Channel Density Minimization by Pin Permutation *

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

Abstract. We present in this paper a linear time optimal algorithm for minimizing the density of a channel (with exits) by permuting the terminals on the two sides of the channel. This compares favorably with the previously known near-optimal algorithm presented in [6] that runs in super-linear time. Our algorithm has important applications in hierarchical layout design of integrated circuits. We also show that the problem of minimizing wire length by permuting terminals is NP-hard in the strong sense.

1 Introduction

Channel routing is an important problem in VLSI layout design and has been extensively studied before [2, 8, 11, 22]. Conventional channel routers assume the positions of the terminals on each side of the channel are fixed. However, it is typical in practice that after the placement phase, the positions of the terminals are not completely fixed, and there is some degree of freedom to choose positions for the terminals. This freedom should be used to our advantage to make the subsequent routing task easier and hence obtain reduction in routing area. This type of problems have been studied by many researchers before [3, 4, 10, 12, 13, 14, 15, 16, 17, 19, 20, 21]. We study in this paper the problem of permuting the terminals on the two sides of a channel to minimize the channel density. An important application of routing channels with permutable (interchangeable) terminals is for solving the pin assignment and global routing problem in building-block layout design, as it was shown that the combined pin assignment and global routing problem can be reduced to routing a set of channels with permutable terminals [7].

Several special cases of the problem of minimizing channel density by permuting terminals have been studied before. In [18], an optimal algorithm was presented for the case where the channel has no exit. Another optimal algorithm was given in [7] for basic channels, i.e., channels containing only two-terminal nets with one terminal on each side of the channel. No optimal polynomial time algorithm for the general case was known before. Recently, a near-optimal polynomial time algorithm was presented in [6], which can guarantee to produce results that are within one of the optimal channel density. We show in this paper that the general problem can be solved optimally in polynomial time by presenting a linear time exact algorithm for this problem. We also show that the closely related problem of permuting the terminals so that the channel can be routed in minimum wire length is NP-hard in the strong sense. Due to space limitation, proofs of the results are not included, details can be found in [5].

2 Preliminaries

With respect to a horizontal channel, a net is said to have a left exit (respectively, right exit) if it has a terminal to the left (respectively, right) of the channel (outside the channel). A channel is said to have an exit if it has a net with either a left or a right exit. Each net $N_k$ is specified by an ordered pair $(t_k, b_k)$, where $t_k$ is the number of terminals on the top of the channel (called top terminals), and $b_k$ is the number of terminals on the bottom of the channel (called bottom terminals).

We say $N_k$ is of the form $(t_k, b_k)$ and use the notation $N_k = (t_k, b_k)$. A net $N_k$ is called a positive net if $t_k \geq b_k$, a negative net if $t_k \leq b_k$. If either $t_k = 0$ or $b_k = 0$, then $N_k$ is called a one-sided net, otherwise it is a two-sided net. If $t_k = 0$, then $N_k$ is a bottom-sided net. If $b_k = 0$, then $N_k$ is a top-sided net.

The Density Minimization (DM) problem is the problem of permuting the terminals on the two sides of a channel to minimize its density. An instance of the DM problem is a 3-tuple $(N, L, R)$, where $N$ is the set of nets to be routed, $L, R \subseteq N$ are, respectively, the set of nets in $N$ with left, right exits. We use $n = |N|$ to denote the number of nets in $N$. For any subset $N' \subseteq N$, we define:

\[
N' = \sum_{N_k \in N'} t_k; \quad b_{N'} = \sum_{N_k \in N'} b_k; \quad \tau_{N'} = \min_{N_k \in N'} t_k; \quad b_{N'} = \min_{N_k \in N'} b_k; \quad m_{N'} = \sum_{N_k \in N'} \max(t_k, b_k).
\]

We may assume that $\tau_{N'} = b_{N'} = l$, where $l$ is the length of the channel. This assumption is possible because we can always realize it by introducing trivial nets, i.e., nets of the form $(0, 1)$ or $(1, 0)$ without exits. Trivial nets represent nets requiring no connections. Without loss of generality, we may also assume no net in $N$ has the form $(0, 0)$. It is easy to see that no matter how the terminals are permuted, the density

\[
\sum_{x \in \phi} x = \min x = \max x = 0.
\]

---

*This work was partially supported by the National Science Foundation under grant MIP-8909586, by an IBM Faculty Development Award, and by an ACM SIGDA scholarship.

