International Journal of Industrial Engineering, 15(4), 341-348, 2008.

A Revised Version of Ant Colony Algorithm for One-Dimensional Cutting Stock Problem

K. Eshghi[1] and H. Javanshir[2]

[1]Department of Industrial Engineering,
Sharif University of Technology, Tehran, Iran

[2]Department of Industrial Engineering,
Islamic Azad University, South Tehran Branch, Tehran, Iran

Corresponding author's e-mail: {K. Eshghi, eshghi@sharif.edu}

The one-dimensional cutting stock problem has many applications in industries and during the past few years has been one of the centers of attention among the researchers in the field of applied operations research. In this paper, a revised version of Ant Colony Optimization (ACO) technique is presented to solve this problem. This paper is a sequel to the previous ACO algorithm presented by the authors. In this algorithm, according to some probabilistic rules, artificial ants will select cutting patterns and generate a feasible solution. Computational results show the high efficiency and accuracy of the proposed algorithm for solving one- dimensional cutting stock problem.

Significance: The well-known one-dimensional cutting stock problem arises in many industries such as steel, paper, wood, glass and fiber. In many industries, the cost of raw materials is the most percentage part of the total cost and the cutting stock problem is defined to make better use of materials. In this paper a metaheuristic method based on Ant Colony Optimization(ACO) technique is presented to solve the one-dimensional cutting stock problem.

Keywords: One-dimensional cutting stock problem; trim loss problem, ant colony optimization

(Received: 15 January 2007; Accepted in Revised Form: 24 October 2008)

1. INTRODUCTION

The one- dimensional cutting stock problem is one of the most representative combinatorial optimization problems, which arises in many industries such as steel, paper, wood, glass and fiber and during the past few years has been one of the center of attentions of researchers (Karelahti (2002) and Umetani *et al*. (2003)). This attention has also been focused on the solution method of the problem in general or special cases.

Most different versions of one-dimensional cutting stock problem are known as NP-complete. However, in many cases these types of problems can be modeled by means of mathematical programming and their solutions can be found by using approximate methods and heuristics. The main objective of the problem is to find a method for cutting a certain number of pieces of same lengths (stock lengths), into a large number of short pieces (order lengths), which will minimize the overall trim loss by considering different conditions that may appear in real situations.

According to typology introduced by Dyckhoff (1990), cutting stock problems can be categorized by dimensionality, type of assignment, assortment of large objects and assortment of small items.

In this paper, the *1/V/O/R* case has been considered, where *1* refers to one-dimensional problem, *V* means that all items must be produced from a selection of large objects; *O* denotes that there is one large object and *R* indicates that many items of relatively few dimensions exist. The algorithm presented in this paper, can also be used with some modifications for *1/V/I/R*, where *I* means that many identical large objects exist.

Dyckhoff has also classified the solution process of one- dimensional cutting stock problem into two groups: *item-oriented* and *pattern-oriented* approaches. The *Item-oriented* approach is characterized by individual treatment of every item to be cut. In the *pattern-oriented* approach, first, order lengths are combined into cutting patterns, for which - in a succeeding step - the cutting frequencies are determined that are necessary to satisfy the demands. The constraints in the pattern-oriented approach are based on the algorithm that Gilmore and Gomory developed [6, 7]. However, a pattern-oriented approach is possible only when the stock is of the same length or of several standard lengths, and an item-oriented

approach is used when all stock lengths are different and frequencies cannot be determined. The *pattern-oriented* approach is selected in this paper for solving the cutting stock problem.

2. ONE-DIMENSIONAL CUTTING STOCK PROBLEM

The one dimensional cutting stock problem is one of the most representative combinatorial optimization problems, which arises in many industries such as steel, paper, wood, glass and fiber. In many industries, the cost of raw materials is the most percentage part of the total cost (sometimes more than 80%). The cutting stock problem is one of the well-known operation research problems that is defined to make better use of materials. In general, the cutting stock problem could be defined as follows:

One or more large items are available and we want to generate some small required items by cutting them. In this problem the cutting method should be determined in a way that minimum trim loss is made.

The cutting stock problem was firstly described in 1939 by Kantorovich for one- dimensional cutting stock problem. In 1960s Gilmore and Gomory (1961, 1963) published their famous papers about one and two-dimensional cutting stock problems. Their first paper was published about the application of linear programming in solving one-dimensional cutting stock problems in 1961 and it was an excellent start for representing techniques used in actual problems. Gilmore and Gomory papers were the inspiration of many researchers in this field and caused a new movement in analyzing and solving the cutting stock problem.

