A Study on Obstacle Avoiding Rectilinear Steiner tree Algorithms

Shyamala G
Department of Computer Science and Engineering
BMS Collegeof Engineerin, Bangalore, Karnataka, India

Dr. G R Prasad
Department of Computer Science and Engineering
BMS College of Engineering, Bangalore, Karnataka, India

Abstract- The Rectilinear steiner tree problem is to find a minimum length rectilinear interconnection of a set of points in the plane. In practical routing applications, macro cells, IP blocks, and prerouted nets are the obstacles present in the routing area. Obstacle avoiding Rectilinear Steiner minimal tree algorithms are useful for practical routing applications. This paper presents a comprehensive survey on Techniques of OARST, with an emphasis on the problems routing multiple nets with good wirelength and time efficiency in VLSI circuits under various design styles. The survey begins with a coverage of traditional approaches such as modeling geometric problem to different kind of graph problem and then discusses other approaches such as steiner point selection method, Full steiner tree method or Maze routing method, which have attracted a good deal of attention recently. The group of OARSMT techniques and several of its variants are then overviewed. While many traditional techniques focus on the conventional objective of managing good wirelength and time among complex rectilinear obstacles, newer objectives have come into play with the advances in VLSI technology. This Paper is intended to survey various Obstacle avoiding Rectilinear Steiner tree algorithms available till date

Keywords – RSMT, OARSMT, MST, Track graph, escape graph, Spanning Graph.

I. INTRODUCTION

Construction of Rectilinear Steiner minimum tree (RSMT) is an important problem in VLSI physical design. It is used both in detailed and global routing phase of VLSI design. The original RSMT problem assumes no obstacles in the routing region. In today' s VLSI design there can be many routing blockages like macro cells, IP blocks and pre-routed nets. Therefore, The RSMT problem with blockages, called OARSMT problem is widely studied.

Let $P=\{p_1,p_2,p_3\ldots p_n\}$ be the set of pins for m pin met.Let $B=\{b_1,b_2,b_3\ldots b_k\}$ be a set of rectangular blockages. Let $V=\{v_1,v_2,v_3\ldots v_m\}$=PU{Corners in B} as the vertex set in the problem where each v_i has coordinates (x_i,y_i). The rectangular blockages has 4 corners, we have n<=m+4k.

The rectilinear distance between v_i & v_j is given as $|x_i-x_j|+|y_i-y_j|$. A OARSMT connects all pins through some extra points to achive a minimal total length, while avoiding the intersection with any blockage in the design.

Hanan[1] was the first to consider the rectilinear adaptation of steiner tree. Garey and Johnson[2] showed the problem of construction of RSMT is NP-Complete. Hwang[3] showed that the ratio of cost of RMST (Rectilinear minimum spanning tree)to the cost of RST(Rectilinear steiner tree)is <= 3/2.

Joseph Ganley & Cohoon[6] gave the first model for OARST corresponding to instance size rather than size of routing area. They introduced a cluster method in which multiterminal net can be partitioned by some 3 to 4 terminal group to construct OARST optimal trees can be computed **as** efficiently **as** good heuristic

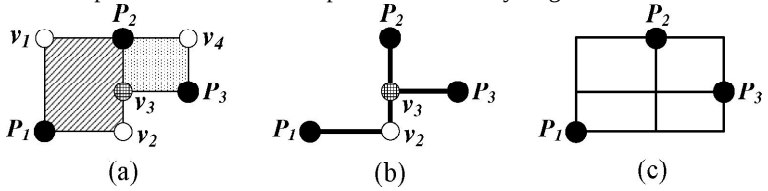


Fig. 1. Steiner points for the RSMT problem. (a) Three-pin instance.[23]

(b) RSMT of (a). (c) Hanan grid.

