AND/OR graphs and project management

Dušan Hvalica
Faculty of Economics, Ljubljana, Slovenia

It is shown that projects can be described by AND/OR graphs and that the cost of the minimal solution subgraph can be defined so as to equal the shortest time for completing a project or the minimal costs of the project represented by the AND/OR graph.

Keywords: project management, network programming, AND/OR graphs

1. Introduction

AND/OR graphs were introduced to model the reduction of problems to their subproblems. Thus, solving a problem by reduction is in the context of AND/OR graphs equivalent to finding a solution subgraph in the corresponding AND/OR graph [5,7]. Moreover, AND/OR graphs provide a natural environment for automated deduction and theorem proving [3]. In all cases the central concept is that of a solution subgraph or — according to the paradigm *small is quick* [6] — a minimal solution subgraph. As we shall show, AND/OR graphs can be useful in planning, scheduling and control of projects as well — the cost of a solution subgraph can be defined so as to correspond to the duration of a project or to the costs of the project.

2. AND/OR graphs

For any graph G the set of its tip nodes (the nodes without child nodes) will be denoted by N_G. It will be assumed that a distinguished subset S of N_G is given; the nodes in S will be called *solved*. For any node x, $\Gamma(x)$ will denote the set of child nodes of x and for any arc (x, y) its cost will be denoted by $c(x, y)$. It will be assumed that $c(x, y) \in \mathbf{R}^*$, where $\mathbf{R}^* = R \cup \{\infty\}$, with the usual extensions of summation and ordering: $x + \infty = \infty + x = \infty$, $\forall x \in \mathbf{R}^*$, and $\forall x \in \mathbf{R}$.

An *AND/OR graph* is a directed graph G for which a partition of the set of its nodes $G = O \cup A \cup N_G$ is given; the nodes in O and A are called *OR-nodes* and *AND-nodes*, respectively. The distinction between these nodes results from the way they are *established* — an OR-node is established when any of its child nodes is established, an AND-node is established when all of its child nodes are established, while a tip-node is established if and only if it is solved.

> In problem reduction, the situation B can be solved by solving B_1 or by solving both B_2 and B_3

and can be represented by the AND/OR graph in Figure 1; a node is established when the problem corresponding to that node is solved. Nodes

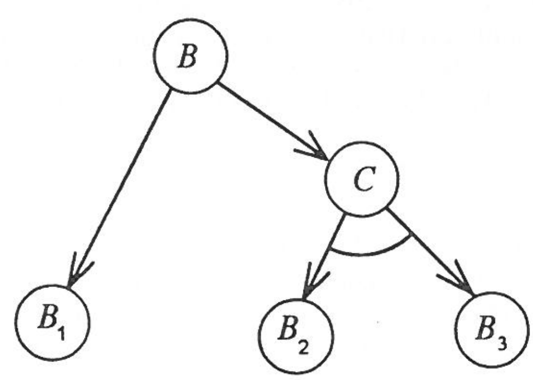

Fig. 1.

in N_G represent problems which either cannot be reduced to subproblems or can be solved without reduction (hence the term *solved* nodes). Node C is an AND-node; it is established when both B_2 and B_3 are. Node B — an OR-node — can be established by establishing any of B_1, C.

Similarly, the following formula can be represented by the AND/OR graph in figure 1:

$$B_1 \vee (B_2 \wedge B_3) \implies B$$

Here a node is established when the corresponding formula is true.

3. AND/OR graphs in project management

A project is traditionally described as a set of activities, equipped with the precedence relation

$$A_1 P A_2 \iff A_1 \text{ must finish before } A_2 \text{ can start} \tag{1}$$

It is tacitly understood that every activity starts as soon as possible, which implies that A_2 starts as soon as every A, such that APA_2, is finished.

Especially in the planning phase, more flexibility can be gained by introducing yet another binary relation

$$A_1 E A_2 \iff \text{the finish of } A_1 \text{ enables the start of } A_2 \tag{2}$$

thus allowing alternative ways of accomplishing a goal. For instance, if the conditions for the start of activity A_2 can be attained by performing any of A_1, A_3 then we have A_1EA_2 and A_3EA_2.

