Transaction E: Industrial Engineering
Vol. 16, No. 2, pp. 91–99
© Sharif University of Technology, December 2009

SCIENTIA IRANICA

Concurrent Project Scheduling and Material Planning: A Genetic Algorithm Approach

M. Sheikh Sajadieh[1], Sh. Shadrokh[1,*] and F. Hassanzadeh[2]

Abstract. *Scheduling projects incorporated with materials ordering results in a more realistic problem. This paper deals with the combined problem of project scheduling and material ordering. The purpose of this paper is to minimize the total cost of this problem by determining the optimal values of activity duration, activity finish time and the material ordering schedule subject to constraints. We employ a genetic algorithm approach to solve it. Elements of the algorithm, such as chromosome structure, unfitness function, crossover, mutation and local search operations are explained. The results of the experimentation are quite satisfactory.*

Keywords: *Project scheduling; Genetic algorithm; Material ordering.*

INTRODUCTION

A traditional strategy was to treat project scheduling and materials ordering separately. Based on this strategy, project scheduling is first carried out and, then by taking the activity schedule as a known parameter, a material ordering plan is determined. Following this method makes the project managers not consider the trade-offs between cost elements such as material ordering costs, material holding costs, procurement costs and the reward or penalty for the project completion time and, therefore, the total cost of the project increases.

The first paper in the integration of Project Scheduling and Material Ordering (PSMO) appears to be attributed to Aquilano and Smith [1]. They developed a model which integrates Material Requirement Planning (MRP) consisting of material, lead-times and inventory level scheduling, and the Critical Path Method (CPM). Smith-Daniels and Aquilano [2] extended that work by proposing a heuristic procedure for scheduling large projects wherein requirements for all renewable and non-renewable resources are incorporated. The heuristic also takes into account variations in activity duration and precedence constraints.

Smith-Daniels and Smith-Daniels [3] presented a mixed integer programming model for a PSMO problem that offers an optimal scheduling of project activities and materials orders. They demonstrated that the latest starting time schedule provides an optimal solution. It was also shown that the problem can be solved optimally when it is decomposed into a derivation of the project schedule and a derivation of the materials ordering plan. They used the Wagner-Whitin algorithm [4] to find the optimal ordering plan of materials for a known project schedule. Erabsi and Sepil [5] presented a heuristic procedure to determine the tradeoff between expediting the materials ordering and delaying the project.

Dodin and Elimam [6] extended that work by including variable activity duration, variable project worth, rewards for early project completion and material quantity discounts into the model. They showed that the variability in activity duration gives more flexibility to project scheduling, resulting in more cost reduction and allowing for savings on the completed activities and materials ordering cost. They formulated the problem as a mixed integer programming model and used some analytical results to reduce the size of the model and improve its solution efficiency. Compu-

1. *Department of Industrial Engineering, Sharif University of Technology, Tehran, Iran.*
2. *Department of Computer Engineering, Ferdowsi University of Mashhad, Mashhad, Iran.*
*. *Corresponding author. E-mail: shadrokh@sharif.edu*

Received 2 June 2007; received in revised form 17 February 2008; accepted 2 July 2008

tational results show that solution times are strongly affected by the increase in problem size and, therefore, their numerical research was limited to the problems with 30 activities at the most.

Schmitt and Faaland [7] used a heuristic for scheduling a recurrent construction. The heuristic first generates an initial schedule that dispatches worker teams to tasks for the backlogged products and, then, solves a series of maximal closure problems to find material release times that maximize the net present value of cash flows.

In this paper, we extend the PSMO problem investigated by Dodin and Elimam [6] by developing a Genetic Algorithm (GA) approach to find the optimal solution for large scale problems. The paper is organized as follows. First, the modeling assumptions are discussed and expected cost functions are calculated. Then, the genetic algorithm approach is described. Following that, numerical results and parametric analysis are presented. Finally, our findings are summarized.

