JOURNAL OF APPLIED QUANTITATIVE METHODS

A LOWER BOUND FOR PROJECT COMPLETION TIME ATTAINED BY DETAILING PROJECT TASKS AND REDISTRIBUTING WORKLOADS

Zohar LASLO[1]

PhD, Associate Professor, Department of Industrial Engineering and Management,
Sami Shamoon College of Engineering, P.O.Box 950, 84100, Beer-Sheva, Israel

E-mail: zohar@sce.ac.il

Hagai ILANI[2]

PhD, Senior Lecturer, Department of Industrial Engineering and Management,
Sami Shamoon College of Engineering, P.O.Box 950, 84100, Beer-Sheva, Israel

E-mail: hagai@sce.ac.il

Baruch KEREN[3,4]

PhD, Senior Lecturer, Department of Industrial Engineering and Management,
Sami Shamoon College of Engineering, P.O.Box 950, 84100, Beer-Sheva, Israel

E-mail: baruchke@sce.ac.il

Abstract: *We evaluate the possible benefit from improving the workload distribution in a directed acyclic graph (DAG) (e.g. a project network), by determining a lower bound for the project completion time. It is shown that a lower bound can be obtained by equally distributing the workload over the max-cut in the graph which separates the nodes 1 and n. It is also shown that for a complete n-node DAG, practitioners can quickly compute a lower bound for the project completion time. The max-cut can be found by any linear programming algorithm or by reducing the problem to the problem of finding a maximum matching in a bipartite graph. Our results can help planners and project managers to characterize the "ideal case" in which the optimal workload distribution among the arc networks minimizes the project completion time. That can be done for a non-complete DAG as well as for a complete one. A lower bound can then be used to evaluate the maximum potential for reducing the project completion time in real-life cases.*

Key words: *Project network; Directed acyclic graph (DAG); Max-cut; Maximum matching, Bipartite graph*

1. Introduction

A reduction in project completion time is one of the main missions in project management and in similar processing contexts. We evaluate the possible benefit from improving the workload distribution in a directed acyclic graph (DAG) (e.g. a project network), by determining a lower bound for the project completion time. Revealing the lower bound of a project's completion time is critical for managers, as the lower bound gives significant information about the potential value for investing effort and management time for reducing the project completion time. If the planned processing time of a given plan/project is near the lower bound, the mangers can deduce that the potential reduction is small (large), and by that decide if it is worthwhile to spend effort and time in trying to accelerate the project.

The objective of shortening the project completion time can be attained by additional budget and/or by redistribution of workloads, the latter being more economical but less feasible. The current common methodologies for shortening the project completion time are based on investment of additional budget in the critical path activities in deterministic models [1-2][5] or in the high criticality activities in stochastic models [3-6]. The procedures for optimal redistribution of budget among project activities ([1- 6]) enable the minimizing of the project completion time under budget constraints. The precedence among the project activities as well as the activities "crash" durations determine a lower bound for the project completion time. It is obvious that releasing the budget constraints by bringing all the activity execution times to “crash” paces, i.e., maximal execution speed can reveal the lower bound of the project completion time that can attained by additional budget.

The problem of scheduling directed acyclic task flow graphs has been examined in many forms in information technology for multiprocessor systems. Hary and Oezguener [7] studied the problem of scheduling directed acyclic task flow graphs to multiprocessor systems using point-to-point networks. Ahmad et al. [8] surveyed 21 algorithms that allocate a parallel program represented by an edge-weighted directed acyclic graph to a set of homogeneous processors, with the objective of minimizing the completion time.

Luh and Lin [9] claimed that it is possible to achieve minimum production time and increased productivity through the use of parallel operations in parts fabrication as well as in assembly, computation and control of industrial robots. However, the coupling between consecutive phases of the operations which results in series-parallel precedence constraints may, in turn, create unavoidable idle time intervals during the operations. Luh and Lin developed an algorithm that determines a minimum time-ordered schedule for the parallel operations. Their algorithm was based on the Program Evaluation and Review Technique (PERT).

For over three decades now researchers have sought effective solution procedures for shortening the critical paths in PERT types scheduling problems under conditions of limited resources availability [10]. Phillips [11] presented a procedure for these problems with multiple parallel processors that locates and verifies an optimal schedule for a project, under conditions of multiple resource constraints. The procedure is network-based and uses a graphical cut-search-approach. The optimal solution can be obtained iteratively by constructing a minimum cost network flow problem and adjusting the durations of activities corresponding to a minimum capacity cut-set. Baker [12] showed that the same can be done simply by using linear programming formulation.

A shorter project completion time can be attained by subdividing the work breakdown structure (WBS), and/or by redistributing the workload among activities, if possible and feasible. Laslo et al. [13] introduced a linear programming optimization that can be implemented for redistributing removable workload fragments among project activities for minimizing the project completion time. This was done without violating the precedence relations within the project.

One of the key features of project planning is the utilization of a WBS to show the hierarchy of tasks within a project and to define work packages [14]. The detailing of project tasks into sub-tasks can be done down to lower levels until reaching the level where the sub-tasks finally become manageable units (so-called work packages) for planning and control [15]. Detailing of project tasks and precedence relationships is theoretically an economic option for shortening the project completion time. The detailing of project tasks is unlimited, but one must take into consideration the manageability of the project, i.e., the DAG size that has to be determined by the number of arcs and by the number of nodes. Over-detailing increases the nodes and the arcs of the DAG up to $\binom{n}{2}$ arcs in a complete *n*-node DAG. However, [16] proved that random acyclic digraphs have only $n^2/4$ arcs on average.

