

Exact Multi-Layer Topological Planar Routing

Olivier Coudert

Synopsys, Inc., 700 East Middlefield Rd.
Mountain View, CA 94043

C.-J. Richard Shi

University of Iowa, ECE Department
Iowa City, Iowa 52242

Abstract—This paper describes an exact algorithm for multi-layer topological planar routing in switchboxes and channels. Using recent developments in set covering resolution, this method produces in a few minutes the optimum solutions for routing problems that were previously solved by heuristics.

I INTRODUCTION

A *routing region* (Fig. 1) is an area enclosed by an external boundary, with some blocks inside the boundary (called holes), and pins on the external and internal boundaries. A routing area is a *switchbox* when there is no hole, and a *channel* when pins are located on the top and bottom of the boundaries. A *net* is a set of pins to be connected without going through any boundary. A *planar subset* is a set of nets that can be routed in one layer without crossing each other. The maximum k -layer problem consists of routing as many nets as possible with k planar subsets. The minimum α -cover problem consists of routing at least a fraction α of the nets with as few planar subsets as possible.

These problems are motivated by MCM, PCB and performance-driven IC routing. For submicron VLSI designs, wire delays dominate gate delays, and the routing area exceeds the area used by transistors. Thus multi-layer routing technology is of increasing importance in such a high performance design. Polynomial algorithms have been developed for special cases, but the general problem is NP-complete [2].

The approach we propose to multi-layer topo-

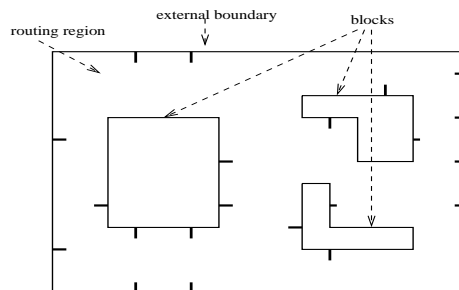


Figure 1: A routing region.

logical routing is made of two steps:

- (a) compute all the maximal planar subsets.
- (b) solve a set covering problem.

Not only is the set covering problem NP-complete, but also the number of maximal planar subsets can be exponential w.r.t the number of nets. However we show that these two steps can be solved *exactly* with a low computational cost in practice.

In this paper, step (a) is addressed in Section II for switchboxes or channels only. Section III shows how step (b) can be solved using a new lattice based paradigm. Section IV explains how this paradigm can be efficiently implemented with ZBDDs (Zero suppressed Binary Decision Diagrams). Experimental results are discussed in Section V.

II MAXIMAL PLANAR SUBSETS FOR SWITCHBOXES AND CHANNELS

Two nets are *compatible* iff they can be routed in a single layer without crossing. In the case of

switchboxes or channels, sets of mutually compatible nets (called clusters) are exactly the planar subsets. Consider Fig. 2, where pins with the same number form a net. Nets 1, 2 and 3 form the cluster $\{1, 2, 3\}$, and these three nets can be routed in a plane without crossing each other. The maximal clusters are $\{1, 2, 3, 4\}$, $\{1, 2, 5\}$ and $\{3, 4, 6\}$. The minimum layer routing is $\{\{1, 2, 5\}, \{3, 4, 6\}\}$.

If we see the set of nets V as a set of vertices, and the set of compatible pairs E as a set of edges, then a cluster is a clique of the undirected graph (V, E) . Thus step (a) amounts to computing all the maximal cliques of this graph.

Given the set U of *uncompatible* pairs, a cluster is a set of nets that does not contain any elements of U . Thus we can express the set of all clusters *Cluster* by:

$$\begin{aligned} \text{Cluster} &= \text{NotSupSet}(2^V, U), \quad \text{where} \\ \text{NotSupSet}(X, Y) &= \{x \in X \mid \forall y \in Y, y \not\subseteq x\}. \end{aligned}$$

The set of all maximal planar subsets *MaxPlanSet* is:

$$\begin{aligned} \text{MaxPlanSet} &= \max_{\subseteq} \text{Cluster}, \quad \text{where} \\ \max_{\subseteq} X &= \{x \in X \mid \forall x' \in X, x \subseteq x' \Rightarrow x = x'\}. \end{aligned}$$

III SOLVING SET COVERING

Let X and Y be two sets, and R a binary relation on $X \times Y$. An element y of Y *covers* an element x of X iff $x R y$. The *set covering problem* $\langle X, Y, R \rangle$ consists of finding a minimum subset S of Y such that any x of X is covered by some y of S . Step (b) amounts to solving:

$$\langle V, \text{MaxPlanSet}, \in \rangle. \quad (1)$$

The classical approach consists of building the corresponding covering matrix, and of solving it using the Quine–McCluskey procedure [1]. The size of the covering matrix is in $O(|V| \times |\text{MaxPlanSet}|)$, which limits this approach since $|\text{MaxPlanSet}|$ can be exponential (> 100000 in some of our examples) w.r.t. $|V|$.

