MCFRoute: A Detailed Router Based on Multi-Commodity Flow Method*

Xiaotao Jia[1], Yici Cai[1], Qiang Zhou[1], Gang Chen[2], Zhuoyuan Li[2], Zuowei Li[2]

Tsinghua National Laboratory for Information Science and Technology
[1]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[2]Nimbus Automation Technologies, Shanghai, China
jxt11@mails.tsinghua.edu.cn, {caiyc, zhouqiang}@mail.tsinghua.edu.cn, {chengang, lizhuoyuan, lizuowei}@nimbus-da.com

ABSTRACT

Detailed routing is an important stage in VLSI physical design. Due to the high routing complexity, it is difficult for existing routing methods to guarantee total completion without design rule checking violations (DRCs) and it generally takes several days for designers to fix remaining DRCs. Studies has shown that the low routing quality partly results from non-optimal net-ordering nature of traditional sequential methods. In this paper, a novel concurrent detailed routing algorithm is presented that overcomes the net-order problem. Based on the multi-commodity flow (MCF) method, detailed routing problem with complex design rule constraints is formulated as an integer linear programming (ILP) problem. Experiments show that the proposed algorithm is capable of reducing design rule violations while introducing no negative effects on wirelength and via count. Implemented as a detailed router following track assignment, the algorithm can reduce the DRCs by 38%, meantime, wirelength and via count are reduced by 3% and 2.7% respectively comparing with an industry tool. Additionally, the algorithm is adopted as an incremental detailed router to refine a routing solution, and experimental results show that the number of DRCs that industry tool can't fix are further reduce by half. Utilizing the independency between subregions, an efficient parallelization algorithm is implemented that can get a close to linear speedup.

Categories and Subject Descriptors

B.7.2 [**Integrated Circuits**]: Design Aids

Keywords

Detailed Routing, Multi-commodity Flow, Design Rule, DRC

*This work was supported by Natural Science Foundation of China (NSFC) No.61274031.

1. INTRODUCTION

Routing is a very important and the most time-consuming stage in modern VLSI physical design. The top-level design may contain millions of nets in a typical hierarchical design. It generally takes days for EDA tools to finish routing, close timing and fix DRCs on a powerful computing server with multi-threading accelerations. However, due to the NP-complete nature of the routing problem, these industry tools hardly guarantee total completion of the DRC-clean connections while satisfying the timing constraints. Designers need to spend much more time to do ECO changes to meet both timing and DRC closure due to the inherent limitations of existing techniques.

Industry tools use divide-and-conquer mechanism to reduce the complexity of the routing problem. A global router [1], combined with or followed by a track assignment [2] engine, is used to obtain the coarse solution on a global routing graph. The routing segments are assigned to different routing tracks. With the guidance of track assignment results, detailed router allocates interconnects to connect the net terminals satisfying all the physical constraints.

Sub-micron technologies generally use multi-layer routing, where metal wires are manufactured in six to twelve routing layers [3]. Each routing layer has a preferred routing direction and adjacent layers are connected by vias. The layout is partitioned into many subregions after global routing and track assignment. Detailed router works on the three-dimensional routing space inside each subregion to connect all the nets. The primary task for the detailed router is to make sure there are no LVS errors (net opens or shorts), while satisfying all the complex design rules. The secondary task for the detailed router is to optimize circuit performance, without detouring timing critical nets while preserving the non-default spacing requirements for delay.

As the decreasing of feature size, more and more design rules are introduced to guarantee performance, manufacture and yield [4]. The explosion of design rule makes detailed routing more difficult. Assigning routing resource for every net reasonably becomes more and more important. It has been found both in industry and academia that the the worse routing solution partly arises from non-optimal net-ordering nature of traditional sequential methods, even if different ordering strategies are employed.

This paper proposes a novel concurrent detailed routing algorithm based on MCF model routing all the nets simultaneously. The main contributions of this paper are listed as follows:

- The proposed router (MCFRoute) generates paths for all nets in each routing region simultaneously. As a result, it overcomes the net-ordering problem and can get the optimal grid-based solution in each routing region[1].

- Besides fixing LVS errors, our detailed router supports complex spacing rules. Some other design rules, like minarea rules are also supported in our routing engine. Our router is proven to be effective in supporting 28nm above technology nodes.

- We also propose several techniques to accelerate the basic MCF routing engine. A multi-threaded framework is implemented for the MCF routing engine and it can get a close to linear speedup on the multi-core computing server.

- Experimental results show that compared with the state-of-the-art industry tool, the MCFRoute is able to obtain solution with less DRCs while get better wirelength and via count in a reasonable runtime. Moreover, it can also be used as a incremental router to fix some hard DRC violations which the industry tool cannot fix.

The remainder of this paper is organized as follows. Section 2 describes previous works and the motivation of our work. The basic MCF model is formulated in section 3. In section 4, some complex design rules are added to the basic model. Section 5 describes several strategies optimizing MCF model. Experimental results are demonstrated in section 6 and section 7 is the conclusion.

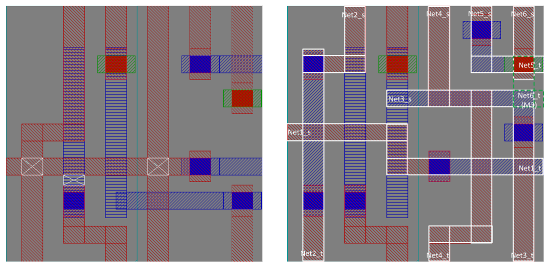
2. PREVIOUS WORK AND MOTIVATION

Detailed routing models can be classified as grid-based and gridless. For grid-based detailed routing model, detailed router finds legal routing path on the given grids. Lee's algorithm [5] is the most widely used grid-based algorithm to find a shortest path for two terminals. Line-search algorithm [6] performs a depth-first-search approach to find the routing path using line segments. Another efficient detailed routing algorithm is A*-search [7]. Some gridless detailed routers are proposed in previous works, such as [8][9]. Recent years, Zhang proposes a RegularRoute [10] for detailed routing. Work in [11] proposes a GDRouter in which FastRoute [12] and RegularRoute [10] are interleaved. Michael et al. [13] presented a detailed router using two efficient data structures of shape grid and fast grid.