PROBLEM DESCRIPTION

Assumptions

The following assumptions are used throughout this paper to develop the model:

1. A project with N activities is considered. The precedence relations of activities are zero-lag, finish-to-start and shown by an activity on the node network with no loop. Activities 1 and N are dummies that represent the project start and completion, respectively.
2. Activity duration is considered to be a variable that can vary from normal time to crash time.
3. The activities need M types of nonrenewable resources.
4. The amount required from material m to process activity j is independent of the activity duration.
5. All the amounts of each material needed for each activity are ordered at the same time.
6. An all-unit discount policy is proposed for each material.
7. The objective function elements, which are presented in the following section, are adopted from Dodin and Elimam [6].

Mathematical Model

The purpose of this paper is to minimize the total cost of the PSMO problem by determining the optimal values of activity duration, activity finish time and the material ordering schedule subject to constraints. We use the same notation as introduced by Dodin and Elimam [6] for similar parameters as follows.

Indices

$j = 1, \cdots, N$	index of project activities,
$m = 1, \cdots, M$	index of materials,
$k = 1, \cdots, K_m$	index of discount ranges,
$t = 0, \cdots, H$	index of time.

Parameters

a. Activity related

P_j	Set of activities preceding activity j,
b_j	Crashing cost of activity j,
c_j	The cost of reducing the duration of activity j by one period,
u_j	Upper bound on the duration of activity j (normal time),
v_j	Lower bound on the duration of activity j (crash time),
e_j	The earliest completion time of activity j, assuming that the project starts at time zero and the duration of each activity is equal to its crash time,
l_j	The latest completion time of activity j, assuming that the project finishes at time H and the duration of each activity is equal to its crash time.

b. Material related

α_{mk}	Limit on quantity range k of material m,
δ_{mk}	Cost of material m purchased in quantity range k,
G_m	Ordering cost of material m,
h_m	Inventory holding cost of one unit of material m for one period,
K_m	Number of quantity discount ranges for material m,
L_m	Lead time of material m in periods,
R_{jm}	Amount required from material m to process activity j.

c. Project related

d	Due date of the project after which a delay penalty cost is paid,
H	The planning horizon,
p	Penalty cost per period for having the project late beyond d,
r	Reward paid per period for completing the project before d,
s	Percentage of the activity's worth representing the holding cost, per period, of the completed activities.

Non-Negative Variables:

$X_{jt}\begin{cases}1 & \text{If activity } j \text{ is completed in period } t.\\ 0 & \text{Otherwise.}\end{cases}$

$P_{mktj}\begin{cases}1 & \text{If } m\text{th material of activity } j \text{ is ordered in period } t \text{ and the total amount of orders in this period falls within quantity range } k.\\ 0 & \text{Otherwise.}\end{cases}$

z_j Duration of activity j.

$\lambda_{mkt}\begin{cases}1 & \text{If material } m \text{ is ordered within quantity range } k \text{ in period } t.\\ 0 & \text{Otherwise.}\end{cases}$

I_{mt} Inventory level of material type m by the end of period t.

w_{jt} Worth of activity j completed by the end of period t.

W_t Worth of project by the end of period t.

Model Formulation

The PSMO problem is formulated as a Mixed Integer Programming (MIP) model as follows:

$$\begin{aligned}\min & \sum_{j=1}^{N}[b_j - c_j(z_j - v_j)] - \sum_{t=e_N}^{d} r(d-t)X_{Nt}\\ & + \sum_{t=d}^{H} p(t-d)X_{Nt} \sum_{t=1}^{H-1} sW_t\\ & + \sum_{m=1}^{M}\sum_{t=1}^{H-L_m} G_m \sum_{k=1}^{K_m}\lambda_{mkt}\\ & + \sum_{m=1}^{M}\sum_{t=1}^{H-L_m}\sum_{k=1}^{K_m}\sum_{j=1}^{N}\delta_{mk}R_{jm}P_{mktj}\\ & + \sum_{m=1}^{M}\sum_{t=1}^{H-1} h_m I_m.\end{aligned}$$