The lower bound of the *n*-node DAG completion time is determined by taking into account the "ideal case" in which the optimal workload distribution among the given arc networks minimizes the project completion time. One important objective of this paper is to characterize the properties of this "ideal case". An additional objective is to show how to find for a pre-given total workload and a pre-given DAG this "ideal case", i.e. to compute a lower bound of the project completion time. To solve the problem of determining the optimal workload distribution, we use a special form of the max-flow problem, a special form of the maximum matching problem and Dilworth's Theorem [17].

The paper proceeds as follows: Section 2 defines the problem; Section 3 presents the linear programming solution for workload distribution; Section 4 presents a solution for the same problem by combinatorial optimization; Section 5 presents an algorithm for finding a max-cut of a DAG; Section 6 suggests a closed-form formula to compute the size of the max-cut of a complete DAG; and Section 7 concludes the paper. The main contributions of this paper for practitioners are that we propose a simple algorithm to find a max-cut in a DAG which is needed for lower bound computation and a closed-form formula to compute the size of the max-cut of a complete DAG.

2. Problem definition

Let us consider an activity-on-arc directed acyclic graph $G(N,A)$ where N is a set of n nodes, topologically sorted, and A is a set of m arcs denoted by $A_{i,j}$. Each of the m arcs has one unit of processing capacity, in which case we can consider the arc workload as the processing time of this arc. The graph $G(N,A)$ has one source node N_1 and one sink node N_n. The $A_{i,j}$'s arc workload $t_{i,j}$ is a non-negative variable. All the series paths in the graph, denoted by P^p and with length T^p, start at the source node N_1 and end at the sink node N_n. Let us also consider unlimited options for distributing the spread-out cumulative total workload T among the arcs $A_{i,j}$, i.e., the workload distribution is subject to $T = \sum_{A_{i,j} \in A} t_{i,j}$.

The task execution time of a DAG is equivalent to the length of the longest path, i.e., the critical path, $T^c = max(T^p)$. The most favorable planning for a given DAG is that in which the cumulative spread-out workload T is distributed such that it minimizes the critical path. The solution for the most favorable planning of workload distribution problem denoted, by $MP(G,T)$, is any feasible solution $\tau = \{t_{i,j}\}_{A_{i,j} \in A}$ that minimizes the objective function $T^c = max(T^p)$ subject to $T = \sum_{A_{i,j} \in A} t_{i,j}$. Thus, the solution of the $MP(G,T)$ problem is equivalent to finding a lower bound for the project completion time.

For the remainder of the paper we use the following notation:

A	-	a set of m arcs.
$A_{i,j}$	-	an arc (arrow), $A_{i,j} \in A$.
$A_{i,j}^D$	-	a dummy arc with $t_{i,j} = 0$, $A_{i,j}^D \in A$.
$A_{i,j}^E$	-	an effective arc with $t_{i,j} > 0$, $A_{i,j}^E \in A$.
$G(N,A)$	-	an activity-on-arc directed acyclic graph.
m	-	the number of arcs $A_{i,j} \in A$.
N	-	a set of n nodes, topologically sorted.
N_k	-	the k-th node, $N_k \in N$ $1 \le k \le n$.
n	-	the number of nodes $N_k \in N$.
P^p	-	the p-th path starting at the source node N_1 and ending at the sink node N_n.
q	-	the number of paths.
r	-	the minimal number of paths that covers the m network arcs $A_{i,j} \in A$.
s^τ	-	the number of effective arcs $A_{i,j}^E \in A$ in a feasible solution τ.
T	-	the cumulative total spread-out workload $T = \sum_{A_{i,j} \in A} t_{i,j}$.
T^c	-	the task execution time, which is the joint workload of the critical path, $T^c = max(T^p)$.
T^p	-	the joint-workload on P^p, $T^p = \sum_{A_{i,j} \in P^p} t_{i,j}$.
$t_{i,j}$	-	the $A_{i,j}$'s workload (variable), $t_{i,j} \ge 0$. Since each $A_{i,j}$ has one unit of processing capacity $t_{i,j}$ can be considered as the processing time.

τ - a feasible distribution, $\tau = \{ t_{i,j} \}_{A_{i,j} \in A}$.

$(X, \overline{X})$ - a cut-set of arcs induced by a set X of nodes that includes N_1 ,and its complement $\overline{X}$ that includes N_n; each arc $A_{i,j}$ in the cut-set has its tail in X and its head in $\overline{X}$.

π - a set of paths P^p that covers all the arcs $A_{i,j} \in A$.

3. The linear programming solution for the workload distribution problem

The $MP(G,T)$ problem can be set as a special case of a flow linear programming problem for any n-node graph. Let T_i be the time of completing all the activities ending at node i. Then the linear programming formulation for the problem is:

$Min\ T_n$

s.t.

$T_1 = 0$

$T_j - T_i \geq t_{i,j} \quad j = 2,...,n; \ i = j-1, j-2,...,1; \ \{t_{i,j}\}_{A_{i,j} \in A}$

$$\sum_{\{t_{i,j}\}_{A_{i,j} \in A}} t_{i,j} = T$$

$T_i \geq 0 \ \ i = 1,...,n$

Now let us consider an example of a DAG with $n = 5$, $m = 8$ and, $q = 5$ as presented in Figure 1.

