SRC TR 88-88

TECHNICAL RESEARCH REPORT

Exact Algorithms for Multilayer Topological Via Minimization

by

C. S. Rim, T. Kashiwabara and K. Nakajima

SYSTEMS RESEARCH CENTER

UNIVERSITY OF MARYLAND

COLLEGE PARK, MARYLAND 20742

Exact Algorithms for Multilayer Topological Via Minimization[1]

Chong S. Rim
Electrical Engineering Department and Systems Research Center
University of Maryland, College Park, MD 20742

Toshinobu Kashiwabara
Electrical Engineering Department and Institute for Advanced Computer Studies
University of Maryland, College Park, MD 20742
on leave from
Department of Information and Computer Sciences
Osaka University, Toyonaka, Osaka 560, Japan

and

Kazuo Nakajima
Electrical Engineering Department,
Institute for Advanced Computer Studies and Systems Research Center
University of Maryland, College Park, MD 20742

[1] This work was supported in part by National Science Foundation grants MIP-84-51510 and CDR-88-03012 (Engineering Research Centers Program), and a grant from AT&T.

Exact Algorithms for Multilayer Topological Via Minimization

by

Chong S. Rim, Toshinobu Kashiwabara *and* Kazuo Nakajima

Abstract

We consider the topological via minimization problem in which each of n nets has two terminals to be connected and the routing area consists of k layers. We first show that this problem is solvable in $O(kn^2)$ time, using a minimum cost network flow algorithm, for the usual channel routing case in which no local net exists. We then consider the problem for the case of "circular" channel routing in which no local net exists, and present an $O(n^{2k+1})$ time algorithm using dynamic programming.

1. Introduction

The design process of integrated circuits and printed circuit boards consists of three phases: partitioning, placement and routing. In the partitioning phase, a given circuit is decomposed into subcircuits of proper size called *modules*. Each module is realized separately and the terminals are placed on its periphery for connection to other modules. The realization of each subcircuit thus obtained is also called a module. In the placement phase, an appropriate position of each module is determined. In the routing phase, electrical connection among terminals of modules is realized in the routing region.

In this routing phase, we are given an area in a plane, a set of modules whose positions in the area are predetermined and whose terminals on the peripheries are fixed in position, and a set of nets each of which is a set of terminals to be connected electrically. The area not occupied by the modules are available for routing and consists of k layers. Each layer of the routing area can accommodate conductor strips called *wire segments* and the routing area can have through-holes called *vias* to connect conductor strips on different layers. Various constraints are to be considered in determining a routing pattern for a given circuit. Among them, the minimization of the number of vias used is important due to reliability, performance, area compactness and yield. Thus, considerable amount of research has been done on this via minimization problem [2,3,4,5,9,11,14,15,16,20].

Recently, Hsu [10] introduced a rather theoretical routing problem. In this problem a routing pattern is determined purely on the basis of via minimization. The width of wires and size of vias are assumed to be infinitely small and no restriction is imposed on the shapes or positions of wires and vias. Hence it is called the *topological via minimization* problem, and is formally defined as follows. We are given a k-layer ($k \geq 2$) routing area bounded by a continuous curve B, a set C of connected regions without holes called modules inside the curve B, a set T of terminals placed on the curve B and boundaries of modules in C, and a set N of nets. We assume that all terminals are available on every layer and that no wire can be placed inside any module in C. Furthermore, in this paper, we assume that

every net consists of two terminals. Fig. 1 shows an example which contains two modules. The problem is to determine a routing pattern which realizes the connection requirements specified by N using the fewest vias. We call such a problem the *k-layer topological via minimization* problem (abbreviated as TVM(k)). In Fig. 2, we show a possible solution to the TVM(k) problem for the instance shown in Fig 1 with $k = 3$.

Imposing restrictions on C, T and N, we have several interesting subproblems. For example, if $C = \phi$, B is a rectangle and the terminals are located on the top and bottom sides only, the problem arises in the channel routing case [9]. We call such a case the *rectangular* TVM(k) problem, or in short the RTVM(k) problem. In this problem, a net with both terminals located on one side only is called a *local net.* We call the RTVM(k) problem without local nets the *split* RTVM(k) problem. On the other hand, if $|C| = 1$, we can regard the curve B and the boundary of the only module as two concentric circles C_{out} and C_{in}, respectively. Thus, the routing area may be called a *circular channel* and we call the problem in this case the *circular* TVM(k) problem, or in short the CTVM(k) problem. In addition, if each net in N has one terminal on C_{in} and the other on C_{out}, that is, if no local net exists, we call such a problem the *split* CTVM(k) problem.

Having suspected that the RTVM(2) problem was NP-hard, Hsu [10] developed a heuristic algorithm. Later, Marek-Sadowska [13] proposed a new approach to the problem and Chang and Du [2] gave another heuristic algorithm. It was very recently reported by Sarrafzadeh and Lee [19] that the RTVM(2) problem is NP-hard. Although their proof contains a minor flaw, it is not difficult to fix it [17]. Sarrafzadeh and Lee [19] also showed that the split RTVM(2) problem is solvable in $O(n^2 \log n)$ time.

In this paper, we first show that the split RTVM(k) problem is solvable in $O(kn^2)$ time even for $k \geq 2$. We reduce the problem to that of finding a maximum k-path covering of a directed acyclic graph, and then solve it by a minimum cost network flow algorithm. The same approach is taken by Gavril [7] to the problem of finding a maximum k-clique covering of a transitive graph [8]. His algorithm can be applied to our problem as it actually solves the

maximum k-path covering problem. Although he claims that his algorithm is implementable to run in $O(kn^2)$ time by referring to Lawler [12], there is no description in [12] of the case which Gavril's network falls in. Therefore, we describe our solution method for completeness. We then consider the split CTVM(k) problem and present an $O(n^{2k+1})$ time algorithm using dynamic programming.

In the next section we describe some basic definitions and a preliminary result. In Section 3, we show how to solve the split RTVM(k) problem in $O(kn^2)$ time by using a minimum cost network flow algorithm. In Section 4, we develop an $O(n^{2k+1})$ algorithm for the split CTVM(k) problem using dynamic programming. Section 5 concludes the paper with some open problems.

2. Preliminaries

Let $S = [i_1, i_2, \ldots, i_r]$ be a sequence of r distinct integers. We denote by $|S|$ the number of elements contained in S as in the case that S is a set. We call S an *increasing sequence* if $i_p < i_{p+1}$ for $p = 1, 2, \ldots, r-1$. A sequence $S' = [j_1, j_2, \ldots, j_q]$ is called a *subsequence* of S if the elements $j_1, j_2, \ldots, j_q$ appear in S in this order. Given sequences $S_1, S_2, \ldots, S_m$, we denote by $S_1 \cup S_2 \cup \cdots \cup S_m$ the union of the sets of elements contained in $S_1, S_2, \ldots, S_m$ and by $|S_1 \cup S_2 \cup \cdots \cup S_m|$ its cardinality. For a permutation of integers $1, 2, \ldots, n$, let $\pi(i)$ denote the i-th element of π. Thus π is denoted by the sequence $[\pi(1), \pi(2), \ldots, \pi(n)]$. For example, if $\pi = [1, 3, 4, 2]$, $\pi(2) = 3$.

