On PERT Networks with Alternatives

Fabrice Chauvet, Eugene Levner, Jean-Marie Proth

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

On PERT Networks with Alternatives

Fabrice Chauvet - Eugène Levner
Jean-Marie Proth

N° 3583
Décembre 1998

THÈME 4

A propos des Réseaux PERT avec Alternatives

Fabrice CHAUVET*, Eugene LEVNER** et Jean-Marie PROTH* et ***

RESUME

La gestion de projet demande souvent que soient choisies les activités à exécuter dans un ensemble d'activités équivalentes. Ensuite, la durée requise pour exécuter le projet (i.e. le "makespan") est calculée. Dans cet article, nous cherchons à sélectionner les activités et à calculer le makespan simultanément. Nous appelons ce problème Problème PERT avec Alternatives (PPA). Le modèle correspondant à ce problème est similaire au graphe PERT conventionnel, excepté que deux types de nœuds sont utilisés pour représenter soit le choix entre les activités, soit le fait qu'un ensemble d'activités doivent être terminées avant qu'un autre ensemble d'activités puisse commencer. Une formalisation du problème et d'importantes propriétés concernant la solution optimale sont données. Plusieurs méthodes de résolution sont proposées et un algorithme polynomial basée sur une décomposition du problème est présentée. Ce dernier algorithme est applicable dans de nombreux cas réels.

MOTS-CLEFS

Modèle PERT, plus court chemin, activités alternatives, makespan.

* INRIA Lorraine, Technopôle Metz 2000, 4 rue Marconi, 57070 Metz, France
** Center for Technological Education Holon, 52 Golomb St., 58102 Holon, Israël
*** Institute for Systems Research, University of Maryland, College Park, MD 20742, USA

Ce travail a été financé par le programme franco-israélien "Arc-en-Ciel/Keshet" n° 64, et par les projets INTAS n° 96-812 et n° 96-820.

On PERT Networks with Alternatives

Fabrice CHAUVET*, Eugene LEVNER** and Jean-Marie PROTH*, ***

ABSTRACT

Management of projects often requires decisions concerning the choice of alternative activities. Then, the completion time of the whole project (i.e. the makespan) is computed. In this paper, we aim at selecting the required activities simultaneously with the computation of the makespan. This problem is referred to as PERT-Time Problem with Alternatives (PPA). The corresponding model is similar to a conventional PERT graph, except that two types of nodes are involved to represent either the choice between activities, or the fact that a set of activities should be completed before starting another set of activities. A formalization of the problem and some important properties concerning the optimal solution are given. Several well-solvable cases of the problem and a powerful decomposition algorithm running in polynomial time are presented. This decomposition is applicable for solving many real-life problems.

KEYWORDS

PERT Network, Shortest Path, Alternative Activities, Makespan Optimization.

* INRIA Lorraine, Technopôle Metz 2000, 4 rue Marconi, 57070 Metz, France
** Center for Technological Education Holon, 52 Golomb St., 58102 Holon, Israël
*** Institute for Systems Research, University of Maryland, College Park, MD 20742, USA

This work has been supported by the French-Israeli Program "Arc-en-Ciel/Keshet" n°64, and INTAS Projects 96-812 and 96-820.

1. INTRODUCTION

In this paper we introduce and analyze PERT-type networks modeling projects with alternative operations. Recall that any PERT network is a finite directed graph G in which arcs represent *activities*, or operations, and nodes represent *events*, that is, starting and ending points of activities. The length of each arc is given; it is assumed to be either a positive or negative constant, or zero.

In the traditional PERT models (see, for instance, [7, 8, 11, 12]), all activities are assumed to be of the *and*-type. An activity i is said to be of the *and*-type if it cannot begin until *all* the activities preceding the activity i in graph G have been completed. However, in many practical cases, a project may include, along with the *and*-type activities, another type of activities known as *alternatives*, or *or*-type activities. An activity is said to be of the *or*-type if it can begin as soon as *at least one* of the preceding activities in G is completed. There are many real-life applications in manufacturing, communication and logistic that are modeled with the help of networks with alternative operations (see, for example, [1, 3-6, 9-10]).

We will consider the following project management problem, called the *PERT-Time Problem with Alternatives* (*PPA*). Given a project containing both *and*-type and *or*-type activities, with two fixed nodes denoted respectively by s (*start*) and f (*final*), the problem is to find starting and ending times of each one of the activities so that the completion time (the makespan) of the entire project is minimum. Obviously, if such a network contains a positive-length directed cycle consisting of only *and*-nodes, or a negative-length directed cycle consisting of only *or*-nodes, the related problem has no finite solution.

The mathematical model of this problem is a combination of the classical PERT-Time (longest path) problem and the routing (shortest path) problem.

Not much have been done to date for efficiently solving scheduling problems with alternative activities. Dinic [3] has found a polynomial-time algorithm for solving the PERT-Time Problem with Alternatives (PPA) on a bipartite graph with arcs of (strictly) positive lengths. This algorithm can be easily extended to find the minimum makespan for any graph (not necessary bipartite) with (strictly) positive arc lengths. To the best of our knowledge, other authors concentrated on the integer-programming, heuristic and branch-and-bound approaches for treating alternative operations in network systems. In an early paper by Stecke and Solberg [13], a simulation experiment is conducted for the real-time control of an FMS with alternative operations. Kusiak and Finke [6] were apparently the first who studied a process selection problem for alternative process plans and presented an integer mathematical programming model to measure the total cost. Nasr and Elsayed [9] considered the problem of minimizing the mean flow time in a machining system with alternative routings and developed a

decomposition method for solving the mixed integer formulation of the problem. Wilhelm and Shin [14] effectively applied a linear programming model and examined the influence of alternative operations on the performance of flexible manufacturing systems. Ahn, He and Kusiak [1] and Gere [4] demonstrated that scheduling in manufacturing systems with alternative process plans and alternative operations permits to increase the throughput rate by a better utilization of scarce resources; these authors developed heuristic algorithms for scheduling alternative operations and studied the effect of alternative operations on the performance of schedules generated by different heuristic rules. Iwata, Murotsu and Oba [5] proposed a branch-and-bound technique to determine a minimum total production time in a flexible system with alternative operations. A branch-and-bound algorithm was developed also by Pan and Chen [10] to minimize the makespan in a two-machine flowshop with alternative operations.

The present paper is devoted to PERT-time Problem with Alternatives (PPA) represented by graphs whose *arc lengths are of any sign*. We will focus on the analysis of some of its main properties. Notice that the optimal solution is neither defined by the longest path (as in the PERT/CPM models with *and*-type activities) nor by the shortest one (as in the routing problem), but rather recursively as a combination of paths of the both types.

The remainder of the paper is organised as follows. In Section 2 we formally define the problem. In Section 3 we prove the uniqueness of the earliest starting times. In Section 4, we present some important properties concerning the optimal solution. In Section 5, we consider several well-solvable cases of the problem. In Section 6 we present a powerful decomposition algorithm running in polynomial time. Conclusion and directions for further research are presented in Section 7.

2. PROBLEM FORMULATION

Let $G=(V, E)$ be a PERT network, that is, a finite directed graph, where V is the set of nodes and E the set of arcs. The network describes a project in which arcs represent activities and nodes represent events (i.e. starting and ending events of the activities of the project). Without loss of generality, we assume that graph G is connected (otherwise, we could treat each one of its connected component separately). The length $l(e)$ of arcs $e \in E$ are given. All arc lengths are assumed to be constant (positive, negative or zero). A positive arc length denotes the duration of the activity corresponding to the arc. Some other arcs of positive lengths are used to represent the *not-before* relations. Such an arc of length $l(i,j)$ means that event j will occur not sooner than $l(i,j)$ units of time after event i occurs. Notice that events i and j may belong to different activities. We also introduce the so-called *no-later* arcs. A *no-later* arc of negative length $l(i;j)$ means that event j must occur at most $l(i,j)$ units of time after event i. A zero-length arc just indicates that an activity preceeds some other activity.

As we mentioned in Introduction, we will consider PERT networks with activities of two types, that is *or*-type and *and*-type. In the remainder of this paper the starting nodes of *or*-type activities will be called *or-nodes*, and the starting nodes of *and*-type activities, *and-nodes* (see Figure 1).

