ISSN 1064–5624, Doklady Mathematics, 2009, Vol. 79, No. 1, pp. 1–3. © Pleiades Publishing, Ltd., 2009.
Original Russian Text © A.A. Lazarev, E.R. Gafarov, 2009, published in Doklady Akademii Nauk, 2009, Vol. 424, No. 1, pp. 7–9.

Transformation of the Network Graph of Scheduling Problems with Precedence Constraints to a Planar Graph

A. A. Lazarev and E. R. Gafarov

Presented by Academician S.N. Vasil'ev March 19, 2008

Received March 19, 2008

DOI: 10.1134/S1064562409010013

For problems on graphs with n vertices, an algorithm of complexity $O(n^5)$ is designed that transforms a nonplanar graph into a planar one in a graph). The sum of the number of vertices and edges in the resulting graph is at most as large as in the original graph.

Suppose that we are Given a set of jobs $N = \{1, 2, \ldots, n\}$ and K renewable resources $k = 1, 2, \ldots, K$, let Q_k units of resource k be available at every time t. The processing time $p_i \in \mathbb{Z}^+$ is specified for each job $i = 1, 2, \ldots, n$. While the job i is processed, $q_{ik} \le Q_k$ units of resource $k = 1, 2, \ldots, K$ are required. After the job is accomplished, all the available resources volume can be instantaneously allocated to other jobs.

Precedence constraints are specified between some pairs of jobs: $i \to j$ means that the job j is processed after the job i has been accomplished. The resources are available starting at the time $t = 0$. The processing of the jobs cannot be interrupted.

The goal is to determine the start times S_i $(i = 1, 2, \ldots, n)$ for processing the jobs that minimize the time required for accomplishing the entire project: $C_{\max} = \max\limits_{i=1,2,\ldots,n} \{C_i\}$, where $C_i = S_i + p_i$. The following constraints must be satisfied:

(i) At each time t, the total resource required does not exceed the resource availability for each resource type.

(ii) The given job precedence constraints are fulfilled.

(iii) The makespan $C_{\max} = \max\limits_{j=1}^{n} C_j$, where $C_j = S_j + p_j$ is the completion time of job j, is minimized.

This problem is called the resource constrained project scheduling problem (RCPSP). This problem can be reduced to the NP-hard multidimensional knapsack problem in polynomial time [2].

The structure of the project is represented by a directed acyclic graph $G = (V, E)$, where each vertex from $V = \{1, 2, \ldots, n\}$ corresponds to some job from $N = \{1, 2, \ldots, n\}$ and the arcs $E = \{(i, j) | \, i, j \in V; i \to j\}$ correspond to the precedence constraints. Obviously, a feasible solution exists only if the precedence graph is acyclic. Such a directed graph is called a network graph.

PLANARITY OF THE NETWORK GRAPH FOR RCPSP AND ITS SPECIAL CASES

Two instances of RCPSP are called similar if one instance is reduced to the other by adding "empty" jobs (with zero duration) and deleting "redundant" connections, so that, if there was at least one path between the vertices i and j, then a path between them still persists. If there was no path between the vertices, then no path appears. Obviously, the optimal values of the objective function for similar instances are equal to each other.

For several resource-constrained project scheduling problems with precedence relations, we have the following theorem.

Theorem 1. *For any instance of RCPSP with n jobs and v relations, there exists an analogous instance with a flat graph G' with n' jobs and v' relations, where $n + v \ge n' + v'$.*

We obtain an analogous instance from the original one by adding "dummy" jobs and deleting all the unnecessary relations. The proof of the theorem follows from Lemmas 1 and 2.

