RISA: Accurate and Efficient Placement Routability Modeling

Chih-liang Eric Cheng

Cadence Design Systems, Inc.
San Jose, CA 95134

Abstract

The prevalence of net list synthesis tools raises great concern on routability of cell placement created with state-of-the-art placement techniques. In this paper, an accurate and efficient placement routability modeling technique is proposed and incorporated into the prevailing simulated annealing approach. This accurate and efficient modeling is based on the supply versus demand analysis of routing resource over an array of regions on a chip. Vertical and horizontal routability is analyzed separately due to the bias of routing resource in multiple-metal-layer ASIC designs. A special technique on net bounding box partitioning is also proposed and critical to the accuracy of this modeling at the presence of mega cells, which tend to cause local routing congestion. By incorporating this efficient modeling into the cost function of simulated annealing, experiments conducted on small to large industrial designs indicate that placement routability evaluated with a global router is greatly improved as a result of the proposed accurate modeling.

1: Introduction

Net list synthesis tools tend to generate net lists with high pin-count nets and poor porosity library cells, which often result in local routing congestion. Traditional placement approaches assume that routing resource demand is more or less even throughout the core area of a chip. By spreading out cells evenly, routing congestion should not exist in chip layout. That assumption is broken as more net list modules are synthesized. In addition, the presence of mega cells in the popular hierarchical design methodology also causes local congestion around their boundaries due to the detouring routing traffic. Existing placement approaches tend to ignore the congestion resulted from this kind of detouring. Designers are forced to enlarge design sizes to get around the routability problem.

State-of-the-art placement approaches such as min-cut [1], PROUD [2], GORDIAN [3], and simulated annealing [4] fail to model routability into their objective functions. Min-cut and crossing count metric in simulated annealing minimize the number of crossing nets without any control over the distribution of crossing along one cut line. Quadratic programming methods minimize wire length, which only indirectly reduces congestion, while spreading out cells. As a result of this failure, severe routing congestion may still reside on a chip even though cells are uniformly placed. Figure 11 & 12 in Section Five shows one of such cases.

A simultaneous min-cut and hierarchical routing approach was proposed in [5]. The effectiveness of min-cut approaches was limited by its one-dimensional projection view. A congestion-driven placement based on a multi-partitioning heuristic was proposed in [6]. A chip was partitioned into an array of rectangles uniformly. Multi-partitioning was applied after the initial one to balance cell area plus wiring area among all regions. The performance was limited by the initial partition and the number of rectangles. Experiment only showed results on small designs. A method that uses initial global routing model for congestion analysis and optimization was proposed in [7]. It is intended for only local improvement based on greedy extensive search within small windows.

A large number of metrics or costs used in simulated annealing can be found in [8]. Wire length, which does not account for net interaction, is widely used as a metric in simulated annealing. [9] The use of crossing count more effectively reduces global congestion and wire length simultaneously as discussed in [10]. However, local congestion introduced by synthesized net list is hardly visible to it. Others tried to resolve routability problem by minimize routing constraints of various kinds [11, 12]. Those were specifically for small row-based designs, not today's huge sea-of-gate designs. A technique was proposed to minimize the maximum wiring density using a two-dimensional array of cells in [13, 14]. Routing resource supply was not taken into account.

Routability problem is a problem of routing resource balancing. A well-balanced cell placement requires smaller die size. In this paper, a placement routability modeling is proposed which can enhance routability with simulated annealing through efficient and accurate balancing of routing resource demand & supply over an array of regions. The analysis result is transformed into a component of the cost function of simulated annealing. Initially, every region contains the same amount of routing resource supply. Existing wiring of power and clocking, and cells reduce routing resource supply of the regions they are located in. Net routing increases routing resource demand. The goal is to make supply meet demand in all regions. A special technique on handling nets overlapping with mega cells is proposed to model congestion around mega cells.

This paper is arranged as follows: Section Two introduces the modeling of pre-wiring, cells, and mega cells in the supply phase. The demand modeling phase is described in Section Three. The technique of net bounding box partitioning is discussed in Section Four. This set of modeling is named RISA, which has applications in various placement algorithms. The excellent improvement on routability is shown in Section Five when RISA is incorporated into simulated annealing on twelve industrial small to large designs. This paper is concluded with a short remark and possible applications of RISA.