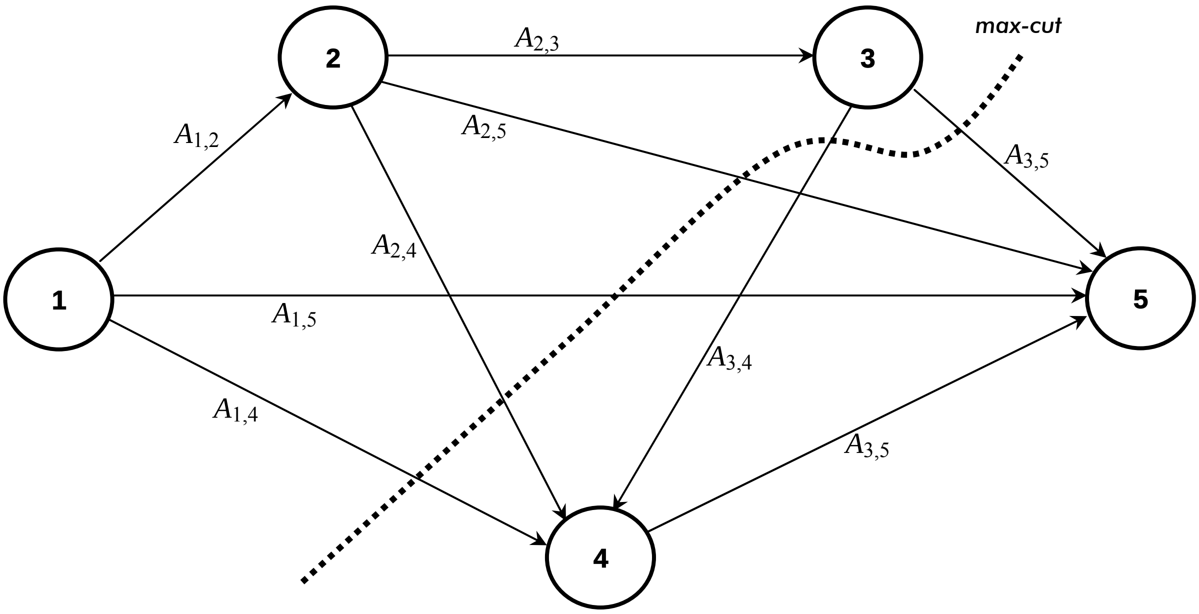

Figure 1. Directed acyclic graph

The linear programming formulation for this example is:

$Min\ \ T_5$

s.t.

$T_1 = 0$

$T_2 - T_1 - t_{1,2} \geq 0$

$T_4 - T_1 - t_{1,4} \geq 0$

$T_5 - T_1 - t_{1,5} \geq 0$

$T_3 - T_2 - t_{2,3} \geq 0$

$T_4 - T_2 - t_{2,4} \geq 0$

$T_5 - T_2 - t_{2,5} \geq 0$

$T_5 - T_3 - t_{3,5} \geq 0$

$T_5 - T_4 - t_{4,5} \geq 0$

$t_{1,2} + t_{1,4} + t_{1,5} + t_{2,3} + t_{2,4} + t_{2,5} + t_{3,5} + t_{4,5} = T$

$t_{i,j} \geq 0\ \ \forall\ \ A_{i,j} \in A.$

The two optimal basic solutions for this example are presented in Table 1.

Table 1. The optimal workload assignment for minimizing the execution time

Variable	Optimum I	Optimum II
$t_{1,2}$	0	0
$t_{1,4}$	0.2T	0.2T
$t_{1,5}$	0.2T	0.2T
$t_{2,3}$	0.2T	0
$t_{2,4}$	0.2T	0.2T
$t_{2,5}$	0.2T	0.2T
$t_{3,5}$	0	0.2T
$t_{4,5}$	0	0
T^c	0.2T	0.2T

Linear programming is an easy way to solve small $MP(G,T)$ problems, but it has disadvantages. The first disadvantage is that it needs many variables and many constraints that must be compiled and satisfied. The linear programming formulation for *n*-node DAG has up to $\binom{n}{2}+1$ constraints, $m=\binom{n}{2}$ viable arrows and $q=2^{n-2}$ paths. The second disadvantage is that with linear programming it is more difficult to explore solution properties. Therefore, in the following sections we present a procedure to solve $MP(G,T)$ problems by combinatorial optimization

4. A solution for the workload distribution problem by combinatorial optimization

Now we use a combinatorial optimization to derive a solution for the n-node graph workload distribution problem $MP(G,T)$. As the following theorems show, some interesting results can be obtained by combinatorial optimization.

Theorem 1 states that an optimal distribution of the cumulative spread-out workload T is obtained by uniform distribution of T over the arcs of some maximum directed cut which separates nodes N_1 and N_n. We define a directed cut as the set of arcs formed by a partition of the node-set N to two parts $X, \bar{X}$ so that $N_1 \in X$, $N_n \in \bar{X}$ and the set of arcs directed from $\bar{X}$ to X is empty. The arcs which belong to a cut are those which are directed from X to $\bar{X}$. Such a cut is denoted as $(X, \bar{X})$. The size of the cut is denoted by $|(X, \bar{X})|$.

Theorem 1. $min(T^c)$ with the cumulative spread-out workload T equals $T/max\{|(X, \bar{X})|\}$, where the maximum is over all the directed cuts $(X, \bar{X})$ that separate the nodes N_1 and N_n. (The proof of Theorem 1 can be found in the Appendix. This proof used Lemma 1 and Dilworth's Theorem which also in the Appendix.)