It will be tacitly understood that when introducing alternatives one is thorough, i.e., when A_2 starts, at least one of A, for which AEA_2, is finished. Hence the following will apply:

$$\forall(AEA_2 \implies BPA) \implies BPA_2 \tag{3}$$

Suppose now that for some A_2 we have A_1PA_2 and $A_1'EA_2$. Then, when A_1' finishes, A_2 starts so that A_1 must have finished by then. Thus the finish of A_1 always precedes the finish of A_1'. It follows that either A_1PA_1' or A_1' splits, $A_1' = \{B_1, B_2\}$, such that A_1PB_2 and B_2EA_2 (B_2 is simply a part of A_1' after the finish of A_1). In the latter case both P and E in a natural way extend to B_1 and B_2:

$$CPB_1 \iff CPA_1',\ CEB_1 \iff CEA_1',\ B_2PC \iff A_1'PC,\ B_2EC \iff A_1'EC,\ B_1EB_2.$$

Since, by (3), A_1PA_2 is implied by A_1PB_2 and B_2EA_2 (or A_1PA_1' and $A_1'EA_2$) corresponding to all A_1' for which $A_1'EA_2$, it follows that A_1PA_2 can be omitted. Clearly, by repeating this step (if necessary) a new relation $P' \subset P$ can be obtained for which

$$\mathcal{R}_{P'} \cap \mathcal{R}_E = \emptyset,$$

i.e., no activity is in the ranges of both P' and E, but which, by (3), nevertheless implies complete original information about precedence.

Clearly, such a project can be represented by an AND/OR graph — activities in the ranges of P' and E are represented by AND-nodes and OR-nodes, respectively; a node is established when the corresponding activity is finished. Actually, AND-nodes can be viewed as representing virtual activities (with zero duration time), corresponding to the completion of all of the activities, corresponding to their child nodes. For instance, the situation

$$B \text{ is triggered by the finish of } B_1 \text{ or by the finish of both } B_2 \text{ and } B_3 \tag{4}$$

can be represented by the AND/OR graph in Figure 1.

A weighted AND/OR graph is then obtained by setting

$$c(x, y) = d(y)$$

where $d(y)$ denotes the duration of the activity represented by y.

(This is the so called activities-on-nodes approach; obviously, the activities-on-arcs approach is also possible — the corresponding AND/OR graph for the situation (4) is in Figure 2.)

We shall assume (clearly without loss of generality) that in our AND/OR graphs there are two distinguished nodes, s and c — representing the start and the completion of the project — and that through every node t there is a path from c to s. Furthermore, we shall assume that $S = \{s\}$.

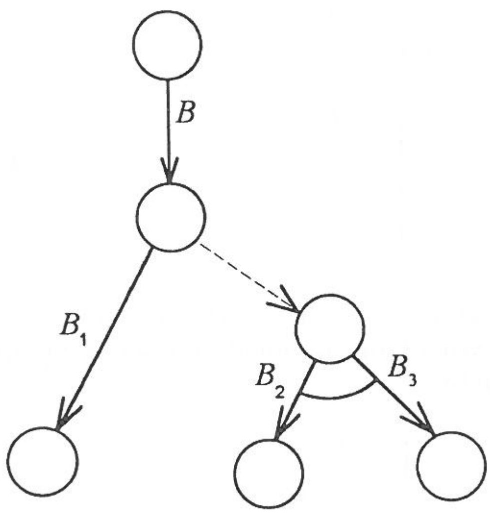

Fig. 2.

While by the traditional approach the corresponding network is necessarily acyclic [4], allowing alternative ways to accomplish certain goals does not rule out cycles any more. Thus, it is to be expected that, in general, these AND/OR graphs will not be acyclic.

4. Solution subgraphs and their cost

A *solution subgraph* of $x \in G$ is such a subgraph $G' \subset G$ that

- $x \in G'$ and G' contains no arcs ending in x,
- for every OR-node $t \in G'$ it contains exactly one arc leaving t,
- for every AND-node $t \in G'$ it contains all arcs leaving t,
- every tip node $t \in G'$ is solved,
- it contains no cycles.