2: Routing resource supply modeling

In order to make local routing congestion visible to a metric, the analysis must be performed over an *NxN* array of regions in core area. In each region *(i, j)*, two variables are defined, D_{ij} (demand) & S_{ij} (supply) in each direction (vertical, horizontal), where *i* and *j* are between *1* and *N*. The proposed new metric is

$$R = \sum_{i=1}^{N} \sum_{j=1}^{N} [Max(_v D_{ij} - t \times {_v}S_{ij}, 0)]^2 +$$
$$w \times \sum_{i=1}^{N} \sum_{j=1}^{N} [Max(_h D_{ij} - t \times {_h}S_{ij}, 0)]^2 \qquad (1)$$

, where *t* is a constant less than *1* for aggressive optimization and *w* represents the relative importance of one direction over the other when routing resource is biased. Let T_v and T_h denote the total number of full tracks available in both directions over the core area including all metal layers. Therefore, initially,

$$_v S_{ij} = T_v / N \qquad _h S_{ij} = T_h / N$$
$$_v D_{ij} = {_h}D_{ij} = 0 \qquad (2)$$

for all *i* and *j*. $_v S_{ij}$ and $_h S_{ij}$ are the numbers of short tracks in each region. Throughout the paper, the discussion on routing resource supply and demand will be in term of tracks. Existing wiring of power and clocking nets, regular cells, and mega cells are considered to be obstacle to routing. The routing resource supply decreases when any of these is found in a region. A definition is given below to represent the relationship between a routing metal layer and its preferential routing direction precisely.
*Definition 1: d: m -> {v, h} is a mapping from a metal layer m to its preferential direction, either vertical or horizontal.*

2.1: Existing wiring modeling

An existing wire in layer m of power or clocking nets found in a region *(i, j)* as shown in Figure 1.a decreases routing resource supply by

$$\Delta_{d(m)} S_{ij} = w \times l / L \qquad (3)$$

, where w is in the unit of routing tracks. Since these wires are fixed throughout placement, no updating is required during run time of annealing process once the resource supply is reduced.

Fig. 1.a: Existing wire | Fig. 1.b: Regular cell
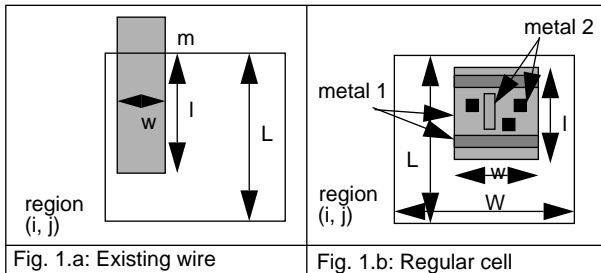
2.2: Regular cell modeling

A regular cell normally consists of a blockage in the first metal layer, which is as large as the cell outline, a few blockages in the second metal layer and pins in the first or the second metal layer. It is implicitly assumed that *d(1)* and *d(2)* are perpendicular in a normal design. Hence, a cell placed in a region *(i, j)* decreases routing

resource supply by

$$\Delta_{d(1)} S_{ij} = T_1 \times \frac{l \times w}{L \times W}$$
$$\Delta_{d(2)} S_{ij} = (1 - P) \times T_2 \times \frac{l \times w}{L \times W} \qquad (4)$$

where *l* and *w* are the length and width of the cell as indicated in Fig. 1.b, and *L* and *W* are those of the region. T_1 and T_2 are numbers of tracks in the first and the second metal layers within a region, respectively. Let *s* be the width of the cell in the unit of vertical tracks, and *c* be the number of vertical tracks occupied by pins and the second metal blockages when projected onto one line parallel to the second metal direction and passing through the center of the cell. Thus, the porosity of a cell, *P* in (4), is defined as

$$P = 1 - c / s \qquad (5)$$

Note that even though pins might be in the first metal layer, pins are still considered to be obstacles in the second metal layer to other nets because of accessibility. Taking porosity into account greatly enhances the accuracy of RISA on synthesized net list designs. Both decremental values in equations in (4-5) can be preprocessed before annealing starts. Since the size of a region is much larger than that of a regular cell, a cell located on multiple regions is assigned to the bottom-left region it covers. A typical region contains about 100 ~ 200 cells such that this error is negligible at region boundaries.