1 For convenience of presentation, we define

\[
\sum_{x \in \phi} x = \min x = \max x = 0.
\]
of the channel is at least \( \max\{|L|, |R|\} \) because all the nets in \( L \) cross the leftmost column of the channel, and all the nets in \( R \) cross the rightmost column of the channel.

Given an instance \( \langle N, L, R \rangle \) of the DM problem, we use \( B = L \cup R \) to denote the set of nets with both left and right exits, and \( M = N - L \cup R \) to denote the set of nets without exits. The set of trivial nets \( M_T \) is a subset of \( M \), and we let \( M_F = M - M_T \). Also, we let \( L^* = L - B \) and \( R^* = R - B \). \( L \) is said to be top critical if

\[
t'_{L^*} > l - b_{M_F \cup R^*} = b_{L M_T} = b_{L} + b_{B \cup M_T},
\]

bottom critical if

\[
b'_{L^*} > l - t_{M_F \cup R^*} = t_{L M_T} = t_{L} + t_{B \cup M_T},
\]

and critical if it is either top critical or bottom critical. That \( R \) is top critical, bottom critical and critical are similarly defined.

\[
dl = \begin{cases} 
1 & \text{if } L \text{ is critical,} \\
0 & \text{otherwise}
\end{cases}
\]

and similarly define \( \delta_R \). We also make use of a variable \( \delta \), such that \( \delta = 1 \) if \( |L| = |R| \), \( \delta_L = \delta_R = 0 \), and either \( b_{M_F \cup B} < (t'_{L^*} - b_{L^*}) + (t_{R^*} - b_{R^*}) \) or \( t_{M_F \cup B} < (b'_{L^*} - t_{L^*}) + (b_{R^*} - t_{R^*}) \), and \( \delta = 0 \) otherwise.

**Example 1**: Consider the channel shown in Figure 1, we have \( n = 5 \), \( l = 10 \), \( N = \{1, 2, 3, 4, 5\} \), \( N_1 = \{1, 2\} \), \( N_2 = \{3, 4\} \), \( N_3 = \{5\} \), \( L = \{N_1, N_2\} \), \( B = \{N_3\} \), \( R = \{N_1, N_2\} \), \( M = \{N_1, N_3, N_4\} \), \( M_T = \{N_5\} \), \( M_F = \{N_1, N_2, N_3\} \). Since \( |L| > |R| = 1 \), we have \( \delta = 0 \). Since \( b_{L M_T} = 1 + 2 + 1 = 4 > t_{L^*} = 3 \) and \( t_{L M_T} = 3 + 4 + 1 + 1 = 9 > b'_{L^*} = 1 \), \( L \) is not critical and hence \( \delta_L = 0 \). Since \( b_{B \cup M_T} = 2 + 1 + 3 < t_{R^*} = 4 \), \( R \) is top critical and hence \( \delta_R = 1 \). \( \square \)

The significance of \( L \) being top critical is that no matter how the terminals are permuted, the density of the resulting channel is at least \(|L| + 1\). To see this, consider the net \( N_k \in L^* \) with the property that the rightmost column that contains a terminal of \( N_k \) is the leftmost among all such rightmost columns of the nets in \( L^* \). Since \( t'_{L^*} > b_{L M_T} \), among the \( t_k \geq t'_{L^*} \) columns with a top terminal of \( N_k \) on it, at least one of them has a bottom terminal of a net in \( R^* \cup M_F \) on it. Hence this net crosses every net in \( L \) at this column, making the local density at this column at least \(|L| + 1\). We can also claim that if \( \delta = 1 \), then the minimum channel density that can be achieved by permuting terminals is at least \(|L| + 1\) or \(|R| + 1\).