Let $G = (V, E)$ be a directed graph where V is a nonempty set of *vertices* and E is a set of ordered pairs (u, v) of vertices, called *edges*. A sequence of vertices $P = [v_{i_1}, v_{i_2}, \ldots, v_{i_r}]$ is called a *path* from v_{i_1} to v_{i_r} in G if all the vertices but possibly two, v_{i_1} and v_{i_r}, are distinct and $(v_{i_p}, v_{i_{p+1}}) \in E$ for $p = 1, 2, \ldots, r-1$, and in the case in which $v_{i_1} = v_{i_r}$, it is called a *cycle*. Paths in G are said to be *vertex disjoint* if no two of them share a common vertex. If G has no cycle, it is called a *directed acyclic graph*. In a directed acyclic graph $G = (V, E)$, a vertex v is called a *child* of u if $(u, v) \in E$.

We now give a more detailed description of the split RTVM(k) problem. Let $t_1, t_2, \ldots, t_n$ and $b_1, b_2, \ldots, b_n$ be the terminals located in this order from left to right on the top and bottom sides, respectively, of the rectangle k-layer routing area. Without loss of generality, we assume that every terminal belongs to some net. In our problem a net specifies the connection between two terminals, say t_i and b_j, on different sides. Thus, we represent such a net by an ordered pair (t_i, b_j) and the given set of all nets by $N = \{(t_i, b_{j_i}) \mid i = 1, 2, \ldots, n, j_p \neq j_q, 1 \leq p \neq q \leq n\}$. Clearly, the sequence $[j_1, j_2, \ldots, j_n]$ forms a permutation and we denote this by π. In other words, $N = \{(t_i, b_{\pi(i)}) \mid i = 1, 2, \ldots, n\}$. The goal of the split RTVM(k) problem is to realize the connection requirements specified by N using the minimum number of vias. Similarly, the split CTVM(k) problem is defined except that the terminals $t_1, t_2, \ldots, t_n$ and $b_1, b_2, \ldots, b_n$ are now located in the clockwise direction on two concentric circles C_{out} and C_{in}, respectively, rather than on two parallel lines. For example, Figs. 3 (a) and (b) show instances of the split RTVM(k) and split CTVM(k) problems, respectively. In both problems, we will denote net $(t_i, b_{\pi(i)})$ by n_i for $i = 1, 2, \ldots, n$.

We now give a fundamental lemma which applies to the TVM(k) problem in general. Let N be the set of nets in a given instance of the TVM(k) problem, and M be any subset of N. If all the nets in M can be routed on a single layer in such a way that no two wires corresponding to different nets intersect each other, we say that M is *single layer routable.* We have the following lemma.

Lemma 1. Let α be an integer such that $0 \leq \alpha \leq |N|$. All nets in N can be routed on k layers using α vias if and only if there exist k single layer routable subsets $M_1, M_2, \ldots, M_k$ of N such that $|\cup_{i=1}^{k} M_i| = |N| - \alpha$.

Proof. (*if part*) Let $M_1', M_2', \ldots, M_k'$ be subsets of $M_1, M_2, \ldots, M_k$, respectively, such that $\cup_{i=1}^{k} M_i' = \cup_{i=1}^{k} M_i$ and $M_i' \cap M_j' = \phi$ $(i \neq j)$. We first route the nets in M_i' on the i-th layer for each $i = 1, 2, \ldots, k$. We can then route the remaining nets on any two adjacent layers with one via per net by using the approach developed by Marek-Sadowska [13]. The number of vias used is $|N| - |\cup_{i=1}^{k} M_i'| = |N| - |\cup_{i=1}^{k} M_i| = \alpha$.

(*only if part*) For a realization of N using α vias, let M_i' be the set of nets routed using only the i-th layer for each $i = 1, 2, \ldots, k$. Clearly, $|\cup_{i=1}^{k} M_i'| \geq |N| - \alpha$. If the equality does not hold in this equation, we can always find such subsets $M_1, M_2, \ldots, M_k$ of $M_1', M_2', \ldots, M_k'$, respectively, that $|\cup_{i=1}^{k} M_i| = |N| - \alpha$. Since M_i' is single layer routable, so is M_i for $i = 1, 2, \ldots, k$. □

Due to Lemma 1, to solve the TVM(k) problem, all we need to do is to find k single layer routable subsets $M_1, M_2, \ldots, M_k$ of N such that $|\cup_{i=1}^{k} M_i|$ is a maximum. As a subset of a single layer routable set is also single layer routable, we may restrict the subsets $M_1, M_2, \ldots, M_k$ to be mutually disjoint. In the following two sections, we show how to find such subsets of N in polynomial time for the split RTVM(k) and split CTVM(k) problems.

3. Channel Routing Case

In this section, we show that the split RTVM(k) problem is solvable in $O(kn^2)$ time using a minimum cost network flow algorithm. We begin with the following lemma without a proof.

Lemma 2. Let $M = \{(t_{i_p}, b_{\pi(i_p)}) \mid p = 1, 2, \ldots, m\} \subseteq N$ be a set of nets, where $i_1 < i_2 < \cdots < i_m$. M is single layer routable if and only if $[\pi(i_1), \pi(i_2), \ldots, \pi(i_n)]$ is an increasing sequence. □

As mentioned in Section 2, to solve the split RTVM(k) problem, we need to find k mutually disjoint single layer routable subsets $M_1, M_2, \cdots, M_k$ of N such that $|\cup_{i=1}^{k} M_i| = \sum_{i=1}^{k} |M_i|$ is a maximum. To solve this problem, we create a directed graph $G = (V, E)$ such that $V = N$ and $(n_{i_1}, n_{i_2}) \in E$ if and only if $i_1 < i_2$ and $\pi(i_1) < \pi(i_2)$ for nets $n_{i_1} = (t_{i_1}, b_{\pi(i_1)})$ and $n_{i_2} = (t_{i_2}, b_{\pi(i_2)})$ in N. Note that G is acyclic. Fig. 4 depicts the directed acyclic graph G constructed for the instance shown in Fig. 3 (a). Due to Lemma 2, a set of nets M is single layer routable if and only if M is the set of vertices on a directed path in G. Therefore, finding k disjoint single layer routable subsets with the largest sum of cardinalities is equivalent to finding k vertex disjoint paths in G such that the total number

of vertices on those paths is a maximum. To find such paths, we use an algorithm for a minimum cost network flow problem.

A *network* $\mathcal{N} = (U, A, s, t, b, c)$ consists of (i) a finite set of vertices U, (ii) two distinguished vertices in U, the *source* s and the *sink* t, (iii) a set of directed edges A that includes the *return edge* (t, s), (iv) a positive capacity $b(a)$ for each edge $a \in A$, and (v) a cost $c(a)$ for each edge $a \in A$ with $c((t, s)) = 0$. Let $\mathcal{R}$ denote the set of real numbers. A *flow* in $\mathcal{N}$ is a function $f : A \rightarrow \mathcal{R}$ such that (1) $0 \leq f(a) \leq b(a)$ for each edge $a \in A$, and (2) $\sum_u f((u, v)) - \sum_w f((v, w)) = 0$ for each vertex $v \in U$ where each sum is over every vertex u (respectively, w) for which the summand is defined. For convenience, we simply denote $f((u, v)), b((u, v))$ and $c((u, v))$ by $f(u, v), b(u, v)$ and $c(u, v)$, respectively. $f(a)$ is called the *edge flow* of edge a. The value of flow f, denoted by $|f|$, is given by the edge flow of the return edge, that is, $|f| = f(t, s)$. If $f(a)$ is an integer for all $a \in A$, the flow is called *integral.* The cost of flow f is the quantity: $COST(f) = \sum_{a \in A} c(a)f(a)$. The *minimum cost network flow problem* asks for a flow of minimum cost among those with a given fixed flow value k. It is shown [6,12] that if the cost of each edge in $\mathcal{N}$ is nonnegative and its capacity is an integer, the minimum cost network flow problem is solvable in $O(kn^2)$ time using Dijkstra's shortest path algorithm [1,12].