2.3: Mega cell modeling

Mega cells mentioned in this paper is defined as follows:
*Definition 2: A **mega cell** is a cell much larger than any regular cell and has a subset of metal layers fully blocked (zero porosity) at current design level.*

In triple or more metal-layer designs, normally mega cells are blocked in the first two layers, and use the third and upper metal layers for feedthrough connection. In two-metal-layer designs, all feedthrough connections must detour around mega cells. A two-layer mega cell placed over several regions as depicted in Figure 2 decreases their routing resource supply by

$$\Delta_{d(1)} S_{ij} = T_1 \times \frac{l \times w}{L \times W}$$
$$\Delta_{d(2)} S_{ij} = T_2 \times \frac{l \times w}{L \times W} \qquad (6)$$

Fig. 2: Mega cell
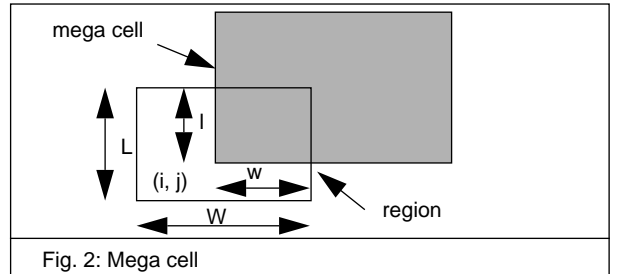
Note that the decrement in routing resource supply of a region depends on where a mega cell is placed. Unless a mega cell is pre-placed and fixed, annealing process has to evaluate equations in (6) during run time. Fortunately, most designs have all mega cells pre-placed and fixed manually because they are normally larger than any placement approach can handle very well. The computation can be performed only once before annealing process as does

a regular cell. Equations in (6) need to be generalized when a mega cell is not two-metal-layer.

At any instant of time during the simulated annealing process, all regular cells and mega cells have their exact locations. Based on equations in (2~6), $_{v,h}S_{ij}$ of each region can be easily obtained without expensive calculation, since most of the expensive divisions and multiplications are performed during pre-processing. Compared with the calculations required for the routing resource demand $_{v,h}D_{ij}$, those for the supply $_{v,h}S_{ij}$ is quite negligible.

3: Routing resource demand modeling

The routing resource demand model is based on net bounding box, which is the smallest rectangle embraces all pins of a net. The wire crossing at a specific cut line through a bounding box depends on not only pin count of the net but also the location of the cut line. Fig. 3 shows that crossing wire count at a given cut is a function of both cut location and pin count.

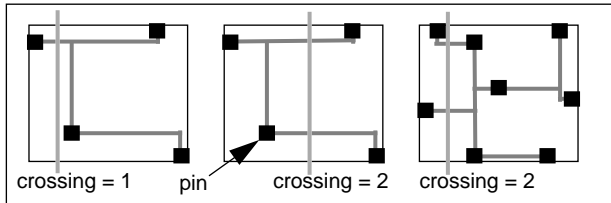crossing = 1   pin   crossing = 2   crossing = 2

Fig. 3: Wire crossing is a function of cut location and pin count

Optimal Steiner tree is not used for estimating routing demand because routers might not route nets in similar patterns and it is more expensive to calculate than bounding box. The probability of having a wire at location *(x, y)* within a net bounding box can be approximated by adding up and normalize K optimal Steiner trees of K sets of randomly located M pins. A wiring distribution map (WDM) so obtained (K = 10000, and M = 20) is depicted in Figure 4, in which only horizontal edges of these tree are considered. The high wiring probability at the top and bottom boundaries comes from the following two facts: 1) the probability of having two pins located at the same boundary is high because of bounding box. 2) when finding an optimal Steiner tree, either a left-L or a right-L is used to reduce the wire length of a minimum spanning tree.

pin count M	net weight q	pin count M	net weight q
1 ~ 3	1.0000	15	1.6899
4	1.0828	20	1.8924
5	1.1536	25	2.0743
6	1.2206	30	2.2334
7	1.2823	35	2.3895
8	1.3385	40	2.5356
9	1.3991	45	2.6625
10	1.4493	50	2.7933

