IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE)
e-ISSN: 2278-1676,p-ISSN: 2320-3331, Volume 9, Issue 6 Ver. I (Nov – Dec. 2014), PP 05-12
www.iosrjournals.org

Performance Analysis of the Algorithms forthe Construction of Multilayer Obstacle Avoiding Rectilinear Steiner Minimum Tree

Divyaprabha[1], Dr. Prasad G.R[2]
[1] *Associate Professor, Department of Electronics and Communication, SSIT, Tumkur, Karnataka, India.*
[2]*Associate Professor, Department of Computer Science and Engineering, BMSCE, Bangalore, Karnataka, India.*

Abstract: *Routing is a phase in the physical design of Electronic Circuits. The main aim of routing in VLSI design is to interconnect the cells that have been assigned positions as a solution of the placement problem. The problem of finding Rectilinear SteinerMinimal Tree (RSMT) is one of the fundamental problems in the field of Electronic Design Automation. The obstacle avoiding rectilinear Steiner minimal tree problem becomes more important for modern nanometer IC designs which considers numerous routing obstacles incurred from power networks, pre-routed nets, IP blocks, feature patterns for manufacturability improvement, antenna jumpers for reliability enhancement etc. But at present the Multilayer Obstacle Avoiding Rectilinear SteinerMinimal Tree (ML-OARSMT) has gained much importance for modern nanometer IC designs which considers multilayers for routing, obstacles incurred from power networks etc., and preferred direction for routing layer. A ML-OARSMT connects a set of pins and set of obstacles incurred from IP blocks, power networks, pre-routed nets etc., on routing layers by rectilinear edges within layers and vias between layers such that no tree or via intersect an obstacle and the total cost of the tree is minimized. This paper provides a survey of various multilayer obstacles avoiding rectilinear Steiner minimal tree algorithms proposed and thus there is a need for an algorithm to produce better solution quality (reduced wire length) in less running time.*

Keywords: *Multilayer, obstacle Avoiding, Rectilinear Steiner Tree*

I. Introduction

Rectilinear Steiner Minimal Tree (RSMT) is a fundamental research issue about routing and is used in global routing phase of VLSI design. In modern nanometer design, there exist multiple routing layers and each routing layer has a predefined direction. Besides there exist numerous obstacles incurred from IP blocks, pre-routed nets, large scale power networks and antenna jumpers etc., distributed over various layers. As a fundamental problem with extensive practical applications to routing and wire length/congestion/timing estimations in early design stages, it is desired to develop an effective algorithm for multilayer obstacle avoiding rectilinear Steiner minimal tree problem to facilitate the design flow [1].

Given constants C_v(cost of a via) and N_l(number of layers) , a set P of pins and a set O of obstacles, a multilayer rectilinear Steiner tree connect the pins in P such that no tree edge or via intersects an obstacle in O and the total cost of the tree is minimized. Rectilinear Steiner minimal tree problem on a plane without obstacle and multilayer considerations is a NP complete problem and the presences of obstacles and multilayer further increases the complexity [2]

II. Basic Definitions

- An obstacle is a rectangle on a layer. No two obstacles overlap with each other, but two obstacles could be point touched at the corner or line touched on the boundary.
- A pin vertex is a vertex on an arbitrary layer. No pin vertex can be located inside any obstacle, but it could be located on the boundary or at the corner of an obstacle.
- Avia is an edge between two points (x, y, z) and (x, y, z+1) on layer z and layer z+1. The two points cannot locate inside an obstacle but they could be at the corner or on the boundary of an obstacle.
- The preferred direction constraint is that the odd layers only allow horizontal (vertical) edges and the even layers only allow vertical (horizontal) edges.

III. Algorithms for construction of Multilayer Obstacle Avoiding RSMT

The multilayer obstacle avoiding rectilinear Steiner minimal tree(ML-OARSMT) problem is significantly different from single-layer counterpart i.e. for the single layer obstacle avoiding rectilinear Steiner minimal tree problem(SL-OARSMT), the shortest path between two pins can be constructed by considering only the pins and corners of obstacles and it only needs to considers the wire length on a layer. However for ML-OARSMT problem considering only pins and corners of obstacles cannot always find the shortest path and via costs should be included. Thus direct extensions of existing methods of SL-OARSMT problem to the ML-

OARSMT problem may have limited solution quality or even generate some infeasible solutions.Thus algorithms for ML-OARSMT problem become important and needs to be considered. The analysis looks into existing multilayer algorithms with respect to two perspectives, the execution time i.e. the time complexity of the algorithm and the quality of the solution obtained i.e. percentage of accuracy of optimal solution. It is still an open issue for the researchers to get optimal solution with reduced execution time. Under this classification the different approaches are grouped together and further expand on the research activity related to the construction of multilayer obstacle avoiding rectilinear Steiner minimal tree with regard to the time complexity and concept behind the algorithm.

