

**A Linear Time Algorithm for  
Circular Permutation Layout**

by

**C.S. Rim, N.J. Naclerio, S. Masuda,  
and K. Nakajima**

# A Linear Time Algorithm for Circular Permutation Layout<sup>1</sup>

**Chong S. Rim**

Electrical Engineering Department and Systems Research Center  
University of Maryland, College Park, MD 20742

**Nicholas J. Naclerio<sup>2</sup>**

Air Force Wright Aeronautical Laboratories  
Wright-Patterson Air Force Base, Ohio 45433

**Sumio Masuda<sup>3</sup>**

Department of Information and Computer Sciences  
Osaka University, Toyonaka, Osaka 560, Japan

**Kazuo Nakajima**

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

<sup>1</sup> This work was supported in part by National Science Foundation grants MIP-84-51510 and CDR-85-00180 and a grant from AT&T.

<sup>2</sup> N. J. Naclerio was with the Electrical Engineering Department and Systems Research Center at the University of Maryland, College Park MD 20783. He was also supported during the course of this work by a fellowship from Westinghouse Electric Corporation.

<sup>3</sup> The work of S. Masuda was done while he was visiting the Electrical Engineering Department and Systems Research Center at the University of Maryland, College Park, MD 20742.

## Abstract

Suppose that two sets of terminals  $t_1, t_2, \dots, t_n$  and  $b_1, b_2, \dots, b_n$  are located on two concentric circles  $C_{out}$  and  $C_{in}$ , respectively. Given a permutation  $\pi$  of integers  $1, 2, \dots, n$ , the circular permutation layout problem is the problem of connecting each pair of terminals  $t_i$  and  $b_{\pi(i)}$  for  $i = 1, 2, \dots, n$  with zero width wires in such a way that no two wires which correspond to different terminal pairs intersect each other. In this paper, we present a linear time algorithm for the following case: (i) no wire can cross  $C_{out}$ , (ii) at most one wire can pass between any two adjacent terminals on  $C_{in}$ , and (iii) no wire can cross  $C_{in}$  more than once. The previously known algorithm for the same case has time complexity  $O(n^2)$ .

# 1. Introduction

Suppose that two sets of terminals  $t_1, t_2, \dots, t_n$  and  $b_1, b_2, \dots, b_n$  are located on two concentric circles  $C_{out}$  and  $C_{in}$ , respectively. We assume that the circle  $C_{out}$  is outside the circle  $C_{in}$  and that the terminals on each circle are labeled in ascending order of their subscripts in the clockwise direction. Given a permutation  $\pi$  of integers  $1, 2, \dots, n$ , the *circular permutation layout* (CPL) problem is the problem of connecting each pair of terminals  $t_i$  and  $b_{\pi(i)}$  for  $i = 1, 2, \dots, n$  with zero width wires in such a way that no two wires which correspond to different terminal pairs intersect each other.

The CPL problem was first proposed by Ozawa [6] as an extension of the *linear permutation layout* (LPL) problem which has widely been studied [2,3,4,5,7,8]. As pointed out by Ozawa [6], the CPL problem may arise in the design of hybrid integrated circuits and printed circuit boards (PCBs). Consider, for example, the problem of connecting a series of incoming wires to the pins on a particular module on a PCB (see Fig. 1). We assume that no two pins on the module are electrically equivalent and hence any two wires for the connections must not intersect each other. The order of the incoming wires is previously determined and may not be the same as that of the pins on the module. If only one layer is available to realize the connections, this problem is equivalent to the CPL problem. The terminals on  $C_{in}$  correspond to the pins on the module and the terminals on  $C_{out}$  represent the incoming wires. Since the pins have fixed spacing and the wires have finite width, we assume that the number of wires which can pass between two adjacent pins is limited to one. No wires may cross the boundary represented by  $C_{out}$  since they would intersect the existing incoming wires.

From the observations mentioned above, we consider the CPL problem with the following constraints on the wires in its solution layout [6]:

1. No wire can cross  $C_{out}$ ,
2. At most one wire can pass between any two adjacent terminals on  $C_{in}$ , and
3. No wire can cross  $C_{in}$  more than once.

We call a layout in which wires satisfy the above constraints a *CPL1-solution*. For example, Fig. 2 shows a CPL1-solution for the permutation  $\pi = (1\ 24\ 22\ 21\ 9\ 6\ 5\ 8\ 7\ 4\ 10\ 20\ 19\ 15\ 14\ 13\ 18\ 16\ 17\ 12\ 11\ 3\ 23\ 2)$ . For simplicity, we label each terminal  $t_i$  or  $b_i$  as  $i$  in the figures throughout this paper. Ozawa [6] developed an  $O(n^2)$  time algorithm for finding a CPL1-solution if one exists, where  $n$  is the number of terminal pairs to be connected.

In this paper, we present an  $O(n)$  time algorithm for the same CPL problem. In Section 2 we introduce some basic definitions and notation. Section 3 describes some useful properties of a CPL1-solution. In Section 4 we explain merging operations, which play an important role in our algorithm. The algorithm is presented in Section 5.

## 2. Definitions and Notation

Let  $\pi$  be a given permutation of integers  $1, 2, \dots, n$ . If there exists a CPL1-solution for  $\pi$ , we say that  $\pi$  is *realizable*. A *net* is an ordered pair of terminals  $(t_i, b_{\pi(i)})$  which must be electrically connected. Let  $N(\pi) = \{n_i = (t_i, b_{\pi(i)}) \mid 1 \leq i \leq n\}$  be the set of nets.

Assume that  $\pi$  is realizable and let  $L_{N(\pi)}$  be a CPL1-solution for it. We denote by  $w_i$  the wire in  $L_{N(\pi)}$  which realizes the net  $n_i = (t_i, b_{\pi(i)})$ . If  $w_i$  crosses the circle  $C_{in}$ , it is called an *indirect wire*; otherwise it is called a *direct wire*. For example, in Fig. 2,  $w_1$  is a direct wire

and  $w_2$  is an indirect wire. We assume that a net is routed by an indirect wire in  $L_{N(\pi)}$  if and only if it can not be replaced with a direct wire.

We define  $l \oplus 1$  to be  $l + 1$  if  $1 \leq l \leq n - 1$  and  $1$  if  $l = n$ . Because of the third constraint on a CPL1-solution, each indirect wire in  $L_{N(\pi)}$  crosses  $C_{in}$  exactly once. If a wire  $w_i$  crosses  $C_{in}$  between two terminals  $b_j$  and  $b_{j \oplus 1}$ , we denote this fact by  $w_i \wr b_{j \oplus 1}$ . For example,  $w_2 \wr b_2$  in Fig. 2.

Let  $p_1, p_2, \dots, p_k$  be distinct integers between 1 and  $n$ . A sequence  $[p_1, p_2, \dots, p_k]$  is called an *increasing consecutive sequence* (**icseq**) if  $p_i \oplus 1 = p_{i+1}$  for  $i = 1, 2, \dots, k - 1$ . Let  $M = \{n_{p_i} = (t_{p_i}, b_{q_i}) \mid 1 \leq i, j \leq k\}$  be a nonempty subset of  $N(\pi)$  such that  $[p_1, p_2, \dots, p_k]$  and  $[q_1, q_2, \dots, q_k]$  are **icseqs**. We call  $M$  a *cluster* in  $N(\pi)$  if  $\pi(p_i) = q_{k-i+1}$  for  $1 \leq i \leq k$ . If a cluster consists of only one net, it is called a *trivial cluster*; otherwise it is called a *nontrivial cluster*. A cluster is *maximal* if and only if it is not contained in any other cluster. For example, the instance shown in Fig. 2 has the following fifteen maximal clusters:  $\{n_3, n_4\}$ ,  $\{n_5\}$ ,  $\{n_6, n_7\}$ ,  $\{n_8, n_9\}$ ,  $\{n_{10}\}$ ,  $\{n_{11}\}$ ,  $\{n_{12}, n_{13}\}$ ,  $\{n_{14}, n_{15}, n_{16}\}$ ,  $\{n_{17}\}$ ,  $\{n_{18}\}$ ,  $\{n_{19}\}$ ,  $\{n_{20}, n_{21}\}$ ,  $\{n_{22}\}$ ,  $\{n_{23}\}$  and  $\{n_{24}, n_1, n_2\}$ . It is easy to see that if two distinct clusters  $C$  and  $C'$  are both maximal, then  $C \cap C' = \phi$ . Furthermore, if  $C \cup C' = N(\pi)$ , then  $N(\pi)$  itself is a cluster. Thus, if  $N(\pi)$  is not a single cluster, it can uniquely be partitioned into three or more maximal clusters.

Let  $M' = \{n_{p'_i} = (t_{p'_i}, b_{q'_i}) \mid 1 \leq i, j \leq m\}$  be another subset of  $N(\pi)$  such that  $[p'_1, p'_2, \dots, p'_m]$  and  $[q'_1, q'_2, \dots, q'_m]$  are **icseqs**. We say that  $M$  is *parallel* to  $M'$ , denoted by  $M \parallel M'$ , if and only if  $p_k \oplus 1 = p'_1$  and  $q_k \oplus 1 = q'_1$ . For example,  $\{n_6, n_7\} \parallel \{n_8, n_9\}$  and  $\{n_{14}, n_{15}, n_{16}\} \parallel \{n_{17}, n_{18}, n_{19}\}$  in Fig. 2.

### 3. Layout of Maximal Clusters

Assume that the given permutation  $\pi$  is realizable and that  $N(\pi)$  has three or more maximal clusters. In this section, we investigate how maximal clusters are realized in a CPL1-solution for  $\pi$ .