2.1. Definition of one-dimensional cutting stock problem

The one-dimensional cutting stock problem can be defined informally as follows:
The stock materials (e.g. paper rolls) of a given length are divided into smaller pieces of desired lengths in order to fulfill the demand orders. The goal is to minimize the total amount of stock material used or, equivalently, to minimize the total waste. The most important characteristic in one- dimensional cutting stock problem is dimensionality. The dimensionality is the minimum number of dimensions that are significant in the determination of the solution. Dyckhoff listed the elementary types of dimensionality as one-, two-, three- and multidimensional problems. In one- dimensional cutting stock problem, dimension is the degree of freedom for decision maker. If the two dimensions of the ordered items are the same as the stock material, the decision maker only need to decide the way of cutting the third dimension and the cutting stock problem is one-dimensional. A typical example of the one-dimensional cutting stock problem is cutting of steel bars where the length of the stock bars is fixed.

The main objective in cutting stock problems is to decrease the total cost of waste materials. Therefore, we need to find how many times we use each cutting pattern to have the minimum trim loss cost.

2.2. Formulation of one-dimensional cutting stock problem

In the one- dimensional cutting stock problem, we are given a sufficient number of stock rolls of the same Length L, and n demand orders with given lengths ($l_1, l_2, \ldots, l_n$) and demands ($D_1, D_2, \ldots, D_n$). A cutting pattern is a combination of demand orders from one stock roll. A cutting pattern j is described as $(P_{1j}, P_{2j}, \ldots, P_{nj})$, where P_{ij} is the number of demand order i cut by cutting pattern j. The problem asks to specify the optimal number of using each pattern to have the minimum waste of materials. It is assumed that, without loss of generality, all Input-data are integer. The following model is a standard model for one- dimensional cutting stock problem (Gilmore and Gomory (1961)) :

$$\text{Min } Z = \sum_{j=1}^{m} S_j X_j + \sum_{i=1}^{n} V_i \ell_i \qquad (1)$$

Subject to:

$$\sum_{j=1}^{m} P_{ij} X_j \geq D_i \qquad i = 1, \ldots, n \qquad (2)$$

$$X_j \geq 0, \text{ Integer} \qquad j = 1, \ldots, m \qquad (3)$$

where:
L: The length of stock material
n: The number of orders
m: The number of cutting patterns
X_j: The number of times cutting pattern *j* is selected
S_j: Amount of loss in the cutting pattern *j*

P_{ij}: The number of times the demand order *i* is produced in the cutting pattern *j*

ℓ_i : The length of demand order *i*

D_i: The demand order *i*

V_i : Surplus variable *i* which is equal to $V_i = \sum_{j=1}^{m} (P_{ij} X_j - D_i)$, $i = 1,...,n$ (4)

It is clear that we have:

$$\sum_{i=1}^{n} P_{ij} \ell_i + S_j = L \quad , \quad j = 1,...,m \tag{5}$$

2.3. Solution methods for the one-dimensional cutting stock problem

The one- dimensional cutting stock problem is a NP-complete problem. Therefore, there are two different approaches to solve it: exact methods and heuristics (Haessle *et al.* (1991). While heuristic methods give good enough solution in a reasonable amount of time, exact methods provide the global optimal solution. However, the cost of exactness usually means slow convergence which often results into high running times, especially with large problems. There different exact methods for solving one- dimensional cutting stock problem have been presented like Branch and Price Algorithm (ILOG, 2004) and Extended Cutting Plane Method (Westerlund *et al.*, 1998). Furthermore, different heuristic methods for solving one-dimensional cutting stock problem can be identified in the literature. The Delayed Column Generation (DCG) presented by Gilmore and Gomory (1961, 1963) is the first heuristic for one- dimensional cutting stock problem. Metaheuristic approaches are the center of interest for the researchers in this field in recent years. Wagner (1999) developed a genetic algorithm for a one-dimensional bundled cutting stock problem with contiguity in the lumber industry. Ostermark (1999) also presented a more complicated Genetic Hybrid Algorithm for solving one- dimensional cutting stock problem of Dyckhoff's type 1/V/D/R in the paper industry. Leung *et al.* (2001) applied a genetic algorithm and a simulated annealing approach to the two-dimensional non-guillotine cutting stock problem. Alvares-Vald´es *et al*. (2002) introduced a Tabu Search algorithm for large-scale guillotine in two-dimensional cutting stock problems of Dyckhoff's type 2/V/I/M. Ducatelle (2001) presented an Ant Colony approach to the cutting stock problem with and without contiguity. Eshghi and Javanshir (2005) developed a simple version of ACO for one dimensional cutting stock problem. This paper is a sequel to the simple version of ACO algorithm in which a revised version of ACO technique is presented to solve the one dimensional cutting stock problem.