We construct from G the corresponding network $\mathcal{N}(G) = (U, A, s, t, b, c)$ in the following two stages. First, we transform the graph G to a new graph $G^l = (V^l, E^l)$ such that $V^l = V \cup \{s, t\}$ and $E^l = E \cup \{(s, v), (v, t) \mid v \in V\}$. Note that G^l is a directed acyclic graph. In G^l, let the level $l(v)$ of a vertex $v \in V^l$ be defined as (i) $l(s) = 0$ and (ii) for $v \in V^l - \{s\}$, $l(v) = \max\{l(u) \mid l(u) \text{ is defined and } (u, v) \in E^l\} + 1$. Note that the level of every vertex can be computed in $O(n^2)$ time using a conventional topological sorting algorithm [1].

The graph G^l with levels of vertices thus computed is then transformed to a network $\mathcal{N}(G) = (U, A, s, t, b, c)$ as follows:

1. $U = \{v, v' \mid v \in V\} \cup \{s, t\}$, where for each $v \in V$ a new vertex v' is created with $l(v') = l(v) + 1$,

2. $A = \{(t,s)\} \cup A' \cup A_V \cup A_{st}$, where (t,s) is the return edge,

$$\begin{aligned} A' &= \{(u',v) \mid (u,v) \in E\}, \quad A_V = \{(v,v') \mid v \in V\}, \text{ and} \\ A_{st} &= \{(s,v),(v',t) \mid v \in V\}, \end{aligned}$$

3. $b(t,s) = k$ and $b(a) = 1$ for every edge $a \in A - \{(t,s)\}$, and

4. $c(a) = 0$ for $a \in A_V \cup \{(t,s)\}$ and
   $c(a) = l(v) - l(u)$ for $a = (u,v) \in A' \cup A_{st}$.

For example, Fig 5 illustrates a network constructed from the graph shown in Fig. 4. It is easy to see that the above transformation can be completed in $O(n^2)$ time.

Note that the capacities of all edges in the network are integers. In such a network, given an integral flow f, there always exists an integral flow of the minimum cost among all flows with flow value $|f|$ [12]. The following two lemmas show the validity of the transformation given above.

Lemma 3. If there exists an integral flow f with $|f| = k$ in $\mathcal{N}(G)$ for some positive integer k, then there exist k vertex disjoint paths $P_1, P_2, \ldots, P_k$ in G such that $\sum_{i=1}^{k} |P_i| = kl(t) - COST(f)$.

Proof. Since f is an integral flow and $|f| = k$, there exist k (possibly overlapping) paths $P_1', P_2', \ldots, P_k'$ in $\mathcal{N}(G)$ such that the flow f is decomposed into k unit flows each of which is carried along P_i' for $i = 1, 2, \ldots, k$. By the definition of $\mathcal{N}(G)$, each path P_i' in $\mathcal{N}(G)$ corresponds to a path P_i in G such that the set of vertices that appear on P_i is $\{v \in V \mid (v,v')$ appears on $P_i'\}$. Since the capacities of all edges except (t,s) are one and every vertex has either one incoming edge or one outgoing edge in $\mathcal{N}(G)$, the paths $P_1, P_2, \ldots, P_k$ are vertex disjoint. The cost associated with the flow f is expressed as follows:

$$\begin{aligned} COST(f) = &\sum_{v \in V} 0 \cdot f(v,v') + \sum_{(u,v) \in E} (l(v) - l(u'))f(u',v) + \sum_{v \in V} (l(v) - l(s))f(s,v) + \\ &\sum_{v \in V} (l(t) - l(v'))f(v',t) - 0 \cdot f(t,s). \end{aligned}$$

Since $c(v,v') = 0 = l(v') - l(v) - 1$ for every $v \in V$, we have

$$COST(f) = \sum_{(u,v)\in A} (l(v) - l(u))f(u,v) + \sum_{v\in V}(-1)f(v,v') - (l(s) - l(t))f(t,s).$$

It is known that the first term is always 0 for any function $l(\cdot)$ [12]. Since for each $v \in V$, $f(v,v') = 1$ if (v,v') appears on some path in $\mathcal{N}(G)$; or 0 otherwise, the second term is $-\sum |\{(v,v') \mid (v,v') \text{ appears on some } P_i'\}| = -|\cup_{i=1}^{k} P_i|$. Thus, we obtain

$$\begin{aligned} COST(f) &= 0 - |\cup_{i=1}^{k} P_i| + l(t)|f| \\ &= kl(t) - \sum_{i=1}^{k} |P_i|. \quad \square \end{aligned}$$

Lemma 4 If there exist k vertex disjoint paths $P_1, P_2, \ldots, P_k$ in G for some positive integer k, then there exists an integral flow f in $\mathcal{N}(G)$ such that $|f| = k$ and $COST(f) = kl(t) - \sum_{i=1}^{k} |P_i|$.

Proof. Let $P_i = [v_{i_1}, v_{i_2}, \ldots, v_{i_{r(i)}}]$ for $i = 1, 2, \ldots, k$. Clearly for each i, there is a path P_i' in $\mathcal{N}(G)$ defined as $P_i' = [s, v_{i_1}, v'_{i_1}, \ldots, v_{i_{r(i)}}, v'_{i_{r(i)}}, t]$. Since $P_1, P_2, \ldots, P_k$ are vertex disjoint, no two paths P_i' and P_j' $(i \neq j)$ share a common vertex other than s and t. Thus, $P_1', P_2', \ldots, P_k'$ each can carry a unit flow simultaneously. The resultant flow f is integral and $|f| = k$. Since all edge flows on P_i''s are 1, the cost of flow f is derived as follows:

$$\begin{aligned} COST(f) &= \sum_{i=1}^{k} \sum_{a\in\{edges\ on\ P_i'\}} c(a)f(a) \\ &= \sum_{i=1}^{k} [\sum_{j=1}^{r(i)} c(v_{i_j}, v'_{i_j}) + \sum_{j=1}^{r(i)-1} c(v'_{i_j}, v_{i_{j+1}}) + c(s, v_{i_1}) + c(v'_{i_{r(i)}}, t)] \\ &= \sum_{i=1}^{k} [\sum_{j=1}^{r(i)} (l(v'_{i_j}) - l(v_{i_j}) - 1) + \sum_{j=1}^{r(i)-1} (l(v_{i_{j+1}}) - l(v'_{i_j})) + l(v_{i_1}) - l(s) + \\ &\quad l(t) - l(v'_{i_{r(i)}})] \\ &= \sum_{i=1}^{k} (-r(i) - l(s) + l(t)) = kl(t) - \sum_{i=1}^{k} r(i) \\ &= kl(t) - \sum_{i=1}^{k} |P_i|. \quad \square \end{aligned}$$

Due to the above two lemmas, their proofs, and the fact that $kl(t)$ is a constant, k vertex disjoint paths in G with the largest sum of cardinalities can be obtained by finding a minimum cost flow of $\mathcal{N}(G)$ with flow value k. Since the edge costs of $\mathcal{N}(G)$ are all nonnegative, we can directly use the minimum cost network flow algorithm described in [6,12], and hence the following theorem is established.

Theorem 1. The split RTVM(k) problem is solvable in $O(kn^2)$ time. □

In Fig. 6, we show a solution to the split RTVM(k) problem for the instance shown in Fig 3 with $k = 2$.