(a) Since T_c and T_d are *or*-activities, at least one of the activities T_a or T_b must be completed before T_c and T_d start. The starting node of T_c and T_d is an *or*-node.

t_a, t_b, t_c and t_d being the starting times of activities T_a, T_b, T_c and T_d respectively, the following inequalities must be satisfied:

$$t_c \geq \min\{t_a + l(T_a);\ t_b + l(T_b)\}, \text{ and}$$
$$t_d \geq \min\{t_a + l(T_a);\ t_b + l(T_b)\}.$$

(b) Since T_c and T_d are *and*-activities, both activities T_a and T_b must be completed before T_c and T_d start. The starting node of T_c and T_d is an *and*-node.

t_a, t_b, t_c and t_d being the starting times of activities T_a, T_b, T_c and T_d respectively, the following inequalities must be satisfied:

$$t_c \geq \max\{t_a + l(T_a);\ t_b + l(T_b)\}, \text{ and}$$
$$t_d \geq \max\{t_a + l(T_a);\ t_b + l(T_b)\}.$$

Figure 1. Description of an *or*-node and an *and*-node.

We will consider the following project management problem, called the *PERT-Time Problem with Alternatives* (*PPA*). Given a project with activities of both *and*- and *or*-types, and two special nodes, s (*start*) and f (*final*), the goal is to find the *starting* and *ending* times of each activity so that the completion time of the project (i.e. the makespan) is minimum.

We assume that there is only one node without predecessors, called the *start* node of the project and denoted by s. (Notice that if there are several nodes having no predecessor, it is always possible to add one *new_start* node and connect this node with the nodes having no predecessors in the initial network by arcs of zero length). Using similar arguments, we assume that there is only one node without successors, called the *final* node of the project and denoted by f.

The set of *or*-nodes (respectively, *and*-nodes) is denoted by R (respectively, A); $V = \{s\} \cup R \cup A$. Notice that node f belongs to either R or A. Let us denote by $Pred(i)$ the set of predecessors of node i, where $i \in R \cup A$: $Pred(i) = \{j / j \in V,\ (j,i) \in E\}$. Notice that if there are several arcs starting from the same node and ending at an *or*-node (respectively, an *and*-node), the arc of the lowest (respectively, greatest) value is the only one which will be considered. Finally, we can assume, without loss of generality, that there exists at most one arc between any two nodes.

For each node i, a parameter t_i, called the *starting time*, is introduced. It represents the starting time of all operations beginning with node i (and the ending time of all operations which end in node i). Without loss of generality, we will assume that the project starts no sooner than at time 0: $t_s \geq 0$.

Our goal is to find for each node i the starting times t_i such that:

$$\begin{Bmatrix} t_i \geq \min\limits_{j \in Pred(i)} \{t_j + l(j,i)\} & \text{if } i \in R \\ t_i \geq \max\limits_{j \in Pred(i)} \{t_j + l(j,i)\} & \text{if } i \in A \end{Bmatrix} \tag{1}$$

and such that the completion time of the entire project (i.e. the time t_f assigned to node f) is minimum.

Formally, we intend to solve the following problem, which we denote as problem P^f_{INEQ}:

Problem P^f_{INEQ} : minimize t_f

subject to

$$\begin{Bmatrix} t_i \geq 0 & \text{if } i = s \\ t_i \geq \min\limits_{j \in Pred(i)} \{t_j + l(j,i)\} & \text{if } i \in R \\ t_i \geq \max\limits_{j \in Pred(i)} \{t_j + l(j,i)\} & \text{if } i \in A \end{Bmatrix} \qquad (S_{\text{INEQ}})$$

3. UNIQUENESS OF THE EARLIEST STARTING TIMES

Let us consider, for a set of arbitrary values $\{\lambda_i\}_{i \in V}$, the following auxiliary problem denoted by $P_{\text{INEQ}}\left(\{\lambda_i\}_{i \in V}\right)$:

Problem $P_{\text{INEQ}}\left(\{\lambda_i\}_{i \in V}\right)$: minimize $\sum\limits_{i \in V} \lambda_i t_i$

so that $\{t_i\}_{i \in V}$ satisfy inequalities (S_{INEQ}).

If there exists a feasible solution to problem P^f_{INEQ}, this solution is also feasible for problem $P_{\text{INEQ}}\left(\{\lambda_i\}_{i \in V}\right)$ for any set of $\{\lambda_i\}_{i \in V}$. Furthermore, if there exists an optimal solution for the problem $P_{\text{INEQ}}\left(\{\lambda_i > 0\}_{i \in V}\right)$ (for a set of (strictly) positive values $\{\lambda_i\}_{i \in V}$), this solution is also optimal to P^f_{INEQ}. This claim follows from Theorem 1 below.

3.1. *Theorem 1 (Uniqueness of the Earliest Starting Times)*

If there exists an optimal solution to problem $P_{\text{INEQ}}\left(\{\lambda_i^0 > 0\}_{i \in V}\right)$ for a certain set of (strictly)

positive values $\{\lambda_i^0\}_{i\in V}$, then:

(i) This solution is the unique optimal solution to $P_{\text{INEQ}}\left(\{\lambda_i > 0\}_{i\in V}\right)$, for any set of arbitrary (strictly) positive values $\{\lambda_i\}_{i\in V}$.

(ii) This solution is also an optimal solution to $P_{\text{INEQ}}\left(\{\lambda_i \geq 0\}_{i\in V}\right)$. Note when some values of the set $\{\lambda_i\}_{i\in V}$ are equal to 0, other optimal solutions may exist.

Proof

(i) Let $\{t_i\}_{i\in V}$ denote an optimal solution to problem $P_{\text{INEQ}}\left(\{\lambda_i^0 > 0\}_{i\in V}\right)$ for a certain set of (strictly) positive values $\{\lambda_i^0\}_{i\in V}$. Assume that there exists another optimal solution to problem $P_{\text{INEQ}}\left(\{\lambda_i^1 > 0\}_{i\in V}\right)$ for a set of (strictly) positive values $\{\lambda_i^1\}_{i\in V}$, denoted by $\{u_i\}_{i\in V}$. Since the two solutions are different, there exists at least one event $i^* \in V$ such that $t_{i^*} \neq u_{i^*}$. Consider three cases.

(a) For all $i \in V$, $t_i \leq u_i$.

Thus, $t_{i^*} < u_{i^*}$. In this case, for any set of (strictly) positive values $\{\lambda_i\}_{i\in V}$, we have: $\sum_{i\in V}\lambda_i t_i < \sum_{i\in V}\lambda_i u_i$. So, for the set of values $\{\lambda_i^1\}_{i\in V}$, the solution $\{t_i\}_{i\in V}$ would be strictly better than $\{u_i\}_{i\in V}$, which contradicts the optimality of $\{u_i\}_{i\in V}$.

(b) For all $i \in V$, $t_i \geq u_i$.

In this case, we have: $\sum_{i\in V}\lambda_i^0 t_i > \sum_{i\in V}\lambda_i^0 u_i$, which contradicts the optimality of $\{t_i\}_{i\in V}$.

(c) There exists $i^{**} \in V$ such that $t_{i^{**}} \neq u_{i^{**}}$ such that either $t_{i^{**}} > u_{i^{**}}$ when $t_{i^*} < u_{i^*}$, or $t_{i^{**}} < u_{i^{**}}$ when $t_{i^*} > u_{i^*}$.

Assume that $t_{i^*} < u_{i^*}$ and $t_{i^{**}} > u_{i^{**}}$ (if not, it is always possible to interchange the two events i^* and i^{**}). Let $\{x_i\}_{i\in V}$ be defined as follows:
$x_i = \min\{t_i, u_i\}$, for all $i \in V$.

If $i = s$ then, by definition of $\{x_i\}_{i\in V}$, we have:
$x_i \geq \min\{t_i, u_i\} \geq 0.$ (2)

For $i \in R$, we can write:
$x_i \geq \min\{t_i, u_i\}$,

$x_i \geq \min\left\{\min_{j\in Pred(i)}\{t_j + l(j,i)\}, \min_{j\in Pred(i)}\{u_j + l(j,i)\}\right\}$ according to (S_{INEQ}).