Subject to:

$$\sum_{t=e_i}^{l_i} tX_{it} + z_j - \sum_{t=e_j}^{l_j} tX_{jt} \le 0, \qquad \forall i \in P_j,$$

$$j = 1, \cdots, N, \tag{1}$$

$$X_{10} = 1,$$

$$v_j \le z_j \le u_j, \qquad j = 1, \cdots, N, \tag{2}$$

$$\sum_{t=e_j}^{l_j} X_{jt} = 1, \qquad j = 1, \cdots, N, \tag{3}$$

$$w_{jt} \ge [b_j - c_j(z_j - v_j)] - b_j(1 - X_{jt}),$$

$$j = 1, \cdots, N, \qquad t = 1, \cdots, H, \tag{4}$$

$$W_t \ge W_{t-1} + \sum_{j=1}^{N} w_{jt}, \qquad t = 1, \cdots, e_N, \tag{5}$$

$$W_0 = 0,$$

$$W_t \ge W_{t-1} + \sum_{j=1}^{N} w_{jt} - \left(\sum_{j=1}^{N} b_j\right) \cdot \sum_{\tau=e_N}^{t} X_{N\tau},$$

$$t = e_N + 1, \cdots, H, \tag{6}$$

$$I_{mt} = I_{m(t-1)} + \sum_{j=1}^{N} R_{jm}\left(\sum_{k=1}^{K_m} P_{mk(t-L_m)j} - X_{jt}\right),$$

$$m = 1, \cdots, M, \qquad t = 1, \cdots, H, \tag{7}$$

$$I_{m0} = 0, \qquad m = 1, \cdots, M,$$

$$\alpha_{m(k-1)}\lambda_{mkt} \le \sum_{j=1}^{N} R_{jm}P_{mktj} \le \alpha_{mk}\lambda_{mkt},$$

$$m = 1, \cdots, M, \qquad t = 1, \cdots, H,$$

$$k = 1, \cdots, K_m, \tag{8}$$

$$\alpha_{m0} = 0,$$

$$\sum_{k=1}^{K_m} \lambda_{mkt} \le 1, \qquad m = 1, \cdots, M, \qquad t = 1, \cdots, H, \tag{9}$$

$$\sum_{k=1}^{K_m}\sum_{t=1}^{H-1} P_{mktj} = 1, \qquad j = 1, \cdots, N,$$

$$m = 1, \cdots, M, \tag{10}$$

$$\sum_{k=1}^{K_m}\sum_{t=1}^{H-1} tP_{mktj} \le \sum_{t=e_j}^{l_j} tX_{jt} - L_m,$$

$$j = 1, \cdots, N, \qquad m = 1, \cdots, M, \tag{11}$$

$$X_{jt} = \{0, 1\}, \qquad \lambda_{mkt} = \{0, 1\},$$

$$P_{mktj} = \{0, 1\}, \qquad \forall\ m, k, t, j, \tag{12}$$

$$z_j \geq 0, \qquad I_{mt} \geq 0, \qquad w_{jt} \geq 0, \qquad W_t \geq 0,$$

$$\forall m, t, j. \tag{13}$$

The total cost function consists of seven elements which are crashing cost, reward for early project completion, project delay penalty, completed activities holding cost, material ordering cost, procurement cost and inventory holding cost, respectively. Constraints 1 take into consideration the precedence relation between each pair of activities (i, j), where i immediately precedes j. Constraints 2 limit activity durations between their normal and crash time. Constraint 3 guarantees that each activity j can only have one finish time. Sets of Constraints 4-6 are used to calculate activities and project worth. Constraint 7 is a balance equation to monitor the inventory level over the planning horizon. Constraints 8 and 9 satisfy the discount conditions, where the order quantity from each material per period should be in an appropriate interval of the discount pricing schedule and only in one interval. Constraint 10 guarantees that each material m is ordered for each activity j just one time. Constraint 11 stipulates that all materials required for each activity should be ready at the activity finish time. The sets of Constraints 12 and 13 denote the domain of variables.