Theorem 2. An arc is an effective arc $A^E_{i,j}$ in some feasible solution τ if it belongs to some max directed cut which separates nodes N_1 and N_n. (For the proof see the Appendix.)

It follows from Theorem 2 that s^τ, the number of effective arcs $A^E_{i,j}$ in a given feasible solution τ, is at least the size of a max directed cut which separates nodes N_1 and N_n: $s^\tau \geq r$

By a reduction to Dilworth's Theorem, as done in the proof for Theorem 1, we get the following corollary:

Corollary 1. The problem $MP(G,T)$ is polynomially solvable by standard min-flow-max-cut algorithms that solve Dilworth's problem.

5. An algorithm for finding a max-cut of a DAG

The proposed algorithm for finding a max-cut of a DAG is based on the concept of maximum matching (see [18], Section 26.3). The basic idea is to present the DAG by a bipartite graph and then to apply an algorithm for finding a maximum matching.

Before describing and applying the algorithm, we need to recall some notions from matching theory. A matching M in a graph is a set of edges of which any two do not have a common end. Given a matching M in a graph, an alternating path with respect to M is a path whose edges alternate between matched edges and non-matched edges. An isolated vertex will also be considered as an alternating path although it is not an end of any edge. A vertex is called M-exposed if it is not an end of a matched edge.

The algorithm proceeds as follows:

Step 1: Construct a bipartite graph *H* associated with the DAG. The vertex set of *H* is then constructed as follows. For any arc $A_{i,j}$ we introduce two vertices: a start vertex $(A_{i,j})^S$ and a finish vertex $(A_{i,j})^F$. Let *S* denote the set of start vertices and *F* denote the set of finish vertices. *S* and *F* partition the vertex set of the bipartite graph *H* into its two parts. The edge set of *H* consists of pairs of vertices ($(A_{i,j})^S$, $(A_{k,l})^F$) where the two arcs $A_{i,j}$ and $A_{k,l}$ belong to some directed path in the DAG starting at $A_{i,j}$ and ending at $A_{k,l}$. An illustration of the bipartite graph *H* associated with the DAG in Figure 1 is given in Figure 2. The bold edges form a maximum matching in *H*.

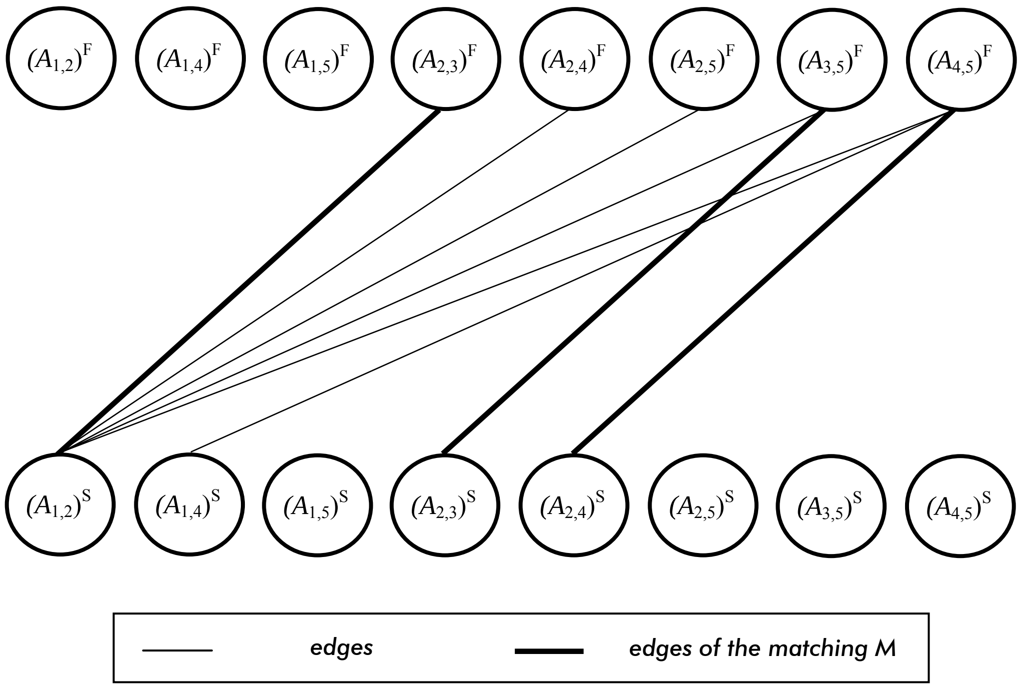

Figure 2. Description of the directed acyclic graph (Figure 1) by a bipartite graph

Step 2: Find a maximum matching *M* in *H*. Efficient algorithms for finding a maximum matching can be found in any combinatorial optimization book, e.g. [18].

Step 3: Scan the graph *H* in order to find all the vertices which belong to some alternating path starting at an *M*-exposed vertex of *S*. A vertex found by the scanning will be called a scanned vertex. This scanning is actually what is done before termination of any algorithm for finding a maximum matching, see [18].

Step 4: Look for the arcs $A_{i,j}$ of the DAG which were scanned exactly once at *H* (either the start vertex $(A_{i,j})^S$ was scanned, or the finish vertex $(A_{i,j})^F$ was scanned, but not both nor none of them) in Step 3. These arcs are the arcs of a max-cut in the DAG, see [19].