Let  $C = \{n_{p_i} = (t_{p_i}, b_{q_{k-i+1}}) \mid 1 \leq i \leq k\}$  be a cluster in  $N(\pi)$ , where  $[p_1, p_2, \dots, p_k]$  and  $[q_1, q_2, \dots, q_k]$  are **icseqs**. It is clear that the layout of  $C$  has at most one direct wire in any CPL1-solution for  $\pi$ . We first define two types of layouts of  $C$  which have a direct wire. Note that  $\lceil x \rceil$  denotes the smallest integer which is not less than  $x$ .

1. A Type DL layout of  $C$  is the layout such that  $w_{p_{\lceil (k+1)/2 \rceil}}$  is a direct wire and  $w_{p_i} \wr b_{q_i}$  for  $i \neq \lceil (k+1)/2 \rceil$  (see Fig. 3 (a)).
2. A Type DR layout of  $C$  is the layout such that  $w_{p_{\lfloor k/2 \rfloor}}$  is a direct wire and  $w_{p_i} \wr b_{q_{i \oplus 1}}$  for  $i \neq \lfloor k/2 \rfloor$  (see Fig. 3 (b)).

We call the layouts defined above Type D layouts for simplicity. In Fig. 2, the layout of  $\{n_{24}, n_1, n_2\}$  is a Type DL layout and that of  $\{n_6, n_7\}$  is a Type DR layout.

It may also be possible to route the nets in  $C$  using only indirect wires. Let  $C' = \{n_{p'_i} = (t_{p'_i}, b_{q'_{m-i+1}}) \mid 1 \leq i \leq m\}$  be another cluster in  $N(\pi)$ , where  $[p'_1, p'_2, \dots, p'_m]$  and  $[q'_1, q'_2, \dots, q'_m]$  are **icseqs**. We define three types of layouts for  $C \cup C'$  which have only indirect wires:

1. A Type IL layout of  $C \cup C'$  is the layout such that  $m = k$ , and  $w_{p_i} \wr b_{q'_i}$  and  $w_{p'_i} \wr b_{q_i}$  for  $1 \leq i \leq k$  (see Fig. 4 (a)).
2. A Type IR layout of  $C \cup C'$  is the layout such that  $m = k$ , and  $w_{p_i} \wr b_{q'_i \oplus 1}$  and  $w_{p'_i} \wr b_{q_i \oplus 1}$  for  $1 \leq i \leq k$  (see Fig. 4 (b)).

3. A Type IM layout of  $C \cup C'$  is the layout such that either (i)  $m = k - 1$ ,  $w_{p_k} \wr b_{q'_m \oplus 1}$ , and  $w_{p_i} \wr b_{q'_i}$  and  $w_{p'_i} \wr b_{q_{i+1}}$  for  $1 \leq i \leq k - 1$  (see Fig 4 (c)) or (ii)  $m = k + 1$ ,  $w_{p'_m} \wr b_{q_k \oplus 1}$ , and  $w_{p_i} \wr b_{q'_{i+1}}$  and  $w_{p'_i} \wr b_{q_i}$  for  $1 \leq i \leq k$ .

We call these three types of layouts Type I layouts. In Fig. 2,  $\{n_3, n_4\} \cup \{n_{22}\}$  forms a Type IM layout. The layout of  $\{n_{12}, n_{13}\} \cup \{n_{20}, n_{21}\}$  is a Type IL layout and that of  $\{n_5\} \cup \{n_{10}\}$  is a Type IR layout. If  $C \cup C'$  forms a Type I layout, we say that  $C$  (resp.,  $C'$ ) is the *mate* cluster of  $C'$  (resp.,  $C$ ). For example, in Fig. 2,  $\{n_{22}\}$  is the mate cluster of  $\{n_3, n_4\}$  with respect to their Type IM layout.

Let  $L_{N(\pi)}$  be a CPL1-solution for  $\pi$  and, for any subset  $M$  of  $N(\pi)$ , let  $L_M$  denote the layout of  $M$  in  $L_{N(\pi)}$ .

**Lemma 1.** *Let  $C$  be a nontrivial cluster in  $N(\pi)$ . If  $L_C$  consists of only indirect wires, then there exists another cluster  $C'$  such that  $|C'| = |C| - 1$  and  $L_{C \cup C'}$  is a Type IM layout.*

**Proof.** Let  $C = \{n_{p_i} = (t_{p_i}, b_{q_{k-i+1}}) \mid 1 \leq i \leq k\}$ , where  $[p_1, p_2, \dots, p_k]$  and  $[q_1, q_2, \dots, q_k]$  are *icseqs*. For  $i = 1, 2, \dots, k-1$ , let  $q'_i$  and  $p'_i$  be the integers such that  $w_{p_i} \wr b_{q'_i}$  and  $\pi(p'_i) = q'_{k-i}$ . Let  $C' = \{n_{p'_i} = (t_{p'_i}, b_{q'_{k-i}}) \mid 1 \leq i \leq k-1\}$ . Since  $[p_1, p_2, \dots, p_k]$  and  $[q_1, q_2, \dots, q_k]$  are *icseqs*,  $w_{p'_i}$  is an indirect wire and passes between  $b_{q_i}$  and  $b_{q_{i+1}}$  for  $i = 1, 2, \dots, k-1$ . By the second constraint on a CPL1-solution, both  $[q'_1, q'_2, \dots, q'_{k-1}]$  and  $[p'_1, p'_2, \dots, p'_{k-1}]$  are *icseqs*. Thus,  $C'$  is a cluster and  $L_{C \cup C'}$  is a Type IM layout.  $\square$

If the only net in a trivial cluster  $C$  is routed by an indirect wire, it may have no mate cluster. In such a case, we say that  $C$  forms a Type IM *layout by itself*. For example, in Fig. 2,  $\{n_{17}\}$  forms a Type IM layout by itself.

We are now ready to show the following theorems.



**Theorem 1.** *Let  $C$  be a maximal cluster. If  $L_C$  has a direct wire, it is a Type D layout.*

**Proof.** Let  $C = \{n_{p_i} = (t_{p_i}, b_{q_{k-i+1}}) \mid 1 \leq i \leq k\}$ , where  $[p_1, p_2, \dots, p_k]$  and  $[q_1, q_2, \dots, q_k]$  are icseqs. If  $C$  is trivial, the theorem clearly holds. Suppose that  $C$  is not trivial and let  $w_{p_j}$  be the direct wire in  $L_C$ . Without loss of generality, we assume that  $j - 1 \geq k - j$ . Let  $C_1 = \{n_{p_i} \mid 1 \leq i \leq j - 1\}$ . It is obvious that  $C_1$  is a cluster and its layout consists of only indirect wires. For  $i = 1, 2, \dots, j - 1$ , let  $q'_i$  and  $p'_i$  be the integers such that  $w_{p_i} \wr \bullet b_{q'_i}$  and  $\pi(p'_i) = q'_{j-i}$ . Note that  $q'_{j-1}$  may be equal to  $q_{k-j+1}$  ( $= \pi(p_j)$ ). Let  $C_2 = \{n_{p'_i} \mid 1 \leq i \leq j - 1\}$ . See Fig. 5. From Lemma 1,  $C_2 - \{n_{p'_1}\}$  is a cluster and  $C_1 \cup C_2 - \{n_{p'_1}\}$  forms a Type IM layout in  $L_{N(\pi)}$ .

Assume that  $q'_{j-1} \neq q_{k-j+1}$ . Due to the constraints on a CPL1-solution,  $w_{p'_1}$  is the unique indirect wire that passes between  $b_{q_{k-j+1}}$  and  $b_{q_{k-j+2}}$ . Thus,  $p'_1 = p_j \oplus 1$  and  $\{n_{p_j}, n_{p'_1}\}$  is a cluster. This implies that  $C_1 \cup C_2 \cup \{n_{p_j}\}$  is a cluster and it forms a Type DL layout in  $L_{N(\pi)}$ . This statement holds even if  $q'_{j-1} = q_{k-j+1}$ . Since  $j - 1 \geq k - j$ ,  $C = C_1 \cup C_2 \cup \{n_{p_j}\}$  or  $C - \{n_{p_k}\} = C_1 \cup C_2 \cup \{n_{p_j}\}$ . It is easy to show that, in the latter case,  $w_{p_k} \wr \bullet b_{q_k \oplus 1}$  and  $L_C$  is a Type DR layout.  $\square$

**Theorem 2.** *Let  $C$  be a maximal cluster. If  $L_C$  consists of only indirect wires, then  $C$  forms a Type IM layout by itself or there exists a maximal cluster  $C'$  such that  $|C| - 1 \leq |C'| \leq |C| + 1$  and  $L_{C \cup C'}$  is a Type I layout.*

**Proof.** Suppose that  $C$  is not trivial. By Lemma 1, there is another cluster  $C_1$  such that  $|C_1| = |C| - 1$  and  $L_{C \cup C_1}$  is a Type IM layout. Let  $C'$  be the maximal cluster which contains  $C_1$ . Clearly,  $|C'| \geq |C| - 1$  and  $C \cap C' = \emptyset$ . Thus,  $L_{C'}$  has only indirect wires due to Theorem 1, which implies that there exists a cluster  $C_2$  such that  $|C_2| = |C'| - 1$  and

$C' \cup C_2$  forms a Type IM layout. Since  $C \cap C_2 \neq \phi$  and  $C$  is maximal,  $C \supseteq C_2$ , and hence  $|C'| \leq |C| + 1$ . If  $|C'| = |C| + 1$ , then  $C = C_2$ . If  $|C'| = |C| - 1$ , then  $C' = C_1$ . As mentioned above,  $C$  and  $C'$  form a Type IM layout in these cases. Furthermore, it is easy to see that if  $|C| = |C'|$ , then  $L_{C \cup C'}$  is either a Type IL or Type IR layout.

