European Journal of Operational Research xxx (2015) xxx–xxx

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Decision Support

Resource allocation in multi-class dynamic PERT networks with finite capacity

Q2 Saeed Yaghoubi [a], Siamak Noori [a], Amir Azaron [b,c,*], Brian Fynes [c]

[a] School of Industrial Engineering, Iran University of Science and Technology, Tehran, Iran
[b] Department of Industrial Engineering, Istanbul Sehir University, Istanbul, Turkey
[c] Michael Smurfit Graduate School of Business, University College Dublin, Dublin, Ireland

ARTICLE INFO

Article history:
Received 4 May 2013
Accepted 15 June 2015
Available online xxx

Keywords:
Project management
Markov processes
Multiple objective programming
Simulated annealing

ABSTRACT

In this paper, the resource allocation problem in multi-class dynamic PERT networks with finite capacity of concurrent projects (COnstant Number of Projects In Process (CONPIP)) is studied. The dynamic PERT network is modeled as a queuing network, where new projects from different classes (types) are generated according to independent Poisson processes with different rates over the time horizon. Each activity of a project is performed at a devoted service station with one server located in a node of the network, whereas activity durations for different classes in each service station are independent and exponentially distributed random variables with different service rates. Indeed, the projects from different classes may be different in their precedence networks and also the durations of the activities. For modeling the multi-class dynamic PERT networks with CONPIP, we first consider every class separately and convert the queueing network of every class into a proper stochastic network. Then, by constructing a proper finite-state continuous-time Markov model, a system of differential equations is created to compute the project completion time distribution for any particular project. The problem is formulated as a multi-objective model with three objectives to optimally control the resources allocated to the service stations. Finally, we develop a simulated annealing (SA) algorithm to solve this multi-objective problem, using the goal attainment formulation. We also compare the SA results against the results of a discrete-time approximation of the original optimal control problem, to show the effectiveness of the proposed solution technique.

1. Introduction

Nowadays, multi-project scheduling is widely used because of the need to consider all projects in the context of an organization as one system where limited resources are shared among multiple projects. Moreover, the project-oriented approach is used in some organizations to schedule operations, and operations are performed depending on the projects. As such, the multi-project management is an attracting widespread attention in project scheduling and management, whereas, conventional project scheduling has focused primarily on single project optimization based on task dependency constraints.

Obviously, scheduling of multi-project systems is more difficult than scheduling of a single project and the problem would be more difficult to schedule when the activity durations are stochastic. On the other hand, in many organizations, not only the activity durations are uncertain, but some new projects are also generated dynamically over the time horizon. In this occasion, multi-project scheduling would be more complex than before. Such a problem is denominated as *"Dynamic PERT Network"* and is suitable for organizations which execute similar projects, for example maintenance projects. Indeed, there are many jobs with a similar structure of activities sharing the same facilities. Although each one acts individually as a single project represented as a classical PERT network, they cannot be analyzed independently since they share the same facilities. Therefore, developing a model under uncertainty and dynamic conditions would be beneficial to scheduling engineers in forecasting a more realistic project completion time.

As the coordination between the projects and departments is rather elaborate in dynamic PERT network, the organizations try to innovate an approach to overcome the challenging tasks of managing and controlling the multi-project environment. For this purpose, a process approach was introduced for dynamic PERT network using simulation by Adler, Mandelbaum, Nguyen, and Schwerer (1995). They envisioned an organization as a stochastic processing network that consists of a collection of service stations (work stations) or

Q3 * Corresponding author. at: Department of Industrial Engineering, Istanbul Sehir University, Istanbul, Turkey. Tel.: +90 216 559 9883.
E-mail address: amirazaron@sehir.edu.tr, aazaron@gmail.com (A. Azaron).

http://dx.doi.org/10.1016/j.ejor.2015.06.038
0377-2217/

resources, where all projects are considered as one system in that the resources are shared among them. In each node of such a network, one or more identical servers have been dedicated to serve in parallel under a pre-specified discipline. Therefore, the organization's behavior can be modeled as a queuing network, in that each activity is served by a resource, queuing up to reach that resource (in a resource queue), or waiting to join a predecessor activity that is being processed or delayed elsewhere (in a synchronization queue).

Subsequently, the concept of CONWIP (CONstant Work-In-Process) in dynamic PERT networks was studied using simulation by Anavi-Isakow and Golany (2003) who introduced two control mechanisms, in which one mechanism restricts the number of projects, CONPIP (COnstant Number of Projects In Process), and the other puts a limit on the total processing time by all active projects, CONTIP (CONstant Time of projects In Process).

Cohen, Golany, and Shtub (2005, 2007) also studied the resource allocation problem in dynamic PERT network, where it was assumed that the resources can work in parallel, namely, the number of resources allocated to the servers are equal (e.g., mechanical work stations with mechanics, electrical work stations with electricians, etc.). They investigated CONPIP systems using Cross Entropy (CE), based on simulation, and obtained near-optimal resource allocations to the entities that perform the projects.

On the other hand, Azaron and Tavakkoli-Moghaddam (2007) presented a multi-objective model for the resource allocation problem in a dynamic PERT network, where new projects are generated according to a Poisson process and the activity durations are exponentially distributed random variables. They assumed that the capacity of system is infinite, the number of servers in every service station is either one or infinity, the discipline of queues is First Come First Served (FCFS) and the allocated resources affect the mean activity durations. In this regard, Azaron, Katagiri, Sakawa, Kato, and Memariani (2006) and Azaron, Katagiri, and Sakawa (2007) also developed some multi-objective models for the time-cost trade-off problem in classical PERT networks with different assumptions on the distributions of activity durations (exponential in one and generalized Erlang in the other paper) and also different solution techniques (goal programming and goal attainment in one and the interactive SWT technique in the other one). The main difference between this paper and the previous two papers is that here we assume that the similar projects are generated according to a Poisson process over the time horizon which also share the same facilities, but the two previous research works were based on the fact that we have only a one-time job consisting of several activities, as the classical definition of project indicates. Moreover, Azaron, Fynes, and Modarres (2011) proposed an algorithm to obtain optimal constant lead time for each particular project in repetitive (dynamic) PERT networks by minimizing the average aggregate cost per each project. A risk element was also considered in dynamic PERT networks by Li and Wang (2009), who presented a multi-objective model for risk time-cost trade-off problem.

Recently, Yaghoubi, Noori, Azaron, and Tavakkoli-Moghaddam (2011) modeled the resource allocation problem in dynamic PERT networks, where the capacity of system is finite and only one type (class) of projects is generated according to a Poisson process. But in practice, most organizations perform different projects in nature, because of the various projects' requirements, whereas new projects from different classes arrive at system dynamically over the time horizon and are served stochastically. On the other hand, it is not possible to execute too many projects concurrently, because of limited resources. Therefore, the organizations are faced with multi-class dynamic PERT networks with finite capacity problem, as studied in this paper. The introduction of this problem may have a number of practical advantages such as the better utilization of limited resources, positive effects on productivity and easier monitoring of projects from different classes, to multi-project systems especially in engineering environments. In this study, the following assumptions are made:

- The capacity of system is finite.
- Different classes of projects exist in the system.
- New projects from different classes, including all their activities, arrive at system according to independent Poisson processes with different rates.
- Each project's end result leaves the system in its finished form from the sink node of the queueing network.
- Each service station consists of one server and can serve different activities.
- Each activity of any project from every class is performed at a devoted service station.
- Service discipline at each service station is based on FCFS.
- The activity duration at each service station is independent of preceding and succeeding activity durations.
- The activity durations for different classes at each service station are independent and exponentially distributed random variables.
- The queueing network is in the steady-state.
- Mean project completion time and project operating costs are controlled through the resources allocated to service stations.
- The mean times spent in each service station for different classes and the operating cost of the service station, respectively, are non-increasing and non-decreasing functions of the amount of resources allocated to that service station.

For modeling the multi-class dynamic PERT networks with CONPIP, we first consider every class separately and then convert the queueing network of every class into an appropriate stochastic network. By constructing a proper finite-state continuous-time Markov model, a system of differential equations is created to compute the project completion time distribution for any particular project.

In practice, activity duration is considered either as a function of cost or as a function of resources committed to it. In the time-cost trade-off problem (TCTP), which is one of the most important topics in project management, the objective is to determine the duration of each activity in order to achieve the minimum total costs of the project. The TCTP has been investigated using various kinds of cost functions such as linear (Fulkerson, 1961; Kelly, 1961), discrete (Demeulemeester, Herroelen, & Elmaghraby, 1993), convex (Berman, 1964; Lamberson & Hocking, 1970), concave (Falk & Horowitz, 1972) and so on.

As noted before, in this paper, we assume that the mean times spent in each service station for different classes and the operating cost of the service station, respectively, are some decreasing and increasing functions of the resources allocated to that particular service station. According to this, we develop a multi-objective model to optimally control the allocated resources in a way that the total operating costs of the service stations per period and also the mean project completion time over all classes in the steady-state to be minimized. On the other hand, having too many idle servers is not desirable. Therefore, the probability that the system becomes empty in the steady-state is considered as the third objective function, which should be minimized as well. This objective function is equivalent to maximizing the utilization factor of the system, because the utilization factor is the probability that the system is busy in the steady-state. The aim of this paper is to obtain a compromise solution for the resource allocation problem, using the goal attainment technique.

Since the resulting mathematical model is continuous-time, it is too complicated to be solved optimally. Therefore, we develop a simulated annealing algorithm to solve it and then compare the results against the results of a discrete-time approximation of the original optimal control problem to show the effectiveness of the proposed metaheuristic approach, which is another contribution of the paper.

The remainder of this paper is organized as follows. First, we present the literature review in the next section. Then, in Section 3, we model the multi-class dynamic PERT networks with finite capacity of concurrent projects as finite-state continuous-time Markov

Please cite this article as: S. Yaghoubi et al., Resource allocation in multi-class dynamic PERT networks with finite capacity, European Journal of Operational Research (2015), http://dx.doi.org/10.1016/j.ejor.2015.06.038

processes. We then develop a multi-objective model to optimally control the resources allocated to the servers in Section 3. In Section 4, we propose a simulated annealing algorithm for solving the multi-objective problem. We then solve 10 numerical examples in Section 5, and finally draw the conclusion in Section 6.

2. Literature review

In the literature of multi-project scheduling, *Multi-Project Resource Constrained Scheduling Problem (MPRCSP)* in static and deterministic environment has attracted significant attention from researchers. Wiest (1967) and Pritsker, Watters, and Wolfe (1969) pioneered the study of MPRCSP and proposed a zero-one programming approach and a heuristic model for analyzing this problem, respectively. Subsequently, Kurtulus and Davis (1982) and Kurtulus and Narula (1985) studied the MPRCSP approach by using the priority rules and explaining measures. In addition, some researchers have analyzed MPRCSP using multi-criteria and multi-objective approaches. For example, Chen (1994) described the application of zero-one goal programming for the maintenance of mineral processing, and Lova, Maroto, and Tormos (2000) developed a multi-criteria model in MPRCSP.

More recently, researchers such as Gonçalves, Mendes, and Resende (2008), Kumanan, Jegan, and Raja (2006), Lova and Tormos (2001), Tsubakitani and Deckro (1990), Ying, Shou, and Li (2009) and Chen and Shahandashti (2009) used heuristic and metaheuristic algorithms for solving MPRCSP. Kruger and Scholl (2010) generalized the MPRCSP by considering transfer times and their costs and Kanagasabapathi, Rajendran, and Ananthanarayanan (2009) proposed scheduling rules for this subject in a static condition by considering performance measures including the mean tardiness and the maximum tardiness of projects. In all above researches, it was assumed that the environment is static and deterministic.

In the literature, a number of studies have focused on MPRCSP under uncertainty. Fatemi Ghomi & Ashjari, 2002 developed a simulation model for multi-project resource allocation with stochastic task durations, modeling as a multi-channel queuing. A nonlinear mixed-integer programming model for optimizing the allocated resources was also proposed by Nozick, Turnquist, and XU (2004), and an event-driven approach was suggested by Kao, Hsieh, and Yeh (2006), whereby all projects are grouped as a dynamic network which can be moderated and rescheduled in reaction to important events. In addition, Critical Chain Project Management (CCPM) approach was used by Byali and Kannan (2008) to cope with the uncertainty in multi-project systems.

Note that MPRCSP is formulated by either considering the projects as independent projects and connecting them because of the resource constraints and using an objective function which includes all projects' performance measures (possibly properly weighted) or synthetically connecting them together into a large single project by adding dummy start and end activities.

It is worth mentioning that a key early research in Finite Capacity Queuing Network (FCQN) was done by Perros (1984). In the real world, FCQN is used in many areas such as manufacturing systems (Papadopoulos & Heavey, 1996; Tan & Gershwin, 2009), call centers (Jouini, Dallery, & Aksin, 2009), health care activities (Osorio & Bierlaire, 2009), software architecture sector (Balsamo, De Nitto Persone, & Inverardi, 2003), and production retrial queues for the telecommunication sector (Artalejo, 1999).

Over the last decade, a number of multi-objective evolutionary methods have been proposed by researchers (for more details see Deb (2001) and Coello, Veldhuizen, and Lamont (2002)). The main reason for the popularity of evolutionary methods for solving multi-objective optimization is their population-based nature and ability to obtain multiple optima simultaneously. Simulated annealing (SA) is a popular search method, proposed by Kirkpatrick, Gelatt, and Vecchi (1983), which employs the principles of statistical mechanics considering the behavior of a large number of atoms at low temperature, for obtaining minimal cost solutions to large-scale optimization problems by minimizing the associated energy.

SA is a robust and compact method, which provides excellent solutions for single and multiple objective optimization problems in relatively short computational times. It is convenient to formulate and can handle both continuous and integer variables with ease. Moreover, it is efficient and has low memory requirement. Note that SA takes generally less computational times than genetic algorithm (GA) to solve optimization problems, because it obtains the optimal solution using point-by-point iteration rather than a search over a population of individuals (Suman & Kumar, 2006). Geman and Geman (1984) proved that SA, if annealed sufficiently slowly, converges to the global optimum. Maffioli (1987) revealed that SA can be regarded as one type of randomized heuristic approaches for combinatorial optimization problems.