Algorithm-1

Chung-Wei Lin et.al [3, 4] has proposed the first algorithm to solve multilayer obstacle avoiding rectilinear Steiner minimal tree construction based on spanning graphs. This algorithm first constructs amultilayer obstacle-avoiding spanning graph (ML-OASG) and then finds an ML-OARSMT init. The overall time complexity of this algorithm is $O(n^3)$ in the worst case and $O(n^2 lgn)$ for practical applications. The algorithm consists of four steps.

1) Multilayer Obstacle Avoiding Spanning Graph(ML-OASG)

ML-OASG is an undirected graph that connects all vertices in P U C and no edge intersects with an obstacle in O, where P = {p1, p2, p3--- pm} is a set of pin vertices for an m-pin net and O= $\{o_1, o2---o_k\}$ is a set of k obstacles and C is corners of obstacle O. For SL-OARSMT problem two connection rules are considered. 1) SL-OASG is constructed on all pins and corner vertices. 2) Two vertices are connected if there is no other vertex inside or on the boundary of bounding box of two vertices and there is no obstacle inside the bounding box of the two vertices. If an ML-OASG is constructed only on pin and corner vertices it results in ML-OARSMT which has a larger total cost than optimal total cost. If ML-OASG is constructed using SL-OASG rule 2 it may result in infeasible solutions for ML-OARSMT problem. To overcome these problems ML-OASG should consider more essential vertices which may be on the same layer or on the other layers. Thus an algorithm is proposed so that some vertices are projected between layers and within a layer without overlapping with any obstacles. This constructs better ML-OARSMT and finds a rectilinear shortest path of any two vertices in ML-OASG.The parameter $T_n$is set as a threshold for total number of pin-vertices and corner vertices, n. This threshold is chosen because there is a tradeoff between the completeness of ML-OASGand the computational efficiency. If the number of vertices is large, the computational efficiency can be improved by removing vertex projection within a layer and the effectiveness is almost not affected. If the number of vertices is small, vertices are completely projected until the ML-OASG is not changed.

2) Multilayer Obstacle Avoiding Spanning Tree (ML-OAST)

An ML-OAST is an undirected tree connecting all pin-vertices without intersecting with any obstacle. It is constructed by selecting some edges from the OASG. The cost of an edge between vertices $v_i$and $v_j$is $|x_i - x_j| + |y_i - y_j| + |z_i - z_j| \times C_v$. This consists of 3 steps. i) pin vertices shortest path computation ii) InitialOAST construction and iii) Local refinement.[5]

3) Multilayer Obstacle Avoiding Rectilinear Steiner Tree (ML-OARST)

It is an undirected graph connecting all pin vertices by rectilinear edges with in layers and vias between layers without intersecting with any obstacles. Each slant edge of given ML-OAST is transformed into vertical and horizontal edges to obtain an ML-OARST.

4) Multilayer Obstacle Avoiding Rectilinear Steiner Minimal Tree (ML-OARSMT)

A redundant vertex is a two-degree non-pin vertex and the two edges connecting to it are parallel on a layer. For a redundant vertex the two edges connecting to it are merged and also remove overlapping edges and mark Steiner vertices.

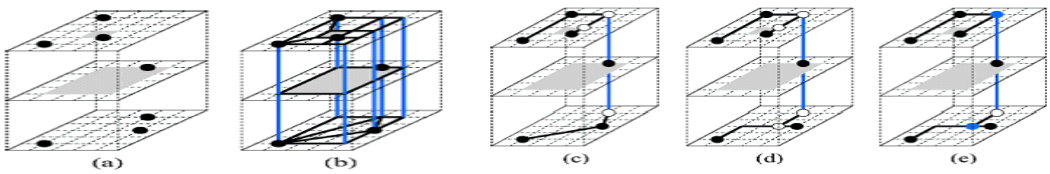

Fig-1 Four steps for ML-OARSMT construction.