This can be rewritten as :

$$x_i \geq \min_{j \in Pred(i)} \left\{ \min\{t_j, u_j\} + l(j,i) \right\}.$$

Finally, by definition of x_i, we obtain : $x_i \geq \min_{j \in Pred(i)} \{x_j + l(j,i)\}$. (3)

For $i \in A$, we have:

$$t_i \geq \max_{j \in Pred(i)} \{t_j + l(j,i)\} \geq \max_{j \in Pred(i)} \left\{ \min\{t_j, u_j\} + l(j,i) \right\}, \text{ according to } (S_{INEQ}), \quad (4)$$

$$u_i \geq \max_{j \in Pred(i)} \{u_j + l(j,i)\} \geq \max_{j \in Pred(i)} \left\{ \min\{t_j, u_j\} + l(j,i) \right\}, \text{ according to } (S_{INEQ}). \quad (5)$$

As a consequence of (4) and (5), and by definition of x_i, we obtain :

$$x_i \geq \min\{t_i, u_i\} \geq \max_{j \in Pred(i)} \left\{ \min\{t_j, u_j\} + l(j,i) \right\} \geq \max_{j \in Pred(i)} \{x_j + l(j,i)\}. \quad (6)$$

Thus, according to (2), (3) and (6), $\{x_i\}_{i \in V}$ is a solution satisfying S_{INEQ}. Since $x_{i*} < u_{i*}$ and $x_{i**} < t_{i**}$, it follows that for any set of (strictly) positive values $\{\lambda_i\}_{i \in V}$, $\sum_{i \in V} \lambda_i x_i < \sum_{i \in V} \lambda_i t_i$ and $\sum_{i \in V} \lambda_i x_i < \sum_{i \in V} \lambda_i u_i$. Thus, there exists a feasible solution to $P_{INEQ}\left(\{\lambda_i\}_{i \in V}\right)$, namely $\{x_i\}_{i \in V}$, which is strictly better than both optimal solutions, $\{t_i\}_{i \in V}$ and $\{u_i\}_{i \in V}$. This contradicts the assumption and completes the proof of claim (i).

(ii) Let $\{t_i\}_{i \in V}$ denote the unique optimal solution to problem $P_{INEQ}\left(\{\lambda_i > 0\}_{i \in V}\right)$. Assume that this solution is not optimal to problem $P_{INEQ}\left(\{\lambda_i^2 \geq 0\}_{i \in V}\right)$ for a set of arbitrary non-negative values $\{\lambda_i^2\}_{i \in V}$. Then there exists another solution, satisfying constraints (S_{INEQ}) and denoted by $\{y_i\}_{i \in V}$ such that $\sum_{i \in V} \lambda_i^2 y_i < \sum_{i \in V} \lambda_i^2 t_i$.

Let $\varepsilon = \dfrac{\sum_{i \in V} \lambda_i^2 t_i - \sum_{i \in V} \lambda_i^2 y_i}{\sum_{i \in V} y_i - \sum_{i \in V} t_i}$. We have: $\sum_{i \in V} \lambda_i^2 y_i + \varepsilon \sum_{i \in V} y_i = \sum_{i \in V} \lambda_i^2 t_i + \varepsilon \sum_{i \in V} t_i$.

Since $\{t_i\}_{i \in V}$ is the unique optimal solution to $P_{INEQ}\left(\{\lambda_i > 0\}_{i \in V}\right)$, $\sum_{i \in V} y_i > \sum_{i \in V} t_i$. It follows that $\varepsilon > 0$.

Consider the set of (strictly) positive values $\{\lambda_i^3 = \lambda_i^2 + \varepsilon\}_{i \in V}$. We have:

$$\sum_{i \in V} \lambda_i^3 y_i = \sum_{i \in V} (\lambda_i^2 + \varepsilon) y_i = \sum_{i \in V} \lambda_i^2 y_i + \varepsilon \sum_{i \in V} y_i = \sum_{i \in V} \lambda_i^2 t_i + \varepsilon \sum_{i \in V} t_i = \sum_{i \in V} (\lambda_i^2 + \varepsilon) t_i = \sum_{i \in V} \lambda_i^3 t_i .$$

This contradicts to the fact that $\{t_i\}_{i \in V}$ is the unique optimal solution to problem $P_{INEQ}\left(\{\lambda_i > 0\}_{i \in V}\right)$. It follows that $\{t_i\}_{i \in V}$ is optimal to $P_{INEQ}\left(\{\lambda_i \geq 0\}_{i \in V}\right)$. QED

3.2. Remarks

The optimal solution of the problem $P_{INEQ}\left(\{\lambda_i > 0\}_{i \in V}\right)$ provides the minimum of each starting time t_i. Thus, they are the *earliest starting times* (otherwise we could decrease the values of the starting times t_i which are not minimal, without violating the constraints). In

other words, the optimal solution of $P_{\text{INEQ}}\left(\{\lambda_i > 0\}_{i\in V}\right)$ is also the optimal solution to the multicriteria problem of minimizing the starting times of each single activity.

In what follows, we will focus on the problem $P_{\text{INEQ}}\left(\{\lambda_i > 0\}_{i\in V}\right)$ only. We denote P_{INEQ} the problem $P_{\text{INEQ}}\left(\{\lambda_i = 1\}_{i\in V}\right)$. According Theorem 1, the optimal solution of P_{INEQ} is the optimal solution of the problem $P_{\text{INEQ}}\left(\{\lambda_i > 0\}_{i\in V}\right)$ for any (strictly) positive values $\{\lambda_i\}_{i\in V}$.

3.3. Problems with non-negative solutions

In what follows, we show that the optimal solution, $\{u_i\}_{i\in V}$ to the problem P_{INEQ} related to G subject to the condition that the solution $\{u_i\}_{i\in V}$ is non-negative, can be obtained from the optimal solution to the problem P_{INEQ} (without the non-negativity condition) related to an auxiliary graph G'. The idea of the transformation is to replace t_i in the initial problem by $\max\{0;t_i\}$, in order to obtain the auxiliary problem.

For any problem P_{INEQ} related to graph $G=(V,E)$, consider the following transformation:

(i) For any *or*-node i in R, add an auxiliary *and*-node i'. Denote the set of the additional *and*-nodes by R'. Set $E'=E$.

(ii) Replace any arc (i,j) in E' which starts from an *or*-node i, by the arc (i',j) where $i'\in R'$ is the auxiliary node corresponding to $i\in R$.

(iii) Between any *or*-node i and its corresponding *and*-node $i'\in R'$, add to E' an arc (i,i') of zero length.

(iv) Delete from E' any arc (s,i) of negative length in E' which starts from node s and ends in an *and*-node $i\in A$.

(v) Between node s and any *and*-node i of $A\cup R'$, add to E' an arc (s,i) of zero length, if such arc (s,i) does not exist in E'.

Given $G=(V,E)$, by using transformation (i)-(v), we obtain a new graph $G'=(V\cup R',E')$.

3.4. Theorem 2

If the problem P_{INEQ} related to G' has no optimal solution, then the problem P_{INEQ} related to G has no non-negative solution.

Assume that $\{t_i\}_{i\in V\cup R}$ is the optimal solution to the problem P_{INEQ} related to G'. We define $\{u_i\}_{i\in V}$ as follows:

$$u_i = \begin{cases} t_i & \text{if } i\in A\cup\{s\} \\ t_j \text{ where } j\in R' \text{ is the auxiliary node of } i & \text{if } i\in R \end{cases}$$

Then, $\{u_i\}_{i\in V}$ is the optimal solution to the problem P_{INEQ} related to G subject to the condition that the solution $\{u_i\}_{i\in V}$ is non-negative.

Proof

(a) *If the problem P_{INEQ} related to G' has no optimal solution, then the problem P_{INEQ} related to G has no non-negative solution.*

Assume that there exists a feasible solution $\{u_i\}_{i\in V}$ to problem P_{INEQ} related to G subject to the condition that the solution $\{u_i\}_{i\in V}$ is non-negative. In this case, all starting times u_i will be lower bounded by 0. It follows that there exists an optimal solution to problem P_{INEQ} related to G subject to the condition that the solution $\{u_i\}_{i\in V}$ is non-negative. Denote this optimal solution by $\{u_i^*\}_{i\in V}$.

Let us define $\{t_i^*\}_{i\in V\cup R'}$ as follows:

$$t_i^* = \begin{cases} u_i^* & \text{if } i \in V \\ u_j^* \text{ where } i \in R' \text{ is the auxiliary node of } j \in R & \text{if } i \in R' \end{cases}$$

We can verify that the solution $\{t_i^*\}_{i\in V\cup R'}$ is feasible to the problem P_{INEQ} related to G' (see (ii)-(iii)). Moreover, the starting times t_i^* of each *and*-node i is lower bounded by 0 (see (iv)-(v)) and the starting times t_i^* of each *or*-node i is lower bounded because its predecessors are nodes of *and*-type or s (see (ii)). Since $\{t_i^*\}_{i\in V\cup R'}$ is feasible to the problem P_{INEQ} related to G', there exists an optimal solution to P_{INEQ} related to G'.

