

## USING NETWORK STRUCTURAL COMPRESSION IN BRANCH AND BOUND ALGORITHM FOR SOLVING THE PROJECT SCHEDULING PROBLEM WITH RESTRICTED RESOURCES

**H.R. Moradi<sup>1</sup>, A. Khayati<sup>2</sup>**

<sup>1</sup>Department of Mathematics, Qaemshahr, Islamic Azad University, Qaemshahr, Iran

<sup>2</sup>Department of Mathematics, Ferdowsi University, Mashhad, Iran

### ABSTRACT

*Project scheduling problem is the generalized mode of the well-known problem of "workshop on industrial orders"; hence it can be categorized in class of complex problems or the NP class. In such problems  $|J|$  is the current activity each of which can be conducted in a fixed period of time  $d_j$  ( $j = 1, \dots, |J|$ ) with or without any break in their time period during performance and with considering two restrictions of prerequisites and resources. One of the solutions for problems of project scheduling with limited resources is the method of branch and bound.*

*In this research, through studying the branch and bound method, we get use of some mechanisms to conduct this method so to increase its efficiency as much as possible. For this purpose, we solve the problem in two phases. In the first phase we try to have structural compression in the network. In this way the following three regulations are applied:*

1. *Eliminating those activities which are not in accordance with any other activity in the network*
  2. *Eliminating the two activities which are in accordance with each other (activity i is only in accordance with activity j and activity j is only in accordance with activity i)*
  3. *Eliminating the activity which is only in accordance with another activity*
- (The two activities which simultaneous performance of them may violate restriction of resources and or prerequisites are called incompatible)*

*In the second phase we conduct the branch and bound method according to compressed network; it is worth mentioning that in calculating bound of the nodes we try to improve the amount of bound through finding incompatible activities and in this way we attain more efficient performance of branch and bound.*

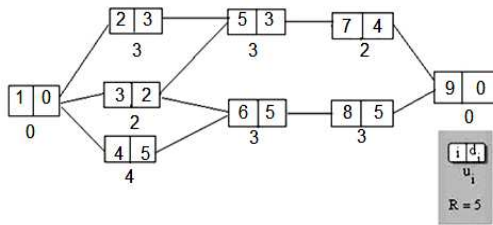
**Keywords:** *The resource constrained project scheduling problem, Lower bound, Compressed the structure of the network, Branch and bound algorithm.*

### The general mode of the problem and the project network

The general mode of problem of resources allocation is as follows:

In every project there are some activities  $i = \{1, \dots, n\}$  which are dependent on each other and each of them has a time period  $d_i$  which has to be performed in order to fulfill the project. There are some precedence relationships between activities. We show these precedence relationships in the project network between the activities of a project with a curve between both activities which have such relations with each other.

Also a project has restriction of resource  $k = \{1, \dots, m\}$ , and in every period of time, maximum amount of  $R_k$  of every kind of resource is available. Every activity in each period of time requires  $r_{ik}$  resource from each kind of  $k$  resource which all these restrictions can be monitored in the project network. The following example indicates the project network with  $n=9$  activity which has a resource of  $(k=1)$  and maximum available resource is  $R_1 = 5$ . The activities 1 and 9 (virtual activities) represent the beginning and the end of project with 0 period of time.



Example (1)

**1. Calculating lower bound by getting use of precedence relationships among activities (LB1)**

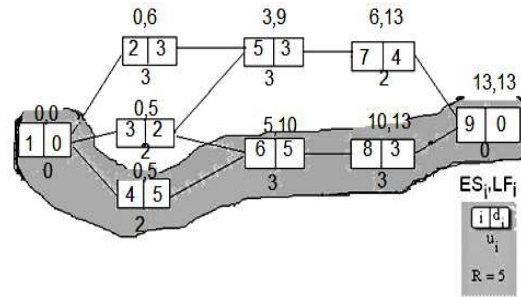
In project scheduling problem with restricted resources, for calculating this bound, the restriction component of the resources must be eliminated from the problem and according to limitlessness of resources, a time for beginning and finishing each activity and finally a time for fulfillment of the whole project must be calculated. In calculating this bound, only component of precedence is regarded between the activities. It is quite obvious that, this bound represents the reality that if there is no component except the precedence relationship on the network, the project cannot be performed in a time less than LB1.

