Abstract

In the traditional gate array technology, cells are placed in an array and wires are routed in channels between the cells. As the advance of VLSI technology, wires are now routed not only in the channels but also over the cells. This is the so called sea-of-gates technology. We study in this paper the global routing of multiterminal nets in the sea-of-gates technology. For an \( m \times m \) array we present a global router that produces the optimal channel width in the 2-direction and guarantees that the channel width in the \( y \)-direction is never larger than \( \min(y, m) \), where \( s^* \) is the largest span of the set of nets (the span \( s \) of a multiterminal net in a two-dimensional array is the length of the largest side of the smallest rectangle enclosing all terminals of that net).

1 Introduction

In the placement-routing approach to the circuit layout, first modules (or cells) are placed in the plane aiming to place “strongly connected” modules next to each other. In the routing step, a topological path is determined for each net (global routing) and then an exact path is specified for each net as dictated by the global routing (detailed routing). In this paper we focus on the global routing problem, for the area of the final layout is strongly dependent on this step. Global routing is known to be NP-complete [5] even if only one-bend routing is allowed [4]. Researchers have proposed heuristic algorithms for global routing in the past two decades. Various approaches have been proposed, for example, hierarchical wiring [1,8], sequential methods [2,7,10,13,12], simulated annealing [14], and one-step techniques [4]. A survey of global routing methodologies appears in [6,12].

In this paper we shall first present an algorithm which achieves optimum global density in one direction and well controls the global density in the other direction. We then present a parallel algorithm which is suitable for the problems in which most nets are “short”. (Most VLSI routing problems have this property since the placement algorithm try to place the modules containing terminals of the same net close to each other.) The key idea of this paper is to transform a global routing problem (GRP) into a channel routing problem (CRP) and to solve it by using channel routers which are rich in literatures. We will omit proofs of theorems in this paper due to the limitation of space (The detailed proofs can be found in [15]).

2 Preliminaries and definitions

The global routing problem for gate-array or sea-of-gates technologies can be modeled as the routing in a two-dimensional array [4,12]. In traditional gate-array technology, wires are routed only in “passive” areas (i.e., region of the plane external to the modules) called channels. In current multilayer technology (e.g., sea-of-gates technology) wires can be routed over the cells, in addition to channels, to achieve a more compact design (Figure 1(a)). Formally, in the global routing of multiterminal nets in a two-dimensional array there is a set \( \eta = \{N_1, \ldots, N_n\} \) of multiterminal nets. The layout environment (plane grid) is a two-dimensional \( m \times m \) grid, which is a square tessellation of the plane. Each net \( N \) is specified by a set of pairs \((x_i, y_i), i = 1, \ldots, k\), where \((x_i, y_i)\) is the tile (cell or module) at whose center a terminal of \( N \) occurs [4]. Hereafter, \( T(i, j) \) denotes a tile with its center at \((i, j)\). A vertical trunk, denoted by \( V(i) \), consists of tiles \( T(i, j), j = 1, \ldots, m \), and a horizontal trunk, denoted by \( H(j) \), consists of tiles \( T(i, j), i = 1, \ldots, m \) (Figure 1(b)).

We assume that there is only one terminal in each tile, since we can always divide the tiles containing more than one terminal into a collection of tiles by means of inserting some columns and rows, each of which contains exactly one terminal.

In a global routing let \( d_v(i, j) \) denote the number of nets crossing the common border of tiles \( T(i, j) \) and \( T(i, j+1) \), \( 1 \leq i \leq m \) and \( 1 \leq j \leq m - 1 \). Similarly, let \( d_h(i, j) \)
3 Minimizing Horizontal Channel Width

In this section, we study the problem of minimizing the horizontal density $d_{h}^{\text{max}}$; simultaneously we aim to bound the vertical density $d_{v}^{\text{max}}$. In an instance of GRP let $d_{h}(i)$ denote the number of nets with one terminal to the left of the right border of the vertical trunk $V(i)$ and the other terminal to the right of the right border of $V(i)$. The horizontal flow of the problem is defined as $d_{h}^{\text{max}} = \max_{i} d_{h}(i)$. The following lemma shows a trivial lower bound on $d_{h}^{\text{max}}$.