3. A NEW VERSION OF ACO FOR ONE-DIMENSIONAL CUTTING STOCK PROBLEM

In this section, an algorithm based on Ant Colony Optimization (ACO) method is represented for solving one- dimensional cutting stock problem. ACO algorithm is an optimization algorithm based on ants' behavior. ACO was represented in the early 1990s by Dorigo, Maniezzo and Colorni (Doriago *et al.*, 1999) . This algorithm is inspired of by ants' social behavior. African ants have no eyesight and can find shortest way from food to their nest by leaving chemical materials called *pheromone* while moving. Other ants smelling the pheromone choose the path with the highest pheromone concentration. If two paths with different lengths between the nest and the food source exist, more pheromone will eventually accumulate on the shorter path because the distance is traveled more rapidly. Finally all the ants choose the shorter path (Dorigo *et al.*, 1999).

First ant colony optimization algorithm used in solving Traveling Salesman Problem (TSP) and then has been applied to different combinational optimization problems like quadratic assignment problems, routing problems, graph coloring problems, etc. In many optimization problems, ACO results indicate superiority of this method to the other metaheuristics. In ACO algorithm for one- dimensional cutting stock problem, we also use artificial ants. First, every artificial ant selects one of the orders according to a stochastic rule and then picks a cutting pattern by using another stochastic rule to satisfy the selected order.

3.1. Generating efficient cutting patterns

The method of generating cutting patterns in our algorithm is Pierce Method (Pierce, 1964) with slight modifications as follows:

In Pierce Method, we assume that ℓ_1 , ℓ_2 , … , ℓ_n are in a descending order and *M* is the maximum trim loss allowable.

Suppose that $\alpha_k = Int\left[(L - \sum_{i=1}^{k-1} \alpha_i \ell_i) \Big/ \ell_k\right]$ where *Int*[*g*] be an integer part of *g*. Now we have:

Step 1: If $L - \sum_{i=1}^{n} \alpha_i \ell_i \le M$, then $[\alpha_1, \alpha_2, ..., \alpha_n]$ is an efficient cutting pattern.

Step 2: If there is *i*, $1 \le i \le n\text{-}1$, such that $\alpha_i > 0$, then let *j* be the largest such *i* and go to step 4. If not, terminate the procedure. All efficient cutting patterns have been identified.

Step 4: Set $\alpha_j = \alpha_j - 1$. Then go to step 2.

Step 5: If there are two patterns $[P_{1k}, P_{2k}, ..., P_{nk}]$, $[P_{1j}, P_{2j}, ..., P_{nj}]$ such that for all *i* we have $P_{ik} \le P_{ij}$, then the cutting patter $[P_{1k}, P_{2k}, ..., P_{nk}]$ will eliminate from further consideration.

3.2. Probabilistic rules for selecting orders and cutting patterns

There are two probabilistic laws in our ACO algorithm. It is necessary to mention that the number of artificial ants in any iteration is a variable depending on the remaining amount of demand. In the proposed algorithm, first one of the artificial ants selects one of the demand orders by using a probabilistic rule as follows:

Suppose that we want to start iteration *t* of the algorithm and P_i is the total production thus far of demand order *i*. Then $(D_i - P_i)$ indicates the remaining demand for order *i* if $(D_i - P_i > 0)$ or if we define M_i to indicate this parameter, we have:

$$M_i = \begin{cases} D_i - P_i & \text{if } D_i - P_i > 0 \\ 0 & \text{if } D_i - P_i \le 0 \end{cases} \tag{6}$$

If the total remaining of all demands is shown by *TM*, then we also have:

$$TM = \sum_{\forall i} M_i \tag{7}$$

An artificial ant will select the demand order *i* at the current stage of the algorithm $(Pr^{(i)})$ according to the following probabilistic rule:

$$Pr^{(i)} = \frac{M_i}{TM} * \{ (1 - \frac{\tau_i^{(t-1)}}{\sum_i \tau_i^{(t-1)}})\}^{\alpha} \quad 0 < \alpha \le 1 \tag{8}$$

where $\tau_i^{(t-1)}$ is the amount of pheromone deposited on each demand order *i* in the last iteration. The process of finding $\tau_i^{(t-1)}$ will discuss later. In fact, the reason for using the latter term in relation (8) is to reduce the chance of selecting a demand order with high surplus production in the last iteration in this iteration and α is an importance coefficient of this factor. We assume that α is equal to 0 at the first iteration. The artificial ants will continue to enter one after another and select different orders until *TM* reaches zero. In this case a feasible solution is found and the current iteration of the algorithm is terminated and the next iteration will start.

