ELSEVIER

European Journal of Operational Research 136 (2002) 541–550

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/dsw

Production, Manufacturing and Logistics

The computational complexity of the criticality problems in a network with interval activity times

Stefan Chanas [a,*], Paweł Zieliński [b]

[a] Institute of Industrial Engineering and Management, Technical University of Wrocław, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland

[b] Institute of Mathematics, Technical University of Opole, Luboszycka 3, 45-036 Opole, Poland

Received 3 April 2000; accepted 12 January 2001

Abstract

The paper analyzes the criticality in a network with interval activities duration times. A natural generalization of the criticality notion (for a path, an activity and an event) for the case of network with interval activity duration times is given. The computation complexity of five problems linked to the introduced criticality notion is presented. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Project management and scheduling; Complexity theory; Interval criticality; Interval CPM

1. Introduction

The critical path method (CPM), worked out at the beginning of the 60s (see [13]), has become one of the tools that is most useful in practice and is applied to the planning and control of complex projects.

The key notion used in this method is that of criticality, what is confirmed by the very name of the method. This notion concerns a path, an activity or an event. We can talk about the criticality of an event only when the network model "activity on arc" is used in the method, i.e., a network in which the activities are represented by arcs and the nodes stand for certain stages, moments (called events) of the project. In the network model "activity on node", in which the activities are represented by network vertices, the notion of event criticality does not occur. In this paper, we make use of the "activity on arc" model. The critical path is the longest path leading from the initial vertex, representing the project start, to the final event, determining the project end. In one network there may be several critical paths, not only one. The critical path length determines the shortest possible duration time of the whole project. If this duration is taken as the target project duration time, then all the activities belonging to the critical

* Corresponding author. Tel.: +48-22-9819; fax: +48-71-320-3432.

E-mail addresses: chanas@ioz.pwr.wroc.pl (S. Chanas), pziel@polo.po.opole.pl (P. Zieliński).

PII: S 0 3 7 7 - 2 2 1 7 (0 1) 0 0 0 4 8 - 0

paths are characterized by a zero time slack. This means that each of them should start and finish in exactly (unequivocally) fixed moments of time, because otherwise the project dead line would be violated. For this reason, those activities are called critical activities. Similarly, the nodes belonging to critical paths correspond to critical events, that have to take place in strictly fixed moments of time (and not in time intervals), if we want to keep the project dead line. Information about all the critical paths, activities and events is very useful to the decision maker, because it allows him to concentrate, while controlling the project, on its most important elements, crucial to the successful project completion within the fixed deadline. All the critical paths, activities and events can be determined effectively, i.e., in a time being a polynomial function of the network size, by the CPM method.

What is essential in the CPM method is the assumption that the activities duration times are deterministic and known. Of course in practice, this assumption cannot always be fulfilled. That is why in the literature several methods have appeared that are destined for the analysis of projects with nondeterministic activities duration times, represented in the network with random variables or fuzzy numbers (see e.g., [2,3,5,8,10,11,14, 16–18]).

In both cases, the problem arises already when it comes to defining the notion of criticality. Of course, the whole problem can be substantially simplified. This is what is done in the classical PERT method (see [15]), where those paths, activities and events are considered to be critical that are such for the mean activities duration times. It is also possible to evaluate the probability (or the degree, in case of fuzzy duration times) with which the path (activity, event) is critical. But then, the problem becomes very complicated from the computational point of view (see e.g. [12]). The simplest way of representing uncertainty with respect to an activity duration time is an interval. Assigning some time interval T to an activity duration means that the actual duration of this activity will take a certain value within T, but it is not known which one.

In this paper, we occupy ourselves with the criticality analysis in a network with interval activities duration times. We will give a natural generalization of the criticality notion (for a path, an activity and an event) for the case of a network with interval activities duration times, and then we will analyze the computational complexity of a number of problems linked to the introduced criticality notion. Three among five analyzed problems will turn out to be difficult ones, although representing activities duration times with intervals seems to be the simplest method of modeling non-deterministic activities times.

Before we pass on to the essential considerations, let us recall the notions of the criticality of a path, an activity and an event in a network with deterministic activities times.

2. The criticality in a network with deterministic activity times

A directed, connected, acyclic graph (network) $G(A, V)$ being a project arc model is given. V ($|V| = n$) is a set of nodes (events are represented by the nodes of the graph) and $A \subset V \times V$ is a set of arcs (activities are represented by the arcs of the graph). A deterministic duration time t_{ij} is associated with each activity $(i, j) \in A$. Two nodes are specified in the graph G. "1" being the start node (the start event of the project) and "n" being the end node (the end event of the project).