Given an instance \( \langle N, L, R \rangle \) of the DM problem, the minimum channel density achievable by permuting terminals is at least

\[
D = \max\{|B| + d^*, \max\{|L| + \delta_L, |R| + \delta_R\} + 1\},
\]

where

\[
d^* = \begin{cases} 
0 & \text{if } t_s, b_s \leq 1 \text{ for all } N_k \in M \\
1 & \text{if } l \geq m_{L^* \cup M_F \cup R^*} \\
2 & \text{otherwise.}
\end{cases}
\]

In other words, the channel cannot be routed in fewer than \( D \) tracks.

**Theorem 2.1** Given an instance \( \langle N, L, R \rangle \) of the DM problem, the minimum channel density achievable by permuting terminals is at least

\[
D = \max\{|B| + d^*, \max\{|L| + \delta_L, |R| + \delta_R\} + 1\},
\]

where

\[
d^* = \begin{cases} 
0 & \text{if } t_s, b_s \leq 1 \text{ for all } N_k \in M \\
1 & \text{if } l \geq m_{L^* \cup M_F \cup R^*} \\
2 & \text{otherwise.}
\end{cases}
\]

In other words, the channel cannot be routed in fewer than \( D \) tracks.

**Theorem 2.1** provides a lower bound on the minimum density of a channel achievable by permuting terminals. For the example shown in Figure 1, we have \( m_{L^* \cup M_F \cup R^*} = (3 + 4 + 10 + 2 + 4) = 23 > 1 \), hence \( d^* = 2 \). Since \( |B| = 1, |L| = 3, |R| = 2, \delta_R = 1 \) and \( \delta_L = \delta = 0 \), we have \( D = 3 \). Figure 1 shows a terminal permutation that actually achieves channel density 3. Later on, we will show that this bound is always achievable by presenting an algorithm that constructs a channel that achieves it. The following lemma is used in showing the optimality of the algorithm.

**Lemma 2.2** Let \( X = (x_1, x_2, \ldots, x_u) \) and \( Y = (y_1, y_2, \ldots, y_v) \) be sequences of non-negative integers, such that \( u \geq v, x_1 \leq x_2 \leq \ldots \leq x_u, y_1 \geq y_2 \geq \ldots \geq y_v \), and \( S = \sum_{i=1}^u x_i = \sum_{j=1}^v y_j \), then for \( 1 \leq k \leq v \), \( \alpha_k = \sum_{i=1}^k x_i \leq \sum_{j=1}^{v-k} y_j = \beta_k \).

**3 Alternate Packing**

It is convenient to present our algorithm using a technique called alternate packing [6, 18], which computes a terminal permutation by "packing nets" one at a time. After a net is packed, the positions of some but not all of its terminals are determined, and the unassigned terminals of the PAN are on the same side of the channel. If these terminals are on the top of the channel, then the next net to be packed is a negative net, if
there is any, otherwise the next net to be packed is a positive net. This technique is described formally in the following algorithm.

Algorithm: AlternatePacking ($W$); 

(*) $W = (N_1, N_2, \ldots, N_n)$ is an ordered sequence of *)

(*) the nets to be routed *)

Begin

pan := 0; $\sigma$ := 0; 
current := 1; (*) index of next net to be packed *)
column := 1; $t$ := 1; $b$ := $b_1$;

while $W \neq \phi$ do

begin

Remove the current net from $W$;

if $\sigma \geq 0$

(* the PAN has $\sigma$ unassigned top terminals *)

(* and the current net is negative *)

then for $k := 1$ to $\min\{t + \sigma, b\}$ do

if $k \leq \min\{\sigma, b - t\}$

then $\text{AC(\text{column} + k - 1, \text{pan}, \text{current})}$

else $\text{AC(\text{column} + k - 1, \text{current})}$

(* $\sigma < 0$, the PAN has $-\sigma$ unassigned bottom *)

