each chromosome could have $\eta_0/k = 1,500/25 = 60$ runs of MCS for an activity network with $n = 24$ activities. Also, we suggest that the total budget could be $T = 1000 \cdot \lceil \sqrt{n} \rceil$ where n is the number of activities in the activity network. Note that it makes sense to have more replications of simulation for lager-size activity networks. For example, an activity network with 11 activities (like the first instance in Section 4.1) will require a total budget of $T = 1000 \cdot \lceil \sqrt{11} \rceil = 4,000$. The selection of Δ is more like the step size in typical optimization search algorithms. One may also refer to Section 4.2.3 in [5] where they present an algorithm that round-off N_i to nearest integers while ensuring the summation of additional simulation replications for all designs (i.e., the chromosomes of the population in our GA) equals Δ. Though Chen and Lee [5] suggests lower values of Δ (e.g., less than 100), we observe that it is preferable to set $\Delta = 200$ for $n \leq 15$; $\Delta = 300$ for $15 < n \leq 30$; $\Delta = 400$ for $30 < n \leq 45$; or $\Delta = 500$ for $n > 45$ where n is the number of activities in the activity network based on our numerical experiments.

3.3.2 Elite Set

Since the obtained solutions from GA have some variability, one is not able to distinguish the best solution simply using the mean (obtained from MCS) of the solution. Therefore, we employ an elite set, denoted as E, keeping a set of candidates that could serve as the "best" solutions on hand during the evolutionary process of GA.

As obtaining the initial population, we compare all of them in pairs (say chromosomes i and j) using *student's t-test* with a null hypothesis $H_0 : \mu_i \leq \mu_j$. We build the initial elite set using the following procedure. If we are able to distinguish the preferred chromosome, we rank that pair of chromosomes accordingly; otherwise, we simply sort those indifferent chromosomes in an ascending order of their sample means. We put the sorted chromosomes in the elite set. In our numerical experiments, we set the size of elite set being equal to the population size, i.e., $|E| = PS$.

We try to update the elite set with the one with the lowest value in this generation after a new set of chromosomes are generated from the selection mechanism, crossover, and mutation.

3.3.3 A summary of the GAwOCBA algorithm

Step 0. Input: Activity network (V, A), precision requirement (or, grid size $m_a, \forall a \in A$), population size PS, crossover rate P_c, mutation rate P_m, the threshold Ψ on the number of generations the "best" solution did not change, initial budget η_0, total budget T, and size of elite set $|E|$. Also, set the invariant counter $\phi = 1$ and $t = 1$.

Step 1. Randomly generate PS chromosomes as an initial population.

Step 2. Use the technique of OCBA in Section 3.2 to allocate number of MCS to estimate the objective function values for all chromosomes in the population of this generation. Sort the chromosomes in this generation by their sample means from MCS. If $t = 1$, we build the initial elite set following the procedure presented in Section 3.3.2. For the case of $t > 1$, we try to update the elite set with the one with the lowest value in the population of this generation. If the "best" solution (i.e., the solution ranked no. 1) in the elite set is updated, we reset the invariant counter $\phi = 1$; otherwise, set $\phi = \phi + 1$.

Step 3. If the termination criterion (e.g., $\phi > \Psi$) is met, then stop and report the elite set E; otherwise, go to Step 4.

Step 4. Calculate the normalized fitness values of all chromosomes, and employ a roulette-wheel mechanism for selection.

Step 5. Conduct crossover for the new population from selection.

Step 6. Do mutation for the population that just experienced crossover. Set $t = t + 1$, and go to Step 2.

4 Numerical Experiments

We will present four parts of numerical experiments in this section. First, we will take two examples in [23] to compare their solutions from Dynamic Programming (DP) with the proposed GAwOCBA. The second part employs the 14 instances in [22] to compare the proposed GAwOCBA with (i) an Electromagnetism Algorithm (EMA), and (ii) a genetic algorithm with only MCS. The third part presents our sensitivity analysis on the parameters in the proposed GAwOCBA. Finally, we show more comparison analysis as the last part of this section.

4.1 GA with OCBA vs. DP

Before presenting our comparison analysis, we provide a brief review on the DP approach proposed by Tereso et al. [23] as follows. Recall that A is the set of activities. In the DP approach, one shall need to fix the resource allocation to some of the activities prior to applying the optimization procedure, and these activities are referred to as the "conditioned upon" activities. We may refer to this subset of activities as the fixed subset, denoted by $\mathcal{F}$. In the DP recursion, the stage is defined as the epoch of decision on the value of x_a for some activity a in the set of decision variables, denoted by $\mathcal{D}$, which is the complementary set of $\mathcal{F}$, i.e. $\mathcal{D} \equiv A - \mathcal{F}$. Tereso et al. [23] proposed a straightforward (but not necessarily optimal) process to select the activities to be "conditioned upon" (the set $\mathcal{F}$) as follows:

1. Determine the longest path (in the sense of number of activities) in the network. The activities on the longest path will be the decision variables (set $\mathcal{D}$).

2. The other activities, $A - \mathcal{D}$, will be the activities to be fixed (set $\mathcal{F}$).

We take two Activity-on-Arc (AoA) activity networks in Tereso et al. [23] as the instances for comparison in this section. For both instances, the work content W_a for each activity a is discretely uniformly distributed with four values so that the corresponding expectation is $1/\lambda_a$. Also, we use the following settings for the two examples in this section: The due date of this project is $T_s = 65$ weeks and the cost of tardiness is $c_L = 5$ dollars/week. For every