Remark. An algorithm for the minimum cost network flow problem is described in [12]. It is implementable to run in $O(n^3l)$ time for the general case in which each edge capacity is an integer and there is no negative directed cycle with respect to edge cost, where l is the flow value. The algorithm finds shortest paths of graphs l times (in our case k times). Since negative edges may appear in graphs, it employs an $O(n^3)$ time shortest path algorithm [1,12]. On the other hand, for the case that all edge costs are nonnegative, a technique developed by Edmonds and Karp [6] is explained in [12]. Using this technique, we can always eliminate negative edges in the graphs to be generated during the execution of the algorithm. Thus, it can always employ an $O(n^2)$ time shortest path algorithm [1,12], resulting in an $O(n^2l)$ time implementation. There is no description in [12] of an extension of the Edmonds-Karp technique to the case where there exists negative edges but no negative cycles.

Very recently, Gavril [7] considered the problem of finding k cliques of a transitive graph [8] with the largest sum of cardinalities. Since our graph G is transitive, our k vertex disjoint path cover problem is a special case of this k-clique cover problem. Gavril transformed his problem into a minimum cost network flow problem and claims that it is solvable in $O(n^2k)$ time using a method described in [12]. Unfortunately, his network contains negative edges and as mentioned above, no description is given for such a case in [12]. Although his claim seems to be correct, we presented our approach for the sake of completeness. Note that there are other network transformations which would result in $O(kn^2)$ time

complexity [18]. □

4. Circular Channel Routing Case

In this section, we present an $O(n^{2k+1})$ time algorithm for the split CTVM(k) problem using dynamic programming. Due to Lemma 1, the CTVM(k) problem can be solved by finding k single layer routable sets of nets such that the cardinality of their union is a maximum.

Consider the instance of the CTVM(k) problem shown in Fig. 3 (b) again. For this instance, the permutation $\pi = [11, 4, 5, 9, 10, 6, 2, 3, 7, 8, 1]$. Given such a permutation π, let $S_\pi(i)$, for each $i = 1, 2, \ldots, n$, denote the sequence obtained from the sequence $[\pi(1), \pi(2), \ldots, \pi(n)]$ by circularly shifting the elements $i-1$ positions to the left, that is, $S_\pi(i) = [\pi(i), \pi(i+1), \ldots, \pi(n), \pi(1), \ldots, \pi(i-1)]$. Let $M = \{(t_{i_p}, b_{\pi(i_p)}) \mid p = 1, 2, \ldots, m\}$ be any subset of N. We have the following lemma for the single layer routability of M.

Lemma 5. M is single layer routable if and only if $\{\pi(i_1), \pi(i_2), \ldots, \pi(i_m)\}$ is the set of elements of an increasing subsequence of $S_\pi(l)$ for some integer l, $1 \leq l \leq n$.

Proof. Without loss of generality, we assume that $i_1 < i_2 < \cdots < i_m$. Then, the terminals $t_{i_1}, t_{i_2}, \ldots, t_{i_m}$ form a clockwise sequence on the circle C_{out}. Therefore, M is single layer routable if and only if the terminals $b_{\pi(i_1)}, b_{\pi(i_2)}, \ldots, b_{\pi(i_m)}$ form a clockwise sequence on the circle C_{in}, that is, $[\pi(i_s), \pi(i_{s+1}), \ldots, \pi(i_m), \pi(i_1), \ldots, \pi(i_{s-1})]$ is an increasing sequence for some $s \in \{1, 2, \ldots, n\}$ or equivalently, it is an increasing subsequence of $S_\pi(i)$ for some $i \in \{1, 2, \ldots, n\}$. This completes the proof. □

By Lemma 5, we can find desired k single layer routable subsets $M_1, M_2, \ldots, M_k$ of N in the following way:

1. For each k-tuple $(i_1, i_2, \ldots, i_k)$ of integers $1, 2, \ldots, n$ such that $i_1 \leq i_2 \leq \cdots \leq i_k$, find k increasing subsequences $S_1, S_2, \ldots, S_k$ from $S_\pi(i_1), S_\pi(i_2), \ldots, S_\pi(i_k)$, respectively, such that $|S_1 \cup S_2 \cup \cdots \cup S_k|$ is a maximum. Let $\mathcal{S}_{i_1 i_2 \cdots i_k} = \{S_1, S_2, \ldots, S_k\}$ and $|\mathcal{S}_{i_1 i_2 \cdots i_k}| = |S_1 \cup S_2 \cup \cdots \cup S_k|$.

2. Select $\mathcal{S}^* = \{S_1^*, S_2^*, \ldots, S_k^*\}$ from $\{\mathcal{S}_{i_1 i_2 \cdots i_k} \mid 1 \leq i_1 \leq i_2 \leq \cdots \leq i_k \leq n\}$ such that $|\mathcal{S}^*|$ is the largest. Find the subsets $M_1^*, M_2^*, \ldots, M_k^*$ of N that correspond to $S_1^*, S_2^*, \ldots, S_k^*$, respectively.

In the remainder of this section, we show how to find, from a given k-tuple $(i_1, i_2, \ldots, i_k)$, k increasing subsequences S_1, $S_2, \ldots, S_k$ from $S_\pi(i_1)$, $S_\pi(i_2), \ldots, S_\pi(i_k)$, respectively, such that $|S_1 \cup S_2 \cup \cdots \cup S_k|$ is a maximum. For each $S_\pi(i_r), 1 \leq r \leq k$, we construct a directed graph $G_{i_r} = (V, E_{i_r})$ in such a way that $V = \{v_1, v_2, \ldots, v_n\}$ and $(v_{\pi(p)}, v_{\pi(q)}) \in E_{i_r}$ if $\pi(p)$ appears before $\pi(q)$ in $S_\pi(i_r)$ and $\pi(p) < \pi(q)$ for integers p and q. Note that G_{i_r} is acyclic. For the instance shown in Fig. 3 (b), for example, we illustrate in Fig. 7 two directed acyclic graph G_{i_1} and G_{i_2} constructed from the sequences $S_\pi(i_1)$ and $S_\pi(i_2)$, respectively, where $i_1 = 2$ and $i_2 = 7$. It is easy to see that for $r = 1, 2, \ldots, k$, a sequence $[p_1, p_2, \ldots, p_q]$ forms an increasing subsequence of $S_\pi(i_r)$ if and only if $[v_{p_1}, v_{p_2}, \ldots, v_{p_q}]$ is a path in G_{i_r}. Note that $[v_1, v_2, \ldots, v_n]$ is a topological order for every G_{i_r}, $r = 1, 2, \ldots, k$. Thus we can transform our problem to the following path cover problem: Given directed acyclic graphs $G_1, G_2, \ldots, G_k$, all of which have a common vertex set $V = \{v_1, v_2, \ldots, v_n\}$ and a common topological order $[v_1, v_2, \ldots, v_n]$, find k paths $P_1, P_2, \ldots, P_k$ on $G_1, G_2, \ldots, G_k$, respectively, such that $|\cup_{i=1}^k P_i|$ is a maximum. For two vertices u and v, we can define a precedence relation "$\prec$" between them; namely $u \prec v$ if and only if u appears before (or precedes) v in the topological order. Thus, $v_i \prec v_j$ if and only if $i < j$. If either $u = v$ or $u \prec v$, we denote this relation by $u \preceq v$. For a path $P = [v_{j_1}, v_{j_2}, \ldots, v_{j_q}]$ and a vertex v, let $v + P$ denote the path $[v, v_{j_1}, v_{j_2}, \ldots, v_{j_q}]$ if such a path exists. Furthermore, let $P - v_{j_1}$ denote the path $[v_{j_2}, \ldots, v_{j_q}]$.

