OPTIMAL CHANNEL PIN ASSIGNMENT

Yang Cai and Martin D. F. Wong

Department of Computer Sciences
University of Texas at Austin
Austin, Texas 78712-1188

TR-90-25 August 1990

Optimal Channel Pin Assignment *

Yang Cai and D.F. Wong
Department of Computer Sciences
University of Texas at Austin
Austin, Texas 78712

August 6, 1990

Abstract

We study in this paper the channel pin assignment problem subject to position constraints, order constraints, and separation constraints. The problem is to assign two sets of terminals to the top and the bottom of a channel to minimize channel density. The position constraints are given by associating with each terminal a set of allowable positions. The order constraints specify the relative orderings of the terminals on the top and the bottom of the channel by a pair of partially ordered sets. The separation constraints require the distance between each pair of consecutive terminals to be within a certain range. We show that the problem is NP-hard in general and present polynomial time optimal algorithms for an important special case in which the order constraints are specified by two linearly ordered sets (*i.e.*, relative orderings of the terminals on the top and the bottom of the channel are completely fixed). We introduce the problem of channel routing with movable modules and show that it is a special case of our problem and hence can be solved optimally. We also discuss how our algorithms can be incorporated into standard-cell and building-block layout design systems. Experimental results indicate that substantial reduction in channel density can be obtained by allowing movable terminals.

1 Introduction

In the physical design of VLSI circuits, a significant portion of the chip area is used for channel routing. Ordinary channel routers assume that all the terminals (pins) on each side of the channel are fixed [4, 5, 6, 9, 16]. However, it is typical in practice that after the placement phase, the positions of the terminals on the boundaries of the modules are not completely fixed, and they still have some degree of freedom to move at the beginning of the routing phase. The existence of movable terminals can be used to our advantage to make the subsequent routing task easier and hence obtain reduction in routing area. Figure 1 shows that by allowing movable terminals, a significant reduction in channel density can be obtained.

The problem of minimizing the width required to route a channel has been shown to be NP-hard [15]. On the other hand, channel density is a fairly accurate measure for the minimum width that can be achieved, and can be computed fast and easily. Although channel density is not the only lower bound

*This work was partially supported by the National Science Foundation under grant MIP-8909586, by the Texas Advanced Research Program under grant 4096, and by an IBM Faculty Development Award.

(a) Before moving: $d = 3$ (b) After moving: $d = 1$

Figure 1: Reducing channel density by moving terminals

for channel width [2, 3], in most practical situations it is a good measure for estimating the width during placement and global routing phases [13]. The situation in which other lower bounds exceed channel density are extremely rare and easily recognizable, they can be dealt with separately. In fact, many existing channel routers achieve widths that are usually within one or two tracks of the corresponding channel densities [4, 5, 6, 16].

The *Channel Pin Assignment* (CPA) problem is the problem of assigning positions for the terminals, subject to all possible constraints imposed by the design rules and the designs of the previous phases, so as to minimize the density of the channel. It has various forms depending on how the pin assignment constraints are specified. Some special forms of this problem have been investigated by several other researchers. In [8], the authors considered the channel routing problem with movable terminals. They assumed that all nets are two-terminal nets, having one terminal on each side of the channel, and the terminals are movable but their relative orderings on both sides of the channel are to be preserved. Under the assumption that the length of the channel is unbounded, they were able to develop an optimal algorithm for minimizing the width of the channel. In [10], the channel offsetting problem was considered. This problem is identical to the ordinary channel routing problem except that the sides of the channel can be laterally shifted. An offset of the sides of the channel is to be computed which minimizes the density of the channel. This problem was generalized to the channel rotation problem in [1]. In this case, one side of the channel is allowed to be circularly shifted and a rotation minimizing the density of the channel is to be found. Optimal algorithms are given in [12] for the CPA problem where there are no pin assignment constraints. Finally, [11] considered the permutation channel routing problem, in which some sets of terminals are allowed to be permuted. Heuristic algorithms were proposed to minimize channel width.

In this paper we consider the CPA problem in which the assignment of terminals is subjected to *position constraints*, *order constraints*, and *separation constraints*. The position constraints are given by associating with each terminal a set of allowable positions. The order constraints specify the relative

orderings of the terminals by a pair of partially ordered sets. The separation constraints require the distance between each pair of consecutive terminals to be within a certain range. We first consider the CPA problem with position and order constraints. We show that the problem is NP-hard in general, even if all nets are two-terminal nets having one terminal on each side of the channel. For an important case where the relative orderings of the terminals are completely fixed, we are able to develop a polynomial time optimal algorithm to minimize channel density. We then extend our algorithm to optimally solve the problem in the case where there are also separation constraints between consecutive terminals in polynomial time. Our algorithms are general enough that the channel offsetting problem [10], the channel routing problem with movable terminals [8], and a generalization of them, the *channel routing problem with movable modules* (Section 5.3) can all be solved by them as special cases. Experimental results show that by allowing movable terminals, significant reduction in channel density (up to 25%) can be obtained. We believe our algorithms can be incorporated into placement/routing systems to further reduce routing area.

We formulate the CPA problem with position and order constraints in Section 2, and present the NP-hardness results in Section 3. An optimal algorithm for the case where the relative orderings of the terminals are completely fixed is presented in Section 4, and its extensions to handle separation constraints are discussed in Section 5. In Section 6, we discuss how our algorithms can be applied to both standard-cell and building-block layout designs. Section 7 reports some of our experimental results. Finally, we conclude the paper in Section 8 with some general remarks.

2 Problem Formulation

In this section we give a mathematical formulation for the CPA problem with position and order constraints. We assume there is a virtual grid superimposed on the channel, and terminals can only be placed at the grid points on the top and bottom of the channel.

Let L be the length of the channel, and N be the set of nets to be routed. The set of terminals on the top of the channel is denoted by TOP, and the set of terminals on the bottom of the channel is denoted by $BOTTOM$. There are two types of constraints imposed on the assignment of terminals, namely, the *order constraints* and the *position constraints*. The order constraints are modeled by two partially ordered sets (posets for short), R_T and R_B, on TOP and $BOTTOM$, respectively. The position constraints are modeled by two sets, T and B, of subsets of all possible positions on the top and the bottom of the channel, respectively. Formally, an instance of the CPA problem is an 8-tuple $\Phi = (L, TOP, BOTTOM, N, T, B, R_T, R_B)$, where

- L is the length of the channel;
- $TOP = \{t_1, t_2, \ldots, t_p\}$, $p \leq L$, is the set of terminals on the top of the channel;
- $BOTTOM = \{b_1, b_2, \ldots, b_q\}$, $q \leq L$, is the set of terminals on the bottom of the channel;
- $N = \{N_1, N_2, \ldots, N_n\}$ is a partition of $TOP \cup BOTTOM$ giving the net list of the channel, where N_i is the set of terminals belonging to net i, $1 \leq i \leq n$;
- $T = \{T_k \subseteq \{1, 2, \ldots, L\} : 1 \leq k \leq p\}$ specifies the position constraints for the terminals on the top of the channel, *i.e.*, t_k can only be assigned to positions in T_k, $1 \leq k \leq p$;
- $B = \{B_k \subseteq \{1, 2, \ldots, L\} : 1 \leq k \leq q\}$ specifies the position constraints for the terminals on the bottom of the channel, *i.e.*, b_k can only be assigned to positions in B_k, $1 \leq k \leq q$;
- $R_T = (TOP, <_T)$ is a poset defining the order constraints for the terminals on the top of the channel, *i.e.*, if $t_i <_T t_j$, then t_i must be assigned to the left of t_j;
- $R_B = (BOTTOM, <_B)$ is a poset defining the order constraints for the terminals on the bottom of the channel, *i.e.*, if $b_i <_B b_j$, then b_i must be assigned to the left of b_j.

A *solution* to Φ is an ordered pair of functions $\pi = (f, g)$, where $f : TOP \mapsto \{1, 2, \ldots, L\}$, and $g : BOTTOM \mapsto \{1, 2, \ldots, L\}$, such that

1. $f(t_k) \in T_k$, $\forall\ k, 1 \leq k \leq p$;
2. $g(b_k) \in B_k$, $\forall\ k, 1 \leq k \leq q$;
3. $t_i <_T t_j \Rightarrow f(t_i) < f(t_j), \forall\ t_i, t_j \in TOP$;
4. $b_i <_B b_j \Rightarrow g(b_i) < g(b_j), \forall\ b_i, b_j \in BOTTOM$;
5. both f and g are injective functions.