(* terminals, and the current net is positive *)

else for $k := 1$ to $\min\{t, b - \sigma\}$ do

if $k \leq \min\{t - b, \sigma\}$

then $\text{AC(\text{column} + k - 1, \text{pan})}$

else $\text{AC(\text{column} + k - 1, \text{current})}$;

if $\sigma + (t - b) > 0$

(* next PAN has only unassigned top terminals *)

then $N_j := \text{the next negative net in } W$

else $N_j := \text{the next positive net in } W$;

if $|t - b| \geq |\sigma|$

then $\text{pan} := \text{current}$;

column := $\text{column} + k$; $\sigma$ := $\sigma + (t - b)$;

current := $j$; $t$ := $t_1$; $b$ := $b_2$;

end

End.

A call to procedure AC($c, i, j$) assigns a top terminal of net $N_i$ to column $c$, a bottom terminal of net $N_j$ to column $c$. Since we can use two linked lists to separately store the positive nets and the negative nets, the algorithm can be implemented to run in $O(l)$ time, where $l$ is the length of the channel. Figure 2 shows a channel obtained by applying Algorithm AlternatePacking on the example shown in Figure 1 with $W = (N_1, N_2, N_3, N_4, N_5, N_6, N_7)$. The nets are actually packed in the following order: $(N_1, N_3, N_2, N_5, N_6, N_7, N_4)$. The density of the channel is equal to 4, instead of the optimal value 3.

Note that if the channel has no exits, then the density of the channel obtained by Algorithm AlternatePacking is at most two. This is because at most two nets intersect each column, one of them is the PAN when the column is under consideration, the other is the net under packing at that time. By carefully ordering the nets in $L^*$ and $R^*$, the algorithm can also be used to compute a terminal permutation for a given channel with exits so that the resulting channel has density within one of the optimal value [6].

Observe that for any net of the form $(k, k)$, Algorithm AlternatePacking assigns all of its terminals to $k$ consecutive columns. In particular, any net of the form $(1, 1)$ have both its terminals assigned to the same column.

4 The Optimal Algorithm

As we have seen in the last section, Algorithm AlternatePacking is not optimal in general. In order to achieve optimality, the terminals of the nets have to be carefully distributed, so that, for example, no net in $M_P \cup R^*$ is to cross all nets in $L$ at some column, if this is necessary and possible (i.e., when $D = |L|$ and hence $\max\{b, b_1\} = 0$). A convenient way of doing this is to partition the nets into a number of "smaller" nets, each containing a subset of the set of terminals of the original net. This is done in the following procedure. (Use $X \times Y$ to denote the concatenation of two sequences $X$ and $Y$. This notation extends to the case that $X$ or $Y$ is a set because we can consider a set as a sequence by arbitrarily ordering its elements.)

Procedure: DistributeNets ($\phi$);

(*) = $(N_1, L, R)$ is an instance of the DM problem *)

Begin

(* decompose the nets in $L^*$ and $R^*$ *)

$L^* := \epsilon; R^* := \epsilon$; (*) $\epsilon$ is the empty sequence *)

for $N_k \in L^*$ do

if $t_k \times b_k = 0$

then $L^* := L^* \cdot (N_k)$

else begin

$N'_k := (t_k, 0); N''_k := (0, b_k); L^* := L^* \cdot (N'_k, N''_k)$

end;

for $N_k \in R^*$ do

if $t_k \times b_k = 0$

then $R^* := R^* \cdot (N_k)$

else begin

$N'_k := (t_k, 0); N''_k := (0, b_k); R^* := R^* \cdot (N'_k, N''_k)$

end;

$S := M_P$;

(* decompose nets in $B$ into trivial nets *)

for $N_i \in B$ do

for $j := 1$ to $t_i + b_i$ do

begin

if $j \leq t_i$

then $N_{ij} := (1, 0)$

else $N_{ij} := (0, 1)$;

$S := S \cdot (N_{ij})$

end;

(* distribute the nets in $S$ *)

$S_L := \phi; S_R := \phi$

if $D = |L|$ then

if $t_L \geq b_L$

else

begin

$\sigma := 0$;

end.