While calculating LB1, the amounts of the latest and the earliest starting time of each activity ( $ES_j/LS_j$ ) as well as the earliest and the latest fulfillment time of each activity ( $EF_j/LF_j$ ) must be calculated; the duration of  $[ES_j, LF_j]$  is called time window of activity  $j$  and we know that LB1 is the critical path method (C). These amounts can be calculated through the following algorithm:

$$\begin{aligned}
 ES &= EF_1 = 0 \\
 ES_j &= \max\{EF_i | i \in P_j\}; \\
 EF_j &= ES_j + d_j \quad \text{for } j = 2, \dots, n. \\
 LF_n &= LS_n = LB1 \\
 LF_j &= \min\{LS_h | h \in F_j\}; \quad LS_j = LF_j - d_j \\
 &\text{for } j = n - 1, \dots, 1.
 \end{aligned}$$

$P_j$ = the set of activities which are prerequisite of activity  $j$   
 $F_j$ = the set of activities which are post-requisite of activity  $j$

For example (1) critical path and calculations are done as follows:



**2. Calculating lower bound by getting use of resources restriction (LB2)**

The same as previous method, for calculating the lower bound of the project scheduling problem with restricted resources, this time we consider another method for facilitating the problem. In this method for the sake of calculating lower bound we only consider that resources restriction is existed and we do not consider precedence restrictions among activities. As we know, for performing every activity in each period of time we require specified amount of resources, and we also know that the amount of existence of the resources in each period of time is a fixed amount. This bound represents the point that, if we only have restriction of resources, due to the fact that every activity uses especial amount of resources, we cannot fulfill the project in shorter time through lower bound of LB2.

This required amount of resource can be calculated through multiplying the amount of resource of each activity by its time and dividing the sum of attained amounts by the existed amount of resource in each period of time. Considering this bound for the problems which require several resources is equal to finding maximum amount between calculated amounts for each resource. The following equation embraces all points mentioned above:

$$\text{Max}_{all k} \left\{ \left( \sum_{all j} r_{kj} d_j \right) / R_k \right\}$$

Where  $k$  is resource index,  $j$  is unfulfilled activities index,  $R_k$  is the amount of

availability in resource k,  $r_{kj}$  is an amount from resource k which activity j requires that, and finally  $d_j$  is the remaining time for fulfilling activity j. For the example of (1) we have:

$$LB2 = \lceil 74/5 \rceil = \lceil 14.8 \rceil = 15$$

**3. Calculating lower bound by getting use of combination of resources restriction and restriction of precedence relationship between activities (LB3)**

Before expressing how to calculate this lower bound we need to express some definitions.

Incompatibility: we call activity j incompatible with activity h, in case we cannot perform these two activities simultaneously with each other; this incompatibility may occur due to precedence relationship between activities j and h, i.e., for instance activity j is prerequisite of activity h, or simultaneous performance of activities j and h may cause the amount of the used resource to be more than the available amount of the resource.

Maximum of performable duration: suppose that through the first method (LB1) lower bound is calculated for the project network and critical path (C) is estimated. Now consider activity j which is not a member of critical path; activity j needs to be performed in parallel with critical path in time window  $[ES_j, LF_j]$  in duration  $d_j$  and through amount of  $r_{kj}$  (from the resource r); for this activity we define an amount of  $e_j^{max}$ . The amount of  $e_j^{max}$  is equal to the duration which activity j can be performed in time window  $[ES_j, LF_j]$  according to resources restriction. In other words, it represents the number of days which activity j in its time window- by having enough resources- can be performed in parallel with critical path. Therefore the amount of  $e_j^{max}$  is always  $0 \leq e_j^{max} \leq LF_j - ES_j$ . If  $e_j^{max}$  is greater than or equal to  $d_j$ , it means that activity j can be performed in its time window and activity j is in accordance with all activities of critical path; otherwise

activity j cannot be performed in its time window completely. As a result, the time of  $d_j - e_j^{max}$  will be remained from activity j, and this time must be added to lower bound of LB1. Now because we want to have the maximum increase for lower bound, we define LB3 as the following:

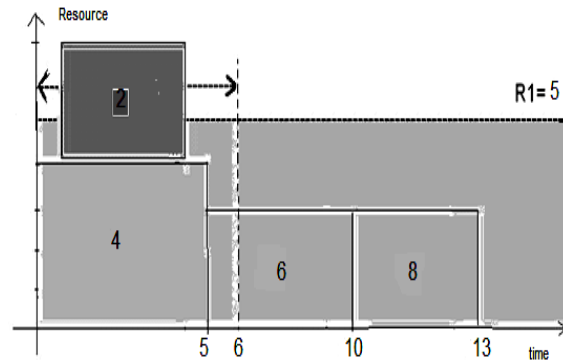
$$A = \max \{ d_j - e_j^{max} \mid j \notin C \}$$

$$LB3 = LB1 + \max \{ 0, A \}$$

The example of required calculations and amount of LB3 is presented in the following table and figure.

The table of calculating maximum of performable duration for the example (1)

Job	2	3	5	7
$d_j$	3	2	3	4
$ES_j$	0	0	3	6
$LF_j$	6	5	9	13
$e_j^{max}$	0	0	0	7



Incompatibility of activity 2 in the example (1)

$$e_2^{max}, d_3 - e_3^{max}, d_5 - e_5^{max}, d_7 - e_7^{max} \}$$

$$LB3 = LB1 + \max \{ 0, \max \{ d_2 = 13 + \max \{ 0, \max \{ 3, 2, 3, -3 \} \} \} \} = 13 + 3 = 16$$

**Structural compression of the project network**

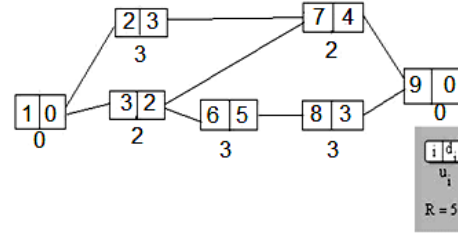
In 1978, Nicos Christofides and R. Alvarez-valdes and J.M. Tamarit presented a method for structural compression of the network for the sake of calculating lower bound. In this section we will deal with the method of structural compression of the network. Of course in this research we get use of this

method for improving the method of branch and bound, not for calculating lower bound. As we know, in order to solve the project scheduling problem while having restricted resources through branch and bound method, the number of produced branches within the process of problem-solving is a function of number of activities of the project. Therefore if we can decrease the number of activities of the project network, the number of produced branches will be decreased as well and we can decrease the duration of reaching to optimized solution. This structural compression of network can be performed through the following methods:

**An activity incompatible with all activities**

Suppose that in the project network, there is a  $j$  activity, in a way that according to restriction of resources or restriction of precedence, it cannot be performed simultaneous with any other activity. In this case we can delete it from the project network without having any problem in generalities of the issue, and we can add duration of activity  $j$  to calculated lower bound of the new network. The time complexity of finding these kinds of activities in the project network is from class  $o(n)$ ; this is because it is required to study all activities which their sum is equal to  $n$ . This kind of activity is called block activity. In the next chapter we will discuss the use of network structural compression method in the process of solving problems through branch and bound method.

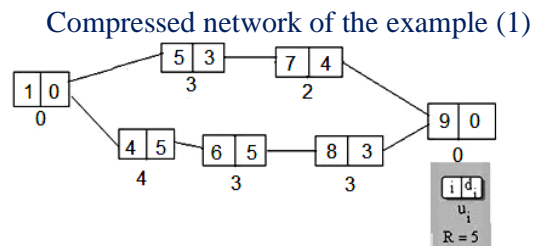
For the example (1), activities 4 and 5 have this feature which cannot be performed simultaneous with any other activity; therefore the compressed network can be achieved as follows:



Compressed network of example (1)