SA was started as a method or tool for solving single objective combinatorial problems and then it has been adapted for the multi-objective framework. Researchers such as Serafini (1985), Van Laarhoven and Aarts (1987), Ulungu and Teghem (1994), Ulungu, Teghem, and Ost (1998), Tuyttens, Teghem, Fortemps, and Nieuwenhuyze (2000), Suppapitnarm, Seffen, Parks, and Clarkson (2000), Suman (2002, 2004) and Bandyopadhyay, Saha, Maulik, and Deb (2008) have proposed SA based approaches to tackle multi-objective problems. Furthermore, Suman and Kumar (2006) have presented a good review of several multi-objective SA algorithms and their comparative performance analysis.

3. Multi-class dynamic PERT networks with CONPIP

3.1. Notations

M	Number of different classes of projects
λ_j	Arrival rate of projects from class $j\ (= 1, \ldots, M)$
λ	Summation of λ_js
λ'	Actual arrival rate of new projects to system
G_j	Directed stochastic network (AoA network) of class j
V_j	Set of nodes of G_j
A_j	Set of arcs of G_j
A	Union of A_js
μ_a^j	Service rate of activity $a(\in A_j)$ from class $f_3 = 0.014$
s_j	Source node of G_j
t_j	Sink node of G_j
$\alpha_j(a)$	Starting nodes of arc a in G_j
$\beta_j(a)$	Ending nodes of arc a in G_j
$I_j(v)$	Set of arcs ending at node v in G_j
$O_j(v)$	Set of arcs starting at node v in G_j
Y	Set of service stations
$s(a)$	Devoted service station to perform activity a
$(X, \bar{X})^j$	Set of arcs of G_j in which the starting node of each arc belongs to X and the ending node of that arc belongs to $\bar{X}$
$X_i^{m_i}(t)$	$= (Y_i(t), Z_i(t), Q_i(t))^{m_i}$, state of project i from class m_i at time t, where $Y_i(t), Z_i(t)$ and $Q_i(t)$ $Q_i(t)$ are active, dormant and in queue activities, respectively
$X(t)$	State of the system at time $[(1^*, 3)^3, (1, 2)^1]$
$(E_i, F_i, Q_i)^{m_i}$	Admissible 3-partition cut of project i from class m_i or (ϕ, ϕ, ϕ), where E_i, F_i and Q_i are active, dormant and in queue activities, respectively
$[E, F, Q]$	Admissible 3-partition cut of the system
N	Capacity of the system
G	Infinitesimal generator matrix
S	Set of system's states
S^i	Subset of S in that the system has $i\ (= 0, 1, \ldots, N)$ projects in processing
K	Number of system's states
$P_i(t)$	Probability of being the system in state $i\ (= 1, 2, \ldots, K)$ at time t, if the system be in sate 1 at time zero
$P(t)$	State vector

(*continued on next page*)

x_a	Amount of resources allocated to service station a
x	Matrix of resources allocated to all service stations
$d_a(x_a)$	Direct cost of service station a per period
$g_a^j(x_a)$	Mean service time in the service station a for the activities of class j
U_a	Maximum available resource to be allocated to the service station a
L_a	Minimum available resource to be allocated to the service station a
J	Amount of resource available to be allocated to all service stations
T	Mean project completion time in the steady-state
P	Average number of projects in the system in the steady-state
ε	A very small quantity approaching zero
T'	Time interval
b_j	Goal of jth objective in goal attainment method
c_j	Weight of jth objective in goal attainment method
u	A random number
θ_k	Temperature of kth times the temperature has been lowered in SA algorithm, while θ_0 and θ_f are the initial and final temperatures, respectively
τ	Decrement factor in SA algorithm
H	Number of iterations in each temperature of SA algorithm
$L(x)$	Cost function of SA algorithm
γ_i	Penalty coefficient

3.2. Mathematical model

In this section, we model the multi-class dynamic PERT networks with finite capacity of concurrent projects (COnstant Number of Projects In Process (CONPIP)) as proper queueing networks. The mathematical model which is developed in this section is the extended version of the one developedby Yaghoubi et al. (2011). For modeling, we first extend the method of Kulkarni and Adlakha (1986), because this method is an analytical one, simple, easy to implement on a computer and computationally stable. In multi-class dynamic PERT networks, the projects from different classes may be different in their precedence networks and also the durations of the activities. A project of class j $(=1,\dots,M)$ is represented as an Activity-on-Node (AoN) graph, where M is the number of different classes of projects. The jth type of projects, including all its activities, arrives according to a Poisson process with the rate of λ_j $(j=1,\dots,M)$, while each activity of the project is performed in a devoted service station settled in a node of the network. Activity a is operated in a devoted service station $s(a)=y$ $(y=1,2,\dots,Y)$, where Y denotes the set of service stations, which means each service station can serve more than one typical activity.

This system can be represented as a network of queues, where the service times represent the durations of the corresponding activities in every class. In each service station, there is only one server, while the service times (activity durations) for different classes of projects are assumed to be exponential with different service rates and the discipline of queue is first come to system, first served.

For modeling the multi-class dynamic PERT networks with CONPIP, we first consider every class separately and then transform the dynamic PERT network of each class, represented as an Activity-on-Node (AoN) graph, to a classic PERT network represented as an Activity-on-Arc (AoA) graph. For this purpose, node a in the AoN graph of class j $(=1,\dots,M)$ is replaced with a stochastic activity. Assume that $b_1,b_2,\dots,b_n$ are the incoming arcs to node a and $d_1,d_2,\dots,d_m$ are the outgoing arcs from it in the AoA graph of class j. Node a is substituted with activity (v,w), whose length is equal to the duration of activity a. Furthermore, all arcs $b_1,b_2,\dots,b_n$ terminate to node v, while all arcs $d_1,d_2,\dots,d_m$ originate from node w (see Azaron and Modarres (2005), for more details).

Let $G_j=(V_j,A_j)$ be a directed stochastic network of class j, in which V_j represents the set of nodes and A_j represents the set of arcs of the AoA network in class j. Note that the words activity and arc will be applied equivalent throughout the article. Let s_j and t_j be the source and sink nodes in the AoA graph of class j, respectively. Length of arc $a\in A_j$ is an exponentially distributed random variable with parameter μ_a^j. The starting and ending nodes of arc a in the AoA network of class j are denoted by $\alpha_j(a)$ and $\beta_j(a)$, respectively.

Definition 1. Let $I_j(v)$ and $O_j(v)$ be the sets of arcs ending and starting at node v in class j, respectively, which are defined as follows:

$$I_j(v)=\{a\in A_j:\beta_j(a)=v\}\quad (v\in V_j), \tag{1}$$

$$O_j(v)=\{a\in A_j:\alpha_j(a)=v\}\quad (v\in V_j). \tag{2}$$

Definition 2. For $X\subset V_j$ such that $s\in X$ and $t_j\in\bar{X}=V_j-X$, a cut of $G_j=(V_j,A_j)$ is defined as follows:

$$(X,\bar{X})^j=\{a\in A_j:\alpha_j(a)\in X,\beta_j(a)\in\bar{X}\}. \tag{3}$$

It is denominated a uniformly directed cut (UDC) of class j, if $(\bar{X},X)=\phi$, i.e. there are no two arcs in the cut belonging to the same path in the project network of class j.

Definition 3. An$(E,F,Q)^j$ $(j=1,\dots,M)$ subset of A_j is defined as an admissible 3-partition of a UDC D if $D=E\cup F\cup Q$ and $E\cap F=E\cap Q=F\cap Q=\phi$, and also $I_j(\beta_j(a))\not\subset F$ for any $a\in F$.

Definition 4. At time t, each activity of class j can be in one and only one of the active, dormant, in queue or idle states, which are defined as follows:

(i) *Active:* an activity a $(a\in A_j)$ is active at time t, if it is being performed at time t.
(ii) *Dormant:* an activity a $(a\in A_j)$ is called dormant at time t, if it has completed but there is at least one unfinished activity in $I_j(\beta_j(a))$ at time t.
(iii) *In queue*: activity a $(a\in A_j)$ is in queue at time t, if all predecessor activities of activity a are completed, but service station $\forall y\in Y$ is serving another project.
(iv) *Idle:* an activity a $(a\in A_j)$ is denominated idle at time t, if it is neither active nor dormant and nor in queue at time t.

Definition 5. The state of project i from class m_i $(m_i\in\{1,\dots,M\})$ at time t is $X_i^{m_i}(t)=(Y_i(t),Z_i(t),Q_i(t))^{m_i}$, where $Y_i(t),Z_i(t)$ and $Q_i(t)$ are denoted as follows:

$Y_i(t)=$ set of active activities in project i from class m_i at time t
$Z_i(t)=$ set of dormant activities in project i from class m_i at time t
$Q_i(t)=$ set of in queue activities in project i from class m_i at time t

If $L(x)$ represents the capacity of the system, then the state of the system at time $g_a^j(x_a)=\frac{1}{\mu_a^j}$ is given by

$$X(t)=\left[(Y_1(t),Z_1(t),Q_1(t))^{m_1},(Y_2(t),Z_2(t),Q_2(t))^{m_2},\dots,(Y_N(t),Z_N(t),Q_N(t))^{m_N}\right] \tag{4}$$

The admissible 3-partition cut of the system is also denoted by:

$$[E,F,Q]\overset{define}{=}[(E_1,F_1,Q_1)^{m_1},(E_2,F_2,Q_2)^{m_2},\dots,(E_N,F_N,Q_N)^{m_N}] \tag{5}$$

where $(E_i,F_i,Q_i)^{m_i}$ can be any admissible 3-partition cut of class m_i for the ith project or (ϕ,ϕ,ϕ) and E_i,F_i and Q_i include active, dormant and in queue activities of a UDC of project i from class m_i, respectively.

When activity a in project i from class m_i is completed with the rate of $\mu_a^{m_i}$ by service station x_4 and there is at least one uncompleted activity in $I_{m_i}(\beta_{m_i}(a))$, it moves from E_i to a new dormant activities set, F_i', and service station y serves another activity waiting in queue which has arrived at system earlier than the other in queue projects. But if the succeeding activities to a, $O_{m_i}(\beta_{m_i}(a))$, become active or in queue, by completing activity a with the rate of $\mu_a^{m_i}$, then it will be deleted from the active activities set and some elements of $O_{m_i}(\beta_{m_i}(a))$ may be added to E_i' and the others will be added to in queue activities set, Q_i'. The service station a also serves another project in queue. On the other hand, if the system has a finite capacity for accepting new projects from class j $(=1,\ldots,M)$ with the rate of λ_j $(j=1,\ldots,M)$, then some elements of $O_j(s_j)$ may be added to the active activities set of the new project, while the others will be added to its in queue activities set.

Thus, the component of the infinitesimal generator matrix of this process, denoted by $G=[g\{(E,F,Q)^m,(E',F',Q')^{m'}\}]$, is calculated as follows:

(i) Transition 1:

If $a\in E_i, I_{m_i}(\beta_{m_i}(a))\not\subset F_i\cup\{a\}$ then
Begin:
$F_i'=F_i\cup\{a\}$,
$L=\phi$,
For $k=1$ to N do
Begin:
For $\forall b\in Q_k$ do if $s(b)=s(a)$ then $L=L\cup\{b\}$,
If $L\neq\phi$ $(|L|\geq 1)$ then
Begin:
Randomly select a member from $L=c$,
If $k\neq i$ then
$E_k'=E_k\cup\{c\}$, $Q_k'=Q_k-\{c\}$, $E_i'=E_i-\{a\}$ and Go to End,
(Transition Rate : $\frac{1}{|L|}\mu_a^{m_i}$)
Else $E_i'=(E_i-\{a\})\cup\{c\}$, and Go to End,
(Transition Rate : $\frac{1}{|L|}\mu_a^{m_i}$),
End,
End,
If $L=\phi$ then $E_i'=E_i-\{a\}$,
End,

(ii) Transition 2:

If $a\in E_i, I_{m_i}(\beta_{m_i}(a))\subset F_i\cup\{a\}$ then
Begin:
$L=\phi$,
For $\forall b\in O_{m_i}(\beta_{m_i}(a))$ do
Begin:
For $k=1$ $(k\neq i)$ to N do
For $\forall c\in E_k$ do if $s(b)=s(c)$ then $L=L\cup\{b\}$,
For $\forall c\in E_i-\{a\}$ do if $s(b)=s(c)$ then $L=L\cup\{b\}$,
End,
$Q_i'=Q_i\cup L$,
$F_i'=F_i-I_{m_i}(\beta_{m_i}(a))$,
$W=\phi$,
For $k=1$ to N do
Begin:
If $k\neq i$ then
Begin:
For $\forall b\in Q_k$ do if $s(b)=s(a)$ then $W=W\cup\{b\}$,
If $W\neq\phi$ $(|W|\geq 1)$ then
Begin:
Randomly select a member from $W=c$,
$E_k'=E_k\cup\{c\}$, $Q_k'=Q_k-\{c\}$ and Go to End
(Transition Rate : $\frac{1}{|W|}\mu_a^{m_i}$),
End,
Else $(k=i)$ Begin:
$W_y=\phi$, $y\in Y$
For $\forall y\in Y$ do
For $\forall b\in O_{m_i}(\beta_{m_i}(a))-L$ do if $s(b)=y$ then $W_y=W_y\cup\{b\}$,
For $\forall W_y\neq\phi$ do randomly select a member from $W_y=w_y$,
$E_i'=\bigcup_{\forall W_y\neq\phi}\{w_y\}$, (Transition Rate : $\frac{1}{\prod_{\forall W_y\neq\phi}|w_y|}\mu_a^{m_i}$),
End,
End,
End,
End,

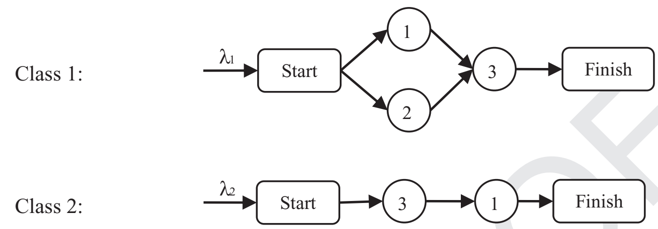

Fig. 1. Example 1.