Let us denote by $P(n)$ the set of all paths in G from the node "1" to the node "n".

Definition 1. A path $p \in P(n)$ is critical if and only if it is the longest path in the graph G (assuming that weights of the arcs are activities duration times).

The length of the path p is the minimum time required for completion of the whole project.

Definition 2. An activity $(i, j) \in A$ (an event $i \in V$) is critical if and only if it belongs to any critical path $p \in P(n)$.

The definitions presented above are equivalent to the definitions known from the literature where the criticality is defined by means of slacks of activities and events. But Definitions 1 and 2 are more suitable for considerations in this paper.

3. The CPM method with interval activity times

A graph $G_I(V,A)$ is given. All elements of this graph are the same as in the deterministic case except for activities duration times which are given by means of interval numbers, i.e., the interval $T_{ij} = [\underline{t}_{ij}, \overline{t}_{ij}]$ containing possible duration times of (i,j) is associated with each activity $(i,j) \in A$.

Now we define notions of criticality.

Definition 3. A path $p \in P(n)$ is *i*-critical (interval critical) in G_I if and only if there exists a set of times $t_{ij}, t_{ij} \in [\underline{t}_{ij}, \overline{t}_{ij}], (i,j) \in A$, such that p is critical in the sense of Definition 1, after replacing the interval times T_{ij} with the exact values $t_{ij}, (i,j) \in A$.

Definition 4. An activity $(i,j) \in A$ (an event $i \in V$) is *i*-critical (interval critical) if and only if it belongs to any *i*-critical path $p \in P(n)$.

The key lemma for the further considerations is Lemma 1 which determines a necessary and sufficient condition of *i*-criticality of a given path $p \in P(n)$.

Lemma 1. *A path $p \in P(n)$ is i-critical in G_I if and only if it is critical in the sense of Definition 1 in G_I, in which the interval activity times $T_{ij} = [\underline{t}_{ij}, \overline{t}_{ij}]$, $(i,j) \in A$, have been replaced with the exact values t_{ij} determined by means of the following formula*:

$$t_{ij} = \begin{cases} \overline{t}_{ij} & \text{if } (i,j) \in p, \\ \underline{t}_{ij} & \text{if } (i,j) \notin p. \end{cases} \tag{1}$$

Proof. The *if* direction: Assume that $p \in P(n)$ is *i*-critical. From Definition 3 we obtain that there exists a set of times $t_{ij}, t_{ij} \in [\underline{t}_{ij}, \overline{t}_{ij}], (i,j) \in A$, such that p is critical in the sense of Definition 1 (i.e., p is the longest path in G_I), after replacing the interval times T_{ij} with the exact values t_{ij}, $(i,j) \in A$. If we increase the activity times t_{ij} to $\overline{t}_{ij}$ of all the activities belonging to p and decrease the activity times t_{ij} to $\underline{t}_{ij}$ of all the activities, which do not belong to p, the path p will remain the longest path in G_I for this new configuration of the activity times.

The *only if* direction: Since there exists a set of activity times t_{ij}, $t_{ij} \in [\underline{t}_{ij}, \overline{t}_{ij}]$, for which p is critical (the set determined by formula (1)), from Definition 3 it follows directly that p is *i*-critical. □

4. Definitions of problems

Now we define five problems, concerning the CPM method with interval activity times, which are the subject of the paper.

1. The problem of determining an *i*-critical path (PDCP)
   Input: A directed, connected, acyclic graph $G_I(A,V)$, where V is a set of nodes (events) and $A \subset V \times V$ is a set of arcs (activities), weights of the arcs (activities duration times) $(i,j) \in A$ are determined by means of interval numbers $T_{ij} = [\underline{t}_{ij}, \overline{t}_{ij}]$, where $\underline{t}_{ij}$ and $\overline{t}_{ij}$ are nonnegative integer numbers (if the ends of the intervals are rational numbers, then one may scale them to obtain the integer ends), two specified nodes "1" being the start node and "n" being the end node.
   Output: An *i*-critical path $p \in P(n)$ in G_I.
2. The *i*-critical path problem (CPP)
   Input: The same as in PDCP and a path $p \in P(n)$.
   Question: Is $p \in P(n)$ *i*-critical in G_I?
3. The *i*-critical activity problem (CAP).
   Input: The same as in PDCP and a specified activity $(i,j) \in A$.
   Question: Is $(i,j) \in A$ *i*-critical in G_I?
4. The *i*-critical event problem (CEP).
   Input: The same as in PDCP and a specified event $i \in V$.
   Question: Is $i \in V$ *i*-critical in G_I?
5. The K *i*-critical paths problem (KCPP)
   Input: The same as in PDCP and a positive integer number $K \leqslant 2^{|V|-2}$.
   Output: All *i*-critical paths in G_I but not more than K.