(b) *If $\{t_i\}_{i\in V\cup R'}$ is the optimal solution to the problem P_{INEQ} related to G', then $\{u_i\}_{i\in V}$ is the non-negative optimal solution to the problem P_{INEQ}.*

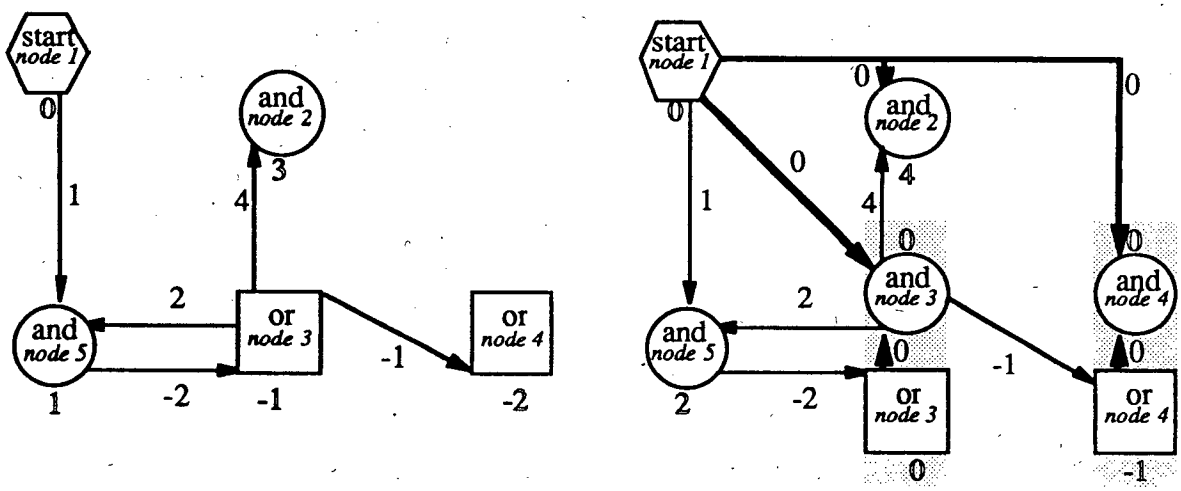
Let $\{t_i\}_{i\in V\cup R'}$ be the optimal solution to the problem P_{INEQ} related to G'. According to Theorem 1, this optimal solution is unique. Since $\{t_i\}_{i\in A\cup\{s\}\cup R'}$ is non-negative (see (iv)-(v)), the solution $\{u_i\}_{i\in V}$ is also non-negative. Moreover, we can see that $\{u_i\}_{i\in V}$ is a feasible solution to problem P_{INEQ} related to G (see (ii)-(iii)).

Assume that there exists a non-negative feasible solution $\{u_i^*\}_{i\in V}$ to problem P_{INEQ} related to G which is better than $\{u_i\}_{i\in V}$. In this case, there would exist $i_0\in V$ such that $u_{i_0}^* < u_{i_0}$. Let us derive $\{t_i^*\}_{i\in V\cup R'}$ from $\{u_i^*\}_{i\in V}$ as we did before (see (a)). We can see that the solution $\{t_i^*\}_{i\in V\cup R'}$ is feasible for the problem P_{INEQ} related to G' (see (ii)-(iii)). According (iii) and by definition of $\{u_i\}_{i\in V}$, for any $i\in V$, we have $u_i \le t_i$. Since there exists $i_0\in V$ such that $u_{i_0}^* < u_{i_0}$, we have $u_{i_0}^* < t_{i_0}$. Moreover, by definition of $\{t_i^*\}_{i\in V\cup R'}$ for any $i\in V$, we have $u_i^* = t_i^*$. This implies that $u_{i_0}^* < t_{i_0}^*$. Thus, the solution $\{t_i^*\}_{i\in V\cup R'}$ is feasible for the problem P_{INEQ} related to G' and there exists $i_0\in V$ such that $u_{i_0}^* < t_{i_0}^*$. This contradicts the uniqueness of the optimal solution $\{t_i\}_{i\in V\cup R'}$ to the problem P_{INEQ} related to G'. It follows that the solution $\{u_i\}_{i\in V}$ is optimal for the problem P_{INEQ} related to G subject to the condition that the solution $\{u_i\}_{i\in V}$ is non-negative. QED

Theorem 2 means that the non-negative optimal solution to problem P_{INEQ} related to graph $G=(V,E)$ can be obtained by solving P_{INEQ} related to the auxiliary graph $G'=(V\cup R', E')$.

Note that solving P_{INEQ} related to G' is not subject to the non-negative constraint on the solution.

Examples 1:



(a) The lengths are indicated near the corresponding arcs. $G=(V,E)$ corresponds to this graph. The solution, given by the outlined numbers under each node, is the optimal solution to problem P_{INEQ} related to G. This optimal solution, namely {0;3;-1;-2;1}, is negative.

(b) The auxiliary graph $G'=(V \cup R', E')$ associated with $G=(V,E)$ corresponds to this graph. The solution, given by the outlined numbers near each node, is the optimal solution to P_{INEQ} related to G'. The solution {0;4;0;0;2} is optimal to the problem P_{INEQ} related to $G=(V,E)$ under constraints saying that the solution should be non-negative.

Figure 2. An initial graph and the corresponding auxiliary graph.

4. SATURATED CONSTRAINTS AND RECIPROCAL PROBLEMS

To study problem P_{INEQ}, we consider the associated problem P_{EQ} in which all constraints (S_{INEQ}) are converted into equalities:

Problem P_{EQ} : minimize $\sum_{i \in V} t_i$

subject to

$$\begin{cases} t_i = 0 & \text{if } i = s \\ t_i = \min\limits_{j \in Pred(i)} \{t_j + l(j,i)\} & \text{if } i \in R \\ t_i = \max\limits_{j \in Pred(i)} \{t_j + l(j,i)\} & \text{if } i \in A \end{cases} \qquad (S_{EQ})$$

4.1. Theorem 3 (Saturation of Constraints in the Optimal Solution of P_{INEQ})
The two problems, P_{INEQ} and P_{EQ}, have the same (unique) optimal solution.

(A version of this claim was communicated to one of the authors by E. Dinic in 1984. The proof proposed hereafter is different.)

Proof

(i) *The optimal solution of P_{INEQ} is optimal to P_{EQ}.*
Let $\{t_i^0\}_{i \in V}$ be any solution satisfying constraints (S_{INEQ}), but not to (S_{EQ}).
Let $\{t_i^1\}_{i \in V}$ be the set of values defined as follows:

$$\begin{cases} t_i^1 = 0 & \text{if } i = s \\ t_i^1 = \min\limits_{j \in Pred(i)} \{t_j^0 + l(j,i)\} & \text{if } i \in R \\ t_i^1 = \max\limits_{j \in Pred(i)} \{t_j^0 + l(j,i)\} & \text{if } i \in A \end{cases}$$

Thus, for all $i \in V$, $t_i^0 \geq t_i^1$ and, as a consequence:

$$\begin{cases} t_i^1 \geq 0 & \text{if } i = s \\ t_i^1 \geq \min\limits_{j \in Pred(i)} \{t_j^0 + l(j,i)\} \geq \min\limits_{j \in Pred(i)} \{t_j^1 + l(j,i)\} & \text{if } i \in R \\ t_i^1 \geq \max\limits_{j \in Pred(i)} \{t_j^0 + l(j,i)\} \geq \max\limits_{j \in Pred(i)} \{t_j^1 + l(j,i)\} & \text{if } i \in A \end{cases}$$

Hence, the solution $\{t_i^1\}_{i \in V}$ satisfies (S_{INEQ}).
Since $\{t_i^0\}_{i \in V}$ does not satisfy (S_{EQ}) but satisfies (S_{INEQ}), there exists $i^* \in V$ such that

$$\begin{array}{lll} \text{either} & t_{i^*}^0 > 0 & \text{if } i^* = s, \\ \text{or} & t_{i^*}^0 > \min\limits_{j \in Pred(i^*)} \{t_j^0 + l(j,i^*)\}. & \text{if } i^* \in R, \\ \text{or} & t_{i^*}^0 > \max\limits_{j \in Pred(i^*)} \{t_j^0 + l(j,i^*)\} & \text{if } i^* \in A. \end{array}$$

It follows that $t_{i^*}^1 < t_{i^*}^0$ and, $\sum_{i \in V} t_i^1 < \sum_{i \in V} t_i^0$. It implies that $\{t_i^0\}_{i \in V}$ is not optimal for P_{INEQ}.

