A BI-OBJECTIVE GENETIC ALGORITHM APPROACH TO RISK MITIGATION IN PROJECT SCHEDULING

Murat Kılıç[1], Gündüz Ulusoy[2], and Funda Sivrikaya Şerifoğlu[3]

[1,2]Sabancı University, Istanbul
mkilic@su.sabanciuniv.edu gunduz@sabanciuniv.edu

[3]Abant Izzet Baysal University, Bolu
serifoglu_f@ibu.edu.tr

Abstract

A problem of risk mitigation in project scheduling is formulated as a bi-objective optimization problem, where the expected makespan and the expected total cost are both to be minimized. The expected total cost is the sum of four cost components: overhead cost, activity execution cost, cost of reducing risks and penalty cost for tardiness. Risks for activities are predefined. For each risk at an activity, various levels are defined, which correspond to the results of different preventive measures. Only those risks with a probable impact on the duration of the related activity are considered here. Impacts of risks are not only accounted for through the expected makespan but are also translated into cost and thus have an impact on the expected total cost. An MIP model and a heuristic solution approach based on genetic algorithms (GAs) is proposed. The experiments conducted indicate that GAs provide a fast and effective solution approach to the problem. For smaller problems, the results obtained by the GA are very good. For larger problems, there is room for improvement.

Keywords: *project scheduling, risk, multiobjective optimization, genetic algorithms*

Appeared in *International Journal of Production Economics*, 112, 202-216, 2008.

1. INTRODUCTION

In this paper, a problem of risk mitigation in project scheduling is considered. A mixed integer programming model and a heuristic solution approach based on genetic algorithms (GAs) is proposed to solve the problem. The problem is formulated as a bi-objective optimization problem, where the expected makespan and the expected total cost are the two objectives both to be minimized.

There are various objectives employed in the project scheduling literature. Minimization of the makespan is the most common objective. Cash flows in the form of expenses and payments are taken into account in the problem formulations aiming at the maximization of the net present value of cash flows. When the aim is to minimize costs resulting from the realization of an activity or resource usage, the objective of cost minimization is employed. Consideration of earliness and tardiness costs is relatively more recent (e.g., Kogan and Shtub, 1999). Incorporation of quality considerations into the project scheduling problem is also relatively new (Erengüç and Içmeli, 1999; Içmeli and Rom, 1997). The problem with the objective of quality maximization is the difficulty in defining quality quantitatively in such a way that different stakeholders agree on the resulting definition.

Research on project scheduling incorporating two or more objectives is rather sparse, although project scheduling is inherently multiobjective: Project managers in real life deal with several objectives at once. This scarcity can be easily concluded from the surveys on project scheduling (Içmeli *et al*., 1993; Özdamar and Ulusoy, 1994; Kolisch and Padman, 2001).

There are some earlier efforts to develop decision support for the scheduling of project activities to attain multiple objectives related to project duration and cost (Davis

et al., 1992; Rys *et al*., 1994; Slowinski *et al*., 1994; Ulusoy and Özdamar, 1996; Hapke *et al*., 1998). Slowinski (1989) provides a review on multiobjective project scheduling under multiple-category resource constraints. An application is provided by Stewart (1991) who develops a multicriteria decision support system for the R&D project selection.

Viana and de Sousa (2000) investigate the applicability of two metaheuristic approaches, Pareto simulated annealing and multiobjective tabu search, to the resource constrained project scheduling problem, in order to minimize the makespan, the weighted lateness of activities, and the violation of resource constraints. A recent study by Hanne and Nickel (2005) on an evolutionary algorithm for scheduling and inspection planning in a software development project tries to minimize three objectives related to quality, time and cost. Quality of a project is measured by the number of defects; time by the project duration and cost by the total cost of time spent by the development team members are tried to be minimized.

In another recent study, Al-Fawzan and Haouari (2005) address the issue of designing a project schedule which is not only short in time, but also less vulnerable to disruptions due to reworks and other undesirable conditions. Based on the concept of schedule robustness, they develop a bi-objective resource-constrained project scheduling model where the two objectives are the robustness maximization and makespan minimization. A tabu search algorithm is used to generate an approximate set of efficient solutions. Another bi-objective resource-constrained project scheduling problem with robustness and makespan criteria is investigated by Abbasi *et al.* (2006) via a simulated annealing algorithm.

Risk is defined by the Project Management Institute (PMI) as an uncertain event or condition that, if it occurs, has a positive or negative effect on a project objective

(2000). In this study, risks with negative impacts will be considered. For example, in an R&D environment, the breakdown of laboratory equipment, a delay in material arrival, or the loss of a key research personnel might negatively affect project objectives. The authors are not aware of any study in project scheduling involving positive risks. In fact, project scheduling under risk has not been studied extensively in the literature relative to the problem of project scheduling under deterministic conditions (see, e.g., İçmeli *et al.*, 1993; Kolisch and Padman, 2001; Özdamar and Ulusoy, 1994; Weglarz, 1999; Chapman and Ward, 1999). In a more recent review on project scheduling under uncertainty, Herroelen and Leus (2005) also state that the literature on project scheduling under risk and uncertainty is rather sparse.

Recognition of the fact that research results on problems based on certainty assumptions and static environments have only slim chances of implementation has recently led to an increasing interest on project management under uncertainty (Elmaghraby, 2005). Different approaches like reactive scheduling, stochastic scheduling, scheduling under fuzziness, proactive (robust) scheduling, and sensitivity analysis are employed to deal with uncertainty. For a detailed discussion, see the review by Herroelen and Leus (2005).

Most of the research approaches on project scheduling involving risk do not model risks explicitly, but try to evaluate the risk of schedule and/or budget overruns using stochastic models for activity durations and/or costs. Tavares (1994) tried to assess the financial risk of a project using a stochastic model considering stochastic nature of activity durations and project expenditures. Tavares *et al*. (1998) introduced a model considering randomness of both the cost and duration of each activity and study the problem of project scheduling in terms of a project's discounted cost and of the risk of not meeting its completion time.

Jaafari (2001) provides a general approach to risk management within project management. He conjectures that a paradigm shift is needed and introduces a strategy-based project management approach, which is called the life cycle project management. This is an integrated and collaborative framework, which installs life cycle objective functions as the basis of evaluation and decision-making throughout project life cycle.

An approach similar to what is presented in this paper for modelling risk considerations in project scheduling is provided by Zafra-Cabeza *et al*. (2004). The researchers model the scheduling of tasks of a project by using a special kind of Petri net, the so-called *p*-timed Petri net. They develop a mixed integer programming model for minimizing a weighted sum of the cost and execution time (makespan) of the project under risk considerations. Risks are identified with occurence probabilities and initial impacts. Mitigation actions may be taken to reduce the consequences of risks. The researchers illustrate their technique on a research project on the assessment of energy saving in oil pipelines, which comprises seven activities including dummy start and end activities, six identified risks, and six mitigation activities.

The outline of this paper is as follows: The problem is defined in detail in Section 2. In Section 3, the mathematical model and in Section 4 the solution approach are presented. Computational results are provided in Section 5 and concluding remarks in Section 6.

2. PROBLEM DEFINITION

In the problem considered here, the decision maker has two objectives both of which are to be minimized: the expected makespan and the expected total cost. The resources utilized in the project have no constraints imposed on them.

A set of activities $\{1,..,J\}$ associated with a single project is given. A set of predefined risks $\{1,..,N_j\}$ is associated with each activity j of the project. These risks represent the events that may occur during the execution of activities. Only those risks with a probable impact on the duration of the related activity are considered here. Thus, the probable impacts of these risks are not only accounted for through the expected makespan but are also translated into cost and therefore have an impact on the expected total cost as well. Hence, the problem defined here is not a pure time-risk problem but a more general one as represented by the two objectives stated.