If  $C$  is trivial and has a mate cluster  $C_1$ , then the maximal cluster containing  $C_1$  has at most two nets. Thus, either  $L_C$  is a Type IM layout by itself or  $C$  forms a Type I layout with a maximal cluster of one or two elements.  $\square$

## 4. Components and Merging Operations

Let  $M = \{n_{p_i} = (t_{p_i}, b_{q_j}) \mid 1 \leq i, j \leq k\}$  be the union of one or more maximal clusters such that  $[p_1, p_2, \dots, p_k]$  and  $[q_1, q_2, \dots, q_k]$  are icseqs. Let  $b_{q_0}$  and  $b_{q_{k+1}}$  be the terminals such that  $q_1 = q_0 \oplus 1$  and  $q_{k+1} = q_k \oplus 1$ . We call  $M$  a *component* of  $N(\pi)$  if it has a property that if  $\pi$  is realizable, then its layout in any CPL1-solution satisfies the following restriction (\*):

(\*) For  $i = 1, 2, \dots, k$ ,  $w_{p_i}$  is either a direct wire or an indirect wire which passes between two adjacent terminals in  $\{b_{q_0}, b_{q_1}, \dots, b_{q_k}, b_{q_{k+1}}\}$ .

If there is a layout of  $M$  which satisfies all of the constraints on a CPL1-solution and the above restriction (\*), we say that  $M$  is *locally realizable* and call such a layout a *local realization* of  $M$ . Locally realizable subsets are not always components. For example, any maximal cluster is locally realizable, but not all maximal clusters are components as  $\{n_3, n_4\}$  in Fig. 2 is not. On the other hand, every component is locally realizable as long as  $\pi$  is realizable.

A *merging operation* is an operation that creates a new component by combining a component called the *core* with another component or one or two maximal clusters. In our algorithm, which will be described in the next section, we first partition  $N(\pi)$  into maximal clusters and find a component among them. We then repeatedly perform merging operations. If we find a component which is not locally realizable, we know that  $\pi$  is not realizable. On the other hand, if all of the maximal clusters are merged into a locally realizable component, then clearly  $\pi$  is realizable. In this section, we define two types of merging operations and explain how we test the local realizability of a component.

Suppose that  $M$  is a component and it has a local realization  $L_M$ . If there is a wire  $w$  in  $L_M$  such that either  $w \wr \bullet b_{q_1}$  or  $w \wr \bullet b_{q_k \oplus 1}$ , then we call  $w$  a *boundary wire*. The only nets which may be the boundary wires in  $L_M$  are  $w_{p_1}$  and  $w_{p_k}$ , and if so, they are routed in such a way that  $w_{p_1} \wr \bullet b_{q_1}$  and  $w_{p_k} \wr \bullet b_{q_k \oplus 1}$ . For example, if  $M$  is a maximal cluster and  $L_M$  is a Type DL (resp., Type DR) layout, then  $w_{p_1}$  (resp.,  $w_{p_k}$ ) is the boundary wire. Although there may be many local realizations of  $M$ , if we only consider the necessity of boundary wires, we can classify  $M$  into one of the following types:

1. *Type 0* if no boundary wire is necessary.
2. *Type x* if exactly one of the wires  $w_{p_1}$  and  $w_{p_k}$  must be a boundary wire but either one may be selected.
3. *Type l* if  $w_{p_1}$  must be a boundary wire while  $w_{p_k}$  need not be.
4. *Type r* if  $w_{p_k}$  must be a boundary wire while  $w_{p_1}$  need not be.
5. *Type lr* if both  $w_{p_1}$  and  $w_{p_k}$  must be boundary wires.

Note that if a component  $M$  is a trivial (resp., nontrivial) maximal cluster, it is of *Type 0*

(resp., *Type x*). Fig. 6 illustrates typical examples of the above five types and their symbolic representations. In a symbolic representation, an arrow indicates that any layout of the component needs a boundary wire on that side. Dotted arrows are used for *Type x* components only.

We now define two types of merging operations. The first one is the merging of two parallel components. Let  $M = \{n_{p_i} = (t_{p_i}, b_{q_j}) \mid 1 \leq i, j \leq k\}$  and  $M' = \{n_{p'_i} = (t_{p'_i}, b_{q'_j}) \mid 1 \leq i, j \leq m\}$  be components of  $N(\pi)$  such that  $[p_1, p_2, \dots, p_k, p'_1, p'_2, \dots, p'_m]$  and  $[q_1, q_2, \dots, q_k, q'_1, q'_2, \dots, q'_m]$  are icseqs. It is clear that  $M_{new} = M \cup M'$  is a component of  $N(\pi)$ . Suppose that both  $M$  and  $M'$  are locally realizable. If  $w_{p_k}$  and  $w_{p'_1}$  must be the boundary wires in any local realizations of  $M$  and  $M'$ , respectively, then both wires have to pass between  $b_{q_k}$  and  $b_{q'_1}$ . Therefore, the nets in  $M \cup M'$  can not be routed without violating the second constraint on a CPL1-solution, and hence  $\pi$  is not realizable. On the other hand, if at least one of  $w_{p_k}$  and  $w_{p'_1}$  need not be a boundary wire,  $M_{new}$  is clearly locally realizable. We call this merging of two parallel components a *P-merging* and call  $M$  and  $M'$  *P-merging operands*. Either  $M$  or  $M'$  may be the core of the merging.

The type of the new component  $M_{new}$  depends on the types of  $M$  and  $M'$ . For example, as shown in Fig. 7 (a), if  $M$  is of *Type l* and  $M'$  is of *Type x*, then  $M_{new}$  becomes a *Type l* component. Table 1 shows the component type of  $M_{new}$  for every possible combination of the component types of  $M$  and  $M'$ . There are four cases in which the merging is not possible due to a conflict of the boundary wires. We denote this by '*nil*' in the table and we say that the merging *fails*. Fig. 7 (b) shows such an example where  $M$  is of *Type r* and  $M'$  is of *Type l*.

The second merging operation is the merging of a component and two maximal clusters. Let  $M$  be a component defined above and assume that it is locally realizable. Let  $C_1 = \{n_{u_i} = (t_{u_i}, b_{v_{m-i+1}}) \mid 1 \leq i \leq m\}$  and  $C_2 = \{n_{u'_i} = (t_{u'_i}, b_{v'_{l-i+1}}) \mid 1 \leq i \leq l\}$  be two disjoint maximal clusters such that  $[u_1, u_2, \dots, u_m]$ ,  $[v_1, v_2, \dots, v_m]$ ,  $[u'_1, u'_2, \dots, u'_l]$  and  $[v'_1, v'_2, \dots, v'_l]$  are icseqs. Suppose that  $u_m \oplus 1 = p_1$ ,  $p_k \oplus 1 = u'_1$ ,  $v'_l \oplus 1 = q_1$  and  $q_k \oplus 1 = v_1$ .

**Lemma 2.** *If  $\pi$  is realizable,  $C_1 \cup C_2$  forms a Type I layout in any CPL1-solution  $L_{N(\pi)}$ .*

**Proof.** Let  $\bar{M} = N(\pi) - (M \cup C_1 \cup C_2)$ . If  $\bar{M}$  is an empty set or a maximal cluster,  $C_1 \cup \bar{M} \cup C_2$  is a cluster, which contradicts the maximality of  $C_1$  and  $C_2$ . Thus,  $\bar{M}$  has at least two maximal clusters.

For any subset  $M'$  of  $N(\pi)$ , let  $L_{M'}$  denote the layout of  $M'$  in  $L_{N(\pi)}$ . If  $L_{C_1}$  is a Type D layout, all wires in  $L_{\bar{M}}$  must pass between  $b_{q_k}$  and  $b_{v_1}$  (see Fig. 8). This is not allowed since  $\bar{M}$  has at least two nets. Thus,  $L_{C_1}$  is not a Type D layout. Similarly,  $L_{C_2}$  is not a Type D layout, either. If  $w_{u_m}$  does not pass between  $b_{v'_l}$  and  $b_{q_1}$ , then  $w_{u'_1} \wr b_{v_1}$ . This implies that  $w_{u_m} \wr b_{v'_l}$  since  $w_{u'_2}$  can not pass between  $b_{q_k}$  and  $b_{v_1}$ . On the other hand, if  $w_{u_m} \wr b_{q_1}$ , then  $w_{u'_1} \wr b_{v_2}$  since  $w_{u_{m-1}}$  can not pass between  $b_{v'_l}$  and  $b_{q_1}$ . It is easy to see that  $L_{C_1 \cup C_2}$  is a Type I layout in either case.  $\square$

Lemma 2 implies that  $M_{new} = C_1 \cup M \cup C_2$  is a component. We call this merging of a component and two maximal clusters an  $X$ -merging and call  $C_1, M$  and  $C_2$   $X$ -merging operands. In this case, the component  $M$  is the core of the merging operation. An example of an  $X$ -merging is shown in Fig. 9(a). In the figure, each maximal cluster is represented by a triangle with the mark 'c' in its inside.