Existing research works are all designed on a sequential manner, where all the nets are routed one-by-one by a prefixed order. In the ripup-reroute iterations, the net orders may change due to current DRC and historical penalties. However, the routing engine works on a single net and all other nets are fixed as routing obstacles when searching for the new path for current net. This sequential behavior suffers from the net-ordering problem, which has great impact on routing quality.

In practical routing problem, those additional iterations and heuristic strategy always fail to find a valid solution as

[1]The MCF model requires all wires and vias are placed on routing grids. For gridless cell pins, it is connected through on-grid pin-access-virtual-vias.



(a) Encounter result (b) Our result

Figure 1: Influence of net ordering on routability

shown in Fig.1 for certain reasons, for instance, the increasingly larger scale of the routing problem. In Fig.1(a), the routing result is generated by a commercial EDA tool of Cadence Encounter. After several optimization iterations, there are still three design rule violations that can not be removed. In Fig.1(b), a routing result without any design rule violation is generated by MCFRoute which considers all nets in the region simultaneously. Comparing Fig.1(a) and Fig.1(b), we find that a reasonable solution can be obtained by adjusting the layouts of six nets in highlighted areas. But it is difficult for sequential strategy to find the DRCs-clean solution. This motives us to develop a concurrent algorithm for detailed routing problem.

MCF problem is a network flow problem with multiple flow demands between different source and sink nodes. Many problems in VLSI design are modeled and solved as MCF problem like global routing [14] and escape routing [15] to achieve a concurrent manner.

3. MULTI-COMMODITY FLOW MODEL

In this section, we will introduce our MCF model which is designed to solve detailed routing problem. Inspired by [14], the basic model described in this section is capable of finding routing solution without opens and shorts based on MCF theory.

3.1 Preliminary

Due to the large scale, the layout of modern chip is divided into many small routing regions after track assignment. A net in netlist is generally divided into many sub-nets located at different small regions (hereafter, we will refer to net in netlist as *logic net* and sub-net in small routing region as *net*). The intersection point of segment and the boundary of a small routing region is usually named as *crosspoint*. Both the terminals of logic net which we refer to as *pins* below, and *crosspoints* in a small routing region are the *components* of the net.

Based on MCF theory, detailed routing problem is modeled as a paths finding problem on a $3-D$ *routing graph* which is named as $G = (V, E)$. Here, $V = \{v_1, v_2, \cdots, v_n\}$ is a set of n vertices of graph G, and $E = \{e_1, e_2, \cdots, e_m\}$ is a set of m edges of graph G. Vertices of G on each layer are the intersection points of routing grids on current layer with grids projected from neighbor routing layers. Then routing grids are divided as edges of G by vertices. In MCF model, graph G is defined as a directed graph, which means there are two edges between adjacent vertices v_{j_1} and v_{j_2}; one is from v_{j_1} to v_{j_2}, while the other one is from v_{j_2} to v_{j_1}. We regard these two edges as *brother edges*. Fig.2(a) shows

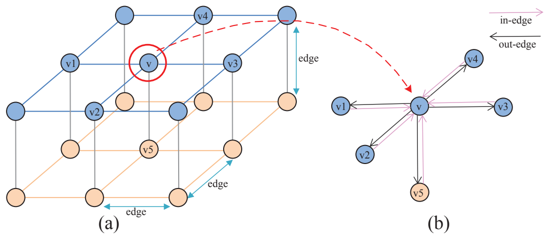


Figure 2: Detailed routing graph

a simple routing graph with 2 routing layers, 66 directed edges and 18 vertices.

On detailed routing graph, each vertex v_j is a terminal point of a set of edges E_{v_j} which contains no more than 12 elements as shown in Fig.2(b). Each set E_{v_j} could be divided into two subsets , named $E_{v_j,out}$ and $E_{v_j,in}$ with the same scale. All the edges in $E_{v_j,out}$ start from v_j and are named as out-edges of v_j, while all the edges in $E_{v_j,in}$ end at v_j and are named as in-edges of v_j. In Fig.2(b), vertex v has 5 in-edges that are marked with a pink color and 5 out-edges with black color. From the assumption given above we can reach the following conclusion:

$$|V| = \sum_{l=1}^{L} R_l C_l$$

$$|E| = 2 * \left(\sum_{l=1}^{L} (2R_l C_l - R_l - C_l) + \sum_{l=1}^{L-1} R_l C_l \right)$$

Here L is the total number of routing layer; R_l and C_l is the row count and column count of layer l respectively.

Under the definition of routing graph, the detailed routing problem containing a set of $N = \{n_1, n_2, \cdots, n_K\}$ nets that need to be routed can be formulated as assigning K paths for all nets while optimizing objective O1 and satisfying constraints C1-C4:

O1) Minimizing the total routing cost.

C1) All components of each net are connected by one path.

C2) There are no intersection segments between any two paths or between path and routing blockage.

C3) There are no intersection points between any two paths or between path and routing blockage.

C4) There are no design rule violations.

3.2 MCF Based Model for Basic Purpose

A basic MCF model which can get a minimal cost solution without net opens and shorts is proposed in this section.

A. Special Case

For the ease to discussion, our basic MCF model considers a special case first with the following two assumptions:

A1) Each component only covers one vertex.