Table 1: Net weighting from WDMs

From Fig. 4 the horizontal routing resource demand within a net bounding box is lower near the left and the right boundaries than the center. Similar phenomenon happens in a vertical WDM. Using WDM in simulated annealing is impractical due to intensive computation or table look-up if a WDM is approximated by a set of equations. A simple net bounding box multiplied by a pin-count-dependent net weight is a good approximation for the sake of computation speed though it over-estimates routing resource demand near those boundaries. The net weights can be calculated

through finding the mean values of WDMs of different pin counts. A set of net weighting so obtained is given in Table 1.
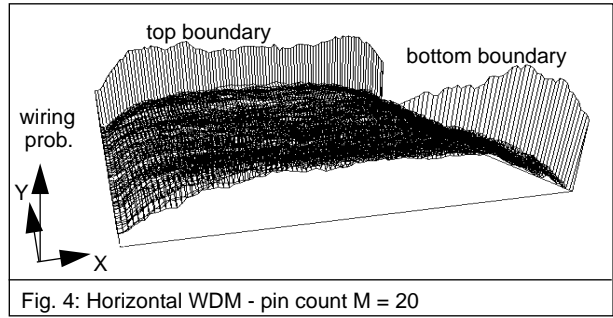

Fig. 4: Horizontal WDM - pin count M = 20

This set of net weighting reflects more realistic wire crossing and length (multiplying semi-perimeter by its net weighting) when a net of high pin count is routed. Net weight *q* represents the expected number of wires crossing a cut line through the bounding box, no matter the cut line is vertical or horizontal.

Routing resource demand increases in a region *(i, j)* if the outline of this region overlaps with a net bounding box as depicted in Figure 5. In addition to all the symbols defined in Fig. 5, let *q* denote the weighting of this net from Table 1, and note that *q* is in unit of tracks. The demand estimation is based on the probability of having a wire within a covered region. The increments in routing resource demand of this region are

$$\Delta_h D_{ij} = q \times \frac{w \times l}{Y \times L}$$

$$\Delta_v D_{ij} = q \times \frac{w \times l}{X \times W} \qquad (7)$$

In some special cases, equations in (7) can be simplified. Some computational techniques can be applied to reduce unnecessary computations for one net to speed up data updating in annealing.
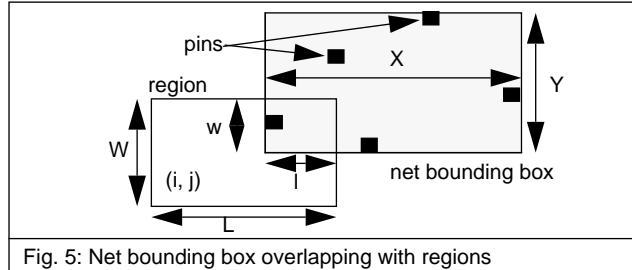

Fig. 5: Net bounding box overlapping with regions

4: Net bounding box partitioning

Routing resource demand modeling is inaccurate when a net bounding box overlaps with mega cell outlines which have usually no less than two metal layers fully blocked. For instance, in three-metal-layer designs, assuming the preferential directions indicated in Figure 6, and the mega cell is blocked in two layers, area A obviously does not allow the second metal wire at all. The net bounding box approach discussed in the previous section has to be revised toward this fact. A net bounding box partitioning technique is presented in the following to solve this problem. The approach is based on partitioning a net bounding box (NBB) into a set of sub-bounding boxes (SBBs) efficiently.

Consider the case depicted in Figure 7, where one NBB of four

pins overlaps with two mega cells. The following steps are taken to insert pseudo pins and determine which SBB of the NBB should be ignored or re-sized.
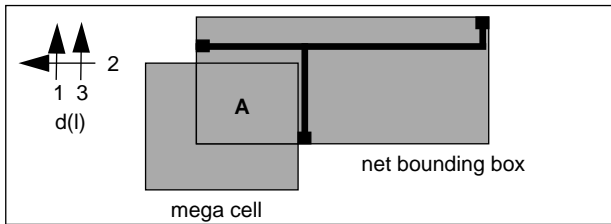