(iii) Transition 3:

If $E_i=Q_i=F_i=\phi$ and $(E_{i-1}\neq\phi$ or $Q_{i-1}\neq\phi$ or $F_{i-1}\neq\phi)$ then
Begin:
$L=\phi$,
For $\forall b\in O_j(s_j)$ do
For $k=1$ to $i-1$ do
For $\forall c\in E_k$ do if $s(b)=s(c)$ then $L=L\cup\{b\}$,
$Q_i'=L$,
If $O_j(s_j)=L$ then Go to End (Transition Rate : λ_j),
$W_y=\phi$, $y\in Y$
For $\forall y\in Y$ do
For $\forall b\in O_j(s_j)-L$ do if $s(b)=y$ then $W_y=W_y\cup\{b\}$,
For $\forall W_y\neq\phi$ do randomly select a member from $W_y=w_y$,
$E_i'=\bigcup_{\forall W_y\neq\phi}\{w_y\}$, (Transition Rate : $\frac{1}{\prod_{\forall W_y\neq\phi}|w_y|}\lambda_j$),
End,

Example 1. It is assumed that two types (classes) of projects exist and the system can perform up to two projects, depicted as the AoN graph in Fig. 1. The new projects are generated according to two independent Poisson processes. Moreover, it is assumed that we just have two service stations, while the first service station serves activities 1 and 2 and the second service station serves activity 3, namely, $s(1)=1, s(2)=1, s(3)=2$.

In Table 1, all admissible 3-partition cuts of the transformed AoA network of Fig. 1 are presented, whereas we use superscript asterisk and q to denote "dormant" and "in queue" activities, respectively. For example, consider state 16, namely, $[(1^*,2)^1,(1^q,2^q)^1]$. In this state, activity 2 of the first project, which is a class 1 type, is active, while activity 1 of the first project is dormant, because activity 2 of this project is not completed yet, and activities 1 and 2 of the second project, which is also a class 1 type project, are in queue and the reason is that the service station 1 is serving activity 2 of an earlier project. In state 16, once activity 2 of the first project, which is again a class 1 type, is completed, the system can either go to state 26 or state 27 with the same transition rates of $\frac{1}{2}\mu_2^1$, because when activity 2 of the first project is completed by service station 1, this service station becomes free to process activities 1 or 2 of the second project. But, since the service station has only one server, it can only serve one activity. Therefore, after completing activity 2 of the first project, the service station 1 has two choices. One choice is to process activity 1 of the second project, while the other choice is to process activity 2 of the second project. Here, a randomized strategy is used and the system's new state will either be $[(3)^1,(1,2^q)^1]$ or $[(3)^1,(1^q,2)^1]$.

Please cite this article as: S. Yaghoubi et al., Resource allocation in multi-class dynamic PERT networks with finite capacity, European Journal of Operational Research (2015), http://dx.doi.org/10.1016/j.ejor.2015.06.038

Table 1
All admissible 3-partition cuts of Example 1.

1. $[\phi, \phi]$	9. $[(1^q, 2)^1, (1^q, 2^q)^1]$	17. $[(1^*, 2)^1, (3)^2]$	25. $[(3)^2, (1, 2^*)^1]$	33. $[(1^q)^2, (1^*, 2)^1]$
2. $[(1, 2^q)^1, \phi]$	10. $[(1^q, 2)^1, (3)^2]$	18. $[(1, 2^q)^1, (1^q)^2]$	26. $[(3)^1, (1, 2^q)^1]$	34. $[(3)^2, (3^q)^1]$
3. $[(1^q, 2)^1, \phi]$	11. $[(1)^2, \phi]$	19. $[(1, 2^*)^1, (1^q, 2^q)^1]$	27. $[(3)^1, (1^q, 2)^1]$	35. $[(1^q)^2, (1, 2^*)^1]$
4. $[(3)^2, \phi]$	12. $[(3)^2, (1, 2^q)^1]$	20. $[(1, 2^*)^1, (3)^2]$	28. $[(3)^1, (3^q)^2]$	36. $[(3)^1, (1^*, 2)^1]$
5. $[(1^*, 2)^1, \phi]$	13. $[(3)^2, (1^q, 2)^1]$	21. $[(1^q, 2)^1, (1^q)^2]$	29. $[(3^q)^1, (3)^2]$	37. $[(3)^1, (1, 2^*)^1]$
6. $[(1, 2^q)^1, (1^q, 2^q)^1]$	14. $[(3)^2, (3^q)^2]$	22. $[(1)^2, (1^q, 2^q)^1]$	30. $[(1^*, 2)^1, (1^q)^2]$	38. $[(3)^1, (1)^2]$
7. $[(1, 2^q)^1, (3)^2]$	15. $[(3)^1, \phi]$	23. $[(1)^2, (3)^2]$	31. $[(1, 2^*)^1, (1^q)^2]$	39. $[(1)^2, (3)^1]$
8. $[(1, 2^*)^1, \phi]$	16. $[(1^*, 2)^1, (1^q, 2^q)^1]$	24. $[(3)^2, (1^*, 2)^1]$	32. $[(1)^2, (1^q)^2]$	40. $[(3)^1, (3^q)^1]$

We are now dealing with a finite-state (number of states is equal to K) continuous-time Markov process. If we define

$$P_i(t) = P(X(t) = i \big| X(0) = 1) \quad i = 1, 2, \ldots, K. \tag{6}$$

Then, the system of differential equations for the vector $P(t) = [P_1(t) \quad P_2(t) \quad \ldots. \quad P_K(t)]$ concerning the dynamic model will be given by (see Yaghoubi et al. (2011) for details)

$$P'(t) = \frac{dP(t)}{dt} = P(t) \cdot G$$
$$P(0) = [1 \quad 0 \quad \ldots. \quad 0] \tag{7}$$

3.3. Resource allocation problem

In this section, we develop a multi-objective model to optimally control the resources allocated to the servers in a multi-class dynamic PERT network with finite capacities, whereas such a system is represented as a queueing network. Moreover, the mean times spent in each service station for different types of projects are decreased and the operating cost of the service station is increased when we allocate more resources to that particular service station. That means the mean times spent in each service station and the operating cost of it, respectively, are non-increasing and non-decreasing functions of the amount of resources allocated to that service station.

3.3.1. Multi-objective resource allocation problem

In our model, the total operating costs of service stations per period is considered as the first objective and the mean project completion time over all types of projects in the steady-state as the second objective, both to be minimized. The third objective is to minimize the probability that the system becomes empty in the steady-state. This objective is equivalent to maximizing the utilization factor of the system, because the utilization factor is the probability that the system is busy. The first and second objectives are in conflict with each other, because if the total operating costs of service stations per period is decreased, then the mean times spent in service stations and consequently the mean project completion time will be increased.

Let x_a be the amount of resources allocated to service station a, and $d_a(x_a)$be the operating cost of service station a per period. It is assumed that $d_a(x_a)$ is a non-decreasing function of x_a. Therefore, the total operating costs per period (TDC) will be equal to $\mathrm{TDC} = \sum_{a \in A} d_a(x_a)$ which should be minimized, whereas $A = \bigcup_{j=1}^{M} A_j$.

In addition, let S^i be the subset of S, the set of system states in that the system has i projects in processing, i.e. the system has $N - i$ capacity for accepting new projects. Also, let λ be the summation of arrival rates of all classes, namely, $\lambda = \sum_{j=1}^{M} \lambda_j$. Let P and T be the average number of projects in the system and the mean project completion time, respectively, in the steady-state. Therefore, according to Little's theorem, we have $P = \lambda' T$, where λ', the actual arrival rate, is equal to $\lim_{t \to \infty} \lambda \sum_{i \in S - S^N} P_i(t)$, and P is given by

$$P = \lim_{t \to \infty} \sum_{k=1}^{N} \sum_{i \in S^k} k P_i(t) \tag{8}$$

Table 2
Classification of states in Example 1.

Number of projects	State	S^i
0	1	S^0
1	2,3,4,5,8,11,15	S^1
2	6,7,9,10,12,13,14,16,…,40	S^2

Consequently, the second objective function to be minimized will be

$$T = \lim_{t \to \infty} \frac{\sum_{k=1}^{N} \sum_{i \in S^k} k P_i(t)}{\lambda \sum_{i \in S - S^N} P_i(t)} \tag{9}$$

Finally, the third objective function to be minimized will be equal to $\lim_{t \to \infty} P_1(t)$.

To illustrate the second objective function, consider Example 1 again. In Table 2, the states of Example 1 are classified based on the number of projects in the system.

Thus, the second objective function for this example is $\mathrm{Min} \lim_{t \to \infty} \frac{\left[\sum_{i=2}^{5} P_i(t) + P_8(t) + P_{11}(t) + P_{15}(t)\right] + 2\left[P_6(t) + P_7(t) + P_9(t) + P_{10}(t) + \sum_{i=12}^{14} P_i(t) + \sum_{i=16}^{40} P_i(t)\right]}{\lambda \cdot \left[\sum_{i=1}^{5} P_i(t) + P_8(t) + P_{11}(t) + P_{15}(t)\right]}$

Moreover, the mean service time in the service station a for the activities of class $j(=1, \ldots, M)$ is a non-increasing function $g_a^j(x_a)$ of the amount of resource x_a allocated to it. Thus, the mean service time in the service station a for the activities of class j will be equal to $\frac{1}{\mu_a^j}(= g_a^j(x_a))$. Let U_a and L_a denote the maximum and minimum available resource to be allocated to the service station a, respectively,$x = [x_a : a \in A]^T$ and J represents the amount of resources available to be allocated to all service stations. In practice, $d_a(x_a)$ and $g_a^j(x_a)$ $(j = 1, \ldots, M)$ can be obtained using linear regression by referring to the similar activities, which have been performed in the past, or the judgments of experts in this area.

Finally, we are going to have a multi-objective stochastic programming problem in that the objective functions are given by

1. Minimizing the project's operating costs per period

$$\mathrm{Min}\ f_1(x) = \sum_{a \in A} d_a(x_a) \tag{10}$$

2. Minimizing the mean project completion time over all classes in the steady-state

$$\mathrm{Min}\ f_2(x) = \lim_{t \to \infty} \frac{\sum_{k=1}^{N} \sum_{i \in S^k} k P_i(t)}{\lambda \sum_{i \in S - S^N} P_i(t)} \tag{11}$$

3. Minimizing the probability that the system becomes empty in the steady-state

$$\mathrm{Min}\ f_3(x) = \lim_{t \to \infty} P_1(t) \tag{12}$$

Please cite this article as: S. Yaghoubi et al., Resource allocation in multi-class dynamic PERT networks with finite capacity, European Journal of Operational Research (2015), http://dx.doi.org/10.1016/j.ejor.2015.06.038

The infinitesimal generator matrix G would be a function of the control vector $x = [x_a : a \in A]^T$. Therefore, the non-linear dynamic model will be

$$P'(t) = P(t) \cdot G(\mu) \tag{13}$$

$$\begin{aligned} &P_i(0) = 0 \quad \forall i = 2, \ldots, K \\ &P_1(0) = 1 \end{aligned} \tag{14}$$

The following constraint should also be considered to guaranty having a response in the steady-state:

$$\frac{\sum_{a\in A_j} \lambda_j \cdot \lim_{t\to\infty} \sum_{i\in S-S^N} P_i(t)}{\frac{\sum_{a\in A_j} (\lambda_j \cdot \mu_a^j)}{\sum_{a\in A_j} \lambda_j}} \prec 1 \Rightarrow \sum_{a\in A_j} (\lambda_j \cdot \mu_a^j) - \left(\sum_{a\in A_j} \lambda_j\right)^2 \cdot \lim_{t\to\infty} \sum_{i\in S-S^N} P_i(t) \succ 0 \quad \forall a \in A \tag{15}$$

In standard mathematical programming problems, we do not have such constraints. Hence, we are going to replace it with

$$\sum_{a\in A_j} (\lambda_j \cdot \mu_a^j) - \left(\sum_{a\in A_j} \lambda_j\right)^2 \cdot \lim_{t\to\infty} \sum_{i\in S-S^N} P_i(t) \geq \varepsilon \quad \forall a \in A \tag{16}$$

Consequently, the proper multi-objective optimal control problem will be

$$\begin{aligned} &\text{Min } f_1(x) = \sum_{a\in A} d_a(x_a) \\ &\text{Min } f_2(x) = \lim_{t\to\infty} \frac{\sum_{k=1}^{N} \sum_{i\in S^k} kP_i(t)}{\lambda \sum_{i\in S-S^N} P_i(t)} \\ &\text{Min } f_3(x) = \lim_{t\to\infty} P_1(t) \\ &s.t: \quad P'(t) = P(t) \cdot G(\mu) \\ &\qquad P_i(0) = 0 \quad \forall i = 2, \ldots, K \\ &\qquad P_1(0) = 1 \\ &\qquad P_i(t) \leq 1 \quad \forall i = 1, 2, \ldots, K \\ &\qquad g_a^j(x_a) = \frac{1}{\mu_a^j} \quad \forall a \in A_j, \ j = 1, \ldots, M \\ &\qquad \sum_{a\in A_j} (\lambda_j \cdot \mu_a^j) - \left(\sum_{a\in A_j} \lambda_j\right)^2 \cdot \lim_{t\to\infty} \sum_{i\in S-S^N} P_i(t) \geq \varepsilon \quad \forall a \in A \\ &\qquad x_a \geq L_a \quad \forall a \in A \\ &\qquad x_a \leq U_a \quad \forall a \in A \\ &\qquad \sum_{a\in A} x_a \leq J \end{aligned} \tag{17}$$

This continuous-time stochastic programming problem is impossible to solve optimally in this form using conventional optimal control tools such as Maximum Principle (see Azaron & Tavakkoli-Moghaddam (2007) for more details). Therefore, we try to solve it using simulated annealing approach, based on a goal attainment formulation. To show the effectiveness of the proposed metaheuristic approach, we also compare its results against the results of a discrete-time approximation of (17), whereas the differential equations are converted into difference equations. Let T' be the time interval, which we divide it into $R(= \frac{T'}{\Delta t})$ equal portions with the length of Δt. Consequently, the resulting discrete-time model will be

$$\begin{aligned} &\text{Min } \sum_{a\in A} d_a(x_a) \\ &\text{Min } \frac{\sum_{k=1}^{N} \sum_{i\in S^k} kP_i(R)}{\lambda \sum_{i\in S-S^N} P_i(R)} \\ &\text{Min } P_1(R) \\ &s.t: \quad P(r+1) = P(r) + P(r)G(\mu)\Delta t \quad r = 0, 1, 2, \ldots, R-1 \\ &\qquad P_i(0) = 0 \quad \forall i = 2, \ldots, K \\ &\qquad P_1(0) = 1 \\ &\qquad P_i(r) \leq 1 \quad i = 1, \ldots, K, \quad r = 1, 2, \ldots, R \\ &\qquad g_a^j(x_a) = \frac{1}{\mu_a^j} \quad \forall a \in A_j, \ j = 1, \ldots, M \\ &\qquad \sum_{a\in A_j} (\lambda_j \cdot \mu_a^j) - \left(\sum_{a\in A_j} \lambda_j\right)^2 \cdot \sum_{i\in S-S^N} P_i(R) \geq \varepsilon \quad \forall a \in A \\ &\qquad x_a \geq L_a \quad \forall a \in A \\ &\qquad x_a \leq U_a \quad \forall a \in A \\ &\qquad \sum_{a\in A} x_a \leq J \end{aligned} \tag{18}$$

3.3.2. Goal attainment method

We now need to use a multi-objective method to solve (17). We actually use goal attainment technique for this purpose, because it is simple and computationally faster. The goal attainment method needs to determinate a goal, b_j, and a weight, c_j, for each objective function. c_j represents the importance of the jth objective, whereas, if an objective has the smallest c_j, then it will be the most important objective. c_js $(j = 1, 2, 3)$ are normalized such that $\sum_{j=1}^{3} c_j = 1$. To determine b_j for the jth objective, we have to solve the corresponding single objective problem first and then to set the value of b_j something very close to the optimal single objective value. Goal attainment method is actually a variation of goal programming method intending to minimize the maximum weighted deviation from the goals.