After selecting a demand order according to the above rule, each ant relation must also choose one of the cutting patterns to produce the selected order. It is obvious that only cutting patterns with production of the selected order must be consider in this process. Now suppose that we define N_j as follows:

N_j : The number of selections of cutting pattern *j* by ants in the previous iteration.

Then the probability of picking pattern *j* for the selected order *i* by artificial ant $(Pr_{(j)})$ is defined as follows:

$$Pr_{(j)} = \frac{\eta_{(j)}}{\sum_{\forall j \in \Omega_i} \eta_{(j)}} \ , \ \Omega_i = \{ j \mid \forall P_{ij} > 0 \text{ , for order } i \}, \ \eta_{(j)} = \{\frac{1}{N_j . S_j}\}^{\gamma} . \{P_{ij}\}^{\beta} \quad (9)$$

In the above rule, $N_j . S_j$ indicates the total loss resulted from selecting the cutting pattern *j* in the previous iteration. Furthermore, β and γ are pattern factors and we assume that $\beta \leq 1$ and $1 \leq \gamma$ .In order to have a nonzero value for the cutting patterns with no selection in the previous iteration; we also assume that in this case $N_j . S_j$ has a small value.

3.3. General structure of algorithm

After each ant selects the cutting pattern, the necessary calculations will perform to update the values for the next ant. The number of ants in any iteration of the algorithm is a variable which controls by the value of *TM*. Suppose that at the iteration *t* of algorithm *TM* = 0. In this case a feasible solution $X^t = (x_1^t, x_2^t, \ldots, x_j^t, \ldots, x_m^t)$ is obtained where x_j^t is the number of times cutting pattern *j* is used by ants in iteration *t*. Then the amount of pheromone deposited on each demand order i, $\tau_i^{(t)}$, is calculated as follows:

$$\tau_i^{(t)} = \sum_{j=1}^{m} (P_{ij} \ x_j^t - D_i) \ , \ i = 1,\ldots,n \quad (10)$$

In fact, $\tau_i^{(t)}$ is the total amount of surplus production of order *i* in the current iteration. After finding X^t, algorithm will perform a "Local Search Procedure" to locally optimize X^t. The main step in this procedure is to find out whether it is possible to decrease the value of each x_j^t and still have a feasible solution. If a better solution than X^t is found by "Local Search Procedure" then X^t will be replaced and the updating process of pheromones will be applied. It is also necessary to mention that the best solution thus far generated by algorithm will always being kept and if the algorithm can not generate a better solution in compare with the best solution during the preset number of iterations, it will be terminated.

The pseudo-code code for the presented ACO algorithm for solving a one-dimensional cutting stock problem can be summarized as follows:

```
Procedure ACO_ cutting_ stock
    Initialize
        Data Entry: L , ℓ_i , D_i ; i = 1,...,n
        Cutting pattern generator: P_ij , S_j ; j = 1,...,m
    For  t=1 to [stopping criteria] do
        For  k=1  to [TM = 0]  do
            Each ant selects a demand order and a cutting pattern by equations (8) and (9)
            Update values of parameters
        End-For
      Apply "Local Search Procedure"
      Update "pheromone" on each order
      Update Best-Solution
    End-For
```

4. COMPUTATIONAL RESULTS

In order to test the efficiency of the proposed algorithm, the ACO algorithm was coded by *Visual Basic 6.0* and run on a personal computer Pentium 4, 2.8 GHZ, 518 MB RAM. Then 140 test problems were randomly generated and solved by the proposed algorithm. The set of test problems are the same as used in [5]. The test problems were categorized to 18 classes of one-dimensional cutting stock problem (TP1 -TP 18) according to different values for *m* and *n*. The number of random instances was set to 10 for the classes (TP1-TP10) and 5 for the classes (TP11-TP18). The procedure for generating the parameters of the test problems is as follows:

The values of L and m were treated as random variables taken from interval [20, 100] and [m_1 , m_2] respectively where the values of m_1 and m_2 are shown in the third column of Table 1 as the lower bound and upper bound on m respectively. The values of D_i were also random variables taken from interval [50, 400]. The values of the parameters α, β and γ used in the algorithm are also shown in Table 1. The number of runs in "stopping criteria" was set to 20 for each test problem.