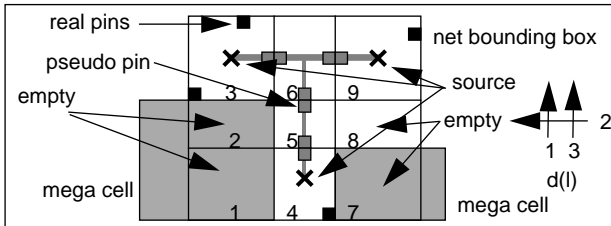Fig. 6: Net bounding box overlapping with a mega cell

Fig. 7: Net bounding box partitioning - example 1

The NBB Partitioning Algorithm:

<1> *Extend each boundary of each mega cell such that it cuts through the NBB*

<2> *For each SBB formed in <1>, if it is on top of a mega cell, mark it BLOCK in layer 1 & 2*

<3> *For each SBB with at least one pin inside, mark it SOURCE.*

<4> *Treat each SBB as a super routing grid and perform three-layer MAZE routing to find a Steiner tree that connects all SOURCE SBBs.*

<5> *If such a Steiner tree does not exist, expand NBB by 2X and go to <1>. If NBB is as large as design size, restore original NBB and ignore mega cells. (net is not routable)*

<6> *For each edge of the Steiner tree, if it passes through any boundary of any SBB, place a pseudo pin at the center of the boundary for each SBB.*

<7> *For each SBB that does not contain either real pins or pseudo pins, mark it EMPTY.*

<8> *For each SBB that is not EMPTY and totally blocked in one direction, then move the pseudo pin, if any, along the boundary, to the center of mass of all real pins inside. Repeat until not applicable.*

<9> *For each SBB that is not EMPTY, resize the SBB such that it embraces both real pins and pseudo pins.*

<10> *Compute routing resource demand of all re-sized SBBs based on equations in (7).*

In Figure 7, the light grey pins are inserted pseudo pins, and the light grey lines represent the Steiner tree the NBB partitioning algorithm finds through three-layer MAZE routing. As a result, SBB 3 contains three pins (two real, and one pseudo), and SBB 6 has three pseudo pins. SBB 1, 2, 7, and 8 are ignored because they are empty. Note that SBB 5, after re-sizing, becomes the shape of a vertical track, which means no horizontal routing resource demand. The result of this partitioning is very close to that of any routing approach that minimizes wire length. Therefore, it is much more accurate than simple net bounding box as in Section 3. Figure 8 shows the final set of SBBs corresponding to those in Figure

7. Any reasonable Steiner tree is very likely to be within the union of the final set of SBBs.
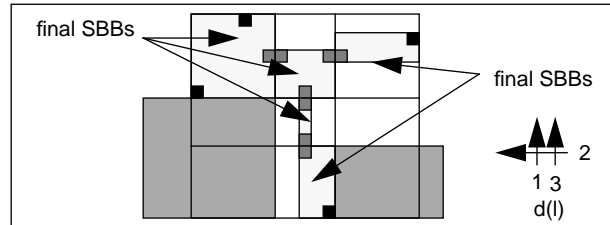

Fig. 8: Final set of SBBs - example 1

The complexity of NBB partitioning is dominated by that of three-layer MAZE routing over a very coarse grid (SBBs) system. Based on the facts that nets related to mega cells are a small portion and pins on mega cells are normally facing the center of core area, the run time of NBB partitioning is not of great concern. In Figure 9, another example of this NBB partitioning is shown. In this case, the NBB has to be expanded in order for the routing to detour around the mega cell. The final set of SBBs, which reflect the fact that the second metal routing is forbidden on mega cell, are shown in Figure 10. Note specially that pins of a mega cell do not have to be at a boundary.
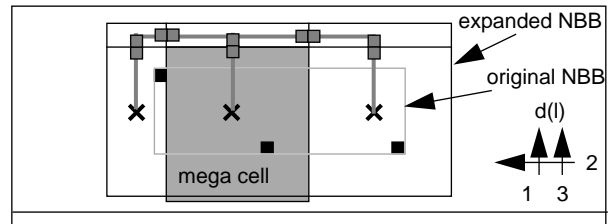

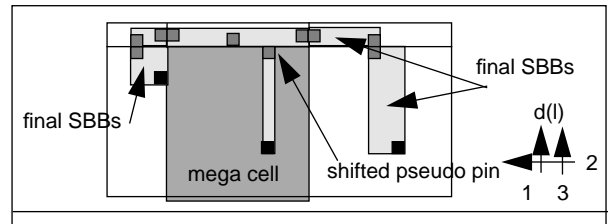Fig. 9: Net bounding box partitioning - example 2

Fig. 10: Final set of SBBs - example 2

For every net in a given design, if its bounding box does not overlap with any mega cell, equations in (7) are used to increase routing demand of covered regions as discussed in Section Three. If its NBB covers mega cells, the NBB partitioning technique in this section is applied first to generate a set of SBBs, and then equations in (7) are calculated to increase regional routing demand similarly.