5. The analysis of computational complexity

In this section, we analyze the computational complexity of the problems which have been formulated in Section 4.

5.1. Complexity of PDCP and CPP

The problem of determining an arbitrary *i*-critical path (PDCP) and the problem of asserting the *i*-criticality of a fixed path $p \in P(n)$ (CPP) are easy ones, i.e., they can be solved in the time bounded by a polynomial in the size of $G_{\mathrm{I}}(A, V)$. In the first case, it is enough to apply the classical algorithm for finding the longest path in an acyclic, directed graph to G_{I} with deterministic weights of the arcs, t_{ij}, which are any values chosen from the respective intervals, i.e., $t_{ij} \in [\underline{t}_{ij}, \overline{t}_{ij}]$, $(i,i) \in A$. And in the second one it is sufficient to apply the classical algorithm for finding the longest path in an acyclic, directed graph to G_{I}, after replacing the interval weights of the arcs $T_{ij} = [\underline{t}_{ij}, \overline{t}_{ij}], (i,j) \in A$, with the exact values t_{ij} determined as in Lemma 1. If the length of a path $p' \in P(n)$, determined in this way, is equal to the length of given path $p \in P(n)$, then from Lemma 1 it follows that p is *i*-critical otherwise p is not *i*-critical.

5.2. Complexity of CAP and CEP

Now we prove that CAP is *strongly NP*-complete (the *strong NP*-completeness has been defined in [6]). First, we give the proof that there exists a pseudo-polynomial transformation from a known *strongly NP*-complete decision problem to CAP. As a known *strongly NP*-complete problem, we have chosen the CNF-satisfiability problem (CNF-SAT). CNF-SAT is defined as follows (see e.g.[7]).

Input: A set of Boolean variables $X = \{x_1, \dots, x_p\}$, a set of clauses $C = \{C_1, \dots, C_q\}$ composed of the variables and their negations, a Boolean expression F in conjunctive normal form, $F = C_1 \wedge \cdots \wedge C_q$, where $C_j = l_1^j \vee \cdots \vee l_{n_j}^j$, and l_i^j, $i = 1, \dots, n_j$, is a literal, i.e., l_i^j is either x_r or $\sim x_r, r = 1, \dots, p$.

Question: Is there an assignment "True"(1) and "False" (0) to the variables $x_1, \dots, x_p$ such that the Boolean expression F is satisfiable (F has value 1)?

The transformation from an instance of CNF-SAT to an instance of CAP consist of two stages. In the first stage we construct, on the basis of an expression F, an expression $F' = C_1' \wedge \cdots \wedge C_{q'}'$ by means of Algorithm 1. The idea of constructing F' is similar to transformation from CNF-SAT to three-CNF-SAT (see e.g., [1, p. 384]).

Algorithm 1 (*Construction of* F').

$F' \leftarrow F$
$q' \leftarrow q + 1$
Construct new clause $C_{q'}' \leftarrow z$ $\{z$ is a new variable$\}$
$F' \leftarrow F' \wedge C_{q'}'$
$j \leftarrow 0$
for $r \leftarrow 1$ to $q - 1$ **do**
  for $i \leftarrow 1$ to n_r **do**
    if literal $l_i'^r$ does not contain a new variable **then**
      for $s \leftarrow r + 1$ to q **do**
        for $k \leftarrow 1$ to n_s **do**
          if $l_i'^r$ is negation of $l_k'^s$ **then**
            $j \leftarrow j + 1$
            $q' \leftarrow q' + 1$
            Construct new clause $C_{q'}' \leftarrow l_k'^s \vee \sim y_j$ $\{ y_j$ is a new variable$\}$
            $F' \leftarrow F' \wedge C_{q'}'$
            In clause C_s' replace literal $l_k'^s$ with literal y_j
          end if
        end for
      end for
    end if
  end for
end for

The construction of the expression F' consists in systematic elimination of the contradictory literals from clauses $C_1' \wedge \cdots \wedge C_q'$.

