Journal of Industrial Engineering 6(2010) 13-23

A Multi Objective Fibonacci Search Based Algorithm for Resource Allocation in PERT Networks

Behrouz Afshar Nadjafi[a,*], Salman Kolyaei[a]

[a]*Islamic Azad University, Qazvin Branch, Department of Industrial and Mechanical Engineering, Qazvin, Iran*

Received 6 Jul., 2010; Revised 27 Jul., 2010; Accepted 8 Aug., 2010

Abstract

The problem we investigate deals with the optimal assignment of resources to the activities of a stochastic project network. We seek to minimize the expected cost of the project include sum of resource utilization costs and lateness costs. We assume that the work content required by the activities follows an exponential distribution. The decision variables of the model are the allocated resource quantities. We construct a continuous time Markov chain model for the activity network and use the PhaseType distribution to evaluate the project completion time. Then we use Fibonacci search over the interval of permissible allocations to the activity to seek the minimum expected cost.

Key words: Fibonacci search; resource allocation; stochastic network; continuous time Markov chain.

1. Introduction

Projects are defined as a finite set of activities that are required to achieve a specific objective; an activity is defined as an action that consumes time and resources. Typically, the use of resources entails an expanse that varies with the magnitude and duration of their use, and the project has a promised completion date which should be considered, else it incurs a tardiness penalty. In this paper we are concerned with the optimal allocation of resources to activities in project under stochastic condition in order to optimize an economic objective function composed of the sum of the cost of resources used and the penalty incurred if the project is completed later than its promised completion date. A great deal of research has been done pertaining to project scheduling since the field's inception [6],[17] [12].

The objective of the RCPSP studies is to schedule activities in such a manner to meet resource availability and precedence constraints in order to optimize an objective function typically related to the project completion time. The RCPSP can be subdivided into unimodal and multimodal cases. unimodal, or single-mode problem, imply that each activity has a single execution mode. In order words both the activity duration and its requirement for a set of resources are fixed and Known. In multimodal problem, on the other hand, each activity can be processed in one of several modes. Resource constraints can be classified as renewable or non-renewable. Renewable resource constraints put limits on the amount of a particular resource that can be utilized at a particular time in the project. Non-renewable resource constraints limit the amount of a particular resource that can be utilized in total throughout the project's life. The multimodal RCPSP is closely related to the problem at hand. These problems are often referred to as Time-Cost Trade off problems (TCTP). At its core, the TCTP assumes that the duration of an activity is a function of the resources assigned to its completion. Normally, greater resource allocation will demand a greater expense, thus we have a trade off between the time to complete a project and its cost. When the task duration is a linear function of the assigned resources we have the Linear Time Cost Trade off (LTCT) problem. Notably, Fulkerson [11] studied this problem under the objective of minimizing project cost subject to completion before a due date. He then solved the problem through the use of primal-dual concepts in linear programming and developed an approach to compute the complete cost curve of the project as a function of the project's due dates [6], [11], [19], [25].

* Corresponding author E-mail: afsharnb@ alum.sharif.edu

If the resource's allocation is limited to distinct values we have the Discrete Time-Cost Trade off (DTCT) problem. Hindelang and Muth developed a dynamic programming (DP) algorithm to solve this problem in pseudo-polynomial time when the corresponding AOA project graph is series-parallel. [14]. In the case where the graph is not series-parallel, De et al provide an update to the DP algorithm of Hindelang and Muth [4].

Valls et al. study the RCPSP with some activities facing stochastic interruptions and processing times [24]. Gutjahr et al. consider scheduling problems similar to our own. However, in their approach activities can have one of two possible distributions depending on whether or not a project is crashed [13].

2. Continuous Time Markov Chains

In this paper, we assume that the work content of any task is an exponentially distributed random variable. As a direct result of this assumption, activity durations are also exponential random variables since the duration is derived from the work content via the relation Y = W/x, in which **W** is the (random) work content and **x** is the resource Allocation. Further, when we assume independence of activity times with respect to one another we have a Markov PERT Network (MPN) [1], [2].

The term Markov PERT Network originated with Kulkarni and Adlakha and referred to stochastic activity networks where activity times followed an exponential distribution. Further, they provided a method for analysis involving uniformly directed cut sets (UDC's) that allowed one to transformed the PERT network into a Continuous Time Markov Chain [15], [5].

3. Resources Optimal Allocation

The literature has been of little help on the problem of resource allocation in project network. Tereso et al. considered resource allocation in multi-modal activity networks [20], [23], [22], [23]. They address the stochastic version of the problem and assume exponential distribution for the work content of activities. Morgan [18] derived a fast method for optimizing resource allocation in a single stage stochastic activity network. Morgan used Sample Path Optimization and appealed to Geometric Programming to solved the single stage problem. Morgan stated that his procedure could be applied to any distribution of work content provided it is amenable to random sampling.[18],[20],[21],[22],[23].
Elmaghrby and Ramachandra studied the problem of optimal resource allocation in a Markov PERT Network with respect to an economic objective [7], [8], [9]. Their objective function was comprised of the expected cost of the resources and the cost of expected tardiness. They described a "Policy Iteration-like" approach to achieve the optimum in finite number of steps. Azaron et al. used control theory to studied resource allocation in Markov PERT Networks [2].

4. Problem Definition

Given a multimodal activity network ('multimodal' means that each activity can be performed at any number of levels of resource intensity applied to it, with resulting shorter or longer duration), with a stochastic work content (W_j), we wish to decide on the amount of resource to apply to each activity (x_j), so that the total cost is minimized. This cost includes the resource cost and the delay cost. The duration of an activity depends on its work content and on the amount of resource allocated to it. To evaluate the delay cost, a due date must be specified (T_s), as well as the unit cost per period tardy (c_L) [3], [26].

We imposed the following assumptions:

- The work content of each activity is a random variable that follows an exponential distribution with parameter λ_j : $Wj \sim \exp(\lambda_j)$.
- The amount of resource applied to any activity is bounded from below and from above; $l_j \leq x_j \leq u_j$, for all activity.
- The availability of the resource is unlimited and is a continuous variable that may be allocated in any intensity within an interval between lower and upper bounds.