One can convert the problem to a multidimensional 0-1 knapsack problem (Freville [8]). For example, if we simplify the model by assuming the activity duration to be constant, then Constraints 4 to 6, which are used to linearize the model, can be omitted and, thus, $\sum_{t=1}^{H-1} sW_t$ will be replaced with $\sum_{j=1}^{N} s(b_j - c_j(z_j - v_j))(\sum_{t=1}^{H} tX_{Nt} - \sum_{t=1}^{H} tX_{jt})$ in the objective function. Now, if we relax all constraints except Constraints 1, 3 and 11 and also reduce the objective function to just minimizing the completed activity holding cost, then we reach the following model:

$$\min \sum_{j=1}^{N} s(b_j - c_j(z_j - v_j)) \left(\sum_{t=1}^{H} tX_{Nt} - \sum_{t=1}^{H} tX_{jt} \right),$$

st:

$$\sum_{t=e_i}^{l_i} tX_{it} + z_j - \sum_{t=e_j}^{l_j} tX_{jt} \leq 0, \qquad \forall i \in P_j,$$

$$j = 1, \cdots, N,$$

$$\sum_{t=e_j}^{l_j} X_{jt} = 1, \qquad j = 1, \cdots, N,$$

$$\sum_{k=1}^{K_m} \sum_{t=1}^{H-1} tP_{mktj} \leq \sum_{t=e_j}^{l_j} tX_{jt} - L_m, \qquad j = 1, \cdots, N,$$

$$m = 1, \cdots, M,$$

$$X_{jt} = \{0, 1\}, \qquad \forall j, t.$$

This is a multidimensional 0-1 knapsack model, which is a known NP-hard. Since PSMO can be converted to an NP-hard problem with some simplification, so PSMO is also NP-hard. Thus, the optimal solution can only be achieved by exact algorithms in small instances. Therefore, heuristic methods are needed to solve large scale problems. We propose a genetic algorithm to solve the problem. However, one may compare the results of GA with other heuristic methods in order to clear up which one is more efficient.

GENETIC ALGORITHM

Basic Scheme

One of the most important aspects of the genetic algorithm is the structure of its chromosomes (Genotype), which is explained in the following section. Each chromosome gives a unique value to the decision variables, which are activity durations (z_j's), activity finish times (f_j's) and the ordering times of materials (ot_{mj}'s). The following is a presentation of the scheme of the algorithm.

The genetic algorithm starts by the random generation of the initial population and by computing its unfitness, which is the cost of its related schedule. The size of the population, $PS = 100\ (N \times M)^{\alpha}$, remains constant for all generations. Each new generation is made from existing generations using three operations: crossover, mutation, and local search. In the crossover operation, P_{cr} percentages of pairs of parents are randomly selected from the existing generation and, on each pair, a crossover operation is performed. Additionally, P_{mu} percentage of individuals, randomly selected, is considered for mutation operation. Moreover, a local search operation is employed to improve some randomly selected individuals. A local search is used for P_l percentage of individuals. The number of individuals generated by each run is assumed to be equal to PS and, thus, $2P_{cr} + P_{mu} + P_l = 1$.

After constructing each generation by the above operations, we choose the members of the next generation by retaining the best P_{se} percentage of the previous generation and selecting the remaining $1 - P_{se}$ percentage with transference rule.

The termination criterion is set as 150 generations with no improvement in the best solution. If this condition has been satisfied, then the GA stops and returns the best individual. Otherwise, a new generation will

be generated. Note that α, P_{cr}, P_{mu}, P_l and P_{se} are adjustable parameters of the algorithm.

Chromosome Representation

Each individual, I, is constructed from a $(M+2)\times N$ matrix. The first row of this matrix represents activity durations. The second row represents activity finish times and the third to $(M+2)$th rows represent the ordering times of materials for each activity. Based on the model, all of these cells contain integer values.

$$I = \begin{bmatrix} z_1^I & z_2^I & \cdots & z_N^I \\ f_1^I & f_2^I & \cdots & f_N^I \\ ot_{11}^I & ot_{12}^I & \cdots & ot_{1N}^I \\ \cdots & \cdots & \cdots & \cdots \\ ot_{M1}^I & ot_{M2}^I & \cdots & ot_{MN}^I \end{bmatrix}.$$