For example, F is given as follows

$$F = (x_1 \vee \sim x_2 \vee \sim x_3) \wedge (\sim x_1 \vee \sim x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \sim x_3),$$

then the expression F', according to the presented construction, has the following form:

$$F' = (x_1 \vee \sim x_2 \vee \sim x_3) \wedge (y_1 \vee \sim x_2 \vee y_3) \wedge (x_1 \vee y_2 \vee \sim x_3) \wedge z \wedge (\sim y_1 \vee \sim x_1) \wedge (\sim y_2 \vee x_2) \wedge (\sim y_3 \vee x_3), \quad (2)$$

where y_1, y_2, y_3, z are new variables.

The expression F' has additional significant properties, which we formulate in the form of two statements.

Statement 1. An expression $F = C_1 \wedge \cdots \wedge C_q$ is satisfiable if and only if the expression $F' = C'_1 \wedge \cdots \wedge C'_{q'}$ is satisfiable.

Statement 2. In the clauses $C'_j, j = 1, \dots, q$, of the expression F' there are no literals l'^r_i, l'^s_k belonging to the different clauses, $r, s = 1, \dots, q (r \neq s)$, such that l'^r_i is negation of l'^s_k. The same property holds for the clauses $C'_j, j = q + 2, \dots, q'$.

In the second stage, on the basis of the expression F' we build an instance of CAP.

Let us denote by $G'_{\mathrm{I}}(A', V')$ a graph (a network) for this instance. Each literal $l'^j_i, i = 1, \dots, n'_j$, of clause $C'_j, j = 1, \dots, q'$, corresponds, in G'_{I}, to an activity (an arc) (u^j_i, v^j_i) with a duration time equal to the [0,1] interval. We link by an activity (v^j_i, u^{j+1}_k) with duration time [1,1] the end-point v^j_i of each activity corresponding to literal of clause C'_j with the start-point u^{j+1}_k of each activity corresponding to literal of the next $j + 1$st clause, $j = 1, \dots, q' - 1$. Then, we link by an activity (v^r_i, u^s_k) with duration time $[2(s - r), 2(s - r)]$ events v^r_i and $u^s_k, r < s$, if literals l'^r_i and l'^s_k corresponding to them are contradictory (l'^r_i is negation of l'^s_k). Finally, we add the start event ("1") and link it by activities with duration times [1,1] with events $u^1_i, i = 1, \dots, n'_1$, then we add the end event ("n") and link events $v^{q'}_i, i = 1, \dots, n'_{q'}$, with it by activities with duration times [1,1].

This completes the definition of $G'_{\mathrm{I}}(V', A')$. Note that G' is connected, acyclic and directed. The rest of the input to CAP is defined as follows. "1" and "n" are specified nodes and (u^{q+1}_1, v^{q+1}_1) corresponding to the only one literal of clause C'_{q+1} is a specified activity.

The construction of F' as well as of G'_{I} can be done in the time bounded by polynomial in the size of CNF-SAT.

The graph G'_{I} built on the basis of expression (2) is shown in Fig. 1. The activities duration times being one pointed intervals have been written as deterministic times – in the form of single numbers.

From the properties of F' and the way of constructing G'_{I}, the following statements result.

Statement 3. A path $p \in P(n)$ in the graph G'_{I} contains the specified activity (u^{q+1}_1, v^{q+1}_1) if and only if it does not contain an activity linking events corresponding to contradictory literals.

Statement 4. A path $p \in P(n)$ in the graph G'_{I} contains the specified activity (u^{q+1}_1, v^{q+1}_1) if and only if it fulfills condition $l = 2q' + 1$, where $l = \sum_{(u,v) \in p} \bar{t}_{uv}$.

Statement 5. A path $p \in P(n)$ in the graph G'_{I} does not contain the specified activity if and only if it fulfills condition $l = 2q' + 2$, where $l = \sum_{(u,v) \in p} \bar{t}_{uv}$.

The main property of presented transformation is included in the following lemma.

Lemma 2. *An expression F is satisfiable if and only if there exists an i-critical path $p \in P(n)$ in G'_{I} containing the specified activity (u^{q+1}_1, v^{q+1}_1).*

Proof. The *if* direction: Suppose that there exists an 0–1 assignment to the variables $x_1, \dots, x_p$ such that F is satisfiable. We determine a path $p \in P(n)$ containing the specified activity (u^{q+1}_1, v^{q+1}_1) and show that it is *i*-critical.