A2) Each net only has two components.

Naturally, each net in N is treated as a commodity with command of unit flow in MCF model. For each net, one component is treated as source and the other one is treated as target and the unit flow is shipped from source to target.

Table 1: Notions of MCF Model

symbol	variable meaning	value range
$u(e_i)$	the capacity of edge e_i	{0, 1}
$c(e_i)$	the cost of edge e_i	positive number
$d(k, v_j)$	the flow command of vertex v_j that commodity n_k demands	{-1, 0, 1}
$f(k, e_i)$	flow of commodity n_k by edge e_i	{0, 1}

Some basic variables related to routing graph are given in table 1. We use these variables to model detailed routing problem through MCF theory.

Detailed routing requires that each edge of routing graph is occupied by one object at most, so edge capacity should be 1 if it isn't occupied by routing blockage, otherwise, is 0.

Edge cost can be any positive real number and the cost values are different with different edge types. All edges of routing graph G fall into two categories, namely, *via-edge* and *wire-edge*. *Via-edge* connects two vertices on different layers and *wire-edge* connects two adjacent vertices on the same layer. Considering the prefer direction of routing layer, wire-edge is further divided into *prefer-edge* and *nonprefer-edge* based on edge direction.

The value of $d(k, v_j)$ is determined by vertex type. If vertex v_j is occupied by a component of net n_k and the component is treated as the source, then $d(k, v_j) = 1$. If vertex v_j is a component of net n_k while the component is treated as the target, then $d(k, v_j) = -1$. Otherwise, $d(k, v_j) = 0$. It is noteworthy that the assignment of source and target will not bring any effects to the result because of the symmetry of routing graph. In the experiments, the first component read in from database is assigned as source.

$f(k, e_i)$ is the decision variable with value range of 0 and 1. $f(k, e_i) = 1$ means edge e_i is occupied by net n_k. All the constraints and objective function are formulated by these variables.

Based on the formulation of detailed routing problem in section 3.1, our MCF-based detailed routing algorithm is introduced as follows.

Firstly, we give connectivity constraints on all vertices for each net based on flow conservation theory to satisfy constraint C1. For the sake of convenient, two variables are introduced. $f_{out}(k, v_j)$ and $f_{in}(k, v_j)$ indicate the total flow of vertex v_j that net n_k stream in and stream out respectively. They can be calculated by the decision variables related with vertex v_j and net n_k as equation (1) and equation (2):

$$f_{out}(k, v_j) = \sum_{e \in E_{v_j,out}} f(k, e) \tag{1}$$

$$f_{in}(k, v_j) = \sum_{e \in E_{v_j,in}} f(k, e) \tag{2}$$

Flow conservation theory requires that the difference of $f_{out}(k, v_j)$ and $f_{in}(k, v_j)$ must be equal to $d(k, v_j)$. The connectivity constraint based on flow conservation theory is formulated as equation (3).

$$f_{out}(k, v_j) - f_{in}(k, v_j) = d(k, v_j) \tag{3}$$

Then, edge capacity constraint and vertex capacity constraint are introduced to satisfy C2 and C3 which can guarantee the routing solution given by MCF model without short violations. The capacity of each routing edge is at most one in detailed routing problem and the flow each net demands is also one. So equation (4) can ensure constraint C2.

$$\sum_{k=1}^{K} \left(f\left(k, e_i\right) + f\left(k, \bar{e_i}\right)\right) \leq min\{u\left(e_i\right), u\left(\bar{e_i}\right)\} \tag{4}$$

Here, e_i and $\bar{e_i}$ are brother edges.

Even through edge capacity constraint is modeled, routing solution with short violations still exists in the solution space of MCF model. Taking a simple example, net A goes through vertex v_j using two vertical direction edges, at the same time, net B goes through the same vertex using two horizontal direction edges. All edges are only used once, but there is a short violation at vertex v_j. To avoid this kind of violations, vertex capacity constraints as equation (5) are added to MCF model.

$$\sum_{k=1}^{K} \sum_{e_i \in E_{v_j}} f(k, e_i) \leq 2 \tag{5}$$

Combining equation (3), (4) and (5), the following conclusions can be obtained:

Conclusion1: $\sum_{e_i \in E_{v_j}} f(k, e_i) = 2 \text{ or } 0, \ \forall n_k \in N.$

Conclusion2: $\sum_{k=1}^{K} \sum_{e_i \in E_{v_j}} f(k, e_i) = 2 \text{ or } 0$

The total cost of detailed routing problem can be calculated by the equation (6).

$$Cost = \sum_{k=1}^{K} \sum_{e \in E} \left(f(k, e) \cdot c(e)\right) \tag{6}$$

With different edge cost assignment, we can achieve different routing goal. In our model, prefer-edges are encouraged to be used by assigning a smaller weight, and assigning a larger cost to via-edge in order to minimize via count.

So far, our basic MCF model of detailed routing problem is completed as follows. Connectivity constraints ensure there are no open nets, and capacity constraints eliminate the solutions with short violations. All the solutions in solution space of MCF model are reasonable if only opens and shorts are considered. The best solution with minimal cost is selected by the objective function.

With the formal description of constraints and objective above, the detailed routing problem can be formulated as the following ILP problem:

$$\min \quad Cost = \sum_{k=1}^{K} \sum_{e \in E} \left(f(k, e) \cdot c(e)\right)$$

$$\begin{aligned}
s.t. \quad & f_{out}(k, v_j) - f_{in}(k, v_j) = d(k, v_j) && \forall v_j \in V, \forall n_k \in N \\
& \sum_{k=1}^{K} f(k, e_i) \leq u(e_i) && \forall e_i \in E \\
& \sum_{k=1}^{K} \sum_{e \in E_{v_j}} f(k, e) \leq 2 && \forall v_j \in V \\
& f(k, e_i) \in \{0, 1\} && \forall e_i \in E, \forall n_k \in N
\end{aligned} .$$

B. General Case

Now we need to consider the general cases without assumptions A1 and A2.