5: Experimental results

Since RISA can be incorporated into most of the existing placement algorithms, in this paper, there is no comparison among different algorithms. Comparison was conducted to demonstrate the performance difference of a simulated annealing algorithm with or without RISA as one component of its cost. To apply RISA, the core areas of designs were partitioned into an array of NxN regions. Weights on other metrics such as crossing count, cell overlap and pin density were carefully tuned to achieve the best placement for comparison.

Routability of cell placements were measured by an area-based global router. Two indices, *overCon* and *dsCost*, were used to compare their routability. The global router partitions a design into an array of global grids. Each global grid contains 10x10 routing tracks. OverCon is the number of global grids in which the routing track demand is higher than that of supply. DsCost is the linear summation of the difference of routing track demand and supply if demand is greater than supply.

5.1: Six small test cases

The first set of experiments were conducted on three small industrial gate-array (Small1-3) designs and three fixed-sized cell-based (Small4-6) ones. The simulated annealing started with random solutions and tried to minimize its total cost function. Table 2 gives the basic statistics about those test cases. No clustering or

items cases	# of cells	# of nets	# of mega cells	area pct. of mega cells
Small1	3416	3750	0	0.0%
Small2	8328	9677	0	0.0%
Small3	16016	16562	1	5.1%
Small4	4281	5243	0	0.0%
Small5	5103	5538	1	2.5%
Small6	7445	7721	2	3.6%

Table 2: Basic statistics on six Industrial small designs

region constraints were imposed on any cell to allow full freedom of cell movement. With RISA, a large routing congestion area is broken into smaller ones. Congestion area smaller than the region size or located on region boundaries is invisible to the new metric. Therefore, it is rarely possible to reduce dsCost and overCon to zero. Table 3 summaries the result of having RISA as an additional cost component.

metrics cases	without RISA			with RISA		
	overCon	dsCost	CPU	overCon	dsCost	CPU
Small1	453	9245	0.31	187	3024	0.56
Small2	1821	37083	1.32	880	24923	2.43
Small3	4185	189256	6.85	787	22091	14.01
Small4	524	11385	0.47	318	9201	1.27
Small5	767	16723	0.52	387	10473	1.34
Small6	1328	34196	1.34	564	16218	2.66

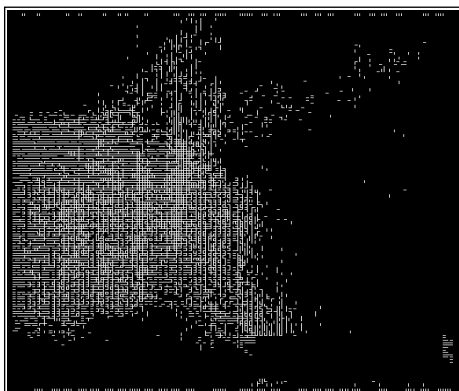Table 3: The performance of RISA on designs in Table 2

Fig. 11: Congestion map of Small3 without RISA

From Table 3, simulated annealing with RISA effectively alleviates congestion in terms of overCon and dsCost at the cost of approximately 2X CPU time, which was measured in the unit of hours on a HP750 workstation rated at ~70 MIPS. Two global

router's congestion maps of test case Small3 are shown in Figure
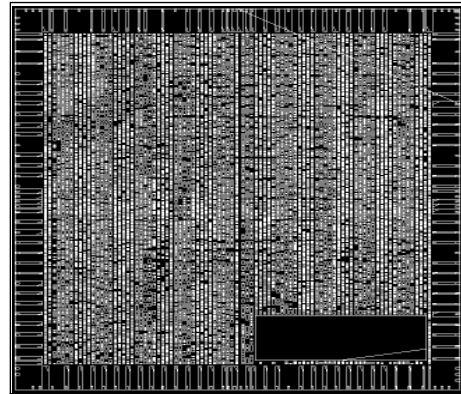