Considering DV_{mq}^I to be the qth set of activities in chromosome I where their mth material is ordered in the same time, and DV_m^I to be the number of different sets of similar ot_{mj}^I in the ordering schedule of material m, the value of unfitness function, $f(I)$, would be $f(I)=\sum_{i=1}^{7}\Phi_i^I$, so that:

$$\Phi_1^I = \sum_{j=1}^{N}[b_j - c_j(z_j^I - v_j)],$$

$$\Phi_2^I = -r(d - f_N^I), \qquad \text{if } f_N^I \le d,$$

$$\Phi_3^I = p(f_N^I - d), \qquad \text{if } f_N^I > d,$$

$$\Phi_4^I = s\sum_{j=1}^{N}[b_j - c_j(z_j^I - v_j)](f_N^I - f_j^I),$$

$$\Phi_5^I = \sum_{m=1}^{M} G_m DV_m^I,$$

$$\Phi_6^I = \sum_{m=1}^{M}\sum_{q=1}^{DV_m^I}\left(\sum_{m\in DV_{mq}^I} R_{jm}\right)\delta_{mk},$$

$$\text{if } \alpha_{m(k-1)} \le \left(\sum_{m\in DV_{mq}^I} R_{jm}\right) \le \alpha_{mk},$$

$$k = 1, \cdots, K_m,$$

$$\Phi_7^I = \sum_{m=1}^{M}\sum_{j=1}^{N} h_m R_{jm}(f_j^I - ot_{mj}^I - L_m).$$

Initial Population

We generate the initial population using two approaches. Creating initial population members in these ways ensures that all chromosomes in the initial population represent feasible solutions. The detailed design of each code follows:

(a) Forward approach

The pseudo-code of this approach is given below:

A= set of activities that can be selected (their precedence activities have been selected before).

PN_i= number of precedents of activity i that have not been selected yet.

Set $j = 1$

While $j \le N$

$z_j \leftarrow$ int $[v_j, u_j]$, a discrete uniform distribution, including v_j and u_j

$j \leftarrow j + 1$

End while

Set $A = \{1\}$, $j = 1$, $f_1 = 0$, $PN_i = \sum_{k\in P_i} 1 \forall i$

While $A \neq \phi$

Select one activity from A by random (e.g. j)

$f_j \leftarrow \max_{i\in P_j}\{f_i\} + z_j$

$PN_i \leftarrow PN_i - 1 \qquad \forall i | j \in P_i$

$PN_j \leftarrow -1$

$A \leftarrow A - \{j\} + \{\forall i | PN_i = 0\}$

End while

Set $m = 1$, $j = 1$

While $m \le M$

While $j \le N$

$ot_{mj} \leftarrow$ int $[0, f_j - L_m]$

$j \leftarrow j + 1$

End while

Set $j = 1$, $m \leftarrow m + 1$

End while

(b) Backward approach
This approach is almost similar to the forward approach, except that we start from H coming back to zero and, after generating the chromosome, all activities are locally shifted so that the start time of the project is equal to zero.

Crossover Operator

We use three crossover operators: one-point, two-point, and three-point crossovers. Let Pa_1 and Pa_2 be a pair of parents selected for crossovers. Two children, I_1 and I_2, are defined from this crossover. The pseudo-code of the one-point crossover operator is given below:

$I_1 \leftarrow Pa_1$, $I_2 \leftarrow Pa_2$

$r \leftarrow$ int $[2, N-1]$, selecting a breaking point

Set $k = 1$

While $k \leq 2$

Set $lf = H$, $j = r + 1$

While $j \leq N$

Set $mf = 0$, $i = 0$

While $i \leq r$

If $i \in P_j$ & $f_i^{I_k} > mf$ then $mf \leftarrow f_i^{I_k}$

$i \leftarrow i + 1$

End while

If $f_j^{I_k} - z_j^{I_k} - mf < lf$ then $lf \leftarrow f_j^{I_k} - z_j^{I_k} - mf$

$j \leftarrow j + 1$

End while

If $lf > 0$ then $lf \leftarrow$ int $[0, lf]$

$f_j^{I_k} \leftarrow f_j^{I_k} - lf \qquad \forall\, j \geq r + 1$

Set $j = r + 1$

While $j \leq N$

$ot_{mj}^{I_k} \leftarrow ot_{mj}^{I_k} - lf + a \qquad \forall m$

If $ot_{mj}^{I_k} < 0$ then $ot_{mj}^{I_k} \leftarrow$ int $[0, f_j - L_m]$

$j \leftarrow j + 1$

End while

$k \leftarrow k + 1$

End while

The second and third crossover operators use the same logic. For example, if in a two point crossover, Pa_1 and Pa_2 are a pair of parents selected for crossover, then CH_1 and CH_2 will be their child, assuming r_1 and r_2 as breaking points (Figure 1).