Algorithm-2

ML_OARSMT problem is impractical and makes inaccurate estimation since it does not consider routing constraint preferred direction (PD). For signal integrity and IC manufacturing a PD is normally assigned to each routing layer such that all connections on a given routing layer are either horizontal or vertical. Yildiz and Madden [8] although considers PD constraint and multilayers but does not consider obstacles. Further for modern routing that handles more than ten thousand nets routability is a very important issue. Hence to increase routability most global routers weigh routing resource differently to control the allocation of routing resource that is more congested layer has a higher unit cost of wire and similarly via cost assignment. In order to estimate the routing costs more accurately and enhance routability for signal nets at the same time Chih-Hung Liu et.al[6] proposed a routing model called obstacle avoiding preferred direction Steiner tree(OAPDST)which deals with more practical routing conditions such as multiple routing layers, obstacles, preferred direction, via costs and different routing resources. ML-OASG could not be suitable for OAPDST due to its edges, that is,ML-OASG not only allows horizontal and vertical edges in the same layer but also uses slant edges. These slant edges have no information about preferred direction such that the transformation from slant edges into preferred direction edges will cause an infeasible solution.

Hence Chih-Hung Liu et.al [6] proposed a new routing graph called preferred direction evading graph (PDEG) which is guaranteed to hold at least one optimal solution for OAPDST problem. Based on PDEG a factor 2 approximation algorithm is proposed for OAPDST that provides stable and effective solutions. The overall running time of approximation algorithm is o (n^2log n). The approximation algorithm consists of two steps.

1)PDEGconstruction

PDEG consider multiple routing layers and PD constraints to make connectivity.Pin vertices and obstacle corners are first projected to adjacent layers recursively until the projection vertex locates inside an obstacle. Then line segments are extended from pin vertices, obstacle corners and projection vertices to meet the PD constraints. Finally vias are constructed on the intersections of the extending line segments between layers and a routing graph called primitive PDEG has been constructed. Although primitive PDEG is suitable for the OAPDST problem, it may lose some essential vertices such that it cannot guarantee the optimality. Thus for constructing the optimality, PDEG construction should consider all possible candidates of essential vertices whose x-coordinates (y-coordinates) are the same as those of pin vertices or obstacle corners. PDEG must be represented by evading line segments, evading via connections and the intersections of them. An evading line segment is a maximal line segment of PDEG, and a line segment is an edge connecting two vertices in the same layer. An evading via connection is a maximal via connection of PDEG, and a via connection is an edge connecting two vertices with the same x-coordinate and y-coordinate in two different layers. Thus find all obstacle avoiding preferred direction shortest paths among pin vertices on PDEG.

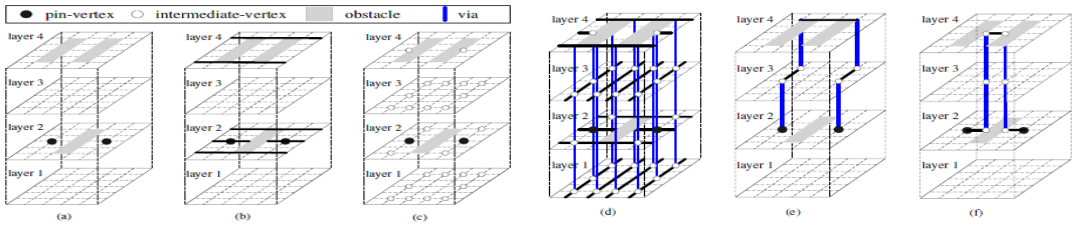

Fig-2 (a) an problem instance. (b) a direct extension of escape graph. (c) vertex projections. (d) edge construction (primitive PDEG). (e) The optimal solution from primitive PDEG. (f) The optimal solution.

2) OAPDSTconstruction

Minimum Spanning Tree Algorithm is applied on PDEG to construct an obstacle avoiding preferred direction minimum spanning tree (OAPDMST). The cost of approximation solution is no more than twice that of the optimal solution. In order to establish the approximation guarantee, cost of the approximation solution needs to be compared with cost of optimal solution.

Algorithm-3

Iris Hui-Ru Jiang [7] has proposed a unified algorithm to solve both single and multiple routing layers. Exact algorithm-1 uses connection graph based approach. This approach first connects all pins and obstacle boundaries by a connection graph and then builds an OARSMT from it. This approach maintains the global geometrical information of pins and obstacles and generates best results. Iris Hui-Ru Jiang [6] has proposed an algorithm based on construction by correction which constructs a spanning or Steiner tree first and then corrects

the edges intersecting obstacles. As obstacles are not included into the initial graph it requires a good modeling of the impact of obstacles. The proposed algorithm consists of three steps.