From Statement 1 it follows that F' is satisfiable. Therefore, each clause $C'_j, j = 1, \dots, q'$, contains at least one literal l'^j_i with assigned value 1. From each of clauses we choose one such literal and complete a set of activities corresponding to these literals by activities linking them to

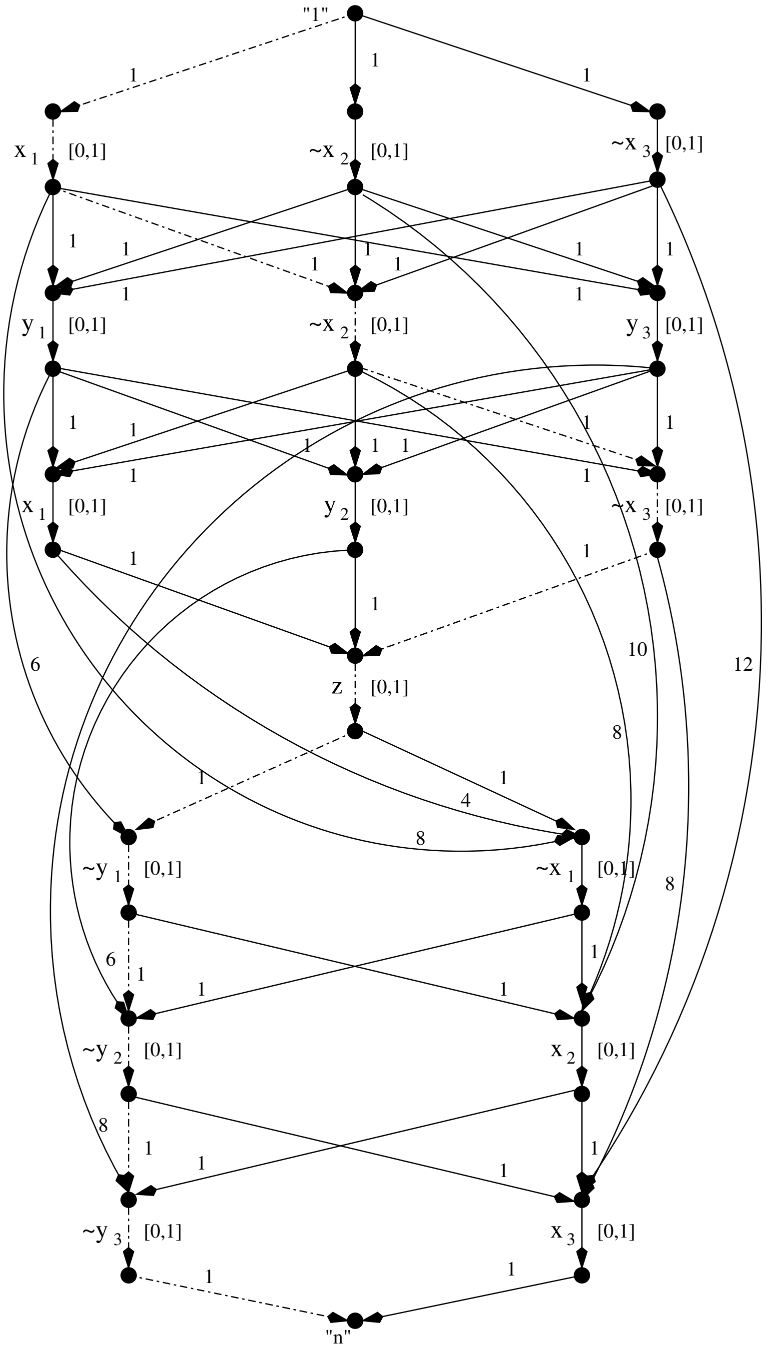

Fig. 1. The graph G'_I obtained on the basis of expression (2).

obtain a path $p \in P(n)$. This construction is unique. We will prove that this path is indeed *i*-critical.

Let us replace the interval activities duration times with the exact values t_{ij} determined as in Lemma 1. From Statement 4 it results that the length $l = \sum_{(u,v)\in p} t_{uv}$ of the path is equal to $2q' + 1$.

If $q' = q + 1$, then paths belonging to the $P(n)$ in G'_I contain the specified activity. From Statement 4, we get that their lengths are at most $2q' + 1$. Hence, p is critical. Lemma 1 implies that p is *i*-critical.

Let $q' > q + 1$. Assume to the contrary that there exists a path $p' \in P(n)$, $p' \neq p$, with length $l', l < l'$. From Statements 3 and 5 we obtain that p' contains an activity which links events corresponding to contradictory literals. The events linked by this activity, belonging to the path p', cannot belong to p simultaneously. Let us denote these events by $v_i^r, u_k^s \in p'$. Statement 2 implies that they fulfill the condition $1 \leqslant r \leqslant q, q+2 \leqslant s \leqslant q'$. Only one of the following three cases may be fulfilled.