Lemma 1. *If there is a subgraph $G' \subset G$ that is isomorphic to the special graph $K_{3,3}$, then we can transform it into a flat subgraph by adding dummy jobs (with $p_j = 0$) and deleting all the unnecessary relations.*

Lemma 2. *If there is a subgraph $G' \subset G$ that is isomorphic to the special graph $K_{5,2}$, then we can transform it into a flat subgraph by deleting all the unnecessary relations.*

Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Profsoyuznaya ul. 65, Moscow, 117997 Russia; e-mail: jobmath@mail.ru

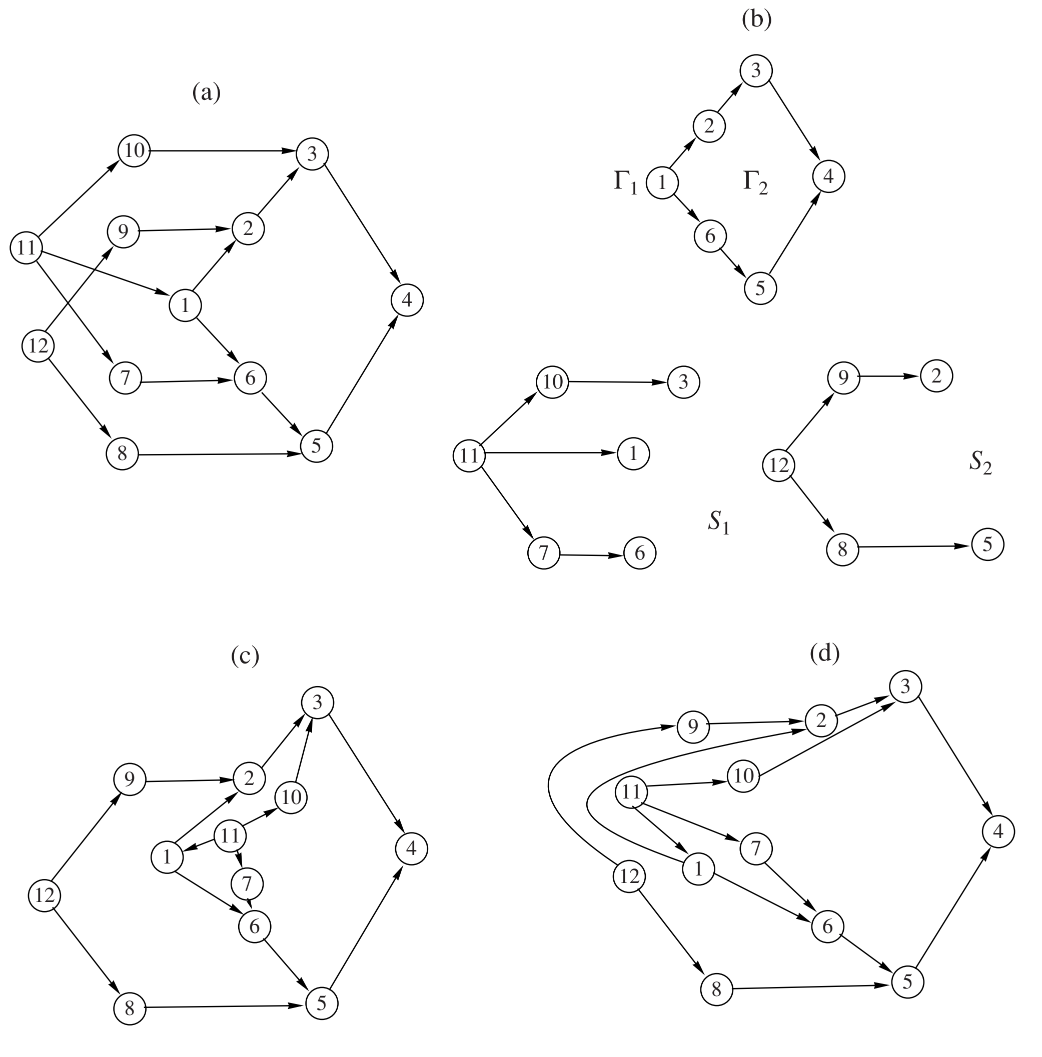

Fig. 1. Layout of the network graph in a plane.

ALGORITHM FOR FINDING A NETWORK GRAPH LAYOUT IN A PLANE

When elements are placed on a chip, an important condition is that the corresponding graph is planar (the connections between the elements do not intersect). Moreover, in practice a network graph has to be represented in a convenient form according to a prescribed design. In the most popular packages (MS Project, Spider), network graphs are represented inconveniently: arcs intersect or merge. In this section, we describe an algorithm for the layout of a planar network graph in a plane without arc intersection.