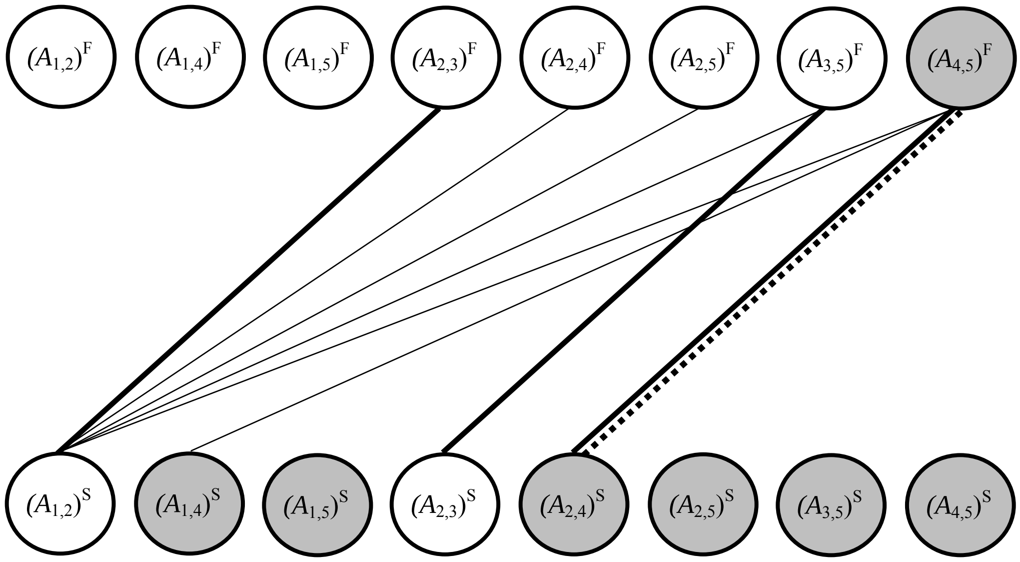

Figure 3. Alternating paths and scanned vertices

Figure 3 illustrates the alternating paths and the scanned vertices of the DAG that presented are in Figures 1 and 2. The alternating paths (see Figure 3) starting at *M*-exposed vertices of *S* are the isolated vertices of S, $(A_{1,5})^S, (A_{2,5})^S, (A_{3,5})^S, (A_{4,5})^S$ and the path $((A_{1,4})^S, (A_{4,5})^F, (A_{2,4})^S)$. Therefore, the scanned vertices are $(A_{1,5})^S, (A_{2,5})^S, (A_{3,5})^S, (A_{4,5})^S, (A_{1,4})^S, (A_{4,5})^F, (A_{2,4})^S$. The arcs of the DAG of which exactly one of their start vertexes and finish vertexes is scanned (Step 4) are $A_{1,4}$, $A_{1,5}$, $A_{2,5}$, $A_{3,5}$, $A_{2,4}$. These are the arcs of a max-cut (see Figure 1).

6. The size of the max-cut of a complete DAG

It is obvious that *n*-node DAG with the maximal number of viable arrows $\binom{n}{2}$ enables us to attain a better solution to the $MP(G,T)$ problem, or at least one that is no worse than the solution for an *n*-node DAG with $m < \binom{n}{2}$ arcs. This is because each additional arc $A_{i,j}$ can be loaded or remain unloaded. From Theorem 1 we get an additional corollary that enables us to easily calculate the size of a max-cut of a complete DAG, and from this to compute a lower bound for T^c.

Corollary 2. If $G(N, A)$ is the complete graph with n nodes and $\binom{n}{2}$ arcs, then the size of the max-cut is: $max\{|(X, \bar{X})|\} = \left\lfloor n^2/4 \right\rfloor$,

and as consequence, the T^c of an optimal distribution τ of T for $MP(G,T)$ is:

$min(T^c) = \frac{T}{\left\lfloor n^2 / 4 \right\rfloor}$. (For the proof see the Appendix.)

7. Summary and conclusions

The aim of this paper was to evaluate the possible benefits from improving the workload distribution in a DAG (e.g. a project network), by determining a lower bound of the project completion time. We proved that a lower bound can be obtained by dividing the total cumulative processing time uniformly among the arcs on a max-cut set. That is the maximum parallel planning for a given DAG. We also presented a combinatorial algorithm for finding the max-cut, based on a maximum matching algorithm. The simple closed-form solution for the complete *n*-node DAG case $\left\lfloor n^2/4 \right\rfloor$ can be a very useful formula to calculate a lower bound of the completion time of any DAG.

The value of the term $\left(T / \left\lfloor n^2/4 \right\rfloor - T / \left|(X, \bar{X})\right|\right)$ can be an indication of the benefit derived from further detailing of the project tasks and from increasing edges up to a complete graph. This is because the difference between $\left\lfloor n^2/4 \right\rfloor$, the max-cut size of a complete DAG, and $\left|(X, \bar{X})\right|$, the max-cut size of any DAG, represents further potential for shortening the project completion time that can attained by more detailing of project tasks.

The results in this paper can help planners and project managers to characterize the "ideal case" in which the optimal workload distribution among the arc networks minimizes the project completion time. That can be done for a non-complete DAG as well as for a complete one. A lower bound can then be used to evaluate the maximum potential for reducing the project completion time in real-life cases.