If  $C_1$  and  $C_2$  do not satisfy that  $|C_2| - 1 \leq |C_1| \leq |C_2| + 1$ , they can not form a Type I

layout by Theorem 2, and thus there is no CPL1-solution for  $\pi$ . Furthermore, if a necessary boundary wire of  $M$  conflicts with any wire for  $C_1 \cup C_2$ , then  $\pi$  is not realizable. For example, if  $w_{p_1}$  must be a boundary wire and  $|C_1| = |C_2| + 1$ , then both  $w_{p_1}$  and  $w_{u_m}$  must pass between  $b_{v'_1}$  and  $b_{q_1}$ , which violates the second constraint on a CPL1-solution. Let  $d = |C_1| - |C_2|$ . The type of the new component  $M_{new}$  depends on the value of  $d$  and the component type of  $M$ . Table 2 shows the type of  $M_{new}$  for every possible case. For example, as shown in Fig. 9 (a), if  $M$  is a *Type 0* component and  $d = -1$ , then  $M_{new}$  becomes a *Type r* component. There are five cases for which an X-merging fails, which is denoted by ‘*nil*’ in the table. They are all due to a conflict between a boundary wire of  $M$  and some wire for  $C_1 \cup C_2$ . Fig. 9 (b) shows such an example where  $d = -1$  and  $M$  is a *Type r* component.

Lemma 2 holds even if  $C_1$  (resp.,  $C_2$ ) is trivial and  $C_2$  (resp.,  $C_1$ ) does not exist. If  $\pi$  is realizable,  $C_1$  (resp.,  $C_2$ ) forms a Type IM layout by itself in any CPL1-solution, and hence  $M_{new} = C_1 \cup M$  (resp.,  $M \cup C_2$ ) is a component. We also call this merging of a component and a maximal cluster an X-merging. Table 2 can still be used for this case by assuming that  $C_2 = \phi$  (resp.,  $C_1 = \phi$ ). Fig. 9 (c) shows an example of such a special X-merging operation.

## 5. Algorithm Description

If  $N(\pi)$  itself is a cluster, then  $\pi$  is realizable and finding a CPL1-solution is trivial. We arbitrarily select a net and route it by a direct wire. The remaining nets can be routed in such a way that the layout is either a Type DL or a Type DR layout. Thus, we assume that  $N(\pi)$  consists of three or more maximal clusters.

The algorithm consists of two main phases, the *merging phase* and *layout type assignment phase*. In the merging phase, the algorithm tests whether  $\pi$  is realizable or not by using the merging operations described in Section 4. If  $\pi$  is realizable, the algorithm constructs a CPL1-solution in the layout type assignment phase by determining the layout type of each maximal cluster.

## 5.1 Merging Phase

We show below the outline of the merging phase.

- Step 1. Partition  $N(\pi)$  into maximal clusters.
- Step 2. Execute the following substeps until a merging operation fails or there remain no merging operands.
- (a) Find P- or X-merging operands.
  - (b) Perform the merging operation.
- Step 3. if Step 2 results in a single component whose type is not *Type lr*  
then go to the layout type assignment phase ( $\pi$  is realizable)  
else terminate the algorithm ( $\pi$  is not realizable).

In Step 1, we construct a circular doubly linked list, called CLIST, which initially stores all the maximal clusters in the order of their appearances on the outer circle  $C_{out}$ . The contents of CLIST will be changed during the execution of Step 2.

In Step 2, the first merging operands can be found according to the following lemmas whose proofs are provided in the Appendix.

**Lemma 3.** *If a subset  $M$  of  $N(\pi)$  is locally realizable and consists of two or more maximal*

clusters, then it has at least one pair of parallel maximal clusters.  $\square$

**Lemma 4.** *Let  $C$  and  $C'$  be maximal clusters. If  $C \parallel C'$ , then both  $C$  and  $C'$  are components.*

$\square$

Since  $N(\pi)$  consists of three or more maximal clusters, if there are no parallel maximal clusters, then  $\pi$  is not realizable by Lemma 3, and thus the algorithm terminates. If  $N(\pi)$  has two parallel maximal clusters, they are components due to Lemma 4. When a maximal cluster is found to be a component, its component type is determined as *Type 0* (resp., *Type  $x$* ) if it is trivial (resp., nontrivial). In Step 2, the algorithm selects an arbitrary one among the components thus found as the initial core and begins the merging operations.

If merging operands including the current core are found, the algorithm tests whether they can be combined into a larger component which is locally realizable. This is done by checking the corresponding item in Table 1 or Table 2 which is identified by the component types (and  $d$  in the case of an  $X$ -merging) of the merging operands. If it is *nil*,  $\pi$  is not realizable owing to the argument in Section 4 and the algorithm terminates. Otherwise, the algorithm produces a new component  $M_{new}$  by performing the merging operation and gives an appropriate component type to it. Then  $M_{new}$  becomes a new core and the merging operands are replaced by  $M_{new}$  in CLIST. At this time, the algorithm may find another new component according to the following lemma whose proof is also given in the Appendix.

**Lemma 5.** *If there exists a maximal cluster  $C$  such that  $C \parallel M_{new}$  or  $M_{new} \parallel C$ , then  $C$  is a component.*  $\square$

If there are no merging operands including the current core for a  $P$ - or  $X$ -merging, the next component in the clockwise direction in CLIST becomes a new core.



**Lemma 6.** *If a component in CLIST becomes the core for the second time, then there remains no merging operands.*

**Proof.** If a component is selected as the core twice, all of the other components in CLIST have been the core before at least once. The lemma clearly follows from this fact.  $\square$

Merging operations are iteratively performed until a merging operation fails for some merging operands or there remain no merging operands in CLIST. Then, in Step 3, the algorithm tests the realizability of  $\pi$  based on the following theorem whose proof is shown in the Appendix.

**Theorem 3.**  *$\pi$  is realizable if and only if all maximal clusters in  $N(\pi)$  are merged into a single component which is not of Type *lr*.*  $\square$

During the merging phase, a directed graph called the *merging tree* is constructed. Initially, the graph has only isolated nodes which correspond to the maximal clusters. If a merging operation succeeds, the algorithm adds to the current graph a new node which corresponds to the resultant component and creates directed edges from this node to the nodes corresponding to the merging operands. Thus, if all the maximal clusters are eventually merged into a single component, the graph becomes a directed tree whose root corresponds to  $N(\pi)$ . And, if  $N(\pi)$  is not of *Type lr*, the merging tree will be used in the next phase to determine the layout types of the maximal clusters. We provide here a simple example. If the algorithm is applied to the instance of Fig. 2 and  $\{n_6, n_7\}$  is first selected as the core of the merging, then the final merging tree becomes as shown in Fig. 10, where the maximal clusters which have been recognized to be components are not marked ‘c’.

**Theorem 4.** *The time complexity of the merging phase is  $O(n)$ .*

**Proof.** It is easy to find all the maximal clusters in  $O(n)$  time. Since the number of maximal clusters is at most  $n$ , both the construction of CLIST and finding parallel maximal clusters can be done in  $O(n)$  time. In Step 2, if the algorithm can not find any merging operands including the current core, it selects a new core. By Lemma 6, such selections require only  $O(n)$  time in total. The total number of the merging operations performed is at most  $n - 1$  because each merging operation reduces the number of elements in CLIST by at least one. Furthermore, one execution of a P- or X-merging operation takes a constant time. Therefore, the merging phase can be completed in  $O(n)$  time.  $\square$

## 5.2 Layout Type Assignment Phase

Once the merging phase is successfully completed, the layout type of each maximal cluster is to be determined. The merging tree is now a rooted directed tree that has at most  $2n - 1$  nodes. Its leaves correspond to the maximal clusters in  $N(\pi)$  and its nonleaf nodes correspond to the components which have been found in the merging phase. For convenience, if a node in the tree corresponds to a subset  $M$  of  $N(\pi)$ , we call it node  $M$ .

In the layout type assignment phase, we first check whether the root  $N(\pi)$  is a *Type x* component, and if so, we change its type to either *Type l* or *Type r*. This selection is arbitrary. Then, starting from the root, the algorithm visits every nonleaf node  $M$  of the merging tree by depth-first search [1]. For each child  $M'$  of  $M$ , if it is a *Type x* component, then its type is modified to either *Type l* or *Type r* so as to be consistent with the component type of  $M$ . If  $M'$  is a maximal cluster, then its layout type is determined. These modifications and determinations are made in the following manner.

*Case 1.*  $M$  was created by a P-merging.

Let  $M_1$  and  $M_2$  be the components such that  $M = M_1 \cup M_2$  and  $M_1 \parallel M_2$ . Suppose that  $M_1$  is a *Type x* component. If the component type of  $M$  is *Type l* or *Type lr*, then we change the type of  $M_1$  to *Type l*, otherwise we change it to *Type r*. Similarly, if the component type of  $M_2$  is *Type x*, then it is changed to *Type l* or *Type r* depending on the component type of  $M$ . If  $M_1$  or  $M_2$  is a maximal cluster, then its layout type is determined by its component type. If it is a *Type 0* component, then it is a trivial cluster and the unique net is routed by a direct wire. Otherwise, it forms a Type DL or Type DR layout depending on whether it is a *Type l* or *Type r* component. We show a simple example in Fig. 11. In this case, the type of  $M_2$  is changed from *Type x* to *Type r* since  $M$  is a *Type r* component. Furthermore, since  $M_2$  is a maximal cluster, its layout type is determined as Type DR.

*Case 2.*  $M$  was created by an X-merging.