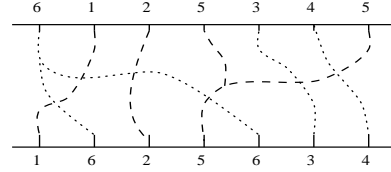


Figure 2: A topological routing channel.

To overcome this limitation, we use the results developed in [3]. The method consists of transforming the original set covering problem (1) into an isomorphic set covering problem (2) expressed on a complete lattice. Here, an obvious translation is:

$$\langle X, \text{MaxPlanSet}, \subseteq \rangle, \quad (2)$$

where $X = \{\{v\} \mid v \in V\}$.

X and *MaxPlanSet* are subsets of the lattice $(2^V, \subseteq)$. Applying the fundamental result of [3, Theorem 8], the following reduction rules:

$$\begin{aligned} \langle X, Y, \subseteq \rangle &\rightarrow \langle X, \max_{\subseteq} \tau_X(Y), \subseteq \rangle, \\ \langle X, Y, \subseteq \rangle &\rightarrow \langle \max_{\subseteq} \tau_Y(X), Y, \subseteq \rangle, \quad \text{and} \\ \langle X, Y, \subseteq \rangle &\rightarrow \langle X - E, Y - E, \subseteq \rangle \quad \text{with } E = X \cap Y, \end{aligned}$$

where τ_X and τ_Y are defined from 2^V into 2^V by:

$$\begin{aligned} \tau_X(y) &= \bigcup_{\substack{x \in X \\ x \subseteq y}} x \\ \tau_Y(x) &= \bigcap_{\substack{y \in Y \\ y \supseteq x}} y \end{aligned}$$

are applied on (2) to yield a new set covering problem, say (3), which is isomorphic to the cyclic core of (1). From the solutions of (3), one can recover the solutions of (1) using basic set operations [3]. The interest of such an approach is that the reduction to the isomorphic cyclic core (3) can be done with ZBDDs in an *implicit* way, i.e., with a complexity that does not depend on the number of elements of X and Y .

IV ALGORITHMS

ZBDD (Zero suppressed Binary Decision Diagram) is a graph data structure to represent subsets of 2^V in a compact and canonical way. We have shown that steps (a) and (b) can be solved by using some complex set operations defined on 2^V . Indeed, these set operations can be efficiently implemented with ZBDDs, so that their costs are not related to the size of the sets they operate on, but to the size of the ZBDDs representing these sets. In practice, very large sets can be represented with small ZBDDs (e.g., the ZBDD of 2^V has size $O(|V|)$), which enables us to solve effectively the multi-layer problems.

Given $V = \{v_1, \dots, v_n\}$, let a *combination* be a subset of V , and X be a set of combinations, i.e., $X \subseteq 2^V$. The set X can be decomposed w.r.t to an element v_k in two unique sets X_{ε_k} and X_{v_k} , where: X_{ε_k} is the set of combinations belonging to X that do not contain v_k ; X_{v_k} is made of the combinations belonging to X that contain v_k , from which v_k has been removed. For instance, with $X = \{\{\}, \{v_1, v_2\}, \{v_1, v_3, v_5\}, \{v_4\}\}$ and $k = 1$, we have $X_{\varepsilon_1} = \{\{\}, \{v_4\}\}$ and $X_{v_1} = \{\{v_2\}, \{v_3, v_5\}\}$.

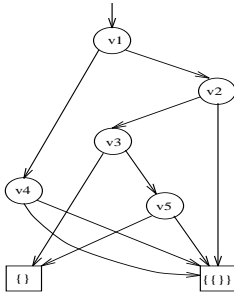


Figure 4: A ZBDD.

If X is recursively decomposed w.r.t. the elements of V by applying the decomposition

$$X = \text{vertex}(v_k, X_{\varepsilon_k}, X_{v_k}),$$

one obtains a binary tree whose leaves are $\{\}$ and $\{\{\}\}$. When representing sets of *sparse* combinations, most of the right branches (which denote the subsets of combination where v_k 's do

not occur) points to $\{\}$. This suggests the following compression scheme [4]: (1) isomorphic subgraphs are shared in memory; (2) a vertex whose right branch points to $\{\}$ is removed and replaced with the vertex pointed by its left branch. The resulting graph is the ZBDD of X , and is canonical w.r.t. to the ordering of the elements of V . Fig. 4 shows the ZBDD of the set X given above.