Mutation Operator

We use two mutations. Let I be the chromosome that is selected for mutation. The first mutation operator selects $\gamma = N/5$ integer numbers randomly from the interval $[2, N-1]$, representing selected activities, and an integer number, m, from the interval $[1, M]$. Then, a new ordering time of the selected material is generated for selected activities. Using this operator, activity durations and finish times remain unchanged. The pseudo-code of this operator is as follows:

$\gamma \leftarrow N/5$

Set $a = 1$

While $a \leq \gamma$

$j \leftarrow int[2, N-1]$

$m \leftarrow int[1, M]$

$ot_{mj}^{I} \leftarrow int[0, f_j^I - L_m]$

$a \leftarrow a + 1$

End while

The second mutation selects $\gamma = N/5$ integer numbers randomly from the interval $[2, N-1]$, representing activities, and replaces the selected activity's duration by a new random value. Thus, activity finish times and ordering times may need change. We use the approaches employed for initial population generation to update the second to the $(M+2)$th rows. The pseudo-code of this operator is as follows:

$\gamma \leftarrow N/5$

Set $a = 1$

While $a \leq \gamma$

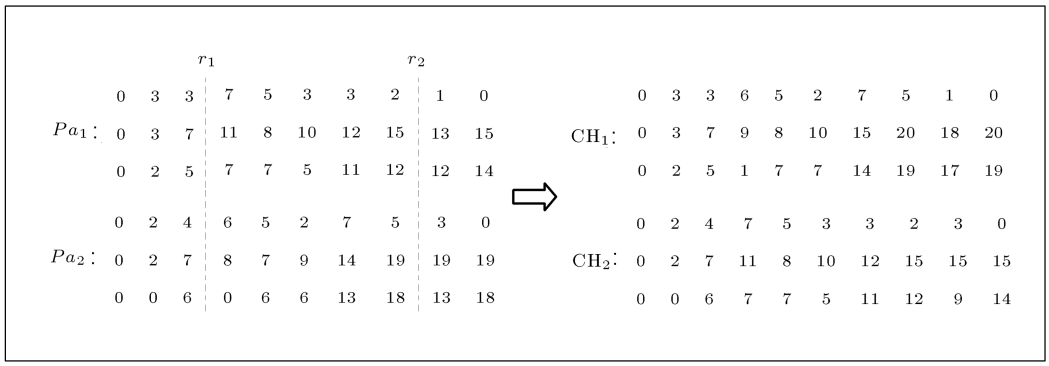

Figure 1. Two-point crossover example.

$j \leftarrow$ int $[2, N-1]$

$z_j^I \leftarrow$ int $[v_j, u_j]$

$a \leftarrow a+1$

End while

Update $\underline{f}^I$ and $\underline{\underline{ot}}^I$, using forward or backward approach.

Local Search Operator

Applying a local search, we are trying to change from existing chromosomes to those with less unfitness, by increasing the ordering times uniformity. This results in less ordering costs. Moreover, higher discount ranges may be achievable and thus, total cost may decrease. Two local search operators are employed in the algorithm. Considering as the selected chromosome for a local search, the pseudo-code of the first operator is as follows:

$\tau \leftarrow N.M/10$

Set $a = 1$

While $a \leq \tau$

$Nf \leftarrow H$

$m \leftarrow$ int $[1, M]$, selecting a material

$s = s' \leftarrow$ int $[2, 5]$, number of activities whose

ordering time will be the same

$j = j' \leftarrow$ int $[s, N]$

While $s \geq 1$, obtaining minimal potential ordering time

If $f_j^I - L_m \leq Nf$ then $Nf \leftarrow f_j^I - L_m$

$j \leftarrow j-1$, $s \leftarrow s-1$

End while

While $s' \geq 1$, replacing new ordering times

$ot_{mj'}^I \leftarrow Nf$

$j' \leftarrow j'-1$, $s' \leftarrow s'-1$

End while

$a \leftarrow a+1$

End while

The second operator is almost the same as the first except that those activities whose ordering times should become similar are selected randomly.