We call a collection of paths $\{P_1, P_2, \ldots, P_k\}$ of $G_1, G_2, \ldots, G_k$ that start from vertices $s_1, s_2, \ldots, s_k$, respectively, a $(s_1, s_2, \ldots, s_k)$ *path set*. We define $f(s_1, s_2, \ldots, s_k)$ to be the maximum value of $|P_1 \cup P_2 \cup \cdots \cup P_k|$ among all possible collections of $(s_1, s_2, \ldots, s_k)$ path sets $\{P_1, P_2, \ldots, P_k\}$. Let $child_r(v)$ denote the set of vertices that are children of a vertex v in G_r for $r = 1, 2, \ldots k$. The following lemma lays a foundation for our dynamic programming algorithm.

Lemma 6.

$$f(s_1, s_2, \ldots, s_k) = \begin{cases} 1 + \max_{v \in child_p(s_p)}\{f(s_1, s_2, \ldots, s_{p-1}, v, s_{p+1}, \ldots, s_k)\} \\ \quad \text{if for some } p,\ s_p \prec s_i \text{ for } i = 1, 2, \ldots, p-1, p+1, \ldots, k. \\ \max_{v \in child_p(s_p)}\{f(s_1, s_2, \ldots, s_{p-1}, v, s_{p+1}, \ldots, s_k)\} \\ \quad \text{if for some } p \text{ and } q,\ s_p = s_q \preceq s_i \text{ for all } i \in \{1, 2, \ldots, k\} - \{p, q\}. \end{cases}$$

Proof. We will show that for the case in which $s_1 \preceq s_2 \preceq \cdots \preceq s_k$,

$$f(s_1, s_2, \ldots, s_k) = \begin{cases} 1 + \max_{v \in child_1(s_1)}\{f(v, s_2, \ldots, s_k)\} \text{ if } s_1 \prec s_2 \\ \max_{v \in child_1(s_1)}\{f(v, s_2, \ldots, s_k)\} \text{ if } s_1 = s_2. \end{cases}$$

It is easy to see that this completes the proof. Let $\{P_1^*(s_1), P_2^*(s_2), \ldots, P_k^*(s_k)\}$ be a $(s_1, s_2, \ldots, s_k)$ path set such that $f(s_1, s_2, \ldots, s_k) = |\cup_{i=1}^k P_i^*(s_i)|$. Let u be the child of s_1 such that $P_1^*(s_1) = [s_1, u, w_1, \ldots, w_r]$. Let $P_1^*(u) = P_1^*(s_1) - s_1$. Suppose that

$$|P_1^*(u) \cup (\cup_{i=2}^k P_i^*(s_i))| < f(u, s_2, \ldots, s_k).$$

Then, there is a $(u, s_2, \ldots, s_k)$ path set $\{P_1'(u), P_2'(s_2), \ldots, P_k'(s_k)\}$ such that

$$|P_1'(u) \cup (\cup_{i=2}^k P_i'(s_i))| > |P_1^*(u) \cup (\cup_{i=2}^k P_i^*(s_i))|.$$

If $s_1 \prec s_2$, s_1 is not contained in any paths in any $(u, s_2, \ldots, s_k)$ path set. If $s_1 = s_2$, $s_1 \in P_2'(s_2) \cup P_3'(s_3) \cdots \cup P_k'(s_k)$ and $s_1 \in P_2^*(s_2) \cup P_3^*(s_3) \cdots \cup P_k^*(s_k)$. Therefore,

$$|\{s_1\} \cup P_1'(u) \cup (\cup_{i=2}^k P_i'(s_i))| > |\{s_1\} \cup P_1^*(u) \cup (\cup_{i=2}^k P_i^*(s_i))| = |\cup_{i=1}^k P_i^*(s_i)|$$

in both cases. Since $\{s_1 + P_1'(u), P_2'(s_2), \ldots, P_k'(s_k)\}$ is a $(s_1, s_2, \ldots, s_k)$ path set, this contradicts the optimality of $f(s_1, s_2, \ldots, s_k)$. Hence,

$$f(u, s_2, \ldots, s_k) \leq |P_1^*(u) \cup (\cup_{i=2}^k P_i^*(s_i))|.$$

On the other hand, since $\{P_1^*(u), P_2^*(s_2), \ldots, P_k^*(s_k)\}$ is a $(u, s_2, \ldots, s_k)$ path set, we have that

$$f(u, s_2, \ldots, s_k) \geq |P_1^*(u) \cup P_2^*(s_2) \cup \cdots \cup P_k^*(s_k)|.$$

Thus, we conclude that

$$f(u, s_2, \ldots, s_k) = |P_1^*(u) \cup P_2^*(s_2) \cup \cdots \cup P_k^*(s_k)|.$$

Therefore, if $s_1 \prec s_2$,

$$\begin{aligned} f(s_1, s_2, \ldots, s_k) &= |\cup_{i=1}^{k} P_i^*(s_i)| = 1 + |P_1^*(u) \cup (\cup_{i=2}^{k} P_i^*(s_i))| = 1 + f(u, s_2, \ldots, s_k) \\ &\leq 1 + \max_{v \in child_1(s_1)}\{f(v, s_2, \ldots, s_k)\}, \end{aligned}$$

and if $s_1 = s_2$,

$$\begin{aligned} f(s_1, s_2, \ldots, s_k) &= |\cup_{i=1}^{k} P_i^*(s_i)| = |P_1^*(u) \cup (\cup_{i=2}^{k} P_i^*(s_i))| = f(u, s_2, \ldots, s_k) \\ &\leq \max_{v \in child_1(s_1)}\{f(v, s_2, \ldots, s_k)\}. \end{aligned}$$

Conversely, let u' be a child of s_1 in G_1 and $\{P_1^*(u'), P_2^*(s_2), \ldots, P_k^*(s_k)\}$ be a $(u', s_2, \ldots, s_k)$ path set such that

$$|P_1^*(u') \cup (\cup_{i=2}^{k} P_i^*(s_i))| = \max_{v \in child_1(s_1)}\{f(v, s_2, \ldots, s_k)\}.$$

Note that $s_1 + P_1^*(u')$ forms a path in G_1. If $s_1 \prec s_2$ (respectively, if $s_1 = s_2$), s_1 is not contained in any (respectively, is contained in some) paths in any $(u', s_2, \ldots, s_k)$ path set. Thus, if $s_1 \prec s_2$,

$$\begin{aligned} f(s_1, s_2, \ldots, s_k) &\geq |(s_1 + P_1^*(u')) \cup (\cup_{i=2}^{k} P_i^*(s_i))| = 1 + |P_1^*(u') \cup (\cup_{i=2}^{k} P_i^*(s_i))| \\ &= 1 + \max_{v \in child_1(s_1)}\{f(v, s_2, \ldots, s_k)\}, \end{aligned}$$

and if $s_1 = s_2$,

$$\begin{aligned} f(s_1, s_2, \ldots, s_k) &\geq |(s_1 + P_1^*(u')) \cup (\cup_{i=2}^{k} P_i^*(s_i))| = |P_1^*(u') \cup (\cup_{i=2}^{k} P_i^*(s_i))| \\ &= \max_{v \in child_1(s_1)}\{f(v, s_2, \ldots, s_k)\}. \end{aligned}$$

By combining the above inequalities together, we obtain the desired equation. □

Based on the above lemma, we can develop the following dynamic programming algorithm for finding desired k paths $P_1, P_2, \ldots, P_k$ from $G_1, G_2, \ldots, G_k$, respectively, such that $|P_1 \cup P_2 \cup \cdots \cup P_k|$ is a maximum.