Fig. 3 describes the complex set operations needed to solve steps (a) and (b) in terms of basic set operations (union, intersection, *vertex*, etc). $\text{MaxSet}(X)$ evaluates $\max_{\subseteq} X$, $\text{MaxTauRow}(X, Y)$ evaluates $\max_{\subseteq} \tau_Y(X)$, and $\text{MaxTauCol}(Y, X)$ evaluates $\max_{\subseteq} \tau_X(Y)$, with both X and Y subsets of the lattice $(2^V, \subseteq)$. Function NotSubSet , defined as $\text{NotSubSet}(X, Y) = \{x \in X \mid \forall y \in Y, x \not\subseteq y\}$, can be implemented in a way similar to NotSupSet . All these operations use a divide-and-conquer strategy on the decompositions of the arguments' ZBDDs w.r.t. their top labels v_k .

V EXPERIMENTAL RESULTS AND CONCLUSION

Table 1 presents some experimental results on multi-layer topological routing problems. The CPU time includes reading the incompatible pairs, building the ZBDD of the maximal planar subsets, and solving the covering problem.

Computing the ZBDD of the maximal planar subsets does not take more than 15 seconds. Note that the size of the ZBDD (i.e., its number of vertices) remains small, even when the number of maximal planar subsets is large.

The cyclic core is implicitly computed using the ZBDD based method presented in Section III and IV. The set covering problem is reduced by a factor > 20 . The cyclic core is then explicitly solved using the minimizer SCHERZO [3].

The difference between the greedy solution [2] and the optimum solution is not significant. However, the interest of this exact approach is as follows:

- Our formulation only uses two aspects of the problem: (1) whether two nets can be routed

```

function NotSupSet( $X, Y$ );
if  $X = \{\}$  or  $\{\} \in Y$  or  $X = Y$  return  $\{\}$ ;
if  $X = \{\{\}\}$  or  $Y = \{\}$  return  $X$ ;
let  $T1 = \text{NotSupSet}(X_{v_k}, Y_{\varepsilon_k}) \cap \text{NotSupSet}(X_{v_k}, Y_{v_k})$ 
     $T0 = \text{NotSupSet}(X_{\varepsilon_k}, Y_{\varepsilon_k})$  in
    return vertex( $v_k, T0, T1$ );

```

```

function MaxTauRow( $X, Y$ );
if  $Y = \{\}$  or  $X = \{\}$  return  $\{\}$ ;
if  $Y = \{\{\}\}$ 
    if  $\{\} \in X$  return  $\{\{\}\}$  else return  $\{\}$ ;
if  $X = \{\{\}\}$  and  $\{\} \in Y$  return  $\{\{\}\}$ ;
let  $X1 = \text{NotSubSet}(X_{\varepsilon_k}, Y_{\varepsilon_k})$ 
     $T1 = \text{MaxTauRow}(X1 \cup X_{v_k}, Y_{v_k})$ 
     $T0 = \text{MaxTauRow}(X_{\varepsilon_k} - X1, Y_{\varepsilon_k} \cup Y_{v_k})$  in
    return vertex( $v_k, \text{NotSubSet}(T0, T1), T1$ );

```

```

function MaxSet( $X$ );
if  $X = \{\}$  or  $X = \{\{\}\}$  return  $X$ ;
let  $T1 = \text{MaxSet}(X_{v_k})$ 
     $T0 = \text{MaxSet}(X_{\varepsilon_k})$  in
    return vertex( $v_k, \text{NotSubSet}(T0, T1), T1$ );

```

```

function MaxTauCol( $Y, X$ );
if  $Y = \{\}$  or  $X = \{\}$  return  $\{\}$ ;
if  $Y = \{\{\}\}$ 
    if  $\{\} \in X$  return  $\{\{\}\}$  else return  $\{\}$ ;
if  $X = \{\{\}\}$  return  $\{\{\}\}$ ;
let  $Y0 = \text{NotSupSet}(Y_{v_k}, X_{v_k})$ 
     $T0 = \text{MaxTauCol}(Y0 \cup Y_{\varepsilon_k}, X_{\varepsilon_k})$ 
     $T1 = \text{MaxTauCol}(Y_{v_k} - Y0, X_{\varepsilon_k} \cup X_{v_k})$  in
    return vertex( $v_k, \text{NotSubSet}(T0, T1), T1$ );

```

Figure 3: Set operations needed to solve steps (a) and (b) implicitly.

in the same layer or not, which is a local information; (2) for switchboxes and channels, a planar subset is a set of mutually compatible nets. Since our method does not exploit the special structure of the problem, it can be extended to handle the general case.