Note that this assumption does not conflict with the earlier assumption of unlimited resource availability. It only contains the resource allocation to any individual activities.

- We assume the total resource availability is abundant enough to accommodate all activities running in parallel at a given time.
- We assume that the cost of resource allocation to activity J is quadratic in allocation over the duration of the activity [20], [21], [22], [23].

4.1. Project completion time analysis in PERT networks

Kulkarni and Adlakha developed a methodology for transforming a project network with exponentially distributed activity completion times into a CTMC. We will now describe their notation and methodology [15].
Let G = (V,A) be a PERT network with set of nodes V = $\{v_1, v_2, \ldots, v_m\}$ and set of activities A = $\{a_1, a_2, \ldots, a_n\}$. Duration of activity $a \in A$ is either an exponentially distributed with the parameter λ_a, or Erlang distributed with the parameters (λ_a, n_a).
First, we transform the original PERT network into a new one, in which all activity durations have exponential distributions. To do that, we substitute each Erlang activity with the parameters (λ_a, n_a) with n_a series of

exponential activities with the parameter λ_a. Now, Let G' = (V',A') be the transformed network, in which V' and A' represent the sets of nodes and arcs of this transformed network, respectively. The source and sink nodes are denoted by s and t, respectively. For a $\in$ A', let $\alpha(a)$ be the starting node of arc a, and $\beta(a)$ be the ending node of arc a

Definition 1. Let I(v) and O(v) be the sets of arcs ending and starting at node v, respectively, which are defined as follows:

$$I(V)=\{a\in A': \beta(a)=V\} \quad (v\in V') \tag{1}$$

$$O(V)=\{ a\in A': \alpha(a)=V\} \quad (v\in V') \tag{2}$$

Definition 2. If $X \subset V'$, such that $s \in X$ and $t \in \bar{X} = V'- X$, then an (s, t) cut is defined as:

$$(x,x)=\{ a\in A' : \alpha(a) \in X , \beta(a) \in \bar{X}\} \tag{3}$$

An (s, t) cut $(X, \bar{X})$ is called an uniformly directed cut (UDC), if$(X, \bar{X})$ is empty.

Example 1. Before proceeding, we illustrate the material by an example. Consider the network shown in Fig. 1.

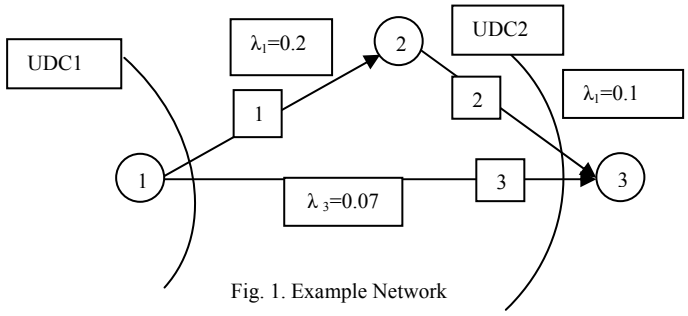

Fig. 1. Example Network

Clearly, (1, 2) is a uniformly directed cut (UDC), because V' is divided into two disjoint subsets X and $\bar{X}$, where $s \in X$ and $t \in \bar{X}$. The other UDCs of this network are (2, 3).

Definition 3. Let $D = E \cup F$ is a uniformly directed cut (UDC) of a network. Then, it is called an admissible 2-partition, if $I(\beta(a)) \not\subset F$, for a $\in$F.

Definition 4. During the project execution and at time t, each activity can be in one of the active, dormant or idle states, which are defined as follows:

Active: an activity is active at time t, if it is being executed at time t.

Dormant: an activity is dormant at time t, if it has finished but there is at least one unfinished activity in $I(\beta(a))$. If an activity is dormant at time t, then its successor activities in O (β (a)) cannot begin.

Idle: an activity is idle at time t, if it is neither active nor dormant at time t.

The sets of active and dormant activities are denoted by Y(t) and Z(t), respectively, and X(t) =(Y(t),Z(t)).

Tabel 1

All admissible 2-partition cuts of the example network

1. (1a, 3a)
2. (2a, 3a)
3. (1a, 3d)
4. (2a, 3d)
5. (2d, 3a)
6. ($\emptyset, \emptyset$)

Consider Example 1, again. If activity 2 is dormant, it means that this activity has finished but the activity 3 is still active. Table 1, presents all admissible 2-partition cuts of this network. E contains all active while F includes all dormant activities. Now, let S denote the set of all admissible 2-partition cuts of the network, and $s=s\cup\{(\emptyset,\emptyset)\}$.Note that $X(t) = (\emptyset,\emptyset)$ implies that $Y(t) = \emptyset$ and $Z(t) = \emptyset$, i.e. all activities are idle at time t and hence the project is completed by time t. It can be proved that $\{X(t), t \geq 0\}$ is a continuous-time Markov process with state space $\bar{S}$.[1],[2]

As mentioned before, E and F contain active and dormant activities of a UDC, respectively. When activity a finishes (with the rate of λ_a), and there is at least one unfinished activity in $I(\beta(a))$, it moves from E to a new dormant activities set, i.e. to F'. Furthermore, if by finishing this activity, its succeeding ones, O(β (a)), become active, then this set will also be included in the new E', while the elements of I(β (a)), which one of them belongs to E and the other ones belong to F, will be deleted from these sets. Thus, the elements of the infinitesimal generator matrix $Q = [q\{(E,F),(E',F')\}],(E,F)$ and$(E',F')] \in S'$, are calculated as follows:

$$\vartheta\{(E,F),(E',F')\} = \begin{cases} \lambda a & if\ a \in E, I(B(a) \not\subset FU\{a\}, E' = E - \{a\}, F' = FU\{\phi\} & (a) \\ \lambda a & if\ a \in E, I(B(a) \subset FU\{a\}, E' = (E - \{a\} UO(B(a)) & (b) \\ \sum_{a \in A} \lambda & if\ E' = E, F' = F & (c) \\ 0 & 0 ther\ wise & (d) \end{cases}$$

In Example 1, if we consider E = {1, 3}, F = (∅), E' = {2, 3} and F' = (∅), then E' = (E _ {1}) ∪O(β (1))], and thus from (4b), q{(E,F), (E',F')} = λ_1.
{X(t), t ≥ 0} is a finite-state absorbing continuous-time Markov process. Since q{(∅,∅), (∅,∅)} = 0, this state would be an absorbing one and obviously the other states are transient. Furthermore, we number the states in S such this Q matrix be an upper triangular one.[1],[2].