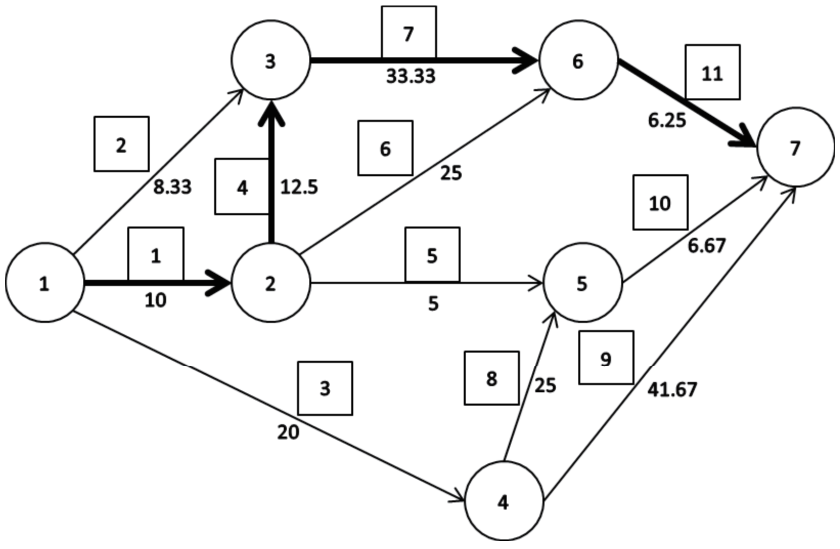


Figure 1: An activity network with 7 nodes and 11 arcs (activities); source: [23].

activity a, the decision maker chooses the value of resource allocation x_a from $[0.5, 1.5]$, namely, $l_a = 0.5$ and $u_a = 1.5$.

Example 4.1. Figure 1 presents an example with 11 activities and 7 nodes, which is the 6^{th} instance in Tereso [22]. Table 1 shows the parameters of the instance in Figure 1.

Tereso et al. [23] determined $x_1 \to x_4 \to x_7 \to x_{11}$ as the 'longest' path since it is one of the paths with the *highest number* of activities in the activity network. They treated the resource allocations on the longest path, i.e., (x_1, x_4, x_7, x_{11}), as the decision variables, but fixed the resource allocations on the other activities at $(x_2, x_3, x_5, x_6, x_8, x_9, x_{10}) = (1, 1.5, 0.5, 0.5, 1, 1.5, 1)$. Then, they used Dynamic Programming (DP) that conducted partial enumeration those 5^3 possible solutions on x_4, x_7, and x_{11} to obtain the solution $\mathbf{x}_{DP}$. We ran MCS for 20,000 times to evaluate $\mathbf{x}_{DP}$ with an objective function value of $\mu_{DP} = 419.54$. Note that the objective function value is different from that reported in Tereso et al. [23] probably since they used only 100 times of MCS for their evaluation.

We used the proposed GAwOCBA for solving this instance, and terminated it as the number of generations reaches 150 with the parameter settings presented in Section 3.3.1. Then, we recorded the obtained solutions as $\mathbf{x}^{150}_{GAwOCBA} = \left\{x_a^{150} | \forall a \in A\right\}$, and evaluated it with 20,000 times of MCS by $\mu^{150}_{GAwOCBA} = E\left[C_T\left(\mathbf{x}^{150}_{GAwOCBA}\right)\right] = 340.20$.

Next, we compared our solution $\mathbf{x}^{150}_{GAwOCBA}$ with $\mathbf{x}_{DP}$ from DP in Tereso et al. [23] with $\mu_{DP} = E\left[C_T\left(\mathbf{x}_{DP}\right)\right]$ and utilized a *z-test* to check the following hypothesis:

$$\begin{aligned} H_0 &: \mu^{150}_{GAwOCBA} \geq \mu_{DP} \\ H_1 &: \mu^{150}_{GAwOCBA} < \mu_{DP} \end{aligned} \tag{4.1}$$

Since it holds that

$$Z^0 = \frac{\mu^{150}_{GAwOCBA} - \mu_{DP}}{\sqrt{\frac{\left(S^{150}_{GAwOCBA}\right)^2}{n^{150}_{GAwOCBA}} + \frac{(S_{DP})^2}{n_{DP}}}} = -31.822 < Z_{(0.05)} = -1.645 \tag{4.2}$$

Table 1: The parameters of the activities in Figure 1.

Activity	AoA representation	Parameter	Expected Work Content
a	(i,j)	λ_a	$1/\lambda_a$
1	(1,2)	0.100	10.00
2	(1,3)	0.120	8.33
3	(1,4)	0.050	20.00
4	(2,3)	0.080	12.50
5	(2,5)	0.200	5.00
6	(2,6)	0.040	25.00
7	(3,6)	0.030	33.33
8	(4,5)	0.040	25.00
9	(4,7)	0.024	41.67
10	(5,7)	0.150	6.67
11	(6,7)	0.160	6.25

the hypothesis H_0 is rejected with a confidence level of 95%. (Obviously, it holds for higher levels of confidence, e.g., 99%, too.) Therefore, we conclude that the solution $\mathbf{x}^{150}_{GAwOCBA}$ from the proposed GAwOCBA is better than the solution $\mathbf{x}_{DP}$ from DP presented in [23].

Example 4.2. Figure 2 presents an example with 11 activities, but 6 nodes, and it is the 5th instance in Tereso [22]. Table 2 shows the parameters of the instance in Figure 2.

Tereso [22] determined $x_1 \to x_4 \to x_5 \to x_8 \to x_{10}$ as the "longest" path since it is a path with the *highest number* of activities in the activity network. They treated $(x_1, x_4, x_5, x_8, x_{10})$ as the decision variables, but fixed the others at $(x_2, x_3, x_6, x_7, x_9, x_{11}) = (1.25, 1, 0.5, 1, 0.5, 1)$. We ran MCS for $20,000$ times to evaluate $\mathbf{x}_{DP}$ with an objective function value of $\mu_{DP} = 235.47$. Next, we solved this instance by the proposed GAwOCBA, and evaluated the obtained solution with 20,000 times of MCS by $\mu^{150}_{\text{GA}wOCBA} = 143.27$. We checked the hypothesis testing in (4.1) and concluded that the hypothesis H_0 is rejected with a confidence level of 95% since the value of $Z^0 = -54.9138$ is far less than the threshold $Z_{(0.05)}$.