That is, a solution to Φ is an assignment of the terminals to the top and the bottom of the channel satisfying the given constraints. Intuitively, $f(t_i) = k$ means that t_i is assigned to the kth grid point (from the left) on the top of the channel (*i.e.*, the top endpoint of column k). Similarly, $g(b_j) = k$ means that b_j is assigned to the kth grid point on the bottom of the channel. Conditions 1 and 2 require the position constraints to be satisfied, and conditions 3 and 4 require the order constraints be satisfied. Condition 5 ensures that no two terminals are assigned to the same grid point. (Recall that a function f is injective if and only if for any x, y in its domain, $x \neq y \Rightarrow f(x) \neq f(y)$.)

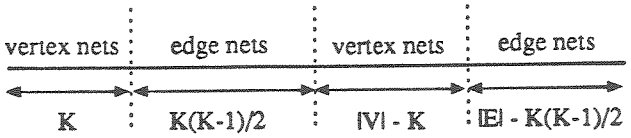


Figure 2: Illustration of the proof of Lemma 3.1

An instance of the CPA problem is *feasible* if and only if it has at least one solution. Note that a solution to an instance of the CPA problem defines an instance of the ordinary channel routing problem. The channel density of this instance of the ordinary channel routing problem is said to be the *density* of that solution. A solution is *optimal* if and only if it has minimum density.

3 NP-hardness Results

In this section we show that the CPA problem with position and order constraints is NP-hard in general, and hence very unlikely to be solved efficiently. Readers who are not familiar with the theory of NP-completeness are referred to reference [7].

Lemma 3.1 *Given any instance Φ of the CPA problem, it is NP-complete to determine whether Φ is feasible, even if all nets are two-terminal nets having one terminal on each side of the channel.*

Proof: The problem is in NP because we can guess an ordered pair of functions $\pi = (f, g)$ and test if it is a solution to Φ in polynomial time. To show that it is NP-complete, we construct a polynomial time transformation from the CLIQUE problem. (See [7] for a definition of the CLIQUE problem.) Given a graph $G = (V, E)$ and a positive integer $K \leq |V|$ (without loss of generality, we may assume $|E| \geq K(K-1)/2$), we construct an instance $\Phi = (L, TOP, BOTTOM, N, T, B, R_T, R_B)$ of the CPA problem, with

$$
\begin{aligned}
L &= |N| \\
&= |V| + |E|; \\
N &= \{N_v = \{t_v, b_v\} : v \in V\} \cup \{N_e = \{t_e, b_e\} : e \in E\}; \\
R_T &= (TOP, \phi), \textit{ i.e.}, \text{ no order constraints on } TOP; \\
R_B &= (BOTTOM, \{(v, e) : e \text{ is incident on } v\}), \textit{ i.e.}, \text{ the endpoints of an edge} \\
&\quad \text{precede the edge in } BOTTOM; \\
TOP &= \{t_x : x \in V \cup E\};
\end{aligned}
$$

$$
\begin{aligned}
BOTTOM &= \{b_y : y \in V \cup E\};\\
T_x &= \{1, 2, \ldots, L\},\ \forall\, x \in V \cup E,\ \textit{i.e.}, \text{ no position constraints on } TOP;\\
B_v &= \{1, 2, \ldots, K\}\\
&\quad \cup \{K(K+1)/2+1, K(K+1)/2+2, \ldots, K(K-1)/2+|V|\},\ \forall\, v \in V;\\
B_e &= \{1, 2, \ldots, L\} - B_v,\ \forall\, e \in E, v \in V.
\end{aligned}
$$

Intuitively, there are no constraints on the assignment of terminals on the top of the channel. The bottom of the channel is divided into four parts, as illustrated in Figure 2. The terminals in $BOTTOM$ which correspond to the vertices of G must be assigned to the first and third parts. The terminals in $BOTTOM$ which correspond to the edges of G must be assigned to the second and fourth parts. Furthermore, for any $e = \{u, v\} \in E$, we have $b_u <_B b_e$ and $b_v <_B b_e$, *i.e.*, both b_u and b_v must be assigned to the left of b_e. We claim that G has a clique of size $\geq K$ if and only if Φ so constructed is feasible. This is because exactly $K(K-1)/2$ terminals in $BOTTOM$ corresponding to the edges of G must be assigned to the second part of the bottom of the channel in any solution to Φ (because of th position constraints), and this is possible if and only if G has a clique of size K, such that the terminals in $BOTTOM$ corresponding to the vertices of the clique are assigned to the first part of the bottom of the channel (because of the order constraints). The lemma follows because the transformation can be carried out in polynomial time. □

The following theorem now follows directly from Lemma 3.1 and the definition of NP-hardness [7].

Theorem 3.2 *The problem of computing an optimal solution to a given instance of the CPA problem is NP-hard, even if all nets are two-terminal nets having one terminal on each side of the channel.*

In fact, we can show a stronger result that the CPA problem remains NP-hard even if the given instance of the CPA problem is known to be feasible (Theorem 3.4). To prove this, we first show in Lemma 3.3 that the following decision version of the problem is NP-complete. Theorem 3.4 then follows directly from this lemma.

CPA-d
INSTANCE: A feasible instance Φ of the CPA problem, an integer $d \geq 0$.
QUESTION: Is there a solution to Φ with density $\leq d$?

Lemma 3.3 *The CPA-d problem is NP-complete for any fixed integer $d \geq 0$, even if all nets are two-terminal nets having one terminal on each side of the channel.*

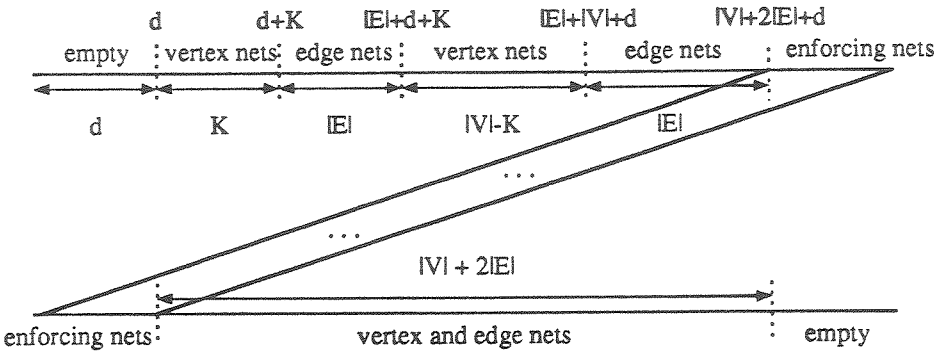


Figure 3: Illustration of the proof of Lemma 3.3

Proof: The problem is in NP by Lemma 3.1. To show that it is NP-complete, we construct a polynomial time transformation from the VERTEX COVER problem. (Again, see [7] for a definition of the VERTEX COVER problem.) Given a graph $G = (V, E)$, we construct an instance of the CPA-d problem so that G has a vertex cover of size $\leq K$ if and only if the instance of the CPA-d problem so constructed has a solution.

For each vertex $v \in V$, we have a "vertex net" $N_v = \{t_v, b_v\}$; for each edge $e \in E$, we have two "edge nets" $N_e = \{t_e, b_e\}$ and $N_{e'} = \{t_{e'}, b_{e'}\}$. In addition, we also have d "enforcing nets" $N_i = \{t_i, b_i\}$, $i = 1, 2, \ldots, d$. The structure of the channel is illustrated in Figure 3. The terminals of the enforcing nets are completely fixed, and all the enforcing nets cross the middle part of the channel. Hence they contribute d to the density at each column in the middle part of the channel. The first and the last d columns of the channel are reserved for the enforcing nets. The remaining nets can only be assigned to the middle $|V| + 2|E|$ columns of the channel. Except for this, there are no additional position constraints on the bottom of the channel. Whereas the middle $|V| + 2|E|$ grid points on the top of the channel is further divided into four parts. The first part consists of K grid points reserved for vertex nets, the second and fourth parts each consists of $|E|$ grid points reserved for the edge nets, and the third part consists of $|V| - K$ grid points reserved for the vertex nets. Furthermore, if $e \in E$ is the ith edge in E, then both t_e and $t_{e'}$ can only be assigned to the ith positions in the two parts reserved for the edge nets. There are no order constraints on the top of the channel. On the bottom of the channel, if $e = \{v_i, v_j\} \in E$ and $i < j$ (here we have assumed that $V = \{v_1, v_2, \ldots, v_{|V|}\}$), then b_{v_i} must be assigned to the left of b_e, and b_{v_j} must be assigned to the left of $b_{e'}$.