The appropriate goal attainment formulation of the resource allocation problem is given by

$$\begin{aligned} &\text{Min } \ z \\ &s.t: \quad \sum_{a\in A} d_a(x_a) - c_1 z \leq b_1 \\ &\qquad \lim_{t\to\infty} \frac{\sum_{k=1}^{N} \sum_{i\in S^k} kP_i(t)}{\lambda \sum_{i\in S-S^N} P_i(t)} - c_2 z \leq b_2 \\ &\qquad \lim_{t\to\infty} P_1(t) - c_3 z \leq b_3 \\ &\qquad P'(t) = P(t) \cdot G(\mu) \\ &\qquad P_i(0) = 0 \quad \forall i = 2, \ldots, K \\ &\qquad P_1(0) = 1 \\ &\qquad P_i(t) \leq 1 \quad \forall i = 1, 2, \ldots, K \\ &\qquad g_a^j(x_a) = \frac{1}{\mu_a^j} \quad \forall a \in A_j, \ j = 1, \ldots, M \\ &\qquad \textstyle\sum_{a\in A_j} (\lambda_j \cdot \mu_a^j) - \left(\sum_{a\in A_j} \lambda_j\right)^2 \cdot \lim_{t\to\infty} \sum_{i\in S-S^N} P_i(t) \geq \varepsilon \quad \forall a \in A \\ &\qquad x_a \geq L_a \quad \forall a \in A \\ &\qquad x_a \leq U_a \quad \forall a \in A \\ &\qquad \sum_{a\in A} x_a \leq J \end{aligned} \tag{19}$$

Please cite this article as: S. Yaghoubi et al., Resource allocation in multi-class dynamic PERT networks with finite capacity, European Journal of Operational Research (2015), http://dx.doi.org/10.1016/j.ejor.2015.06.038

Since the goal attainment method has fewer variables to work with and is a one-stage method, unlike interactive multi-objective techniques, it will be computationally faster. Therefore, in terms of computational time, it is one of the best techniques to solve our complex multi-objective problem.

4. Simulated annealing algorithm

Metaheuristic methods such as simulated annealing, tabu search, genetic algorithm, artificial neural networks and their hybrids are applied in various fields, while these methods have been rarely used in multi-project scheduling (Chen & Shahandashti, 2009; Kumanan et al., 2006). In this section, we develop a simulated annealing (SA) algorithm to solve the multi-objective problem (19). Simulated annealing is known as a powerful optimization method and is based on the similarity between the solid annealing process and solving combinatorial optimization problems. The primary advantage of simulated annealing is to escape from local optima by allowing non-improver solutions according to a certain probability in each temperature.

As mentioned before, SA is a robust and compact method, which supplies excellent solutions to single and multiple objective optimization problems with a primary reduction in computational times. It is convenient to formulate and can handle continuous and mixed-integer problems with ease. Moreover, it is efficient and has low memory requirements. Geman and Geman (1984) proved that SA, if annealed sufficiently slowly, converges to the global optimum.

Simulated annealing consists of several decreasing temperatures, while each temperature has a few iterations. In the SA algorithm, the initial temperature is first chosen and a beginning solution is randomly selected. The cost (energy) function value of the SA algorithm will be calculated according to the current solution. Then, a new solution from the neighborhood of the current solution will be generated. The new cost function value will be obtained, according to the new solution, and then compared to the current cost function value. If the new cost function value is less than the current value, it will be accepted because of minimizing the cost function. Otherwise, if the difference between the cost function values of the current and the newly generated solutions, ΔL, is equal or larger than zero, it will be accepted only when Metropolis's criterion, which is based on Boltzman's probability, is met. For this purpose, a random number u is generated according to a uniform distribution. If $u \le e^{-(\Delta L/\theta)}$, then the newly generated solution is accepted as the current solution, where θ is the current temperature (see Kirkpatrick et al. (1983) for more details).

Simulated annealing needs a proper temperature schedule to warrant that the algorithm becomes convergent to a good solution. Therefore, we consider the classical rule updating of simulated annealing as $\theta_k = \tau \cdot \theta_{k-1}$, where θ_k is the temperature of kth times the temperature has been lowered and τ $(0.85 \le \tau \le 0.95)$ is the decrement factor or cooling ratio. Moreover, let θ_0 and θ_f be the initial and final temperatures, respectively. The solution's quality and the convergence's speed depend on τ, while $\tau = 0.95$ and $\tau = 0.85$ determine the slow and fast cooling, respectively.

SA algorithm is comprised of two loops (processes), namely inner loop and outer loop. The inner loop is iterated until the equilibrium condition is satisfied, while the outer loop conducts the annealing process and is iterated until the stoppage criterion is satisfied. In this paper, the number of H iterations is considered for each temperature as the equilibrium condition and reaching the final temperature θ_f is also determined as the stoppage criterion.

In order to have the simple form of (19) and to prepare it for the SA algorithm implementation, we reformulate it as follows:

$$\text{Min} \quad z = \text{Max}\left\{ \frac{\sum_{a\in A} d_a(x_a) - b_1}{c_1}, \frac{\lim_{t\to\infty} \frac{\sum_{k=1}^{N}\sum_{i\in S^k} kP_i(t)}{\lambda \sum_{i\in S-S^N} P_i(t)} - b_2}{c_2}, \frac{\lim_{t\to\infty} P_1(t) - b_3}{c_3} \right\}$$

$$\text{s.t}: \quad P'(t) = P(t)\cdot G(\mu), \quad P(0) = [1 \quad 0 \quad \ldots \quad 0] \quad (a)$$

$$g_a^j(x_a) = \frac{1}{\mu_a^j} \quad \forall a \in A_j,\ j = 1, \ldots, M \quad (b)$$

$$\sum_{a\in A_j} (\lambda_j \cdot \mu_a^j) - \left(\sum_{a\in A_j} \lambda_j\right)^2 \cdot \lim_{t\to\infty} \sum_{i\in S-S^N} P_i(t) \ge \varepsilon \quad \forall a \in A \quad (c)$$

$$\sum_{a\in A} x_a \le J \quad (d)$$

$$x_a \in [L_a, U_a] \quad \forall a \in A \tag{20}$$

To determine the solution representation schema in SA, we apply $x = [x_a : a \in A]^T$ as the amount of resources allocated to service stations, where $x_a \in [L_a, U_a] \quad \forall a \in A$. Generation of a new solution is also done through selecting one service station randomly in the current solution and allocating new acceptable resources to it randomly. To illustrate the proposed method for generating new solution, we consider $x = [x_1 \quad \ldots \quad x_a \quad \ldots \quad x_{|A|}]^T$ as the current solution, where $|A|$ is the number of elements of A. It is assumed that the service station a is randomly selected for generating new solution. Then, the new acceptable resources is randomly assigned to the service station a, namely $x_a \to x_a^{new}$ where $x_a^{new} \in [L_a, U_a]$. The related pseudo-code is shown as follows:

- consider $x = [x_1 \quad \ldots \quad x_a \quad \ldots \quad x_{|A|}]^T$ as the current solution, where $|A|$ is the number of elements of A;
- select one service station randomly in the current solution (e.g. service station a);
- allocate new acceptable resources to the selected service station, randomly (e.g. $x_a \to x_a^{new}$ where $x_a^{new} \in [L_a, U_a]$).

Furthermore, to determine the cost function of SA algorithm, we use Lagrange's function. Lagrange's function denotes the objective function plus the sum of penalty terms corresponding to the model's constraints. With regard to the objective function of (20), the cost function of SA algorithm, $L(x)$, is written as Eq. (21). This equation is the original objective function plus the penalty terms corresponding to the violation of constraints (c) and (d) in (20). Parameters γ_1 and γ_2 are penalty coefficients, which should be relatively large.

$$L(x) = \text{Max}\left\{ \frac{\sum_{a\in A} d_a(x_a) - b_1}{c_1}, \frac{\lim_{t\to\infty} \frac{\sum_{k=1}^{N}\sum_{i\in S^k} kP_i(t)}{\lambda \sum_{i\in S-S^N} P_i(t)} - b_2}{c_2}, \frac{\lim_{t\to\infty} P_1(t) - b_3}{c_3} \right\}$$
$$+ \gamma_1 \cdot \sum_{a\in A} \text{Max}\left\{ \varepsilon + \left(\sum_{a\in A_j} \lambda_j\right)^2 \cdot \lim_{t\to\infty} \sum_{i\in S-S^N} P_i(t) - \sum_{a\in A_j} (\lambda_j \cdot \mu_a^j), 0 \right\}$$
$$+ \gamma_2 \cdot \text{Max}\left\{ \sum_{a\in A} x_a - J, 0 \right\} \tag{21}$$

Consequently, the proposed SA algorithm to solve (20) will be as follows:

- Determine the values of the initial temperature, θ_0, the penalty coefficients, γ_1 and γ_2, the decrement factor, τ, and ε
- Generate one initial solution, $x = [x_a : a \in A]^T$, randomly, where $x_a \in [L_a, U_a]$

- Obtainμ_a^js ($\forall a \in A_j,\ j = 1, \ldots, M$) based on the constraint (b) in (20), $g_a^j(x_a) = \frac{1}{\mu_a^j}$, and construct matrix G for the initial solution
- Solve the system of differential equations based on constraint (a) in (20), and compute $P_i(T')$ $(i = 1, \ldots, K)$ $\left(\lim\limits_{t\to\infty} P_i(t)\right)$ for the initial solution, where T' is something large
- Calculate the cost (energy) function, $L(x)$, according to (21) for the initial solution
- Set the temperature change counter $k = 0$
- Repeat (Cooling loop)
- Set the repetition counter $h = 0$
- Repeat (Equilibrium loop)
- Generate a new solution from the neighborhood of the current solution, where $x_a^{new} \in [L_a, U_a]$
- Obtain μ_a^j's ($\forall a \in A_j,\ j = 1, \ldots, M$) based on constraint (b) in (20), $g_a^j(x_a) = \frac{1}{\mu_a^j}$, and construct matrix G for the new solution x^{new}
- Solve the system of differential equations based on the constraint (a) in (20), and compute $P_i(T')$ $(i = 1, \ldots, K)$ for the new solution x^{new}
- Calculate the cost (energy) function, $L(x^{new})$, according to (21) for the new solution
- Calculate $\Delta L = L(x^{new}) - L(x)$
- If $\Delta L \prec 0$ then $x = x^{new}$, $L(x) = L(x^{new})$
- else if $u \le e^{-(\Delta L/\theta_k)}$ then $x = x^{new}$, $L(x) = L(x^{new})$, where $u \in U[0, 1]$
- $h = h + 1$
- Until $h = H$
- $k = k + 1$
- $\theta_k = \tau \cdot \theta_{k-1}$
- Until the stoppage criterion holds true ($\theta_k \prec \theta_f$)

Note that determining the initial temperature is a crucial issue in SA, but there is no general method to set it. Based on the principal concepts of SA, non-improver solutions are accepted in the primary iterations with high probability. We determine the initial temperature in such a way that the non-improver solutions are accepted with a probability of almost 95% in the primary iterations. For this purpose, we select two solutions x^1 and x^2 at random and the initial temperature is therefore obtained as follows:

$$\theta_0 = \frac{-\left|L(x^1) - L(x^2)\right|}{\ln(0.95)} \tag{22}$$

5. Numerical examples

To demonstrate the performance of the proposed method, we consider 10 typical small and medium-sized cases with different configurations including four to twelve service stations and two or three types (classes) of projects, taken from Anavi-Isakow and Golany (2003), Cohen et al. (2007) and Yaghoubi (2012). In 6 out of the 10 cases, we consider two classes of projects, while the system has three classes of projects in 4 remaining sample cases. The cases and the structure of cost and expected value functions (different linear and non-linear forms) have been chosen so as to represent a wide variety of problems encountered in allocating resources in PERT networks. In all cases, each activity of the project is performed at the devoted service station with one server located in a node of the network, where the activity durations for different classes in each service station are independent and exponentially distributed random variables with different service rates. The capacity of system in all cases is two projects, and the value of ε set to 0.01 in all experiments. For simplicity, we assume $s(a) = a$ in all cases. As mentioned, it is possible to have service stations which can serve more than one typical activity, but we do not consider those cases whose state spaces have remarkably larger sizes, in our numerical experiments.

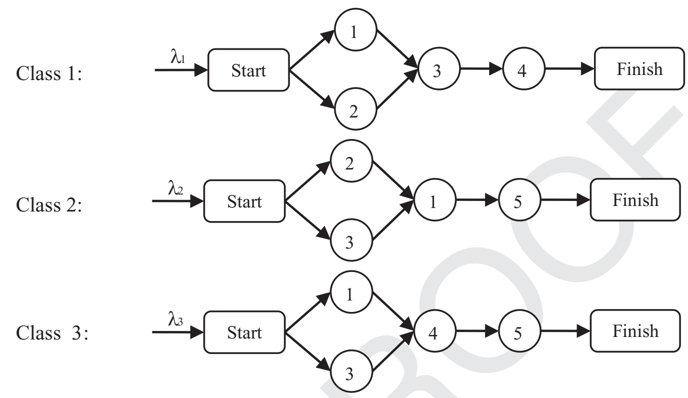

Fig. 2. Case I.