1. **procedure** k-sequences
2. **begin**
3. $f(v_n, v_n, \ldots, v_n) \leftarrow 1$;
4. **for** $t = n - 1$ **to** 1 **do**
5. **begin**
6. **for** $r = 1$ **to** $k - 1$ **do**
7. **for** every r-tuple $(c_1, c_2, \ldots, c_r)$ of integers $1, 2, \ldots, k$ such that $c_1 < c_2 < \cdots < c_r$ **do**
8. **for** $t + 1 \leq j_1, j_2, \ldots, j_{k-r} \leq n$ **do**
9. **begin**
10. **if** $r = 1$ **then**
11. $f(v_{j_1}, v_{j_2}, \ldots, v_{j_{c_1}-1}, v_t, v_{j_{c_1}}, \ldots, v_{j_{k-1}})$ $\leftarrow 1 + \max_{v \in child_{c_1}(v_t)}\{f(v_{j_1}, v_{j_2}, \ldots, v_{j_{c_1}-1}, v, v_{j_{c_1}}, \ldots, v_{j_{k-1}})\}$;
12. **else**
13. $f(v_{j_1}, \ldots, v_{j_{c_1}-1}, v_t, v_{j_{c_1}}, \ldots, v_{j_{c_2}-2}, v_t, v_{j_{c_2}-1}, \ldots, v_{j_{c_r}-r}, v_t, v_{j_{c_r}-r+1}, \ldots, v_{j_{k-r}})$ $\leftarrow \max_{v \in child_{c_1}(v_t)}\{f(v_{j_1}, v_{j_2}, \ldots, v_{j_{c_1}-1}, v, v_{j_{c_1}}, \ldots, v_{j_{c_2}-2}, v_t, v_{j_{c_2}-1}, \ldots, v_{j_{c_r}-r}, v_t, v_{j_{c_r}-r+1}, \ldots, v_{j_{k-r}})\}$;
14. **end**
15. $f(v_t, v_t, \ldots, v_t) \leftarrow \max_{v \in child_1(v_t)}\{v, v_t, v_t, \ldots, v_t\}$;
16. **end**
17. optimal $\leftarrow \max_{1 \leq d_1, d_2, \ldots, d_k \leq n}\{f(v_{d_1}, v_{d_2}, \ldots, v_{d_k})\}$;
18. **end**

The correctness of the above algorithm directly follows Lemma 6. Note that desired optimum paths can easily be obtained by adding some appropriate pointer operations to the above procedure.

We now analyze the time complexity of our algorithm. We assume that the values $f(v_{d_1}, v_{d_2}, \ldots, v_{d_k})$ are stored in a table, where $1 \leq d_1, d_2, \ldots, d_k \leq n$, and hence those

values can be accessed in constant time. Then, the execution of operations on lines 11,13 and 15 each needs $O(n)$ time since each vertex in V has at most $n-1$ outgoing edges. The iterations shown on lines 4,6,7 and 8 are to calculate the value $f(v_{d_1}, v_{d_2}, \ldots, v_{d_k})$ for every $1 \leq d_1, d_2, \ldots, d_k \leq n$ except $f(v_n, v_n, \ldots, v_n)$ in such a way that a value of $f(\cdot)$ can be computed using the values already computed. Since there are $n^k - 1$ values to be computed, it takes $O(n^{k+1})$ time to execute the above procedure. We need to perform the above procedure for every possible k-tuples $(i_1, i_2, \ldots, i_k)$ of integers $1, 2, \ldots, n$ such that $i_1 \leq i_2 \leq \cdots \leq i_k$. Since the number of such tuples is $O(n^k)$, we arrive at the following theorem.

Theorem 2. The split CTVM(k) problem is solvable in $O(n^{2k+1})$ time. □

A solution to the split CTVM(k) problem for the instance shown in Fig. 3 (b) is given in Fig. 8.

5. Conclusion

In this paper, we have presented polynomial time algorithms for two special cases of the k-layer topological via minimization problem. One corresponds to the channel routing case (split RTVM(k)) and the other, the circular channel routing case (split CTVM(k)), in which every net consists of two terminals and no local net exists. We have first shown that the problem is solvable in $O(kn^2)$ time for the channel routing case, using a minimum cost network flow algorithm. We have then presented an $O(n^{2k+1})$ time algorithm for the circular channel routing case using dynamic programming.

There are two interesting problems related to ours. One is to find the minimum number of layers required to route all the nets without any vias. It is easy to see that this problem for the channel routing case is equivalent to the minimum vertex coloring problem for permutation graphs for which an $O(n \log n)$ algorithm is available [8]. The same problem for the circular channel routing case is under investigation. Another problem is to find a maximum cardinality single layer routable subset of nets. For convenience we call this problem the

TVM(1) problem, although strictly speaking, no via minimization is involved. Clearly, the first algorithm finds a solution in $O(n^2)$ time for the channel routing case, and the second algorithm with appropriate modifications solves it in $O(n^3)$ time for the circular routing case. On the other hand, the split RTVM(1) problem is equivalent to that of finding a maximum independent set of a permutation graph and thus solvable in $O(n \log n)$ time [8]. Furthermore, the split CTVM(1) problem can be transformed to a split RTVM(1) by cutting out the circular channel. By solving such split RTVM(1) problems n times, we can eventually find a solution to the split CTVM(1) problem, leading to an $O(n^2 \log n)$ time implementation. The development of a faster algorithm for the split CTVM(1) problem is under way.

References

[1] A.V. Aho, J.E.Hopcroft, and J.D. Ullman, *The Design and Analysis of Computer Algorithms.* Reading, MA: Addison Wesley, 1974.

[2] K. C. Chang and D. H-C. Du, "Efficient algorithms for layer assignment problem," *IEEE Trans. on Computer-Aided Design*, vol. CAD-6, pp. 67–78, Jan. 1987.

[3] K. C. Chang and H. C. Du, "Layer assignment problem for three-layer routing," *IEEE Trans. on Computers*, vol. 37, No. 5, pp 625–632, May 1988.

[4] R. W. Chen, Y. Kajitani and S. P. Chan, "A graph-theoretic via minimization algorithm for two-layer printed circuit board," *IEEE Trans. on Circuit and Systems*, vol. CAS30-5, pp. 284–299, May 1983.

[5] H-A. Choi, K. Nakajima and C. S. Rim, "Graph bipartization and via minimization," *SIAM Jour. on Discrete Mathematics*, vol. 2, Feb. 1989; see also *Proc. of the 25th Annual Allerton Conf. on Communication, Control, and Computing*, Montecillo, IL, Sept. 1987 pp. 4–13.

[6] J. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *JACM*, vol. 14, pp 248–264, 1972.

[7] F. Gavril, "Algorithms for maximum k-colorings and k-coverings of transitive graphs," *Networks*, vol. 17, pp. 465–470, 1987.

[8] M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs.* New York, Academic Press, 1980.

[9] A. Hashimoto and J. Stevens, "Wire routing by channel assignment within large apertures," *Proc. 8th Design Automation Workshop*, June 1971, pp. 155–169.

[10] C. P. Hsu, "Minimum via topological routing," *IEEE Trans. on Computer-Aided Design of ICAS*, vol. CAD-2, pp. 235–246, Oct. 1983.

[11] Y. Kajitani, "On via hole minimization of routing on a 2-layer board," *Proc. 1980 Int'l Conference on Circuits and Computers*, June 1980, pp. 295–298.

[12] E. L. Lawler, *Combinatorial optimization: networks and metroids.* Reading, Holt, Rinehart and Winston, 1976.

[13] M. Marek-Sadowska, "An unconstraint topological via minimization," *IEEE Trans. on Computer-Aided Design of ICAS*, vol. CAD-3, pp. 184–190, July 1984.