Formally, we have $\Phi = (L, TOP, BOTTOM, N, T, B, R_T, R_B)$, where

$$\begin{aligned} L &= |V| + 2|E| + 2d; \\ TOP &= \{t_v : v \in V\} \cup \{t_e, t_{e'} : e \in E\} \cup \{t_1, t_2, \ldots, t_d\}; \end{aligned}$$

$$
\begin{aligned}
BOTTOM &= \{b_v : v \in V\} \cup \{b_e, b_{e'} : e \in E\} \cup \{b_1, b_2, \ldots, b_d\};\\
N &= \{N_i = \{t_i, b_i\} : 1 \le i \le d\} && \text{(* enforcing nets *)}\\
&\quad \cup \{N_v = \{t_v, b_v\} : v \in V\} && \text{(* vertex nets *)}\\
&\quad \cup \{N_e = \{t_e, b_e\}, N_{e'} = \{t_{e'}, b_{e'}\} : e \in E\}; && \text{(* edge nets *)}\\
T_i &= \{|V| + 2|E| + d + i\}, \text{ for } i = 1, 2, \ldots, d;\\
T_v &= \{d+1, d+2, \ldots, d+K\}\\
&\quad \cup \{|E| + K + d + 1, |E| + K + d + 2, \ldots, |V| + |E| + d\}, \ \forall\, v \in V;\\
T_e &= T_{e'}\\
&= \{K + d + j, |V| + |E| + d + j\}, \text{ if } e \text{ is the } j\text{th edge of } E,\ 1 \le j \le |E|;\\
B_i &= \{i\}, \text{ for } i = 1, 2, \ldots, d;\\
B_v &= B_e\\
&= B_{e'}\\
&= \{d+1, d+2, \ldots, |V| + 2|E| + d\}, \ \forall\, v \in V, e \in E;\\
R_T &= (TOP, \phi), \ \textit{i.e.}, \text{ no order constraints on } TOP;\\
R_B &= (BOTTOM, \{(v_i, e), (v_j, e') : e = \{v_i, v_j\} \in E \text{ and } i < j\}).
\end{aligned}
$$

It is straightforward to verify that Φ can be constructed in polynomial time and that it is feasible. Furthermore any solution to Φ has density $\ge d$ (due to the enforcing nets). If Φ has a solution with density d, then the two terminals of each vertex or edge net must be assigned to the same column. This is possible if G has a vertex cover S of size $\le K$. Since S is a vertex cover of G, for each edge $e = \{u, v\} \in E$, either $u \in S$ or $v \in S$. Hence for any edge e, either b_e or $b_{e'}$ is preceded only by terminals b_w with $w \in S$, depending on which endpoint of e is in S. Therefore if we assign the vertex nets corresponding to the vertices of S, together with $K - |S|$ other vertex nets to the first K columns of the middle part of the channel, then there exist $|E|$ edge nets which can be legally assigned to the next $|E|$ columns without violating the order or position constraints (one edge net from each pair N_e, $N_{e'}$). Now the next $|V| - K$ columns can be assigned to the remaining vertex nets and the remaining $|E|$ columns can be assigned to the remaining edge nets. On the other hand, if G has no vertex cover of size $\le K$, then exactly K grid points on the top of the first $K + |E|$ columns of the middle part of the channel must be assigned to terminals of vertex nets (due to the position constraints). Whereas either more than K grid points on the bottom of these columns are assigned to terminals of vertex nets, or for some $e \in E$, both b_e and $b_{e'}$ are assigned to grid points on the bottom of these columns (due to the

order constraints). In either case there exists some net whose terminals are not assigned to the same column. Hence Φ cannot have a solution with density d. This proves the lemma. □

Theorem 3.4 *The problem of finding an optimal solution for a feasible instance of the CPA problem is NP-hard even if all nets are two-terminal nets having one terminal on each side of the channel.*

4 An Optimal Algorithm

We present in this section an optimal algorithm for an important case of the CPA problem with position and order constraints where both R_T and R_B are linearly ordered sets. We will refer to it as the *Linear CPA* problem. Without loss of generality, we may assume that $t_1 <_T t_2 <_T \ldots <_T t_p$ and $b_1 <_B b_2 <_B \ldots <_B b_q$. In other words, t_i (respectively, b_i) must be assigned to the left of t_j (respectively, b_j) if and only if $i < j$. We simplify our notation by writing $f(i)$ for $f(t_i)$ and $g(j)$ for $g(b_j)$. For convenience of description, we augment the channel with an auxiliary column 0, and introduce two trivial nets (nets consisting of only one terminal) $N_0 = \{t_0\}$ and $N_0' = \{b_0\}$ with $t_0 <_T t_1$, $b_0 <_B b_1$ and $T_0 = TOP$, $B_0 = BOTTOM$. In the rest of this section, Φ will stand for such an augmented instance of the Linear CPA problem.

We define *density functions* in Section 4.1, and introduce *crossing numbers* in Section 4.2, which will be used in our algorithm. The algorithm is then presented in Section 4.3. Some remarks pertinent to the algorithm are given in Section 4.4.

4.1 Density Functions

A solution to Φ is called an (i, j, k)-*solution* if it assigns exactly $t_1, t_2, \ldots, t_i, b_1, b_2, \ldots, b_j$ to the first k columns of the channel. The maximum of the densities at the first k columns of an ordinary channel routing instance is called the *k-density* of the channel. Recall that the density at a column c in an instance of the ordinary channel routing problem is equal to the number of nets crossing column c, *i.e.*, the number of nets with its leftmost terminal to the left or on column c, its rightmost terminal to the right or on column c, and at least one terminal outside of column c.

A function $d : \{0, 1, \ldots, p\} \times \{0, 1, \ldots, q\} \times \{0, 1, \ldots, L\} \mapsto I^0 \cup \{+\infty\}$, where I^0 denotes the set of nonnegative integers, is called a *density function with respect to* Φ if it satisfies:

P1 For all $0 \leq i \leq p$, $0 \leq j \leq q$ and $0 \leq k \leq L$, if Φ has an (i, j, k)-solution, then $d(i, j, k)$ is equal to the minimum of the k-densities of all (i, j, k)-solutions to Φ;

P2 For all $0 \leq k \leq L$, if Φ has no (p, q, k)-solution, then $d(p, q, k) = +\infty$. In particular, if Φ is not feasible, then $d(p, q, L) = +\infty$.

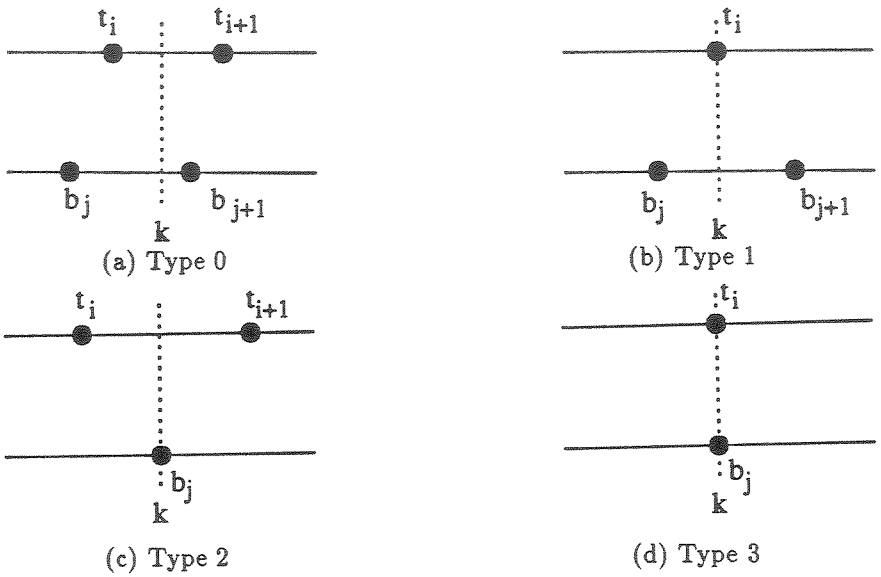


Figure 4: Four types of (i, j, k)-solutions

The following lemma can be directly deduced from the definition of density functions.