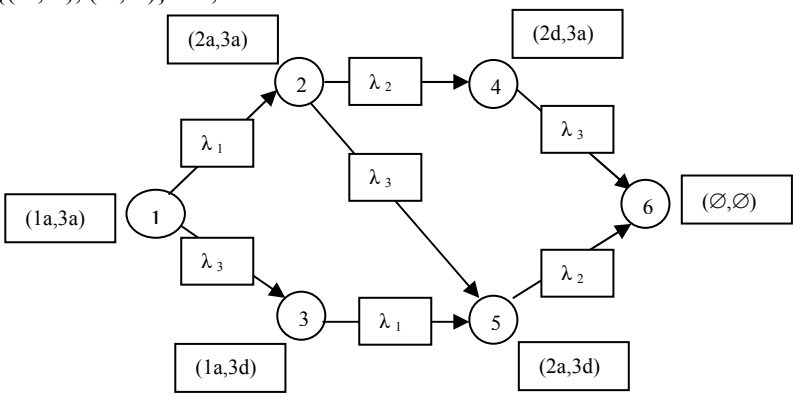

Fig. 2**.** Graphical Representation of the six states of the CTMC

4.2. Continuous Phase-Type Distributions

The name "phase type" distribution stems from the fact that an Erlang distribution is derived as the sum of "stages" or "phases", all exponentially distributed with the same parameter λ. The generalized Erlang distribution of order m has m phases (stages) each is exponentially distributed but with possibly different parameters $\lambda_1, \cdots, \lambda_m$.[1], [16], [20], [21], [22], [6].

Definition 5. A continuous probability distribution F(·) is of the phase type (PH-distribution)
if it is the distribution of the time until absorption in a finite-state Markov process with a single absorbing state; that is, there exists a probability vector (α, α_{m+1}) and an infinitesimal generator matrix of the form

$$Q = \begin{bmatrix} T_{(m\times m)} & T^0_{(m\times 1)} \\ O_{(1\times m)} & O_{(1\times 1)} \end{bmatrix}$$

where e is a m × 1 vector of ones, Ti,i < 0 for 1 ≤ i ≤ m, and Ti,j ≥ 0 for i ≠ j.

Also,

$$Te + T^0 = 0 \qquad (4)$$

and the initial probability vector of Q is given by the vector (α, α_{m+1}), with

$$\alpha \cdot e + \alpha_{m+1} = 1 \qquad (5)$$

The pair (α,T) is called 'representation' of F (·) . In our case, the process starts in state 1 with probability 1. States 1, . . . , m are transient so that absorption into state m + 1 from any initial state is certain.
The first result is that the matrix T is non-singular (a necessary and sufficient condition for the states 1, . . . , m to be transient). Observe that

$$T^k \longrightarrow 0 \text{ as } k \longrightarrow \infty.$$

Assuming an initial probability vector (α, α_{m+1}), the c.d.f. of the time to absorption in state m + 1 corresponding to the initial probability vector (α, α_{m+1}) is given by

$$F(z) = 1 - \alpha \cdot e^{Tz} \cdot e, \text{ for } z \geq 0 \qquad (6)$$

We make the following observations about the properties of the distribution F (·):
It has a jump of height αm+1at z = 0. Evidently this is the probability that the process starts in the absorbing state. This case is of no concern to us since it implies that the project is complete at its start, which would imply that there is no project.
Its density portion f (z) = F' (z) on (0, ∞) is given by

$$f(z) = F'(z) = \alpha \cdot e^{Tz} \cdot T^0 \qquad (7)$$

In our case the "portion" of the domain of x is the whole real line including the point at the origin because $\alpha_{m+1} = 0$.

The non-central moments (about the origin) Φ'_i of F (·) are all finite and given by

$$\Phi'_i = (-1)^i * i!(\alpha T^{-i} e), \text{for } i \geq 0 \qquad (8)$$

5. Description of Algorithm

In order to solve the problem at hand, we focus our research on a modified steepest descent algorithm. We first compute an approximation to the partial derivative of the expected cost with respect to the allocation to each of the activities and then choose to modify the allocation to the activity that has the greatest negative derivative. Next we use line search techniques to find the minimum expected cost allocation to the chosen activity while holding the other allocations constant. The algorithm iterates using these two steps until a specified stopping criterion is met. In order to execute any of these steps, we first must be able to compute the expected cost at a particular allocation.

5.1. Cost Computation

As previously described, the expected cost of the project contains two terms:

$$E[C(X)]=\Sigma(X_a/\lambda_a)+CL.\{\max(0,m_1-T_s)\} \quad (9)$$

The first term relates exclusively to the level of resource allocation. Since the work content distribution parameter, λ_i, is given for each activity the expected cost of resource allocation can be computed directly. The second term, which relates to the lateness of the project with respect to a given due date, T_s, can be computed by the non-centeral moments of the phase-type distribution.
As previously described, the non-centeral moments function of the phase-type distribution is obtained following the relation:

$$m_k=(-1)^k*k!(\alpha T^{-k}e) \quad (10)$$

Since that we use the p-state Erlang distribution in our computation then:

$$f(u)=\frac{\lambda^p u^{p-1} e^{-\lambda u}}{P!} \quad (11)$$

$$\alpha=(1\ 0\ 0\ \dots\ 0) \quad (12)$$

$$T=\begin{bmatrix} -\lambda & \lambda & 0 & 0 & \dots\dots & 0 \\ 0 & -\lambda & \lambda & 0 & \dots\dots & 0 \\ 0 & 0 & -\lambda & \lambda & \dots\dots & 0 \\ 0 & 0 & 0 & 0 & \dots\dots & \lambda \end{bmatrix}$$