Note that Tereso et al. [23] and Tereso [22] restricted the possible values of resources to only several discrete values in the feasible range as applying DP (probably, to make it possible to obtain its solution within a reasonable run time). On the other hand, the precision of the resource allocation for each activity in the proposed GAwOCBA could reach the 4th place of decimal point. Therefore, the proposed GAwOCBA is able to obtain a much better solution than that from DP in [23] and [22].

$\boxed{4.2}$ GA with OCBA vs. EMA and a GA without OCBA

Since DP is only able to solve small-size problems, Tereso et al. [24] proposed a metaheuristic, namely, an Electromagnetism Algorithm (EMA), for solving the RAMSAN. Also, one may be curious about if the OCBA technique does equip the proposed GAwOCBA with significant advantage. To demonstrate the effectiveness of the proposed GAwOCBA, we would compare it with the EMA proposed by [24] and a GA without using the technique of OCBA with the 14 instances in [22].

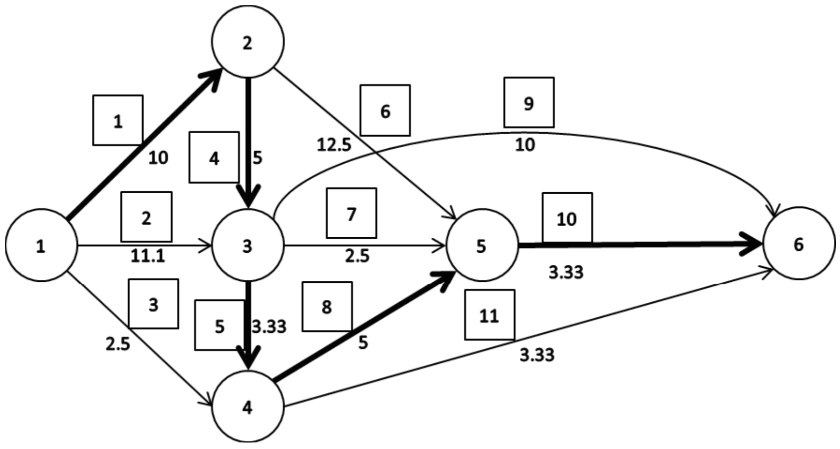


Figure 2: An activity network with 6 nodes and 11 arcs (activities); source: [22].

Table 2: The parameters of the activities in Figure 2.

Activity	AoA representation	Parameter	Expected Work Content
a	(i,j)	λ_a	$1/\lambda_a$
1	(1,2)	0.100	10.00
2	(1,3)	0.090	11.11
3	(1,4)	0.400	2.50
4	(2,3)	0.200	5.00
5	(3,4)	0.300	3.33
6	(2,5)	0.080	12.50
7	(3,5)	0.400	2.50
8	(4,5)	0.200	5.00
9	(3,6)	0.100	10.00
10	(5,6)	0.300	3.33
11	(4,6)	0.300	3.33

Table 3: Parameter Setting of the 14 Instances in [22].

Instance	No. of Nodes	No. of Arcs	Population Size PS	Initial Budget η_0	Total Budget T	Incremental Budget Δ
1	3	3	20	1000	4000	200
2	4	5				
3	5	7				
4	6	9				
5	6	11				
6	7	11				
7	8	12				
8	7	14				
9	10	14				
10	10	17	25	1500	5000	300
11	14	18				
12	17	24				
13	20	38	35	2030	7000	400
14	36	76	45	3015	9000	500

We used the same parameter settings for the implementation of the EMA in [24], and the obtained solutions were denoted as $\mathbf{x}_{EMA}$. On the other hand, we employed the parameter settings suggested in Section 3.3.1 for the proposed GAwOCBA, and denoted the obtained solution as $\mathbf{x}_{GAwOCBA}$. Since both the EMA and the proposed GAwOCBA are metaheuristics embedded with randomized mechanism, we solved each instance for 20 times using both approaches. Then, we recorded the average:

$$\overline{C}_T^{EMA} = \frac{1}{20}\sum_{i=1}^{20} E(C_T(\mathbf{x}_{EMA}^i))$$

and

$$\overline{C}_T^{GAwOCBA} = \frac{1}{20}\sum_{i=1}^{20} E(C_T(\mathbf{x}_{GAwOCBA}^i))$$

for these 20 runs for the EMA and the proposed GAwOCBA, respectively. Also we recorded the standard deviation of the expected total cost for these 20 runs for the EMA and the proposed GAwOCBA, respectively. We list the parameter settings of the 14 instances in Table 3. The results from both approaches for the 14 instances are summarized in Table 4.

The first three columns in Table 3 provide the information on the configuration of the 14 instances in [22]. One may see that the largest instance is an activity with 36 nodes and 76 arcs (activities). The fourth column of Table 3 presents the population size PS for genetic algorithm, and the fifth-seventh columns present the initial budget η_0, total budget T, and one-time incremental budget Δ for OCBA. Using these parameters, we implement GAwOCBA on the 14 instances, and we list the results in Table 4.

One may observe that the average run time of the EMA is faster than that of the proposed GAwOCBA for each instance by comparing the third column with the fifth in Table 4. On the other hand, the second and fourth columns of Table 4 present the values of $\overline{C}_T^{EMA}$ and

Table 4: Comparison between the EMA and the proposed GAwOCBA.