Insert Figure 1 about here

A project manager can decrease the probability of occurrence and impact of each risk by taking some preventive measures at an estimated cost. The levels of preventive measures taken are modelled by a set of states $\{1,..,K_{jn}\}$ for each risk n associated with each activity j. State 1 corresponds to the do-nothing case of taking no preventive measures against the associated risk. The cost associated with this state is therefore zero. States with increasing indices correspond to increasing levels of preventive measures and hence to decreasing levels of occurrence probabilities and impacts of risks. The hierarchy of the relationship between the project, activities, risks, and risk states is displayed in Figure 1.

If preventive measures associated with the state k of a risk n associated with an activity j are taken, then the risk occurs with a probability of p_{jnk}. The impact is taken to be only in the direction of an expansion of the activity duration by a given factor, which is denoted by I_{jnk}. It is assumed that the preventive measures, when taken, will pay-off, and they will produce the expected results. It is also assumed that:

i. The risks are independent and their impacts are additive at the activity level.

ii. All the risks associated with an activity are identified.

iii. They are static throughout the project life.

Table 1 illustrates an activity, activity *X*, for which there is only one risk involved with three states. TU and MU stand for time unit and monetary unit, respectively. The duration of the activity with no risks involved is 20 time units. At state 1, no preventive measures are taken and thus no cost is incurred. As a result, the expected duration of activity *X* becomes d'_x= 20+0.7*0.5*20=27 TU. If necessary measures are taken to reduce the risk level from state 1 to 2, the expected duration of the activity drops down to d'_x= 20+0.6*0.5*20=26 TU. The cost of taking necessary measures to reduce the risk level from state 1 to 2 is 150 MU.

Insert Table 1 about here

The cost of preventive measures against risks is only one cost component considered in the problem formulation. The cost function of the model covers three other costs that may occur during the execution of the project. These are the overhead cost, the activity execution cost, and the penalty cost for tardiness and they are all assumed to be linear in project duration, activity duration and tardiness, respectively. The activity execution cost is the sum of the costs of the different resources being employed throughout the execution of the activity.

3. A MATHEMATICAL MODEL

The problem is represented on an activity-on-node (AON) network with a single starting and a single ending node both corresponding to dummy activities.

Notation:

j: index for activities, j=1,...,J;

l: index for resource types, l =1,…,L;

$\{P_j\}$: Set of immediate predecessors of activity j;

$\{N_j\}$: Set of risks associated with activity j;

d_j: Duration of activity j with no risks involved;

d_j': Expected duration of activity j under associated risks;

C_p: Unit penalty cost of being tardy;

C_o: Unit cost of overhead;

C_l: Unit cost of resource type l;

C_{jnk}: Cost of reducing the risk level from state 1 to state k for risk n at activity j, where $C_{jn1}=0$ for $j=1,...,J$ *and* $n=1,..,N_j$

K_{jn}: Number of states for risk n at activity j;

p_{jnk}: Probability of the occurrence of risk n for activity j at state k;

I_{jnk}: Impact of risk n, if it occurs, on activity j at state k;

W_{lj}: Number of units of resource type l assigned to activity j;

T_{plan}: Due date set for the project;

The decision variables are as follows:

T_j: Starting time of activity j;

y: Expected tardiness of the project;

$$X_{jnk} = \begin{cases} 1, & \text{if the } k^{th} \text{ state is chosen for } n^{th} \text{ risk of } j^{th} \text{ activity} \\ 0, & \text{otherwise} \end{cases};$$

The mathematical programming model is given by expressions (1) through (8) below, where $E(TC)$ stands for the expected total cost and $E(C_{max})$ stands for the expected makespan of the project.

$$MinE(TC) = y * C_p + \sum_{j=1}^{J} \sum_{n=1}^{N_j} \sum_{k=1}^{K_{jn}} C_{jnk} * X_{jnk} + C_0 * T_J + \sum_{j=1}^{J} \sum_{l=1}^{L} W_{lj} * d_j' * C_l \quad (1)$$

$$MinE(C_{\max}) = T_J \quad (2)$$

subject to:

$$T_1 = 0 \quad (3)$$

$$T_i + d_i' \le T_j \qquad i \in P_j;\ j = 2,...,J \quad (4)$$

$$d'_j = d_j + d_j * \sum_{n=1}^{N_j} \sum_{k=1}^{K_{jn}} X_{jnk} * I_{jnk} * p_{jnk} \qquad j = 1,...,J \tag{5}$$

$$\sum_{k=1}^{K_{jn}} X_{jnk} = 1 \qquad j = 1,...,J; n = 1,...,N_j \tag{6}$$

$$y = \begin{cases} T_J - T_{plan}, & \text{if } T_J > T_{plan} \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

$$X_{jnk} \in \{0,1\} \qquad j = 1,...,J;\ n = 1,...,N_j;\ k = 1,...,K_{jn} \tag{8}$$

The model aims to minimize objective (1) [the expected total cost] and objective (2) [the expected makespan of the project]. These two objectives are conflicting. The expected total cost is represented as the sum of four cost components: the penalty cost for tardiness, the cost of risk reductions, the overhead cost, and the activity execution cost of the project.

Equations (3) and (4) determine the starting times of the activities. Equation (5) is used to calculate the expected duration of an activity by adding the additional risk related duration extensions to the normal activity duration. Equation (6) assures that one and only one state for each one of the risks is selected.

There are ($\sum_{j=1}^{J} \sum_{n=1}^{N_j} \sum_{k=1}^{K_{jn}} 1$) number of 0-1 decision variables in the model depending on the number of risks and number of the states related. Problems of small size can be solved easily by a mathematical programming solver. But for larger problems, computational costs become prohibitive as the search space becomes very large very quickly. For problem sets with 15, 25 and 35 activities used in this study, the average number of possible solutions is on the order of 10^{15}, 10^{28} and 10^{39} respectively. Therefore, a heuristic approach is proposed to solve the problem.

4. SOLUTION APPROACH

In multiobjective mathematical programming, the solution techniques are classified according to the three pure approaches of articulation of the decision maker's preference structure: prior, progressive, and a posteriori articulation of preferences (Hwang and Masud, 1979; Hwang *et al*., 1980; Van Veldhuizen and Lamont, 2000). In multiobjective optimization with prior preference articulation, the objectives are combined into a scalar function prior to optimization, which effectively transfers the problem into a single objective optimization problem. Survey of techniques for that purpose can be found, for example, in Keeney and Raiffa (1976). In the second approach, preference articulation is done progressively so that the decision-making and optimization processes are intertwined. Optimization is performed upon partial preference information provided by the decision maker. The "updated" set of solutions are submitted to the decision maker, who finetunes his/her preferences based on these solutions. An early example of progressive articulation is provided by Zionts and Wallenius (1976). Fonseca and Fleming (1998) present a multiobjective genetic algorithm (GA) approach to be employed together with progressive articulation. In a posteriori articulation of preferences, the decision maker is presented with a set of Pareto optimal candidate solutions, from which s/he chooses (see, e.g., Hwang and Masud, 1979). A posteriori preference articulation is a developing topic also in the literature on the multiobjective metaheuristics (Jones *et al.*, 2002).

In this paper, a posteriori preference articulation approach is employed, and a multiobjective GA solution procedure is developed to generate Pareto optimal solutions to the problem defined in Section 2. The reasons for this choice are that preference articulation is not required for the testing of the methodology developed and the

approximation of the true Pareto front provides a natural testing ground even if for relatively small problems.