With consider the above information we can compute the expected cost of lateness based on the following relation:
Lateness cost expected = $c_l*\{\max(0,$ project complete real time-$T_s)\}$
The expected of project complete time is as first-stage moment of phase-type distribution that with consider the k-stage moment then:

$$m_1=(-1)^1*1!(\alpha T^{-1}e)=-\alpha T^{-1}e \quad (13)$$

that in the above relation ,e, is a ones vector and α is :

$$\alpha=(1\ 0\ 0\ 0\ \dots\ 0) \quad (14)$$

and with substitution in 5.3 relation we have :

$$\text{Lateness cost expected}=c_l*\{\max(0,-\alpha T^{-1}e-T_s)\} \quad (15)$$

With consider the above relation the total costs expect is:

$$E[C(X)]=\Sigma(X_a/\lambda_a)+C_L.\{\max(0,-\alpha T^{-1}e-T_S)\} \quad (16)$$

5.2. Selecting Candidate Activities

In general, our algorithm optimizes the project cost by changing the allocation to one activity at-a-time. We now need to explain the procedure to select activities that are the best candidates for optimization. The best candidates in our procedure are those that could lead to the greatest decrease in expected project cost.

The two terms of the project cost behave diametrically opposite in their response to changing resource allocations. The project resource cost increases linearly with respect to an increase in resource allocation to any activity. On the other hand, increased resource allocation to the activities tends to shorten their expected duration and so the expected lateness cost would decrease with respect to such a change in allocation. A decrease in overall cost, therefore, could be obtained from either a decrease or an increase in resource allocation.

An important observation is that the expected project cost is a convex function with respect to the allocation to a single activity. This fact is a result of the two convex components of cost. The expected cost of lateness decreases convexly due to the exponential distributions on which it is based. Further, since the resource costs are linear (hence convex) in the resource allocation, the sum of the two costs is convex. Thus, repeatedly optimizing allocations to single activities one-at-a-time will descend monotonically to reach the optimal solution. Since no analytical expression is known for the partial derivative of the cost function with respect to the allocation to any activity a_j, we must rely on approximations to proceed.
Such an approximation would enable us to select the activities to which a change in allocation could cause the greatest change in expected cost. If we compute the cost associated with a small change in allocation, δ, to a particular activity we can approximate the derivative of the cost function by first taking the difference between this cost and the cost of the initial resource allocation and then dividing by the magnitude of δ. Since a decrease in cost can occur via an increase or a decrease in allocation, and thanks to the convexity of the cost function, the best candidate allocations are those with the steepest- descent causing a decrease in cost. Further, if δ is constant across all activities, we can simply find the change in allocation reflecting the greatest decrease in cost. Further, if δ is

constant across all activities, we can simply find the change in allocation reflecting the greatest decrease in cost. Letting $\boldsymbol{X}_{+a} = \boldsymbol{X} + \delta$, denote the allocation $\boldsymbol{X}$ with the value of the a th allocation increased by the amount δ, further we allow $\boldsymbol{X}_{-a} = \boldsymbol{X} - \delta$ to denote the allocation with the a th allocation decreased. We calculate $E[C(\boldsymbol{X}_{+a})]$ and $E[C(\boldsymbol{X}_{-a})]$ for each activity. With $E[C(\boldsymbol{X})]$ in hand, we find $E[C(\boldsymbol{X}_{+a})] - E[C(\boldsymbol{X})]$ and $E[C(\boldsymbol{X}_{-a})] - E[C(\boldsymbol{X})]$ for each activity, a, and use this as a value of the derivatives of the cost function with respect to the allocation to a. We select the activity corresponding to the greatest decrease in cost as the candidate activity. If the decrease in cost comes from $\boldsymbol{X}_{+a}$, the allocation to activity a must be increased. Likewise, if this comes from $\boldsymbol{X}_{-a}$, we must decrease the allocation to activity a. Since we are dealing with minimization of the cost function, for certain network structures both increasing and decreasing the allocation can cause an increase in cost. Thus it is important to check both an increase and decrease in allocation rather than assuming that an increase in cost in one direction implies a decrease in cost in the opposite.

5.3. Improving Computational Elements of Total Cost

With respect to the cost function in section 4.1 we must calculate $E[C(\boldsymbol{X}_{+a})]$ and $E[C(\boldsymbol{X}_{-a})]$ for each activity**.** We let $\boldsymbol{Q}$ $(\boldsymbol{X}_{+a})$ represent the matrix $\boldsymbol{Q}$ with the allocation to activity a increased by the value δ. Note that increasing the allocation to a single activity may correspond to increasing the value of several entries in the original $\boldsymbol{Q}$ matrix, as well as the decreasing their corresponding elements along the main diagonal. Changing the allocation to activity i by δ implies that we must change the values in the $\boldsymbol{Q}$ matrix by $\delta\lambda_i$. This change can be viewed as matrix addition. These facts are best illustrated with an example from the project in Figure 1, where we change the allocation to activity 1 ($\lambda_1 = 0.2$) with$\delta = 0.1$:

$$Q(X_{+1})= \begin{bmatrix} -0 .27-0.02 & 0.2+0.02 & 0.07 & 0 & 0 & 0 \\ 0 & -0.17 & 0 & 0.07 & 0.1 & 0 \\ 0 & 0 & -0.2-0.02 & 0.2 +0.02 & 0 & 0 \\ 0 & 0 & 0 & -0.1 & 0 & +0.1 \\ 0 & 0 & 0 & 0 & -0.07 & 0.07 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Q(X_{+1})= Q+ \begin{bmatrix} -0.02 & 0.02 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.02 & 0.02 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Q(X_{+1})= Q+\Phi_{+1} \qquad (17)$$

where, Q_{+a} represents the matrix corresponding to the increase in allocation to activity a.

Note that the matrix Q_{+a} ,does not depend on the current allocation to the other activities. Here are several ways in which you can enter and format.

5.4. Optimizing the Allocation to a Candidate Activity

With a candidate variable in hand, we now seek to optimize the allocation of resources to that activity while leaving the other allocations unchanged. As previously stated, the expected project cost is a convex function with respect to the allocation to a single activity. Any convex optimization procedure could therefore be applicable here. However, due to the difficulty involved in finding exact analytical expressions for the partial derivatives of the cost function, we opt to use Fibonacci search as our method of determining the optimal allocation to the selected activity. Note that this "optimal" allocation is "locally optimal" in a sense as it depends upon the allocation to the other activities.

