OPRON: A New Approach to Planar OTC Routing

S. Danda, S. Madhwapathy, N. Sherwani and A. Sureka
Department of Computer Science
Western Michigan University
Kalamazoo, MI 49008

Abstract

In this paper, we solve the planar over-the-cell routing problem, in which, nets must have at least one terminal on the boundary. Such nets allow for non-traditional cell designs, where all terminals must be placed on the boundaries, giving a degree of freedom to the cell designer. We present a dynamic programming algorithm that optimally solves this problem, in $O(K^2n^5)$ time, where $K$ is the number of tracks available over the cell for a given cell row region, and $n$ is the number of nets to be routed.

1 Introduction

Channel routing in standard cell layouts is an important problem in VLSI design and as a result it has been studied extensively. Over-the-Cell (OTC) routing is used to minimize the height of the channels. Several Over-the-Cell Routing algorithms for standard cell layouts with different cell models were presented [1, 3, 4, 7]. Generally in two metal layer process, the M1 layer is typically used for connections internal to the cell, and the M2 layer is available for routing over-the-cell. As more metal layers are made available for routing in the standard cell design technology, routing over the cells becomes both practical and important. Traditionally, the terminals were conveniently located at the boundaries (Boundary Terminal Model or BTM) and a planar subset of nets were routed on the OTC area, and the remaining nets were routed in the channels between the cell rows.

Consider the nets $N_1$ and $N_2$ as shown in Figure 1(a). Either $N_1$ or $N_2$, but not both, could be selected for routing over the cell, since $N_1$ and $N_2$ overlap. However, if one of the nets (say $N_1$), has one terminal on the boundary and the other terminal between the boundaries, then both the nets could be selected for OTC routing (Figure 1(b)). We define a Closed Net, as a two terminal net if both of its terminals on the boundary, and an Open Net is a net with only one terminal on the boundary, and the other terminal between the boundaries (Figure 1). Closed Nets have the flexibility of being routed on any of the horizontal tracks over the cell rows, but for open nets the horizontal track is fixed, if we allow only two M2 segments (a horizontal segment and a vertical segment) for routing the net. Hence, open nets can sometimes (but not always) help in routing more nets to be routed over the cell.

Consider the problem of selecting the maximum number of nets that can be routed on the OTC area in M2, in a planar fashion, given an instance of open nets and closed nets in a cell row. We call this problem as Over-the-cell Planar Routing with Open Nets (OPRON). Figure 2 shows an optimal solution to an instance of OPRON. In this paper, we present an optimal algorithm to solve the OPRON problem. The time complexity of our algorithm is $O(K^2n^5)$, where $K$ is the number of tracks over a cell row, and $n$ is the number of nets. The rest of the paper is organized as follows. In Section II we formally define the OPRON problem and present an overview of our algorithm in Section III. In Sections IV and Section V we describe our algorithm in detail and conclude with Section VI.

2 Preliminaries

The routing model for OPRON is similar to the BTM-HCVC model, except for the presence of open nets [2]. Let $C$ be the number of columns numbered
left to right, and \( K \) be the number of tracks in a cell row numbered top to bottom. Let the upper boundary of the cell row be denoted by track 0 and the lower boundary be denoted by track \( K + 1 \). Let \( N \) be the set of nets, \( N_L \) be the set of lower nets and \( N_U \) be the set of upper nets, then \( N = N_U \cup N_L \). \( N_L \) (or \( N_U \)) is the set of closed nets with both the terminals on track 0 (\( K + 1 \)), and open nets, with one terminal on track 0 (\( K + 1 \)) and the other terminal on track \( t \), where \( 1 \leq t \leq K \).

Each net in \( N_U \) is denoted by \( N_U\{i, t_1, j, t_2\} \), where \( i \) and \( j \) are the columns, and \( t_1 \) and \( t_2 \) are the tracks on which the terminals of the nets are located, such that, (1) \( 1 \leq i, j \leq C, i \neq j \), (2) \( 0 \leq t_1, t_2 \leq K \), and (3) \( \min\{t_1, t_2\} = 0 \). Similarly, each net in \( N_L \) is denoted by \( N_L\{i, t_1, j, t_2\} \), such that, (1) \( 1 \leq i, j \leq C, i \neq j \), (2) \( 1 \leq t_1, t_2 \leq K + 1 \), and (3) \( \max\{t_1, t_2\} = K + 1 \).