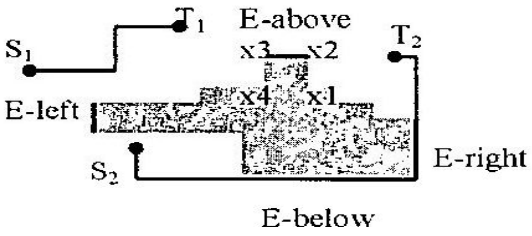


Fig. 2. Obstacles

II. TERMINOLOGIES

Minimum spanning Tree
Given a set of points P, minimum spanning tree(MST) is the minimum length tree over P and contains no cycles.
Spanning Graph
Given a set of points V, an undirected graph G=(V,E) is called spanning graph if it contains minimum spanning tree.

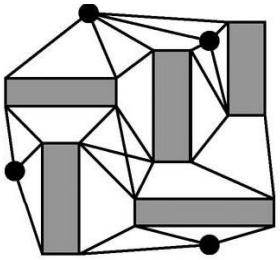


Fig. 3. Spanning graph

Track graph
A grid like structure, which consist of rectilinear tracks defined by the obstacles and terminals.The Vertex and edge number in a track graph is $O(r^2)$ where r is the extreme edge number of all obstacles

Escape graph
Escape graph is constructed by escape segments.The number of vertex in escape graph can be O(n2), in worst case, where n is the sum of pins and blockage boundries.

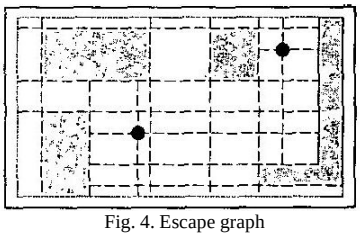


Fig. 4. Escape graph

III. LITERATURE SURVEY

This section provides various concepts and techniques related to OARSMT construction. The works of different authors have been classified based on the basic concepts involved.

A. *Escape graph based Algorithms*

Joseph L Ganley and James Cohoon[6] proposed an optimal obstacle avoiding rectilinear Steiner tree in time corresponding to instance size rather than size of the routing area. Line search routing heuristic[5] is used to describe escape graph.
First, horizontal and vertical line sweeps of the boundary segments of obstacles are performed to construct the escape segments[6]. The sweeps are done in O(m log m) time, where m is the number of obstacle boundary segments. Construction of escape graph from escape segments is done in O(n) time, n is the number of instructions. The total time complexity of generating the escape graph is O(max{n,m log n}) where n is $O(m^2)$in worst case.
[7] they present a theorem, for standard RST problem, that transforms an instance of OARST problem to a graph problem whose size is a function of input size rather than the routing area. Steinerization heuristics can be speed up by batching technique B3S where time complexity is O(rkn) where r is the iterations required.
Yang, Zhu et al[8] proposed a efficient 2 step heuristic for RSMT with obstacles where running time is O(mn), m is the number of obstacles and n is number of terminals.
Step1 is to construct RSMT without obstacles and then in step2 the primary tree T is transformed by substituting edges around the obstacles for those edges of T inside the obstacles.

Yu Hu, Zhe Feng[30] proposed a FORSTer algorithm which is a 3-step heuristic.
In *Step1*, all the terminals are partitioned into some subsets in the presence of
obstacles by some rules. Connected components, regarded as hyper-graph, are generated. Then in *Step2*, terminals are connected in each connected graph with one or more trees, respectively. As a result, the separated sub-trees are constructed after this step. In *Step3* the forest is connected consisting of the trees constructed in *Step2* into a completed Steiner tree spanning all terminals while avoiding all obstacles. Two kinds of algorithms, called ACO-RSMT and GFST-RSMT, are proposed to construct OARSMT in a connected graph in *Step2*, which are suitable for different situations.

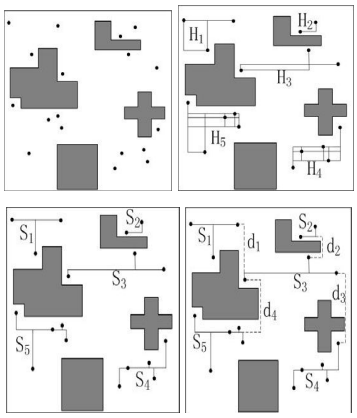


Fig. 5. Step 1-4 of FORST-er[30]

B. *Track graph based algorithms*
Yu,jing,Hong et,al[9] proposed a heuristic to tackle complex obstacles and keep high length performance. A track graph T is generated and then it is reduced to T' by T-reduction algorithm. After that ants are placed on terminals the need to be connected using ant colony approach[31].This approach gave a good speedup with 100 terminals connected in 32sec with 9 rectangular obstacles.