Instance	$\overline{C}_T^{EMA}$	Avg. Run time of EMA (seconds)	$\overline{C}_T^{GAwOCBA}$	Avg. Run time of GAwOCBA (seconds)	Improvement of GAwOCBA
1	46.34	0.46	45.75	92.38	1.29%
2	436.04	1.41	382.89	135.91	13.88%
3	277.16	2.79	237.10	177.67	16.90%
4	491.68	5.06	447.34	219.94	9.91%
5	204.48	7.78	147.11	255.70	39.00%
6	452.96	8.26	339.17	262.05	33.55%
7	234.38	10.20	199.07	284.23	17.74%
8	141.30	13.78	127.47	318.52	10.85%
9	934.07	15.00	842.60	331.25	10.86%
10	232.50	23.95	168.24	489.30	38.20%
11	629.14	30.39	446.51	533.25	40.90%
12	2517.90	63.07	1619.32	697.38	55.49%
13	1139.50	190.36	956.47	1441.22	19.14%
14	679.12	1268.22	625.03	3649.23	8.65%

$\overline{C}_T^{GAwOCBA}$, and the last column shows the (average) improvement (in percentage) in the objective function value comparing the proposed GAwOCBA to the EMA. We note that the value of $\overline{C}_T^{EMA}$ in [22] is significantly different from ours probably because Tereso [22] reported their results with only 100 runs of MCS. To make the comparison fair, we use $20,000$ runs to evaluate the objective function value for the best (final) solutions from both approaches in our experiments. One may notice that the proposed GAwOCBA obtained better solutions than the EMA by more than 25% for 5 (out of 14) instances. Furthermore, we checked the hypothesis in (4.3) with the 20 runs of samples from both approaches by *student t-test.*

$$\begin{aligned} H_0 &: \overline{C}_T^{GAwOCBA} \geq \overline{C}_T^{EMA} \\ H_1 &: \overline{C}_T^{GAwOCBA} < \overline{C}_T^{EMA} \end{aligned} \tag{4.3}$$

For *each* instance, the hypothesis H_0 in (4.3) is rejected with a confidence level of 95%. Thus, our numerical experiments show the solution quality of the proposed GAwOCBA is superior to the EMA in [24]. However, we did not see any trend or particular pattern that the (average) improvement of the proposed GAwOCBA (comparing to the EMA) varies with the size of the instances.

In order to observe the effectiveness of OCBA, we would like to compare the proposed GAwOCBA with a GA without using OCBA. The setting of the GA without OCBA is the same as the proposed GAwOCBA except making use of the technique of OCBA. We ran the GA without OCBA for 150 generations, recorded its run time, and picked the "best" solution in the elite set at termination for each of the 14 instances in [22]. Then, we used the same run time, employed the proposed GAwOCBA to solve each instance, and also picked the "best" solution in the elite set at termination. For each sample run, we ran 20,000 runs of MCS for the "best" solutions from both approaches and recorded their sample means. Then, we summarized the results of 20 runs of samples for each instance from both approaches in Table 5 where the second column shows the average run time of the GA without OCBA for

Table 5: Comparison between a GA without OCBA and the proposed GAwOCBA.

Instance	Avg. Run time of GA150 (seconds)	$\overline{C}_T^{GA150}$	$\overline{C}_T^{GAwOCBA}$	Improvement of GAwOCBA
1	70.61	46.96	44.83	4.75%
2	103.83	432.33	366.38	18.00%
3	136.78	276.44	231.98	19.17%
4	167.14	483.08	455.76	5.99%
5	195.02	178.70	135.78	31.61%
6	198.59	390.59	332.33	17.53%
7	218.72	210.01	203.21	3.35%
8	241.50	136.62	131.94	3.55%
9	251.14	971.16	912.45	6.43%
10	366.48	201.34	161.54	24.64%
11	404.63	564.10	498.41	13.18%
12	523.86	1792.18	1575.63	13.74%
13	1092.27	1066.37	953.31	11.86%
14	2727.55	789.93	762.31	3.61%

150 generations.

The third and fourth columns of Table 5 report the average of the sample means of both GA approaches from 20,000 runs of MCS. The last column shows the (average) improvement in the sample mean as comparing the proposed GAwOCBA with the GA without OCBA. For *each* instance, we checked the hypothesis H_0 in (4.4), and concluded the hypothesis is rejected with a confidence level of 95% by *student t-test*.

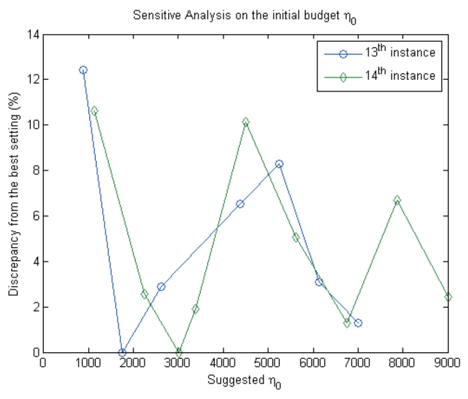
$$\begin{aligned} H_0 &: \overline{C}_T^{GAwOCBA} \geq \overline{C}_T^{GA150} \\ H_1 &: \overline{C}_T^{GAwOCBA} < \overline{C}_T^{GA150} \end{aligned} \tag{4.4}$$

One may observe that using the technique of OCBA obtained more than an average of 20% improvement in the objective function values for Instances No. 5 and No. 10, and more than 10% improvement for 6 other instances. In fact, following our experiments, for all the samples of all instances, the proposed GAwOCBA solved better solutions than that without OCBA. Therefore, we assert that the technique of OCBA effectively assists GA to allocate the computational budget of MCS runs to those promising chromosomes, and successfully distinguishes the better candidates from inferior ones during the evolutionary process so as being able to obtain out-performed solutions.

$\boxed{4.3}$ Sensitivity Analysis

We conducted sensitivity analysis by testing different settings of parameters of OCBA based on the 13th and 14th instances in [22]. Here, we would suggest the parameter settings for the technique of OCBA, namely, initial budget $\eta_0 = n_0 \cdot PS$ (where PS is the population size in GA), total budget T, and one-time incremental computing budget Δ based on the results of our numerical experiments.