Transference Rule

Using the transference rule, the next generation of chromosomes is chosen from the list of existing individuals. This way, we first obtain $\vartheta_I = f(I)/\max_I f(I)$ and then, a random value is generated between $(0, \vartheta_I)$ for each individual. The individuals with fewer values are selected for the next generation.

COMPUTATIONAL RESULTS

Computational studies on the proposed GA for the PSMO problem have been carried out. The purpose of these experiments was to evaluate the performance of our GA across a variety of situations. To test the proposed GA, we developed experiments for networks with 10, 30, 60, 90 and 120 non-dummy activities and 1 to 4 materials. The structure of the networks as well as activities' normal duration was generated using

ProGen software [9]. The remaining input parameters were generated at random from a set of parameters; one for each of the underlying parameters.

All of the computations were performed on an IBM-compatible PC with Pentium 4 and 2.80 GHz, CPU speed and 512 MB RAM memory under Windows XP as the operating system. All procedures have been coded in C++ and compiled with the Microsoft Visual C++ 6.0 compiler.

Parameter Setting

In order to improve the performance of GA, we use the design of experiment techniques to find the good GA set of parameters before running the GA. The preliminary tests were conducted to determine appropriate values for α, P_{cr}, P_{mu} and P_{se}. The experimental layout is a full-factorial design [10], involving four factors. This experimental layout used three levels for each factor. The following selections for the parameter values are used: $\alpha \in \{0.10, 0.15, 0.20\}$, $P_{cr} \in \{0.1, 0.2, 0.3\}$, $P_{mu} \in \{0.1, 0.2, 0.3\}$ and $P_{se} \in \{0.4, 0.6, 0.8\}$. Each row of experimental runs is one combination of these factors and, hence, there are 81 rows. In each row, we run the GA for 4 seconds for 10 certain instances with 60 non-dummy activities and four resources. Thus, 810 instances were solved. We used the average of 20 runs of each instance for the parameter setting. The distribution of these average values is needed for statistical analysis. Using the Shapiro-Wilk test, the assumption of normality distribution was strongly supported.

The average and maximum percent deviation from minimal cost, among all combinations of parameter, were calculated for each row. Using the Duncan Multiple Range test with a significance level equal to 0.05, we see that there are no significant differences between levels of P_{se}.

Table 1 shows some of the good combinations of parameter values and their results with 4-second computing time.

After the test, it was determined that the best results can be obtained by setting 0.2, 0.3, 0.1 and 0.8 for α, P_{cr}, P_{mu} and P_{se}, respectively. Based on the relationship between P_{cr}, P_{mu} and P_l, the best value of P_l will be 0.3.

Table 1. Good combinations of parameter values.

α	P_{cr}	P_{mu}	P_{se}	Av. Dev. %	Max. Dev. %
0.20	0.30	0.10	0.80	1.03	1.79
0.15	0.30	0.10	0.80	1.36	2.22
0.10	0.30	0.10	0.60	1.54	2.37
0.15	0.30	0.20	0.80	1.66	2.86
0.10	0.20	0.10	0.80	1.74	2.56
0.10	0.30	0.10	0.80	1.85	3.12
0.15	0.30	0.10	0.60	2.03	3.22

Performance Analysis

For an analysis of the performance of the GA, 12 instances with 10 activities and 1 to 4 resources were generated and tested. The parameters obtained from the parameter setting step (the previous section) are used for further GA runs. Table 2 shows the results of comparing optimal solutions attained by LINGO 8.0 with the solutions proposed by GA.

We also compared the solutions obtained from GA with the best randomly generated solutions after a time limit. Moreover, for testing the efficiency of the GA, the best solution of the initial population is compared with the best solution found by the algorithm after a time limit. We generated 80 instances with 30, 60, 90 and 120 activities and 1 to 4 materials. The following measures are calculated:

$$\text{Reduction } \% = 100\ (\text{BI} - \text{BA})/\text{BI},$$

$$\text{Improve } \% = 100\ (\text{BR} - \text{BA})/\text{BR},$$

where BR, BA and BI are best random solutions, the best solution found by the algorithm, and the best solution of initial population, respectively.

Table 3 shows that the algorithm improves the best unfitness value obtained from the initial population and random generation by 32.51 and 34.85 percentages on average, respectively.

