Recherche opérationnelle/Operations Research
(vol. 27, n° 2, 1993, p. 153 à 168)

ON THE MINIMUM DUMMY-ARC PROBLEM (*)

by David J. MICHAEL ([1]), Jerzy KAMBUROWSKI ([2])
and Matthias STALLMANN ([3])

Abstract. – *A precedence relation can be represented non-uniquely by an activity on arc (AoA) directed acyclic graph (dag). This paper deals with the NP-hard problem of constructing an AoA dag having the minimum number of arcs among those that have the minimum number of nodes. We show how this problem can be reduced in polynomial time to the set-cover problem so that the known methods of solving the set-cover problem can be applied. Several special cases that lead to easy set-cover problems are discussed.*

Keywords: Activity networks, dummy activities, set-cover.

Résumé. – *Une relation de précédence peut être représentée (d'une manière non-unique) par un graphe direct acyclique (gda) avec activités sur arcs (AsA). Cet article traite du problème NP-difficile de la construction d'un gda AsA ayant le nombre minimal d'arcs parmi ceux qui ont le nombre minimal de sommets. Nous montrons comment ce problème peut se réduire en un temps polynomial au problème de recouvrement d'ensembles, en sorte que l'on peut appliquer les méthodes connues de résolution de ce dernier problème. Nous examinons plusieurs cas spéciaux qui conduisent à des problèmes faciles de recouvrements d'ensembles.*

Mots clés : Réseaux d'activités, activités fictives, recouvrement d'ensembles.

1. INTRODUCTION

The activities of a project are often constrained by conditions such as "activity v cannot start until activity u has finished". Assuming that no activity is repeated we can define a precedence relation $\prec$ on the activities, so that $u \prec v$ means that u must finish before v starts. The relation $\prec$ can be represented graphically in two different ways, by either assigning the

(*) Received November 1990.
([1]) International Business Machines Corporation, P.O. Box 60000, Cary, NC 27511
([2]) Department of Information Systems and Operations Management, University of Toledo, Toledo, OH 43606-3390.
([3]) Department of Computer Science, North Carolina State University, Raleigh, NC 27695-8206.

Recherche opérationnelle/Operations Research, 0399-0559/93/02 153 16/$3.60

activities to the nodes or to (a subset of) the arcs. In either case a *directed acyclic graph (dag)* is defined. In an *activity on node* (AoN) dag each activity corresponds one-to-one with a node, and we say that $u \prec v$ is *represented* if there is a directed path of arcs leading from v's node to u's node. Thus an AoN dag is unique except for possible transitive arcs. In an *activity on arc* (AoA) dag, each activity v corresponds to an arc, where parallel arcs that share the same start and terminal nodes are permitted. We say $u \prec v$ is *represented* in an AoA dag if there is a path from t_u, the terminal node of the arc for u, to s_v, the start node of the arc for v (the path is empty if $t_u = s_v$). Additional *dummy arcs* may have to be added to represent all constraints of $\prec$, and no canonical method for adding dummy arcs has been agreed to. Thus an AoA dag is not unique.

This paper shows how to construct an AoA dag that has the minimum number of dummy arcs given that it has the minimum number of nodes. This defines what we will refer to here as the *dummy-arc* problem. Note that this definition implies that AoA dags have one initial node and one terminal node. Sysło [14] gives a good overview of the problem and provides a simple counter-example that shows we cannot minimize both the number of arcs and the number of nodes simultaneously. The problem of minimizing only the number of nodes can be solved in polynomial time using the algorithm of Cantor and Dimsdale [1], or the algorithm of Sterboul and Wertheimer [12].

The dummy-arc problem was shown by Krishnamoorthy and Deo [7] to be NP-hard. Several heuristics have been proposed, some of which construct an AoA dag directly, while others construct a dual graph. These include algorithms proposed by Corneil *et al.* [3], Dimsdale [4], Fisher *et al.* [5], Hayes [6], Spinrad [13], and Sysło [14, 15, 16]. Mrozek [10] gives an algorithm to verify if heuristically produced solutions are optimal. Only Corneil *et al.* claimed to have an optimal algorithm, but this was disproved by Mrozek [11]. Some of the heuristics will solve the problem for very restricted classes of precedence relations.