The objective is to obtain the optimal allocated resources using the SA and also the discrete-time approximation technique based on goal attainment method for different combinations of the parameters including goals and weights of the objective functions to reach Pareto-optimal solutions. Note that all SA experiments are replicated 10 times using different random initial solutions. Then, the SA results are compared against the results of the discrete-time approximation of the original optimal control problem in all 10 cases using the statistical inference. For this purpose, a paired sample Wilcoxon signed-rank test analysis is utilized to investigate whether solutions obtained by the SA algorithm differ from the discrete-time approximation or not.

To enhance the paper's readability, we only present the characteristics of three cases in details, whereas the results are presented based on all cases.

5.1. Case I

It is assumed that we have a system with the capacity of two concurrent projects, three different classes of projects and also five service stations, depicted as the AoN graph in Fig. 2, which was taken from Yaghoubi (2012). The new projects, including all their activities, are generated according to three independent Poisson processes with the rates of $\lambda_1 = 3$, $\lambda_2 = 1$ and $\lambda_3 = 1$ per year for the projects of class 1, class 2 and class 3, respectively. Moreover, the amount of resource available to be allocated to all service stations is 12.

Table 3 shows the characteristics of the activities, where the time unit and the cost unit are in year and in thousand dollars, respectively. After determining the system's states, depicted in Table 4, and transition rates, we obtain the infinitesimal generator matrix $G(\mu)$.

We also consider the goals, $b_1 = 30$, $b_2 = 1.6$, $b_3 = 0.05$ and the following sets of c for the three objectives, to generate a set of Pareto-optimal solutions, according to the goal attainment formulation (19):

Set 1: $c_1 = 0.3, c_2 = 0.1, c_3 = 0.6$,
Set 2: $c_1 = 0.2, c_2 = 0.1, c_3 = 0.7$,
Set 3: $c_1 = 0.2, c_2 = 0.2, c_3 = 0.6$,
Set 4: $c_1 = 0.1, c_2 = 0.2, c_3 = 0.7$.

5.2. Case II

The second case, which is depicted in Fig. 3, has been taken from Anavi-Isakow and Golany (2003). In Case II, it is assumed that two classes of projects exist and the system can perform up to two projects. The new projects, including all their activities, are generated according to two independent Poisson processes with the rates of $\lambda_1 = 3$ and $\lambda_2 = 1$ per year for the projects of class 1 and class 2, respectively. In addition, the amount of resource available to be allocated to all service stations is 15.

Please cite this article as: S. Yaghoubi et al., Resource allocation in multi-class dynamic PERT networks with finite capacity, European Journal of Operational Research (2015), http://dx.doi.org/10.1016/j.ejor.2015.06.038

Table 3
Characteristics of the activities in Case I.

Activity (a)	$d_a(x_a)$	$g_a^1(x_a)$	$g_a^2(x_a)$	$g_a^3(x_a)$	L_a	U_a
1	$2x_1+1$	$0.5-0.05x_1$	$0.6-0.06x_1$	$0.8-0.08x_1$	1	5
2	$2x_2$	$0.6-0.1x_2$	$0.5-0.08x_2$	–	1	5
3	$3x_3+4$	$0.7-0.12x_3$	$0.6-0.1x_3$	$0.5-0.05x_3$	1	5
4	x_4^2+1	$0.8-0.08x_4$	–	$0.7-0.07x_4$	1	6
5	x_5^2	–	$0.7-0.1x_4$	$0.6-0.09x_4$	1	5

Table 4
All admissible 3-partition cuts of Case I.

1. $[\phi,\phi]$	37. $[(2,3)^2,(1^*,3^q)^3]$	73. $[(2,3^*)^2,(1^*,3)^3]$	109. $[(1)^2,(1^q,3^*)^3]$	145. $[(5)^3,(1,2^*)^1]$
2. $[(1,2)^1,\phi]$	38. $[(1^*,3)^3,(1,2)^1]$	74. $[(2,3^*)^2,(1,3^*)^3]$	110. $[(1)^2,(1^*,3)^3]$	146. $[(4)^3,(3)^1]$
3. $[(2,3)^2,\phi]$	39. $[(1,3^*)^3,(1^q,2)^1]$	75. $[(4)^3,(1,2)^1]$	111. $[(1^q)^2,(1,3^*)^3]$	147. $[(5)^3,(2^*,3)^2]$
4. $[(1,3)^3,\phi]$	40. $[(1,3)^3,(1^q,2^*)^1]$	76. $[(1^*,3)^3,(1^*,2)^1]$	112. $[(2,3^*)^2,(4)^3]$	148. $[(5)^3,(2,3^*)^2]$
5. $[(1^*,2)^1,\phi]$	41. $[(1^*,3)^3,(2,3^q)^2]$	77. $[(1^*,3)^3,(1,2^*)^1]$	113. $[(5)^3,(1,2)^1]$	149. $[(4)^3,(1)^2]$
6. $[(1,2^*)^1,\phi]$	42. $[(1,3^*)^3,(2,3)^2]$	78. $[(1,3^*)^3,(1^q,2^*)^1]$	114. $[(4)^3,(1^*,2)^1]$	150. $[(5)^3,(1^*,3)^3]$
7. $[(1,2)^1,(1^q,2^q)^1]$	43. $[(1,3)^3,(2^*,3^q)^2]$	79. $[(4)^3,(2,3)^2]$	115. $[(4)^3,(1,2^*)^1]$	151. $[(5)^3,(1,3^*)^3]$
8. $[(1,2)^1,(2^q,3)^2]$	44. $[(1^*,3)^3,(1,3^q)^3]$	80. $[(1^*,3)^3,(2^*,3^q)^2]$	116. $[(1^*,3)^3,(3^q)^1]$	152. $[(4)^3,(4^q)^3]$
9. $[(1,2)^1,(1^q,3)^3]$	45. $[(1,3^*)^3,(1^q,3)^3]$	81. $[(1,3^*)^3,(2^*,3)^2]$	117. $[(5)^3,(2,3)^2]$	153. $[(4)^1,(3)^1]$
10. $[(2,3)^2,(1,2^q)^1]$	46. $[(4)^3,\phi]$	82. $[(1,3^*)^3,(2,3^*)^2]$	118. $[(4)^3,(2^*,3)^2]$	154. $[(4)^1,(1)^2]$
11. $[(2,3)^2,(2^q,3^q)^2]$	47. $[(4)^1,\phi]$	83. $[(4)^3,(1,3)^3]$	119. $[(4)^3,(2,3^*)^2]$	155. $[(3)^1,(5)^2]$
12. $[(2^*,3)^2,\phi]$	48. $[(3)^1,(1,2)^1]$	84. $[(1^*,3)^3,(1^*,3^q)^3]$	120. $[(1,3^*)^3,(1^q)^2]$	156. $[(4)^1,(4^q)^3]$
13. $[(2,3^*)^2,\phi]$	49. $[(3)^1,(2,3^q)^2]$	85. $[(1,3^*)^3,(1^q,3^*)^3]$	121. $[(5)^3,(1,3)^3]$	157. $[(4^q)^1,(4)^3]$
14. $[(2,3)^2,(1,3^q)^3]$	50. $[(1^*,2)^1,(1^*,2^q)^1]$	86. $[(5)^3,\phi]$	122. $[(4)^3,(1^*,3)^3]$	158. $[(3)^1,(5)^3]$
15. $[(1,3)^3,(1^q,2)^1]$	51. $[(1^*,2)^1,(2^q,3^*)^2]$	87. $[(4)^1,(1,2)^1]$	123. $[(4)^3,(1,3^*)^3]$	159. $[(5)^2,(3)^1]$
16. $[(1,3)^3,(2,3^q)^1]$	52. $[(3^q)^1,(2,3)^2]$	88. $[(4)^1,(2,3)^2]$	124. $[(4)^1,(1^*,2)^1]$	160. $[(1)^2,(4)^1]$
17. $[(1,3)^3,(1^q,3^q)^3]$	53. $[(1,2^*)^1,(1^q,2^*)^1]$	89. $[(3)^1,(1^*,2)^1]$	125. $[(4)^1,(1,2^*)^1]$	161. $[(5)^2,(1)^2]$
18. $[(1^*,3)^3,\phi]$	54. $[(1,2^*)^1,(2^*,3)^2]$	90. $[(3)^1,(1,2^*)^1]$	126. $[(4)^1,(2^*,3)^2]$	162. $[(5)^2,(4)^3]$
19. $[(1,3^*)^3,\phi]$	55. $[(1^*,2)^1,(1^*,3)^3]$	91. $[(3)^1,(2^*,3^q)^2]$	127. $[(4)^1,(2,3^*)^2]$	163. $[(1)^2,(5)^3]$
20. $[(3)^1,\phi]$	56. $[(1,2^*)^1,(2,3^*)^2]$	92. $[(3)^1,(2,3^*)^2]$	128. $[(3)^1,(3^q)^1]$	164. $[(5)^3,(3)^1]$
21. $[(1^*,2)^1,(1,2^q)^1]$	57. $[(3)^1,(1,3^q)^3]$	93. $[(3^q)^1,(2^*,3)^2]$	129. $[(3)^1,(1)^2]$	165. $[(4)^3,(4^q)^1]$
22. $[(1^*,2)^1,(2^q,3)^2]$	58. $[(3^q)^1,(1,3)^3]$	94. $[(1,2^*)^1,(1^q)^2]$	130. $[(4)^1,(1^*,3)^3]$	166. $[(5)^3,(1)^2]$
23. $[(1^*,2)^1,(1,3)^3]$	59. $[(1,2^*)^1,(1^q,3^*)^3]$	95. $[(4)^1,(1,3)^3]$	131. $[(4)^1,(1,3^*)^3]$	167. $[(4)^3,(5)^2]$
24. $[(1,2^*)^1,(1^q,2)^1]$	60. $[(1^*,2)^1,(1,3^*)^3]$	96. $[(3)^1,(1^*,3^q)^3]$	132. $[(3)^1,(4)^3]$	168. $[(5)^3,(4)^3]$
25. $[(1,2^*)^1,(2,3)^2]$	61. $[(1^q)^2,(1,2)^1]$	97. $[(3^q)^1,(1^*,3)^3]$	133. $[(1^*,2)^1,(5)^3]$	169. $[(4)^1,(4^q)^1]$
26. $[(1,2^*)^1,(1^q,3)^3]$	62. $[(2^*,3)^2,(1^*,2)^1]$	98. $[(3)^1,(1,3^*)^3]$	134. $[(5)^2,(1^*,2)^1]$	170. $[(4)^1,(5)^2]$
27. $[(1,2)^1,(2^q,3^*)^2]$	63. $[(2^*,3)^2,(1,2^*)^1]$	99. $[(1^*,2)^1,(4)^3]$	135. $[(1)^2,(3)^1]$	171. $[(4)^1,(5)^3]$
28. $[(1,2)^1,(1^q,3^*)^3]$	64. $[(2,3^*)^2,(1^*,2^q)^1]$	100. $[(1)^2,(1^*,2)^1]$	136. $[(5)^2,(2^*,3)^2]$	172. $[(5)^2,(4)^1]$
29. $[(2^*,3)^2,(1,2)^1]$	65. $[(1)^2,(2,3)^2]$	101. $[(1^q)^2,(1,2^*)^1]$	137. $[(5)^2,(2,3^*)^2]$	173. $[(5)^2,(5^q)^2]$
30. $[(2,3^*)^2,(1,2^q)^1]$	66. $[(2^*,3)^2,(2^*,3^q)^2]$	102. $[(2^*,3)^2,(3^q)^1]$	138. $[(1)^2,(1^q)^2]$	174. $[(5)^2,(5^q)^3]$
31. $[(2,3)^2,(1^*,2^q)^1]$	67. $[(2,3^*)^2,(2^q,3^*)^2]$	103. $[(5)^2,(2,3)^2]$	139. $[(5)^2,(1,2^*)^1]$	175. $[(5^q)^2,(5)^3]$
32. $[(2^*,3)^2,(2,3^q)^2]$	68. $[(1)^2,(1^q,2)^1]$	104. $[(1)^2,(2^*,3)^2]$	140. $[(5)^2,(1^*,3)^3]$	176. $[(5)^3,(4)^1]$
33. $[(2,3^*)^2,(2^q,3)^2]$	69. $[(5)^2,\phi]$	105. $[(1)^2,(2,3^*)^2]$	141. $[(5)^2,(1,3^*)^3]$	177. $[(5)^3,(5^q)^2]$
34. $[(1)^2,\phi]$	70. $[(1)^2,(1^q,3)^3]$	106. $[(5)^2,(1,2)^1]$	142. $[(1)^2,(4)^3]$	178. $[(5^q)^3,(5)^2]$
35. $[(2^*,3)^2,(1,3^q)^3]$	71. $[(1^q)^2,(1,3)^3]$	107. $[(1)^2,(1^q,2^*)^1]$	143. $[(2,3^*)^2,(5)^3]$	179. $[(5)^3,(5^q)^3]$
36. $[(2,3^*)^2,(1,3)^3]$	72. $[(2^*,3)^2,(1^*,3^q)^3]$	108. $[(5)^2,(1,3)^3]$	144. $[(5)^3,(1^*,2)^1]$	

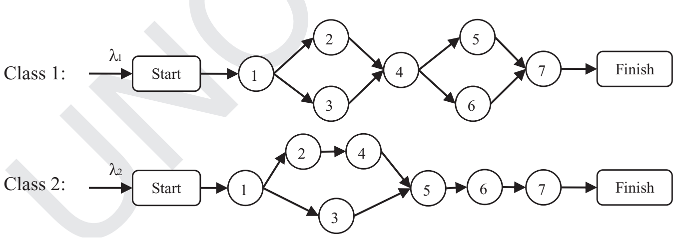

Fig. 3. Dynamic PERT network of Case II.

Table 5
Characteristics of the activities in Case II.