Thus, the optimal solution of P_{INEQ} cannot satisfy (S_{INEQ}) without satisfying (S_{EQ}). Since the two problems P_{INEQ} and P_{EQ} refer to the same criterion, the optimal solution to P_{INEQ} is an optimal solution to P_{EQ}.

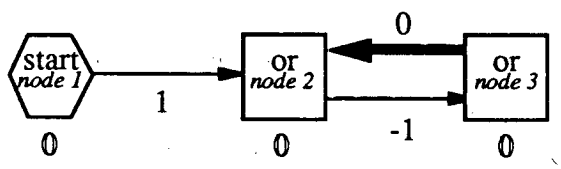
(ii) *An optimal solution of P_{EQ} is optimal to P_{INEQ}.*

It is obvious that any solution which satisfies (S_{EQ}) also satisfies (S_{INEQ}). Thus, the optimal solution of P_{EQ} satisfies (S_{INEQ}). Since the two problems refer to the same criterion, an optimal solution to P_{EQ} is an optimal solution to P_{INEQ}. QED

4.2. Remarks

Since the two problems P_{INEQ} and P_{EQ} have the same optimal solution (if any), if one of the two problems has an optimal solution, the other one has the same optimal solution. And, if one of the two problems P_{INEQ} and P_{EQ} has no optimal solution, then the other one has no optimal solution either.

A feasible solution for problem P_{EQ} is also feasible for P_{INEQ}. But, the reverse of this claim is not true. Indeed, if there exists a feasible solution for P_{INEQ} but no optimal solution to P_{INEQ} is reachable, it is possible that P_{EQ} has no feasible solution (as can be seen in Figure 3).



The lengths are indicated near the corresponding arcs. The solution, given by the outlined numbers under each node, belongs to (S_{INEQ}). But, there is no feasible solution satisfying constraints (S_{EQ}). Thus, problem P_{INEQ} is *well-defined* while the corresponding problem P_{EQ} is *inconsistent*. Both problems P_{INEQ} and P_{EQ} have no optimal solutions.

Figure 3. A problem instance well-defined for P_{INEQ} but inconsistent for P_{EQ}.

4.3. Reciprocal problem

Let us introduce $P_{INEQ\text{-}OPP}$, the *reciprocal problem* of P_{INEQ}, in which all inequalities are the reverse of the constraints in problem P_{INEQ}:

Problem $P_{INEQ\text{-}OPP}$: maximize $\sum_{i \in V} t_i$

subject to

$$\left\{ \begin{array}{ll} t_i \le 0 & \text{if } i = s \\ t_i \le \min\limits_{j \in Pred(i)} \{t_j + l(j,i)\} & \text{if } i \in R \\ t_i \le \max\limits_{j \in Pred(i)} \{t_j + l(j,i)\} & \text{if } i \in A \end{array} \right\} \qquad (S_{INEQ\text{-}OPP})$$

Replacing t_i by $(-t'_i)$, we reformulate $P_{\text{INEQ-OPP}}$ as follows:

Problem $P_{\text{INEQ-OPP2}}$: minimize $\sum_{i \in V} t'_i$

subject to

$$\begin{cases} t'_i \geq 0 & \text{if } i = s \\ t'_i \geq \max\limits_{j \in Pred(i)} \{t'_j - l(j,i)\} & \text{if } i \in R \\ t'_i \geq \min\limits_{j \in Pred(i)} \{t'_j - l(j,i)\} & \text{if } i \in A \end{cases}$$

Since problem $P_{\text{INEQ-OPP}}$ has the same structure as a problem of type P_{INEQ} in which the nodes of the two types are interchanged and arc length values are of the opposite sign, Theorem 3 is valid for $P_{\text{INEQ-OPP}}$.

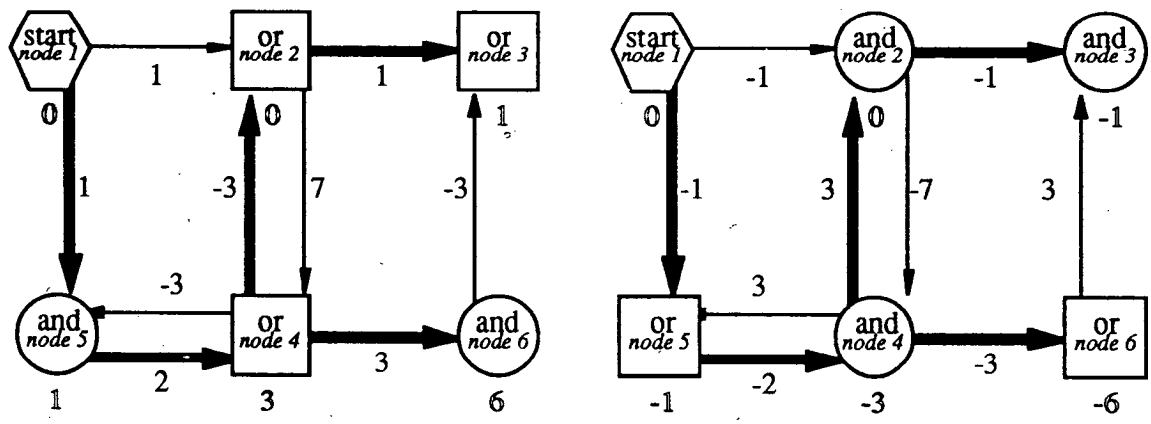
*4.4. **Theorem 4** (Saturation of Constraints in the Optimal Solution of $P_{\text{INEQ-OPP}}$)*

The three problems P_{INEQ}, $P_{\text{INEQ-OPP}}$ and P_{EQ} have the same (unique) optimal solution (if any). This solution is obtained by changing signs of the optimal solution to $P_{\text{INEQ-OPP2}}$.

Proof

Similarly to the proof of Theorem 3, we prove that the two problems, $P_{\text{INEQ-OPP}}$ and P_{EQ}, have the same (unique) optimal solution. The proof is straighforward, using Theorem 3. QED

Example 2:



(a) The lengths are indicated near the corresponding arcs. The solution given by the outlined numbers under each node, is optimal to P_{INEQ}, P_{EQ} and $P_{\text{INEQ-OPP}}$.

(b) The solution given by the outlined numbers under each node, is optimal to $P_{\text{INEQ-OPP2}}$ and is composed with the opposite values of the optimal solution to P_{INEQ}.

Figure 4. An illustration to Theorem 4.

5. WELL SOLVABLE CASES

5.1 Definitions

A node is said to be *neutral* if it has exactly one successor. It is obvious that relation (1) is identical for a neutral *and*-node and a neutral *or*-node. Thus, such a neutral node can be considered either as an *and*- or *or*-node.

A set of nodes is called *heterogeneous* if it contains non-neutral nodes of both types. A set of nodes which is not heterogeneous is called *homogeneous*.

5.2. Case of a homogeneous graph

If the set of all nodes V is homogeneous, then we can solve P_{INEQ} using classical algorithms devoted to the PERT-Time Problem (if the nodes are either *and*-nodes or neutral nodes) or routing algorithms (if the nodes are either *or*-nodes or neutral nodes). For example, the $O(|V|^2)$-time Dijkstra algorithm (in the case of non-negative lengths) and the $O(|V|.|E|)$-time Bellman-Ford algorithm (for arc length of any sign) can be used to solve the latter problems (see, for instance, [2]).

5.3. Case of a graph with arcs of positive length

Dinic [3] has modified the Dijkstra algorithm in order to solve the PERT-Time Problem with Alternatives (PPA) in the case of positive arc lengths. His algorithm runs in $O(|V|^2)$ time. Although this author has restricted his considerations to bipartite graphs only, his algorithm can be extented to any graph with non-negative arc lengths and no null-circuit. This extented version (using labeling techniques and notations somewhat different from [3]) is presented below.