First, we would discuss our experiments on initial budget η_0 on the 13^{th} instance in [22]. Note that Law and Kelton [17] and Bechhofer et al. [2] suggested that a good choice of n_0 is usually somewhere between 5 and 20. However, we observed that the estimates of the mean and variance in [24] with 100 runs of MCS are reasonable for small-size instances, but poor for larger-size ones (referring to Section 4.2). We decided to test eight settings of initial budget $\eta_0 = n_0 \cdot PS$ with $n_0^{(1)} = 25$, $n_0^{(2)} = 50$, $\cdots$, and $n_0^{(8)} = 200$. Note that when the initial budget for a chromosome is set to 200 and the population size is fixed as 35, the algorithm GAwOCBA is reduced to the GA *without* OCBA. For each instance, we obtained 20 samples for each setting. We present the results graphically for the 13^{th} instance in Figure 3 (please refer to the line linked with "circles"). One may also refer to the details of the experimental results in Table A-1.



.

Figure 3: Sensitive Analysis on the initial budget η_0.

Figure 3 indicates the best choice of initial budget is located at $n_0^{(2)} = 50$, i.e. $\eta_0 = n_o \cdot PS = 50 \cdot 35 = 1750$ with $PS = 35$, for the 13^{th} instance. Then, we did pairwise comparison between $n_0^{(2)} = 50$ and the other settings $n_0^{(i)} (i = 1, 3, 4, 5, 6, 7, 8)$, and concluded the hypothesis H_0 that the setting of $n_0^{(2)} = 50$ is worse than any other setting $n_0^{(i)} (i = 1, 3, 4, 5, 6, 7, 8)$ is rejected with a confidence level of 95%.

We also tested several settings on initial budget n_0 on the 14^{th} instance in [22], similar to what we did for the 13^{th} instance, i.e., by conducting pairwise comparison using the results of 20 runs from each setting. We chose nine settings of initial budget $\left(n_0^{(1)}, ..., n_0^{(9)}\right) = (25, 50, 67, 75, 100, 125, 150, 175, 200)$ and summarized the results in Figure 3 (please refer to the line linked with "diamonds"). One may also refer to the details of the experimental results in Table A-2.

The best choice of initial budget is $\eta_0 = n_0 \cdot PS = 67 \cdot 45 = 3015$ with $n_0^{(3)} = 67$ and $PS = 35$ for the 14^{th} instance from Figure 3. Also, we concluded the hypothesis H_0 that the setting of $n_0^{(3)} = 67$ is worse than any other setting $n_0^{(i)} (i = 1, 2, 4, 5, 6, 7, 8, 9)$ is rejected with a confidence level of 95%.

Following our numerical results, we suggest the setting for the initial budget $\eta_0 = n_0 \cdot PS$ (where PS is the population size in GA) by $\eta_0 = 1,000$ for $n \leq 15$; $\eta_0 = 1,500$ for $15 < n \leq 30$; $\eta_0 = 2,030$ for $30 < n \leq 45$; or $\eta_0 = 3,015$ for $n > 45$, as presented in

Section 3.3.1.

Next, we tested the settings of total budget T for the 13^{th} and 14^{th} instances. For the 13^{th} instance, we tested 9 levels of T with each one being a multiple of 500, namely, from $T^{(1)} = 5,000$ to $T^{(9)} = 9,000$. For the 14^{th} instance, we tested 9 levels of T with each one being a multiple of 500, namely, from $T^{(1)} = 6,000$ to $T^{(9)} = 10,000$. Among all the experiments, we obtained the results of 20 runs from each setting and conducted pairwise comparison as we did in Figure 3. We summarized the results of the 13^{th} and 14^{th} instances in Figure 4, and one may refer to the details of the experimental results in Tables A-3 and A-4, respectively. (Please refer to the lines linked with "circles" and "diamonds" for the 13^{th} and 14^{th} instances, respectively.)

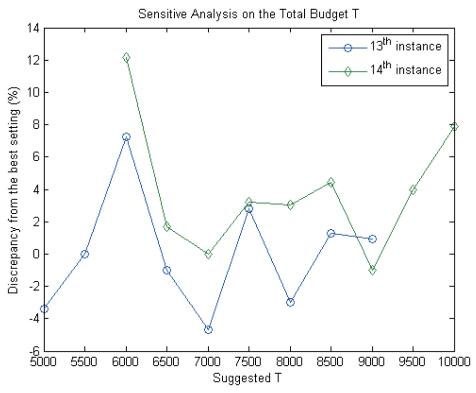


Figure 4: Sensitive Analysis on the Total Budget T.

The recommended settings for the 13^{th} and 14^{th} instances meet the expression of $1000 \cdot \lceil\sqrt{n}\rceil$, namely, $1000 \cdot \lceil\sqrt{n}\rceil = 7000$ for the 13^{th} instance with $n = 38$ and $1000 \cdot \lceil\sqrt{n}\rceil = 9000$ for the 14^{th} with $n = 76$. (For *both* instances, the hypothesis H_0 that the recommended setting is worse than any other setting is rejected with a confidence level of 95%.) Therefore, following our results, we recommend the suggested setting for total budget T by $T = 1000 \cdot \lceil\sqrt{n}\rceil$ where n is the number of activities in the activity network. Note that it is reasonable to use more replications of simulation for lager-size instances.

Finally, we present the way of choosing the suggested setting of one-time incremental computing budget, Δ. We tested 5 levels of Δ with each one being a multiple of 100, namely, from $\Delta^{(1)} = 200$ to $\Delta^{(5)} = 600$, collected the results of 20 runs from each setting, and conducted pairwise comparison. We present our results for the 13^{th} and 14^{th} instances in Figure 5, and one may refer to the details of the experimental results in Tables A-5 and A-6, respectively. (Please refer to the lines linked with "circles" and "diamonds" for the 13^{th} and 14^{th} instances, respectively.)