Lemma 4.1 *If d is a density function with respect to Φ, then Φ is feasible if and only if $d(p, q, L) < +\infty$. Furthermore, if Φ is feasible, then $d(p, q, L)$ is equal to the density of any optimal solution to Φ.*

The main idea of our algorithm is the following: Given Φ, we first compute a density function with respect to Φ using dynamic programming, and then use backtracking to reconstruct an optimal solution to Φ. Note that in our definition of density functions, we do not require the condition that $d(i, j, k) = +\infty$ if Φ has no (i, j, k)-solution to hold for all $0 \leq i \leq p$, $0 \leq j \leq q$ and $0 \leq k \leq L$. This condition is not necessary for Lemma 4.1 to hold, and the inclusion of it would increase the complexity of our algorithm. Hence it is possible for the density function (with respect to Φ) d computed by our algorithm to have $d(i, j, k) < +\infty$ even if Φ has no (i, j, k)-solution.

4.2 Crossing Numbers

The (i, j, k)-solutions to Φ can be classified into the following four types, according to the pin assignment at column k as illustrated in Figure 4.

- *Type 0:* No terminal is assigned to either endpoints of column k (Figure 4(a));
- *Type 1:* Only t_i is assigned to (the top endpoint of) column k (Figure 4(b));
- *Type 2:* Only b_j is assigned to (the bottom endpoint of) column k (Figure 4(c));
- *Type 3:* Both t_i, b_j are assigned to column k (Figure 4(d)).

Lemma 4.2 *If π, π' are two (i,j,k)-solutions to Φ of the same type, then column k has the same density in both π and π'.*

Proof: Let $R_1(i,j)$ denote the set of nets with one terminal in $\{t_1, t_2, \ldots, t_{i-1}, b_1, b_2, \ldots, b_j\}$ and one terminal in $TOP \cup BOTTOM - \{t_1, t_2, \ldots, t_i, b_1, b_2, \ldots, b_j\}$, and the net containing t_i, if it is not trivial. Let $R_2(i,j)$ denote the set of nets with one terminal in $\{t_1, t_2, \ldots, t_i, b_1, b_2, \ldots, b_{j-1}\}$ and one terminal in $TOP \cup BOTTOM - \{t_1, t_2, \ldots, t_i, b_1, b_2, \ldots, b_j\}$, and the net containing b_j, if it is not trivial. Let $R_3(i,j)$ denote the set of nets with one terminal in $\{t_1, t_2, \ldots, t_{i-1}, b_1, b_2, \ldots, b_{j-1}\}$ and one terminal in $TOP \cup BOTTOM - \{t_1, t_2, \ldots, t_i, b_1, b_2, \ldots, b_j\}$, and the nontrivial nets containing t_i or b_j, if it is not the case that t_i and are the only terminals of the same net. If both π and π' are of type 1 (Figure 4(b)), then the density of column k is equal to $|R_1(i,j)|$ in both of them; If they are both of type 2 (Figure 4(c)), then the density of column k is equal to $|R_2(i,j)|$ in both of them; If they are both of type 3 (Figure 4(d)), then the density of column k is equal to $|R_3(i,j)|$ in both of them. Finally, if both π and π' are of type 0 (Figure 4(a)), then the density of column k in both of them is equal to the number of nets with one terminal in $\{t_1, t_2, \ldots, t_i, b_1, b_2, \ldots, b_j\}$ and one terminal in $TOP \cup BOTTOM - \{t_1, t_2, \ldots, t_i, b_1, b_2, \ldots, b_j\}$. Hence the lemma follows. □

We can now define the *crossing numbers at* (i,j,k) as follows:

$$x(i,j,k) = \begin{cases} +\infty & \text{if } k \notin T_i \\ |R_1(i,j)| & \text{otherwise;} \end{cases}$$

$$y(i,j,k) = \begin{cases} +\infty & \text{if } k \notin B_j \\ |R_2(i,j)| & \text{otherwise;} \end{cases}$$

$$z(i,j,k) = \begin{cases} +\infty & \text{if } k \notin T_i \cap B_j \\ |R_3(i,j)| & \text{otherwise,} \end{cases}$$

where $R_1(i,j)$, $R_2(i,j)$, $R_3(i,j)$ are as defined in the proof of Lemma 4.2, and $x(i,j,k)$ ($y(i,j,k)$, $z(i,j,k)$, respectively) is called the *crossing number of type 1* (*type 2, type 3*, respectively) *at* (i,j,k). According to Lemma 4.2, if π is any (i,j,k)-solution to Φ of type 1, (type 2, type 3, respectively), then the density of column k in π is equal to $x(i,j,k)$ ($y(i,j,k)$, $z(i,j,k)$, respectively).

The crossing numbers of type 1 can be computed by the following procedure:

```
Procedure: Crossing Numbers of Type 1;
  begin
    for i := 0 to p do
      for j := 0 to q do
        begin
          Compute |R1(i,j)|;
          for k := 0 to L do
            x(i,j,k) := +∞
```

```
      end;
    for i := 0 to p do
      for j := 0 to q do
        for k ∈ T_i do
          x(i, j, k) := |R_1(i, j)|
  end;
```

It is obvious that the above procedure can be completed in $O(pqL)$ time and space. Similarly, crossing numbers of type 2 and type 3 can be computed in $O(pqL)$ time and space.

4.3 The Algorithm

We are now ready to present our algorithm for the Linear CPA problem. The input to the algorithm is an instance $\Phi = (L, TOP, BOTTOM, N, T, B, R_T, R_B)$ of the Linear CPA problem, and the output is an optimal solution to Φ, or an indication that no such solution exists. (We have assumed $0 - 1 = 0$ for convenience.)

```
Algorithm: Linear CPA;
  begin
    (* Initialization *)
    for i := 0 to p do
      for j := 0 to q do
        d(i, j, 0) := +∞;
    for k := 0 to L do
      d(0, 0, k) := 0;
    Compute the crossing numbers;
    (* Computing d(p, q, L) using dynamic programming *)
    for k := 1 to L do
      for i := 0 to p do
        for j := 0 to q do
          begin
            D_1 := max{d(i − 1, j, k − 1), x(i, j, k)};          (* type 1 solution *)
            D_2 := max{d(i, j − 1, k − 1), y(i, j, k)};          (* type 2 solution *)
            D_3 := max{d(i − 1, j − 1, k − 1), z(i, j, k)};      (* type 3 solution *)
            d(i, j, k) := min{d(i, j, k − 1), D_1, D_2, D_3}
          end;
    if d(p, q, L) = +∞
    then return "Φ is not feasible."
    else begin (* Constructing an optimal solution using backtracking *)
           i := p; j := q;
           for k := L downto 1 do
             if d(i, j, k) = D_1
             then f(i) := k; i := i − 1
             else if d(i, j, k) = D_2
                    then g(j) := k; j := j − 1
                    else if d(i, j, k) = D_3
                           then f(i) := k; g(j) := k; i := i − 1; j := j − 1;
           return π = (f, g)
         end
  end.
```

Theorem 4.3 *Algorithm Linear CPA correctly solves the Linear CPA problem in $O(pqL)$ time and space, where p, q, L is the number of terminals on the top of the channel, the number of terminals on*

the bottom of the channel, the length of the channel, respectively.

Proof: We first show that the function d as computed in Algorithm Linear CPA is a density function with respect to Φ. We prove by induction on k that if Φ has an (i,j,k)-solution, then $d(i,j,k)$ is equal to the minimum of the k-densities of all (i,j,k)-solutions to Φ. This is true for $k=0$ by the way we chose the boundary values. Assume it is true for $k-1$, $1 \leq k \leq L$. If Φ has an (i,j,k)-solution S with minimum k-density d_S, then S must be either an $(i,j,k-1)$-solution, or an $(i-1,j,k-1)$-solution, or an $(i,j-1,k-1)$-solution, or an $(i-1,j-1,k-1)$-solution to Φ. By the induction hypothesis and the fact that k-density $= \max\{(k-1)$-density, density at column $k\}$, if S is an $(i,j,k-1)$-solution to Φ, then $d_S = d(i,j,k-1)$; if S is an $(i-1,j,k-1)$-solution to Φ, then $d_S = \max\{d(i-1,j,k-1), x(i,j,k)\}$; if S is an $(i,j-1,k-1)$-solution to Φ, then $d_S = \max\{d(i,j-1,k-1), y(i,j,k)\}$; and if S is an $(i-1,j-1,k-1)$-solution to Φ, then $d_S = \max\{d(i-1,j-1,k-1), z(i,j,k)\}$. Hence we have

$$\begin{aligned} d_S &= \min\{d(i,j,k-1), \\ &\qquad \max\{d(i-1,j,k-1), x(i,j,k)\}, \\ &\qquad \max\{d(i,j-1,k-1), y(i,j,k)\}, \\ &\qquad \max\{d(i-1,j-1,k-1), z(i,j,k)\}\} \\ &= d(i,j,k). \end{aligned}$$

Therefore, the claim holds for k and consequently d satisfies P1.