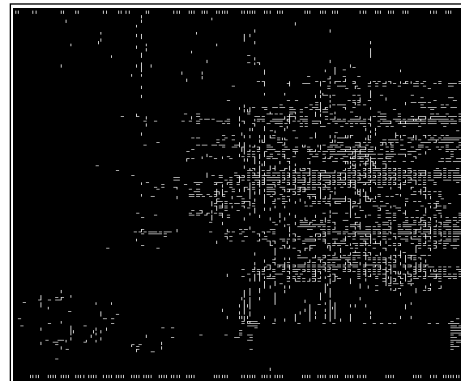Fig. 12: Cell placement of Small3 without RISA

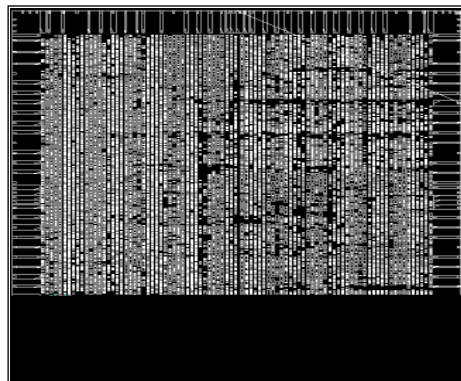Fig. 13: Congestion map of Small3 with RISA

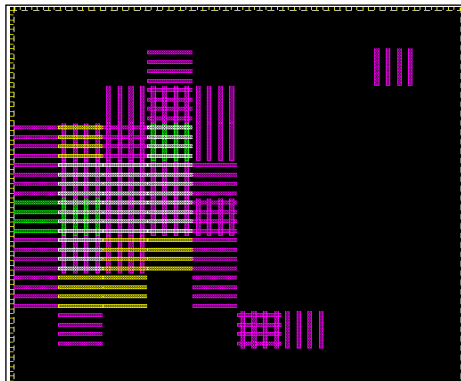Fig. 14: Cell placement of Small3 with RISA

Fig. 15: Estimated congestion map by RISA
corresponding to Fig. 11

11 & 13. A small horizontal bar in these congestion maps represents vertical routing congestion in one global grid, so does a vertical bar for horizontal. Cell placements of Small3 with/without RISA are shown in Figure 12 & 14. Note specially, in Figure 14, the unevenness of cell density, which is almost inversely proportional to congestion in Figure 13. In contrast, cell density in Figure 12 is very uniform. This demonstrates that evenness of cell does not guarantee routability. Also pay attention to the existence of congestion around the mega cell in Figure 11 but not in Figure 13 due to NBB partitioning. In Figure 15, an estimated congestion map generated by RISA based on $Max(_{v,h}D_{ij} - _{v,h}S_{ij}, 0)$ is shown. The similarity between Figure 11 & 15 proves the overall accuracy of the proposed placement routability modeling, RISA.

5.2: Six large test cases

For large test cases, RISA was incorporated into a two-level simulated annealing approach. A net list was partitioned into cell groups with ratio-cut [15]. Each group contained about 25 to 100 cells. The first level of simulated annealing finds an optimal location in the core area for each group, and transforms it into a small region that confines each cell. RISA was not applied at the first level. This greatly reduced CPU time just like the clustering in TimberWolf 7.0 [9]. At the second level, RISA was transformed into a component in the cost function of simulated annealing, together with other metrics. The experiments were conducted on six industrial large three-metal-layer designs ranging from 18,053 cells to 100,629 cells. Table 4 shows the basic statistics about these designs.

items cases	# of cells	# of nets	# of mega cells	area pct. of mega cells
Large1	18053	19321	7	7.1%
Large2	18911	25445	0	0.0%
Large3	22709	23980	1	1.5%
Large4	29532	31254	3	11.5%
Large5	38804	49130	4	4.7%
Large6	100629	106519	27	21.2%

Table 4: Basic statistics on six industrial large designs

metrics cases	without RISA			with RISA		
	overCon	dsCost	CPU	overCon	dsCost	CPU
Large1	1912	50726	1.43	977	24871	3.08
Large2	2301	67592	1.94	1423	45835	3.54
Large3	2487	75298	2.79	1350	49923	5.32
Large4	2378	66769	2.58	1429	49001	5.37
Large5	3108	104853	4.44	2395	90525	10.02
Large6	5027	243852	16.32	3827	182183	45.53