Evolutionary algorithms are increasingly used in multiobjective optimization (Deb, 2001; Abraham *et al*., 2005; Tan *et al*., 2005). GAs, in particular, constitute a popular solution procedure for multiobjective optimization problems (Fonseca and Fleming, 1995; Jaskiewicz, 2002). Approximately 70 per cent of the metaheuristic approaches suggested and published between 1991 and 2000 are GAs (Jones *et al*., 2002). Since GAs use parallel search techniques and multiobjective optimization problems have several nondominated solutions, this problem class and the solution procedure make a good match. A recent survey of GA approaches to multiobjective optimization problems is provided by Coello *et al*. (2002).

In this paper, a GA approach is developed to solve the bi-objective project scheduling problem modelled in the previous section. Two improvement heuristics are proposed for further improving the solutions found by the GA. They try to decrease the expected total cost while keeping the critical path fixed.

4.1. The Genetic Algorithm Approach

In the GA approach proposed here, a direct representation is used for encoding a solution. Each gene on the chromosome corresponds to a risk. The number in the gene represents the state that will be chosen for the corresponding risk. Three cells added to the end of the chromosome are used to display the expected makespan, the expected total cost and the fitness value. These cells do not undergo any evolutionary process and are used only for information storage purposes. The chromosome representation is depicted in Figure 2.

Insert Figure 2 about here

The selection mechanism is the well-known roulette wheel selection, which selects chromosomes based on their relative fitness in the current population. One point crossover is used to generate two offspring from two parent chromosomes: The parts of the parent chromosomes on the two sides of a randomly chosen cut point are exchanged to generate the offspring. The mutation operator is bit mutation, which replaces the value on a randomly chosen gene of the chromosome with another randomly generated value.

The first population is generated randomly. For generating the subsequent populations, crossover, mutation and reproduction operators are applied in a parallel fashion contrary to the serial application in traditional GAs. First an operator is chosen: crossover with a probability of P_c, mutation with a probability of P_m and reproduction with a probability of $(1-(P_c+P_m))$. If the chosen operator is the crossover, then two chromosomes are chosen from the population by the roulette wheel selection mechanism. If the chosen operator is the mutation operator or the reproduction operator, then a single chromosome is chosen. Finally, the chosen operator is applied to the chosen chromosome(s). The reproduction operator simply generates a copy of the chosen chromosome.

The fitness of a chromosome is computed as the product of two numbers as given in Equation (9). In that equation, *NNR* denotes the nearest neighbourhood radius. As represented in Equation (10), it is the ratio of the distance between the nearest neighbour and the chromosome in question ($d_{nearest}$) to the maximum such distance in the population (d_{maxpop}). The distance is measured as the Euclidean distance in the phenotypic space. It is assumed that if an individual is close to its nearest neighbour, it is in a crowded region. Both *NNR* and the fitness value decrease with decreasing

distance to nearest neighbour, $d_{nearest}$. *NNR* thus plays the role of a sharing function. In multiobjective GA applications, a sharing function is employed to avoid crowding of individuals around one of several optima. A sharing function determines for a given chromosome the sharing values of all other individuals in the current population (Goldberg and Richardson, 1987). The function is a decreasing function of the distance between any two individuals, so that individuals close to the chromosome at hand attain a higher sharing value, while those farther apart attain lower values. The fitness of the chromosome is computed by dividing the 'raw' fitness value by the sum of all sharing values.

N_{dom} in Equation (11) represents the number of individuals (chromosomes) dominated by the current individual in a Pareto domination tournament involving the entire population. In other words, the current individual is compared to every other individual in the population, and N_{dom} gives the number of individuals dominated by the current one. N_{pop} is the population size. Both N_{dom} and N_{pop} are increased by one to ensure a positive value for R_{dom}. Thus, R_{dom} represents a measure of the fitness in the domination sense.

By multiplying R_{dom} and *NNR,* the fitness in the above sense and the sharing concept are combined. Thus, the fitness function is a new resolution combining the sharing mechanism and the dominance in the Pareto sense.

$$\text{Fitness} = NNR * R_{dom} \tag{9}$$

$$NNR = d_{nearest} / d_{maxpop} \tag{10}$$

$$R_{dom} = (N_{dom} + 1) / (N_{pop} + 1) \tag{11}$$

Elitism is applied by transferring all of the nondominated individuals in each population into the next one.

4.2. Improvement Heuristics

For some of the GA solutions further improvement might be possible for decreasing the expected total cost, while keeping the critical path fixed. Therefore, improvement heuristics are developed to apply to the solutions obtained by the GA. Such heuristics are needed primarily to avoid trivially inferior solutions. The aim is to prevent investing more resources than needed to reduce the risks of the non-critical activities by choosing less costly risk states for non-critical activities (i.e. by taking less precautions against the risks associated with them), while not changing the chosen risk levels of the activities on the critical path and hence the makespan. Thus, when relaxing the duration of the non-critical activities, a change in the makespan provided by the GA solution is not allowed.

To this end, a multi-mode project scheduling problem is formulated. Multi-mode project scheduling problem includes time/cost trade-offs in the form of multiple modes for executing the project activities. The resource-constrained version involves also time/resource and resource/resource trade-offs (Herroelen *et al*. 1998; Elmaghraby, 1977). Here, the problem is formulated by assigning every possible combination of the states of the risks associated with an activity to different modes. For example, for an activity with two risks having three states each, there will be nine modes. The duration associated with a mode represents the expected duration, which will be realized when the corresponding risk states are chosen. The cost represents the sum of the expected activity execution cost and the risk reduction costs, which constitute a local trade-off with the expected duration. A domination search over the modes is performed to eliminate the dominated modes. The resulting problem is a discrete time-cost trade-off problem. Table 2 and Table 3 illustrate the formulation of modes and the domination search over them for an example activity, say activity *Y*, for which there are two risks

with two states each. As shown in Table 3, only two out of four possible modes are non-dominated, and these are to be used in subsequent operations.

Insert Table 2 about here

Insert Table 3 about here

For solving the resulting discrete time-cost trade-off problem; the makespan of the solution obtained by GA is taken as a limit on the expected project duration and the modes of activities on the critical path are fixed, i.e., the critical activities are assumed to have only one mode and that as assigned by the GA solution. The expected cost is reduced, if feasible, by increasing the duration of non-critical activities (i.e. by choosing smaller numbered modes for noncritical activities), while preserving the critical path and makespan.

This problem is a special case of the discrete time-cost trade-off problem, which is shown to be NP hard by De *et al.* (1997). The researchers have shown that under a due date constraint, multi-mode project scheduling problem with the cost minimization as the objective is an NP hard problem. Exact solution approaches provided by Demeulemeester *et al*. (1996) for this problem may become computationally very costly. Therefore, two heuristic approaches are proposed.

In these approaches, a starting point solution is generated, which is then subjected to an improvement routine. The improvement routine is common to both of the approaches.

4.2.1. A heuristic based on a continuous cost-versus-duration model (CCDMH)

In this heuristic, first, a monotonically decreasing polynomial is fit to the time/cost scatter of the non-dominated modes for an activity resulting in a continuous time-cost curve. The problem is thus transformed to a continuous project compression problem, which is easier to solve. In the application here, all possible fits (linear, polynomial,

logarithmic, power, and exponential) are tried, and the best fit, i.e. the one with the largest R^2 value, is chosen. If the best fit is linear, that linear approximation is used in subsequent operations. If the best fit is not linear, a piecewise linear underestimator is applied to the nonlinear fit. For that purpose, one of the methods proposed by Wei and Wang (2003) is employed.