We proceed to show that d also satisfies P2 by induction on k. By the way we chose the boundary values, we have $d(p,q,0) = +\infty$ if either $p > 0$ or $q > 0$. Hence the claim holds for $k=0$. Assume that the claim holds for $k-1$, $1 \leq k \leq L$ and Φ has no (p,q,k)-solution. Then Φ has no $(p,q,k-1)$-solution either. Hence $d(p,q,k-1) = +\infty$ by the induction hypothesis. Furthermore, either Φ has no $(p-1,q,k-1)$-solution or $k \notin T_p$ (and hence $x(p,q,k) = +\infty$). Therefore $\max\{d(p-1,q,k-1), x(p,q,k)\} = +\infty$. Similarly, we can show that $\max\{d(p,q-1,k-1), y(p,q,k)\} = \max\{d(p-1,q-1,k-1), z(p,q,k)\}\} = +\infty$. Consequently

$$\begin{aligned} d(p,q,k) &= \min\{d(p,q,k-1), \\ &\qquad \max\{d(p-1,q,k-1), x(p,q,k)\}, \\ &\qquad \max\{d(p,q-1,k-1), y(p,q,k)\}, \\ &\qquad \max\{d(p-1,q-1,k-1), z(p,q,k)\}\} \\ &= +\infty. \end{aligned}$$

Hence the claim also holds for k. Therefore P2 is also satisfied.

We next show that $\pi = (f, g)$ returned by Algorithm Linear CPA is an optimal solution to Φ. By the construction of π, conditions 3, 4, and 5 in the definition of a solution to Φ are satisfied. If there exists i, $1 \leq i \leq p$, such that $f(i) \notin T_i$, then $x(i, j, f(i)) = z(i, j, f(i)) = +\infty$ for all j's, $0 \leq j \leq q$, implying $d(p, q, L) = +\infty$. A contradiction to the fact that Φ is feasible. So condition 1 is also satisfied. Similarly, condition 2 is satisfied. Thus π is a solution to Φ. We can apply similar argument to show by induction that π is indeed optimal. Therefore, the correctness of the algorithm follows from Lemma 4.1.

The complexity of the algorithm is clearly dominated by the computation of the crossing numbers and $d(p, q, L)$. Since the crossing numbers can be computed in $O(pqL)$ time and space, the time and space complexities of the algorithm are both $O(pqL)$. □

4.4 Remarks

I. The channel routing problem with movable terminals considered in [8] can be solved by our algorithm as follows: We first apply Algorithm Linear CPA with channel length $p + q$. Because the existence of empty columns cannot decrease channel density, the solution so obtained has minimum density over all solution to the problem with arbitrary channel length. We then remove all vertical constraints [16] by splitting any column whose endpoints are assigned to terminals of different nets into two adjacent columns, with one terminal on each column. This operation does not change the density of the solution, but the new solution can be routed with channel width equals to its density by the Left Edge Algorithm [9] because there is no vertical constraint. Empty columns can now be removed to make the length of the resulting channel at most $p + q$. The solution so obtained is a minimum width solution.

II. The complexity of Algorithm Linear CPA can be reduced by an order of magnitude if the terminals on one side of the channel are completely fixed. This is because we can discard one of the first two variables in the definitions of density functions and crossing numbers.

III. There are possibly many minimum density solutions to a feasible instance Φ of the Linear CPA problem. Figures of merit other than channel density can be used to guide the backtracking process in Algorithm Linear CPA. In particular, minimum density solutions with the following properties are of special interest:

1. Acyclicy of the vertical constraint graph;

2. Short longest path in the vertical constraint graph.

One method of reconstructing a minimum density solution with the above properties is the following: First we compute $L^* = \min\{l : d(p,q,l) = d(p,q,L)\}$. We then construct an optimal solution to $\Phi^* = (L^*, TOP, BOTTOM, N, T, B, R_T, R_B)$. The $L - L^*$ columns at the end of the channel can now be used to break cycles and long paths of the vertical constraint graph. This is because if (N_i, N_j) is an edge of the vertical constraint graph G, then it can be removed from G by splitting every column with its top endpoint assigned to a terminal of N_i, its bottom endpoint assigned to a terminal of N_j into two adjacent columns, one for each terminal. Hence our problem can now be restated as follows: Given an arc-weighed directed graph (the weight of an arc corresponds to the number of columns needed to be split in order to remove that arc) and an integer C $(= L - L^*) \geq 0$, remove a set of arcs such that the sum of the arc weights does not exceed C, so that the resulting directed graph has the properties listed above.

This problem is NP-hard in general because it contains the directed Hamiltonian path problem [7]. Hence we have to resort to heuristics for efficiency. We can first compute a feedback arc set (a set of arcs whose removal eliminates all directed cycles) of G and remove it from G to obtain a directed acyclic graph, if its weight does not exceed C. We then adjust the value of C appropriately. The problem now becomes that of removing a set of arcs such that the sum of the arc weights is minimized, so that the length of a longest path in the resulting directed acyclic graph does not exceed some prechosen threshold value. This graph problem is of special interest because it can find applications in channel routing, where doglegs are being introduced (instead of empty columns) to break vertical constraints [14]. We do not know if this problem is polynomial time solvable. Heuristic methods can be applied to solve it.

5 Extensions

We discuss in this section how our algorithm can be extended to handle additional pin assignment constraints. In particular, we consider *separation constraints*, which require the distance between each pair of consecutive terminals to be within a certain range.

In Section 5.1 we present an optimal algorithm for the CPA problem with position constraints, linear order constrains, and general separation constraints. The complexity of our algorithm is $O(pqL^3)$. In Section 5.2, we present $O(L^3)$ time algorithms for several important cases of the problem which have restricted forms of separation constraints. Section 5.3 shows how our algorithm can be used to solve the *channel routing problem with movable modules.*

5.1 General Separation Constraints

In the most general case, separation constraints have the following form: for $1 \leq i \leq p-1$, $1 \leq j \leq q-1$ the distance s_i between t_i and t_{i+1} must satisfy $l_i \leq s_i \leq r_i$; and the distance $s_j^{'}$ between b_j and b_{j+1} must satisfy $l_j^{'} \leq s_j^{'} \leq r_j^{'}$. Algorithm Linear CPA can be generalized to solve the CPA problem with position constraints, linear order constraints, and general separation constraints optimally in polynomial time.

Let $d : \{0, 1, \ldots, p\} \times \{0, 1, \ldots, q\} \times \{0, 1, \ldots, L\}^2 \mapsto I^0 \cup \{+\infty\}$, be a function defined by the following recurrence relations:

$$\begin{aligned} d(i, j, 0, v) &= +\infty, \ \forall\, i, j, v, \ i > 0; \\ d(i, j, u, 0) &= +\infty, \ \forall\, i, j, u, \ j > 0; \\ d(0, 0, u, v) &= 0, \ \forall\, u, v; \end{aligned}$$

$$d(i, j, u, v) = \begin{cases} \min_{u' \in I_u}\{\max\{d(i-1, j, u', v), x(i, j, u)\}\} & \text{if } u > v \\ \min_{u' \in I_u, v' \in I_v}\{\max\{d(i-1, j-1, u', v'), z(i, j, u)\}\} & \text{if } u = v \\ \min_{v' \in I_v}\{\max\{d(i, j-1, u, v'), y(i, j, v)\}\} & \text{if } u < v, \end{cases}$$

where

$$I_u = \{u' \in \{1, 2, \ldots, L\} : u - r_{i-1} \leq u' \leq u - l_{i-1}\}$$

and

$$I_v = \{v' \in \{1, 2, \ldots, L\} : v - r_{j-1}^{'} \leq v' \leq v - l_{j-1}^{'}\}.$$

We have again assumed $0 - 1 = 0$ for convenience.