It is common that several vertices are covered by the same component, especially if the component is a pin of net. Furthermore, in order to handle the complexity caused by irregular pins, *pinaccesses* are introduced in our model to guide the access point of pins. As Fig.3 illustrates, component A (blue polygon) is a pin and locates at METAL1, pinaccesses of this pin are created on METAL2 with determined via rotation. During path finding, when the path hits pinaccess, a via with determined rotation is created automatically.

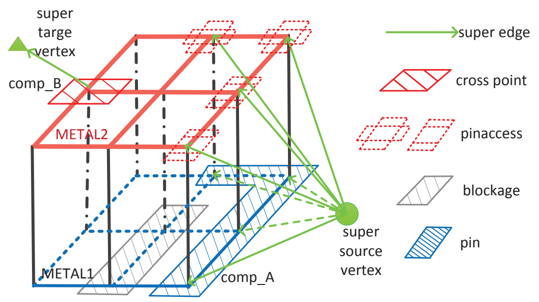


Figure 3: Pinaccess and Super Vertex.

Fig.3 describes a detailed routing example on $3-D$ routing graph. Component A is a pin which covers 4 vertices and has 4 pinaccesses while component B is a crosspoint on boundary that covers one vertex. Without loss of generality, component A is regarded as source and component B is target. Based on MCF theory, one commodity flow must outflow from any one vertex which is covered by Component A or it's pinaccess, and inflow to the vertex that Component B covers.

A strategy of super vertex is proposed to make our basic model capable of handling this general case. First we make some extension to routing graph. Two virtual vertices SV_{com_A} and SV_{com_B} are added to the routing graph that are regarded as the *super source vertex* and the *super target vertex* respectively. Source vertex and target vertex are denoted by green circle and triangle respectively in Fig.3. These two super vertex are connected with corresponding graph vertices through *super edges* denoted by green arrow line. With the extensional routing graph, we generalize the basic model. Assuming that V_{com_A} denotes the set of vertices relevant to component A and SE_{com_A} denotes the set of super edges relevant to super vertex SV_{com_A}. Each super edge $e' \in SE_{com_A}$ introduces a binary variable $g(e')$ to MCF model. Combining with connectivity constraints, the constraint shown in equation (7) ensures that the stream outflow from one and only one vertex in SV_{com_A}.

$$\sum_{e' \in SE_{com_A}} g(e') = 1 \tag{7}$$

Before proposing strategy for assumption A2, we made a survey about components count distribution of net with different scale of routing window in some academic and industry designs. The statistical results show that nets with 2 components accounts for about 75% of total nets and nets with 3 components is about 20%. Nets with more than 3 components (hereafter, referred to as multi-component net)only accounts for 5%. Guided by this statistics, 3-component net is decomposed into 3 subnets using the steiner point in order to accommodate MCF router and assign routing priority to multi-component nets. In the routing flow, multi-component nets are routed first by Mazerouter, then other nets are routed by MCF router using remainder routing resource.

4. COMPLEX DESIGN RULE MODELING

In deep sub-micron technology nodes, there are many other complex design rules besides the LVS checking. The routing paths generated by the detailed router have to pass the DRC checking as well. These design rules need to be modeled correctly and efficiently in the basic MCF model. In this section, we enumerate some of these design rules' modeling in MCF model.

4.1 Spacing Rule

There are different types of spacing rules in typical 45/28nm technology, including metal spacing, end-of-line (EOL) spacing, cut to metal spacing, cut to cut spacing, etc. Most of them can be modeled in the MCF problem using similar method.

To describe spacing rule constraints, we define two disjunction variables $\varphi(k, v_j)$ and $\eta(k, v_j)$ as equation (8) and (9) respectively, for all $n_k \in N$ and $v_j \in V$.

$$\varphi(k, v_j) = \begin{cases} 1 & \text{if } \exists e_i \in E_{v_j} \text{ and } f(k, e_i) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

$$\eta(k, v_j) = \begin{cases} 1 & \text{if } \exists e_i \in E^{\eta}_{v_j} \text{ and } f(k, e_i) = 1 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

Here, $E^{\eta}_{v_j}$ denotes a set of via edges of vertex v_j. $\varphi(k, v_j)$ indicate whether net n_k occupies vertex v_j. $\eta(k, v_j)$ indicate whether net n_k occupies the via edge of vertex v_j.

For two vertices $v_{j1} \in V$ and $v_{j2} \in V$ on the same metal plane, let (x_1, y_1, z) and (x_2, y_2, z) be the coordinate of v_{j1} and v_{j2}. The distance of them is defined as:

$$\Gamma(v_{j1}, v_{j2}) = \max(|x_1 - x_2|, |y_1 - y_2|)$$

A. Metal Spacing Rule

Basic spacing rule specifies the minimum distance allowed between two geometries of different nets on metal layer [16].

On metal layer, there are two type geometries of metal wire and via enclosure. Let α_m, β and γ_m be the value of metal wire width, via enclosure length and metal spacing specified in the technology file respectively. Given any one vertex pair $(v_{j1},\ v_{j2}) \in (V \times V)$ $(j1 \neq j2)$ on the same layer, $\Gamma(v_{j1}, v_{j2})$ can be classified as following 4 cases.