Clearly, every node of a solution subgraph is established; to establish x it suffices to find any of its solution subgraphs.

In an AND/OR graph representing some project node c is established if and only if all activities, corresponding to the nodes of some solution subgraph of c are finished. Thus solution subgraphs of c are in one-to-one correspondence with feasible ways of completing the project.

Let B be a solution subgraph for x; its *cost* is defined as $w_B(x)$, where the function $w_B : B \to \mathbf{R}$ is given recursively by

$$w_B(u) = \begin{cases} h(u), & u \in N_G \\ w_B(v) + c(u,v), & u \in O \\ \max\limits_{v \in \Gamma(u)} \{w_B(v) + c(u,v)\}, & u \in A \end{cases}$$

where

$$h(u) = \begin{cases} 0, & u \in S \\ \infty, & u \in N_G \setminus S. \end{cases}$$

By induction one easily verifies that $w_B(x)$ is equal to the cost of the most expensive path in B from x to a tip-node of B (where the cost of a path is equal to the sum of the costs of its arcs).

Clearly, in any cycle free AND/OR graph the cost of the *minimal solution subgraph* is given by the function $w : G \to \mathbf{R}^*$, satisfying

$$w_B(u) = \begin{cases} h(u), & u \in N_G \\ \min_{v \in \Gamma(u)} \{w(v) + c(u,v)\}, & u \in O \\ \max\limits_{v \in \Gamma(u)} \{w_(v) + c(u,v)\}, & u \in A. \end{cases} \tag{5}$$

This function can be computed recursively and can be applied to direct the search for the minimal solution subgraph [2,5,6,7]. Of course, if there are cycles, w can no longer be computed by (pure) recursion, but algorithms for its computation are known [1]. With w it is then easy to find the minimal solution subgraph M : for any $x \in M$ — unless $x \in N_G$ — it contains

- all child nodes of x if x is an AND-node,
- the node $z \in \Gamma(x)$ for which

$$w(z) + c(x,z) = \min_{y \in \Gamma(x)} \{w(y) + c(x,y)\}$$

if x is an OR-node.

When a project is represented by an AND/OR graph, then clearly two activities, lying on the same path, cannot be performed simultaneously, moreover, the one lying further down the path must be executed before the other. Thus the time, necessary to complete all activities on some path, cannot be shorter than the cost of that path. It follows that the time for the completion of a project in some feasible way is equal to the cost of the most expensive path in the corresponding solution subgraph of c, i.e. to the cost of that solution subgraph. Therefore the following applies:

The shortest time for the completion of a project is equal to the cost of the minimal solution subgraph of c.

As already told, algorithms for finding the cost of the minimal solution subgraph and the minimal solution subgraph itself are available. Knowing minimal solution subgraph M it is then easy to determine the slack of all activities that constitute it — for any $t \in M$ a delay δ in the execution of the activity corresponding to t affects the activities lying on any path from c to t; if $T \subset M$ is such a path and c_T its cost, then, if there is to be no delay for c, we must have

$$w(t) + \delta + c_T \leq w(c)$$

and therefore

$$w(t) + \delta + \max_{T \in \mathcal{P}} c_T \leq w(c),$$

where $\mathcal{P}$ is the set of all paths in M from c to t. Thus the slack equals

$$s(t) = w(c) - \max_{T \in \mathcal{P}} c_T - w(t).$$

Since there is no difference between an AND-node and an OR-node with only one child node (both are established exactly when their child node is), all nodes in M can be considered AND-nodes. Then clearly

$$w_{M^*}(t) = \max_{T \in \mathcal{P}} c_T,$$

where M^* is the converse graph of M, i.e.,

$$(x, y) \in M \iff (y, x) \in M^*,$$

with $S = \{c\}$, so that we finally have .

$$s(t) = w(c) - w_{M^*}(t) - w(t), \quad \forall t \in M.$$

Sometimes, the costs of the project are more important than the time dimension; if there are more feasible ways of completing the project, one naturally tries to minimize the costs. In this case too, AND/OR graphs provide appropriate environment — if the cost of a solution subgraph is defined as the sum of the costs of its arcs, then, as one easily verifies, the cost of a solution subgraph is equal to the costs of the project, if realized in the way corresponding to that solution subgraph.