Lemma 5.1 *Let Φ be an instance of the CPA problem with position constraints, linear order constraints, and general separation constraints. For all $0 \leq i \leq p$, $0 \leq j \leq q$, $0 \leq u, v \leq L$, if Φ has an (i, j, w)-solution which assigns t_i to (the top endpoint of) column u and assigns b_j to (the bottom endpoint of) column v, where $w = \max\{u, v\}$, then $d(i, j, u, v)$ is equal to the minimum of the w-densities of all such solutions to Φ.*

To prove Lemma 5.1, consider the case where $u > v$ and hence $w = u$. Because of the separation constraints, terminal t_{i-1} must be assigned to the top endpoint of a column located at a position $u' \in I_u$. Since columns $w' + 1$, $w' + 2$, ..., $u - 1$ are empty (if t_{i-1} is assigned to the top endpoint of column u'), where $w' = \max\{u', v\}$, the densities at these columns are less than or equal to the maximal of the

densities at column w' and u in such an (i,j,u)-solution to Φ. Let D_k denote the minimum k-density of any (i,j,w)-solution to Φ, $0 \leq k \leq w$. Then we have

$$\begin{aligned} D_w &= \max_{u' \in I_u} \{D_{w'}, \text{ density at column } u \text{ in any } (i,j,w)\text{-solution to } \Phi\} \\ &= \min_{u' \in I_u} \{\max\{d(i-1,j,u',v), x(i,j,u)\}\} \\ &= d(i,j,u,v). \end{aligned}$$

The other two cases can be similarly proven, details are omitted here.

Lemma 5.2 *Let Φ be an instance of the CPA problem with position constraints, linear order constraints, and general separation constraints. For all $0 \leq u, v \leq L$, if Φ has no (p,q,w)-solution which assigns t_p to (the top endpoint of) column u and assigns b_q to (the bottom endpoint of) column v, where $w = \max\{u,v\}$, then $d(i,j,u,v) = +\infty$.*

Using dynamic programming, $d(i,j,u,v)$ can be computed from the previously computed d values in $O(L)$ time if $u \neq v$, or in $O(L^2)$ time if $u = v$. Hence the function d can be computed in $O(pqL^2) * O(L) + O(pqL) * O(L^2) = O(pqL^3)$ time and $O(pqL^2)$ space. (However, if $\max\{r_i - l_i, r_j' - l_j' : 1 \leq i \leq p-1, 1 \leq j \leq q-1\} \leq c$ for some constant c, then $O(pqL^2)$ time and space suffices.) The density of an optimal solution to Φ is equal to $D = \min\{d(p,q,u,v) : p \leq u \leq L, q \leq v \leq L\}$. Once D is computed, an optimal solution can be reconstructed by backtracking. Therefore, we have:

Theorem 5.3 *The CPA problem with position constraints, linear order constraints, and general separation constraints can be solved optimally in $O(pqL^3)$ time and $O(pqL^2)$ space, where p, q, L is the number of terminals on the top of the channel, the number of terminals on the bottom of the channel, the length of the channel, respectively.*

5.2 Special Separation Constraints

The complexity of our algorithm for the CPA problem with position constraints, linear order constraints, and general separation constraints is relatively high because the separation constraints are specified in its most general form. In practice, only some special forms of separation constraints are frequently encountered, and our algorithm can be tailored to reduce its running time. In the sequel, we consider the following special forms of the separation constraints:

1. No separation constraints, *i.e.*, the separation constraints are all of the form:

$$l_i, l_j' = 1, \text{ and } r_i, r_j' = +\infty, \forall\, i, j.$$

2. *Adjacency constraints* only, *i.e.*, the separation constraints are all of the form:

$$l_i, l_j^{'} = 1, \text{ and } r_i, r_j^{'} = 1 \text{ or } +\infty, \forall\, i, j.$$

3. *Fixed distance constraints* only, *i.e.*, the separation constraints are all of the form:

$$(l_i = r_i, l_j^{'} = r_j^{'}), \text{ or } (l_i, l_j^{'} = 1 \text{ and } r_i, r_j^{'} = +\infty), \forall\, i, j.$$

4. *Minimum separation constraints* only, *i.e.*, the separation constraints are all of the form:

$$r_i, r_j^{'} = +\infty, \forall\, i, j.$$

5. Any combination of the above.

Observe that if $l_i = 1$ ($l_j^{'} = 1$), then t_i, t_{i+1} (b_j, b_{j+1}) can be assigned as close to each other as possible (*i.e.*, no minimum separation constraint). If $r_i = +\infty$ ($r_j^{'} = +\infty$), then t_i, t_{i+1} (b_j, b_{j+1}) can be assigned as far away from each other as possible (*i.e.*, no *maximum separation constraint*). If $l_i = r_i$ ($l_j^{'} = r_j^{'}$), then t_i, t_{i+1} (b_j, b_{j+1}) must be assigned exactly l_i ($l_j^{'}$) grid points away from each other (fixed separation constraint). If $l_i = r_i = 1$ ($l_j^{'} = l_{j+1}^{'} = 1$), then t_i, t_{i+1} (b_j, b_{j+1}) must be assigned adjacent to each other (adjacency constraint). Hence Case 1 is precisely the case we considered in the Section 4. In Case 2, the only separation constraints are the requirements that some pairs of consecutive terminals be assigned adjacent to each other (fixed separation of unit distance). In Case 3, the only separation constraints are the requirements that some pairs of consecutive terminals be assigned some fixed distances away from each other (it includes Case 2 as a special case). In Case 4, the only separation constraints are the requirements that some pairs of consecutive terminals be assigned at least some distances away from each other.

Note that Cases 3-5 can be reduced to either Case 1 or Case 2. Case 4 can be reduced to Case 1 by inserting $l_i - 1$ trivial nets between t_i and t_{i+1}, and by inserting $l_j^{'} - 1$ trivial nets between b_j and b_{j+1}. Similarly, Case 3 can be reduced to Case 2 by inserting $l_i - 1$ trivial nets between t_i and t_{i+1} if $l_i = r_i$, and by inserting $l_j^{'} - 1$ trivial nets between b_j and b_{j+1} if $l_j^{'} = r_j^{'}$. Hence Case 5 can be reduced to either Case 1 or Case 2. In the rest of this section, we show how our algorithm can be adapted to solve Case 2 and hence Case 5.

Because of the adjacency constraints, we need to remember the assignment of terminals on the previous column when we consider the terminal assignment on the current column (for example, if t_i is to be assigned to column k, and if $r_{i-1} = l_{i-1} = 1$, then t_{i-1} must have been assigned to column $k - 1$, otherwise the adjacency constraint would be violated). This is done by introducing three new

functions: $d_1, d_2, d_3 : \{0,1,\ldots,p\} \times \{0,1,\ldots,q\} \times \{0,1,\ldots,L\} \mapsto I^0 \cup \{+\infty\}$, with function d_1 (d_2, d_3, respectively) corresponds to type 1 (type 2, type 3, respectively) solutions (see Figure 4). They can be computed using the following recurrence relations (the symbol h is used to denote any of the function symbols d, d_1, d_2, d_3):

$$h(0,0,k) = 0, \ \forall\, k \geq 0;$$

$$h(i,j,0) = +\infty, \ \forall\, i,j,\ i+j>0;$$

$$d_1(i,j,k) = \begin{cases} \min\{\max\{d_1(i-1,j,k-1), x(i,j,k)\}, \\ \qquad \max\{d_3(i-1,j,k-1), x(i,j,k)\}\} & \text{if } r_{i-1} = 1 \\ \max\{d(i-1,j,k-1), x(i,j,k)\} & \text{if } r_{i-1} = +\infty; \end{cases}$$

$$d_2(i,j,k) = \begin{cases} \min\{\max\{d_2(i,j-1,k-1), y(i,j,k)\}, \\ \qquad \max\{d_3(i,j-1,k-1), y(i,j,k)\}\} & \text{if } r'_{j-1} = 1 \\ \max\{d(i,j-1,k-1), y(i,j,k)\} & \text{if } r'_{j-1} = +\infty; \end{cases}$$

$$d_3(i,j,k) = \begin{cases} \min\{\max\{d_1(i-1,j-1,k-1), z(i,j,k)\}, \\ \qquad \max\{d_3(i-1,j-1,k-1), z(i,j,k)\}\} & \text{if } r_{i-1} = 1 \text{ and } r'_{j-1} = +\infty \\ \min\{\max\{d_2(i-1,j-1,k-1), z(i,j,k)\}, \\ \qquad \max\{d_3(i-1,j-1,k-1), z(i,j,k)\}\} & \text{if } r'_{j-1} = 1 \text{ and } r_i = +\infty \\ \max\{d_3(i-1,j-1,k-1), z(i,j,k)\} & \text{if } r_i = r'_{j-1} = 1 \\ \max\{d(i-1,j-1,k-1), z(i,j,k)\} & \text{if } r_i = r'_{j-1} = +\infty; \end{cases}$$

$$d(i,j,k) = \min\{d(i,j,k-1), d_1(i-1,j,k-1), d_2(i,j-1,k-1), d_3(i-1,j-1,k-1)\}.$$

Here we have again assumed $0-1=0$ for convenience.