The average of CPU time in seconds for solving each class of the test problems is also reported in Table 1 in the column denoted by "ACO Time". In order to compare the performance of our algorithm with other techniques, all test problems were also solved by "Column Generation" method. The average computational time of solving our test problems by this technique is also shown in the column denoted by "C.G. Time". In the last column, the percentage of time improvement by our new version of ACO algorithm in compare with the old one used in [5] is presented.

The first interesting result of our algorithm is that the average computational time of ACO algorithm for all instances is 47.19 seconds and it is almost 37 seconds faster than "Column Generation" method. Moreover, in all cases except one, the average computational time of our model is shorter than "Column Generation" method. As it is shown, the efficiency of our algorithm is more significant in medium and large scale classes (TP12-TP18). Furthermore, the best solutions found by ACO algorithm for all instances in the classes (TP1-TP15) are optimal solution. We have verified this result by solving all test problems in the classes (TP1-TP15) by the version of "Lingo Solver" which we had. Unfortunately, the optimal solution of many instances in the classes (TP16-TP18) cannot be found by the "Lingo Solver" due to large scale size of variables. On the other hand, the average relative error of the worst solution in all iterations of each test problem is less than 63% of the optimal solution. This result implies that the gap between the best and worst solutions found by our algorithm is not very huge.

As it was mentioned in the first section of this paper, most versions of one-dimensional cutting stock problem are NP-complete. Therefore, an optimal solution of this problem when n and m are increased can not be obtained in a reasonable and short amount of time. For this reason, an ACO algorithm presented in this paper is a very fast method for generating near optimal solution. An ACO algorithm can also be used very easily in practical cases by considering this fact that appropriate adjustment may be needed. In summary, the computational results show the efficiency and accuracy of the proposed algorithm in compare with the other techniques for solving one-dimensional cutting stock problem.

Table 1: Computational results obtained for18 groups of sample problems

%	C.G. Time	ACO Time	γ	β	α	m	n	TP
0	2	2	1	1	1	$4 \leq m \leq 6$	3	TP01
0	2	2	1	1	1	$7 \leq m \leq 15$	3	TP02
33	3	4	1	1	1	$8 \leq m \leq 15$	4	TP03
35	7	4.5	1	1	1	$16 \leq m \leq 30$	4	TP04
45	11	6	1	1	1	$8 \leq m \leq 20$	5	TP05
11	17	15	1	1	1	$21 \leq m \leq 40$	5	TP06
33	27	18	1	1	1	$41 \leq m \leq 70$	5	TP07
73	41	11	1	1	1	$10 \leq m \leq 30$	6	TP08
42	70	40	1	1	0.5	$31 \leq m \leq 60$	6	TP09
46	81	43	1	1	1	$61 \leq m \leq 110$	6	TP10
29	79	56	1.5	1	1	$25 \leq m \leq 70$	7	TP11
47	123	65	1.5	1	1	$71 \leq m \leq 150$	7	TP12
46	115	61	1.5	1	1	$30 \leq m \leq 80$	8	TP13
39	133	81	2	1	1	$81 \leq m \leq 165$	8	TP14
46	158	84	1.5	1	0.5	$30 \leq m \leq 90$	9	TP15
49	182	92	2	1	0.5	$91 \leq m \leq 170$	9	TP16
47	226	119	1.5	1	1	$30 \leq m \leq 90$	10	TP17
38	239	146	2	1	1	$91 \leq m \leq 180$	10	TP18

5. CONCLUSION

Although many different methods for solving the one-dimensional cutting stock problem exist, only a limited number of them are applicable to the large problems. In this paper a new metaheuristic method based on Ant Colony Optimization (ACO) technique presented to solve the one-dimensional cutting stock problem. In this algorithm, according to some probabilistic rules, artificial ants can select the cutting patterns and generate a feasible solution. Computational results show the high efficiency of the proposed algorithm for solving one- dimensional cutting stock problem.

One of the advantages of using our algorithm rather than exact methods such as integer programming in solving one-dimensional cutting stock problem is the simplicity of applying it to a large scale problem. As it was shown in our test problems, even for large instances, our algorithm easily found the near optimal solution in a short amount of time. Nonetheless, our model itself has many limitations arising from its construction as an algorithm for solving only the *1/V/O/R* case in Dyckhoff classification.