References

1. Ahmad, I., Kwok, Y.-K. and Wu, M.-Y. **Analysis, evaluation and comparison of algorithms for scheduling task graphs on parallel processors,** "Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks", Beijing, China, 1996, pp.207-213
2. Baker, B.M. **Cost time trade-off analysis for the critical path method: a derivation of the network flow approach,** Journal of the Operations Research Society, vol. 48, 1997, pp. 1241-1244
3. Brightwell, G. **Partial orders, in graph connections,** (Beineke, L.W. and Wilson, R.J. Eds.) Oxford University Press, 1997, pp.52–69
4. Cook, W.J., Cunningham, W.H., Pulleyblank, W.R. and Schrijver, A. **Combinatorial Optimization,** John Wiley & Sons, INC., New York, 1998, pp. 53-62
5. Cormen, T.H., Leiserson, C.E., Rivest, R.L.and Stein, C. **Introduction to Algorithms,** Second Edition, The MIT Press and McGraw-Hill, 2001
6. Golenko-Ginzburg, D. and Gonik, A. **A heuristic for network project scheduling with random activity durations depending on the resource reallocation,** International Journal of Production Economics, vol. 55(2), 1998, pp. 149-162

7. Golenko-Ginzburg, D., Gonik, A. and Kesler, S. **Hierarchical decision-making model for planning and controlling stochastic projects,** International Journal of Production Economics, vol. 46/47, 1996, pp. 39-54
8. Golenko-Ginzburg, D., Gonik, A. and Sitniakovski, S. **Resource supportability model for stochastic network projects under a chance constraint,** Communications in Dependability and Quality Management, vol. 3(1), 2000, pp. 89-102
9. Hary, S.L. and Oezguener, F. **Precedence-constrained task allocation onto point-to-point networks for pipelined execution,** IEEE Transactions on Parallel and Distributed Systems, vol. 10(8), 1999, pp. 838-851
10. Kelley, J.E. and Walker, M.R. **Critical path planning and scheduling,** in "Proc. of the Eastern Joint Computer Conference", 1959, pp. 160-173
11. Kelley, J.E. **Critical path planning and scheduling: mathematical basis,** Operations Research, vol. 9(3), 1961, pp. 296-320
12. Laslo, Z. **Activity time-cost tradeoffs under time and cost chance constraints,** Computers & Industrial Engineering, vol. 44(3), 2003, pp. 365-384
13. Laslo, Z., Keren, B. and Ilani, H. **Minimizing task completion time with the execution set method,** European Journal of Operational Research, 187, 2008, pp. 1513–1519
14. Lavold, G. **Developing and using the work breakdown structure,** in Clealand, D. and King, W. (Eds.), "Project Management Handbook", Van-Nostrand Reinhold Company Inc., New York, 1988
15. Luh, J.Y.C. and Lin, C.S. **Scheduling parallel operations in automation for minimum execution time based on PERT,** Computers & Industrial Engineering, vol. 9(2), 1985, pp.149-165
16. Melançon, G., Dutour, I.and Bousquet-Mélou, M. **Random generation of directed acyclic graphs,** in Nesetril, J., Noy, M. and Serra, O. (Eds.), Comb01, "Euroconference on Combinatorics, Graph Theory and Applications, in: Electronic Notes in Discrete Mathematics", vol. 10, 2001
17. Phillips, S. Jr. **Project management duration/resource tradeoff analysis: an application of the cut search approach,** Journal of the Operations Research Society, vol. 47, 1996, pp. 697-701
18. Simpson, W.P. III and Patterson, J.H. **A Multiple-tree search procedure for the resource-constrained project scheduling problem,** European Journal of Operational Research, vol. 89(3), 1996, pp. 525-542
19. * * * **DOD and NASA Guide, PERT Cost Systems Design,** by the Office of the Secretary of Defense and National Aeronautics and Space Administration, U.S. Government Printing Office, Washington, D.C., June 1962

Appendix

Lemma 1. If $\pi=\{P^1,...,P^r\}$, $1\le r\le q$ is a cover of the arcs $A_{i,j}$ by paths P^p, $1\le p\le r$ and τ is a feasible solution, then

$$T^c \ge \frac{T}{r}. \qquad (1)$$

Moreover, if equality holds, then all paths in π are of joint-workload $T^p = T/r$ and each arc $A_{i,j}$ which is covered more than once by π has workload $t_{i,j}=0$, i.e. it is a dummy or unnecessary arc $A_{i,j}^D$.

Proof of Lemma 1. Let π be a cover of the arcs $A_{i,j}$ by paths P^p, $1 \le p \le r$, and let τ be a feasible solution. Since τ is a feasible solution, it satisfies the constraint $T = \sum_{A_{i,j} \in A} t_{i,j}$.

π is a cover of the arcs and hence,

$$\sum_{p=1}^{r} T^p = \sum_{p=1}^{r} \sum_{A_{i,j} \in P^p} t_{i,j} \ge \sum_{A_{i,j} \in A} t_{i,j} = T, \qquad (2)$$

It follows that there must be a path P^p in π satisfying, $\sum_{A_{i,j} \in P^p} t_{i,j} \ge T/r$ and therefore $T^c \ge T/r$. From (2) it follows that equality holds if all paths in π have the same workload T^p and each arc $A_{i,j}$, which is covered more than once by π, has workload $t_{i,j} = 0$, i.e. each such arc should be classified as $A_{i,j}^D$. □

The following theorem is a version of Dilworth's Theorem [4].