Lemma 5.4 *Let Φ be an instance of the CPA problem with position constraints, linear order constraints, and adjacency constraints. For all $0 \leq i \leq p$, $0 \leq j \leq q$, $0 \leq k \leq L$, if Φ has an (i,j,k)-solution of type l $(= 1,2,3)$, then $d_l(i,j,k)$ is equal to the minimum of the k-densities of all (i,j,k)-solutions of type l to Φ. Furthermore, for $0 \leq k \leq L$, if Φ has no (p,q,k)-solution of type l, then $d_l(p,q,L) = +\infty$.*

The proof of Lemma 5.4 is tedious but routine. Consider for example, the definition of $d_1(i,j,k)$. Here t_i is to be assigned to (the top endpoint of) column k. If there is adjacency constraint between t_{i-1} and t_i, then t_{i-1} must have been assigned to (the top endpoint of) column $k-1$. Hence the $(i-1,j,k-1)$-solution must be either of type 1 or type 3, therefore we have

$$d_1(i,j,k) = \min\{\max\{d_1(i-1,j,k-1), x(i,j,k)\}, \max\{d_3(i-1,j,k-1), x(i,j,k)\}\}.$$

Otherwise there is no adjacency constraint between t_{i-1} and t_i, and the $(i-1, j, k-1)$-solution can be any one of the four types. Hence

$$d_1(i,j,k) = \max\{d(i-1,j,k-1), x(i,j,k)\}.$$

Similar arguments can be applied to show that the recurrence relations for d_2, d_3 are also correct. Hence the following follows directly:

Corollary 5.5 *The function d as defined by the recurrence relations above is a density function with respect to Φ.*

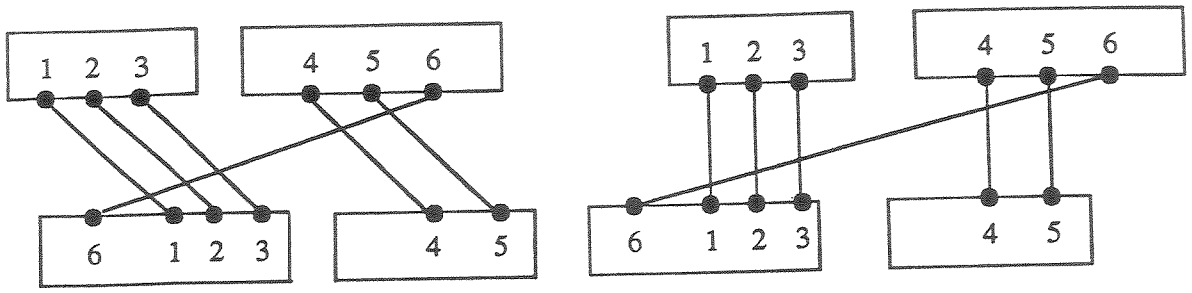
Using the recurrence relations, $d(p, q, L)$ can be computed in $O(pqL)$ time and space using dynamic programming. Hence an optimal solution can be reconstructed in the same amount of time and space by backtracking. However, in the cases where fixed separation constraints or minimum separation constraints are involved, the trivial nets inserted also contribute to the values of p and q, hence the worst case complexity of our algorithm for these cases is $O(L^3)$.

Theorem 5.6 *The CPA problem with position constrains, linear order constraints, and any combination of adjacency constraints, fixed distance constraints and minimum separation constraints can be solved optimally in $O(L^3)$ time and space, where L is the length of the channel. Furthermore, if only adjacency constraints present, then the problem can be solved in $O(pqL)$ time and space, where p, q is the number of terminals on the top, bottom of the channel, respectively.*

In fact, if no maximum separation constraints are involved, the complexity of our algorithm for the CPA problem with order constraints, linear order constraints and separation constraints remains at $O(L^3)$.

5.3 Channel Routing with Movable Modules

The CPA problem with order constraints, linear order constraints and fixed and minimum separation constraints (Case 5 in Section 5.2) can be applied to solve the *channel routing problem with movable modules*. In this problem, each side of the channel consists of the boundaries of a number of completely designed modules. The terminals have fixed positions inside each module, and the relative orderings of the modules on each side of the channel are completely fixed. However, the exact positions of the modules are not fixed (*i.e.*, they are movable). The objective is to determine the positions of the modules to minimize channel density. Figure 5 shows an example of the channel routing problem with movable terminals. This problem can be modeled as a special case of the CPA problem with position constraints,



(a) Before moving: $d = 4$ (b) After moving: $d = 1$

Figure 5: Channel routing with movable modules

linear order constraints and fixed and minimum separation constraints by introducing fixed separation constraints between the terminals inside the same module, and minimum separation constraints between the extreme terminals of consecutive modules on the same side of the channel, *i.e.*, between the rightmost (leftmost) terminal of a module and the leftmost (rightmost) terminal of the module on the same side of the channel immediately to its right (left). Hence we have the following corollary of Theorem 5.6:

Corollary 5.7 *The channel routing problem with movable modules can be solved in $O(L^3)$ time and space, where L is the length of the channel. Furthermore, if there is only one module on one side of the channel, then it can be solved in $O(L^2)$ time and space.*

Note that the channel offsetting problem [10] is a special case of the channel routing problem with movable modules, where there is exactly one module on each side of the channel. By Corollary 5.7, our algorithm solves the channel offsetting problem in $O(L^2)$ time and space.

6 Applications

We describe in this section some applications our algorithms. In standard-cell layout design, cells of equal heights are packed into rows (see Figure 6). Feedthrough terminals (b, b', c, c', d, d' in Figure 6) can be inserted in between the cells. The space between the rows are reserved for routing, which is usually carried out by a channel router in a row by row fashion. The rows can be horizontally expanded by inserting extra spaces in between the cells. By doing so, the positions of the terminals on the boundaries of the cells are changed. Hence these are instances of the CPA problem.

There are certain constraints on the assignment of terminals. The feedthrough terminals and their relative orderings can be predetermined before the routing phase. The terminals on the boundaries of the cells usually have fixed positions inside the cells, although the cells can be moved as a whole. Hence the relative orderings of the terminals are completely fixed. The requirement that the terminals on the boundaries of the cells must have fixed positions inside the cells can be modeled by the fixed separation

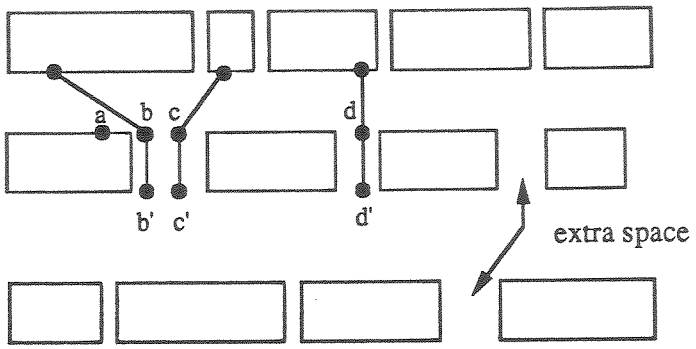


Figure 6: Standard-cell layout design

constraints. Furthermore, to avoid feedthrough terminals being inadvertently placed on the boundary of a cell, and to avoid inadvertently moving some terminals on the boundaries of the cells out of the cells, we can impose minimum separation constraints on the leftmost (rightmost) terminals on the boundaries of the cells and the terminals immediately to their left (right). Hence our algorithms can be applied to minimize the densities of the channels.