Case1: No violation. If the distance of v_{j1} and v_{j2} satisfies

$$\alpha_m + 2\beta + \gamma_m \leq \Gamma(v_{j1}, v_{j2}),$$

v_{j1} and v_{j2} can be occupied by either geometry type and no spacing violation will emerge, as shown in Fig.4(a). So we do not need to design any constraints for these vertex pair.

Case2: Enclosure-to-enclosure violation. If the distance of v_{j1} and v_{j2} satisfies

$$\alpha_m + \beta + \gamma_m \leq \Gamma(v_{j1}, v_{j2}) < \alpha_m + 2\beta + \gamma_m,$$

v_{j1} and v_{j2} can't be occupied by two vias at the same time, otherwise, there will be a spacing violation, as shown in Fig.4(b). So constraint (10) should be added to MCF model to prohibit this routing solution.

$$\eta(k_1, v_{j1}) + \eta(k_2, v_{j2}) \leq 1, \forall n_{k_1}, n_{k_2} \in N, k_1 \neq k_2 \tag{10}$$

Case3: Wire-to-enclosure violation. When the distance of v_{j1} and v_{j2} satisfies

$$\alpha_m + \gamma_m \leq \Gamma(v_{j1}, v_{j2}) < \alpha_m + \beta + \gamma_m,$$

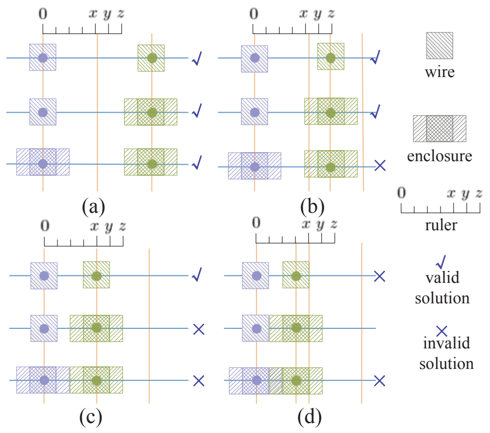


Figure 4: Four cases of $\Gamma(v_{j1}, v_{j2})$. The x, y and z on the ruler are $\alpha_m + \gamma_m$, $\alpha_m + \beta + \gamma_m$, $\alpha_m + 2\beta + \gamma_m$ respectively.

if v_{j1} and v_{j2} are used at the same time, the type of both geometries should be metal wire as Fig.4(c) shows. Constraints (12) and (11) need to be introduced to MCF model.

$$\varphi(k_1, v_{j1}) + \eta(k_2, v_{j2}) \leq 1, \forall n_{k_1}, n_{k_2} \in N, k_1 \neq k_2 \tag{11}$$

$$\eta(k_1, v_{j1}) + \varphi(k_2, v_{j2}) \leq 1, \forall n_{k_1}, n_{k_2} \in N, k_1 \neq k_2 \tag{12}$$

Case4: Wire-to-wire violation. When the distance of v_{j1} and v_{j2} only satisfies

$$\Gamma(v_{j1}, v_{j2}) < \alpha_m + \gamma_m,$$

even if these two vertices are occupied by two metal wire of different nets, there will be a short violation as shown in Fig.4(d). In this case, the MCF model disallow v_{j1} and v_{j2} to be used by different net simultaneously with constraint (13).

$$\varphi(k_1, v_{j1}) + \varphi(k_2, v_{j2}) \leq 1, \forall n_{k_1}, n_{k_2} \in N, k_1 \neq k_2 \tag{13}$$

B. EOL Spacing Rule

The EOL spacing rule ensures that Optical Proximity Correction (OPC) can be performed without interference between the OPC shapes added at the EOLs [16]. The line-end of metal wires (EOL-wires) generally require larger spacing to separate themselves from the neighbor wires. If the line-end is within some distance to via cut, it becomes an EOL-via and has another spacing requirement. The spacing requirements for EOL-wires and EOL-vias are modeled in our MCF problem as well. Given a pair of vertices v_{j1} and v_{j2}, similar to metal spacing rule, designing EOL spacing also need to check which case v_{j1} and v_{j2} falls into based on $\Gamma(v_{j1}, v_{j2})$, α_m, β and γ_{eol}, here γ_{eol} is the EOL spacing defined in the technology file. Then relevant constraint(s) are introduce to MCF model.

C. Cut-to-cut Spacing Rule

Cut spacing rule specifies the minimum spacing allowed between different via cuts [16]. Let α_c and γ_c be the cut width and cut spacing value in technology file respectively. If the distance of the given vertex pair $(v_{j1},\ v_{j2})$ makes the following inequality to be true, constraint (10) will be added to MCF model.

$$\Gamma(v_{j1}, v_{j2}) < \alpha_c + \gamma_c,$$

4.2 Minarea Rule

Minarea is another important design rule, which specifies the minimum metal area required for polygons on the routing layer [16]. To satisfy this rule, the routing path segment on each layer needs to be longer than some threshold. Modeling this rule in the MCF problem directly will generate too many constraints which makes the runtime unacceptable. As a result, we handle this rule separately in a post-processing step:

After MCF model decides the routing paths for each net, the router creates the metal wires and vias on each layer based on the result, during which it checks whether the polygon on each layer is larger than the minimal area requirement. If not, the router will extend that polygon to meet the Minarea requirement. If there is no DRC clean solution, a quick ripup-reroute engine is invoked to fix the min area violations. Experimental results show that less than 1% routing regions need this post-routing cleanup step.

5. MCF MODEL OPTIMIZATION

Given the MCF model for the detailed routing problem, an ILP solver [17] is used to solve the MCF problem. Many efforts are done to reduce the runtime.

5.1 Linearize MCF Model

The spacing constraints in section 4.1 introduce many logic expressions which make the basic model very complex and nonlinear. Several methods are introduced here to optimize those constraints with better attributes and the same results by equivalent transformation.