Dilworth's Theorem. Let $G(N, A)$ be an acyclic network where the nodes $N_1, \dots\dots, N_n$ are topologically sorted and let d be a non-negative integral function on the arc set A. Then, the minimum number of P^ps which cover each arc $A_{i,j}$ at least $d(A_{i,j})$ times equals $max \sum_{A_{i,j} \in (X, \bar{X})} d(A_{i,j})$ where the maximum is over the cuts $(X, \bar{X})$ that separate the nodes N_1 and N_n, i.e. $max \left\{ \sum_{A_{i,i} \in (X, \bar{X})} d(A_{i,j}) \mid 1 \in X, n \in \bar{X} \right\}$.

Proof of Theorem 1. Let $(X, \bar{X})$ be a cut separating nodes N_1 and N_n of cardinality r. Also, let $\tau_{(X, \bar{X})}$ be the distribution of the cumulative spread-out workload T obtained by distributing T uniformly on the edges of the cut $(X, \bar{X})$. Since every P^p contains a single arc of the cut $(X, \bar{X})$ the following inequality is valid:

$$min\ T^c(\tau) \le T^c \tau_{(X, \bar{X})} = \frac{T}{r}. \qquad (3)$$

for any feasible solution τ.

Since the above is true for any cut, it is also true for a cut with maximum cardinality and hence,

$$min\ T^c(\tau) \le \frac{T}{max\{|(X, \bar{X})|\}}. \qquad (4)$$

On the other hand, let $(X, \bar{X})$ be a cut that separates nodes N_1 and N_n of maximum cardinality and τ be a distribution of the cumulative spread-out workload T that minimizes the left-hand side of (4). By taking $d \equiv 1$ in Dilworth's Theorem we get that the cardinality of the cut $(X,\bar{X})$ is equal to some covering π of the arc set A by paths P^p. It follows from Lemma 1 that $T^c(\tau) \geq T/max\{|(X,\bar{X})|\}$. This completes the theorem's proof. □

Proof of Theorem 2. Let $A_{i,j}$ be an arbitrary arc. If $A_{i,j}$ belongs to some max-cut which separates nodes N_1 and N_n, then the optimum solution $\tau_{(X,\bar{X})}$ assigns the workload $t_{i,j} = T/(X,\bar{X})$ to $A_{i,j}$ and therefore it is an effective arc $A^E_{i,j}$. Conversely, suppose $A_{i,j}$ doesn't belong to any max-cut which separates nodes N_1 and N_n. We again apply Dilworth's Theorem with *d* equaling 1 everywhere except at the arc $A_{i,j}$ where it equals 2. Since $A_{i,j}$ doesn't belong to any max-cut which separates nodes N_1 and N_n, it follows that any max-cut which separates nodes N_1 and N_n maximizes the objective function $\sum_{A_{i,j} \in (X,\bar{X})} d(A_{i,j})$. Therefore, by the theorem there is a cover π of the arcs by paths P^p that covers the arc $A_{i,j}$ twice and has the cardinality of a max-cut. Thus by the last claim of Lemma 1, in any distribution τ of T for which inequality (1) holds with equality the arc $A_{i,j}$ is assigned $t_{i,j} = 0$. But for cover π, which has the cardinality of a max-cut, a distribution τ of T fulfills equality in (1) if it is an optimum solution. Therefore, $A_{i,j}$ is a dummy or unnecessary arc $A^D_{i,j}$. □

Proof of Corollary 2. By Theorem 1, the $T^c(\tau)$ for an optimal distribution τ of T equals T divided by the size of a max-cut which separates nodes N_1 and N_n. When the given network is a complete network, the cut that separates nodes N_1 and N_n is the set of arcs connecting the first r nodes $N_1,\ldots\ldots,N_r$ with the last $n-r$ nodes $N_{r+1},\ldots\ldots,N_n$ for some r $(1 \leq r \leq n-1)$. By straightforward calculus, the max-cut separates the first $n/2$ nodes ($\lfloor n/2 \rfloor$ or $\lceil n/2 \rceil$ when n is odd) from the last $n/2$ nodes ($\lceil n/2 \rceil$or $\lfloor n/2 \rfloor$ when n is odd) and is of size $\lfloor n^2/4 \rfloor$. □

[1]Zohar Laslo is Associate Professor and Dean of the Industrial Engineering and Management, SCE - Shamoon College of Engineering, Israel. Zohar was born in 1945. He received his B.Sc. and M.Sc. in Industrial Engineering and Management from the Technion - Israel Institute of Technology, Haifa, and his Ph.D. degree in Industrial Engineering and Management (Summa Cum Laude) from the Ben Gurion University of the Negev, Israel. His professional experience includes 24 years of engineering and management positions in Israeli aeronautical, electro-optical, pharmaceutical industries. His current research interests are in matrix structure performance, network design, planning scheduling and control, planning under time-cost uncertainty, quality of planning, maintainability and hazard analysis techniques, and technological education. His recent publications have appeared in International Journal of Project Management, International Journal of Production Economics, Journal of Applied Quantitative Methods, European Journal of Operational Research, Computer Modelling and New Technologies, Journal of Business Economics and Management, Technological and Economic Development of Economy, Transport and Telecommunication, Communications in Dependability and Quality Management, Computers & Industrial

Engineering, Mathematics and Computers in Simulation, Automation and Remote Control, OMO the Journal of Terotechnology, IEEE Transactions on Engineering Management, Tribology in industry and European Journal of Engineering Education.