Let  $C_1$  and  $C_2$  be the maximal clusters and  $M'$  be the component such that  $M = C_1 \cup M' \cup C_2$  and that the terminals of  $C_1$ ,  $M'$  and  $C_2$  appear on the outer circle  $C_{out}$  in this order in the clockwise direction. Let  $d = |C_1| - |C_2|$ . If  $d = 0$ , then the layout type of  $C_1 \cup C_2$  is determined as Type IL or Type IR depending on whether the component type of  $M$  is *Type l* or *Type r*. If  $d = -1$  or  $d = 1$ , then  $C_1$  and  $C_2$  form a Type IM layout. In particular, if  $C_1 = \phi$  (resp.,  $C_2 = \phi$ ), then  $C_2$  (resp.,  $C_1$ ) forms a Type IM layout by itself. Suppose that  $M'$  is a *Type x* component. Its layout type is changed to the same as that of  $M$  if  $d = 0$ . And, if  $d = 1$  (resp.,  $d = -1$ ), then it is changed to *Type r* (resp., *Type l*). For example, in Fig. 12, the layout type of  $C_1 \cup C_2$  is determined as Type IM and the component type of  $M'$  is changed to *Type l*.

Since the number of nodes in the merging tree is  $O(n)$  and the algorithm spends a

constant time at each node, the time complexity of the layout assignment phase is  $O(n)$ .

Since the merging phase takes  $O(n)$  time by Theorem 4, we obtain our main theorem.

**Theorem 5.** *Given a permutation  $\pi$  of  $1, 2, \dots, n$ , a CPL1-solution for  $\pi$  can be found, if one exists, in  $O(n)$  time.  $\square$*

## 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] S. Choe, T. Kashiwabara, and T. Fujisawa, "A Permutation Layout with Limited Between-pin Congestion," *Papers of Technical Group on Circuit and Systems of the IECE of Japan*, CAS 83-74, 1983.
- [3] S. Choe, T. Kashiwabara, and T. Fujisawa, "Restricted Permutation Layout," *Trans. of the IECE of Japan*, vol. J68E, pp. 269-276, 1985.
- [4] M. Cutler and Y. Shiloach, "Permutation Layout," *Networks*, vol. 8, pp. 253-278, 1978.
- [5] T. Kashiwabara, K. Itagaki, S. Masuda, and T. Fujisawa, "On Certain Permutation Layout," *Proc. 1985 IEEE Int. Symp. on Circuits and Systems, Kyoto, Japan*, June 1985, pp. 1043-1046.
- [6] T. Ozawa, "A Routing Procedure for an IC Module with Many Pins - A Solution to a Circular Permutation Layout Problem," *Networks*, vol. 15, pp. 33-48, 1985.

[7] I. Shirakawa, “Some Comments on Permutation Layout,” *Networks*, vol. 10, pp. 179–182, 1980.

[8] S. Tsukiyama and E. S. Kuh, “Double-Row Planar Routing and Permutation Layout,” *Networks*, vol. 12, pp. 287–316, 1982.

## Appendix. Proofs of Lemmas 3,4 and 5 and Theorem 3

Let  $M = \{n_{p_i} = (t_{p_i}, b_{q_j}) \mid 1 \leq i, j \leq k\}$  be a locally realizable subset of  $N(\pi)$  such that  $[p_1, p_2, \dots, p_k]$  and  $[q_1, q_2, \dots, q_k]$  are icseqs. Suppose that  $M$  consists of two or more maximal clusters. We denote by  $cn_M$  the number of maximal clusters contained in  $M$ . Let  $L_M$  be any local realization of  $M$ , and for any subset  $M'$  of  $M$ , let  $L_{M'}$  be the layout of  $M'$  in  $L_M$ . We start with the following lemma.

**Lemma A-1.** *If  $L_M$  contains a Type I layout, then  $M$  has a proper subset which is locally realizable and consists of two or more maximal clusters.*

**Proof.** Suppose that two maximal clusters  $C = \{n_{p_i} \mid h \leq i \leq l\}$  and  $C' = \{n_{p_i} \mid m \leq i \leq s, l < m\}$  form a Type I layout. Let  $M' = \{n_{p_i} \mid l+1 \leq i \leq m-1\}$ . It is clear that  $M'$  is locally realizable. Furthermore,  $cn_{M'} \geq 2$ ; otherwise  $C \cup M' \cup C'$  would be a single cluster, a contradiction.

Next, suppose that  $M$  contains a trivial maximal cluster  $C = \{n_{p_h}\}$  which forms a Type IM layout by itself. Let  $m$  be the integer such that  $q_m = \pi(p_h)$ . In the following, we consider the case in which  $w_{p_h} \perp \bullet b_{q_l}$  for some integer  $l < m$ . The other cases can be treated similarly. Let  $B_1 = \{b_{q_l}, b_{q_{l+1}}, \dots, b_{q_{m-1}}\}$ ,  $B_2 = \{b_{q_1}, b_{q_2}, \dots, b_{q_k}\} - \{b_{q_m}\} - B_1$ ,  $T_1 = \{t_{p_{h+1}}, t_{p_{h+2}}, \dots, t_{p_{h+m-l}}\}$  and  $T_2 = \{t_{p_1}, t_{p_2}, \dots, t_{p_k}\} - \{t_{p_h}\} - T_1$ . Note that  $|B_1| = |T_1| = m - l \geq 1$ . Assume that  $M$  contains a net  $(t_{p_s}, b_{\pi(p_s)})$  such that  $t_{p_s} \in T_2$  and

$b_{\pi(p_s)} \in B_1$ . See Fig. A-1. Let  $M' = \{n_{p_i} \mid t_{p_i} \in T_1\}$ . Since  $|B_1 - \{b_{\pi(p_s)}\}| < |T_1|$ ,  $L_{M'}$  has a wire, say  $w'$ , which connects a terminal in  $T_1$  and a terminal in  $B_2$ . However, because of  $w_{p_h}$  and  $w_{p_s}$ ,  $w'$  has to cross  $C_{in}$  at least twice, which violates the third constraint on a CPL1-solution. Therefore, every net in  $M'$  contains a terminal in  $B_1$ , which implies that  $M'$  is locally realizable. Furthermore,  $cn_{M'} \geq 2$ ; otherwise  $C \cup M'$  would be a single cluster.  $\square$

**Proof of Lemma 3.** We prove the lemma by induction on the value of  $cn_M$ . If  $M$  has only two maximal clusters, then they must be parallel clusters; otherwise, their union would be a single cluster. Assume that, for an integer  $j \geq 2$ , if  $2 \leq cn_M \leq j$ , then  $M$  has at least one pair of parallel maximal clusters. Suppose that  $cn_M = j + 1$ . If  $L_M$  contains no Type I layout, clearly  $M$  has parallel maximal clusters. On the other hand, if  $L_M$  has a Type I layout, then  $M$  has a locally realizable subset  $M'$  such that  $2 \leq cn_{M'} \leq j$  by Lemma A-1. Thus,  $M$  has at least one pair of parallel maximal clusters by the induction hypothesis.  $\square$

**Proof of Lemma 4.** If  $C$  forms a Type I layout with a maximal cluster  $C'' \neq C'$ , then the nets in  $C'$  can not be routed. Furthermore, if  $C$  and  $C'$  form a Type I layout, then no nets in  $N(\pi) - (C \cup C')$  can be routed. Assume that  $C$  is trivial and it forms a Type IM layout by itself in any CPL1-solution  $L_{N(\pi)}$ . Let  $C = \{(t_{p_1}, b_{q_1})\}$ , and let  $[p_1, p_2, \dots, p_n]$  and  $[q_1, q_2, \dots, q_n]$  be two icseqs. Suppose that  $w_{p_1} \wr b_{q_j}$  in  $L_{N(\pi)}$  for some integer  $j$ . We define four sets of terminals  $B_1 = \{b_{q_j}, b_{q_{j+1}}, \dots, b_{q_n}\}$ ,  $B_2 = \{b_{q_2}, b_{q_3}, \dots, b_{q_{j-1}}\}$ ,  $T_1 = \{t_{p_2}, t_{p_3}, \dots, t_{p_{n-j+2}}\}$ , and  $T_2 = \{t_{p_{n-j+3}}, t_{p_{n-j+4}}, \dots, t_{p_n}\}$ . By the same argument as in the proof of Lemma A-1, every terminal in  $T_1$  (resp.,  $T_2$ ) is connected to a terminal in  $B_1$  (resp.,  $B_2$ ). Since  $n_{p_2}$  and the net containing  $b_{q_2}$  belong to  $C'$ , every net containing a terminal in  $B_2$  belongs to  $C'$ . This implies that  $n_{p_n} \in C'$  and hence  $C' = N(\pi) - C$ , which contradicts the assumption that

$N(\pi)$  consists of more than two maximal clusters. Consequently,  $C$  forms a Type D layout in any CPL1-solution, and hence it is a component. Similarly,  $C'$  is a component.  $\square$

**Proof of Lemma 5.** Assume that  $\pi$  is realizable and let  $L_{N(\pi)}$  be any CPL1-solution for it. Using the same argument as in the proof of Lemma 4, we can show that  $C$  forms a Type D layout in  $L_{N(\pi)}$  if (i)  $C$  is not trivial or (ii) there exists a component  $M$  such that  $C \parallel M$  or  $M \parallel C$  and  $M \cup C \neq N(\pi)$ . If  $M_{new}$  is the result of a P-merging operation, then it has a subset that satisfies the condition (ii). Suppose that  $C$  is trivial, that  $C \cup M_{new} = N(\pi)$  and that  $M_{new}$  was created by an X-merging operation. It is clear that the net in  $C$  can not pass between the terminals of the nets in  $M_{new}$ , and thus it is routed by a direct wire in  $L_{N(\pi)}$ . This completes the proof.  $\square$

**Proof of Theorem 3.** It is easy to see that if all the maximal clusters can be merged into one component which is not of *Type lr*, then  $\pi$  is realizable. Assume that the algorithm fails in making such a component. If all the maximal cluster are merged into a *Type lr* component,  $\pi$  is not realizable since the two boundary wires of the component have to pass between the same two adjacent terminals. Suppose that  $\pi$  is realizable and the algorithm terminates when CLIST has two or more elements. Let  $L_{N(\pi)}$  be any CPL1-solution for  $\pi$ . CLIST at the termination of the algorithm has a maximal cluster which forms a Type I layout in  $L_{N(\pi)}$ ; otherwise any two consecutive elements in the list would constitute P-merging operands. Let  $Y$  be the set of all such maximal clusters and  $S$  be the set of indirect wires  $w_i$  in  $L_{N(\pi)}$  such that  $n_i \in \cup_{C \in Y} C$ .