A well-known algorithm for the layout of a planar graph in a plane was presented in [1]. An important difference between a network graph and a directed graph is that the vertices of the former are arranged on some virtual time axis (usually horizontal). A predecessor job is always located to the left of its successors.

We briefly describe an algorithm for the layout of a planar graph in a plane and show how it can be transformed for the layout of a planar network graph.

The algorithm γ [1] for the layout of a graph G is a process in which some laid out subgraph G' of G is consecutively supplemented with a new chain such that both of its ends belong to G'. Thus, this chain divides a face of G' into two. Any simple cycle of G' is used as an initial planar graph G'. The process continues until a planar graph isomorphic to G is constructed or a chain fails to be added. In the latter case, G is not planar.

We introduce certain *definitions.* Suppose that a layout of the subgraph G' of G has been constructed. A *segment S with respect to* G' (or merely a segment) is a subgraph of G of one of the following two forms:

(i) an edge $e = uv \in EG$ such that $e \notin EG'$, and u, $v \in VG'$, where EG is the edge set of G, VG is the vertex set of G, EG' is the edge set of G', and VG' is the vertex set of G';

(ii) a connected component of G–G' supplemented by all the edges of G incident to the vertices of this component and to the ends of these edges.

A vertex v of S is called a *contact* vertex if $v \in VG'$.

An *admissible face* for S is a face Γ of G' that contains all the contact vertices of S.

$\Gamma(S)$ is the set of admissible faces for S. A simple chain of S that joins two different contact vertices and does not contain any other contact vertices is called an α-*chain.*

Two segments S_1 and S_2 with respect to G' are called *conflicting* if

(i) $\Gamma(S_1) \cap \Gamma(S_2) \neq \emptyset$.

(ii) There exist two α-chains $L_1 \in S_1$ and $L_2 \in S_2$ that cannot be simultaneously laid out without intersections in any face.

It was shown in [1] that any α-chain in any segment can be used in the algorithm γ and that chain can be placed in any admissible face.

Figure 1a displays a network graph with an intersection of arcs. Figure 1b shows the initial chain (the graph G') and its faces and segments S_1 and S_2. The segments are conflicting.

In the network graph layout algorithm, a face for the current α-chain is chosen according to the following rule: a predecessor is to the left of a successor. Accordingly, the face Γ_1 was chosen for the layout of the α-chain corresponding to S_2 and the face Γ_2 was chosen for the layout of S_1.

A layout of the corresponding graph is presented in Fig. 1c.

In this layout, successor vertex 1 (see the circled index) is to the left of predecessor vertex 11, which contradicts the design rules for the network graph. The resulting layout can be stretched, as shown in Fig. 1d. The edge (1, 2) then becomes arcwise.

A layout without intersections is frequently better to transform to admit intersections. To minimize the number of intersections, it is better to apply such transformations to a planar layout.

Thus, while scheduling a project, we can construct a planar network graph. Indeed, the dimension $n + e$ of the problem does not increase in this case. According to the Euler theorem, the number of edges in a planar graph is at most $3n - 6$, which can be taken into account in estimating the complexity of algorithms.

Any nonplanar graph G can be transformed into an analogous planar graph G' in $O(n^5)$ operations. Here, the sum of the numbers of vertices and edges does not increase and the path from vertex i to vertex j is preserved, with the only difference being that "empty" vertices are possibly added to the path. Moreover, a planar network graph can be conveniently represented in a plane, which is useful in practice.

ACKNOWLEDGMENTS

This work was supported by the program "Leading Scientific Schools" (project no. NSh-5833. 2006.1) and was partially supported by the Deutsche Akademische Austrauschdienst (DAAD, grant AI206237/Ref. 325).

REFERENCES

1. V. A. Emelichev, O. I. Mel'nikov, V. I. Sarvanov, and R. I. Tyshkevich, *Lectures on Graph Theory* (Nauka, Moscow, 1990) [in Russian].
2. P. Brucker and S. Knust, *Complex Scheduling* (Springer-Verlag, Berlin, 2006).