Activity (a)	$d_a(x_a)$	$g_a^1(x_a)$	$g_a^2(x_a)$	L_a	U_a
1	$2.5x_1$	$0.3-0.03x_1$	$0.35-0.03x_1$	1	5
2	$3x_2+1$	$0.3-0.05x_2$	$0.25-0.04x_2$	1	5
3	$3.5x_3+2$	$0.45-0.05x_3$	$0.4-0.04x_3$	1	5
4	x_4^2	$0.35-0.04x_4$	$0.4-0.05x_4$	1	6
5	$2.7x_5+3$	$0.3-0.04x_5$	$0.35-0.05x_5$	1	5
6	$3.2x_6+2$	$0.35-0.03x_6$	$0.4-0.04x_6$	1	6
7	$2.5x_7$	$0.3-0.03x_7$	$0.2-0.02x_7$	1	6

Table 5 shows the characteristics of the activities, where the time unit and the cost unit are in year and in thousand dollars, respectively. The corresponding stochastic process $\{X(t), t \geq 0\}$ has 211 states. We set the goals as $b_1 = 45$, $b_2 = 2$, $b_3 = 0.05$. We also consider the similar sets of c as the first case.

5.3. Case III

The multi-class dynamic PERT network of Case III is shown in Fig. 4 (taken from Yaghoubi, 2012). In this case, it is assumed that two classes of projects exist and the system can perform up to two projects. The new projects, including all their activities, are generated according to two independent Poisson processes with the rates of $\lambda_1 = 3$ and $\lambda_2 = 1$ per year for the projects of class 1 and class 2, respectively. Furthermore, the amount of resource available to be allocated to all service stations is 25.

Table 6 shows the characteristics of the activities, where the time unit and the cost unit are in month and in hundred dollars, respectively. The corresponding stochastic process $\{X(t), t \geq 0\}$ has 683

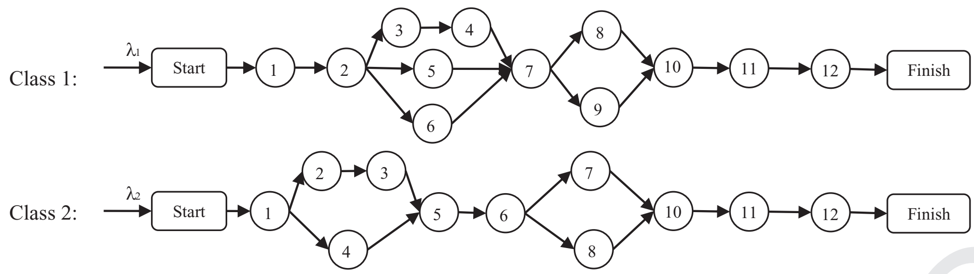

Fig. 4. Dynamic PERT network of Case III.

Table 6
Characteristics of the activities in Case III.

Activity (a)	$d_a(x_a)$	$g_a^1(x_a)$	$g_a^2(x_a)$	L_a	U_a
1	$2.5x_1$	$0.3-0.03x_1$	$0.35-0.03x_1$	1	5
2	$3x_2+1$	$0.3-0.05x_2$	$0.25-0.04x_2$	1	5
3	$3.5x_3+2$	$0.45-0.05x_3$	$0.4-0.04x_3$	1	5
4	x_4^2	$0.35-0.04x_4$	$0.4-0.05x_4$	1	6
5	$2.7x_5+3$	$0.3-0.04x_5$	$0.35-0.05x_5$	1	5
6	$3.2x_6+2$	$0.35-0.03x_6$	$0.4-0.04x_6$	1	6
7	$2.5x_7$	$0.3-0.03x_7$	$0.2-0.02x_7$	1	6
8	$3x_8$	$0.3-0.03x_8$	$0.25-0.04x_8$	1	5
9	$3.2x_9+1$	$0.4-0.05x_9$	–	1	5
10	$2.9x_{10}+2$	$0.4-0.04x_{10}$	$0.3-0.03x_{10}$	1	6
11	$3x_{11}$	$0.35-0.04x_{11}$	$0.4-0.05x_{11}$	1	6
12	$3.3x_{12}+1$	$0.3-0.03x_{12}$	$0.35-0.05x_{12}$	1	5

states. We set the goals as $b_1 = 82$, $b_2 = 3.6$, $b_3 = 0.05$. We also consider the similar sets of c as the first case.

5.4. Cases 4–10

The dynamic PERT networks of the other cases are shown in Figs. 5 and 6.

5.5. Simulated annealing results

According to the proposed SA algorithm in Section 4, we set the penalty coefficients to be $\gamma_1 = 10$, $\gamma_2 = 20$ and final temperature to be $\theta_f = 0.001$ in all numerical examples. The initial temperature is also determined based on (22). To do so, we use MATLAB 7 on a PC Pentium IV, CPU 3 gigahertz.

We first consider Case I. The optimal allocated resources, according to the SA algorithm, the computational times CT (mm:ss), and also the values of all objectives functions in Case I for the different combinations of T', H and τ according to the first set of weights ($c_1 = 0.3$, $c_2 = 0.1$, $c_3 = 0.6$) are shown in Table 7.

The optimal allocated resources in Case I with the parameters of set 1 are: $x_1 = 1.639$, $x_2 = 4.848$, $x_3 = 3.117$, $x_4 = 1.087$, $x_5 = 1.195$, and the objective function values are $f_1 = 30.932$, $f_2 = 1.888$, $f_3 = 0.014$ ($z = 3.108$). Based on Table 7, if the values of H or τ are increased, for any specific T', the quality of solution is increased, but the computational times will also be increased, which is undesirable.

Similar to Table 7, we obtain the optimal allocated resources in Case I according to the other sets of c. Moreover, we achieve the optimal allocated resources in all 10 cases with four indicated sets of c, considering the SA algorithm. For instance, the Pareto-optimal solutions and the optimal resources of Cases I–III are given in Tables 8–10.

5.6. The discrete-time approximation technique

We consider the various combinations of T', R and Δt for every set of c. To do so, we use LINGO 8 on a PC Pentium 4, CPU 3 gigahertz. The optimal allocated resources, the computational times, and the values of all objectives functions in Case I for the different combinations of T', R and Δt according to set 1 ($c_1 = 0.3$, $c_2 = 0.1$, $c_3 = 0.6$) are shown in Table 11. So, the optimal allocated resources are: $x_1 = 2.535$, $x_2 = 3.471$, $x_3 = 3.366$, $x_4 = 1.348$, $x_5 = 1$, and the objective function values are: $f_1 = 30.927$, $f_2 = 1.932$, $f_3 = 0.013$ ($z = 3.321$).

According to Table 11, if the length of Δt is decreased, for any specific T', the accuracy of solution is increased, but the computational times will also be increased, which is undesirable.

Similar to Table 11, we also obtain the optimal allocated resources in Case I with other sets of c. Moreover, similar to Table 11, we obtain the optimal allocated resources in all 10 cases according to the 4 indicated sets of c, considering the discrete-time approximation technique. For instance, the Pareto-optimal solutions and the optimal allocated resources of Cases I–III are given in Tables 12–14.

Table 7
The computational results of SA algorithm in Case I according to set 1 ($c_1 = 0.3$, $c_2 = 0.1$, $c_3 = 0.6$).

T'	H	τ	x_1	x_2	x_3	x_4	x_5	z	f_1	f_2	f_3	CT
12	100	0.9	2.787	3.617	3.143	1.246	1.240	5.075	31.326	2.016	0.012	1′:36″
12	100	0.95	2.921	3.228	3.177	1.332	1.342	4.697	31.406	1.932	0.014	2′:27″
12	200	0.9	3.512	2.131	2.580	2.205	1.150	4.037	31.211	2.004	0.013	2′:51″
12	200	0.95	3.298	2.885	2.984	1.542	1.139	3.319	30.996	1.933	0.012	3′:50″
16.8	100	0.9	3.304	1.588	2.207	2.620	1.490	4.957	31.487	2.096	0.012	1′:44″
16.8	100	0.95	3.194	3.036	3.027	1.518	1.217	4.423	31.327	2.036	0.011	2′:19″
16.8	200	0.9	3.285	2.338	2.355	2.343	1.221	4.293	31.288	2.023	0.013	2′:53″
16.8	200	0.95	1.716	4.296	3.439	1.151	1.140	3.209	30.963	1.921	0.014	3′:42″
18	100	0.9	3.654	2.104	2.562	1.960	1.181	4.412	30.441	2.041	0.013	1′:51″
18	100	0.95	1.180	4.693	3.702	1.098	1.020	3.840	31.098	1.984	0.014	2′:46″
18	200	0.9	3.675	2.093	2.566	2.151	1.188	4.239	31.272	2.024	0.013	3′:02″
18	200	0.95	2.248	3.465	2.193	2.328	1.245	3.266	30.975	1.927	0.015	3′:55″
21.6	100	0.9	3.298	2.885	3.084	1.542	1.206	5.164	31.453	1.933	0.011	1′:39″
21.6	100	0.95	3.589	2.493	2.832	1.656	1.342	4.022	31.203	2.002	0.013	2′:38″
21.6	200	0.9	2.281	4.128	2.624	1.192	1.709	3.443	31.033	1.904	0.012	3′:42″
21.6	200	0.95	1.639	4.848	3.117	1.087	1.195	3.108	30.932	1.888	0.013	3′:56″

Please cite this article as: S. Yaghoubi et al., Resource allocation in multi-class dynamic PERT networks with finite capacity, European Journal of Operational Research (2015), http://dx.doi.org/10.1016/j.ejor.2015.06.038

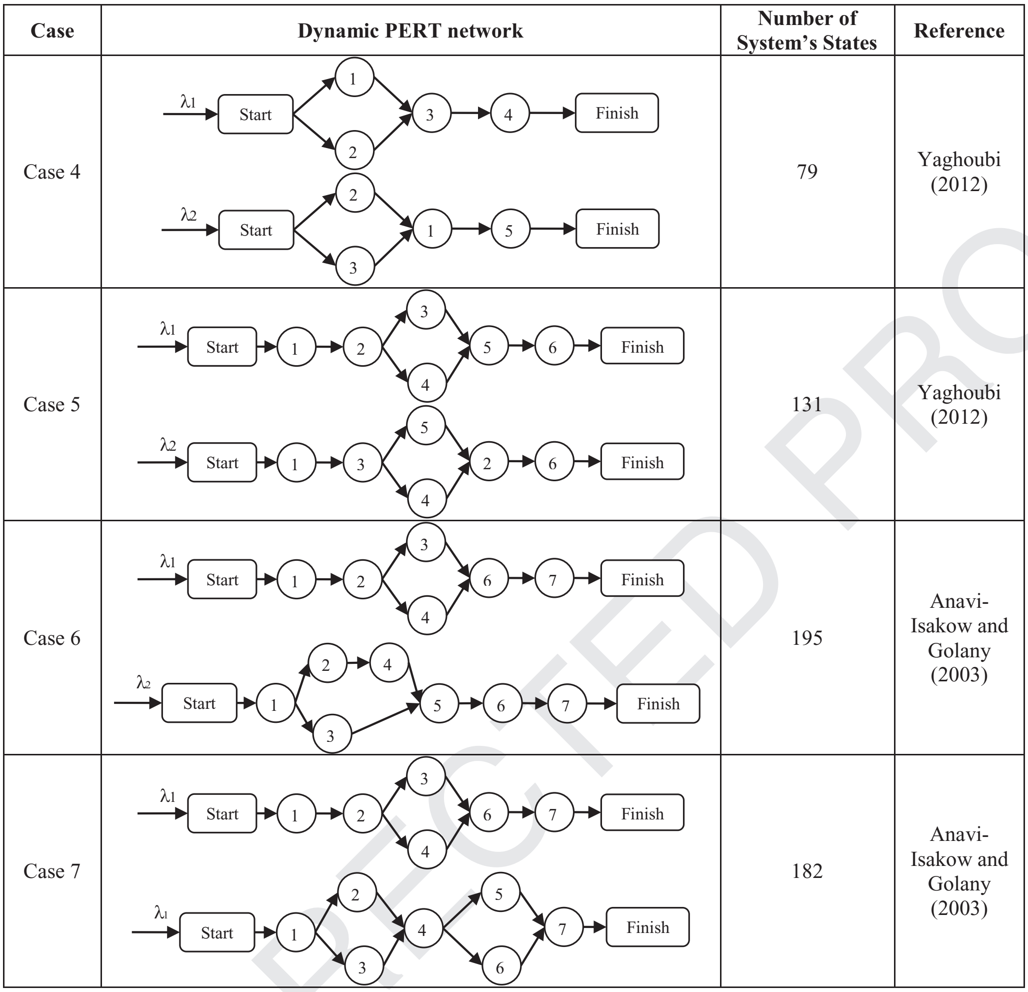

Fig. 5. Dynamic PERT networks of Cases 4–7 with two classes of projects.