At its core, Fibonacci search, often referred to as "golden mean" search, finds the optimal point in a range of feasible values by repetitively shrinking the range, stopping when the range is sufficiently small to suggest a single optimal point. We define r as the inverse of the "golden ratio" and use r to give values for $\boldsymbol{X}_l$ and $\boldsymbol{X}_u$.

$r = 1/1.618 \approx 0.618$ and $1-r = 0.382$

Wilde shows is the most computationally efficient method of finding the optimal value of a variable when searching along a line. Given a lower bound, $\boldsymbol{l}$, and an upper bound, $\boldsymbol{u}$, on the range, two new points are calculated within the range: $\boldsymbol{X}_l$ and $\boldsymbol{X}_u$ where $\boldsymbol{X}_l < \boldsymbol{X}_u$ **:**

$$X_L = L+(U-L)*0.382 \qquad (18)$$

$$X_U = L+(U-L)*0.618 \qquad (19)$$

For our purposes, these points represent different resource allocations [7].

If activity j was selected as the candidate activity and its cost decreases with increased allocation we let $\boldsymbol{l}$ be the current allocation and let $\boldsymbol{u}$ be upper bound for activity j while keeping the allocations to the other activities unchanged. If its cost decreases with decreased allocation, we define the upper bound as the current allocation and let $\boldsymbol{l}$ be the lower bound on the allocation to activity j. With these four values in hand ($\boldsymbol{l}$, $\boldsymbol{u}$, $\boldsymbol{X}_l$, and $\boldsymbol{X}_u$), we recursively redefine the new bounds on the range to our search to the minimum expected cost allocation. Optimization proceeds by first calculating the expected costs of allocations $\boldsymbol{X}_l$ and $\boldsymbol{X}_u$. if $E[C(\boldsymbol{X}_l)] < E[C(\boldsymbol{X}_u)]$ then the optimal value must lie between $\boldsymbol{l}$ and $\boldsymbol{X}_u$, otherwise $E[C(\boldsymbol{X}_l)] > E[C(\boldsymbol{X}_u)]$ and the optimal value must lie between $\boldsymbol{X}_l$ and $\boldsymbol{u}$. If the two expected costs are equal, either bound can be used.

5.5. Stopping Criteria

The optimizing algorithm repeats the derivative approximation and Fibonacci search steps, finding a resource allocation with a lower expected cost in each iteration. The algorithm stops when the expected cost in a given iteration improved the expected cost by only a small amount. If $\boldsymbol{X}_k$ represents the best resource allocation after iteration k, the algorithm terminates when:
$E[C(\boldsymbol{X}_{k-1})] - E[C(\boldsymbol{X}_k)] < \in_p$

5.6. An Illustrative Example

Consider the project represented by the graph in Figure 1. We assume an initial allocation of $\boldsymbol{X}_0 = (1, 1, 1)$ with lower and upper bounds on allocation defined at 1 and 3, respectively. In our calculations we let δ =0.05, c_L = 3, T_s = 8, and $\in_p$ =0 .005. First we compute the expected cost of the initial allocation: $E[C(\boldsymbol{X}_0)]$ = 68.9517. Next, we approximate the derivative of the expected cost with respect to the allocations to activities 1, 2, and 3**(Table 2)**:

Table 2
Example Derivative Approximation

Activity	X_a	$E[C(X_a)]$	$E[C(X_a)]$-$E[C(X_0)]$
1, increase	(1.05،1،1)	68.7290	-0.2227
1, decrease	(0.95،1،1)	69.2479	0.2962
2, increase	(1،1.05،1)	68.4048	-0.5469
2, decrease	(1،0.95،1)	69.6427	0.691
3, increase	(1،1،1.05)	68.2031	-0.7486
3, decrease	(1،1،0.95)	69.9052	0.9535

Since activity 3, yields the approximate derivative of greatest decrease in cost, it becomes the candidate activity. Thus, we take the bounds l= (1, 1, 1) and u = (1, 1, 3) and we begin the Fibonacci search procedure with $\in_d$ =، .01. The following table (Table 3) details the optimization procedure: $\boldsymbol{X_5}$ = (1.4306, 1.4977, and 1.4796) at a cost of 62.3555. Ramachandra &Elmaghrby solved the same example problem, though they limited his resource allocations to 0.25 increments. [7] His procedure resulted in a solution of $\boldsymbol{X}^*$ = (1.5, 1.5, 1.5) at an expected cost of 62.38.

Table 3
Fibonacci Search Procedure Example

Iteration	L_3	U_3	X_{L3}	$E[C(X_{L3})]$	X_{U3}	$E[C(X_{U3})]$	U_3-L_3
0	1	3	1.7640	67.9215	2.2360	72.1344	2
1	1	2.2360	1.4722	66.4928	1.7638	67.9202	1.236
2	1	1.7638	1.2918	66.4519	1.4720	66.4923	0.7638
3	1	1.4720	1.1803	66.9318	1.2917	66.4522	0.4720
4	1.1803	1.4720	1.2917	66.4522	1.3606	66.3659	0.2917
5	1.2917	1.4720	1.3606	66.3659	1.4032	66.3789	0.1803
6	1.2917	1.4032	1.3343	66.3823	1.3606	66.3659	0.1115
7	1.3343	1.4032	1.3606	66.3659	1.3769	66.3653	0.0689
8	1.3606	1.4032	1.3769	66.3653	1.3869	66.3684	0.0426
9	1.3606	1.3869	1.3706	66.3647	1.3769	66.3653	0.0263
10	1.3606	1.3769	1.3668	66.3648	1.3706	66.3647	0.0163
11	1.3668	1.3769	1.3707	66.3647	1.3730	66.3648	0.0101
12	1.3668	1.3730	1.3692	66.3647	1.3706	66.3647	0.0062

6. Computational Results

The algorithm was tested on a set of 90 networks representing a variety of project network structures. Project networks were generated randomly using RanGen2, developed by Vanhoucke et al [25] The RanGen2 generator allows the user to select networks by specifying values of six parameters. Work content distribution parameters were generated randomly in MATLAB. These parameters were sampled from a uniform distribution between 0.1 to 2 (table 4),2 to 4 (table 5),4 to 6 (table 6). The lower bound on resource

allocation was taken as 1 and the upper bound was 3 units for all activities. Relative lateness costs, *cL*, were fixed at 3 for all examples and $\in_p$ was fixed at 0.005.
Testing was conducted using the optimization algorithm initiated from 5 input allocations for each network; the optimization procedure was repeated 5 times, each with a different initial resource allocation. Details regarding the test cases and the test results can be found in Tables 4, 5 and 6.