1) Delaunay Triangulation on apseudo plane

All pins are projected onto a pseudo plane and a Delaunay Triangulation (DT) is then constructed over all pins on this plane. A DT for a set p of vertices in a plane is a triangulation such that no vertex in p is inside the circumcircle of any triangle in DT (P). To maintain this illegal edge is flipped to a legal one, instead of discarding them. These extra edges are preserved because they may possess more global information than legal ones and may result in better solution. This step takes O (m lgm) pattern time for DT construction.[11]

2) Obstacleweighted MST construction

Kruskal's algorithm is used to build an obstacle weighted minimum spanning tree over the DT. The edge weights is not required to be exact but expected to be correlated to the cost of the final RSMT. Hence a simple but effective formula is used to estimate the impact of obstacles. The obstacle penalty considers only the obstacles that completely pass through the bounding box between pi, pj vertices horizontally or vertically. A parameter α is used to further tune the weight. This parameter is used to reflect the congestion of obstacles. This step takes o (m $(lgm)^2$) time for kruskal's algorithm. $W(p_i, p_j) = \alpha (| x_j - x_i | + | y_j - y_i| + C_v|z_j - z_i|) + op(p_i, p_j)$

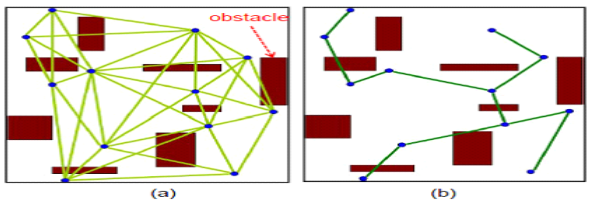

Fig-3 (a) Delaunay Triangulation for pins on apseudo plane. (b) The obstacle-weighted MST.

3) Rectilinearizationand 3D U-shaped pattern refinement

Each edge on MST is rectilinearized and then the new edge is further used to reduce the total costby novel three dimensional u shaped pattern refinement. In this algorithm the two operations are combined into one and applied to the MST edge by edge. Rectilinearization if performed on a three dimensional escape graph, based on dijkstra's shortest path algorithm. An escape graph introduces obstacles into the Hanan grid, lines are stretched from pins and obstacle boundaries along three axis. The segments intersecting obstacles are prohibited. At least one OARSMT is embedded in escape graph. If processing edge have zero obstacle penalties it is rectilinearized by a l-shape or zigzag. After rectilinearization generalized 3D U-shaped pattern refinement is applied to the edge for cost reduction. Three connected vertices form a U-shaped pattern. All 3 U-shaped patterns are of 2 types.

Standard U-shape: In this middle vertex is located within the middle segment of the u. There are3 cases and pattern can be rerouted using their Steiner point.

Degenerated U-shape: In this the middle vertex is located in one turning corner .step(c) takes $O(n^3)$ time for the 3D extended escape graph construction, dijkstra's algorithm,3DU-shaped pattern refinement.

If there are obstacles around a U-shaped pattern, the Steiner point should be shifted accordingly which improves the total cost. Steps 1 and 2 consume very short runtimes. Refinement may rip-up and reroute many times thus taking more runtime. Even so runtimes are still stable not increasing much from SL-OARSMT to ML-OARSMT.[13]

Algorithm-4

Interconnect optimization for VLSI circuits have been an area of research in VLSI physical design automation. With rapid scaling of feature sizes, interconnect constitutes a substantial portion of system delay. Modern performance driven cannot ignore the impact of interconnect wires. Most interconnect optimization research focuses on a rectilinear planar abstraction of the routing surface. Issues like via counts, preferred direction routing and even layer dependent design rules are frequently ignored. But Mehmet can Yildiz Patrick h. Madden [8] has proposed preferred direction multilayer routing model and has developed an effective Steiner tree approximation algorithm for it. This algorithm considers the multilayer, preferred direction constraints, cost and location of vias, layer dependent routing costs. In modern design circuit interconnect can be placed on a number of different routing layers. Current fabrication processes may have as many as 8 layers, with 4 layers being common. Vias are used to connect adjacent layers and may have varying cost in terms of electrical resistance, routing area required and reliability. Routing cost on each layer is dependent on material used, for IC's electrical resistance on lower metal layers is quite different from that of an upper layer and each layer may

have a unique minimum feature size and spacing, some layers may be heavily congested, routing space limited and global router wish to weigh individual layers differently resulting in a preference for Steiner tree topologies that useless congested areas. This papers models cost by associating a constant cost factor with routing in particular direction on each layer and fixed cost for via between pair of layers. The performance bound of minimum spanning tree cost to Steiner minimal tree cost under this model is 2:1.