C. *Escape and spanning graph based*
Chang,wen,et,al[10] proposed a construction by correction approach which builds graph for Steiner tree construction. Then an obstacle –weighted minimum spanning tree(OWMST)is constructed via the connection of all terminals in T. Then OWMST is routed on the escape graph to construct an OARSMT with shortest average total wire length in a short time.

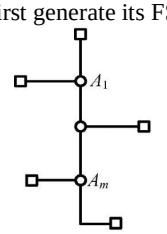
D. *Spanning graph based algorithm*

Zion shen, Chirs Chu et.al[11] proposed a O(nlogn) time algorithm to construct spanning graph for Rectilinear steiner minimal tree with blockages.
If V_s is the point set which includes all corners of B_s and pins of a given net, all points in V_s are sorted by their x coordinates in non decreasing order. A sweep line algorithm which uses active set A is built and dynamically keep A by adding and deleting points for set A. RSMT is constructed by first constructing undirected graph G, to find minimum spanning tree T1 of G1 and construct sub graph G_s of G.Then find MST T_s of G_s and construct Steiner tree T_n from T_s by deleting edges in T_s if necessary.
Wu,Gao,Wang et al[12] proposed an algorithm which applies partitioning method to find MST on the graph and remove the segments intersecting obstacles. Then ant colony based approach is used to connect the subtree into single tree and get an OASMT. This method works on spanning graph as it is efficient connection graph which can produce OASMT with good wire length performance. After obtaining OASMT a rectilinearization heuristic is used to generate OARSMT and refinement method is applied to improve it further. Runtime is better than previous works and has 54.37% compared with best.Long, Zhou & Memik[13] Proposed a algorithm which generates sparse obstacle avoiding spanning graphs. Then a fast algorithm for minimum terminal spanning tree(MTST) is applied. An edge based heuristic to perform local and global refinement is performed which give small wirelength. To increase the quality a refinement technique called segment translation is applied. The time complexity is O(n log n). Compared to [11] this approach achieves better quality and lower complexity. A sweep line algorithm is used to construct OASG . Then an extended Dijikstra's algorithm generates shortest path terminal forest for any non-negative weighted graph.Extended kruskal's is used for MTST construction. This extended Dijkstra-kruskal algorithm solves the MTST in O(nlogn) time.
Borah-Zhou edge based refinement involves computing the closest on MTST vertex for each vertex, sorting the vertex-edge pairs according to their gain, transforming edge pairs into vertex pairs and performing merging tree least common ancestor query. Refinement is done using segment translation where OARST is dissected into disjoint segment by doing tree traversal in O(n) time
Chung-wei Lin, et al[14] proposed a algorithm based on spanning graph and guarantees to find an optimal OARSMT for any two pin nets and higher pin nets with shorter wire length comparatively. An OASG connecting all vertices in PUC is constructed. Edges are selected complexity is O(n3)in worst case and O(n2 lg n) for practical applications.
The previous works for OARST was mainly to generate initial solution without considering obstacles, and then legalize edges intersecting obstacles. This kind of approach may lack global view of obstacles and solution quality may be limited [8]. Another previous work is to construct a connected graph embedding a valid solution and apply graph algorithms[21]
Liu, Yuan et.al[15] proposed a new framework to develop a O(nlogn)time algorithm which directly generates essential solution components without constructing a routing graph or generating invalid solutions. Analysis of the geometry mapping of previous works is done and critical path is derived which guarantees the existence of optimal solution and also increases the overlapping between different paths to improve wire length. This algorithm first generates critical paths as solution components which ensures existence of desirable solution. An OAST connecting all pin-vertices is constructed by selecting those critical paths. OARST is constructed from OAST by transforming slant edges to rectilinear. The wire length of OARST is reduced by O(n log n)using dynamic refinement scheme.A new framework to directly generate O(n) critical paths as essential solution componemts without constructing a routing graph or generating invalid initial solutions. It increases the overlapping between different paths for improving the wire length and performs on O(n log n) time using dynamic local refinement scheme.
Critical paths are generated in O(n) and O(nlogn) which guarantee the existence of optimal solutions analyzing OASG and MTST algorithms. Longs[13] MTST algorithm is simulated on Lin's[14] OASG to analyze the corresponding geometry mapping and conclude the bottleneck of the time complexity.
A greedy method is proposed to construct an OAST on those critical paths and attempts to increase the overlapping between different paths.
OARST is constructed from OAST by transforming all slant edges inti rectilinear ones as all edges of OAST are visible.Dynamic local refinement are performed to reduce the wire length in O(n log n) time based in U-shaped refinement, if a segment has more adjacent segments on one side then the other side, moving this segment to former side may reduce the wirelength.The time complexity of whole construction of each phase is O(nlogn)