In the method, Wei and Wang propose to use tangents to the continuous curve. The first line is drawn tangent to the curve at the starting point and the last line is drawn tangent at the ending point. The other lines are drawn tangent at points located equidistant along the x-axis between the starting and ending points. The intersection points of the tangent lines constitute the starting and ending points of segments (Figure 3). Obviously, as the number of segments increases, the precision of the piecewise linear approximation increases. To decide on this number, Wei and Wang compare the length of the curve and the piecewise linear underestimator and increase the number of segments until the ratio of their absolute difference divided by the length of the curve becomes smaller than some predetermined bound ε.

Insert Figure 3 about here

This model is also valid for situations, where the mode scatter can best be represented as a single line segment (Figure 4).

Insert Figure 4 about here

The continuous form is represented by the following mathematical model for the case, where the time-cost curves associated with the individual activities can be approximated by piecewise linear segments.

Notation:

$\{K\}$: Set of critical activities k (subset of J);

$\{U\}$: Set of non-critical activities u (subset of J);

p_k: Duration of the activities on the critical path;

s_u^m: Slope of the m^{th} segment of the time-cost curve for non-critical activities (note increasing negative slope as m increases in Figure 3);

d'_u: Duration of the non-critical activity u;

a_u^m: The endpoint of m^{th} segment of the time-cost curve, smallest duration of segment;

b_u^m: The endpoint of m^{th} segment of the time-cost curve, largest duration of segment;

M_u: The number of segments of the time-cost curve for activity u;

X_u^m: Duration on segment m of non-critical activity u;

CPL: Critical path length;

Model:

$$Min \sum_{u \in U} \sum_{m=1}^{M_u} X_u^m * s_u^m \tag{12}$$

subject to:

$$T_1 = 0 \tag{13}$$

$$T_i + d_i' \le T_j \qquad i \in P_j;\ j = 2,...,J \tag{14}$$

$$d'_k = p_k \qquad \text{for } k \in K \tag{15}$$

$$0 \le X_u^m \le (b_u^m - a_u^m) \qquad \text{for } u \in U;\ m = 1,...,M_u \tag{16}$$

$$d'_u = a_u^1 + \sum_{m=1}^{M_u} X_u^m \qquad \text{for } u \in U \tag{17}$$

$$T_J \le CPL \tag{18}$$

GAMS© is employed to solve the model. Using the durations assigned by the GAMS© solution, the modes of the non-critical activities are determined. For each non-critical activity, if the duration assigned corresponds to the duration of a non-dominated mode, then that mode is assigned to the activity. Otherwise, the non-dominated mode with the closest but smaller duration is assigned.

The starting point solution thus found is subjected to an improvement routine explained in Section 4.2.3.

4.2.2. A heuristic based on GA results (GABH)

In the heuristic, which is based on GA results (GABH), solutions provided by the GA are used to generate a starting point solution. The GA may result with a solution, which assigns dominated modes to non-critical activities. When this is the case, the non-dominated mode with a lower duration is found, and this mode is assigned to the corresponding activity. The starting point solution thus found is subjected to the improvement routine explained in the next subsection.

4.2.3. The improvement algorithm

The starting point solution obtained by either one of the two methods explained above is consequently subjected to the improvement routine. For each non-critical activity, slacks and the so-called saving per duration value are computed. A specific amount of saving will result, if the activity is performed at its next higher duration mode. The saving per duration ratio is the ratio of the expected cost decrease to expected duration increase between the respective non-dominated modes of the activity. For the example activity illustrated in Table 3, the saving per duration ratio (from mode 4 to mode 2) is 18.37 (=(5178.4-5034.6)/(33.39-25.56)).

Starting with the non-critical activity with the highest saving per duration value, the durations are expanded without violating the slacks until slacks diminish to zero or there is no further mode to expand to. Figure 5 lists the steps of the algorithm.

5. COMPUTATIONAL STUDY

To use as test cases, projects with 15, 25 and 35 activities are generated. Networks for these projects containing one starting and one ending node are generated randomly. For the sake of simplicity, labor is taken as the only resource needed to execute an activity and the unit cost of labor is taken to be the same for all activities. For each of the activities; the duration, the number of labor units, the number of risks, and for each risk, the number of states are chosen randomly from the integer valued sets {10,...,20}, {4,5,6,7}, {1, 2, 3}, {2, 3, 4}, respectively. There are three parameters related to states: the probability of occurrence p_{jnk}, the impact I_{jnk} and the cost of reducing the risk level C_{jnk}. These three parameters are selected such that the product of p_{jnk} and I_{jnk} is decreasing in C_{jnk}.

For each problem, unit penalty cost and unit overhead costs are taken constant over the project duration. The project deadline, T_{plan}, is chosen as a point between the minimum expected project duration and the maximum expected project duration. $State_{1jn}$ is employed for all risks and activities, when obtaining the maximum expected project duration; and $State_{Kjn}$ is employed for all risks and activities, when obtaining the minimum expected project duration. T_{plan} is determined by adding 20% of the gap between the maximum the minimum expected project durations to the minimum expected project duration.

In order to provide a base for performance comparison of the different algorithms proposed, a performance metric is developed based on the idea of hyperarea ratio (Knowles and Corne, 2002). In hyperarea ratio, the hyperarea formed by the solutions provided by an algorithm is divided to the hyperarea of the true Pareto front. This metric is a subjective but a good measure to compare the performances of algorithms on problems, for which true Pareto fronts are known. Since this is not the case here, a different metric is developed instead. The new metric, "Extreme Hyperarea Ratio" (EHR) is defined as the ratio of the hyperarea of the front (area H in Figure 6(a)) to the area bounded by the origin and the so-called reference point defined by the maximum values of the two objective functions (area A in Figure 6(b)) (Kılıç, 2003). EHR is used to compare GAs with different parameter settings in the fine-tuning process and to compare performances of different heuristic algorithms on test problems.

Insert Figure 6 about here

5.1. Fine-tuning of the Genetic Algorithm

The values of four GA parameters, namely the population size, the number of generations, and the probabilities of crossover and mutation, are fine-tuned over a range of values. The population size and the number of generations are related in such a way that their product gives 50,000. The pairs of values for the population size and the number of generations used in the experiments are as follows: (100,500), (200,250), (250,200), (500,100).

The values for the probabilities of crossover and mutation are chosen from among the set {0.15, 0.30, 0.45, 0.60, 0.75} in such a way that their sum does not exceed 1.00. This pattern gives rise to 15 possible value pairs for these two probabilities. Combined with the four possible value pairs for the population size and the number of generations,

there are 60 sets of parameter value combinations to be tried in the fine-tuning experiments. Therefore, 60 experiments on every problem (in three sets of five problems each) are conducted, i.e., a total of 900 experiments. The three sets of problems each contain five projects with 15, 25 and 35 activities, respectively.

The results of GA runs employing different parameter value combinations are compared on the basis of EHR values using one-way ANOVA in Systat©. The tests indicate that the results are not statistically significant at a significance level of 0.05. Thus, the set of parameter values, for which the average EHR value is best, is used in the GA. The probabilities of crossover and mutation are thus determined to be 0.15 and 0.75, respectively. The population size is taken to be 200 and the number of generations 250.

The fine-tuning experiments have shown that the bound of 50,000 chromosomes for GA runs may not be sufficient for larger problems. As the size of the search space increases, further exploration is needed. Therefore, the number of generations is increased with increasing problem size (Table 4). It should be noted that the product of the population size and the number of generations does not actually give the exact number of chromosomes generated during the evolution process. An elitist strategy is employed such that non-dominated chromosomes in any generation are transferred as they are into the next generation without being subjected to any changes. Therefore, depending on the problem, the number of individuals generated and evaluated is nearly 20% less than the number indicated by the product of the population size and the number of generations given in Table 4.