The proposed approach is based on an edge based heuristic for Steiner routing. First minimum spanning tree is constructed using prim's algorithm and approximate edge cost method. First a pair of an edge is connected to vertex (at suitable point) a cycle is created. This cycle is broken by removing the longest edge on any generated cycle. The tree length can be reduced. Gain computations are performed for each vertex edge pair. An optimal Steiner point to merge an edge and vertex is not necessarily median of co-ordinates. This modification is performed in greedy manner and process is repeated until no further improvement can be made. The complexity of finding possible pairing is O (n^2) and the number of passes required is generally small resulting in a complexity of O (n^2) for entire algorithm.

Algorithm-5

Hsin-Hsiung Huang et.al [9] have formulated the timing-driven multiple-layer obstacle-avoiding rectilinear Steiner minimal tree (TML-OARSMT) problem and proposed an algorithm to solve the problem. The objective is simultaneously to minimize the maximum source-to-terminal length and minimize the number of vias. The proposed algorithm contains the three steps, including the multiple-level timing-driven partitioning, the multiple-level routing tree construction with obstacles, and the timing-driven multiple-layer balanced tree construction. Lin et al [2, 3]had proposed a wire length driven algorithm which achieved excellent routing results of the total wire length with many number of vias. But they can achieve the much better results if the timing issue such as maximum source-to-terminal length (L_{max}) is considered. Also to minimize the maximum source-to-terminal delay, the $L_{max}$and number of vias should be considered. Thus a timing driven algorithm which considers $L_{max}$and the number of vias during a routing tree construction is proposed.

1) Timing-Driven Multiple-Level Partitioning

1.1. 1st-Level partitioning by asource position

To minimize the L_{max}between layers, the terminals are divided into the k sub-regions by the source position. During this step, all terminals except source are projected onto a layer. For each sub-region, a terminal is taken as the center of gravity, if a center of gravity is inside the obstacles, it will be removed outside the obstacles to build a straight via. Further a center of gravity is projected to all layers and many additional vias are generated.

1.2. 2nd-Level partitioning by the center of gravity

To minimize the L_{max} withina layer, acenter of gravity is taken to divide each sub-region. Actually the 3st-level (or 4nd-level. . .) takes the center of gravityto divide the routing area into aset of sub-regions. This procedure is called the second partitioning which minimizes the detour routing within alayer. For these terminals in each region, they will be directly connected to a center of gravity. If we project the terminals of each layer and discover the center of gravity is located in the obstacles the center of gravity is removed outside the obstacles.

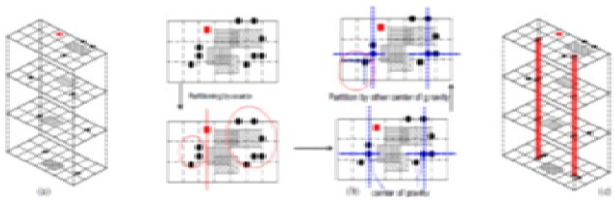

Fig-4 Multiple level partitioning for amultiple layer system

2) TheRouting Tree construction

A concept of the two-level routing tree construction is incorporated to make a tradeoff between the $L_{max}$and runtime. A Delaunay triangulation-based (DT-based) method is first applied and the result is accepted if all edges are valid. Otherwise a spanning graph-based method is used to get the feasible result.

2.1. 1st-Level Routing tree construction

DT is applied for the all corner points of obstacles and terminals and then removes the redundant edges. If the result contains an invalid edge inside the obstacles the following method is applied.

2.2. 2nd-Level Routing tree construction

A spanning graph is constructed according to the terminals and obstacles that always obtain valid edges and then the minimal spanning tree is generated. Finally a spanning tree is transformed into a Steiner tree.

3) Timing-Driven Balanced tree construction

The routing tree is constructed for a source and many centers of gravity, the L_{max}will be degenerated. Therefore a source-projection technique projects a source to the middle layer and produces an additional projected terminal, to minimize the L_{max}.