1. $v_i^r \notin p, u_k^s \notin p$,
2. $v_i^r \in p, u_k^s \notin p$,
3. $v_i^r \notin p, u_k^s \in p$.

From Statement 5, we conclude that $2q' + 2$ is the maximum length of p' and it can be reached only if duration times of activities belonging to the path p' are equal to the right ends of intervals (value 1). For each above case it is easy to show that p' with activities duration times determined as in Lemma 1 contains at least one activity with duration time which is equal to the left end of interval (value 0). Hence, the length $l' = \sum_{(u,v)\in p'} t_{uv}$ of p' fulfills the condition $l' \leqslant 2q' + 1$ which contradicts that $l < l'$. Thus, the path p is critical with activities duration times determined as in Lemma 1. By this lemma p is *i*-critical.

The *only if* direction: Let $p \in P(n)$ be any *i*-critical path in G'_I containing the specified activity (u_1^{q+1}, v_1^{q+1}). Each clause $C'_j, j = 1, \ldots, q'$ is represented in p by one activity which corresponds to one of literals of this clause.

We claim that p does not contain an activity corresponding to contradictory literals. Let us replace the interval activities duration times with the exact values t_{ij} determined as in Lemma 1. From *i*-criticality of p it implies that p is the longest (critical) in G'_I. By Statement 4 p is of length $2q' + 1$. Assume to the contrary that there exist activities $(u_i^r, v_i^r), (u_k^s, v_k^s) \in p$ corresponding to contradictory literals $l_i'^r, l_k'^s$, where $1 \leqslant r \leqslant q, q+2 \leqslant s \leqslant q'$ (see Statement 2). But then there exists a path $p' \in P(n)$ which does not use the specified activity. It is composed of the parts of p: from the node "1" to the node v_i^r, from u_k^s to "n" and activity $(v_i^r, u_k^s) \notin p$ which links events v_i^r, u_k^s corresponding to contradictory literals. From Statement 5 it follows that p' is of length $2q' + 2$ which contradicts criticality of p.

Now we may assign the 1's to all literals $l_i'^j$, corresponding to activities $(u_i^j, v_i^j) \in p, j = 1, \ldots, q'$, without risking assigning simultaneously the 1 to a literal as well as to its negation. Hence, each clause C'_j, $j = 1, \ldots, q'$, of F' is satisfiable. From Statement 1 it follows that F is also satisfiable. □

For the expression (2) the following 0–1 assignment to the variables is satisfiable: $x_1 = 1, x_2 = 0, x_3 = 0, y_1 = 0, y_2 = 0, y_3 = 0, z = 1$. The *i*-critical path in G'_I corresponding to this assignment has been marked in Fig. 1 by the dotted line.

Theorem 1. *CAP is strongly NP-complete.*

Proof. We show that CAP is in *NP*. A Nondeterministic Turing Machine (NDTM) first "guesses" sequence of nodes p being an *i*-critical path in G_I which uses the specified activity. Then NDTM verifies p i.e., it checks:

1. if for all consecutive nodes $i, j \in p$ it holds $(i, j) \in A$,
2. if p contains the specified activity,
3. if $p \in P(n)$,
4. if p is critical in G_I after replacing the interval weights of arcs, with the exact values determined as in Lemma 1.

The verification can be done in the time bounded by a polynomial in the size of G_I, so CAP belongs to *NP*.

Now we prove that there exists a pseudo-polynomial transformation (see definition in [6]) from an instance of CNF-SAT to an instance of CAP.

Let us consider the transformation from the instance of CNF-SAT to the instance of CAP, presented in this section. Since this transformation can be done in the time bounded by a polynomial in the size of CNF-SAT, the first condition of pseudo-polynomial transformation is fulfilled. The second one, i.e., the answer for the instance CNF-SAT is "yes" if and only if the answer for the corresponding instance CAP is also "yes", follows from Lemma 2. The rest of the conditions of pseudo-polynomial transformation are also satisfied. Thus, CAP is *strongly NP*-complete. □

Theorem 2. *CEP is strongly NP-complete.*

Proof. Analogous to the proof of Theorem 1. It is enough to set one of the events u_1^{q+1} or v_1^{q+1} to be the specified event in G'_I. □

5.3. Complexity of KCPP

Now we occupy ourselves with *K*CPP and prove that it is *NP*-hard. Let us define the decision *K i*-critical paths problem (D-*K*CPP).

Input: The same as in *K*CPP.

Question: Are there K different *i*-critical paths in G_I?

And let us additionally define the extended decision *K i*-critical paths problem (ED-*K*CPP).

Input: The same as in *K*CPP and a natural number l.