In this algorithm, we denote by u_i the temporary label assigned to an *or*-node i and by t_i the final label of any node i (which is the optimal starting time). By F, we denote the set of nodes i for which t_i is defined. At the beginning, s is the only node for which the optimal starting time is known: $t_s := 0$ and $F := \{s\}$. The temporary label u_i of any *or*-node i is initially set at $u_i := +\infty$. A current step of the algorithm consists of three stages:

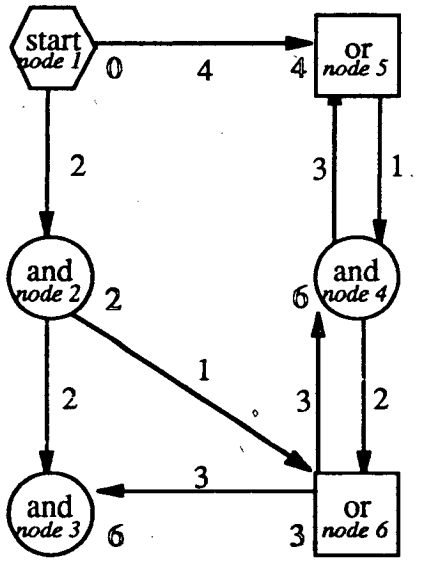
(i) The final label t_i of any *and*-node i is computed as soon as all the predecessors of i belong to F, using the rule: $t_i := \max_{j \in Pred(i)} \{t_j + l(j,i)\}$. Such node i is added to the set F, i.e. $F := F \cup \{s\}$.

(ii) The temporary labels $\{u_i\}$ of the *or*-node are computed iteratively, using the rule: $u_i := \min\left\{ \min_{j \in (Pred(i)/F) \cap R} \{u_j + l(j,i)\}; \min_{j \in Pred(i) \cap F} \{t_j + l(j,i)\} \right\}$, until none of them can be modified.

(iii) The *or*-node i^* is selected such that $u_{i^*} = \min_{i \in R/F} \{u_i\}$ and the final label t_{i^*} is set at $t_{i^*} := u_{i^*}$. Node i^* is added to the set F.

Stages (i)-(iii) are repeated until all the nodes reach their final labels (i.e. fall into F), or until some nodes cannot reach a (finite) final label. This happens when all the u_i values of the *or*-nodes which do not belong to F are equal to infinity, and, for each *and*-node which does not belong to F, there exists at least one predecessor which also does not belong to F. In the latter case, the problem has no feasible solution.

Example 3:



step	node 1	node 2	node 3	node 4	node 5	node 6
init.	$t_1=0$				$u_5=+\infty$	$u_6=+\infty$
(i)		$t_2=2$				
(ii)					$u_5=4$	$u_6=3$
(iii)						$t_6=3$
(i)			$t_6=6$			
(ii)					$u_5=4$	
(iii)					$t_5=4$	
(i)				$t_6=3$		

Figure 5. Solving a problem with positive arc lengths.

5.4. Case of a graph with arcs of negative length

For any problem P_{INEQ} with non-positive arc lengths and no null-circuit, it is possible to transform it in a problem $P_{\text{INEQ-OPP2}}$ with non-negative arc lengths and no null-circuit. This problem $P_{\text{INEQ-OPP2}}$ is solvable by the previous algorithm (see Section 5.3). According to Theorem 4, from the optimal solution of $P_{\text{INEQ-OPP2}}$, we can obtain the optimal solution of the problems $P_{\text{INEQ-OPP}}$ and also of P_{INEQ}.

5.5. Case of a circuit

We consider the case where the graph $G=(V,E)$ constains node s and a circuit (directed cycle) of p nodes, namely $(i_0, i_1, \ldots, i_p=i_0)$. Thus, $V=\{s, i_0, i_1, \ldots, i_{p-1}\}$. For any node i_q, $0\leq q\leq p-1$, there is only one arc starting from i_q (this arc ends in node i_{q+1}). The other arcs start from node s and end in one of the node i_q, where $0\leq q\leq p-1$.

It follows from Theorem 2 that in the optimal solution of P_{INEQ}, the starting time of node s is equal to 0: $t_s=0$. We define a starting label of node i_q as follows:

$$e_{i_q} = \begin{cases} l(s,i_q) & \text{if } s \in Pred(i_q) \\ +\infty & \text{if } s \notin Pred(i_q) \text{ and } i_q \in R \\ -\infty & \text{if } s \notin Pred(i_q) \text{ and } i_q \in A \end{cases} \quad (7)$$

We propose, a priori, a formulation of the starting times, we show that this formulation is consistent, and we provide an algorithm which leads to the solution. Assume that the starting time of node i_q, $0 \le q \le p-1$, is defined as follows:

$$t_{i_q} = t_{i_q}(x) = \begin{cases} a_{i_q} + d_{i_q} & \text{if } \quad x < a_{i_q} \\ x + d_{i_q} & \text{if } a_{i_q} \le x \le b_{i_q} \\ b_{i_q} + d_{i_q} & \text{if } b_{i_q} < x \end{cases} \quad (8)$$

for $x \in \Re \cup \{+\infty; -\infty\}$,
where $a_{i_q} \le b_{i_q} \in \Re \cup \{+\infty; -\infty\}$, and $d_{i_q} \in \Re$ are given.

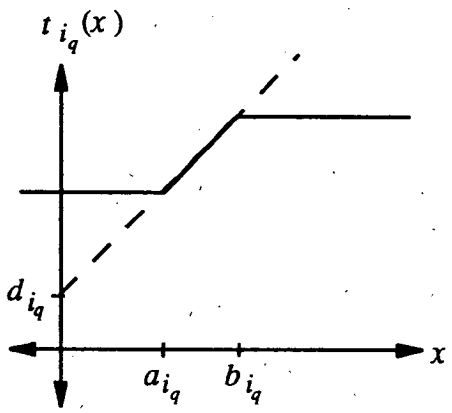


Figure 6. Representation of t_{i_q} function of value x.

Consider i_{q+1}, the sucessor of node i_q in the cycle. According to (S_{EQ}), we have:

$$t_{i_{q+1}} = \begin{cases} \min\limits_{j \in Pred(i_{q+1})} \{t_j + l(j,i_{q+1})\} = \min\{t_{i_q} + l(i_q,i_{q+1}); e_{i_{q+1}}\} & \text{if } i_{q+1} \in R \\ \max\limits_{j \in Pred(i_{q+1})} \{t_j + l(j,i_{q+1})\} = \max\{t_{i_q} + l(i_q,i_{q+1}); e_{i_{q+1}}\} & \text{if } i_{q+1} \in A \end{cases} \quad (9)$$

From (8) and (9), we derive:

$$t_{i_{q+1}} = t_{i_{q+1}}(x) = \begin{cases} \min\{a_{i_q} + d_{i_q} + l(i_q,i_{q+1}); e_{i_{q+1}}\} & \text{if } i_{q+1} \in R \text{ and if } \quad x < a_{i_q} \\ \max\{a_{i_q} + d_{i_q} + l(i_q,i_{q+1}); e_{i_{q+1}}\} & \text{if } i_{q+1} \in A \text{ and if } \quad x < a_{i_q} \\ \min\{x + d_{i_q} + l(i_q,i_{q+1}); e_{i_{q+1}}\} & \text{if } i_{q+1} \in R \text{ and if } a_{i_q} \le x \le b_{i_q} \\ \max\{x + d_{i_q} + l(i_q,i_{q+1}); e_{i_{q+1}}\} & \text{if } i_{q+1} \in A \text{ and if } a_{i_q} \le x \le b_{i_q} \\ \min\{b_{i_q} + d_{i_q} + l(i_q,i_{q+1}); e_{i_{q+1}}\} & \text{if } i_{q+1} \in R \text{ and if } b_{i_q} < x \\ \max\{b_{i_q} + d_{i_q} + l(i_q,i_{q+1}); e_{i_{q+1}}\} & \text{if } i_{q+1} \in A \text{ and if } b_{i_q} < x \end{cases} \quad (10)$$

We consider three cases. It turns out from (10) that:

(a) if $e_{i_{q+1}} < a_{i_q} + d_{i_q} + l(i_q, i_{q+1})$, then:

$$d_{i_{q+1}} = \begin{cases} e_{i_{q+1}} & \text{if } i_{q+1} \in R \\ d_{i_q} + l(i_q, i_{q+1}) & \text{if } i_{q+1} \in A \end{cases}$$

$$a_{i_{q+1}} = a_{i_q} \qquad (11)$$

$$b_{i_{q+1}} = \begin{cases} a_{i_q} & \text{if } i_{q+1} \in R \\ b_{i_q} & \text{if } i_{q+1} \in A \end{cases}$$