(*Transformation* 1) Constraint (13) is equivalent to new constraint (14).

$$\sum_{k=1}^{K}\left(\sum_{e_i\in E'_{v_{j1}}} f(k,e_i)+\sum_{e_i\in E'_{v_{j2}}} f(k,e_i)\right)\le 2 \tag{14}$$

where, $E'_{v_{j1}} = E_{v_{j1}}\backslash\{(v_{j1},v_{j2}),(v_{j2},v_{j1})\}$ and
$E'_{v_{j2}} = E_{v_{j2}}\backslash\{(v_{j1},v_{j2}),(v_{j2},v_{j1})\}$

prove: let $F(k) = \sum_{e_i\in E'_{v_{j1}}} f(k,e_i)$ and $F'(k) = \sum_{e_i\in E'_{v_{j2}}} f(k,e_i)$.

(13) $\Rightarrow$ (14): If routing solution satisfies constraint (13), there are 3 valid cases.

1) Neither of two vertices are occupied, i.e. $\varphi(k,v_{j1})=0$ and $\varphi(k,v_{j2})=0$ $\forall n_k\in N$, which mean $F(k)=0$ and $F'(k)=0$, so (14) is true.

2) Either v_{j1} or v_{j2} is occupied. Without loss of generality, we suppose v_{j1} is occupied by net $n_{k'}$ i.e. $\varphi(k',v_{j1})=1$ and $\forall n_k\in N, k\neq k', \varphi(k,v_{j1})=0, \varphi(k,v_{j2})=0$. In this case, $\sum_{k=1,k\neq k'}^{K}(F(k)+F'(k))+F'(k')=0$ and $F(k')=2$, (14) is also true.

3) v_{j1}, v_{j2} and edge (v_{j1},v_{j2}) are occupied by the same net i.e. $\exists k', \varphi(k',v_{j1})=1, \varphi(k',v_{j2})=1$, and $\forall n_k\in N, k\neq k', \varphi(k,v_{j1})=0, \varphi(k,v_{j2})=0$.

In this case, $\sum_{k=1,k\neq k'}^{K}(F(k)+F'(k))=0$, and $F(k')+F'(k')+2f(k',(v_{j1},v_{j2}))=4$. Because $f(k',(v_{j1},v_{j2}))=1$, (14) is still true.

(14) $\Rightarrow$ (13): (14) $\Rightarrow$

1) $\sum_{k=1}^{K}F(k)=0$ and $\sum_{k=1}^{K}F'(k)=0\Rightarrow$ (13).

2) $\sum_{k=1}^{K}F(k)=2$. $\sum_{k=1}^{K}F'(k)$ must be 0. Considering *conclusiton1*, $\Rightarrow \exists n_{k'}\in N$, $F(k')=2$ and $\forall n_k\in N, k\neq k'$, $F(k)=0\Rightarrow$ (13).

3) $\sum_{k=1}^{K}F'(k)=2$. The proof is similar to 2).

4) $\sum_{k=1}^{K}F(k)=1$ and $\sum_{k=1}^{K}F'(k)=1$. Considering *conclusion2*, $\Rightarrow$ edge (v_{j1},v_{j2}) is used, and v_{j1} and v_{j2} is occupied by the same net and only by this net. So $\Rightarrow$ (13) is correct.

(*Transformation*2) Constraint (11) and (12) are equivalent to new constraint (15) and (16) respectively.

$$\sum_{k=1}^{K}\left(\sum_{e_i\in E^{\eta}{}_{v_{j1}}} f(k,e_i)+\sum_{e_i\in E'{}_{v_{j2}}} f(k,e_i)\right)\le 2 \tag{15}$$

$$\sum_{k=1}^{K}\left(\sum_{e_i\in E'{}_{v_{j1}}} f(k,e_i)+\sum_{e_i\in E^{\eta}{}_{v_{j2}}} f(k,e_i)\right)\le 2 \tag{16}$$

(*Transformation*3) Constraint (10) is equivalent to new constraint (17).

$$\sum_{k=1}^{K}\left(\sum_{e_i\in E^{\eta}{}_{v_{j1}}} f(k,e_i)+\sum_{e_i\in E^{\eta}{}_{v_{j2}}} f(k,e_i)\right)\le 2 \tag{17}$$

The proof of *Transformation* 2 and 3 is similar to *Transformation* 1.

New spacing constraints (14)-(17), keep the linear nature of the basic model, and make the spacing constraints more concise. To describe the spacing rule requirement between two vertices, old method need $K(K-1)/2$ constraints, and new method only need 1 constraint.

5.2 Reduce the Scale of MCF Model

The complexity of the ILP problem depends heavily on the total count of variables, constraints and non-zeros in the model. Reducing the scale of MCF model can make the problem easier to solve.

First, capacity constraint for brother edges e_i and $\bar{e_i}$ could be removed from MCF model if $u(e_i)=1$ and $u(\bar{e_i})=1$. Given two adjacent vertices v_{j1} and v_{j2}, let e_1 be edge (v_{j1},v_{j2}) and e_2 be edge (v_{j2},v_{j1}). Both $u(e_1)$ and $u(e_2)$ are 1. Without losing its generality, assuming that there is unit flow of commodity $n_{k^\star}$ streaming from v_{j1} to v_{j2} by edge e_1. Connectivity constraint (3) on vertex v_{j2} ensures that the commodity will stream out from one edge in $E_{v_{j2},out}$. Connectivity constraint (5) on vertex v_{j2} ensures that other commodity can not go through vertex v_{j2}. So if we don't add capacity constraint for edge e_1 and e_2, there won't be any shorts, and the only remaining problem we need to pay attention to is whether commodity $n_{k^\star}$ makes a circle, i.e. this commodity streams out from v_{j2} by edge e_2. Absolutely, this problem won't occur. Because the solution has a detour, the objective function (6) will reject it and select a better one.