**Lemma 1** $d_{h}^{\text{max}} \geq \left\lceil \frac{d_{v}^{\text{max}}}{m} \right\rceil$.

Now we discuss the algorithm which creates a global routing in which $d_{h}^{\text{max}} = \left\lceil \frac{d_{v}^{\text{max}}}{m} \right\rceil$. We define three operations (see Figure 2):

1. **move-up($i$)** is the operation of moving the terminals in tiles $T(i,j)$, $j = m/2 + 1,...,m$, to the upper border of tile $T(i,m)$.

2. **move-down($i$)** is the operation of moving the terminals in tiles $T(i,j)$, $j = 1,...,m/2$, to the lower border of tile $T(i,1)$.

3. **move-back($i$)** is the reverse operation of move-up($i$) and move-down($i$), that is, each terminal is placed at its original position.

![Figure 1: An instance of global routing.](image)

![Figure 2: Transforming a GRP into a CRP.](image)
To construct an instance of CRP from a GRP, we use the move-up and move-down operations to move the terminals inside the tiles to the upper and the lower borders of the array. Then we have a CRP with channel density $\delta_h^\text{max}$. (The detailed proof is omitted here.) Channel routing algorithms achieving channel width $t = \delta_h^\text{max}$ (in the restricted wire overlap model) have been proposed in [3,11,16,15]. Since we are concerned here with the number of nets crossing the borders of the tiles, allowing overlap does not cause any problem. Upon completing the channel routing, we move terminals on the upper and lower borders back to their original positions by using the move-back operation. The result is a global routing of the problem. We call respectively the left and right terminals of a net the starting and ending terminals.

A formal description of the proposed algorithm is given below:

```
Procedure ONE-DIRECTION-GLOBAL-ROUTING (q);
begin
for i = 1 to m do (V(i) is the current vertical trunk*)
begin
Insert m/2 columns into the vertical trunk V(i);
Move-up(i) and move-down(i);
Place ending terminals one by one starting from the leftmost column
of V(i) and place starting terminals one by one starting from the rightmost column
of V(i) on the upper and lower shores, respectively;
Align all the terminals of trivial nets
end;
Call the channel router;
for i = 1 to m do (V(i) is the current vertical trunk*)
begin
Move-back(i);
Connect the nets with both of their two terminals in V(i) by vertical
wire segments from their lower terminals to their upper terminals
and delete the routing of these nets created by the channel router
end;
end.
```

Since both the construction of an instance of CRP from a GRP and the routing of the constructed instance of CRP takes time $O(m^2)$, we claim:

**Theorem 1** The algorithm ONE-DIRECTION-GLOBAL-ROUTING solves an arbitrary instance of multiterminal net GRP achieving $\delta_h^\text{max} \leq \frac{\delta_h^\text{max}}{m}$ and $\delta_v^\text{max} \leq m$, and runs in $O(m^3)$ time.

A result obtained by using this algorithm for the GRP in Figure 1 is shown in Figure 2(c) where $\delta_h^\text{max} = 1$ (the optimum horizontal density). This example is adopted from the reference [4] where $\delta_v^\text{max} = 2$ (Figure 2(d)).

4 A Parallel Global Routing Algorithm

We call the distance between the leftmost terminal and the rightmost terminal of net $N_i$ the horizontal span of $N_i$. Similarly, we call the distance between the lowermost terminal and the uppermost terminal of net $N_i$ the vertical span of $N_i$. The span $s_i$ of $N_i$ is defined as the maximum of its vertical span and its horizontal span. The span of $\eta$ is $s^* = \max_i(s_i)$. We consider instances with a small span $S^*$ in this section. To simplify the discussion, we assume that $\frac{s^*}{S^*}$ is an integer.