Question: Are there K different *i*-critical paths in G_I fulfilling the following condition

$$\sum_{(u,v)\in p_i} \bar{t}_{uv} \geqslant l \qquad (3)$$

for $i = 1, \ldots, K$.

Lemma 3. *There exists polynomial transformation from ED-KCPP to D-KCPP (ED-KCPP∝D-KCPP).*

Proof. A polynomial transformation of an instance of ED-*K*CPP to an instance of D-*K*CPP consists in adding to G_I an activity (arc) with duration time $[l', l']$ which links the start event ("1") and the end event ("n"). l' is defined as follows

$$l' = \begin{cases} \underline{l} & \text{if } l < \underline{l}, \\ l & \text{if } l \geqslant \underline{l}, \end{cases}$$

where $\underline{l}$ is length of the longest path in G_I, after replacing the interval activities duration times $T_{ij}, (i,j) \in A$, with the exact values $\underline{t}_{ij}$. It is easy to show that there are K different *i*-critical paths in G_I fulfilling (3) if and only if there are $K+1$ such paths in G''_I. Note that the set of *i*-critical paths in G''_I is the set of *i*-critical paths in G_I extended by additional path $(1, n)$. □

Theorem 3. *D-KCPP is NP-hard.*

Proof. *NP*-hardness of D-*K*CPP is result of using polynomial Turing transformation from CNF-SAT to ED-*K*CPP, CNF-SAT$\propto_T$ED-KCPP (see definition in [7,9]). It is enough to give an algorithm ***A*** solving CNF-SAT by using a procedure ***P*** where ***P*** solves ED-*K*CPP. The algorithm ***A*** would be polynomial if procedure ***P*** was polynomial.

Suppose that $\boldsymbol{P}(G_\mathrm{I}(A, V), T, K, l)$ is a procedure solving ED-*K*CPP for the following parameters: $G_\mathrm{I}(A, V)$ – an acyclic directed graph, $T = \{T_{i,j}\}$, $(i,j) \in A$, – interval activities duration times, $0 < K \leqslant 2^{|V|-2}$ and $l > 0$ – constants. Algorithm 2 corresponds to the algorithm ***A*** which solves CNF-SAT by using the procedure ***P***.

After calling $\boldsymbol{P}(G'_\mathrm{I}(A', V'), T', L^*, 2q' + 2)$ in ***A*** we obtain the answer "yes" or "no" If it is "yes", then all *i*-critical paths in G'_I satisfy inequality (3) in the strong sense. Statement 5 implies that none of them contain the specified activity (u_1^{q+1}, v_1^{q+1}). Making use of Lemma 2 we get the answer "no" for CNF-SAT. Otherwise, there exits an *i*-critical path in G'_I such that $l = 2q' + 1$. Statement 4 implies that it contains the specified activity (u_1^{q+1}, v_1^{q+1}). Again from Lemma 2 we obtain the answer "yes" for CNF-SAT.

Algorithm 2 (*Algorithm **A** – solving CNF-SAT by means of procedure **P***).

```
Build an instance of CAP, G'_I(V', A'), T',
  (u_1^{q+1}, v_1^{q+1}), on the basis of an instance of
  CNF-SAT according to the presented in Sec-
  tion 5.2 construction
L_min ← 0
L_max ← 2^{|V'|-2}
while L_max − L_min ≠ 1 do
  L ← (L_max + L_min)/2
```

$\boldsymbol{P}(G'_{\mathrm{I}}(A',V'),T',L,2q'+1)$
if Answer after calling $\boldsymbol{P}$ is "yes" **then**
  $L_{\min} \leftarrow L$
else
  $L_{\max} \leftarrow L$
end if
end while
$L^* \leftarrow L_{\min}$ { L^* is the maximum number of i-critical paths, in G'_{I}, fulfilling inequality (3) for $l = 2q'+1$}
$\boldsymbol{P}(G'_{\mathrm{I}}(A',V'),T',L^*,2q'+2)$
if Answer after calling $\boldsymbol{P}$ is "yes" **then**
  write("no") {the answer for CNF-SAT is "no"}
else
  write("yes") {the answer for CNF-SAT is "yes"}
end if

Since the construction of instance of CAP can be done in polynomial time and the number of calls of $\boldsymbol{P}$ is equal to $|V'|-1$, the presented algorithm $\boldsymbol{A}$ would be polynomial for CNF-SAT if the procedure $\boldsymbol{P}$ was polynomial for ED-*K*CPP.