Each wire  $w_i \in S$  divides the sets  $\{b_1, b_2, \dots, b_n\} - \{b_{\pi(i)}\}$  into two subsets (like  $B_1$  and  $B_2$  in the proof of Lemma 4). We define the *width* of  $w_i$ , denoted by  $wd_i$ , as the smaller

cardinality of such two sets. Let  $w_l$  be a wire in  $S$  whose width is the minimum and  $k$  be the integer such that  $w_l \bullet b_k$  in  $L_{N(\pi)}$ . Let  $M = \{(t_{p_i}, b_{q_j}) | 1 \leq i, j \leq h\}$ , where  $[k = q_1, q_2, \dots, q_h]$  is an **icseq**. Without loss of generality, we assume that  $h = wd_l$ . See Fig. A-2. Let  $C$  be the maximal cluster which contains  $n_l$ . It is clear from the definition of  $w_l$  that  $M \cap C = \phi$ . Furthermore,  $M$  is not a cluster; otherwise  $M \cup C$  would be a single cluster. These imply that if  $M$  is an element of CLIST at the termination of the algorithm, the list has **X**-merging operands. Thus, CLIST at that time has at least two elements whose union is equal to  $M$ . Since those elements contain no **P**-merging operands, one of them is a maximal cluster which forms a Type I layout in  $L_{N(\pi)}$ . This implies that  $S$  has a wire whose width is less than  $wd_l$ , which contradicts the definition of  $w_l$ . Therefore, if the algorithm terminates when CLIST has two or more elements, then  $\pi$  is not realizable.  $\square$



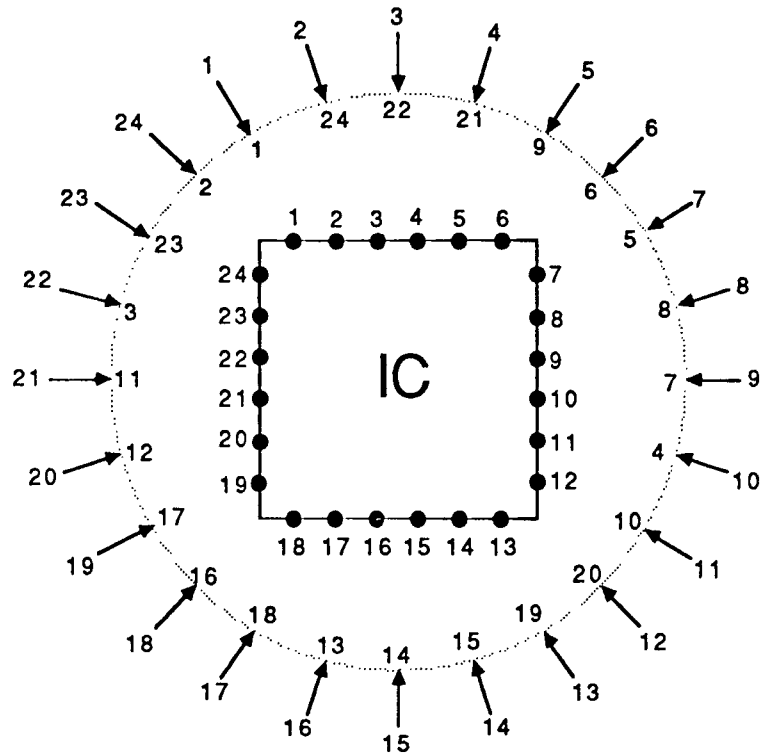


Fig. 1. Application of circular permutation layout in a single layer routing around an IC (module).

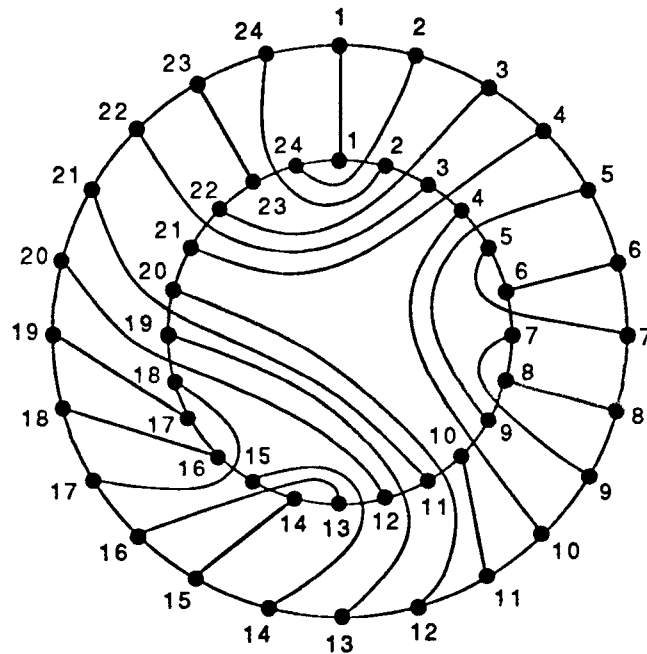
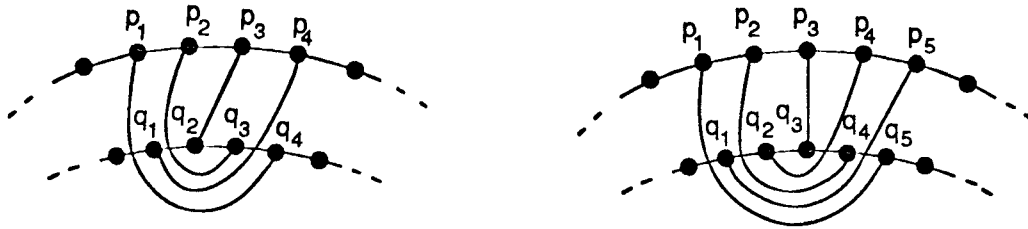
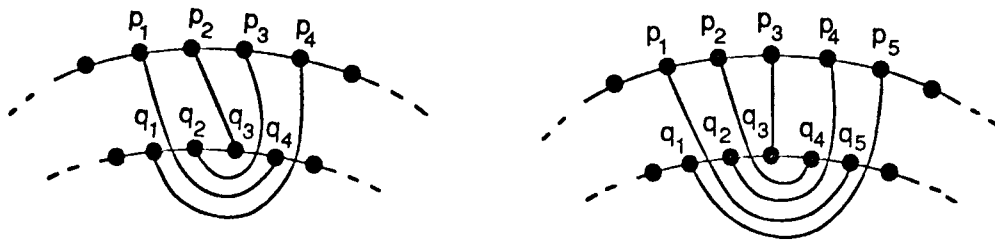


Fig. 2. A CPL1-solution for the permutation  $\pi = (1\ 24\ 22\ 21\ 9\ 6\ 5\ 8\ 7\ 4\ 10\ 20\ 19\ 15\ 14\ 13\ 18\ 16\ 17\ 12\ 11\ 3\ 23\ 2)$ .

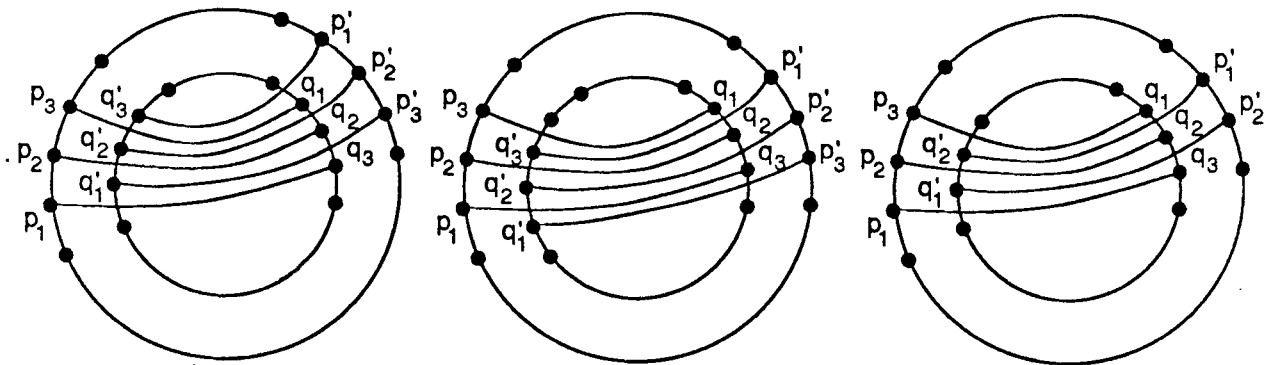


(a) Type DL layouts.



(b) Type DR layouts.

Fig. 3. Type D layouts of clusters.



(a) A Type IL layout.

(b) A Type IR layout.

(c) A Type IM layout.

Fig. 4. Type I layouts of clusters.