(b) if $a_{i_q} + d_{i_q} + l(i_q, i_{q+1}) \le e_{i_{q+1}} \le b_{i_q} + d_{i_q} + l(i_q, i_{q+1})$, then:

$$d_{i_{q+1}} = d_{i_q} + l(i_q, i_{q+1})$$

$$a_{i_{q+1}} = \begin{cases} a_{i_q} & \text{if } i_{q+1} \in R \\ e_{i_{q+1}} - d_{i_q} - l(i_q, i_{q+1}) & \text{if } i_{q+1} \in A \end{cases} \qquad (12)$$

$$b_{i_{q+1}} = \begin{cases} e_{i_{q+1}} - d_{i_q} - l(i_q, i_{q+1}) & \text{if } i_{q+1} \in R \\ b_{i_q} & \text{if } i_{q+1} \in A \end{cases}$$

(c) if $b_{i_q} + d_{i_q} + l(i_q, i_{q+1}) < e_{i_{q+1}}$, then:

$$d_{i_{q+1}} = \begin{cases} d_{i_q} + l(i_q, i_{q+1}) & \text{if } i_{q+1} \in R \\ e_{i_{q+1}} & \text{if } i_{q+1} \in A \end{cases}$$

$$a_{i_{q+1}} = \begin{cases} a_{i_q} & \text{if } i_{q+1} \in R \\ b_{i_q} & \text{if } i_{q+1} \in A \end{cases} \qquad (13)$$

$$b_{i_{q+1}} = b_{i_q}$$

We see that the starting time of node i_{q+1}, $0 \le q \le p-1$, satisfies (8). We propose an algorithm derived from the above computations aimed to obtain the starting times of each node in the directed cycle:

(i) **Set** $a_{i_0} := -\infty$, $b_{i_0} := +\infty$, $d_{i_0} := 0$.

(ii) **For** q:=1 **to** p **do:**

Compute e_{i_q} (according to relation (7)).

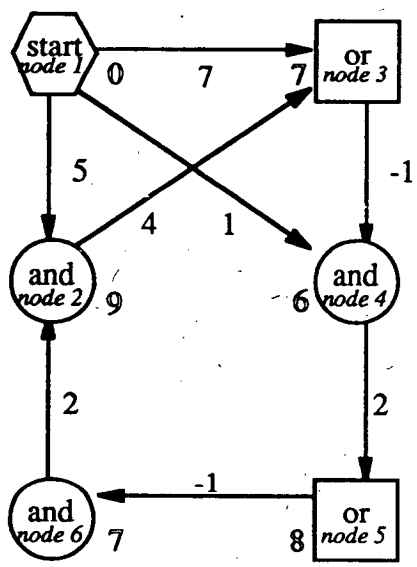
Compute a_{i_q}, b_{i_q} and d_{i_q} (according to relations (11)-(13)).

(iii) **If** $d_{i_p} \leq 0$ **then set** $t_{i_0} := d_{i_p} + a_{i_p}$,
else set $t_{i_0} := d_{i_p} + b_{i_p}$.

[**Comment.** According to (i), $t_{i_0} = t_{i_0}(x) = x$, and we know that $t_{i_0}(x)$ and $t_{i_p}(x)$ should be equal. This justifies (iii).]

(iv) **If** $t_{i_0} = +\infty$ **then return** "No solution",
else if $t_{i_0} = -\infty$ **then return** "No (finite) optimal solution",
else for q:=1 **to** p-1 **do:**
Compute t_{i_q} (according to relation (9)).

Example 4:



step	node i_q	e_{i_q}	a_{i_q}	b_{i_q}	d_{i_q}	t_{i_q}
(i)	node 2		-∞	+∞	0	x
(ii) (12)	node 3	7	-∞	3	4	
(ii) (12)	node 4	1	-2	3	3	
(ii) (13)	node 5	+∞	-2	3	5	
(ii) (11)	node 6	-∞	-2	3	4	
(ii) (12)	node 2	5	-1	3	6	
(iii)	node 2					x =9
(iv)	node 3					7
(iv)	node 4					6
(iv)	node 5					8
(iv)	node 6					7

Figure 7. Solving a problem with a circuit.

Since the graph is constituted by node s and a directed cycle of p nodes, we have: $|E| \leq 2|V|-2$. Thus, the complexity of the previous algorithm is $O(|V|)$.

5.6. Case of an acyclic graph

If the graph $G=(V,E)$ does not contain any directed cycle, it is possible to use the breadth first search (BFS) which marks the nodes of the network in layers, one after another. Nodes are labelled using the relations of (S_{BQ}). The complexity of this algorithm is $O(|V|+|E|)$ (see [2]). This method is extented in the next section.

6. DECOMPOSITION SCHEME

6.1. Notations

Let U be a non-empty set of nodes in $G=(V,E)$.

$Prec(U)$ is the set of the predecessors of the nodes of U:
$Prec(U)=\{j$ such that $j=Prec(i)$ and $i\in U\}$.

$Prec(U)/U$ is the set of the nodes of $Prec(U)$ which do not belong to U:
$Prec(U)/U=\{j$ such that $j=Prec(i)$, $i\in U$ and $j\in V/U\}$.

Let $P_{INEQ}(U)$ be the problem P_{INEQ} defined on U such that the t_i values for $i\in Prec(U)/U$ are known. It is formally defined as follows:

$$\text{Problem } P_{INEQ}(U): \quad \text{minimize } \sum_{i\in U} t_i$$

subject to

$$(S_{INEQ}(U)) \qquad \begin{cases} t_i \geq 0 & \text{if } i=s \text{ and } s\in U \\ t_i \geq \min\limits_{j\in Pred(i)}\{t_j+l(j,i)\} & \text{if } i\in R\cap U \\ t_i \geq \max\limits_{j\in Pred(i)}\{t_j+l(j,i)\} & \text{if } i\in A\cap U \end{cases}$$

where t_i are fixed, for $i\in Pred(U)/U$.

Note that the t_i values for $i\notin U\cup Prec(U)$ are neither fixed, nor included in the set of variables of $P_{INEQ}(U)$. In fact, these values do not influence $P_{INEQ}(U)$ as indicated in the following Theorem.

6.2. Theorem 5 (Decomposition Scheme)

Assume that U is a set of nodes of $G=(V,E)$ such that the given starting times of all the nodes of $Prec(U)/U$ are optimal to P_{INEQ} then the starting times of the nodes of U in the optimal solution to P_{INEQ} and in the optimal solution to $P_{INEQ}(U)$ are identical.

Proof

Let $\{t_i^*\}_{i\in V}$ be the unique optimal starting times of P_{INEQ}. According to Theorem 1, none of the t_i^*, $i\in V$, can be reduced. So, the restriction of $\{t_i^*\}_{i\in V}$ to U is optimal to the problem $P_{INEQ}(U)$. Since the problem $P_{INEQ}(U)$ has a *unique* optimal solution (see Theorem 1), the starting times of the nodes of U in the optimal solution to P_{INEQ} and in the optimal solution to $P_{INEQ}(U)$ are identical. QED

6.3. Remark

It follows from Theorem 2 that in the optimal solution of P_{INEQ}, the starting time of node s is equal to 0, i.e. $t_s=0$. From Theorem 5, we see that if the problem $P_{INEQ}(U)$ has no optimal solution, the problem P_{INEQ} has no optimal solution, either.

Theorem 5 allows us to solve, in some particular cases, problems P_{INEQ} using the solutions of its subproblem $P_{INEQ}(U)$. We propose to decompose the graph $G=(V,E)$ corresponding to P_{INEQ} into subproblems $P_{INEQ}(U_0)$, $P_{INEQ}(U_1)$, ..., $P_{INEQ}(U_c)$, where $\{U_0,U_1,\ldots,U_c\}$ is the

set of strongly connected components of graph $G=(V,E)$. The strongly connected components of G can be sorted solved in a topological order (see [2]) and then the subproblems $P_{INEQ}(U_0)$, $P_{INEQ}(U_1)$, ..., $P_{INEQ}(U_c)$ will be solved in this order. Each one of the subproblems $P_{INEQ}(U_0)$, $P_{INEQ}(U_1)$, ..., $P_{INEQ}(U_c)$ is to be solved separately. Doing so, we are sure that the starting times of the nodes of $Prec(U)/U$ have been computed before considering the subproblem $P_{INEQ}(U)$. Thus, in order to solve each subproblem $P_{INEQ}(U)$, we can replace the arcs (j,i) from nodes j of $Prec(U)/U$ to each node i of U by an arc (s,i) having a length equal to $l(s,i)=t_j+l(j,i)$ where t_j is the optimal starting time of node j. We may use one of the efficient algorithm presented in Section 5 when solving each subproblem $P_{INEQ}(U)$.