Given a cell row with \( C \) columns numbered left to right and \( K \) tracks numbered top to bottom, a set of nets \( N = N_U \cup N_L \) as described earlier, the objective of OPRON is to find the maximum planar subset \( N' \subseteq N \) of nets that can be routed in the \( K \) tracks.

We call an instance of the OPRON problem as a Canonical Instance, if each column has at least one terminal located on it.

3 Overview of our Algorithm

The algorithm to compute an optimal solution for the OPRON problem, which we denote by ALG-OPRON, consists of three phases as described below.

Phase 1: In the first phase we compute the single row solutions, for the set of nets \( N_U \) and \( N_L \), respectively, which we denote by \( S_U \) and \( S_L \). Set \( S_U \) is a set of solutions \( S_U(i, j, t) \) \( 1 \leq i, j \leq C, 1 \leq t \leq K \). Each \( S_U(i, j, t) \) solution denotes the maximum planar set of nets \( N_U \subseteq N_U \), that can be routed in a single layer, in the top \( t \) tracks, between the columns \( i \) and \( j \) (inclusive of the columns \( i \) and \( j \)). The set \( S_L \) solutions can also be defined similarly. The computation of single row solutions is described in Section IV.

Phase 2: In the second phase, we compute the two row solutions, which we denote by \( T \), where \( T \) is a set of solutions \( T(j) \) \( 1 \leq j \leq C \). Each \( T(j) \) solution denotes the maximum planar set of nets \( N' \subseteq N \), that can be routed in \( K \) tracks between the track 0 and track \( K + 1 \). The computation of the two row solutions is described in Section V.

Phase 3: In the above two phases, \( S_L \), \( S_U \), and \( T \) consist of the number of nets, but not the actual set of nets. Using the information stored in the tables created during phases 1 and 2, the actual set of nets in the solution to a given instance of OPRON problem is constructed by a backtracking procedure, which is not discussed in this paper, for the sake of brevity.

4 Phase I

In order to compute \( S_L \) or \( S_U \) it is necessary to know the solutions in L-shaped regions as shown in Figure 3. After computing \( S_L(i, j, t) \) we always compute the solutions in all the L-shaped regions in the region bounded by \( i, j, \) and \( t \). Similar to \( S_U \) and \( S_L \), we have \( L_U \) and \( L_L \), where \( L_U \) (or \( L_L \)) is a set of solutions in the L-shaped regions exclusively with \( N_U \) (or \( N_L \)). Each L-shaped solution in \( L_U \) (or \( L_L \)) is represented as \( L_U(i, t_1, m, t_2, j) \) or \( L_L(i, t_1, m, t_2, j) \). We can easily get the order of this notation by traveling on the boundary of the L-shaped region in clockwise (anti clockwise) direction starting from the bottom-left (top-left) corner of that L-shaped region. Based on the type of net with one terminal at column \( j \) we can have 3 different cases. In this section we discuss only the case in which there exists a net \( N_L(m, t', j, K + 1) \) as shown in Figure 3.

In this case \( S_L(i, j, t) \) is computed as follows.

\[
S_L(i, j, t) = \max\{S_L(i, j, t-1), m_{i, j, t, K+1}\} = \max\{L_U(i, t, m, t', j), L_L(i, t, m, t', j)\}.
\]

4.1 Solutions for L-shaped regions with Lower Nets

Consider an L-shaped region bounded by \( (i, t_1, m, t_2, j) \), where \( m < j \). Based on the existence of a net with a terminal on column \( m \), the L-shaped region is partitioned at a column \( r, i \leq r < m \), into sub-regions, which may be L-shaped or rectangular regions, as shown in Figure 4. The solutions of the sub-regions are known, since these solutions were computed earlier. The solution for each partition is the sum of the solutions of the sub-regions. The optimal solution of a given L-shaped region is the maximum of the solutions obtained for all the partitions, as given below. The optimal solutions for L-shaped regions bounded by \( (i, t_1, m, t_2, j) \) can also be computed similarly. Based on the presence of a net, with a terminal at column \( m \), we have six distinct cases, of which we discuss only the most generic cases.

Case 1: If there exists a net \( N_U(p, t', m, K+1) \).