Insert Table 4 about here

5.2. Comparison of the GA with the Approximation of the True Pareto Front

To provide a comparison base, the true Pareto front is approximated by employing GAMS©. The makespan objective is removed from the objectives set and added to the constraint set to obtain a single objective cost minimization model. The resulting model is solved repeatedly by Cplex solver embedded in GAMS©. The upper bound on the makespan in the initial model formulation is the maximum makespan. The maximum makespan is found by solving the project scheduling problem with risk states in all activities set to their State 1 values. In a series of runs, the limit on the makespan resulting from the last GAMS© run is decreased by increments of 0.01 until the minimum makespan is reached.

In order to compare the proposed GA with the approximation of the true Pareto front, 20 problems are solved for each problem size: 15-activity, 25-activity and 35-activity problems. For each problem, the non-dominated solutions resulting from the proposed GA application are plotted together with the approximate Pareto front. Figures 7, 8 and 9 illustrate a sample of three such plots with problem sizes being 15-activity, 25-activity and 35-activity problems, respectively.

The plot in Figure 8 shows that some results provided by the GA may be better than the ones provided by GAMS©. This may have occurred because of the fact that GAMS© stops the search within some tolerance limit before reaching the optimum. Also the decrements of 0.01 for the makespan constraint may not be small enough allowing some solutions to skip.

Insert Figure 7 about here
Insert Figure 8 about here
Insert Figure 9 about here

The plot in Figure 9 illustrates results for another problem, where the solutions in the approximation of the Pareto front dominate the ones obtained by the GA. As Figures 7 to 9 also illustrate, the result of the experiments indicate that the performance of the GA is very good for small problems with 15 activities. For larger problems with 35 activities, the deviations are larger, and there is room for improvement.

An analysis of the EHR values for the approximation of the Pareto front and for the GA is also performed. The average EHR value for the Pareto front approximation is 0.2573. This value is close to the average EHR value for the GA, which is 0.2404. But they statistically significantly differ at a significance level of 0.05. In Table 5, average percent deviations of EHR values for the GA from the EHR values for the approximation of the true Pareto front are given for each problem set consisting of 20 problems. The deviations are relatively small for 15- and 25-activity problems but increase substantially for 35-activity problems.

Insert Table 5 about here

5.3. Comparison of the Improvement Heuristics

As the comparison of GA solutions to the approximation of the Pareto front also indicates, there is room for improving GA's performance especially on larger problems. Improvement heuristics CCDMH and GABH are applied to the solutions found by the GA. To provide a comparison basis, two criteria are used: Average value of the improvement (i.e. the decrease in expected cost) and the ratio of the number of improved solutions to the number of non-dominated solutions found. Average value of improvement (AI) is expressed in percentage as in Equation (19) below, where l is the number of Pareto optimal solutions, C_i^{GA} is the expected cost of i^{th} Pareto optimal

solution in which GA resulted, C_i^H is the expected cost after the improvement heuristic is applied to solution *i*.

$$AI = (\sum_{i=1}^{l} \frac{C_i^{GA} - C_i^H}{C_i^{GA}}) * 100 / l \tag{19}$$

The improvement heuristic CCDMH improves a significant portion of the solutions. But, since an exact linear under-estimator to the modes of the problem has not been used, some of the expected cost values increase instead of decreasing. Table 6 shows statistics related to the performance of CCDMH.

Insert Table 6 about here

The results show that as the problem size increases the performance of CCDMH improves. This may be a result of the deteriorating performance of the GA with increasing problem size. Since GA cannot explore the search space adequately, there remains more to do for the improvement heuristic.

Table 7 represents the performance of the improvement heuristic GABH. GABH improves a larger percentage of solutions than the improvement heuristic CCDMH but does not decrease the expected cost of the solutions as much as CCDMH does. GABH starts from a GA solution, which may be a local optimum. Hence it might not be able to move away from this local optimum.

Insert Table 7 about here

It might be conjectured that CCDMH may become more effective, if a more precise piecewise linear underestimator is used. But as the precision of the estimator increases, the effort to generate the underestimator and to solve the continuous model will increase. This leads to a trade-off to be resolved.

Results given in Table 6 and Table 7 indicate that the average percentages of improvements in expected cost values are low. But this should be less of a concern, as

the primary mission of the improvement heuristics is to avoid a trivially inferior solution.

An analysis into the cost breakdown of the resulting improved GA solutions might provide further insights. Since a change in the makespan of the project is not allowed during the execution of the improvement heuristics, the overhead and penalty costs remain the same. Improvement heuristics deal with the activity based (local) costs, which are the cost of preventive measures and the labor cost. As it can be observed in Table 8, these cost components are quite close to each other. This closeness also helps to understand the poor performance of improvement heuristics. For another problem set, in which the cost structure is different, the performance of the improvement heuristics may be different.

Insert Table 8 about here

5.4. Computational Times

The computational times in CPU milliseconds for the GA, for the improvement heuristic CCDMH and for the generation of the Pareto front approximation (PFA) are given in Table 9. These values are the average of computational times on five problems each from three problem classes with 15-, 25- and 35-activities, respectively. The computational times for the improvement heuristic GABH are not given, since they are too small to be measured accurately. The program code is compiled using Visual Studio's C/C++ compiler, and the runs are performed on a Celeron 800 with 128 MB RAM.

As it is clearly seen from Table 9, the computational times of PFA are very high compared to the computational times for the GA. PFA is computationally costly even for the relatively small problems. For larger problems, it may become very costly to

generate the Pareto front approximation because of excessive computational times and limitations of mathematical programming software.

6. CONCLUSIONS AND FUTURE RESEARCH TOPICS

In this paper, the problem of project scheduling under risk has been addressed. The experiments conducted indicate that GAs provide a fast and effective solution approach to the problem. For smaller problems, the results obtained by the GA are very good. For larger problems, there is room for improvement.

The problem formulation provided in this paper can easily be revised to include also risks with positive effects. The states for risks with positive effects would be modelled such that

- $I_{jnk}<0$
- State k, where $k>1$, is associated with increasing probabilities of occurrence and/or absolute impacts.

No revision is needed in the mathematical model or in the solution approach.

There are several extensions and variations to be investigated as future research work. Other metaheuristic approaches may be tried. A priori and progressive preference articulation may be used, in case real problem data and decision maker preference data are available. The problem may be recast in a form, where modes are to be decided upon rather than the risk states. The problem formulation may be made more realistic by allowing for dependent risks and/or resource constraints. The impacts and probability of occurrences of risks may be formulated using continuous functional forms.

References

Abbasi,B., S. Shadrokh, and J. Arka, 2006, Bi-objective resource-constrained project scheduling with robustness and makespan criteria, Applied Mathematics and Computation, article in press.

Abraham, A., L.C. Jain, and D.E. Goldberg, 2005, Evolutionary Multiobjective Optimization (Springer Verlag, Berlin).

Al-Fawzan, M.A. and M. Haouari, 2005, A bi-objective model for robust resource-constrained project scheduling, International Journal of Production Economics 96, 175–187.

Chapman, C. and S. Ward, 1999, Project Risk Management (John Wiley and Sons, New York).

Coello, C.A.C., D.A.van Veldhuizen, and G.B. Lamont, 2002, Evolutionary Algorithms for Solving Multi-objective Problems (Kluwer Academic Publishers, Boston).

Davis, K.R., A. Stam, and R.A. Grzybowski, 1992, Resource constrained project scheduling with multiple objectives: A decision support approach, Computers and Operations Research 19, 7, 657-669.

De, P.E., J. Dunne, J.B. Ghosh, and C.E. Wells, 1997, Complexity of the discrete time-cost trade-off problem for project networks, Operations Research 45-2, 302-306.

Deb, K., 2001, Multiobjective Optimization Using Evolutionary Algorithms (John Wiley and Sons, New York).