Algorithm-6

Jia-Ru Chuang et.al [10] has proposed a very simple and effective approach to deal with multilayer obstacle avoiding preferred direction rectilinear Steiner minimal tree (ML-OAPDRSMT) problem. The time complexity of this algorithm is O((m+n) log (m+n)) time. Where m is number of pins in p and n obstacles in o. This algorithm uses less running time to build a smaller RSMT in multiple routing layers with preferred direction and consider obstacles. This is because an algorithm is proposed that connect every 2 pins directly and thus does not construct a spanning graph and then find a spanning tree from it has done in previous algorithms.

In this methodology all pins and all obstacles are sorted in increasing order according to their x-coordinates(y-coordinates) to determine a sequence for pin connection called pin ordering stage. The results are recorded by a list L= {p_1,p_2, -----p_m}. Then it serially connects every two neighboring pins by a two pin full Steiner tree called multilayer obstacle avoiding preferred direction rectilinear full Steiner tree (ML-OAPDRFST). After all neighboring pins are connected the final ML-OAPDRSMT can be obtained. There are two cases to be considered depending on whether there exist obstacles between 2 pins to be serially connected.

1) Withoutobstacles

For the first pair of pins (P_1, P_2) in L they are directly connected by an L-shaped route since the shortest distance between them is Manhattandistance. Next aproper point must be chosen in the built sub-rectilinear Steiner tree to obtain a shorter wire length for connecting to the next pin. If pin P_{i+1}'sy-co-ordinate is inside the Steiner range of latest FST F_i, pin p_{i+1} is directly connected to Steiner segment siby ahorizontal line. Otherwise it tries to connect to one of endpoint in $s_i$to obtain a shorter distance. For two pin full Steiner tree that connects two continuous points $p_i$and p_j(x_i< x_j), the Steiner segment denoted by $s_j$of the FST is the vertical segment formed by two points $p_a$and p_j, where the coordinate of pa is(xi, xj) and the Steiner range indicates the vertical range of sj, which is between y_a and y_j. Thus it uses backtracking algorithm to build Steiner tree to find a suitable connection point $c_i$for connecting to pin p_{i+1}.thus using this algorithm the wire length in the resulting Steiner tree can be reduced.

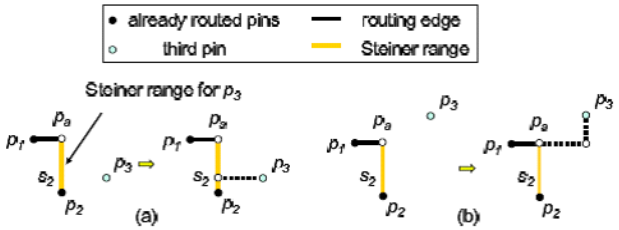

Fig-5 Finding aconnection point in aSteiner segment.

2)Withobstacles

Even if obstacles exist between 2 pins a shorter full tree is constructed by passing through obstacles directly with a horizontal path by adding an additional Steiner point.it uses a greedyheuristic algorithm to construct a Steiner tree.This uses five functions.

a)R_{k+1} = project(r_k,o_{k+1}): Which projects temporary connection point r_k to left boundary of o_{k+1} to obtain a new temporary point r_{k+1} that has same y-coordinate and layer number as r_k.

b)L (r_k') = perpendicular project L (r_k): It tries to project rk to nearest horizontal layer parallel to its plane if not blocked by an obstacle. Otherwise projected to nearest vertical layer. rk' has same coordinate as rk except layer number is changed.

c)cover (r_{k+1}, ok_{+1}): It checks whether point r_{k+1} is completely covered by obstacles located above and below the layer of r_{k+1}

d) Block (r_k,o_k): It checks whether r_k is blocked by o_ki.e. layer of r_k is same as layer of o_k and a horizontal path from r_k will intersect o_k.

e) cover-obstacle (r_{k+1}): It returns one of theobstacles that covers the r_{k+1}.