Though Chen and Lee[5] suggests lower values of Δ (e.g., less than 100), we observe that $\Delta = 400$ for the 13^{th} instance and $\Delta = 500$ for the 14^{th} gave us good and stable performance based on our numerical experiments. For *both* instances, the hypothesis H_0 that the recommended setting is worse than any other setting is rejected with a confidence level of 95%. Following our numerical experiments, we recommend $\Delta = 200$ for $n \leq 15$; $\Delta = 300$ for $15 < n \leq 30$; $\Delta = 400$ for $30 < n \leq 45$; or $\Delta = 500$ for $n > 45$ where n is the number of activities in the activity network.

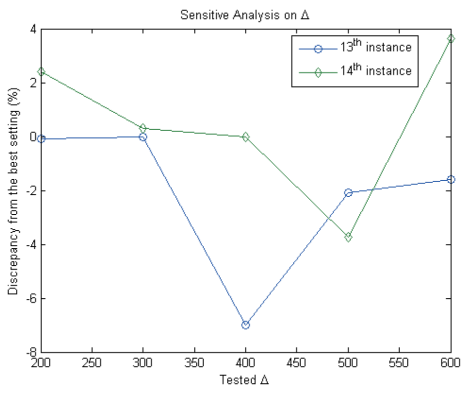


Figure 5: Sensitive Analysis on Δ.

4.4 More Comparison Analysis

We would present two more sets of comparison analysis between the proposed GAwOCBA and the GA without OCBA in this subsection. The first part concerns with the evolutionary process of the best-on-hand solution versus computational time. The other investigates how different types of probability distribution for the durations of activities affect the results.

In the first set of comparison analysis, we collect the best-on-hand solution obtained by the proposed GAwOCBA with the GA without using OCBA, and observe that how the best-on-hand solution changes versus computational time.

Note that for both approaches, we need to evaluate the obtained solutions for every chromosome in each generation. The computational load could be too demanding if we use a large value of Total Budget T. But, on the other hand, we need a large number of MCS to secure a precise estimate of the objective function for each chromosome. Therefore, we evaluate the objective function value of the newly obtained solutions among the top-10 list with 10,000 times of MCS every 10 generations, and the top-10 list was updated accordingly. We would observe how the best-on-hand of both approaches solution change before terminating at the 500^{th} generation. Figure 6 shows the evolution process of the best-on-hand solution versus computational time.

One may observe that the best-on-hand solution of the proposed GAwOCBA updates with a faster and more frequent pace than the GA without using OCBA. Also, the proposed GAwOCBA updated the best-on-hand solution for 5 times, (more frequently than the GA without OCBA for 3 times) before the 200^{th} generation. We consider the frequent updates resulted from the OCBA technique is able to "smartly" make use of the computation budget to those preferable solutions. The GA without OCBA had a better solution than the proposed GAwOCBA before the 200^{th} generation due to the randomness characteristics of GA. But, the proposed GAwOCBA successfully surpassed the GA without OCBA and enjoyed a significant improvement following the update at the 200^{th} generation. At the end of the 500^{th} generation, the proposed GAwOCBA obtained a superior solution (with its objective function value being 351.18) than the GA without OCBA (with its objective function value being 351.38). Again, our numerical experiments demonstrate that the technique of OCBA assisted GA to effectively allocate the computational budget to those promising candidates so that the search of GA became more efficient.

We would like to investigate how different types of probability distribution for the du-

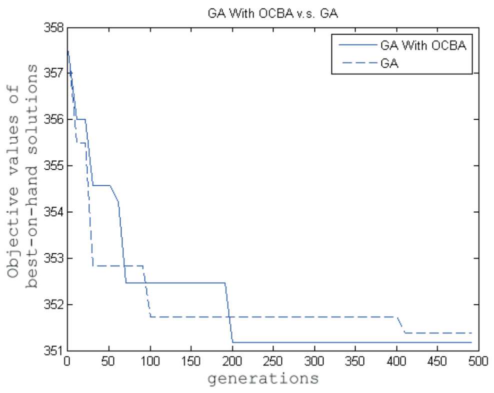


Figure 6: The best-on-hand solution versus computational time (in generations).

Table 6: Comparison of the improvement of GAwOCBA for different types of probability distribution.

Type of distribution	$\overline{C}_T^{GA150}$	$\overline{C}_T^{GAwOCBA}$	Improvement of GAwOCBA
Normal	203.34	186.89	8.80%
Gamma	203.33	190.72	6.61%
Exponential	390.59	332.33	17.53%

rations of activities affect the performance of the proposed GAwOCBA in the second set of comparison analysis. Here, we employ the 6th instance in [22], which is shown in Figure 1, as the example for comparison analysis and evaluate the performance by the (average) improvement, viz. $\left(\overline{C}_T^{GA150} - \overline{C}_T^{GAwOCBA}\right)/\overline{C}_T^{GAwOCBA} \cdot 100\%$, of 20 runs where $\overline{C}_T^{GA150}$ and $\overline{C}_T^{GAwOCBA}$ were obtained by the same way presented in Section 4.2.

To ensure a fair starting point for comparison, we calculate the parameters of normal distribution and gamma distribution from the parameter of exponential distribution (λ_a) so that they will get the same values of mean and variance. Table 6 encapsulates our experimental results.

We have the following observations from our experimental results in Table 6.

1. For each type of probability distribution, the hypothesis (H_0) that "the solution of GAwOCBA is worse than the GA without OCBA" is rejected with a confidence level of 95%. Therefore, Table 6 verifies the robustness of the proposed GAwOCBA as comparing the solution quality with the GA without OCBA again even using different types of probability distribution for the durations of activities.

2. The proposed GAwOCBA had the most significant advantage over the GA without OCBA when assuming that the duration of any activity conforms to an exponential distribution comparing to Normal and Gamma distributions as solving the RAMSAN.

$\boxed{5}$ Conclusions

In this study, we propose a Genetic Algorithm with an Optimal Computing Budget Allocation (OCBA) approach, which is abbreviated as GAwOCBA, for solving the problem of optimal Resource Allocation in Multimodal Stochastic Activity Networks (RAMSAN). The technique of OCBA assisted the proposed GAwOCBA in optimally allocating the computational budget for Monte Carlo Simulation among the candidate solutions in the evolutionary process of GA for evaluating their objective functions.