If the solution subgraph is a tree, then clearly its cost equals $w_B(x)$, where

$$w_B(u) = \begin{cases} h(u), & u \in N_G \\ w_B(v) + c(u, v), & u \in O \\ \sum\limits_{v \in \Gamma(u)} (w_B(v) + c(u, v)), & u \in A. \end{cases}$$

Consequently, if the AND/OR graph is a tree, the cost of the minimal solution subgraph is given by

$$w_B(u) = \begin{cases} h(u), & u \in N_G \\ \min\limits_{v \in \Gamma(u)} \{w_B(v) + c(u, v)\}, & u \in O \\ \sum\limits_{v \in \Gamma(u)} (w_B(v) + c(u, v)), & u \in A. \end{cases} \tag{6}$$

Algorithms for the computation of the function, satisfying (5), can be easily modified so as to compute the function satisfying (6) in any AND/OR graph. However, if the AND/OR graph is not a tree, $w(x)$ may be different from the cost of the minimal solution subgraph of x — cf. Fig. 3.

Of course, since the function, giving the cost of the minimal solution subgraphs, cannot be described recursively, its computation is substantially more demanding than that of the function satisfying (6).

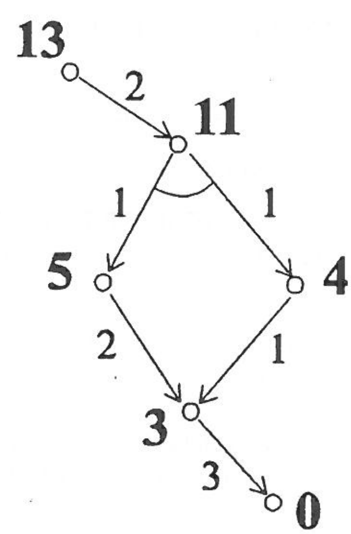

Fig. 3.

5. Conclusion

We have shown that instead of understanding a project as a fixed set of activities which must all be finished to complete the project, one can admit alternatives to some activities, which extends the applicability of the model from the realization phase to the planning phase of the project. As we have seen, AND/OR graphs provide a natural environment for modelling such projects — solution subgraphs correspond to feasible ways of completing the project; the cost of a solution subgraph can be defined so as to be equal to the duration or to the costs of the corresponding realization of the project. Algorithms for the former are available while effective algorithms for the latter are still to be developed.

References

[1] D. HVALICA, On the cost of potential solution subgraphs. In Proceedings, Symposium on Operations Research '95 (V. Rupnik, M. Bogataj, eds.), (1995) pp. 81–88. Slovenian Society Informatika, Portorož.

[2] D. HVALICA, Best First Search Algorithm in AND/OR Graphs with Cycles. J. of Algorithms, 21 (1996), 102–110.

[3] D. W. LOVELAND, Automated Theorem Proving: A Logical Basis. North-Holland Publishing Company, Amsterdam, New York, Oxford, 1978.

[4] J. J. MODER, C. R. PHILLIPS, E. W. DAVIS, Project Management with CPM, PERT and Precedence Diagramming. Van Nostrand Reinhold Company, New York, 1983.

[5] N. J. NILSSON, Problem-solving Methods in Artificial Intelligence. McGraw–Hill, New York, 1971.

[6] J. PEARL, Heuristics: Intelligent Search Strategies for Computer Problem Solving. Addison–Wesley, Reading, Massachusetts, 1985.

[7] E. V. POPOV, G. R. FIRDMAN, Algoritmicheskie osnovi intellektual'nih robotov i iskusstvenogo intellekta. Nauka, Moskva, 1976.

Received: January, 1997
Accepted: October, 1997

Contact address:

Dušan Hvalica
Faculty of Economics
Kardeljeva ploščad 17
1000 Ljubljana
Slovenia
E-mail: dusan.hvalica@uni-lj.si

DUŠAN HVALICA was born in 1950. He studied mathematics at the University of Ljubljana, where he graduated in 1972 and received his Ph.D. degree in 1985. He is an associate professor at the Faculty of Economics in Ljubljana.