[2] Hagai Ilani is Senior Lecturer and Head Head of the Industrial Engineering and Industrial Management department in Ashdod Campus, SCE - Shamoon College of Engineering, Israel. Hagai was born in 1966. He received his B.Sc., M.Sc, and Ph.D. in Mathematics from the Ben Gurion University of the Negev. His current research interests are in combinatorial optimization and meta-heuristics algorithms for NP-hard optimization problems. His recent publications have appeared in European Journal of Operational Research and Journal of Combinatorial Theory.

[3] Corresponding author:
Tel: +972-8-647-5641/42/48
Fax: +972-8-647-5643
E-Mail: baruchke@sce.ac.il

[4] Baruch Keren is a Senior Lecturer and Head of the Industrial Systems Track in the Industrial Engineering and Management Department, SCE - Shamoon College of Engineering, Israel, and lecturer at the Open University of Israel. Baruch was born in 1958. He received his B.Sc., M.Sc. and Ph.D. (Summa Cum Laude) in industrial engineering from Ben-Gurion University. His professional experience includes 13 years of several professional positions in the Israel Chemicals Ltd. and its subsidiaries in the areas of industrial engineering, economics and auditing.His current research interests are in decision-making under uncertainty, production planning, project management and operations research. His recent publications have appeared in International Journal of Production Economics, Omega, European Journal of Operational Research and IEEE Transactions on Reliability.

[5] Codification of references:

[1]	Kelley, J.E. and Walker, M.R. **Critical path planning and scheduling,** in "Proc. of the Eastern Joint Computer Conference", 1959, pp. 160-173
[2]	Kelley, J.E. **Critical path planning and scheduling: mathematical basis,** Operations Research, vol. 9(3), 1961, pp. 296-320
[3]	Laslo, Z. **Activity time-cost tradeoffs under time and cost chance constraints,** Computers & Industrial Engineering, vol. 44(3), 2003, pp. 365-384
[4]	Golenko-Ginzburg, D., Gonik, A. and Kesler, S. **Hierarchical decision-making model for planning and controlling stochastic projects,** International Journal of Production Economics, vol. 46/47, 1996, pp. 39-54
[5]	Golenko-Ginzburg, D. and Gonik, A. **A heuristic for network project scheduling with random activity durations depending on the resource reallocation,** International Journal of Production Economics, vol. 55(2), 1998, pp. 149-162
[6]	Golenko-Ginzburg, D., Gonik, A. and Sitniakovski, S. **Resource supportability model for stochastic network projects under a chance constraint,** Communications in Dependability and Quality Management, vol. 3(1), 2000, pp. 89-102
[7]	Hary, S.L. and Oezguener, F. **Precedence-constrained task allocation onto point-to-point networks for pipelined execution,** IEEE Transactions on Parallel and Distributed Systems, vol. 10(8), 1999, pp. 838-851
[8]	Ahmad, I., Kwok, Y.-K. and Wu, M.-Y. **Analysis, evaluation and comparison of algorithms for scheduling task graphs on parallel processors,** "Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks", Beijing, China, 1996, pp.207-213
[9]	Luh, J.Y.C. and Lin, C.S. **Scheduling parallel operations in automation for minimum execution time based on PERT,** Computers & Industrial Engineering, vol. 9(2), 1985, pp.149-165
[10]	Simpson, W.P. III and Patterson, J.H. **A Multiple-tree search procedure for the resource-constrained project scheduling problem,** European Journal of Operational Research, vol. 89(3), 1996, pp. 525-542
[11]	Phillips, S. Jr. **Project management duration/resource tradeoff analysis: an application of the cut search approach,** Journal of the Operations Research Society, vol. 47, 1996, pp. 697-701
[12]	Baker, B.M. **Cost time trade-off analysis for the critical path method: a derivation of the network flow approach,** Journal of the Operations Research Society, vol. 48, 1997, pp. 1241-1244
[13]	Laslo, Z., Keren, B. and Ilani, H. **Minimizing task completion time with the execution set method,** European Journal of Operational Research, 187, 2008, pp. 1513–1519
[14]	* * * **DOD and NASA Guide, PERT Cost Systems Design,** by the Office of the Secretary of Defense and National Aeronautics and Space Administration, U.S. Government Printing Office, Washington, D.C., June 1962

[15]	Lavold, G. **Developing and using the work breakdown structure,** in Clealand, D. and King, W. (Eds.), "Project Management Handbook", Van-Nostrand Reinhold Company Inc., New York, 1988
[16]	Melançon, G., Dutour, I.and Bousquet-Mélou, M. **Random generation of directed acyclic graphs,** in Nesetril, J., Noy, M. and Serra, O. (Eds.), Comb01, "Euroconference on Combinatorics, Graph Theory and Applications, in: Electronic Notes in Discrete Mathematics", vol. 10, 2001
[17]	Cook, W.J., Cunningham, W.H., Pulleyblank, W.R. and Schrijver, A. **Combinatorial Optimization,** John Wiley & Sons, INC., New York, 1998, pp. 53-62
[18]	Cormen, T.H., Leiserson, C.E., Rivest, R.L.and Stein, C. **Introduction to Algorithms,** Second Edition, The MIT Press and McGraw-Hill, 2001
[19]	Brightwell, G. **Partial orders, in graph connections,** (Beineke, L.W. and Wilson, R.J. Eds.) Oxford University Press, 1997, pp.52–69