We consider that this study contributes in the following aspects.

1. The proposed GAwOCBA is an effective solution approach for solving the RAMSAN. Based on our numerical experiments, we demonstrate the proposed GAwOCBA is more effective than the dynamic programming approach in [23] and the Electromagnetism Algorithm (EMA) in [24]. Also, using the same run time, the solutions of the proposed GAwOCBA significantly out-performed that resulted from a GA without using OCBA.

2. The proposed GAwOCBA is a solution approach with considerable flexibility. First, the activity's work content could be any probability distribution in this study, which is different from the cases in [23, 24] where they assumed that activity's work content conforms to exponential distribution. Second, one may easily employ the proposed GAwOCBA to solve the extended version of the RAMSAN with multiple resources (though we consider only a single type of resource in this study) by augmenting the encoding representation of chromosomes for the resource allocations to the activities of all types of resources. Third, the function form for the resource usage cost (for activity) could be any complicated and nonlinear function without pre-requested properties.

3. This study illustrates the technique of OCBA could be a useful and effective approach for simulation optimization based on our numerical experiments since it facilitates to efficiently allocate the computational budget to those candidate solutions so as to let the search of GA focus on promising solutions.

From our numerical experiments in Table 6, we observe that the proposed GAwOCBA enjoyed the most significant advantage over the GA without OCBA when assuming that the duration of any activity conforms to an exponential distribution comparing to Normal and Gamma distributions as solving the RAMSAN. Researchers may dig into such an interesting phenomenon for further investigation on why the technique of OCBA is able to better results as the duration of any activity conforms to an exponential distribution than other probability distributions. Also, one may easily find variant versions of the RAMSAN for which the proposed GAwOCBA cannot be directly applied. For example, a possible case is that one would like to maximize the probability of meeting the due-date subject to a restriction on the total financial budget for purchasing all the allocated resources. Though the financial constraint could be a simple linear inequality, it leads to the needs to derive different chromosome representations, appropriate penalty functions, or repair mechanisms for fixing infeasible solutions if one plans to apply GA for solving the problem. The authors are working on this problem and other possible extensions of the problem of Resource Allocation in Multimodal Stochastic Activity Networks.

References

[1] A. Basso and L.A. Peccati, Optimal resource allocation with minimum activation levels and fixed costs, *Eur. J. Oper. Res.* 131 (2001) 536–549.

[2] R.E. Bechhofer, T.L. Santner and D.M. Goldsman. *Design and Analysis of Experiments for Statistical Selection, Screening, and Multiple Comparisons*, John Wiley & Sons, 1995.

[3] J. Boesel, B. L. Nelson, and N. Ishii, A framework for simulation-optimization software, *IIE Trans.* 35 (2003) 221–229.

[4] C.H. Chen, An effective approach to smartly allocate computing budget for discrete event simulation, in *Proceedings of the 34th IEEE Conference on Decision and Control*, 1995, pp. 2598–2603.

[5] C.H. Chen and L.H. Lee, *Stochastic Simulation Optimization: An Optimal Computing Budget Allocation*, World Scientific Publishing Co. Inc., 2010.

[6] C.H. Chen, J. Lin and E. Yücesan, Simulation budget allocation for further enhancing the efficiency of ordinal optimization, *Discrete Event Dyn. Syst.* 10 (2000) 251–270.

[7] H.-C. Chen, L. Dai, C.-H. Chen and E. Yücesan, New development of optimal computing budget allocation for discrete event simulation, in *Proceedings of the 29th Conference on Winter Simulation*, WSC '97, Washington, DC, USA, 1997, IEEE Computer Society, pp. 334–341.

[8] W.M. Chu, M.J. Yao, S.E. Elmaghraby and T.Y. Tseng, A simulation-based genetic algorithm for the optimal resource allocation in probability activity networks, in *the Fifth Metaheuristic International Conference*, 2003.

[9] R.J. Dawson and C.W. Dawson, Practical proposals for managing uncertainty and risk in project planning, *Int. J. Project Manag.* 16 (1998) 299 – 310.

[10] S.E. Elmaghraby, Resource allocation via dynamic programming in activity networks, *Eur. J. Oper. Res.* 64 (1993) 199–215.

[11] S.E. Elmaghraby and G. Ramachandra, Optimal resource allocation in activity networks - stochastic environment, *Int. J. Project Organ. Manag.* 6 (2014) 67–95.

[12] C.W. Feng, L. Liu and S.A. Burns, Stochastic construction time-cost trade-off analysis, *J. Comput. Civil Eng.* 14 (2000) 117–126.

[13] P. Godinho and F.G. Branco, Adaptive policies for multi-mode project scheduling under uncertainty, *Eur. J. Oper. Res.* 216 (2012) 553–562.

[14] A. Hunter, Crossing over genetic algorithms: the Sugal generalised GA, *J. Heuristics* 4 (1998) 179–192.

[15] E. Klerides and E. Hadjiconstantinou, A decomposition-based stochastic programming approach for the project scheduling problem under time/cost trade-off settings and uncertain durations, *Comput. Oper. Res.* 37 (2010), 2131–2140.

[16] V.G. Kulkarni and V.G. Adlakha, Markov and Markov-regenerative PERT networks, *Oper. Res.* 34 (1986) 769–781.

[17] A.M. Law and D.M. Kelton, *Simulation Modeling and Analysis*, McGraw-Hill Higher Education, 3rd edition, 1999.

[18] H. Pohlheim, *Evolutionary algorithms: principles, methods, and algorithms*, Technical Report, Technical University Ilmenau, 2001.

[19] C.E. Sigal, A.A.B. Pritsker and J.J. Solberg. The use of cutsets in Monte Carlo analysis of stochastic networks, *Math. Comput. Simulation* 21 (1979) 376 – 384.