[14] N. J. Naclerio, S. Masuda and K. Nakajima, "Via minimization for gridless layout," *Proc. 24th Design Automation Conference*, June 1987, pp. 159–165.

[15] N. J. Naclerio, S. Masuda and K. Nakajima, "The via minimization problem is NP-complete," *IEEE Trans. on Computers*, to appear; see also *Proc. 1987 Conference on Infor. Sci. and Systems*, Baltimore, MD, March 1987, pp. 611–616.

[16] R. P. Pinter, "Optimal layer assignment for interconnect," *Proc. Int'l Conf. on Circuits and Computers*, New York, NY, Sept. 1982, pp. 398–401.

[17] C. S. Rim, T. Kashiwabara and K. Nakajima, "A note on the NP-completeness proof of the topological via minimization problem," Tech. Report, Systems Research Center, University of Maryland, College Park, MD, 1988.

[18] C. S. Rim and K. Nakajima, "Finding k Independent Sets of a Graph with the Maximum Total Size," presented at the Conf. on Infor. Sci. and Systems, Princeton, NJ, March 1988.

[19] M. Sarrafzadeh and D. T. Lee, "A new approach to topological via minimization," Tech. Report CIMS-87-01, Northwestern University, Evanston, IL, Nov. 1987.

[20] X-M. Xiong and E. S. Kuh, "The constrained via minimization problem for PCB and VLSI design," *Proc. 25th Design Automation Conference*, June 1988, pp 573–578.

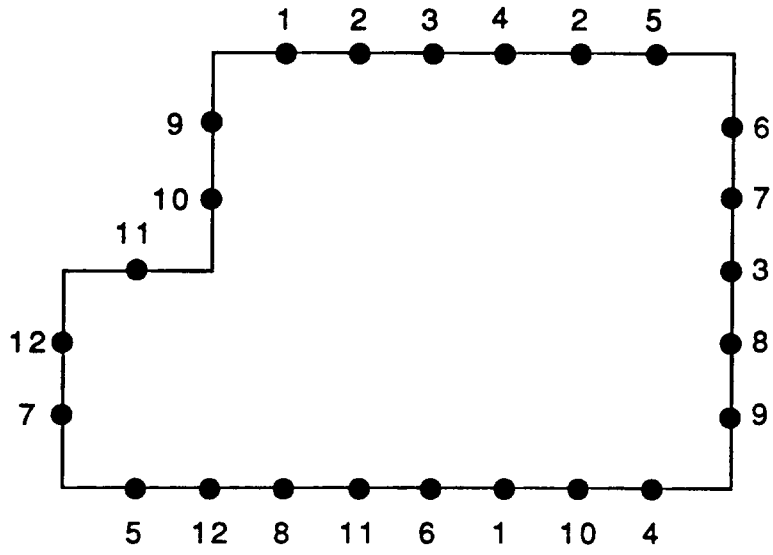


Fig. 1. An instance of the TVM(k) problem (terminals with the same number are to be connected).

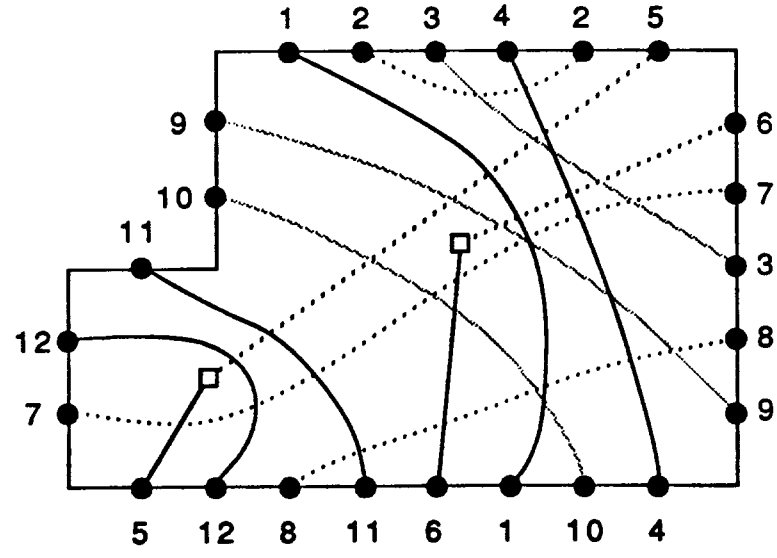
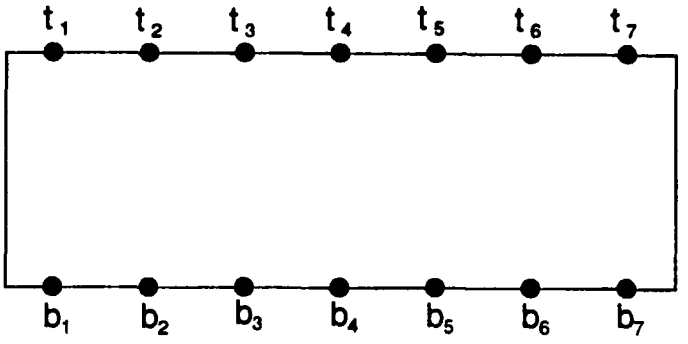
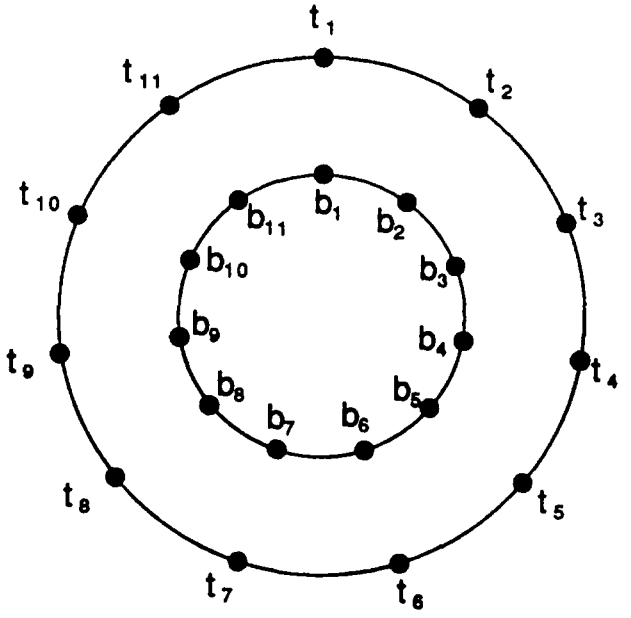


Fig. 2. A possible solution to the TVM(k) problem for the instance of Fig. 1 with $k = 3$.



(a) $N = \{(t_1, b_5), (t_2, b_2), (t_3, b_7), (t_4, b_1), (t_5, b_4), (t_6, b_3), (t_7, b_6)\}$.



(b) $N = \{(t_1, b_{11}), (t_2, b_4), (t_3, b_5), (t_4, b_9), (t_5, b_{10}), (t_6, b_6), (t_7, b_2), (t_8, b_3), (t_9, b_7), (t_{10}, b_8), (t_{11}, b_1)\}$.

Fig. 3. (a) An instance of the split RTVM(k) problem.
(b) An instance of the split CTVM(k) problem.

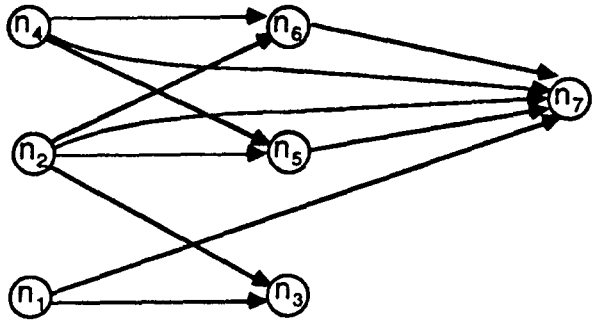


Fig. 4. The directed acyclic graph G for the instance of Fig. 3 (a).