Table 5: The performance of RISA on six industrial large designs

Table 5 summarizes the experimental results. Both overCon and dsCost indices show that RISA effectively reduces congestion. Simulated annealing with RISA also runs 2X slower on large designs. The higher the percentage of mega cell area, the slower it runs. That is the trade-off between CPU time and routability. N, as in the NxN array of regions, ranges from 10 to 15 in all six test cases. The improvement on large designs could be more dramatic if cells were not confined by their regions.

6: Conclusion

In this paper, an efficient and accurate placement routability modeling strategy, RISA, was presented. It is efficient such that, given a cell placement, a congestion map could be generated based on RISA in a few seconds of CPU time. Incremental updating takes even less time. It is also accurate enough such that, as a cost component of simulated annealing, the routability of cell placement was significantly enhanced. Since routability problem normally occurs in larger ASIC designs, experiments were conducted on designs of various sizes, sea-of-gate as well as fixed-sized cell-based ones.

RISA, the placement routability modeling, has many applications in floorplanning, placement, and placement refinement. For instance, an initial placement is created with PROUD or GORDIAN, and RISA (with WDM, for better accuracy) is then applied to estimate congestion and optimize cell spacing accordingly. This part of work is currently under development and experimenting. Preliminary result has been very promising.

References

[1] U. Lauther, "A Min-cut Placement Algorithm for General Cell Assemblies Based on a Graph Representation", Proceedings of DAC, pp1-10, 1979

[2] R.-S. Tsay, E. Kuh, and C.-P. Hsu, "PROUD: A Sea-of-gates Placement Algorithm", IEEE Design & Test of Computers, pp44-56, 1988

[3] J.M. Kleinhans, G. Sigl, and F.M. Johannes, "GORDIAN: A New Global Optimization Rectangle Dissection Method for Cell Placement", IEEE International Conf. on CAD, pp427-432, 1988

[4] W.-J. Sun and C. Sechen, "Efficient and Effective Placement for Very Large Circuits", IEEE International Conf. on CAD, pp170-177, 1993

[5] M. Burstein and S.J. Hong, "Hierarchical VLSI Layout: Simultaneous Placement and Wiring of Gate Arrays" In VLSI 1983, pp45-60, 1983

[6] S. Mayrhofer and U. Lauther, "Congestion-Driven Placement Using a New Multi-partitioning Heuristic", IEEE International Conf. on CAD, pp332-335, 1990

[7] R.-S Tsay, and S.C. Chang, "Early Wirability Checking and 2-D Congestion-Driven Circuit Placement", IEEE International Conf. on ASIC, 1992

[8] Bryan Preas & Michael Lorenzetti, "Physical Design Automation of VLSI Systems", 1988, pp.96-97.

[9] W.-J. Sun & C. Sechen, "Efficient and Effective Placement for Very Large Circuits", IEEE International Conference on CAD, 1993, pp.170-177

[10] J. Soukup, "Circuit Layout", Proceedings of the IEEE, Vol.69, No.10, Oct. 1981.

[11] G. Persky, "PRO - An Automatic String Placement Program for Polycell Layout", IEEE Proceedings of the 13th DAC., p.353-361, 1977

[12] P. G. Karger & M. Malek."Formulation of Component Placement as a Constrained Optimization Problem", Proceedings of the ICCAD., p.814-819, 1984

[13] J. Jung, S. Goto & H. Hirayama, "A New Approach to the Two-dimensional Placement of Wire Congestion in Master-Slice LSI Layout Design", Trans. Inst. of Electronics and Communications Engineers of Japan, vol J64-A, No.1, p55-62, 1981

[14] S. Goto & T. Mitsuda, "Partitioning, Assignment and Placement", Advances in CAD for VLSI, Vol. 4: Layout Design and Verification, T. Ohtsuki, ed., Amsterdam, the Netherlands, North Holland, Chapter 5, 1986.

[15] Y.-C. Wei, "Circuit Partitioning and its Applications to VLSI Designs", Ph.D Thesis, UCSD CSE Dept.,September 1990.