Table 4
Minimum and average cost and CPU time of proposed algorithm compared with Ramachandara & Elmaghraby algorithm (Work content between 0.1 to 2)

network	Indicators as RanGen2					Number of States CTMC	Min CPU Time, sec	Average CPU Time, sec	Min cost (Proposed algorithm)	Min cost (Ramachandara & Elmaghraby)	Average CPU Time, sec (Ramachandara & Elmaghraby)
	I_1	I_2	I_3	I_4	I_5						
1	8	0.25	0	0.16	1	74	0.2656	0.3844	23.457	23.7495	0.5781
2	8	0.25	0	0.83	1	33	0.0781	0.2813	24.1752	24.5335	0.2344
3	8	0.25	1	0.16	1	160	1.3438	2.0812	21.2507	23.1238	3.1406
4	8	0.25	1	0.83	1	130	0.7813	0.8469	30.5796	21.4961	2.0313
5	8	0.25	0.33	1	1	39	0.0938	0.2625	25.2088	25.5161	0.25
6	8	0.75	0.8	1	1	11	0.0938	0.1313	37.8066	38.0995	0.1563
7	8	0.75	0.8	0.5	1	18	0.0781	0.1094	37.7046	37.7862	0.1719
8	8	0.75	0.8	0.5	0.5	17	0.0938	0.1187	37.2581	38.3859	0.1563
9	8	0.75	0.99	0	1	25	0.1406	0.1781	36.0264	36.9784	0.2188
10	8	0.75	0.99	0	0.25	21	0.0781	0.1125	34.6114	36.0189	0.1875
11	12	0.25	0	0.5	0.8	77	1.2813	1.5594	32.2444	32.889	0.3594
12	12	0.25	0	0.16	1	228	4.3906	10.8031	29.5245	32.5352	3.2831
13	12	0.25	0.33	0.21	0.66	220	4.4844	13.143	29.7745	30.5352	3.1094
14	12	0.75	0.75	1	1	16	0.1719	0.2375	49.1488	49.4606	0.1094
15	12	0.75	0.75	0.5	1	18	0.1563	0.2344	50.1333	50.5557	0.0781
16	12	0.75	0.75	0.5	0.61	20	0.1094	0.15	50.5208	50.9168	0.1094
17	12	0.75	0.99	0	1	80	0.6094	0.6719	50.6085	50.0889	0.2969
18	12	0.75	0.99	0	0.28	68	0.3281	0.4031	47.8124	49.4768	0.4063
19	16	0.25	0.22	1	1	93	2.7031	3.0063	41.9197	42.3425	0.7831
20	16	0.75	0.9	0	1	176	6.2188	6.9875	59.2728	59.9054	3.2969
21	16	0.75	0.72	0	0	126	1.9063	2.2563	60.225	59.8496	1.3906
22	16	0.75	0.72	0	1	118	2.1250	2.6781	59.7431	59.9645	1.2031
23	16	0.75	0.72	0	0.5	64	0.4375	0.5938	59.4664	59.0757	0.3906
24	16	0.75	0.81	0.5	0.92	26	0.375	0.45	59.1842	59.9391	0.1875
25	20	0.25	0.08	0.82	1	224	13.475	15.971	65.7850	65.8190	7.0151
26	20	0.75	0.71	0	1	202	10.8125	14.6844	66.9252	67.8665	6.0469
27	20	0.75	0.71	0	0.71	66	0.8281	0.8438	65.2744	66.1366	0.5156
28	20	0.75	0.71	0.42	0.9	178	6.3438	9.1312	67.6033	67.9182	4.3281
29	20	0.75	0.78	1	1	28	0.5469	0.6281	68.2606	68.7138	0.2031
30	20	0.75	0.78	0.62	1	31	0.7031	0.7531	68.6739	69.1353	0.2344

Table 5

Minimum and average cost and CPU time of proposed algorithm compared with Ramachandara & Elmaghraby algorithm (Work content between 2 to 4)