Table 8
Pareto-optimal solutions of Case I, using SA algorithm.

c	x_1	x_2	x_3	x_4	x_5	z	f_1	f_2	f_3	CT
Set 1	1.639	4.848	3.117	1.087	1.195	3.108	30.932	1.888	0.013	3′:56″
Set 2	2.506	3.185	3.558	1.011	1.247	3.190	30.635	1.919	0.014	3′:11″
Set 3	2.193	3.299	3.244	1.263	1.410	1.620	30.299	1.924	0.013	3′:46″
Set 4	1.789	4.310	3.161	1.085	1.144	1.686	30.169	1.933	0.014	3′:45″

Table 9
Pareto-optimal solutions of Case II, using SA algorithm.

c	x_1	x_2	x_3	x_4	x_5	x_6	x_7	z	f_1	f_2	f_3	CT
Set 1	3.787	1.297	1.259	1.253	2.196	1.218	3.662	4.513	46.316	2.451	0.010	4′:02″
Set 2	4.063	1.519	1.093	1.166	2.658	1.395	2.557	4.898	45.932	2.490	0.010	4′:07″
Set 3	3.699	1.259	1.200	1.169	2.017	1.142	3.924	2.508	45.502	2.501	0.010	3′:56″
Set 4	3.234	1.182	1.132	2.161	2.266	1.308	2.659	2.580	45.219	2.516	0.010	4′:03″

Table 10
Pareto-optimal solutions of Case III, using SA algorithm.

c	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	z	f_1	f_2	f_3	CT
Set 1	3.071	1.527	2.412	1.379	1.395	1.854	2.483	1.736	1.639	3.735	2.127	1.629	5.180	83.548	4.118	0.005	6′:22″
Set 2	2.607	2.237	1.623	1.469	2.136	1.734	2.541	2.046	1.156	3.486	2.028	1.935	5.779	83.156	4.159	0.005	6′:05″
Set 3	3.496	1.830	1.775	1.276	2.256	1.539	2.131	2.157	1.376	3.068	1.825	2.254	5.526	83.105	4.167	0.005	5′:53″
Set 4	3.234	1.182	1.531	2.161	1.927	1.326	2.352	1.933	1.272	3.473	2.237	2.028	3.367	82.337	4.178	0.005	6′:12″

Please cite this article as: S. Yaghoubi et al., Resource allocation in multi-class dynamic PERT networks with finite capacity, European Journal of Operational Research (2015), http://dx.doi.org/10.1016/j.ejor.2015.06.038

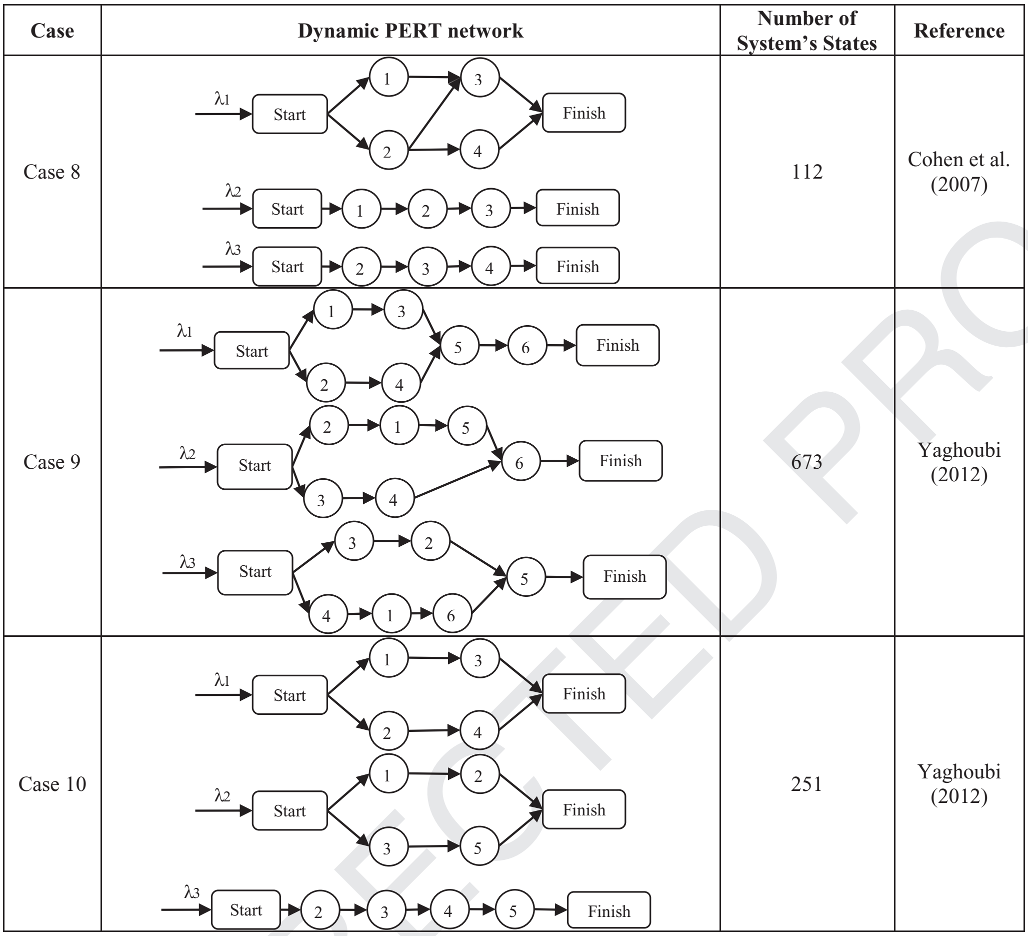

Fig. 6. Dynamic PERT networks of Cases 8–10 with three classes of projects.

Table 11
The computational results of the discrete-time approximation technique in Case I according to set 1.

T'	R	Δt	x_1	x_2	x_3	x_4	x_5	z	f_1	f_2	f_3	CT
12	80	0.15	2.887	1.246	1.935	2.883	1.802	5.427	31.628	2.143	0.011	78′:17″
12	100	0.12	3.589	2.493	2.832	1.656	1	4.022	30.402	2.002	0.013	106′:47″
12	120	0.1	2.431	3.569	3.510	1.213	1	3.336	31.001	1.934	0.014	135′:15″
12	150	0.08	2.535	3.471	3.366	1.348	1	3.321	30.927	1.932	0.013	163′:20″
16.8	112	0.15	2.706	1.175	1.908	2.956	1.855	5.555	31.667	2.156	0.011	188′:21″
16.8	140	0.12	3.547	1.890	2.401	2.304	1.397	4.455	31.337	2.046	0.012	154′:32″
16.8	168	0.1	2.431	3.569	3.510	1.213	1	3.336	31.001	1.934	0.014	241′:45″
16.8	210	0.08	2.535	3.471	3.366	1.348	1	3.321	30.927	1.932	0.013	316′:29″
18	120	0.15	2.993	1.319	1.981	2.819	1.755	5.305	31.591	2.130	0.011	126′:33″
18	150	0.12	3.725	2.247	2.662	1.973	1.180	4.046	31.214	2.005	0.013	192′:17″
18	180	0.1	2.431	3.569	3.510	1.213	1	3.336	31.001	1.934	0.014	304′:24″
18	225	0.08	2.535	3.471	3.366	1.348	1	3.321	30.927	1.932	0.013	382′:51″
21.6	144	0.15	1	5	3.824	1	1	4.910	31.473	2.091	0.012	148′:32″
21.6	180	0.12	3.490	2.846	3.083	1.476	1	3.664	31.099	1.966	0.014	261′:49″
21.6	216	0.1	2.431	3.569	3.510	1.213	1	3.336	31.001	1.934	0.014	375′:18″
21.6	270	0.08	2.535	3.471	3.366	1.348	1	3.321	30.927	1.932	0.013	403′:36″

Table 12
Pareto-optimal solutions of Case I, using the discrete-time approximation technique.

c	x_1	x_2	x_3	x_4	x_5	z	f_1	f_2	f_3	CT
Set 1	2.535	3.471	3.366	1.348	1	3.321	30.927	1.932	0.013	403′:36″
Set 2	3.445	1.739	2.301	2.252	1.379	2.458	30.246	2.092	0.012	361′:59″
Set 3	2.635	3.296	3.295	1.271	1	1.810	30.362	1.962	0.014	355′:17″
Set 4	3.064	2.908	3.106	1.390	1	1.928	30.193	1.986	0.014	387′:22″

Table 13
Pareto-optimal solutions of Case II, using the discrete-time approximation technique.

c	x_1	x_2	x_3	x_4	x_5	x_6	x_7	z	f_1	f_2	f_3	CT
Set 1	2.734	1	1	1.759	2.601	2.060	3.380	4.979	46.494	2.498	0.010	401′:14″
Set 2	2.792	1	1	1.754	2.648	1.850	3.358	5.096	46.019	2.510	0.010	386′:02″
Set 3	2.847	1	1	1.749	2.693	1.636	3.335	2.612	45.523	2.522	0.009	412′:45″
Set 4	2.874	1	1	1.747	2.716	1.527	3.324	2.646	45.264	2.529	0.010	392′:17″

Table 14
Pareto-optimal solutions of Case III, using the discrete-time approximation technique.

c	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	z	f_1	f_2	f_3	CT
Set 1	2.548	2.465	1	2.456	2.621	1.815	1.681	2.135	1	2.317	2.061	2.742	6.479	83.943	4.248	0.005	576′:07″
Set 2	3.172	2.835	1	1.715	2.274	2.215	1.269	1.925	1	2.956	2.371	2.26	6.985	83.397	4.298	0.005	553′:41″
Set 3	2.419	2.032	1	1.884	2.283	1.832	2.961	1.857	1	2.612	2.363	2.748	5.608	83.121	4.722	0.005	604′:12″
Set 4	3.248	1.624	1	2.156	2.149	1.338	2.432	2.76	1	2.534	2.912	1.793	7.833	82.783	4.982	0.005	592′:33″

Table 15
Comparing the SA results against the discrete-time approximation technique in Case I according to set 1.

No.	T'	SA algorithm		Discrete-time approximation technique		$z_{SA}^{Cont.}$	$z_{Dis.}^{Cont.}$	$\lvert z_{Dis.}^{Cont.} - z_{SA}^{Cont.} \rvert$	Rank
		z_{SA}	CT	$z_{Dis.}$	CT				
1	12	5.075	1′:36″	5.427	78′:17″	4.422	5.539	1.117	16
2	12	4.697	2′:27″	4.022	106′:47″	4.685	4.115	|−0.570|	10
3	12	4.037	2′:51″	3.336	135′:15″	4.037	3.464	|−0.573|	11
4	12	3.319	3′:50″	3.321	163′:20″	3.319	3.442	0.123	3
5	16.8	4.957	1′:44″	5.555	188′:21″	4.957	5.658	0.701	12
6	16.8	4.423	2′:19″	4.455	154′:32″	4.423	4.531	0.108	2
7	16.8	4.293	2′:53″	3.336	241′:45″	4.293	3.464	|−0.829|	14
8	16.8	3.209	3′:42″	3.321	316′:29″	3.209	3.442	0.233	6
9	18	4.412	1′:51″	5.305	126′:33″	4.412	5.413	1.001	15
10	18	3.840	2′:46″	4.046	192′:17″	3.840	4.115	0.275	7
11	18	4.239	3′:02″	3.336	304′:24″	4.239	3.464	|−0.775|	13
12	18	3.266	3′:55″	3.321	382′:51″	3.266	3.442	0.176	4
13	21.6	5.164	1′:39″	4.910	148′:32″	4.843	5.027	0.184	5
14	21.6	4.022	2′:38″	3.664	261′:49″	4.022	3.701	|−0.321|	8
15	21.6	3.443	3′:42″	3.336	375′:18″	3.443	3.464	0.021	1
16	21.6	3.108	3′:56″	3.321	403′:36″	3.108	3.442	0.334	9

5.7. The SA results vs. the discrete-time approximation technique

As we noted in Section 3, solving the goal attainment formulation (19), optimally, and consequently, comparing the simulated annealing results against the optimal results are impossible. Therefore, we try to compare the simulated annealing results against the results of the discrete-time problem (19). For this purpose, we evaluate both solutions of the SA and the discrete-time approximation technique using the objective function of the continuous-time model (20) in all 10 cases using the statistical inference. Note that the models of (19) and (20) are equivalent.

A paired sample Wilcoxon signed-rank test analysis with $\alpha = 0.05$ is utilized to investigate whether solutions obtained by solving the SA algorithm differ from the discrete-time approximation or not. The paired sample Wilcoxon signed-rank test is alternative non-parametric methods of paired sample t-test. When the normality assumption is not satisfied or the sample size is too small, t-test is not valid (for more details see Siegel (1956)). Therefore, in this paper, the paired sample Wilcoxon signed-rank test is used to test their means in all 10 cases. Let n be the sample size, the number of pairs. For $i = 1, \ldots, n$, let $z_{SA,i}^{Cont.}$ and $z_{Dis.,i}^{Cont.}$ be the objective values obtained by SA algorithm and the discrete-time approximation technique, respectively. Also, let $z_{SA}^{Cont.}$ and $z_{Dis.}^{Cont.}$ be the mean of the objective function values of model (20) obtained by the SA algorithm and the discrete-time approximation technique, respectively. Null hypothesis (H_0) is considered as $z_{Dis.}^{Cont.} - z_{SA}^{Cont.} = 0$, while alternate hypothesis (H_1) is $z_{Dis.}^{Cont.} - z_{SA}^{Cont.} \succ 0$.

Consequently, the test procedure will be as follows:

- For $i = 1, \ldots, n$, calculate $|z_{Dis.,i}^{Cont.} - z_{SA,i}^{Cont.}|$ and $\text{sgn}(z_{Dis.,i}^{Cont.} - z_{SA,i}^{Cont.})$, where sgn is the sign function.
- Exclude pairs with $|z_{Dis.,i}^{Cont.} - z_{SA,i}^{Cont.}| = 0$. Let n_r be the reduced sample size.
- Order the remaining n_r pairs from smallest absolute difference to largest absolute difference, $|z_{Dis.,i}^{Cont.} - z_{SA,i}^{Cont.}|$.
- Rank the pairs, starting with the smallest as 1. Ties receive a rank equal to the average of the ranks they span. Let R_i denote the rank.
- Calculate the test statistic $\sum_{i=1}^{n} (\text{sgn}(z_{Dis.,i}^{Cont.} - z_{SA,i}^{Cont.}) \cdot R_i)$, the absolute value of the sum of the signed ranks.
- As n_r increases, the sampling distribution of W converges to a normal distribution. Thus, for $n_r \geq 10$, a z_W-score can be calculated as $z_W = \frac{W-0.5}{\sigma_W}$, where $\sigma_W = \sqrt{\frac{n_r(n_r+1)(2n_r+1)}{6}}$. If $z_W \succ z_{critical}$, reject H_0. For $n_r \prec 10$, W is compared to a critical value from a reference table. If $W \geq W_{critical,n_r}$, reject H_0. Alternatively, a p-Value can be calculated from enumeration of all possible combinations of W given n_r.

For instance, we again consider Case I with the parameters of set 1. Based on the results expressed in Table 15, $z_W = 0.608$, while z_W-critical one-tail is 1.65. Therefore, H_0 is accepted. Namely, the quality of the optimal solution obtained by solving the SA algorithm is equal to the discrete-time approximation technique. Moreover, according to Table 16, we find that the quality of the solutions obtained by the SA algorithm in 35 out of the 40 sample cases is equal to the discrete-time approximation technique, while in 5 sample cases is better than

Table 16
The accepted hypotheses in comparing the SA algorithm against the discrete-time approximation technique in all 10 case.

c	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10
Set 1	H_0	H_0	H_0	H_0	H_1	H_0	H_0	H_0	H_0	H_0
Set 2	H_0	H_0	H_1	H_0	H_0	H_0	H_0	H_0	H_0	H_0
Set 3	H_0	H_0	H_0	H_0	H_0	H_0	H_0	H_1	H_0	H_0
Set 4	H_0	H_0	H_1	H_0	H_0	H_0	H_0	H_0	H_1	H_0

Table 17
The computational results of the SA algorithm with $CT = 2':00''$ and $CT = 3':00''$ in case I according to set 1.