Demeulemeester, E.L., W.S. Herroelen, and S.E. Elmaghraby, 1996, Optimal procedures for the discrete time/cost trade-off problem in project networks, European Journal of Operational Research, 88, 50-68.

Elmaghraby, S. E., 1977, Activity Networks: Project Planning and Control by Network Models (John Wiley and Sons, New York).

Elmaghraby, S.E., 2005, On the fallacy of averages in project risk management, European Journal of Operational Research 165, 307–313.

Erengüç, S.S. and O. Içmeli, 1999, Integrating quality as a measure of performance in resource constrained project scheduling problems, in: J. Weglarz, Ed., Project Scheduling-Recent Models, Algorithms and Applications (Kluwer Academic Publishers, Boston) 433-450.

Fonseca, C.M. and P.J. Fleming, 1995, An overview of evolutionary algorithms in multiobjective optimization, Evolutionary Computation 3, 1-16.

Fonseca, C. M. and P.J. Fleming, 1998, Multiobjective optimization and multiple constraint handling with evolutionary algorithms – Part I: A unified formulation, IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans, 28, 1, 26-37.

Goldberg, D.E. and J. Richardson, 1987, Genetic algorithms with sharing for multimodal function optimization, In: Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms, 41-49.

Hanne, T. and S. Nickel, 2005, A multiobjective evolutionary algorithm for scheduling and inspection planning in software development projects, European Journal of Operational Research 167, 3, 663-678.

Hapke, M., A. Jaszkiewicz and R. Slowinski, 1998, Interactive analysis of multiple-criteria project scheduling problems, European Journal of Operational Research 107, 315–324.

Herroelen, W., B.De Reyck, and E. Demeulemeester, 1998, Resource-constrained project scheduling: A survey of recent developments, Computers and Operations Research 25(4), 279-302.

Herroelen, W. and R. Leus, 2005, Project scheduling under uncertainty: Survey and research potentials, European Journal of Operational Research 165, 289–306.

Hwang, C.L. and A. Masud, 1979, Multiple Objective Decision Making - Methods and Applications: A State of the Art Survey, Lecture Notes in Economics and Mathematical Systems 164 (Springer Verlag, Berlin).

Hwang, C.L., S.R. Paidy, K. Yoon, and A. Masud, 1980, Mathematical programming with multiple objectives: A tutorial, Computers & Operations Research, 7, 1-2, 5-31.

Içmeli, O. and W.O. Rom, 1997, Ensuring quality in resource constrained project scheduling, European Journal of Operational Research 103, 483-496.

Içmeli, S.S. Erengüç, and C.J. Zappe, 1993, Project scheduling problems: A survey, International Journal of Operations and Production Management, 13, 11, 80-91.

Jaafari, A., 2001, Management of risks, uncertainties and opportunities on projects: Time for a fundamental shift, International Journal of Project Management 19, 89-101.

Jaskiewicz, A., 2002, Genetic local search for multi-objective combinatorial optimization, European Journal of Operational Research 137, 50-71.

Jones, D.F., S.K. Mirrazavi, and M. Tamiz, 2002, Multiobjective metaheuristics: An overview of the current state-of-the-art, European Journal of Operational Research 137-1, 1-9.

Keeney, R.L. and H. Raiffa, 1976, Decisions with Multiple Objectives: Preferences and Value Tradeoffs (John Wiley and Sons, New York).

Kılıç, M., 2003, Multiobjective genetic algorithm approaches to project scheduling under risk, MSc Thesis, Sabancı University, Istanbul.

Knowles, J. and D. Corne, 2002, On metrics for comparing non-dominated sets, Proceedings of the Congress on Evolutionary Computation (CEC'2002), IEEE Service Center, NJ, 1, 711-716.

Kogan, K. and A. Shtub, 1999, Scheduling projects with variable intensity activities: The case of dynamic earliness and tardiness costs, European Journal of Operational Research 118, 1, 65-80.

Kolisch, R. and R. Padman, 2001, An integrated survey of deterministic project scheduling, Omega 29, 249-272.

Özdamar, L. and G. Ulusoy, 1994, A survey on the resource-constrained project scheduling problem, IIE Transactions, 27, 574-586.

Project Management Institute (PMI), 2000, A Guide to the Project Management Body of Knowledge (PMI, Pittsburgh).

Rys, T., R. Stanek, and W. Ziembla, 1994, MIPS: A DSS for multiobjective interactive project scheduling, European Journal of Operational Research 79, 2, 196-207.

Slowinski, R., B. Soniewicki, and J. Weglarz, 1994, DSS for multiobjective project scheduling, European Journal of Operational Research 79, 2, 220-229.

Slowinski, R., 1989, Multiobjective project scheduling under multiple-category resource constraints, in: : R. Slowinski, J. Weglarz (Eds.), Advances in Project Scheduling (Elsevier, Amsterdam).

Stewart, T.J., 1991, A multicriteria decision support system for the R&D project selection, Journal of the Operational Research Society 421, 17-26.

Tan, K.C., E.F. Khor and T.H. Lee, 2005, Multiobjective Evolutionary Algorithms and Applications (Springer Verlag, Berlin).

Tavares, L.V., 1994, A stochastic model to control project duration and expenditure, European Journal of Operational Research 78, 2, 262-266.

Tavares, L.V., J.A.A. Ferreira, and J.S. Coelho, 1998, On the optimal management of project risk, European Journal of Operational Research 107, 2, 451-469.

Ulusoy, G. and L. Özdamar, 1996, A framework for an interactive project scheduling system under limited resources, European Journal of Operational Research 90, 362 – 375.

Van Veldhuizen, D. and G. Lamont, 2000, Multiobjective evolutionary algorithms: Analyzing the state-of-the-art, Evolutionary Computation, 8, 2, 125-147.

Viana, A. and J.P. de Sousa, 2000, Using metaheuristics in multiobjective resource constrained project scheduling, European Journal of Operational Research, 120, 2, 359-374.

Weglarz (editor), J., 1999, Project Scheduling-Recent Models, Algorithms and Applications (Kluwer Academic Publishers, Boston).

Wei, C.C. and C.M.F. Wang, 2003, Efficient approaches of linearization in project compression, Computers and Industrial Engineering 44, 695-706.

Zafra-Cabeza, A., M. A. Ridao and E.F. Camacho, 2004, An algorithm for optimal scheduling and risk assessments of projects, Control Engineering Practice 12, 1329-1338.

Zionts, S. and J. Wallenius, 1976, An interactive programming method for solving the multiple criteria problem, Management Science, 22, 652-663.

Table 1. Risk states for an example activity X

Activity X Duration (d_X): 20 TU

State	Probability of Occurrence (p_{jnk})	Impact (I_{jnk})	Cost (MU) (C_{jnk})
1	0.7	0.5	0
2	0.6	0.5	150
3	0.6	0.4	300

Table 2. Risk levels for an example activity Y

Activity Y Duration (d_Y): 18 TU $|L_Y|=1$ $W_{1Y}=7$ $C_l=20$ MU/TU

Risk 1				Risk 2			
State	Probability of Occurrence (p_{jnk})	Impact (I_{jnk})	Cost (C_{jnk}) (MU)	State	Probability of Occurrence (p_{jnk})	Impact (I_{jnk})	Cost (C_{jnk}) (MU)
1	0.55	0.90	0	1	0.95	0.70	0
2	0.20	0.30	1240	2	0.90	0.40	360

Table 3. Mode generation and non-dominated mode selection for the example activity

Mode	State chosen for Risk 1	State chosen for Risk 2	Cost (MU)	Duration (TU)	Domination statue
1	1	1	5443.20	38.88	dominated (by modes 2 and 4)
2	1	2	5034.60	33.39	non-dominated
3	2	1	5587.00	31.05	dominated (by mode 4)
4	2	2	5178.40	25.56	non-dominated