The Effect of Local Search

In order to test the effect of local search, 80 instances were generated. We obtained the best unfitness values with and without a local search after a certain time

Table 2. GA vs. optimal results.

No. of Activities	No. of Materials	LINGO CPU Time (s)	GA Av. CPU Time (s)	Av. Dev. %	Max. Dev. %
10	1	36.7	0.27	0.27	0.93
10	2	133.3	0.66	0.97	2.09
10	3	604.7	1.02	1.50	2.52
10	4	2462.0	1.64	1.88	2.85

Table 3. GA vs. random and initial values.

No. of Activities	Time Limit (s)	Av. Reduction %	Min. Reduction %	Av. Improve %	Min. Improve %
30	2	30.73	28.41	31.54	30.36
60	3	32.63	30.69	35.28	33.77
90	4	32.81	30.69	35.97	34.55
120	5	33.88	32.02	36.63	35.26

Table 4. Effect of local search.

No. of Materials	Time Limit (s)	Average Percent Improvement %
1	2	5.66
2	2	5.84
3	2	14.45
4	2	19.14
1	3	5.29
2	3	16.13
3	3	20.75
4	3	29.59
1	4	12.05
2	4	21.49
3	4	26.12
4	4	27.51
1	5	10.48
2	5	24.59
3	5	31.77
4	5	34.67

limit. The parameters of the GA without the local search are: $\alpha = 0.2$, $P_{cr} = 0.4$, $P_{mu} = 0.2$ and $P_{se} = 0.8$. Considering FL and FWL to be the best unfitness values for the algorithm with and without a local search, the percentage improvement is defined as 100 (FWL-FL)/FWL. The average improvement for each set of problems is shown in Table 4.

CONCLUSION AND DIRECTIONS FOR FUTURE RESEARCH

A model was developed to integrate the problem of project scheduling with material ordering. This paper is an extension of the PSMO problem investigated by Dodin and Elimam [6] by developing a solution approach so that the model can be solved for large scale problems. The problem is formulated as a Mixed Integer Programming model and a genetic algorithm approach is employed to solve the problem. Finally, the performance of the algorithm was evaluated by solving different instances so that the results were quite satisfactory.

One of the future research directions is to extend this study for stochastic supply lead-times in which the objective function is to minimize the expected total cost per unit time. Other discount policies can also be considered as an extension.

REFERENCES

1. Aquilano, N.J. and Smith, D.E. "A formal set of algorithms for project scheduling with critical path method-material requirements planning", *J. of Oper. Manag.*, **1**(2), pp. 57-67 (1980).
2. Smith-Daniels, D.E. and Aquilano, N.J. "Constrained resource project scheduling subject to material constraints", *J. of Oper. Manag.*, **4**(4), pp. 369-388 (1984).
3. Smith-Daniels, D.E. and Smith-Daniels, V.L. "Optimal project scheduling with materials ordering", *IIE Trans.*, **19**(4), pp. 122-129 (1987).
4. Wagner, H.M. and Whitin, T.M. "Dynamic version of the economic lot size model", *Manag. Sci.*, **5**(1), pp. 89-96 (1958).
5. Erabsi, A. and Sepil, C. "A modified heuristic procedure for materials management in project networks", *Int. J. of Ind. Eng.-Theory*, **6**(2), pp. 132-140 (1999).
6. Dodin, B. and Elimam, A.A. "Integrated project scheduling and material planning with variable activity duration and rewards", *IIE Trans.*, **33**, pp. 1005-1018 (2001).
7. Schmitt, T. and Faaland, B. "Scheduling recurrent construction", *Naval Res. Logist.*, **51**(8), pp. 1102-1128 (2004).
8. Freville, A. "The multidimensional 0-1 Knapsack problem: An overview", *Eur. J. Oper. Res.*, **155**, pp. 1-21 (2004).
9. Kolisch, R., Sprecher, A. and Drexl, A. "Characterization and generation of a general class of resource-constrained project scheduling problems", *Manag. Sci.*, **41**, pp. 1693-1703 (1995).
10. Montgomery, D.C., *Design and Analysis of Experiments*, 4th Edn., John Wiley & Sons (1996).