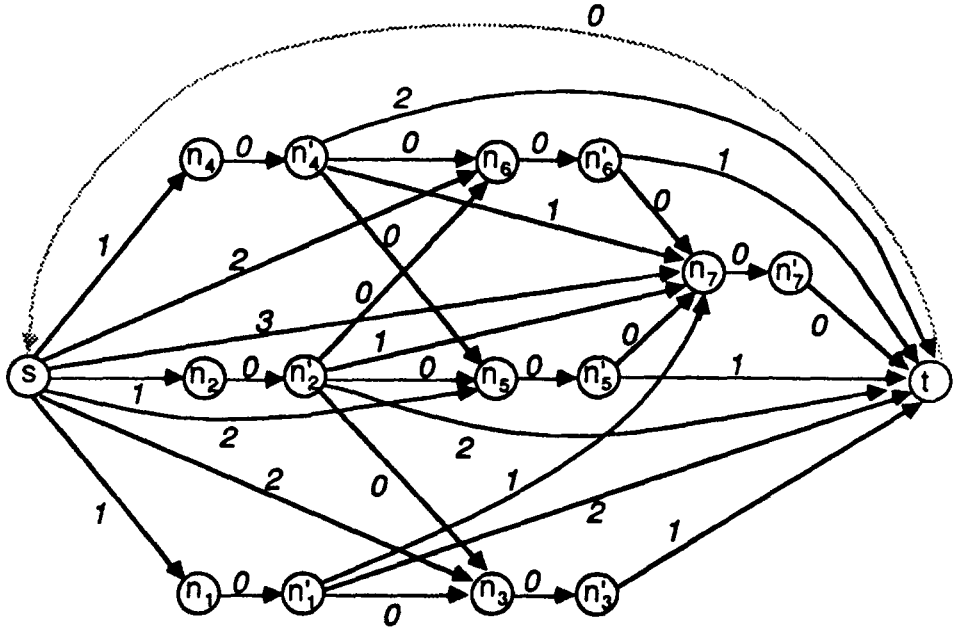


Fig. 5. The network $\mathcal{N}(G)$ constructed from the graph G of Fig. 4.

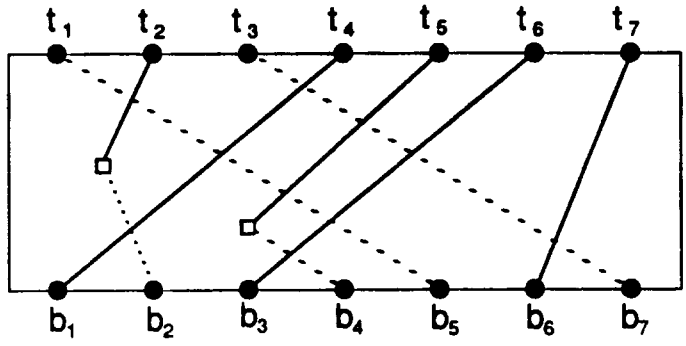
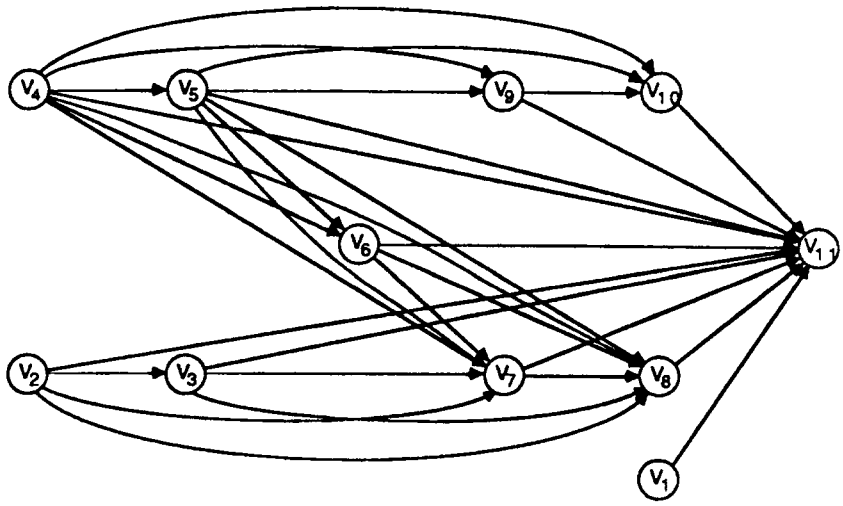
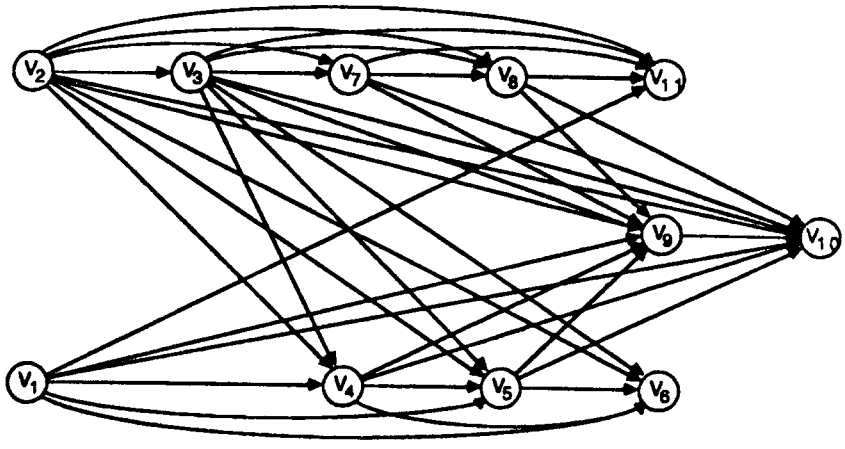


Fig. 6. A possible solution to the split RTVM(k) problem for the instance of Fig. 3 (a) with $k = 2$.



(a) G_2.



(b) G_7.

Fig. 7. Two directed acyclic graphs constructed from the sequences $S_\pi(2)$ and $S_\pi(7)$ for the instance of Fig. 3 (b).

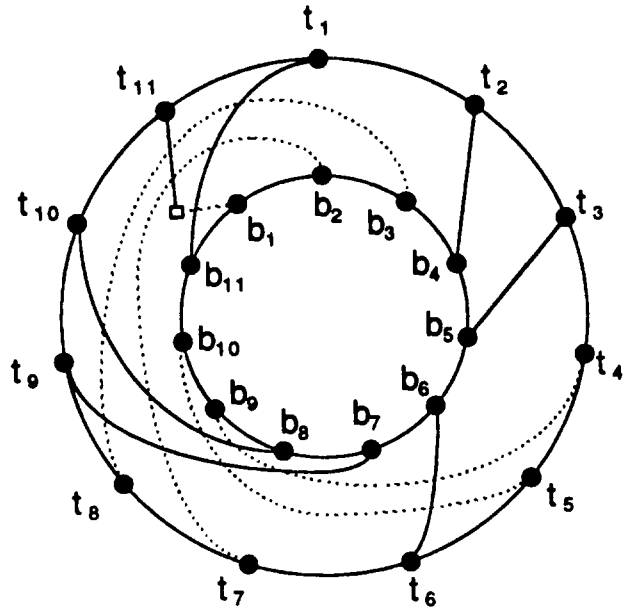


Fig. 8. A possible solution to the split CTVM(k) problem for the instance of Fig. 3 (b) with $k = 2$.