- The method can handle practical design constraints, like the physical capacity constraint, i.e., the maximal number of nets that are allowed to be routed in one layer.

REFERENCES

- [1] R.K. Brayton, G.D. Hachtel, C.T. McMullen, A.L. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*, Kluwer Academic Pub., Dordrecht, 1984.
- [2] J. Cong, M. Hossain, N. A. Sherwani, “A Provably Good Multi-Layer Topological Planar Routing Algorithm in IC Layout Designs”, *IEEE Trans. CAD*, **12**-1, pp. 70–78, Jan. 1993.
- [3] O. Coudert, “2-Level Logic Minimization: An Overview”, *Integr.*, **17**-2, pp. 97–140, Oct. 1994.
- [4] S. Minato, “Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems”, *Proc. 30th DAC*, Dallas, TX, pp. 272–277, June 1993.
- [5] C.-J. Shi, J. A. Brzozowski, “A Framework for the Analysis and Design of Algorithms for a Class of VLSI-CAD Optimization Problems”, *Proc. ASP-DAC*, pp. 67–74, Chiba, Japan, Aug. 1995.

example			max planar subset		CC		Sol		
Name	#net	#uncomp	ZBDD	#plan	row	col	greedy	exact	CPU
<i>bus</i>	24	133	62	45	6	8	9	9	0.13
<i>ex1</i>	21	77	71	50	9	14	9	7	0.11
<i>ex3a</i>	44	176	149	495	16	41	11	10	0.44
<i>ex3b</i>	47	283	139	7685	31	804	10	9	1.85
<i>ex3c</i>	54	336	224	3329	32	621	13	12	1.46
<i>ex4b</i>	54	298	524	6885	31	535	13	11	2.05
<i>ex5</i>	64	405	282	61269	47	6749	9	9	135.36
<i>ex5b</i>	64	427	5988	44938	46	4773	11	10	38.05
<i>deut</i>	72	763	196	101427	50	5777	18	16	18.49
<i>exam1</i>	200	17124	734	6711	107	1512	135	126	51.02
<i>exam2</i>	250	26081	1061	19409	146	1897	150	141	159.53
<i>exam3</i>	300	36801	2294	100520	228	13316	176	162	597.53

For each **example**, **#net** is the number of nets, and **#uncomp** the number of incompatible pairs. **|ZBDD|** is the size (number of vertices) of the ZBDD representing the **#plan** maximal planar subsets. The cyclic core **CC** has size **row** \times **col** (note that the size of the explicit initial covering problem is **#net** \times **#plan**). **Sol** gives the **greedy** and **exact** solutions for the minimum 1-cover problem. The **CPU** time to solve exactly the problem is given in seconds on a 60 MHz SuperSparc (85.4 SpecInt).

example	k = 1		k = 2		k = 3		k = 4		k = 5	
<i>bus</i>	10/10	41%/41%	14/14	58%/58%	17/17	70%/70%	19/19	79%/79%	20/20	83%/83%
<i>ex1</i>	8/8	38%/38%	11/12	52%/57%	14/15	66%/71%	16/17	76%/80%	17/19	80%/90%
<i>ex3a</i>	20/21	45%/47%	26/27	59%/61%	30/31	68%/70%	33/34	75%/77%	35/36	79%/81%
<i>ex3b</i>	15/15	31%/31%	25/26	53%/55%	32/33	68%/70%	37/37	78%/78%	39/40	82%/85%
<i>ex3c</i>	20/20	37%/37%	30/31	55%/57%	34/36	62%/66%	38/40	70%/74%	41/43	75%/79%
<i>ex4b</i>	22/22	40%/40%	31/31	57%/57%	37/38	68%/70%	41/43	75%/79%	44/46	81%/85%
<i>ex5</i>	20/20	31%/31%	31/32	48%/50%	38/42	59%/65%	45/49	70%/76%	50/55	78%/85%
<i>ex5b</i>	20/20	31%/31%	31/32	48%/50%	39/41	60%/64%	46/48	71%/75%	51/54	79%/84%
<i>deut</i>	17/17	23%/23%	28/29	38%/40%	36/39	50%/54%	42/46	58%/63%	46/50	63%/69%

The first column under each k describes the number of nets routed by greedy vs optimum algorithm. The second column describes the percentage of nets routed by greedy vs optimum algorithm.

Table 1: Experimental results: minimum 1-cover problem and maximum k -cluster problem.