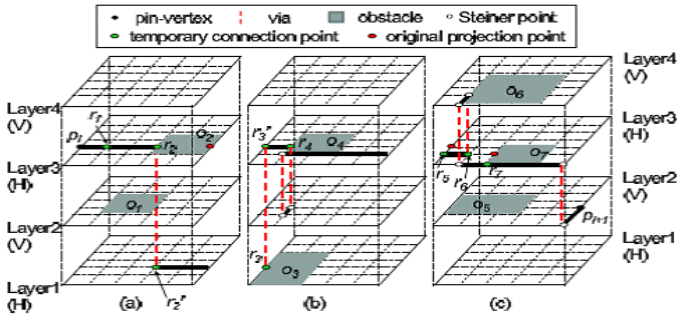

Fig-6 Example for connecting $P_i$in (a) to P_{i+1}(c) with seven obstacles

Algorithm-7

Lin et.al [3,4] ML-OARSMT algorithm and Liu et al [6] PDST algorithm and Chuang [10] algorithm behaves like a quadratic time algorithm. The inefficiency of the algorithms in [3,4] and [6] results from quadratic size of a routing graph and inefficiency of the algorithm in [10] results from their backtracking procedure of finding Steiner points. Therefore Chih-Hung Liu [11] has developed a more efficient algorithm for ML-OARSMT problem and the OAPDST problem which behaves like a sub quadratic time algorithm for practical applications. The size of a ML-OARSMT is Ω (n^2) which indicates that the multilayer model is very different from its 2d counterpart and thus transforms a multilayer instance into a 3d instance and thus enables to employ computational geometry techniques.

He proposed a reduction that transforms a multilayer instance into a 3Dinstance and thus enables to employ computational geometry techniques. Based on the reduction visibility graph is computed and utilize Steiner point based framework to develop a 4-step algorithm for the ML-OARSMT problem. The ML-OARSMT construction takes O ($mn^2 \log^2 n$) time in the worst case and O($n \log^2 n$) time for practical applications.

1) Multilayer Visibility Graph Construction (ML-VG)

Based on the reduction the Chih-Hung Liu has used the algorithm in [14] to construct the (ML-VG) G(V, E) for a multilayer instance I. It first computes 3DVGG' (V', E') for rectilinear box R (I) and then delete vertices in z_∞ and $z_{-\infty}$ from G' to construct G. The 3D-VG algorithm can handle rectilinear obstacles each of which are a union of intersecting boxes and are feasible for the reduction.

2) Steiner point selection: Steiner points are selected from ML-VG. Kruskal algorithm is used to select Steiner points.

2) Minimum Terminal Spanning Tree (MTST)

All pin vertices and selected Steiner pints are considered as terminals and thus construct the MTST of ML-VG as the initial solution. For a graph G (V, E) and a terminal set s $\underline{c}$v , a minimum terminal spanning tree(MTST) connects all the vertices in s using a set of terminal paths among vertices in s such that the sum of lengths of those terminal paths is minimized, where a terminal pathis a path in G between two vertices in s passing through vertices in v \ s.[13]

3) Refinement

The initial solution obtained in step 3 is refined using U-shaped patterns to reduce more wire length.

In general, a U-shaped patterns results from a line segment which has different number of edges incident on its two sides. However in multilayer model, line segments indifferent layers may affect each other. The line segment of a 2DU-shaped pattern is replaced with a collection of line segments.

The last three steps are directly applicable for the OAPD-ST problem, and thus only graph construction changes.

Lee and Yang [11] proposed an O(n log n)-time algorithm to transform a 2-layer instance with PD constraint into a 1-layer instance without PD constraint.The transformation in [11] and the ML-VG construction is combined to construct preferred direction visibility graph (PD-VG) for an OAPD-ST problem instance I. The OAPD-ST construction takes O($mn^2 \log^2 n$) time in the worst case and O($n \log^2 n$) time for practical applications.

Table-1 Comparison of important ML-OARSMTalgorithms

Sl.No	Algorithm	Variant Constraints Considered	Time Complexity	Advantages	Disadvantages
1	ML-OARSMT	Multilayer, Obstacles	$O(n^3)$	Optimal Solution	High runtime quadratic time
2	Efficient Multilayer Routing based on OAPDST	Multilayer, Obstacles, Preferred Direction	$O(n^2 \log n)$	Optimal Solution	High runtime quadratic Time

3	Unification of OARST construction	Multilayer, Obstacles, Preferred Direction	------	Unifies solution for SL And ML-OARSMT	High runtime
4	Preferred Direction Steiner trees	Multilayer, Preferred Direction	$O(n^2)$	Fast(Good runtime)	Obstacles not considered
5	Timing Driven RSMT	Multilayer, Obstacles	-----	Minimizes maximum source to terminal length and number of vias	Preferred Direction not considered
6	Efficient ML-OAPDRST construction	Multilayer, Obstacles, Preferred Direction, Different routing resources	$O((m+n)\log(m+n))$	Fast (Good runtime) Sub quadratic time	Solution quality inferior to efficient algorithm ML-OARSMT
7	Efficient algorithm for ML-OARSMT and OAPDST construction	Multilayer, Obstacles, Preferred Direction	$O(n\log^2 n)$	Good quality, fast (Good runtime)	Difficult to test for random cases