Notice that the fully connected components of $G=(V,E)$ can be found and numbered by a topological order using a $O(|V|+|E|)$-time algorithm (see [2]).

6.4. Decomposition Algorithm

Let $\{U_0,U_1,\ldots,U_c\}$ be the set of the strongly connected components of graph $G=(V,E)$, where $\{s\}=U_0$ and $\bigcup_{l=0}^{c} U_l = V$. Assuming that we know how to solve efficiently each subproblem $P_{INEQ}(U_1)$, $P_{INEQ}(U_2)$, ..., $P_{INEQ}(U_c)$, (independently) then, it is possible to solve the initial problem P_{INEQ}, by applying the following algorithm:

(i) **Set** $W:=\{s\}$; **Set** $t_s:=0$.

(ii) **While** $W\neq V$ **do:**

Select U_m, a trongly connected component of graph $G=(V,E)$, such that $U_m \not\subset W$ and $Prec(U_m)/U_m \subseteq W$.

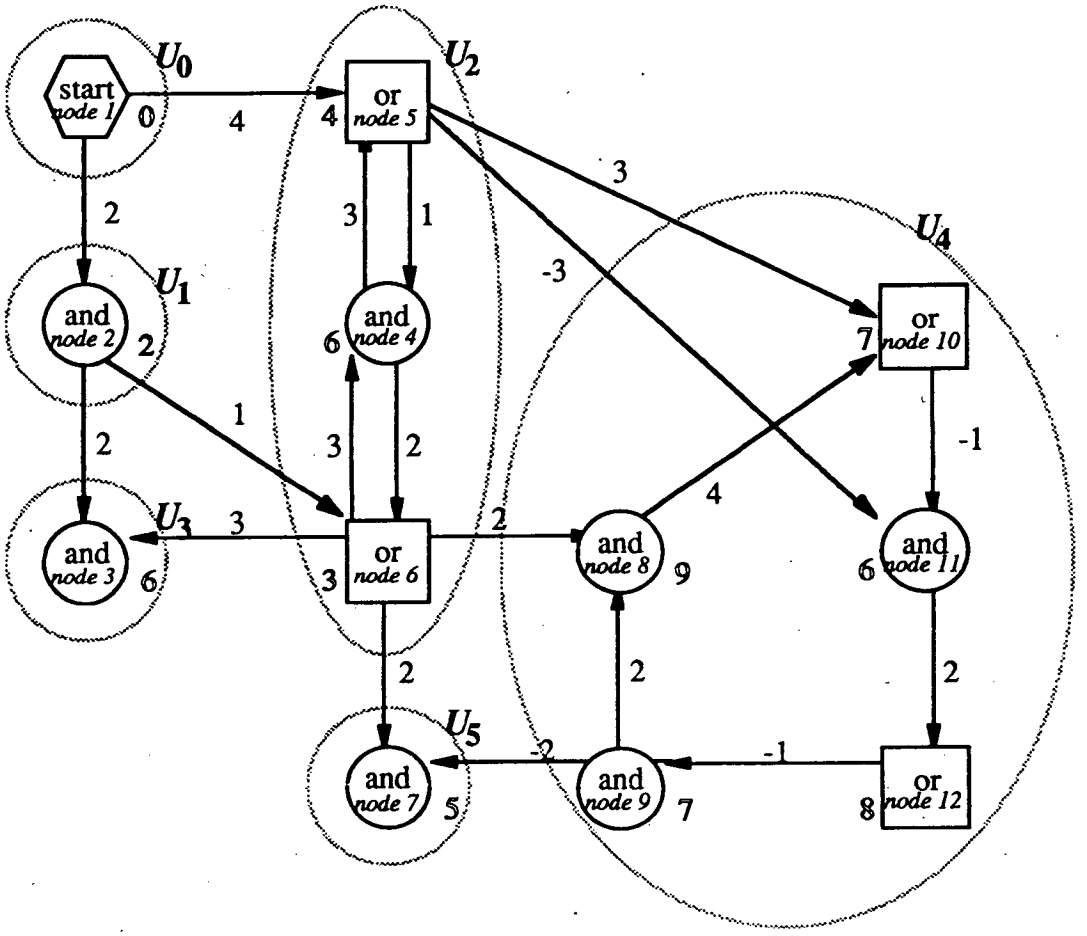
Solve $P_{INEQ}(U_m)$, taking into account the optimal starting times of the nodes of $Prec(U_m)/U_m$.

If $P_{INEQ}(U_m)$ has no optimal solution

then return "the problem P_{INEQ} has no optimal solution", and **stop**.

Set $W:=U_m \cup W$.

End



The fully connected components are encircled and numbered in a topological order. Problem $P_{INEQ}(U_3)$ can be solved using the algorithm described in Section 5.3 while problem $P_{INEQ}(U_4)$ is described in Section 5.5.

Figure 8. An example of the decomposition scheme.

7. CONCLUSION

This work is devoted to the analysis of PERT-Time Problems with Alternatives (PPA). This mathematical model is shown to be combinations of the classical PERT-Time (longest path) problem and the routing (shortest path) problem. While in the traditional PERT models all activities are assumed to be of the *and*-type, we introduce another type of activities known as *alternative*, or *or*-type activities. We are interested in PPA whose *arc lengths are of any sign.*

A formalisation of the problem and important properties concerning the optimal solution are given. Several well-solvable cases of the problem, and a powerful decomposition algorithm running in polynomial time, are presented. We are currently trying to extend this work in order to solve the general PPA.

8. REFERENCES

1. Ahn, J., He, W., and Kusiak, A., Scheduling with alternative operations, *IEEE Journal of Robotics and Automation* (1993) 9, 297-303.

2. Cormen, T.H., Leiserson C.E., and Rivest, R.L., *Introduction to Algorithms*, The MIT Press, Cambridge, Massachussetts (1994).

3. Dinic, E.A., The fastest algorithm for the PERT problem with AND- and OR-nodes. (The New-Product-New-Technology Problem), *Proceedings of the Workshop on Combinatorial Optimization*, Waterloo, 1990, University of Waterloo Press, Waterloo, 1990, 185-187.

4. Gere, W.S., Heuristics in job shop scheduling, *Management Science* (1996) 13, 167-190.

5. Iwata, K., Murotsu, Y., and Oba, F., Solution of large-scale scheduling problems for job-shop machining systems with alternative machine tools, *Annals of CIRP* (1980) 29, 335-338.

6. Kusiak, A., and Finke, G., Selection of process plans in automated manufacturing systems, *IEEE Journal of Robotics and Automation* (1988) 4, 397-402.

7. Levner, E.V., and Nemirovsky, A.S., A network flow algorithm for just-in-time project scheduling, *European Journal of Operational Research* (1994) 79, 167-175.

8. Moder, J.J., Phillips, C.R., and Davis, E.W., *Project Management with CPM, PERT and Precedence Diagramming*, Van Nostrand Reinhold, New York, (1983).

9. Nasr, N., and Elsayed, E.A., Job shop scheduling with alternative machines, *International Journal of Production Research* (1990) 28, 1595-1609.

10. Pan, C.H., and Chen, J.S., Scheduling alternative operations in two-machine flow-shops, *Journal of the Operational Research Society* (1997) 48, 553-540.

11. Shtub, A., Bard, J., and Globerson, C., *Project Management: Engineering Technology and Implementation*, Prentice Hall (1994).

12. Slowinski, R., Weglarz, J., (eds.) *Recent Advances in Project Scheduling*, Elsevier, Amsterdam (1989).

13. Stecke, K.E., and Solberg, J.J., Loading and control policies for a flexible manufacturing system, *International Journal of Production Research* (1981) 19, 481-490.

14. Wilhelm, W.E., and Shin, H., Effectiveness of alternate operations in a flexible manufacturing system, *International Journal of Production Research* (1985) 23, 65-79.

Unité de recherche INRIA Lorraine
Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - B.P. 101 - 54602 Villers lès Nancy Cedex (France)

Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot St Martin (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - B.P. 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, B.P. 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399