Figure 2: Alternate Packing
then $S_L := \text{first } (t_{L*} - b_{L*}) \text{ negative nets of } S$
else $S_L := \text{first } \max\{0, b_{L*} - t_{L*}\} \text{ positive nets of } S$;
if $D = |R|$
then if $t_{R*} \geq b_{R*}$
then $S_R := \text{last } (t_{R*} - b_{R*}) \text{ negative nets of } S$
else $S_R := \text{last } \max\{0, b_{R*} - t_{R*}\} \text{ positive nets of } S$;
$S_M := S - S_L \cup S_R$
End.

A net of the form $(t, b)$ is said to have disparity $|t - b|$. A tag of $L$ is a net $N_k \in L$ such that $t_k = t_{L*}$ if $t_{L*} \geq b_{L*}$, and $b_k = b_{L*}$ otherwise. The tag of $R$ is similarly defined.

Procedure Distribute_Nets decomposes each net in $L^* \cup R^*$ into two one-sided nets, if it is not itself one sided. It also partitions each net in $B$ into nets of the form $(1,0)$ and $(0,1)$. These nets together with the trivial nets in $M_P$ form the set $S$. Note that the set of new nets obtained from decomposing the same net are grouped together and the ordering of the new nets are consistent with the original ordering of the old nets (i.e., if an old net $N_i$ appeared before $N_j$, then all the new nets obtained from decomposing $N_i$ appear before all the nets obtained from decomposing $N_j$). The set of new nets $S$ is partitioned into three subsets $S_L$, $S_R$ and $S_M$. The set $S_L$ is introduced to avoid having some net in $M_P \cup R^*$ crossing every net in $L$ at some column. This is necessary if $D = |L|$, because otherwise the density of the channel is at least $|L| + 1 > D$. It is also possible in this case because $D = |L|$ implies $\max\{\delta, \delta_L\} = 0$, hence there are enough terminals in $L \cup M_T$ to pad the columns with terminals of the tag of $L^*$ on it. The set $S_R$ is similarly introduced to avoid having a net in $L^* \cup M_P$ crossing every net in $R$. The set $S_M$ is introduced to keep positive nets in $M_P$ from crossing negative nets in $M_P$. This is necessary if $D = |B| + 1$, for otherwise the density of the channel would be at least $|B| + 2$. It is possible if $d^* \leq 1$.

With the nets decomposed as described in Procedure Distribute_Nets, we can distribute the terminals to the where we want them to be in applying Algorithm Alternate_Packing by carefully ordering the nets. This is enough to achieve optimality.

Algorithm: Optimal_Packing();
\begin{itemize}
\item [*] $(N, L, R)$ is an instance of the DM problem
\end{itemize}
\begin{itemize}
\item Begin
\item Sort $L^*$ into increasing order of net disparities;
\item Sort $R^*$ into decreasing order of net disparities;
\item If $D = |L|$ and $(t_{L*} > b_{L*} \text{ or } t_{L*} > t_{L*})$
then Select a tag of $L^*$ as its first net;
\item If $D = |R|$ and $(t_{R*} > b_{R*} \text{ or } b_{R*} > t_{R*})$
then Select a tag of $R^*$ as its last net;
\item $M_P^+ := \{N_k \in M_P: t_k \geq b_k\}; M_P^- := M_P - M_P^+$;
\item Distribute_Nets();
\item If $t_{L*} > b_{R*}$
then $W := S_L \cup L^* \cdot M_P^- \cdot S_M \cdot M_P^+ \cdot R^* \cdot S_R$
else $W := S_L \cup L^* \cdot M_P^- \cdot S_M \cdot M_P^+ \cdot R^* \cdot S_R$
\item Alternate_Packing($W$);
\item End.
\end{itemize}