IV. Conclusion

This paper presents various algorithms for computing the multilayer obstacle avoiding rectilinear Steiner minimum tree.Table 1 provides the comparison of the important existing algorithms for the construction of multilayer, obstacle avoiding and preferred direction RSMT. The selection of an algorithm is based on various parameters, the important ones being good solution quality (wire length) and runtime (time complexity).all existing heuristics for ML-OARSMT and PDST construction cannot achieve both small wire length and small runtime for large values of multiple pin nets.But, the decision regarding the selection of the algorithm for multilayer, obstacle avoiding and preferred direction RSMT is a trade-off between efficiency and runtime. Therefore, there is a need for an algorithm or approach for the construction of multilayer obstacle avoiding preferred direction RSMT to address the above challenge.

References

[1]. Sherwani Naveed A. Algorithms forVLSI Physical Design Automation. Kluwer Academic Publishers, 1998.
[2]. Garey, Michael R., andDavid S. Johnson, “Therectilinear Steiner tree problem is NP-complete, SIAM Journal ofApplied Mathematics, pp. 826-834, 1977.
[3]. Chung-Wei Lin, Shih-Lun Huang, Kai-Chi Hsu, Meng-Xiang Lee, and Yao-Wen Chang,” Multilayer obstacle-avoiding rectilinear Steiner tree construction based on spanning graphs”, IEEE Transactions On Computer –Aided Design, Vol.27, No. 11, pp.2007-2016,November2008.
[4]. Chung-Wei Lin, Shih-Lun Huang, Kai-Chi Hsu, Meng-Xiang Lee, and Yao-Wen Chang,” Efficient Multi-Layer obstacle-avoiding rectilinear Steiner tree construction”, in Proc.ICCAD.pp 380-385,2007.
[5]. Yao-Wen Chang C-W.Lin, S-Y. Chen “Obstacle-avoiding rectilinear Steiner tree construction based on spanning graphs,”IEEE Transactions, Computer-Aided Design,vol.27,no4,pp.643-653,2008.
[6]. Chih-Hung Liu, Yao-Hsin Chou, Shih-Yi Yuanx, and Sy-Yen Kuo,” Efficient multilayer routing based on obstacle-avoiding preferred direction Steiner tree”, in Proc. ISPD, pp.118-125,2008.
[7]. Iris Hui-Ru Jiang, Shung-Wei Lin and Yen-Ting Yu,” Unification of Obstacle Avoiding Rectilinear Steiner Tree Construction”, 978-1-4244-2596-9/08 2008 IEEE.
[8]. Mehmet Can Yildiz Patrick H. Madden,” Preferred Direction Steiner Trees”,IEEE Transactions, Computer-Aided Design Of Integrated Circuits And Systems,Vol.21,No.11.
[9]. Hsin-Hsiung Huang, Hui-Yu Huang, Yu-Cheng Lin, Tsai-Ming Hsieh “Timing-driven obstacles-avoiding routing tree construction for amultiple-layer system”, 978-1-4244-1684-4/08 ,2008 IEEE.
[10]. Jia-Ru Chuang and Jai-Ming Lin, “Efficient Multi-Layer Obstacle-Avoiding Preferred Direction Rectilinear Steiner Tree Construction”, 978-1-4244-7516-2/11/ 2011 IEEE.
[11]. Chih-Hung Liu, I-Che Chen, and D. T. Lee, “An Efficient Algorithm for Multi-Layer Obstacle-Avoiding Rectilinear Steiner Tree Construction”, DAC 2012, June 3-7, 2012, San Francisco, California, USA ACM, 978-1-4503-1199-1/12/06.
[12]. Iris Hui-Ru Jiang and Yen-Ting Yu,” Configurable Rectilinear Steiner Tree Construction for Soc and Nano Technologies”, 978-1-4244-2658-4/08 2008 IEEE.
[13]. D. T. Lee And C. D. Yang, “Finding rectilinear paths among obstacles in atwo-layer interconnection model," Internal. J.Comput. Geom. Appl., Vol. 7, No.6, Pp. 581-598, 1997.
[14]. K.L.Clarkson, S. Kapoor, and P.M. Vaidya, “Rectilinear shortest paths through polygonal obstacles in O (n log ^{2}n) time,’ in Proc SCG, pp. 251-257,1987.