**The two activities which can be blocked**

a) Suppose that in the project network, activity  $j$  – due to restriction of resources or precedence restriction – can only be performed simultaneous with activity  $i$ ; also activity  $i$  has the same condition (it can only be performed simultaneous with activity  $j$ ); in this way, we can regard activities  $i$  and  $j$  as a blocked activity and we can perform as previous style. In such a condition the duration of new-block activity is considered as duration of an activity which is bigger ( $d_i, d_j$ ). The time complexity of such activities in the project network is from class  $o(n^2)$ ; this is because two nested searching loops (equal to the number of activities  $n$ ) are required for finding these kinds of activities and their combination. Another mode which can be regarded is that, three  $i, j$  and  $k$  activities have such condition, which means that they can only be performed together; in such situations, we can also operate the same as the mode of the two-activity, but because time ranking of these kinds of activities might go up, and the possibility of existence of such activities may be decreased, we ignore using this mode. In example (1), activities 2 and 3 have such conditions; therefore the new network can be demonstrated as follows:

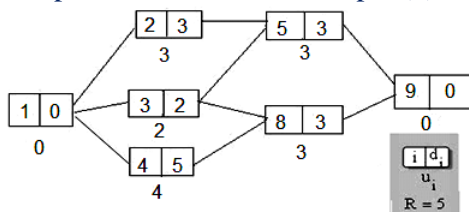


Compressed network of the example (1)

b) Imagine that in the project network, activity  $i$  – due to restriction of resources or precedence restriction – can only be performed simultaneous with activity  $j$ ; but activity  $j$  except with activity  $i$  can be performed simultaneous with another activity like  $k$ ; in this situation if  $d_j > d_i$  we can consider activities  $i$  and  $j$  as a blocked activity and we can operate the same as the first mode. In such condition,  $d_i$  is regarded as the time of new-block activity, but in the case  $d_j < d_i$  it is not possible to consider activities  $i$  and  $j$  as a block activity; because it is possible that in the created duration, no activity be performed; this is while if these two activities were not combined in such duration, an activity could be performed. The time complexity of such activities in the project network is from class  $o(n^2)$ ; this is because two nested searching loops (equal to the number of activities  $n$ ) are required for finding these kinds of activities and their combination.

In the example of figure (1), activity 6 can only be performed simultaneous with activity 7, but activity 7 can also be performed simultaneous with activity 8; yet because the time duration of activity 6 is more than that of performance in activity 7 we can regard these two activities as block activities with time duration  $d_6=5$ ; therefore the network of new project can be presented as the following figure:

Compressed network of example (1)



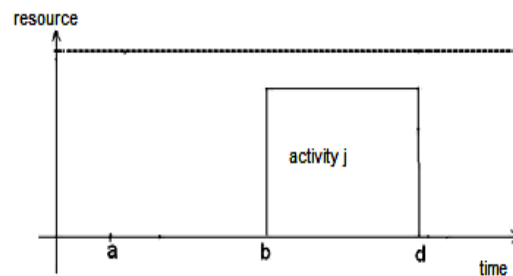
**Proposition**

Suppose that in problem RCPSP there is an activity which cannot be performed simultaneous with another activity due to restriction of resources or precedence

restriction (in other words, there is a blocked activity). In this case if we schedule this activity in the first time it is allowed to be scheduled, there would be no harm conducted on the optimized solution.

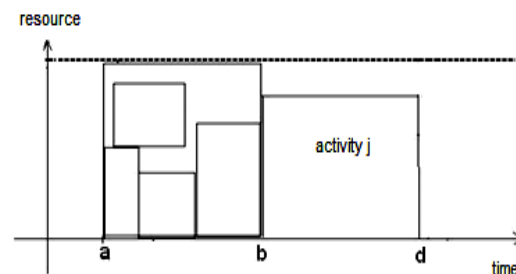
**Proof**

To prove the proposition, assume that we have the optimized solution of RCPSP problem, and activity  $j$  applies in the proposition conditions. Activity  $j$  is scheduled with time length  $c$  and time  $b$ , and time  $a$  is the first time that activity  $j$  was allowed to be scheduled.