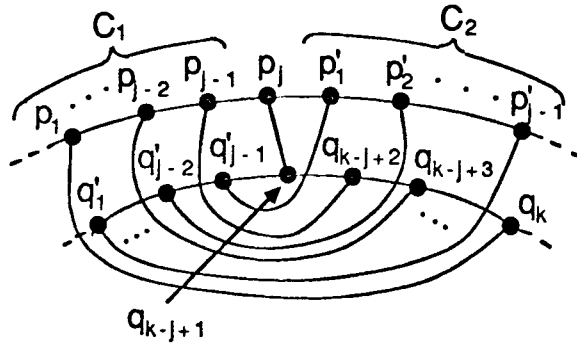
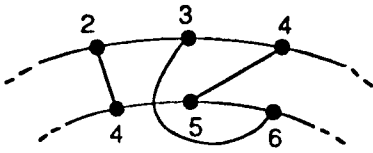
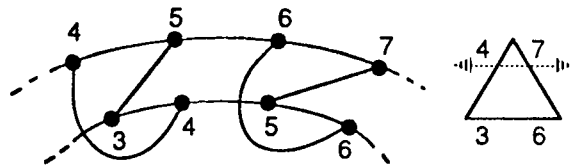


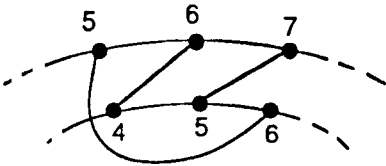
Fig. 5. An illustration for the proof of Theorem 1.



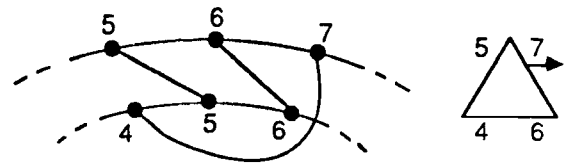
(a) A *Type 0* component.



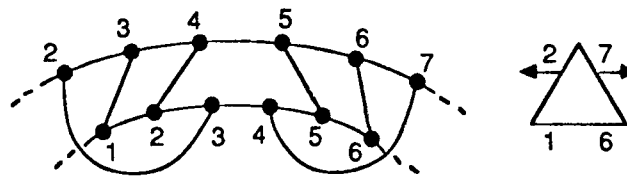
(b) A *Type x* component.



(c) A *Type l* component.

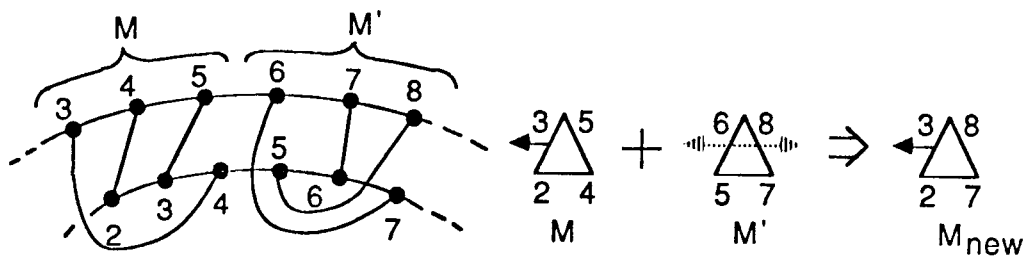


(d) A *Type r* component.

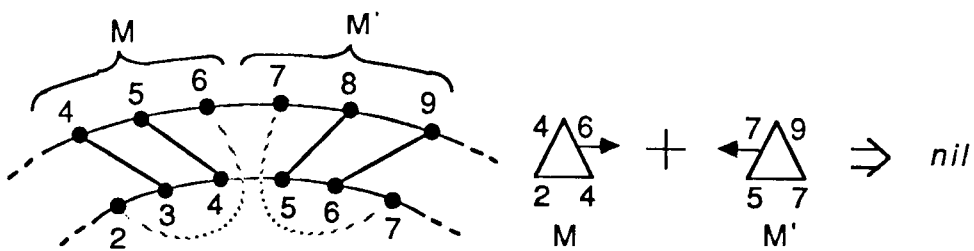


(e) A *Type lr* component.

Fig. 6. Typical components of the five types.




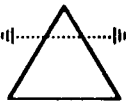
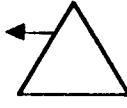
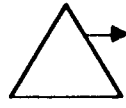
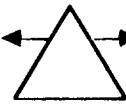




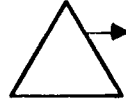

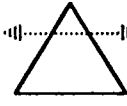

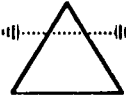
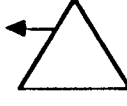

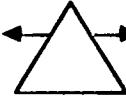
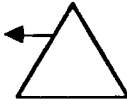
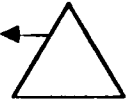
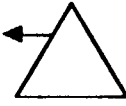
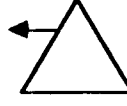
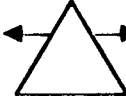
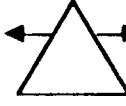
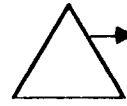
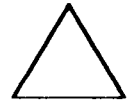
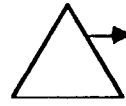

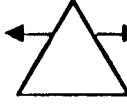
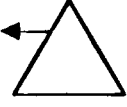
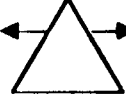
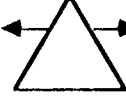
(a) A successful P-merging.



(b) A failed P-merging.

Fig. 7. Examples of P-merging operations.

Table 1. P-merging table.

$M' \backslash M$					
					
					
					
			<i>nil</i>		<i>nil</i>
			<i>nil</i>		<i>nil</i>

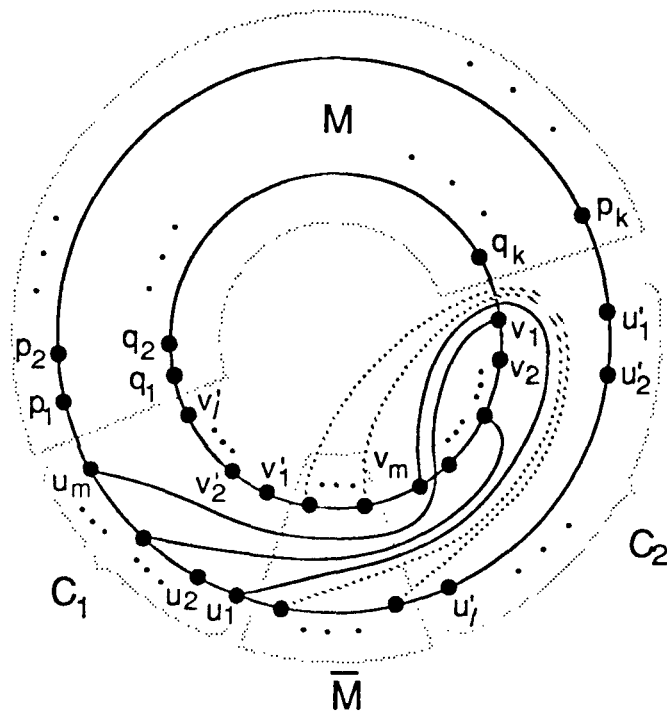
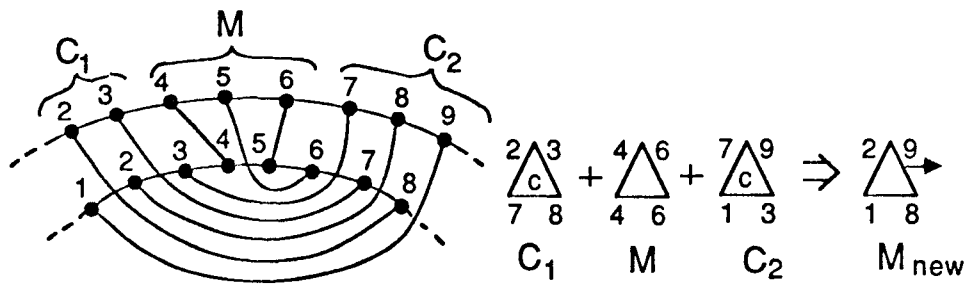
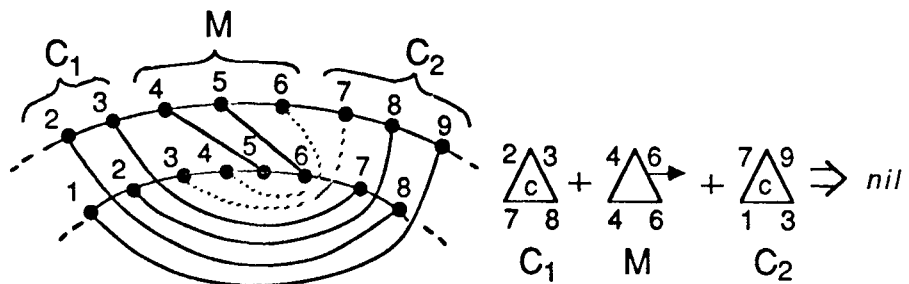


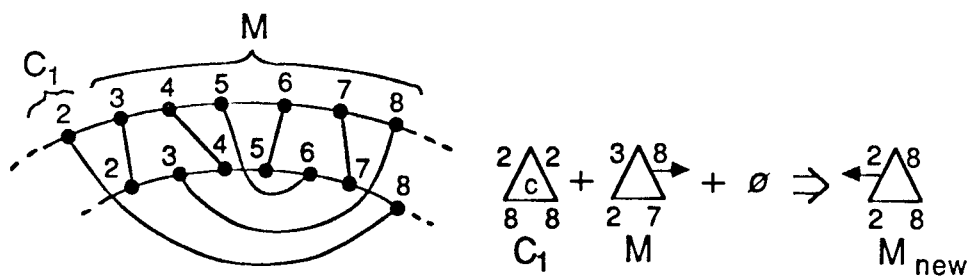
Fig. 8. An illustration for the proof of Lemma 2.