We solve the dummy-arc problem by showing how an instance of the dummy-arc problem may be reduced (in polynomial time) to an instance of the well-known set-cover problem. This allows us to solve the dummy-arc problem, either heuristically or optimally, using established set-cover algorithms and heuristics. Such algorithms are reviewed by Christofides and Korman [2], while heuristics are reviewed by Vasko and Wilson [17]. Mrozek [11] gives nearly the same reduction to the set-cover problem. However, our reduction is much simpler in presentation, it leads to a more concise instance of the set-cover problem, and we show that it subsumes efficient algorithms for several previously studied and some new special cases.

After developing some simple notation at the end of this section, we present a simple construction of the minimum set of nodes for AoA dags in section 2, and develop the reduction in section 3. Section 4 describes special cases for which the dummy-arc problem can be solved in polynomial time.

We let G refer to the AoN dag of a given precedence relation. We assume that G is transitively reduced, *i.e.*, $(u,v) \in G$ implies there exists no activity (node) w such that $(u,w) \in G$ and $(w,v) \in G$. The transitive closure of G is denoted by $tc(G)$, where $(u,v) \in tc(G)$ if there is a (possibly empty) path in G from node u to node v. Let $P(v)$ and $S(v)$ denote the sets of immediate predecessors and successors of activity v:

$$P(v) = \{u \mid (u, v) \in G\},$$
$$S(v) = \{w \mid (v, w) \in G\}.$$

Let $P^*(v)$ and $S^*(v)$ be the sets of all (not necessarily immediate) predecessors and successors of v:

$$P^*(v) = \{u \mid (u, v) \in tc(G)\},$$
$$S^*(v) = \{w \mid (v, w) \in tc(G)\}.$$

Note that $P^*(u) \subseteq P^*(v)$ iff $S^*(v) \subseteq S^*(u)$, but that this is not necessarily true for $P(v)$ and $S(v)$. We extend our terminology to say that a *constraint* $(u,v) \in G$ is *represented* in an AoA dag D if there is a path in D from t_u to s_v.

2. CONSTRUCTION OF THE MINIMUM SET OF NODES

The construction of the minimum set of nodes was first developed by Cantor and Dimsdale [1]. Their construction can be simplified as follows. Another version of this algorithm can be found in Sysło [16]. Recall that (s_v, t_v) is the arc of an AoA dag corresponding to activity $v \in G$. For each $v \in G$, we define two pairs of activity-sets, denoted $(P^*(s_v), S^*(s_v))$ and $(P^*(t_v), S^*(t_v))$, where

$$P^*(s_v) = P^*(v) \quad \text{and} \quad S^*(s_v) = \bigcap_{u \in P(v)} S^*(u),$$
$$S^*(t_v) = S^*(v) \quad \text{and} \quad P^*(t_v) = \bigcap_{w \in S(v)} P^*(w),$$

provided that the intersection over an empty set is equal to the set of all activities.

The minimum set of nodes is defined by the set of distinct pairs of activity-sets. There is then a single activity-set pair, denoted $(P^*(j), S^*(j))$, for each node j. The set $P^*(j)$ is the set of activities that precede node j, while $S^*(j)$ is the set of activities that follow node j. The construction of the minimum set of nodes is illustrated using the dag G shown in Figure 1. Table I lists the pairs of activity-sets for each of the ten activities, while Table II lists the nine distinct pairs defining the minimum set of nodes. By construction we have $s_v = i$ if $P^*(i) = P^*(s_v)$, and $t_v = j$ if $S^*(j)=S^*(t_v)$. The (AoA) *framework*, depicted by the solid arcs in Figure 2, is the set of activity arcs on the minimum set of nodes, but does not include any dummy arcs. In general, the framework may have many initial nodes and many terminal nodes. However, it always has a single initial node $s_\emptyset$ with $P^*(s_\emptyset) = \emptyset$, and a single terminal node $t_\emptyset$ with $S^*(t_\emptyset) = \emptyset$. The nodes $s_\emptyset$ and $t_\emptyset$ represent the project initiation and the project termination in every AoA dag having the minimum number of nodes. In Figure 2, $s_\emptyset = 1$ and $t_\emptyset = 9$.