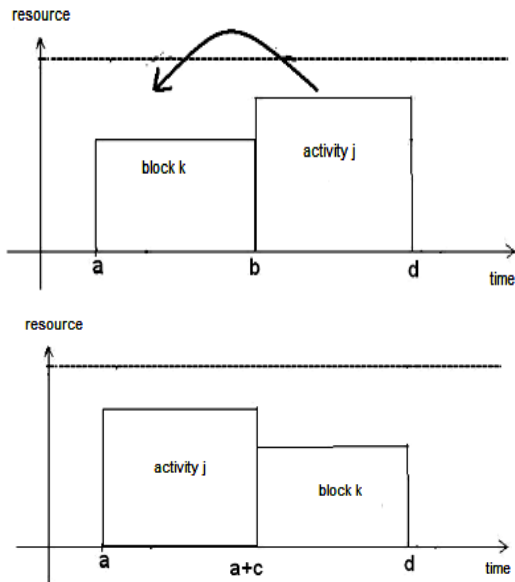


If  $a=b$ , then activity  $j$  in the optimized solution has been allowed to schedule at the first time.

Consider that this is not the case; then from time  $a$  to  $b$ , some activities are scheduled. If we consider all these activities as a block  $k$  as the following:



since activity  $j$  at time  $a$  was performable, there is no precedence relationship between activity  $j$  and block  $k$ , i.e. they are not correlated. Therefore, we can interchange activity  $j$  and  $k$ .

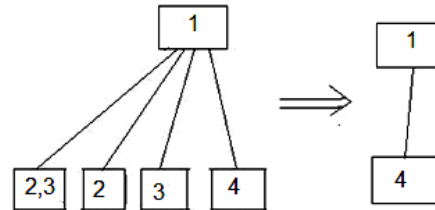


Therefore, the proposition is proved. Now considering the stated proposition, this method is expressed as follows. Imagine that during the process of structural compression of the network method, set A is the set of activities which has been achieved from the method of section 1-4, which includes the set of activities which are a block activity themselves; also set B is the set of activities which have been attained from the method of section 2-4, which means the set of activities which have become a block activity through being combined with another activity. Now suppose that during the process of problem solving of RCPSP by branch and bound method we want to select a node; hence we would have the following modes:

**Existence of an activity which is block**

If in this step the activity  $j \in A$  exists and is allowed to be scheduled, since activity j is a block activity, according to the previous point we can allocate time to activity j at the first performable time. In such condition the node which includes activity j is selected and the other nodes are eliminated. For the example of figure1, activities 2,3 and 4 are the activities which are permitted to be scheduled, and since activity 4 is a block activity and (due to lack of resources)

it cannot be performed simultaneously with the other activities, in this step, node 4 is selected, which means the branch is selected like the following mode and other modes are trimmed.



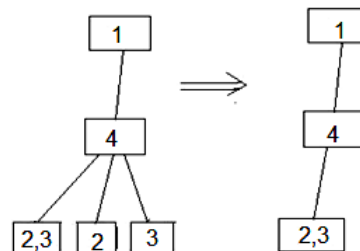
Selecting the node which includes block activity

**Existence of activities which have formed one block**

In this mode, imagine that we are in 4<sup>th</sup> step of branch and bound method and there is  $j \in B$  activity which is permitted to be scheduled. In this situation since the activity is  $j \in B$ , activity k is existed in the network which can be seen as a block activity through being combined with activity j. In this condition one of the two possible modes may take place:

1. Activity k the same as activity j is an activity which is allowed to be scheduled in this step; in this situation, the combination of activities j and k must be considered as the selected branch and other branches will be trimmed.

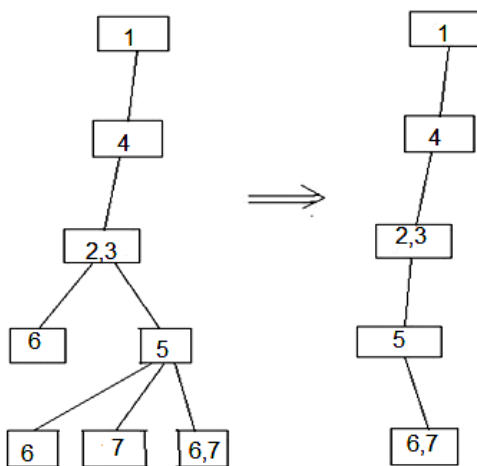
For the example of figure1, we know that activities 2 and 3 are those activities which can be combined with each other and become blocked as one activity; both of these activities are also allowed to be performed, therefore the target branch can be selected as the following:



The figure of selecting the node which includes block activity

2. If activity  $k$  is not allowed to be scheduled, in this situation we do not schedule activity  $j$  and we trim the branch which includes activity  $j$ . This is because we know that if branching process is continued we reach a mode in which activities  $j$  and  $k$  can be simultaneously allowed to be scheduled. The reason is that activities  $j$  and  $k$  have been able to be combined with each other during the process of structural compression of network and they have been able to create a block; therefore by continuing the process of branching we may reach the former mode.