Similar situations occur in building-block layout design. The routing area in a building-block layout can be decomposed into channels, and these channels have to be routed in certain order. For example, in Figure 7, channel C must be routed after channels A, B, D and E. Similarly, channel F must be routed after channels D, E, G and H. The junction terminals (those terminals located on the dotted lines) usually have fixed ordering and are movable. Furthermore, the widths of the horizontal channels (channels A, B, D, E, F, G and H in Figure 7) can be expanded. Hence channels C and F are again instances of the CPA problem. In this case, the pin assignment constraints are similar to those encountered in the standard-cell layout, hence our algorithms again apply.

In another situation in building-block layout design, where the modules are not fully designed, and only the relative orderings of the terminals on the boundaries of the modules have been determined, but not their exact positions. Again consider Figure 6. Suppose we have routed channels A, B, D, E, G, H and fixed their widths. The terminals on the sides of the vertical channels C and F can be moved within the modules. Again the vertical channels are instances of the CPA problem. The pin assignment constraints in this case are slightly different from those in the previous examples. Instead of fixed separation constraints, we have position constraints for the terminals on the boundaries of the modules (they must stay inside the modules), as well as for the junction terminals (they cannot be moved to the boundaries of modules). Again we have minimum separation constraints imposed on the topmost (bottommost) terminals on the vertical boundaries of the modules and the terminals adjacent

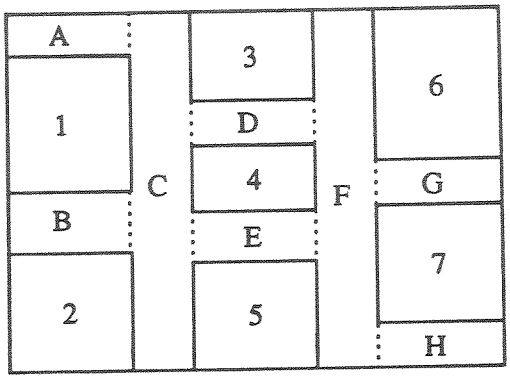


Figure 7: Building-block layout design

to them. Our algorithms still apply in this case.

In fact, whenever the terminals have some degree of freedom of choosing their own positions before the channel routing phase, we have an instance of the CPA problem. Our algorithms apply in many situations arisen in practice. Due to its simplicity, we believe that our algorithm can be incorporated into existing placement/routing systems to further reduce routing area.

7 Experimental Results

We have implemented our algorithms in Pascal language on a Sun 3/50 workstation running on Unix 4.2BSD. The program was tested on several benchmark channel routing problems, including Deutsch's famous difficult example (Ex8). The results are shown in Table 1. We have assumed $T_i = B_j = \{1, 2, \ldots, L\}$ for $1 \leq i \leq p$ and $1 \leq j \leq q$, and no separation constraints. Example Ex1 is from [8], examples Ex2 to Ex8 are the examples given in [16]. The "LB" items in the table refer to the minimum channel densities achievable by moving the terminals (while preserving their relative orderings) if the length of the channels are allowed to be arbitrarily extended. From Table 1 we can see that reductions in the channel densities by as much as 25% were obtained.

We have also implemented the algorithm in [8] and compared the minimum lengths of the channels when density lowerbounds were achieved. The algorithm in [8] differs from our algorithms in that they assume the length of the channel is unbounded, whereas we assume the length of the channel is prefixed and cannot be arbitrarily extended. Also their algorithm does not handle position and separation constraints. The results in Table 2 show that our algorithms use substantially shorter channels to achieve minimum densities. This implies that even if there is only relatively small room for the terminals to move, our algorithms are still able to exploit this freedom and to reduce the channel density significantly.

Ex.	#Nets	Len.	Old d	New d	LB	CPU(s)
Ex1	15	20	n.a.	5	5	1.05
Ex2	21	35	12	9	7	7.72
Ex3	30	62	15	14	14	26.92
Ex4	47	61	17	16	16	40.97
Ex5	54	79	18	16	16	66.52
Ex6	57	119	17	15	15	169.37
Ex7	62	119	20	17	17	137.65
Ex8	72	175	19	18	18	656.78

Table 1: Experimental results

	Channel length	
Examples	Gopal etc.	Our algorithm
Ex1	23	20
Ex2	52	45
Ex3	80	52
Ex4	89	57
Ex5	108	68
Ex6	165	102
Ex7	143	89
Ex8	255	157

Table 2: Comparisons with the algorithm of Gopal etc.

8 Conclusions

In this paper we study the channel pin assignment problem subject to position constraints, order constraints, and separation constraints. We show that the problem is NP-hard in general and present polynomial time optimal algorithms for an important case where the relative orderings of the terminals on the top and the bottom of the channel are completely fixed. We introduce the problem of channel routing with movable modules and show that it is a special case of our problems and hence can be solved optimally in polynomial time. We also discuss how our algorithms can be incorporated into standard-cell and building-block layout systems. Experimental results indicate that substantial reduction in channel density can be obtained by allowing movable terminals.

Several problems still remain open. For example, can our algorithm be generalized to handle the case of multiple parallel channels as occurred in the standard-cell layout design? What are the other special cases of the CPA problem which can be solved efficiently in polynomial time, especially when some sets of terminals are permutable? In general, we feel that the channel pin assignment is a very promising field of study, both from the theoretical point of view and from the practical point of view, and many open problems are well worthy of further pursuing.

References

[1] M.J. Atallah and S.E. Hambrusch, "Optimal rotation problems in channel routing", Tech. Report #CSD-TR-468, Dept. of Comp. Sci., Purdue Univ., West Lafayette, Indiana.

[2] B.S. Baker, S.N. Bhatt, and F.T. Leighton, "An approximation algorithm for Manhattan routing", *Proc. of the 1983 Symposium of Theory on Computing*, 477-486, 1983.

[3] D. Brown, and R.L. Rivest, "New lower bounds for channel routing", *Proc. of the CMU Conf. on VLSI Systems and Computation*, 178-185, Oct. 1981.

[4] M. Burstein and R. Pelavin, "Hierarchical channel router", *INTEGRATION, the VLSI journal, vol.* **1(1)**, 21-38, 1983.

[5] D.N. Deutsch, "A dogleg channel router", *Proc. of the 13th Design Automation Conference*, 425-433, 1976.

[6] C.M. Fiduccia and R.L. Rivest, "A greedy channel router", *Proc. of the 19th Design Automation Conference*, 418-424, 1982.

[7] M.R. Garey and D.S. Johnson, *Computers and intractability: a guide to the theory of NP-completeness*, W.H. Freeman & Co., San Francisco, 1979.

[8] I.S. Gopal, D. Coppersmith and C.K. Wong, "Optimal wiring of movable terminals", *IEEE Trans. on Computers, vol. C-***32(9)**, 845-858, 1983.

[9] A. Hashimoto and J. Stevens, "Wire routing by optimizing channel assignment within large apertures", in *Proc. of the 8th Design Automation Workshop*, 155-163, 1971.

[10] A.S. LaPaugh and R.Y. Pinter, "On minimizing channel density by lateral shifting", *Proc. of the 1983 International Conference on Computer-Aided Design*, 121-122, 1983.

[11] H.W. Leong and C.L. Liu, "Permutation channel routing", *Proc. of the 22nd Design Automation Conference*, 579-584, 1985.

[12] H.W. Leong, "*Routing problems in physical design of integrated circuits*", *Ph.D Thesis*, Dept. of Comp. Sci., Univ. of Illinois at Urbana-Champaign, 1986.

[13] G. Persky, "PRO — an automatic string placement program for polycell layout", *Proc. of the 13th Design Automation Conference*, 417-424, 1976.

[14] B. Preas, "Channel routing with non-terminal doglegs", *Proc. of the 1st European Design Automation Conference*, 451-458, 1990.

[15] T.G. Szymanski, "Dogleg channel routing is NP-complete", *IEEE Trans. on CAD, vol. CAD-4(1)*, 31-41, 1985.

[16] T. Yoshimura and E.S. Kuh, "Efficient algorithms for channel routing", *IEEE Trans. on CAD, vol. CAD-1*, 25-35, 1982.