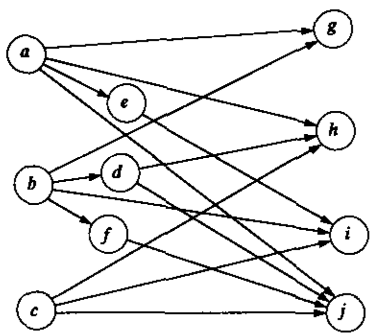

Figure 1

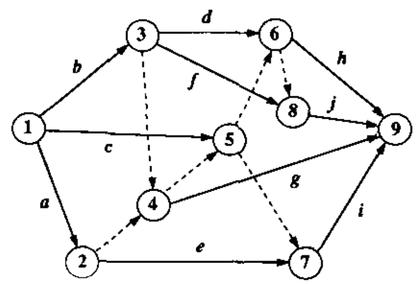

Figure 2

3. REDUCTION TO THE SET-COVER PROBLEM

A constraint $(u,v) \in G$ is not represented in the framework defined by G if $t_u \neq s_v$. In order to represent (u,v), a dummy path $\pi(t_u, s_v)$ (a path of dummy arcs which leads from t_u to s_v) must be added to the framework. Identifying all unrepresented constraints can be inefficient because two or more activities may share the same start and/or finish nodes. Therefore, we will instead identify the set R of all node pairs (i, j) of the framework that must be represented (*i.e.*, connected) by dummy paths,

$$R = \{(i, j) \mid i \neq j \text{ and } \exists (u, v) \in G \text{ such that } t_u = i \text{ and } s_v = j\}.$$

TABLE I

Activity-set-pairs

v	$P^*(s_v)$	$S^*(s_v)$	$P^*(t_v)$	$PS^*(t_v)$
a	–	a b c d e f g h i j	a	e g h i j
b	–	a b c d e f g h i j	b	d f g h i j
c	–	a b c d e f g h i j	a b c	h i j
d	b	d f g h i j	a b c d	h j
e	a	e g h i j	a b c e	i
f	b	d f g h i j	a b c d f	j
g	a b	g h i j	a b c d e f g h i j	–
h	a b c d	h j	a b c d e f g h i j	–
i	a b c e	i	a b c d e f g h i j	–
j	a b c d f	j	a b c d e f g h i j	–

TABLE II

Minimum Set of Nodes

i	$P^*(i)$	$S^*(i)$	Labels
1	–	a b c d e f g h i j	$s_a\ s_b\ s_c$
2	a	e g h i j	$t_a\ s_e$
3	b	d f g h i j	$t_b\ s_d\ s_f$
4	a b	g h i j	s_g
5	a b c	h i j	t_c
6	a b c d	h j	$t_d\ s_h$
7	a b c d e	i	$t_e\ s_i$
8	a b c d f	j	$t_f\ s_j$
9	a b c d e f g h i j	–	$t_g\ t_h\ t_i\ t_j$

A set of dummy arcs D is a (feasible) *solution* to the dummy-arc problem if D represents all pairs in R, *i.e.*, if there exists a dummy path $\pi(i, j)$ in D for every $(i, j) \in R$. An optimal solution is a solution D for which $|D|$ is minimum. A pair (i, j) is *feasible* if $P^*(i) \subset P^*(j)$ (equivalently $S^*(j) \subset S^*(i)$). In other words, a pair is feasible if it can be added as a dummy arc without introducing any constraints not consistent with $\prec$. This definition of feasible allows redundant pairs (i, j) for which there is a path in the framework from i to j, but it is evident from subsequent definitions that such pairs are not used in our reduction.

For each pair $(i, j) \in R$ we define

$$X(i, j) = \{k \mid (k, j) \in R \text{ and } (i, k) \text{ is feasible}\},$$
$$Y(i, j) = \{l \mid (i, l) \in R \text{ and } (l, j) \text{ is feasible}\}.$$