For the example of figure1, we know that activities 6 and 7 are those activities which can be combined with each other and become blocked as one activity. During the process of problem-solving after scheduling the activities 2 and 3, activity 6 would be allowed to be scheduled, but activity 7 is not still allowed to be so. Therefore, in this situation we trim the branch of activity 6 and select branch 5. We witness that in the next step, activities 6 and 7 are allowed to be scheduled and we can act as the previous mode. Therefore, the target branch can be selected as the following:



Selecting the node which includes block activity

## The comparison method

As it was already mentioned, our procedure in evaluating operation of any method is comparing number of nodes which each of them produce when implementing branch and bound method, and the related counting tree. Of course the time which has been spent by computer processor – for reaching an appropriate answer – can be considered as a criterion for comparison, but the main factor for comparison is the number of produced branches.

## The results of the tests

In this section the results of the numeric tests have been presented for evaluating the function of suggested algorithm with the other algorithms. For this purpose, and for programming, we have got use of Matlab7 software and for the sake of conducting the programs, a PC with the features of 2 GH CPU, Pentium IV and main memory of 1 Gigabyte have been used. For every problem of the tests, four algorithms have been separately conducted and the results have been presented. These algorithms are called A, B, C, and D in the following. The description of each of them is presented in the following:

A: this algorithm includes the branch and bound method in which for the sake of calculating bound, we get use of LB1 method. The LB1 method has already been described in chapter 3.

B: this algorithm includes the branch and bound method in which for the sake of calculating bound, we get use of LB2 method. The LB2 method has already been described in chapter 3.

C: this algorithm includes the branch and bound method in which for the sake of calculating bound, we get use of LB3 method. The LB3 method has already been described in chapter 3.

D: in this algorithm we have utilized structural compression of the network and lower bound of LB3 in branch and bound algorithm. In table 1, the obtained results are presented:

Table 1: examples of implementing the stated algorithms

Optimized solution	Number of nodes	Processor time	Algorithm type	Problem
43	5127	669.46	A	A1
	4926	581.23	B	
	4681	520.12	C	
	4434	445.32	D	

Table 1: examples of implementing the stated algorithms

Optimized solution	Number of nodes	Processor time	Algorithm type	Problem
38	1468	131.9	A	A2
	1430	118.1	B	
	1396	102.44	C	
	1215	86.6	D	
72	55	3.99	A	A3
	57	4.2	B	
	48	3.72	C	
	39	3.31	D	
49	56	3.25	A	A4
	63	3.68	B	
	45	2.83	C	
	43	2.66	D	
53	26065	8395.8	A	A5
	27847	9112.37	B	
	24320	7533.3	C	
	22336	6841.26	D	

Table 1: examples of implementing the stated algorithms

Optimized solution	Number of nodes	Processor time	Algorithm type	Problem
59	10591	1305.9	A	A6
	10911	1412.46	B	
	9439	1006.29	C	
	8561	827.6	D	
55	60	3.22	A	A7
	56	3.03	B	
	52	2.88	C	
	41	2.65	D	
44	53	3.15	A	A8
	49	3.01	B	
	53	3.15	C	
	43	2.73	D	
58	3383	351.1	A	A9
	3410	383.64	B	
	3112	303.36	C	
	2778	251.9	D	

Table 1: examples of implementing the stated algorithms

Optimized solution	Number of nodes	Processor time	Algorithm type	Problem
58	25463	7619.4	A	A10
	25323	7016.43	B	
	23671	6168.1	C	
	20408	5373.81	D	
54	1062	77.61	A	A11
	1093	79.91	B	
	953	70.61	C	
	807	59.4	D	
47	60	3.24	A	A12
	62	3.96	B	
	60	3.24	C	
	43	2.61	D	
46	77	3.99	A	A13
	69	3.52	B	
	65	3.08	C	
	58	2.91	D	