1.1: If \( t' < t_2 \) as shown in Figure 4(a), then

\[
L_U(i, t_1, m, t_2, j) = \max\{L_U(i, t_1, m, t_2, r) + S_U(r, m, t_2 + 1)\}
\]

1.2: If \( t' \geq t_2 \) as shown in Figure 4(b), then

\[
L_U(i, t_1, m, t_2, j) = \max\{L_U(i, t_1, m, t_2, m - 1)\}
\]
Figure 4: L-shaped Regions in Phase I of ALGO-OPRON

\[ \max_{i=1}^{m-1} \{ L_L(i, t_1, j, t' - 1, r) + S_L(r, m, t') \} \]

Case 2: There exists a net \( n_L(p, K + 1, m, K + 1) \) (Figure 4(c)).

\[ L_L(i, t_1, j, t_2, m) = \max_{\tau \in \{i, j, t_2, \tau\}} \{ L_L(i, t_1, j, t_2, \tau) + S_L(r, m, t_2 + 1) \}, \max_{t' = t_2} \{ L_L(i, t_1, j, t' + 1, p - 1) + S_L(p, m, t') \} \]

Case 3: There exists a net \( N_L(m, K + 1, p, t') \) (Figure 4(d)).

\[ L_L(i, t_1, j, t_2, m) = \max \{ L_L(i, t_1, j, t_2, m - 1), L_L(i, t_1, j, t' - 1, m - 1) + 1 \} \]

After computing the solution in an L-shaped region, we perform a backtracking procedure, to determine whether the net at column \( i \) (if it exists), is in the solution. This information is necessary to compute the two row solutions in the next phase. Using a similar approach \( S_U \) and \( S_L \) can be computed.

Lemma 1 The computation of the optimal solution in an L-shaped region takes \( f(K, n) = \max \{ K, n \} \), where \( n \) is the number of nets available and \( n \) represents the number of nets.

Theorem 1 The time complexity for computing \( S_U \) or \( S_L \) is \( O(K^2 n^4) \).

5 Phase II

In order to compute \( T \), the maximum planar subsets in L-shaped regions as shown in (Figure 5), are required. The solutions in the L-shaped regions, bounded by either track 0 or track \( K + 1 \), on one side were already computed in phase I. In this phase, we are required to compute the solutions in the L-shaped regions bounded by both the tracks 0 and \( K + 1 \). Each L-shaped region of the above type is denoted by a 4-tuple \( (i, m, j, t) \), where \( 1 \leq i, m, j \leq L \), and \( 1 \leq t \leq K \), and the solution in this region is denoted by \( L_T(i, m, t, j) \). The order of the notation can be obtained by starting at track 0 of column \( i \), and moving along the boundary of the L-shaped region, in the clockwise direction. After computing each \( T(j) \), we will also compute the \( L_T(i, m, t, j) \) solutions, where \( 1 \leq i, m \leq j \).

Figure 5: Phase II of ALGO-OPRON

Based on the presence of nets with one of their terminals at column \( j \), we can have 15 different cases. In this section we shall discuss only the case in which, there are two open nets \( N_L(i, t_2, j, K + 1) \) and \( N_U(m, t_1, j, 0) \), where \( 1 \leq i, p < j \), and \( 0 \leq t_1 \leq K \) and \( 1 \leq t_2 \leq K + 1 \).

In this case, the \( T(j) \) solution can be computed as follows.

A: Let \( T_a(j) \) be the \( T(j) \) solution such that, none of the nets \( N_L(i, t_1, j, K) \) or \( N_U(p, t_2, j, 0) \), are in the optimal solution.

B: Let \( T_b(j) \) be the \( T(j) \) solution such that, only the net \( N_L(i, t_1, j, K + 1) \) is in the optimal solution.

C: Let \( T_c(j) \) be the \( T(j) \) solution such that, only the net \( N_U(p, t_2, j, 0) \) is in the optimal solution.