Table 4. Population size and number of generations

Number of activities in the problem	Population size	Number of generations
15	200	250
25	200	375
35	200	500

Table 5. Average percent deviation of EHR values for the GA from the ones for the approximation of the true Pareto front and the *p*-values of the paired *t*-test

Problem type	Average percent deviation	*p* value
Overall	6.44	$3.94*10^{-15}$
15 Activities	5.62	$2.78*10^{-5}$
25 Activities	4.74	$2.14*10^{-6}$
35 Activities	8.97	$2.03*10^{-7}$

Table 6. Result summary for the performance of the improvement heuristic CCDMH

Problem type	Number of non-dominated solutions	Average improvement (%)	Number of solutions improved	Ratio of improved solutions (%)
Overall	2546	0.50	1948	76.51
15 Activities	717	0.16	422	58.86
25 Activities	848	0.63	693	81.72
35 Activities	981	0.71	833	84.91

Table 7. Result summary for the performance of the improvement heuristic GABH

Problem type	Number of non-dominated solutions	Average improvement (%)	Number of solutions improved	Ratio of improved solutions (%)
Overall	2546	0.27	2039	80.09
15 Activities	717	0.19	421	58.72
25 Activities	848	0.32	723	85.26
35 Activities	981	0.30	895	91.23

Table 8. Labor and preventive cost percentages within the expected total cost

Problem type	Labor cost %	Preventive cost %
Overall	50.71	43.54
15 Activities	46.28	46.46
25 Activities	52.87	41.99
35 Activities	53.00	42.18

Table 9. Computational times in CPU milliseconds

Problem Type	Proposed GA	PFA	CCDM
15 Activities	6 339	167 075	17 588
25 Activities	10 094	443 936	23 385
35 Activities	16 291	849 036	27 976

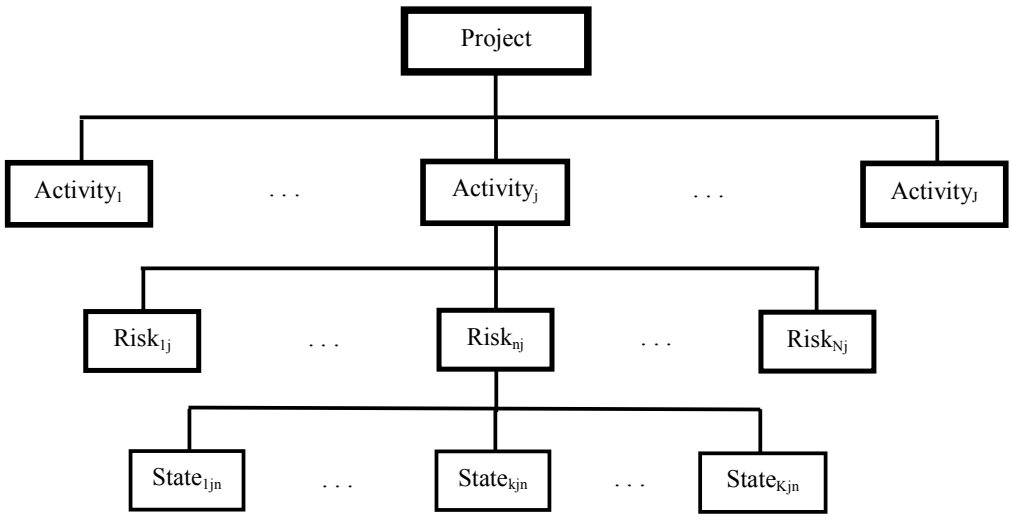

Figure 1. Elements of the model for the problem of project scheduling under risk

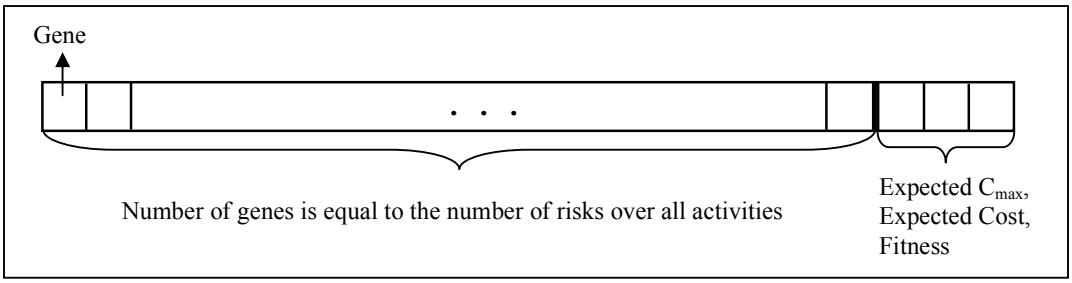

Figure 2. Chromosome representation

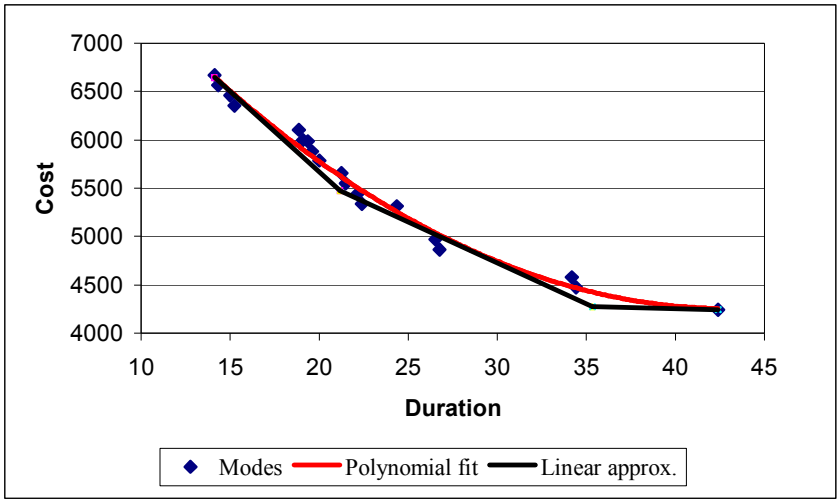

Figure 3. An example of piecewise linear curve fitting

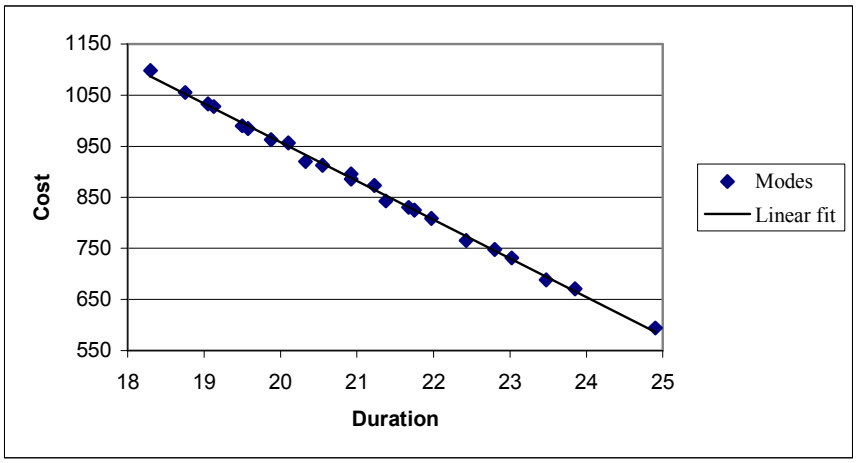

Figure 4. An example of linear fit for time/cost scatter of an activity

Step 1 - Find a starting point solution with mode assignments to all activities using one of the two heuristics explained in 4.2.1 and 4.2.2.