[20] R.M. van Slyke, Letter to the editor−Monte Carlo methods and the PERT problem, *Oper. Res.* 11 (1963) 839–860.

[21] L.V. Tavares, J.A.A. Ferreira and J. S. Coelho, On the optimal management of project risk, *Eur. J. Oper. Res.* 107 (1998) 451–469.

[22] A.P. Tereso, *Optimal resource allocation in stochastic activity networks results obtained using the electromagnetism approach*, Technical reports, 2004.

[23] A.P. Tereso, M.T. Araújo and S.E. Elmaghraby, Adaptive resource allocation in multimodal activity networks, *Int. J. Prod. Econ.* 92 (2004) 1–10.

[24] A.P. Tereso, M.T. Araújo, and S.E. Elmaghraby, The optimal resource allocation in stochastic activity networks via the electromagnetism approach, in *PMS04 Conference*, 2004.

[25] A.P. Tereso, M.T. Araújo, R. Moutinho and S.E. Elmaghraby, Quantity oriented resource allocation strategy on multiple resources projects under stochastic conditions, in *International Conference on Industrial Engineering and Systems Management (IESM 2009)*, 2009.

[26] A.P. Tereso, J.R.M. Mota and R.J.T. Lameiro, Adaptive resource allocation to stochastic multimodal projects: a distributed platform implementation in Java, *Control Cybernet.* 35 (2006) 661–686.

[27] S. Ward and C. Chapman, Transforming project risk management into project uncertainty management, *Int. J. Project Manag.* 21 (2003) 97–105.

Manuscript received 21 June 2016
revised 25 November 2016, 1 November 2017
accepted for publication 3 November 2017

Jen-Yen Lin
Department of Applied Mathematics, National Chiayi University,
No. 300 Syuefu Rd., East Dist., Chiayi, 60004, Taiwan(R.O.C.)
E-mail address: jylinor@gmail.com

Ming-Jong Yao
Department of Transportation and Logistics Management, National Chiao Tung University,
Assembly Building 1, No.1001, Daxue Rd., East Dist., Hsinchu City 30010, Taiwan(R.O.C.).
E-mail address: myaoie@gmail.com

Yi-Hua Chu
Department of Transportation and Logistics Management
National Chiao Tung University,
Assembly Building 1, No.1001, Daxue Rd., East Dist
Hsinchu City 30010, Taiwan(R.O.C.)

Appendix

Table A-1: Sensitive Analysis on the initial budget η_0 for the 13th instance.

Suggested n_0	Suggested $\eta_0 = n_0 \cdot PS$ $(PS = 35)$	$\overline{C}_T^{GAwOCBA}$	Avg. Run time of GAwOCBA (seconds)	Discrepancy from the best setting
25	875	1073.51	1488.84	11.04%
50	1750	955.02	1500.95	0.00%
75	2625	982.66	1424.70	2.81%
100	3500	988.58	1436.55	3.39%
125	4375	1017.36	1454.80	6.13%
150	5250	1034.02	1480.86	7.64%
175	6125	984.66	1494.95	3.01%
200	7000	967.43	1376.97	1.28%

Table A-2: Sensitive Analysis on the initial budget η_0 for the 14th instance.

Suggested n_0	Suggested $\eta_0 = n_0 \cdot PS$ $(PS = 35)$	$\overline{C}_T^{GAwOCBA}$	Avg. Run time of GAwOCBA (seconds)	Discrepancy from the best setting
25	1125	707.47	3551.19	9.59%
50	2250	655.86	3633.95	2.48%
67	3015	639.61	3649.23	0.00%
75	3375	651.69	3656.13	1.85%
100	4500	704.43	3550.27	9.20%
125	5625	671.96	3556.19	4.81%
150	6750	647.86	3632.22	1.27%
175	7875	682.43	3695.48	6.28%
200	9000	655.10	3745.48	2.36%

Table A-3: Sensitive Analysis on the Total Budget T for the 13th instance.

Suggested T	$\overline{C}_T^{GAwOCBA}$	Avg. Run time of GAwOCBA (seconds)	Discrepancy from the best setting
5000	970.74	1034.44	1.37%
5500	1004.76	1130.02	4.71%
6000	1077.59	1221.95	11.15%
6500	995.00	1377.42	3.77%
7000	957.44	1441.22	0.00%
7500	1032.62	1527.16	7.28%
8000	974.93	1605.67	1.79%
8500	1017.66	1749.44	5.92%
9000	1014.22	1828.95	5.60%

Table A-4: Sensitive Analysis on the Total Budget T for the 14th instance.

Suggested T	$\overline{C}_T^{GAwOCBA}$	Avg. Run time of GAwOCBA (seconds)	Discrepancy from the best setting
6000	724.63	2443.70	11.73%
6500	656.69	2646.36	2.60%
7000	645.92	2858.19	0.98%
7500	666.70	3052.20	4.06%
8000	665.61	3245.20	3.91%
8500	674.47	3446.02	5.17%
9000	639.61	3649.23	0.00%
9500	671.58	3863.44	4.76%
10000	696.98	4058.41	8.23%

Table A-5: Sensitive Analysis on Δ for the 13th instance.

Tested Δ	$\overline{C}_T^{GAwOCBA}$	Avg. Run time of GAwOCBA (seconds)	Discrepancy from the best setting
200	1027.58	1423.69	6.92%
300	1028.34	1419.89	6.99%
400	956.47	1441.22	0.00%
500	1006.70	1400.19	4.99%
600	1012.03	1480.20	5.49%

Table A-6: Sensitive Analysis on Δ for the 14th instance.

Tested Δ	$\overline{C}_T^{GAwOCBA}$	Avg. Run time of GAwOCBA (seconds)	Discrepancy from the best setting
200	680.60	3671.27	6.02%
300	666.47	3663.39	4.03%
400	664.54	3656.33	3.75%
500	639.61	3649.23	0.00%
600	688.76	3649.91	7.14%