### Conclusion

As you can see in table 5-1, number of nodes in algorithm C has decreased in comparison with algorithms A and B. We know that in algorithm C, lower bound of LB3 has been utilized. According to the results of experiments we can see that, this lower bound in comparison with lower bounds of LB1 and LB2 – which have been presented in algorithms A and B – have caused more trimming in useless nodes and subsequently have caused decrease in the number of produced branches in the tree of the optimized solution. This is because in lower bound of LB3, both

precedence restriction between activities and resources limitation have been regarded.

According to table 5-1, by comparing acquired results from conducting algorithms A, B, C and D we realize that the number of nodes in algorithm D compared to other algorithms have been decreased. Averagely this decrease in the number of nodes compared to algorithm A has been 22%, compared to algorithm B, 21%, and compared to algorithm C it has been 13%. Therefore, according to our first guess, simultaneous usage of structural compression network and lower bound of LB3 during the process of problem-solving through branch and bound method can decrease the number of produced nodes.

### References

- Baar, T., Brucker, P., Knust, S. (1998).** Tabu-search algorithms and lower bounds for the resource- constrained project scheduling problem, in: S. Voss, S. Martello, I. Osman, C. Roucairol (Eds.), *Meta-heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer, Dordrecht, pp. 1-18.
- Bactor, F.F. (1996).** Resource-constrained project scheduling by simulated annealing, *International Journal of Production Research* 34(8), 2335–2351.
- Chen, R.M., Lo, S.T. (2006).** Using an Enhanced Ant Colony System to Solve Resource- Constrained Project Scheduling Problem, *IJCSNS International Journal of Computer Science and Network Security*, VOL.6 No.11.
- Christofides, N. Alvarez-Valdes, R. Tamarit, J.M. (1987).** Project scheduling

with resource constraints: A branch and bound approach, *European Journal of Operational Research* 29, 262-273.

**Conway, R., Maxwell, W., Miller, L.W. (1967).** *Theory of Scheduling*, Addison-Wesley Reading, MA.

**Davis, L. (1985).** Job shop scheduling with genetic algorithms, in: *Proceedings of the 1<sup>st</sup> international conference on genetic algorithm*.

**Demeulemeester, E. Herroelen, W. (1992).** A branch-and-bound procedure for multiple resource-constrained project scheduling problem, *Management Science* 38, 1803-1818.

**Klein, R. Scholl, A. (1999).** Computing lower bounds by destructive improvement: An application to resource-constrained project scheduling, *European Journal of Operational Research*, 112, 322-346.

**Kolisch, R. (1995).** *Project Scheduling Under Resource Constraints, Efficient Heuristics for Several Problem Cases*. Physica-Verlag, Wurzburg.

**Kolisch, R., Hartmann, S. (2006).** Experimental investigation of heuristics for resource-constrained project scheduling: An update, *European Journal of Operational Research* 174, 23-37.

**Igelmund, G., Radermacher, F.J. (1983).** Algorithmic approaches to preselective strategies for stochastic scheduling problems, *Networks* 13, 29-48.

**Merkle, D., Middendorf, M., Schmeck, H. (2002).** Ant colony optimization for resource-constrained project scheduling. *IEEE Transactions on Evolutionary Computation*, Vol. 6, Issue: 4 On page(s): 333- 346.

**Mingozzi, A., Maniezzo, V., Ricciardelli, S., Bianco, L. (1998).** An exact algorithm for the resource-constrained project scheduling based on a new mathematical formulation, *Management Science* 44, 714-729.

**Nonobe, K., Ibaraki, T. (2001).** Formulation and tabu search algorithm for the resource constrained project scheduling problem, in: C.C. Ribeiro, P. Hansen (Eds.), *Essays and Surveys in Metaheuristics*, Kluwer Academic Publishers, pp. 557-588.

**Stinson, J.P., Davis, E.W., Khumawala, B.M. (1978).** Multiple resource-constrained scheduling using branch and bound, *AIIE Transactions* 10, 252-259.