We have proved that CNF-SAT$\propto_T$ED-*K*CPP. Lemma 3 implies that ED-*K*CPP$\propto$D-*K*CPP. A polynomial transformation is a special case of a polynomial Turing transformation so ED-*K*CPP$\propto_T$D-*K*CPP. Making use of transitivity of a polynomial Turing transformation we get CNF-SAT$\propto_T$D-*K*CPP. This completes the proof. □

Corollary 1. *KCPP is NP-hard.*

Proof. It follows from existence of polynomial Turing transformation D-*K*CPP$\propto_T$KCPP. □

6. Conclusions

In the paper, we have given a natural generalization of the notion of criticality for the case of a network with interval duration times of the activities, by introducing the notion of *i*-critical path, activity and event. We have shown that both the problem of determining an arbitrary *i*-critical path and that of the estimation of the *i*-criticality of a fixed path are easy problems. Solving them reduces itself to applying the classical CPM method to a network with deterministic duration times of the activities. Unfortunately, the problem of determining K *i*-critical paths, as well as that of estimating the *i*-criticality of a fixed activity and a fixed event have turned out to be hard ones, as shown in this paper. The results obtained are important per se. They can be useful in situations where the interval as the set of all possible duration times of an activity can be considered a suitable (sufficient) model of the activity duration time. However, we would like to emphasize that we have made a substantial use of the results presented here in the criticality analysis of a network where the duration times of the activities are represented by interval fuzzy numbers (see [4]). It was that problem that inspired us to examine criticality in a network with interval activities duration times, as an interval fuzzy number can be unambiguously represented by a set of intervals (the set of its λ-cuts). There exist close links between the notion of *f*-criticality (fuzzy criticality) introduced in [4], in a network with fuzzy activities duration times, and that of *i*-criticality in a network with interval activities duration times analyzed here.

Acknowledgements

This work was supported by grant no. 7T11F02120 from the State Committee for Scientific Research (Komitet Badań Naukowych).

References

[1] A.V. Aho, J.E. Hopcroft, J.D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, MA, 1974.
[2] J.J. Buckley, Fuzzy PERT, in: G.W. Evans, W. Karwowski, M.R. Wilhelm (Eds.), Applications of Fuzzy Set Methodologies in Industrial Engineering, Elsevier, Amsterdam 1989, pp. 103–114.
[3] S. Chanas, J. Kamburowski, The use of fuzzy variables in PERT, Fuzzy Sets and Systems 5 (1981) 1–19.
[4] S. Chanas, P. Zieliński, Critical path analysis in the network with fuzzy activity times, Fuzzy Sets and Systems 122 (2001) 195–204.

[5] S.E. Elmaghraby, On criticality and sensitivity in activity networks, European Journal of Operational Research 127 (2000) 220–238.
[6] M.R. Garey, D.S. Johnson, Strong NP-completeness results: motivation, examples, and implications, Journal of the ACM 25 (4) (1978) 499–508.
[7] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freemam, San Francisco, 1979.
[8] I. Gazdik, Fuzzy network planning, IEEE Transactions on Reliability R-32 3 (1983) 304–313.
[9] D.S. Johnson, S.D. Kashdan, Lower bounds for selection in $X + Y$ and other multisets, Journal of the ACM 25 (4) (1978) 556–570.
[10] J. Kamburowski, An upper bound on the expected completion time of PERT networks, European Journal of Operational Research 21 (1985) 206–212.
[11] J. Kamburowski, Normally distributed activity durations in networks, Journal of Operational Research Society 36 (1985) 1051-1057; corrigendum 37 (1986) 529.
[12] J. Kamburowski, On the computational complexity of the shortest route and maximum flow problems in stochastic networks, Foundations of Control Engineering 11 (4) (1986) 167–175.
[13] J.E. Kelley, Critical path planning and scheduling – mathematical basis, Operations Research 9 (1961) 296–320.
[14] F.A. Loostma, Stochastic and fuzzy PERT, European Journal of Operational Research 43 (1989) 174–183.
[15] D.G. Malcolm, J.H. Roseboom, C.E. Clark, W. Fazar, Application of a technique for research and development project evaluation, Operations Research 7 (5) (1959) 646–669.
[16] S.H. Nasution, Fuzzy critical path method, IEEE Transactions on Systems Man and Cybernetics 24 (1) (1994) 48–57.
[17] H. Prade, Using fuzzy sets theory in a scheduling problem: A case study, Fuzzy Sets and Systems 2 (1979) 153–165.
[18] H. Rommelfanger, Network analysis and information flow in fuzzy environment, Fuzzy Sets and Systems 67 (1994) 119–128.