Step 2 - For each noncritical activity, calculate the slack and the saving per duration value that will result, if the activity is performed at its next higher duration mode.

Step 3 - Starting with the activity with the highest saving per duration value, expand the activity without violating the slacks.

Step 4 - If there are other activities, whose slacks allow for expansion, go to step 2; else stop.

Figure 5. The improvement algorithm

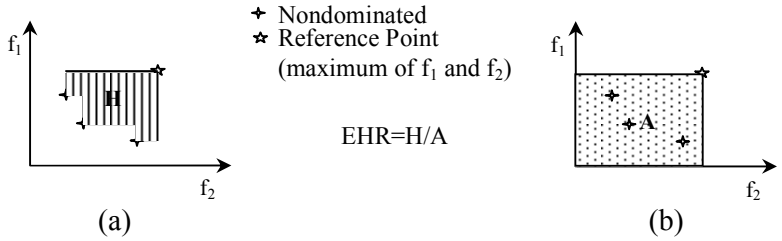

Figure 6. Illustration of the performance metric denoted as extreme hyperarea ratio (EHR)

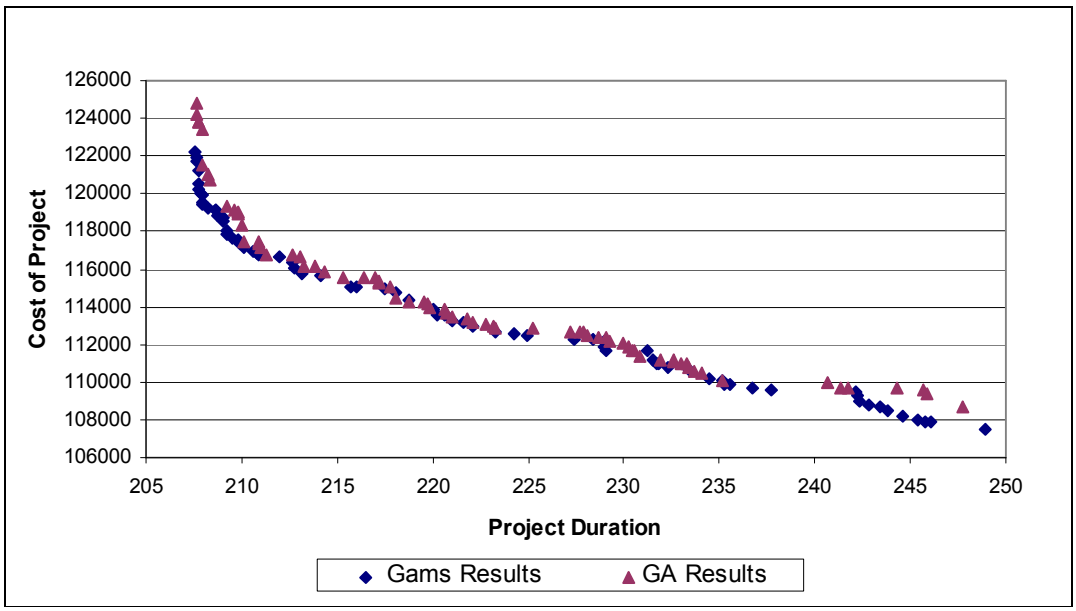

Figure 7. An example plot: The performance of the GA is very good (|J|=15)

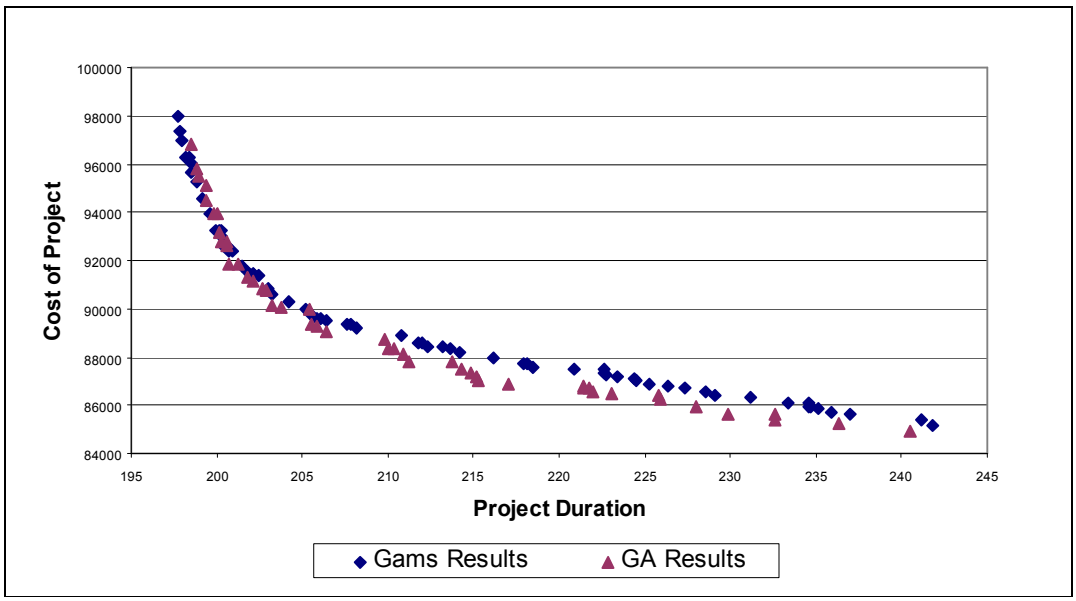

Figure 8. An example plot: Performance of the GA is better than GAMS' results (|J|=25)

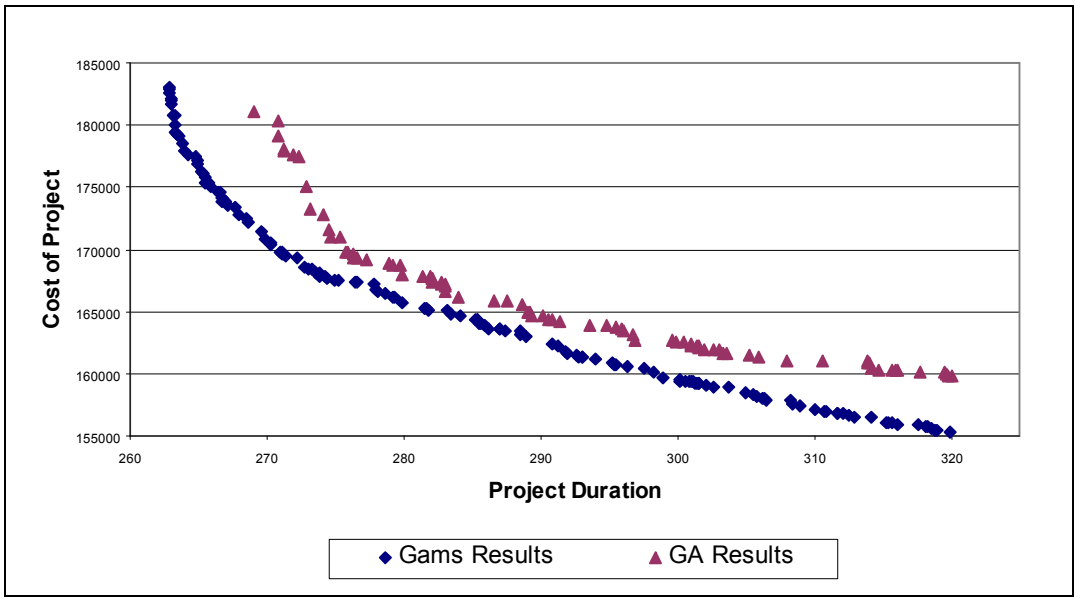

Figure 9. An example plot: Performance of the GA is poor (|J|=35)