D: Let \( T_d(j) \) be the \( T(j) \) solution such that, both the nets \( N_L(i, t_1, j, K + 1) \) and \( N_U(p, t_2, j, 0) \), are in the optimal solution. After computing the solutions \( T_a(j), T_b(j), T_c(j), \) and \( T_d(j) \), the optimal \( T(j) \) solution is given by

\[ T(j) = \max \{ T_a(j), T_b(j), T_c(j), T_d(j) \} \]

Now, we shall describe the method of computing the \( T(d) \) solution. We partition the rectangular region corresponding to \( T(j) \), as shown in Figure 5(a). Let \( TEMPL_1(j, t_1, x) \) be the maximum planar subset in the L-shaped region \( (1, j, t_1, x) \), which includes the net \( N_U(m, t_1, j, 0) \). \( TEMPL_1 \) can be computed by partitioning \( (1, j, t_1, x) \) at a column \( z \), where \( z \leq x \leq m - 1 \), into taking the maximum among the solutions for all the partitions, as given below.

\[ TEMPL_1(j, t_1, x) = \max_{z \in \mathbb{Z}} \{ L_T(1, z - 1, t_1, x) + \]
The $T_d(j)$ solution for this type of partitioning, which we denote by $T_1(j)$, is the sum of the solutions of $TEMPL$ and the other sub-regions shown in Figure 5(a), is given by,

$$T_1(j) = \max_{x = m-1, y = i-1} \{L_L(y, t_2, i - 1, t_2 + 1, j - 1) + L_L(x, t_1 + 1, t_1, t_1, t_2 - 1, y) + TEMPL(j, x, t_1) + 1\}$$

However, there is a possibility that a net $N_L(p, t'_1, x, t'_2)$, $1 \leq p < x, 0 \leq t'_1, t'_2 < K$ in the $TEMPL$ solution, overlaps with a net $N_L(x, t'_3, q, t'_4)$, $x < q \leq y, t_1 < t'_3, t'_4 < K$, in $L_L(x, t_1 + 1, 1, t_2 - 1, y)$. The existence of the net $N_L(x, t'_3, q, t'_4)$ in the $L_L$ solution, is determined in Phase I. The existence of the net $N_L(p, t'_1, x, t'_2)$ in the $TEMPL(j, x, t_1)$, can be determined from $TEMPL(j, x, x - 1)$. In this case, the $T_1(j)$ solution is decremented by one, and after the backtracking phase, any one of the overlapping nets is deleted from the solution. $TEMPL$ is recomputed whenever the value of $x$ changes. Figure 5(b) shows a different type of partitioning, which is symmetric to the partitioning shown in Figure 5(a). The $T_d(j)$ solution for this type of partitioning, which we denote by $T_2(j)$, can be similarly computed. The optimal $T_d(j)$ solution is given by,

$$T_d(j) = \max\{T_1(j), T_2(j)\}$$

5.1 Solutions for L-Shaped Regions with Upper and Lower Nets

Consider an L-shaped region $(i, m, j, t)$. Based on the type of nets at columns $j$ and $m$, as shown in Figure 6, we have 15 different cases. In this section we will discuss a few typical cases.

![L-shaped Regions in Phase II of ALGO-OPRON](image)

Suppose there are two nets $N_L(m, 0, p, t_1)$ and $N_L(q, t_2, j, K + 1)$ as shown in Figure 6(a). The optimal solution is computed by partitioning the L-shaped region at column $x$ and track $t'$, where $i + 1 < x \leq m$ and $t + 1 < t' \leq K$, and taking the maximum among the solutions for all the partitions, as given below. The column $x$ is moved from right to left during partitioning, such that, smaller L-shaped solutions can be used to compute larger L-shaped solutions. $L_d(i, m, t, j)$ is given by,

$$L_d(i, m, t, j) = \max_{z = m-1, t' = t+1} \{L_L(i, z, t', j, i) + L_L(j, t' + 1, i + 1, t, j)\}$$

Any conflicts that occur due to overlapping nets, are resolved by the same approach described for the $T(j)$ solutions.

Theorem 2 ALGO-OPRON optimally solves the OPRON problem in $O(K^2n^4)$ time.

6 Conclusions

In this paper, we have presented an optimal algorithm for the planar over-the-cell routing problem with open nets (OPRON). We get an optimal solution for the OPRON problem using dynamic programming in $O(K^2n^4)$ time, where $K$ is the number of tracks available in the over the cell region and $n$ is the number of nets. Currently we are working on several variants of the OPRON problem, such as OPRON problem in the presence of obstacles and when doglegs are allowed for nets.

References