T'	H	τ	z with $CT = 2':00''$	z with $CT = 3':00''$
12	100	0.9	4.794	4.372
12	100	0.95	4.763	4.218
12	200	0.9	4.721	4.259
12	200	0.95	4.697	4.184
16.8	100	0.9	4.882	4.402
16.8	100	0.95	4.711	4.321
16.8	200	0.9	4.736	4.175
16.8	200	0.95	4.681	4.163
18	100	0.9	4.594	4.348
18	100	0.95	4.608	4.251
18	200	0.9	4.733	4.276
18	200	0.95	4.519	4.193
21.6	100	0.9	4.825	4.379
21.6	100	0.95	4.532	4.216
21.6	200	0.9	4.641	4.240
21.6	200	0.95	4.505	4.187

the discrete-time approximation technique. This shows that the SA algorithm is a very good algorithm for obtaining the optimal allocated resources.

Moreover, according to Table 15 and the results of other cases, the computational times (CT), using the SA algorithm, are remarkably decreased, comparing against the discrete-time approximation. Therefore, it is clearly concluded that the SA algorithm is computationally superior in terms of finding optimal or near-optimal solutions to large-scale problems than the discrete-time approximation technique.

On the other hand, to illustrate the impact of computational time in SA algorithm, we now consider reaching $CT = 2':00''$ and $CT = 3':00''$ as stoppage criterion. Based on Table 17 and the other cases, the algorithm converges to the same solution at the same computational time.

6. Conclusion

In this paper, we modeled the multi-class dynamic PERT networks with finite capacity (CONPIP) as queuing networks and developed a multi-objective model to optimally control the resources allocated to the servers.

Our proposed model is applicable for organizations which get similar projects of different classes, for example building construction projects, where similar successive installations are created and built over time, or maintenance projects. Another important application of the proposed approach is the analysis of product development projects. While product development efforts are often viewed as unique configurations of idiosyncratic tasks, in reality different projects within an organization often exhibit substantial similarity in the flow of their constituent activities and their precedence requirements. For example, activities such as "Manufacturing Process Development" or "Product Testing" are common in almost all product development projects, but the processing times to perform each of these activities are varied for different projects. Moreover, new product development projects are generated over time.

In this paper, it was assumed that the capacity of system is finite and the new projects from different classes, including all their activities, are generated according to independent Poisson processes with different rates over the time horizon. Each activity of a project is performed at a devoted service station with one server located in a node of the network based on FCFS discipline, whereas activity durations for different classes are independent and exponentially distributed random variables. Moreover, it was assumed that the mean times spent in each service station for different classes are decreased and the operating cost of the service station is increased when we allocate more resources to that particular service station.

For modeling the multi-class dynamic PERT networks with CONPIP, we first considered every class separately and converted the queueing network of every class into an appropriate stochastic network and then developed a continuous-time Markov model for the problem. The number of system states grows exponentially with the number of UDCs of different classes and the system capacity, which is the main drawback of the proposed approach. However, this is a major drawback in most analytical approaches in this area.

In our model, the total operating costs of service stations per period was considered as the first objective and the mean project completion time over all classes in the steady-state as the second one, both to be minimized. Moreover, the probability that the system becomes empty in the steady-state was considered as the third objective function to be minimized as well.

Since this continuous-time stochastic programming problem is impossible to solve optimally, we used the simulated annealing algorithm and also the discrete-time approximation technique based on goal attainment method for different combinations of the parameters including goals and weights of the objective functions to reach Pareto-optimal solutions. All SA experiments were replicated 10 times using different random initial solutions.

To show the effectiveness of the proposed metaheuristic approach, the SA results were then compared against the results of the discrete-time approximation of the original optimal control problem in all 10 cases. For this purpose, a paired sample Wilcoxon signed-rank test analysis was utilized to investigate whether solutions obtained by the SA algorithm differ from the discrete-time approximation.

According to the numerical experiments of Section 5, it is seen that the SA algorithm is an efficient method for multi-objective resource allocation problems in multi-class dynamic PERT networks with finite capacity. It was shown that the quality of the solutions obtained by the SA algorithm is equal or better than the discrete-time approximation technique. Moreover, the computational times using the SA algorithm are remarkably decreased, comparing against the discrete-time approximation. Therefore, it is concluded that the SA algorithm is efficient in terms of finding optimal or near-optimal solutions for medium and large scale cases.

References

Adler, P. S., Mandelbaum, A., Nguyen, V., & Schwerer, E. (1995). From project to process management: an empirically-based framework for analyzing product development time. *Management Science, 41*(3), 458–484.
Anavi-Isakow, S., & Golany, B. (2003). Managing multi-project environments through constant work-in-process. *International Journal of Project Management, 21*(1), 9–18.

Please cite this article as: S. Yaghoubi et al., Resource allocation in multi-class dynamic PERT networks with finite capacity, European Journal of Operational Research (2015), http://dx.doi.org/10.1016/j.ejor.2015.06.038

Artalejo, J. R. (1999). Accessible bibliography on retrial queues. *Mathematical and Computer Modelling, 30*(3–4), 1–6.

Azaron, A., & Modarres, M. (2005). Distribution function of the shortest path in networks of queues. *OR Spectrum, 27*(1), 123–144.

Azaron, A., Katagiri, H., Sakawa, M., Kato, K., & Memariani, A. (2006). A multi-objective resource allocation problem in PERT networks. *European Journal of Operational Research, 172*(3), 838–854.

Azaron, A., Katagiri, H., & Sakawa, M. (2007). Time-cost trade-off via optimal control theory in Markov PERT networks. *Annals of Operations Research, 150*(1), 47–64.

Azaron, A., & Tavakkoli-Moghaddam, R. (2007). Multi-objective time–cost trade-off in dynamic PERT networks using an interactive approach. *European Journal of Operational Research, 180*(3), 1186–1200.

Azaron, A., Fynes, B., & Modarres, M. (2011). Due date assignment in repetitive projects. *International Journal of Production Economics, 129*(1), 79–85.

Balsamo, S., De Nitto Persone, V., & Inverardi, P. (2003). A review on queueing network models with finite capacity queues for software architectures performance prediction. *Performance Evaluation, 51*(2–4), 269–288.

Bandyopadhyay, S., Saha, S., Maulik, U., & Deb, K. (2008). A simulated annealing-based multiobjective optimization algorithm: AMOSA. *IEEE Transactions on Evolutionary Computation, 12*(3), 269–282.

Berman, E. (1964). Resource allocation in a PERT network under continuous activity time-cost function. *Management Science, 10*(4), 734–745.

Byali, R. P., & Kannan, M. V. (2008). Critical chain project management – a new project management philosophy for multi project environment. *Journal of Spacecraft Technology, 18*(1), 30–36.

Chen, P. H., & Shahandashti, S. M. (2009). Hybrid of genetic algorithm and simulated annealing for multiple project scheduling with multiple resource constraints. *Automation in Construction, 18*(4), 434–443.

Chen, V. (1994). A 0–1 goal programming-model for scheduling multiple maintenance project at a copper mine. *European Journal of Operational Research, 6*(1), 176–191.

Coello, C., Veldhuizen, D. V., & Lamont, G. (2002). *Evolutionary algorithms for solving multi-objective problems*. Norwell, MA: Kluwer.

Cohen, I., Golany, B., & Shtub, A. (2005). Managing stochastic, finite capacity, multi-project systems through the Cross Entropy methodology. *Annals of Operations Research, 134*(1), 183–199.

Cohen, I., Golany, B., & Shtub, A. (2007). Resource allocation in stochastic, finite-capacity, multi-project systems through the cross entropy methodology. *Journal of Scheduling, 10*(3), 181–193.

Deb, K. (2001). *Multiobjective optimization using evolutionary algorithms*. New York: Wiley.

Demeulemeester, E., Herroelen, W., & Elmaghraby, S. (1993). *Optimal procedures for the discrete time-cost trade-off problem in project networks. Research report*. Leuven , Belgium: Department of Applied Economics, Katholieke Universiteit Leuven.

Falk, J., & Horowitz, J. (1972). Critical path problem with concave cost curves. *Management Science, 19*(4), 446–455.

Fatemi Ghomi, S. M. T., & Ashjari, B. (2002). A simulation model for multi-project resource allocation. *International Journal of Project Management, 20*(2), 127–130.

Fulkerson, D. (1961). A network flow computation for project cost curves. *Management Science, 7*(2), 167–178.

Geman, S., & Geman, D. (1984). Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transaction on Pattern Analysis and Machine Intelligence, 6*(6), 721–741.

Gonçalves, J. F., Mendes, J. J. M., & Resende, M. G. C. (2008). A genetic algorithm for the resource constrained multi-project scheduling problem. *European Journal of Operational Research, 189*(3), 1171–1190.

Jouini, O., Dallery, Y., & Aksin, Z. (2009). Queueing models for full-flexible multi-class call centers with real-time anticipated delays. *International Journal of Production Economics, 120*(2), 389–399.

Kanagasabapathi, B., Rajendran, C., & Ananthanarayanan, K. (2009). Performance analysis of scheduling rules in resource-constrained multiple projects. *International Journal of Industrial and Systems Engineering, 4*(5), 502–535.

Kao, H. P., Hsieh, B., & Yeh, Y. (2006). A petri-net based approach for scheduling and rescheduling resource-constrained multiple projects. *Journal of the Chinese Institute of Industrial Engineers, 23*(6), 468–477.

Kelly, J. (1961). Critical path planning and scheduling: Mathematical basis. *Operations Research, 9*(3), 296–320.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*(4598), 671–680.

Kruger, D., & Scholl, A. (2010). Managing and modelling general resource transfers in (multi-) project scheduling. *OR Spectrum, 32*(2), 369–394.

Kulkarni, V., & Adlakha, V. (1986). Markov and Markov-regenerative PERT networks. *Operations Research, 34*(5), 769–781.

Kumanan, S., Jegan, J. G., & Raja, K. (2006). Multi-project scheduling using an heuristic and a genetic algorithm. *The International Journal of Advanced Manufacturing Technology, 31*(3–4), 360–366.

Kurtulus, I. S., & Davis, E. W. (1982). Multi-project scheduling: categorization of heuristic rules performance. *Management Science, 28*(2), 161–172.

Kurtulus, I. S., & Narula, S. C. (1985). Multi-project scheduling: analysis of project performance. *IIE Transactions, 17*(1), 58–66.

Lamberson, L., & Hocking, R. (1970). Optimum time compression in project scheduling. *Management Science, 16*(10), 597–606.

Li, C., & Wang, K. (2009). The risk element transmission theory research of multi-objective risk-time-cost trade-off. *Computers and Mathematics with Applications, 57*(11–12), 1792–1799.

Lova, A., & Tormos, P. (2001). Analysis of scheduling schemes and heuristic rules performance in resource-constrained multiproject scheduling. *Annals of Operations Research, 102*(1–4), 263–286.

Lova, A., Maroto, C., & Tormos, P. (2000). A multicriteria heuristic method to improve resource allocation in multiproject scheduling. *European Journal of Operational Research, 127*(2), 408–424.

Maffioli, F. (1987). Randomized heuristic for NP-hard problem. In G. Andreatta, F. Mason, & P. Serafini (Eds.), *Advanced School on Stochastic in Combinatorial Optimization* (pp. 760–793). Singapore: World Scientific.

Nozick, L. K., Turnquist, M. A., & XU, N. (2004). Managing portfolios of projects under uncertainty. *Annals of Operation Research, 132*(1–4), 243–256.

Osorio, C., & Bierlaire, M. (2009). An analytic finite capacity queueing network model capturing the propagation of congestion and blocking. *European Journal of Operational Research, 196*(3), 996–1007.

Pritsker, A. A. B., Watters, L. J., & Wolfe, P. M. (1969). Multiproject scheduling with limited resources: a zero–one programming approach. *Management Science, 16*(1), 93–108.

Serafini, P. (1985). *Mathematics of multiobjective optimization*: 289. Berlin: Springer Verlag CISM courses and lectures.

Siegel, S. (1956). *Non-parametric statistics for the behavioral sciences* (pp. 75–83). New York: McGraw-Hill.

Suman, B. (2002). Multiobjective simulated annealing—a metaheuristic technique for multiobjective optimization of a constrained problem. *Foundations of Computing and Decision Sciences, 27*(3), 171–191.

Suman, B. (2004). Study of simulated annealing based algorithm for multiobjective optimization of a constrained problem. *Computers and Chemical Engineering, 28*(9), 1849–1871.

Suman, B., & Kumar, P. (2006). A survey of simulated annealing as a tool for single and multiobjective optimization. *Journal of the Operational Research Society, 57*(10), 1143–1160.

Suppapitnarm, A., Seffen, K. A., Parks, G. T., & Clarkson, P. J. (2000). A simulated annealing algorithm for multiobjective optimization. *Engineering Optimization, 33*(1), 59–85.

Tan, B., & Gershwin, S. B. (2009). Analysis of a general Markovian two-stage continuous-flow production system with a finite buffer. *International Journal of Production Economics, 120*(2), 327–339.

Tsubakitani, S., & Deckro, R. F. (1990). A heuristic for multi-project scheduling with limited resources in the housing industry. *European Journal of Operational Research, 49*(1), 80–91.

Tuyttens, D., Teghem, J., Fortemps, P. H., & Nieuwenhuyze, K. V. (2000). Performance of the MOSA method for the bicriteria assignment problem. *Journal of Heuristics, 6*(3), 295–310.

Ululgu, L. E., & Teghem, J. (1994). Multiobjective combinatorial optimization problems: A survey. *Journal of Multicriteria Decision Analysis, 3*(2), 83–104.

Ulungu, L. E., Teghem, J., & Ost, C. (1998). Efficiency of interactive multi-objective simulated annealing through a case study. *Journal of the Operational Research Society, 49*(10), 1044–1050.

Van Laarhoven, P. J. M., & Aarts, E. H. L. (1987). *Simulated annealing: theory and practice*. Dordrecht: Kluwer Academic Publishers.

Wiest, J. D. (1967). A heuristic model for scheduling large projects with limited resources. *Management Science, 13*(6), 359–377.

Yaghoubi, S., Noori, S., Azaron, A., & Tavakkoli-Moghaddam, R. (2011). Resource allocation in dynamic PERT networks with finite capacity. *European Journal of Operational Research, 215*(3), 670–678.

Yaghoubi, S. (2012). *Resource allocation in a multi-project system under stochastic and dynamic conditions* (Ph.D. thesis). Tehran, Iran: Iran University of Science & Technology.

Ying, Y., Shou, Y., & Li, M. (2009). Hybrid genetic algorithm for resource constrained multi-project scheduling problem. *Journal of Zhejiang University (Engineering Science), 43*(1), 23–27.