Let $B(i,j)$ consist of tiles $T(i + k, j + l), i, j = hs^* + 1, h = 0, \ldots, \frac{m - s^*}{s^*}$, and $k, l = 1, \ldots, s^*$. (For simplicity, we assume $\frac{m - s^*}{s^*}$ is an integer.) Call $B(i,j)$ a block. Let $GH(j)$ consist of blocks $B(i,j), i = 1, s^* + 1, \ldots, m - s^* + 1$. Call $GH(j)$ the global horizontal trunk. Denote by $V(k)B(i,j)$ the portion of the vertical trunk $V(k)$ in a block $B(i,j)$. (Recall that the vertical trunk $V(k)$ consists of the tiles $T(k,j), j = 1, \ldots, m$, which is defined in Section 2.) These definitions are illustrated in Figure 3.

**Figure 3:** Dividing an $m \times m$ array into $m/s^2$ blocks.
We start by inserting \( s^* \) columns into \( V(i), i = 1, 2, \ldots, m \), and apply the move-up operation to the terminals in the upper half and the move-down operation to the terminals in the lower half of all \( V(k)_{B(i)} \). Hence, we have constructed \( m/s^* \) CRPs of routing nets in \( GH(i), i = 1, s^* + 1, \ldots, m - s^* + 1 \). Next, we use the channel router \([9,3]\) to route each \( GH(i) \). Finally, we apply the move-back operation to all \( V(k)_{B(i)} \) to complete the global routing. A formal procedure PARALLEL-GLOBAL-ROUTING\( (\eta) \) for the above described algorithm has been given in \([15]\) and is omitted here due to the limitation of space.

Now we analyze the time complexity of the algorithm. To construct an instance of CRP in \( GH(i) \) (the move-up and move-down operations) it takes time \( O(s^*m) \) since there are \( s^*m \) tiles in \( GH(i) \) and to move a single terminal takes time \( O(1) \). To route \( GH(i) \) it takes time \( s^*m \) since the channel router \([9,3,16]\) uses time \( O(s^*m) \) (\( s^*m \) is the number of columns in the channel). Similarly, the move-back operation in \( GH(i) \) takes time \( O(s^*m) \). There are \( m/s^* \) \( GH(i) \), so that we have the following theorem (if the algorithm is executed sequentially):

**Theorem 2** The algorithm PARALLEL-GLOBAL-ROUTING produces a global routing result with \( d^{max}_h \leq s^* \) and \( d^{max}_v \leq \frac{2}{\eta} \) in time \( O(m^2) \), where \( s^* \) is the span of the problem.

Notice that the construction and the routing of each \( GH(i) \) can be done independently. We then obtain a parallel algorithm using the EREW (exclusive read and exclusive write) model.

**Theorem 3** The algorithm PARALLEL-GLOBAL-ROUTING\( (\eta) \) produces a global routing result with \( d^{max}_h \leq s^* \) and \( d^{max}_v \leq \frac{2}{\eta} \) in time \( O(m^2s^*) \) by using \( \frac{m^2}{s^*} \) processors, where \( s^* \) is the span of the problem.

### 5 Discussion and Open Problem

We have presented algorithms for the global routing in two-dimensional arrays. The algorithm ONE-DIRECTION-GLOBAL-ROUTING\( (\eta) \) achieves the optimum horizontal density and well controls the vertical density in a global routing. This algorithm will also be useful to the standard-cell technology where channel capacity in one direction is much smaller than that in the other one. A parallel global routing algorithm has been proposed for the cases where span \( s^* \) of nets is small. The algorithm uses the EREW PRAM model which is rather practical. This algorithm also has important applications to the design of reconfigurable arrays where one wishes to know the number of spare wires needed if the reconfiguration takes place among modules within the distance \( s^* \). Based on the result of this paper, we know that \( 3s^*/2 \) spare wires for each module will suffice.

### References