Because this paper is an early step in the study of developing a metaheuristic algorithm for the cutting stock problem, it is also necessarily restricted to simple assumptions. For example, it may be that in some circumstances the selection of the demand order by an artificial ant might be better described by a new probabilistic rule. To see whether our algorithm is also a powerful method in more practical cases, further research is needed. It could be worthwhile to study further the possibility of changing the probabilistic rules used in this article. Another interesting area of further study would be the implementation of proposed ACO method for solving the different types of one-dimensional cutting stock problem or cutting stock problem with higher dimensions. Given the growing importance of cutting stock problem in industry, such research would be of great practical significance.

6. REFERENCES

1. Alvares-Vald´es, R., Paraj´on, A. and Tamarit, J. M. (2002). A Tabu Search Algorithm for Large-scale Guillotine (un) Constrained Two-dimensional Cutting Problems. Computers & Operations Research, 29: 925–947.
2. Dorigo, M. and Di Caro, G. (1999). Ant Colony Optimization: A New Meta-heuristic. Proceedings of the 1999 Congress on Evolutionary Computation.
3. Ducatelle, F. (2001). Ant Colony Optimisation for Bin Packing and Cutting Stock Problems, Master of Science in Artificial Intelligence, University of Edinburgh.
4. Dyckhoff, H. (1990). A Typology of Cutting and Packing Problems. European Journal of Operational Research, 44: 145-159.
5. Eshghi, K. and Javanshir, H. (2005), An ACO algorithm for one-dimensional Cutting Stock Problem, Journal of Industrial Engineering International,1(1), 10-19.
6. Gilmore, P. C. and Gomory, R. E. (1961). A Linear Programming Approach to the Cutting Stock Problem. part I, Operations Research, 9: 849-859.
7. Gilmore, P. C. and Gomory, R. E. (1963). A Linear Programming Approach to the Cutting Stock Problem, part II, Operations Research, 11: 863-888.
8. Haessler, R. W. and Sweeney, P. E. (1991). Cutting Stock Problems and Solution Procedures. European Journal of Operational Research, 54: 141-150.
9. ILOG (2004). ILOG CPLEX 8 User's Manual, France, ILOG.
10. Karelahti, J. (2002). Solving the Cutting Stock Problem in the Steel Industry. Master Thesis in Engineering Physics and Mathematics, Helsinki University of Technology, Finland.
11. Leung, T.W., Yung , C.H., Troutt , M.D. (2001). Applications of Genetic Search and Simulated Annealing to the Two-dimensional Non-guillotine Cutting Stock Problem. Computers and Industrial Engineering, 40(3): 201-214.
12. Pierce, J. F. Jr. (1964) Some Large-Scale Production Scheduling Problems in the Paper Industry. Prentice-Hall.
13. Suliman, S. M. A. (2001). Pattern Generating Procedure for the Cutting Stock Problem. International Journal of Prduction Economics, 74: 293-301.
14. Umetani, S., Yagiura, M. and Ibaraki, T. (2003). One-dimensional Cutting Stock Problem to Minimize the Number of Different Patterns. European Journal of Operational Research, 146: 388-402.
15. Wagner, B. J. (1999). A Genetic Algorithm Solution for One-dimensional Bundled Stock Cutting. European Journal of Operational Research, 117: 368-381.
16. Westerlund, T., Skrifvars, H., Harjunkoski, I., Pörn, R. (1998). An Extended Cutting Plane Method for a Class of Non-convex MINLP Problems. Computers and Chemical Engineering, 22:357-365.
17. Östermark, R. (1999). Solving a Nonlinear Non-convex Trim Loss Problem with a Genetic Hybrid Algorithm. Computers & Operations Research, 26: 623–635.

BIOGRAPHICAL SKETCH

Kourosh Eshghi is a Professor of Industrial Engineering Department at Sharif University of Technology, Tehran, Iran. He received his Ph.D. in Operations Research from the University of Toronto in 1997. His research interests include graph theory, integer programming and combinatorial optimization. He is the author of 2 books and over 50 journal papers.

Hasan Javanshir is an Assistant Professor of Industrial Engineering Department at Islamic Azad University, South Tehran Branch. He received his Ph.D. in Industrial Engineering from Islamic Azad University in 2006. His research interests include mathematical programming, transportation planning and applications of operations reserach. He is the author of 2 books and over 40 journal papers.