Second, we restrict the routing layers that a net can use. For a two-component net, assuming that two components locates at layer z_1 and layer z_2 ($z_1\le z_2$) respectively. The final routing resource this net can use is from layer z_{min}(=

max$\{z_1 - \Delta, 0\}$) to layer $z_{max}(= \min\{z_2 + \Delta, L\})$. This restriction makes a tradeoff between routing quality and runtime. Larger Δ means better quality and worse runtime while smaller Δ means worse quality and better runtime. In our experiment, if $z_{max} \leq 2$, Δ is 2; otherwise Δ is 1.

5.3 Constraints Relaxation

In a highly congested design, detailed router will search and repair DRC violations using different sizes and locations of routing windows by many iterations. It must create a unbroken path for each net even if there are some DRCs. This indicates that some solutions outside MCF model solution space arising from violating capacity and spacing rule constraints are also acceptable. In our experiments, one penalty variable which must be positive integer is added to every capacity and spacing constraints, and all penalty variables multiplied with large penalty factor are added to the objective function. Taking one vertex capacity constraint for example, inequation (5) is substituted by $\sum_{k=1}^{K} \sum_{e_i \in E_{v_j}} f(k, e_i) - o(v) \leq 2$, and $o(v) * p(v)$ is added to objective function (6). Here, $p(v)$ is a large penalty factor.

5.4 Multi-Thread Strategy

Multi-thread is an common and effective speedup strategy. In detailed routing stage, the chip is generally divided into many subregion. Utilizing the independency of subregion, an multi-thread strategy is used in the following experiments. The multi-thread strategy is on the basis of *map-reduce* model. In order to decrease the boundary influence on routing quality, a method is applied to guarantee that two adjacent subregion won't be solved at the same time. Comparing between the CPU time and real time, the multi-thread strategy we implemented achieves a close to linear speedup.

6. EXPERIMENTAL RESULT

The MCF-based detailed router is implemented with C++ language on linux server with 2.4GHz Intel Xeon CPU and 24G memory. To show the effectiveness of the proposed algorithm, we compared it with Cadence router, Encounter v10.10 on 45nm technology benchmarks.

Our experiments are executed on two series benchmarks. The first one is 5 industrial benchmarks and the second one is 6 academic benchmarks derived from ISPD05 [18] and ISPD14 [19]. ISPD05 benchmarks are transformed from bookshelf format to LEF/DEF format by a publicly available conversion tool[2]. All benchmarks are mapped to a library with 45nm design rules. Using the APIs from OpenAccess (OA) system, we convert the LEF/DEF inputs to an OA database. Both Encounter and our MCFRoute work on and exchange data through the OA database.

A. Results of Model Optimization

In section 5.2, two optimization methods are proposed to optimization our MCF model. We take some experiments to make a comparison between original model and optimized model on total count of variables, constraints and non-zeros within several regions of different size. In table 2, the first column lists the region size by vertical track and horizontal track. The second column is the net count in each region. The following six columns show number ($\times 10^5$) of variable, constraint and non-zero in original model and optimize model respectively. Experiments show that three metrics are reduced by 31%, 30% and 33% respectively that can reduce the difficulty of ILP a lot.

Table 2: Results of model optimization

size	#net	#variable		#constraint		#non-zeros	
		orig	opt	orig	opt	orig	opt
10×10	29	1.14	0.72	0.25	0.16	0.86	0.55
20×10	35	2.78	1.86	0.58	0.39	2.09	1.41
20×20	52	8.3	5.42	1.66	1.1	6.54	4.22
40×20	90	28.7	20.3	5.51	3.96	22.5	15.5
40×40	135	86.4	65.6	16.21	12.4	69.4	48.4

B. Results of Industrial Benchmarks

Table 3 shows the effectiveness of the proposed MCFRoute in industrial benchmarks. The first four columns are the information of benchmarks. '#net' is the total net count. 'size' denotes the G_cell grid size. Both the height and width of G_cell is equal to the height of standard cell. It gives the scale of every benchmarks. '#layer' is the total number of routing layer. The following eight columns show the wirelength ('wirelen.'), via count ('#via' $\times 10^3$), violation count ('#vio.') and runtime ('runtime') of routing solution generated by MCFRoute and Encounter respectively. The wirelength is measured in *millimeter*, and runtime is measured in *second* with 8 threads.

For these complex industrial benchmarks, the proposed algorithm can achieve a desirable results. Compared with Encounter, MCFRoute can reduce the DRCs by 38%. At the same time, the wirelength and via count are also reduce by 3% and 2.7% respectively. The effective multi-thread strategy make the runtime only 1.73× of Encounter.

C. Results of ISPD Contest Benchmarks

First we compare Encounter and MCFRoute on four ISPD05 benchmarks. The routing results comparison is shown in table 4. Different from table 3, runtime is measured in $\times 10^3$ *second* with 8 threads. Both Encounter and MCFRoute can get DRC-clean solution for these 4 benchmarks after several iterations of DRC fixing. Experimental results show that not only can MCFRoute get a routing result with DRC-clean but also better wirelength and via count by 2% and 5% respectively. However, we also observe that the runtime is a great disadvantage of MCFRoute when dealing with larger scale benchmarks.