(a) A successful X-merging.




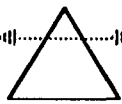
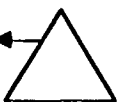
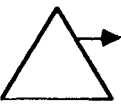
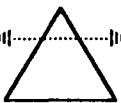
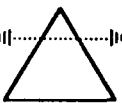
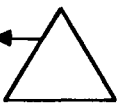
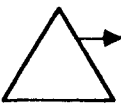




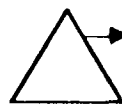
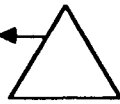

(b) A failed X-merging.



(c) A successful X-merging with  $C_2 = \phi$ .

Fig. 9. Examples of X-merging operations.

Table 2. X-merging table.

M \ d	0	1	-1
			
			
		<i>nil</i>	
			<i>nil</i>
	<i>nil</i>	<i>nil</i>	<i>nil</i>



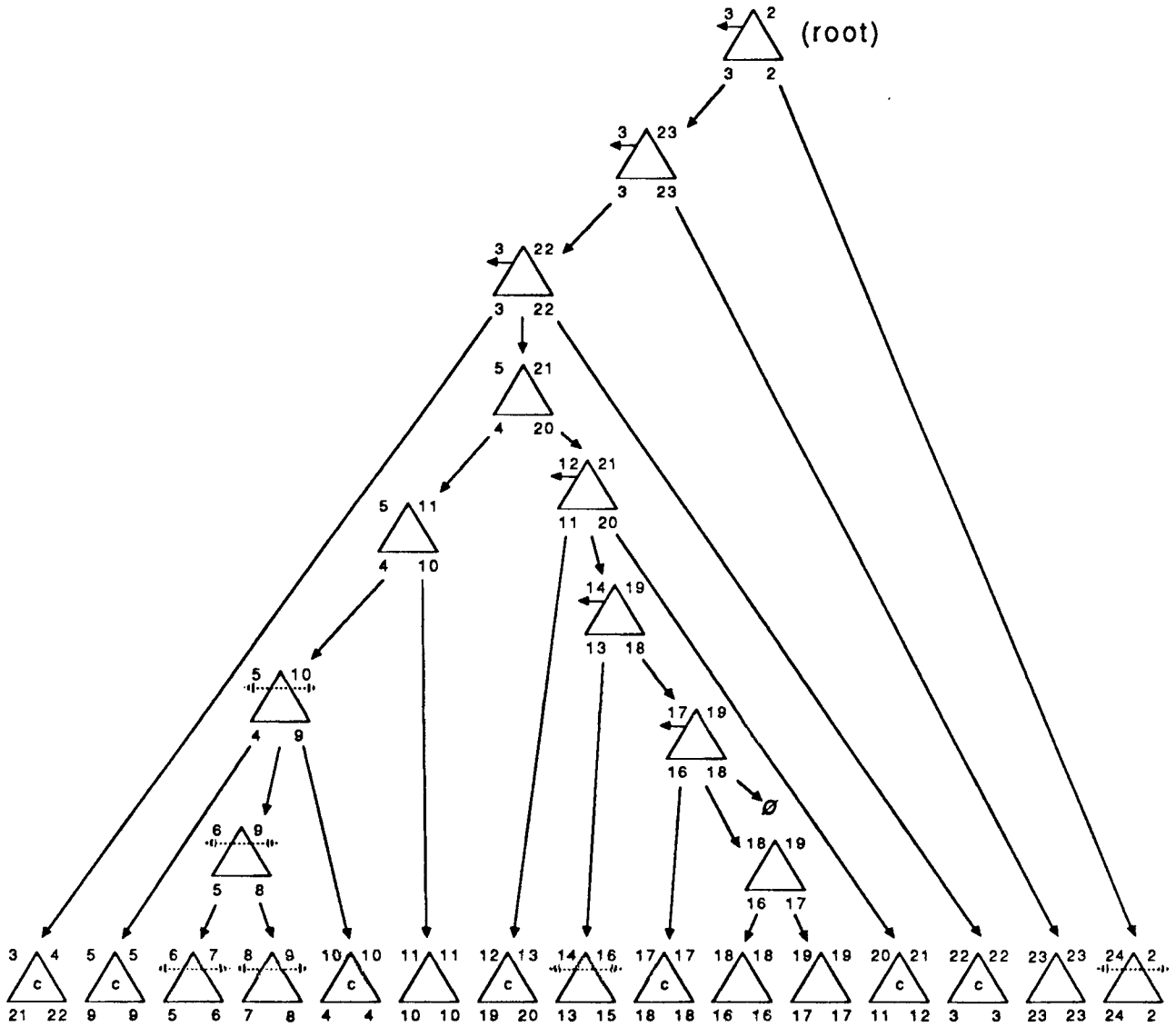


Fig. 10. Result of merging operations for the instance of Fig. 2.

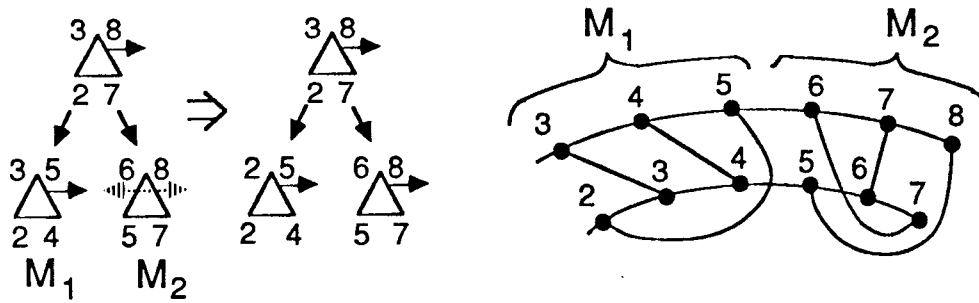


Fig. 11. An example of layout type assignment for P-merging.

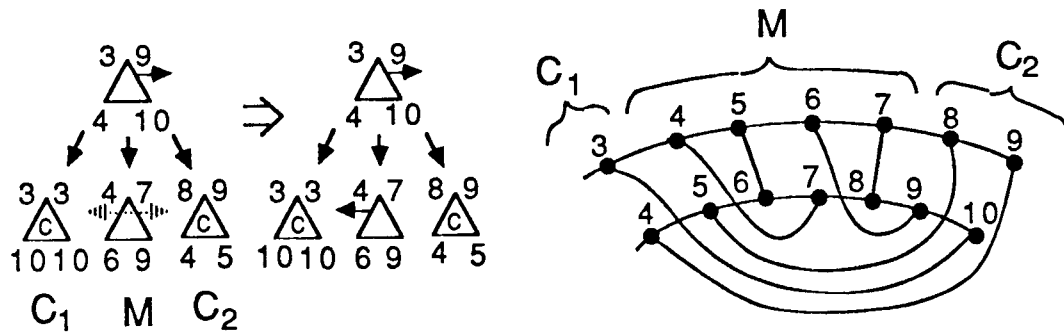


Fig. 12. An example of layout type assignment for X-merging.

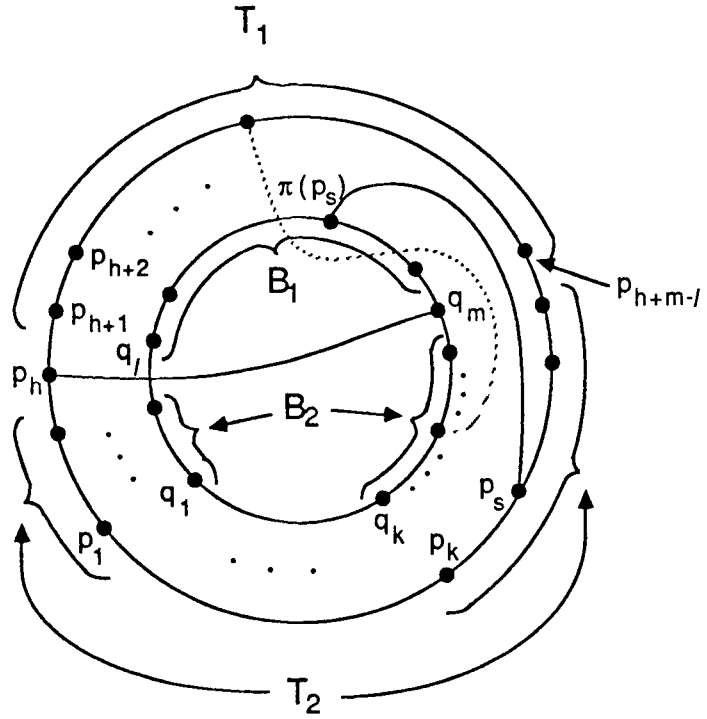


Fig. A-1. An illustration for the proof of Lemma A-1.

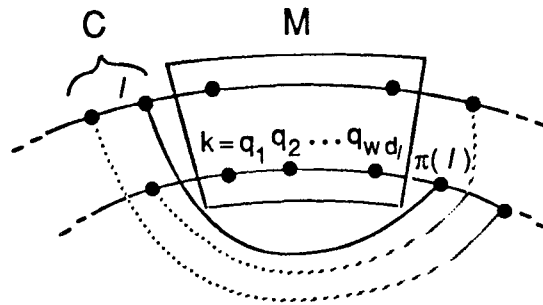


Fig. A-2. An illustration for the proof of Theorem 3.