Example 2: Consider the channels shown in Figure 3. We have $n = 7$, $N_1 = (1, 3)$, $N_2 = (1, 5)$, $N_3 = (1, 2)$,

$$N_4 = (5, 0), N_5 = (3, 2), N_6 = (3, 1), N_7 = (0, 1); L = \{N_1, N_2\}, R = \{N_4, N_5, N_6\}, B = \emptyset; S = M_T = \{N_7\}, M_P = \{N_3\}.$$ Since $D = 3 > |L| = 2$ and $t_{R*} = 3 > b_{R*}$, $b_{R*} = 0 < t_{R*} = 8$, we have $S_L = S_R = \emptyset$, and $L$ is sorted into $(N_1, N_2)$, $R$ is sorted into $(N_4, N_5, N_6)$. After decomposing the nets, we obtain

$$W = (N_1', N_2', N_2'', N_3', N_7, N_4, N_6, N_6', N_3', N_5').$$

The channel constructed by our algorithm is shown in Figure 3(a) which achieves optimal density 3. Figure 3(b) shows the channel constructed by the algorithm in [6] which has density equal to 4, one more than the optimal value. The difference is that because the nets in $L^*$ and $R^*$ are decomposed in our algorithm, net $N_3$ does not cross every net in $R$ in the channel constructed by our algorithm, whereas it does in the channel constructed by the algorithm in [6].

5 The Optimality of Our Algorithm

We show in this section that Algorithm Optimal_Packing constructs a channel with minimum density given an instance of the DM problem. We first state the correctness of the algorithm, which follows from the correctness of Algorithm Alternate_Packing.

Theorem 5.1 Algorithm Optimal_Packing computes a valid terminal permutation of a given instance of the DM problem, i.e., each terminal is assigned a unique position and no two terminals are assigned to the same position.

To establish the optimality of the algorithm, we need the following lemmas.

Lemma 5.2 In the channel produced by Algorithm Alternate_Packing, no net in $L^* \cup R^*$ crosses two nets in $M_P$ at some column.
Lemma 5.3 If \( L^* = R^* = \phi \), then Algorithm OptimalPacking produces a minimum density channel.

Lemma 5.4 If \( M_P = \phi \), then Algorithm OptimalPacking produces a minimum density channel.

Based on the above lemmas, we can now show:

Theorem 5.5 Algorithm OptimalPacking computes a minimum density channel.

Since both \( L^* \) and \( R^* \) can be sorted in linear time by bucket sort [1], the running time of the algorithm is easily seen to be linear, as stated in the following theorem.

Theorem 5.6 Algorithm OptimalPacking runs in \( O(1) \) time, where \( l \) is the length of the channel.

6 NP-Hardness Results

In this section, we show that the problem of permuting the terminals of a channel so that it can be routed in minimum wire length is NP-hard in the strong sense by proving the following decision version of the problem is NP-complete in the strong sense.

WireLengthMinimization (WLM)

Instance: A set of nets \( N = \{N_1, N_2, \ldots, N_n\} \) to be routed with \( \sum_{k=1}^n f_k = \sum_{k=1}^n b_k = l \), and a positive integer \( W \).

Question: Is there a permutation of the terminals such that the channel can be routed in total wire length \( \leq W \)?

Theorem 6.1 The WLM problem is NP-complete in the strong sense even for channels with no exits.

We prove Theorem 6.1 by a polynomial time transformation from the following 3-Partition problem, which is known to be NP-complete in the strong sense [9]. We now have

Corollary 6.2 The problem of computing a terminal permutation so that the resulting channel can be routed in minimum wire length is NP-hard in the strong sense even for channels without exits.

References


[18] H.W. Leong, Routing problems in physical design of integrated circuits, Ph.D. Dissertation, Department of Computer Science, University of Illinois at Urbana-Champaign, 1986.