We have mentioned that traditional sequential methods generally result in low quality routing results because of their non-optimal net-order nature. However, they are faster than concurrent method. To take both advantage of sequential method in runtime and advantage of concurrent method in quality, the proposed algorithm is implemented as an incremental detailed router. A detailed router of sequential method route the nets first, then MCFRoute execute incrementally to refine the solution with DRCs. When working as an incremental router, MCFRoute selects the routing region automatically and ripups part of nets close to violation in the region, then routes these nets simultaneously. If the DRC hasn't been fixed, MCFRoute will expand the routing

[2]PlaceUtil: developed by University of Michigan

Table 3: Comparison Results on Industrial Benchmarks

Benchmark information				MCFRoute				Encounter			
name	#net	size	#layer	wirelen	#via	#vio	runtime	wirelen	#via	#vio	runtime
chip1	36760	297×309	6	664	390	282	2140	716	406	267	1680
chip2	52612	352×367	6	833	419	281	2191	861	436	533	1320
chip3	100479	463×484	6	1706	824	503	4857	1750	843	922	2520
chip4	90905	446×466	6	1564	744	510	4426	1599	760	795	2280
chip5	100473	462×483	6	1742	826	537	4822	1785	846	910	2880
avg	-	-	-	1	1	1	1.73	1.03	1.03	1.62	1

Table 4: Comparison Results on ISPD05 Benchmarks

Benchmark information				MCFRoute			Encounter		
name	#net	size	#layer	wirelen.	#via	runtime	wirelen.	#via	runtime
bigblue1	280281	964×965	5	163	2920	42	167	3117	1.97
bigblue2	560340	1558×1566	5	241	5023	50	246	5179	2.8
adaptec1	215621	964×965	5	125	2321	31	128	2420	1.51
adaptec2	255730	1268×1268	5	151	2655	36	154	2829	1.85
avg.	-	-	-	1	1	19.5	1.022	1.048	1

region. The two benchmarks in table 5 are chosen from ISPD14 which is more difficult that Encounter doesn't obtain DRC-clean solution. Column 2-4 list number of different type DRC generated by Encounter, and column 5-7 list the result after the refinement of MCFRoute. Experimental results shows that the number of DRCs is reduced by half after the refinement of MCFRoute. It is obviously that MCFRoute are capable of handling some difficult detailed routing problem which can not be solved by sequential strategy.

Table 5: Results of Incremental Detailed Router

Benchmark	Encounter			MCFRoute		
	spce	shrt	oth.	spac	shrt	oth.
matrix_mult_wt	9	58	0	5	34	0
matrix_mult_nt	10	93	1	2	41	0

7. CONCLUSION

Based on MCF method, we present a novel concurrent detailed routing algorithm which takes complex design rules into account in this paper. Since all nets are routed simultaneously by the proposed router, net-ordering problem which directly affects the routing quality in traditional ripup-and-reroute methods no longer arises. More reasonable resource assignment improves the routing quality. And several useful strategy are introduced to optimize MCF model. Experimental results show the effectiveness and potential of the proposed algorithm. The future work will further improve the ability of MCFRoute by model optimization and reduce the runtime by distributed method. We also plan to incorporate more objective into the proposed model such as timing, power and design for manufacture.

8. REFERENCES

[1] M. Burstein and R. Pelavin. Hierarchical wire routing. *IEEE Trans. on CAD*, 2(4):223–234, 1983.

[2] S Batterywala et al. Track assignment: a desirable intermediate step between global routing and detailed routing. In *Proc. of ICCAD*, pages 59–66, 2002.

[3] C. J. Alpert et al. What makes a design difficult to route. In *Proc. of ISPD*, pages 7–12, 2010.

[4] D. Abercrombie et al. Restrictive design rules and their impact on 22 nm design and physical verification. In *Proc. of EDPS*, 2009.

[5] C. Y. Lee. An algorithm for path connections and its applications. *IRE Trans. on Electronic Computers*, 3:346–365, 1961.

[6] K. Mikami and K. Tabuchi. A computer program for optimal routing of printed circuit conductors. *IFIP Congress*, 2:1475–1478, 1968.

[7] P. E. Hard, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on SSC*, 4(2):100–107, 1968.

[8] S. Q. Zheng, J. S. Lim, and S. S. Iyengar. Finding obstacle avoiding shortest paths using implicit connection graphs. *IEEE Trans. on CAD*, 15(1):103–110, 1996.

[9] J. Cong, J. Fang, and K. Y. Khoo. An implicit connection graph maze routing algorithm for eco routing. In *Proc. of ICCAD*, pages 163–167, 2005.

[10] Y. Zhang and C. Chu. Regularroute: An efficient detailed router with regular routing patterns. In *Proc. of ISPD*, pages 146–151, 2011.

[11] Y. Zhang and C. Chu. Gdrouter: Interleaved global routing and detailed routing for ultimate routability. In *Proc. of DAC*, pages 597–602, 2012.

[12] Y. Xu, Y. Zhang, and C. Chu. Fastroute 4.0: Global router with efficient via minimization. In *Proc. of ASPDAC*, pages 597–602, 2009.

[13] Michael Gester et al. Algorithm and data structures for fast and good vlsi routing. In *Proc. of DAC*, pages 459–464, 2012.

[14] E. Shragowitz and S. Keel. A global router based on a multicommodity flow model. *the VLSI journal*, 5(1):3–16, 1987.

[15] M. Ozdal. Detailed-routing algorithms for dense pin clusters in integrated circuits. *IEEE Trans. on CAD*, 28(3):340–349, 2009.

[16] Lef/def language reference. ftp://ftp.sitsemi.ru/pub/cadence/lefdefref.pdf.

[17] Gurobi optimizer. http://www.gurobi.com/download/gurobi-optimizer, 2013.

[18] Ispd 2005 placement contest. http://archive.sigda.org/ispd2005/contest.htm, 2005.

[19] Vladimir Yutsis et al. Ispd 2014 benchmarks with sub-45nm technology rules for detailed-routing-driven placement. In *Proc. of ISPD*, 2014.