network	Indicators as RanGen2					Number of States CTMC	Min CPU Time, sec	Average CPU Time, sec	Min cost (Proposed algorithm)	Min cost (Ramachandara &Elmaghraby)	Average CPU Time, sec (Ramachandara &Elmaghraby)
	I_1	I_2	I_3	I_4	I_5						
1	8	0.25	0	0.16	1	74	0.1563	0.1906	5.6026	5.1558	0.6406
2	8	0.25	0	0.83	1	33	0.0469	0.0688	5.0840	5.1558	0.2500
3	8	0.25	1	0.16	1	160	0.8125	0.8812	5.0600	5.1558	2.4844
4	8	0.25	1	0.83	1	130	0.5469	0.5781	5.0300	5.1558	1.5156
5	8	0.25	0.33	1	1	39	0.0625	0.0813	4.8900	5.1558	0.2188
6	8	0.75	0.8	1	1	11	0.0313	0.0563	4.7400	5.1558	0.1406
7	8	0.75	0.8	0.5	1	18	0.0313	0.0563	5.0900	5.1558	0.1406
8	8	0.75	0.8	0.5	0.5	17	0.0313	0.0563	4.7500	5.1558	0.1250
9	8	0.75	0.99	0	1	25	0.0469	0.0750	4.4461	5.1558	0.1536
10	8	0.75	0.99	0	0.25	21	0.0313	0.0594	5.2045	5.1558	0.1875
11	12	0.25	0	0.5	0.8	77	0.2188	0.2375	8.3667	9.5901	0.2813
12	12	0.25	0	0.16	1	228	2.5938	2.9469	8.2595	9.5901	2.8438
13	12	0.25	0.33	0.21	0.66	220	2.5000	2.8000	8.1880	9.5901	0.1094
14	12	0.75	0.75	1	1	16	0.0469	0.0813	11.8230	9.5901	0.0938
15	12	0.75	0.75	0.5	1	18	0.0625	0.0813	8.5453	9.5901	0.0938
16	12	0.75	0.75	0.5	0.61	20	0.0469	0.0844	9.3016	9.5901	0.3281
17	12	0.75	0.99	0	1	80	0.2500	0.2938	7.2338	9.5901	0.2656
18	12	0.75	0.99	0	0.28	68	0.0938	0.1469	7.6864	9.5901	0.2344
19	16	0.25	0.22	1	1	93	0.3750	0.4156	11.0511	11.9808	0.3906
20	16	0.75	0.9	0	1	176	1.8594	1.9406	10.0459	11.9808	1.5625
21	16	0.75	0.72	0	0	126	0.8594	0.8906	10.0106	11.9808	0.7344
22	16	0.75	0.72	0	1	118	0.6875	0.7250	10.5003	11.9808	0.6563
23	16	0.75	0.72	0	0.5	64	0.1875	0.2062	10.7162	11.9808	0.2344
24	16	0.75	0.81	0.5	0.92	26	0.0625	0.0969	10.7580	11.9808	0.1094
25	20	0.25	0.08	0.82	1	224	15.5648	17.2568	15.0010	14.1161	2.5011
26	20	0.75	0.71	0	1	202	3.5156	3.5875	12.3616	14.1161	2.5000
27	20	0.75	0.71	0	0.71	66	0.2656	0.3063	11.5614	14.1161	0.2344
28	20	0.75	0.71	0.42	0.9	178	2.2656	2.3969	11.1153	14.1161	1.5469
29	20	0.75	0.78	1	1	28	0.0781	0.1187	11.6613	14.1161	0.1250
30	20	0.75	0.78	0.62	1	31	0.0938	0.1250	11.9852	14.1161	0.1563

Table 6

Minimum and average cost and CPU time of proposed algorithm compared with Ramachandara & Elmaghraby algorithm (Work content between 4 to 6)

network	Indicators as RanGen2					Number of States CTMC	Min CPU Time, sec	Average CPU Time, sec	Min cost (Proposed algorithm)	Min cost (Ramachandara &Elmaghraby)	Average CPU Time, sec (Ramachandara &Elmaghraby)
	I_1	I_2	I_3	I_4	I_5						
1	8	0.25	0	0.16	1	74	0.1406	0.1596	3.4964	3.6884	0.2813
2	8	0.25	0	0.83	1	33	0.0313	0.0625	3.3076	3.6884	0.1563
3	8	0.25	1	0.16	1	160	0.7344	0.7656	2.5637	3.6884	1.2969
4	8	0.25	1	0.83	1	130	0.3125	0.4406	2.3734	3.6884	0.8906
5	8	0.25	0.33	1	1	39	0.0469	0.1094	3.2650	3.6884	0.1563
6	8	0.75	0.8	1	1	11	0.0156	0.0437	2.7133	3.6884	0.0781
7	8	0.75	0.8	0.5	1	18	0.0156	0.0437	3.3269	3.6884	0.0938
8	8	0.75	0.8	0.5	0.5	17	0.0156	0.0469	2.7610	3.6884	0.0781
9	8	0.75	0.99	0	1	25	0.0313	0.0500	3.2623	3.6884	0.1250
10	8	0.75	0.99	0	0.25	21	0.0156	0.0500	2.6876	3.6884	0.1094
11	12	0.25	0	0.5	0.8	77	0.1875	0.2188	4.5422	5.3054	0.2656
12	12	0.25	0	0.16	1	228	2.1406	2.3750	4.4821	5.3054	3.2188
13	12	0.25	0.33	0.21	0.66	220	2.1563	2.2313	4.6764	5.3054	2.8906
14	12	0.75	0.75	1	1	16	0.0313	0.0563	5.0176	5.3054	0.0938
15	12	0.75	0.75	0.5	1	18	0.0313	0.0563	4.0127	5.3054	0.0938
16	12	0.75	0.75	0.5	0.61	20	0.0313	0.0594	4.5733	5.3054	0.0938
17	12	0.75	0.99	0	1	80	0.1875	0.2313	4.5974	5.3054	0.3281
18	12	0.75	0.99	0	0.28	68	0.1719	0.2000	5.1726	5.3054	0.2344
19	16	0.25	0.22	1	1	93	0.3438	0.3688	5.9100	6.8014	0.4063
20	16	0.75	0.9	0	1	176	1.4375	1.4969	6.4102	6.8014	1.6250
21	16	0.75	0.72	0	0	126	0.6250	0.6531	5.9549	6.8014	0.6250
22	16	0.75	0.72	0	1	118	0.2969	0.5250	5.3433	6.8014	0.6563
23	16	0.75	0.72	0	0.5	64	0.1563	0.1844	6.0259	6.8014	0.2031
24	16	0.75	0.81	0.5	0.92	26	0.0313	0.0750	5.7961	6.8014	0.1094
25	20	0.25	0.08	0.82	1	224	3.9652	4.0125	6.2356	8.1932	2.9652
26	20	0.75	0.71	0	1	202	2.4531	2.6500	6.7732	8.1932	2.4688
27	20	0.75	0.71	0	0.71	66	0.2188	0.2437	7.6308	8.1932	0.2656
28	20	0.75	0.71	0.42	0.9	178	1.0469	1.6625	6.6994	8.1932	0.2344
29	20	0.75	0.78	1	1	28	0.0625	0.0938	6.6575	8.1932	1.6719
30	20	0.75	0.78	0.62	1	31	0.0625	0.0969	7.4988	8.1932	0.1094

These results show that this algorithm is relatively efficient as many of the networks were solved a few seconds on average. The computed optimal costs across all 5 input allocations differed by less than 2% in the entire test networks. One important observation that immediately can be seen is the correlation between the number of states in the CTMC and the computation time required for optimization. Clearly, as the number of states in the CTMC grows the computational requirement of the algorithm increases exponentially. This algorithm in comparison with Elmaghrby & Ramachandra method is efficient with respect to quality solution. This topic statistically is proof as following:

Null hypothesis = H_0:$\mu_D = 0$

Inverse hypothesis = H_1: $\mu_D > 0$

α=0.05 , $\bar{D}$=-0.7089, ΣD_i^2=200.804

$S_D=((\Sigma D_i^2 - n\ \bar{D}^2)/n-1)^{1/2}$=1.3221

t_0= -5.0890

$t_{0.05,89}$= 1.665 Acceptance area =$(-\infty,1.665]$

With consider the value of the t_0 statistical and the acceptance limit, the null hypothesis is accepted and the inverse hypothesis is not accepted. In the other words the

procedure is stated in this paper improved the solution quality respect to the Elmaghrby and Ramachandra method. In continuance about the solution time we have:

Null hypothesis = H_0:$\mu_D = 0$

Inverse hypothesis = H1: $\mu_D > 0$

α=0.05 , $\bar{D}$=0.7139, ΣD_i^2=600.6387

$S_D=((\Sigma D_i^2 - n\ \bar{D}^2)/n-1)^{1/2}$=2.4966

t_0=9.4891 Acceptance area = $(-\infty, 1.665]$

That the value of t is not in acceptance area thus Elmaghrby & Ramachandra method is better from Viewpoint solution time.

7. Conclusion

In this paper, we investigated the optimal assignment of resources to the activities of a stochastic project network. We constructed a continuous time Markov chain (CTMC) model for the activity network and we used the PhaseType distribution to evaluate the project completion time. Then we used Fibonacci search over the interval of permissible allocations to the activity to seek the minimum expected cost. Computational results showed that our proposed algorithm is better than Elmaghrby & Ramachandra method with respect to solution quality and computational time.

8. References

[1] Azaron, H. Katagiri, and M. Sakawa. A multi-objective resource allocation problem in PERT networks, European journal of operational research , 2004.

[2] A. Azaron, H. Katagiri, and M. Sakawa. Time cost trade off via optimal control theory in Markov PERT networks. Annals of Operations Research, 2007.

[3] E. Berman, resource allocation in a PERT network under continuous activity time-cost function ، Management science، 1964.

[4] P. De, E. J. Dunne, J. B. Ghosh, and C. E. Wells. Complexity of the discrete time-cost trade off problem for project networks, Operations Research, 1997.

[5] R. F. Deckro, J. E. Hebert, W. A. Verdini, P. H. Grimsrud and S. Venkateshwar, Nonlinear time/cost trade off models in project management, Computers and Industrial Engineering، 1994.

[6] E. L. Demeulemeester, and W. S. Herroelen. Project Scheduling: A Research Handbook Springer, 2002.

[7] S. E. Elmaghraby and G. Ramachandra. Optimal resource allocation in activity networks under stochastic conditions, 2006.

[8] S. E. Elmaghraby, C. D Morgan, Resource allocation in activity networks under stochastic condition: A geometric programming – sample path optimization approach، 2006.

[9] S. E. Elmaghraby, on the expected duration of PERT type networks ، Management science، 1967.

[10] R. M. Freund, The steepest descent algorithm for unconstrained optimization and a bisection line-search method, 2004.

[11] D. R. Fulkerson, A network flow computation for project curves, Management Science, 1961.

[12] D. G. Ginzburg, and A. Gonik, Stochastic network project scheduling with non-consumable limited resources. International Journal of Production Economics, 1997.

[13] W. J. Gutjahr, C. Strauss and E. Wagner. A stochastic branch and bound approach to activity crashing in project management. INFORMS Journal on Computing, 2000.

[14] T. J. Hindelang and J. F. Muth. A dynamic programming algorithm for decision CPM networks, Operations Research, 1976.

[15] V. G. Kulkarni and V. G. Adlakha. Markov and Markov-Regenerative PERT networks, operations Research, 1986 .

[16] Z. Laslo, I. Albert, Goldberg.Resource allocation under uncertainty in a multi-project matrix environment. International Journal of Project Management, 2008.

[17] J. Matthew, G. Szmerekovsky and V. Tilson, Scheduling project with stochastic activity duration to maximize expected net present value, European Journal of Operational Research, 2009.

[18] C. D. Morgan, A sample-path Optimization Approach for Optimal Resource Allocation in Stochastic Projects. North Carolina State University, Department of Industrial and Systems Engineering, 2006.

[19] F. Stork, Stochastic Resource-Constrained Project Scheduling, PhD thesis, Technical University of Berlin, School of Mathematics and Natural Sciences, 2001.

[20] A. P. Tereso, M. M. T. Araújo and S. E. Elmaghraby, Adaptive Resource Allocation in Multimodal Activity Networks, International Journal of Production Economics, 2001.

[21] A. P. Tereso, M. M. Araújo, and S. E. Elmaghraby, Experimental results of an adaptive resource allocation technique to stochastic multimodal projects. Technical report, Universidade do Minho, Guimarães, Portugal, 2003a.

[22] A. P. Tereso, M. M. Araújo, and S. E. Elmaghraby, Basic approximations to an adaptive resource allocation technique to stochastic multimodal projects, Technical report, Universidade do Minho, Guimarães, Portugal, 2003b.

[23] A. P. Tereso, M. M. Araújo, and S. E. Elmaghraby, The optimal resource allocation in stochastic activity networks via the electromagnetism approach. In Ninth International Work shop on Project Management and Scheduling (PMS '04), Nancy, France, 2004.

[24] V. Valls, M. Laguna, P. Lino, A. Perez, and S. Quintanilla, Project scheduling with stochastic activity interruptions, Project Scheduling: Recent Models, Algorithms and Applications, Kluwer Academic Publishers, 1998.

[25] M. Vanhoucke, J. Coelho, D. Debels, B. Maenhout and L. V. Tavares, An evaluation of the adequacy of project network generators with systematically sampled networks, European Journal of Operational Research ،2008.

[26] R. D. Wollmer, Critical path planning under uncertainty, Mathematical Programming Study, 1985.