E. Algorithms Based on Full Steiner Tree

A full Steiner tree (FST) is a rectilinear Steiner minimum tree in which every terminal is a leaf node (of degree one). Any Steiner minimum tree can be decomposed into a set of edge-disjoint FSTs by splitting at terminals with degree

more than one Since FSTs are much easier to construct than Steiner minimum trees, most of the exact algorithms for the construction of a Steiner minimum tree will first generate its FST components.



GeoSteiner is a software package for computing Steiner minimum trees. The algorithm makes use of a two-phase approach, consisting of a FST generation phase and a FST concatenation phase, to construct a Steiner minimum tree. In the first phase, a set of FSTs are generated such that there is at least one Steiner minimum tree composed of the FSTs in the set only. In the second phase, a subset of FSTs are selected and combined to form a Steiner minimum tree. On the rectilinear plane, GeoSteiner remains the fastest exact algorithm for the RSMT problem, but it cannot be applied when obstacles exist in the routing plane.

Liang Li et,al[16] Proposed a work base on geosteiner approach, modified and extended to allow rectilinear blockages. FST is the basic concept used in geosteiner which can handle hundreds of pins with multiple blockages, generating optimal solution in a reasonable time. This framework of geosteiner is composed of FST generation and FST concatenation. All FST are generated after adding the virtual pins. Then an ILP is setup with those virtual pins can be included. After generating useful FST we setup ILP to select and concatenate a subset of FST to construct OARSMT connecting all points in v with minimum total length.

Tao Huang, Evangeline Y Young[17]proposed a more efficient method to construct OARSMT based on [16]. Only the virtual terminals were introduced to each obstacle to reduce the total number of constraints and improve the performance of the algorithm. More efficient separation algorithm is used and obstacles are handled in incremental way to reduce running time for large problems

The previous work was continued in [18] with a geometric approach to solve the OARSMT problem among complex obstacles. The optimal solution is constructed by concatenation of Full Steiner Tree FST among obstacles. Virtual terminals are added in such a way that there is at least one virtual terminal on every essential edge of all the obstacles. OARSMT construction is done in the following steps.

1) FST Generation- To generate FST of 3 or more terminals, a modified version of recursive algorithm of Zacherian[35] is used. FST with 2 terminals are generated by using more efficient approach.

2) Pruning of FST is done to reduce the number of FST' required to construct OARSMT. It works by growing FST f to larger trees and test if these trees can exist in an optimal solution.

3) Concatenation of FST is done using the algorithm of [17]

The performance of this method for all bench marks around 60% of FST can be eliminated and Running time of the pruning procedure is less than half the total time.

A new approach was proposed by [19] based on geosteiner approach in which FST are constructed and then combined into a rectilinear steiner minimal tree(RSMT). Virtual terminals are introduced to each obstacle and with these virtual terminals the structure of FST with blockages are same as those FST in absence of obstacles.

In FST generation phase, virtual terminal pruning process is developed which can effectively reduce the number of resulting FST. Screening tests are done to handle virtual terminals and blockages, and an efficient approach to construct a 2 terminal FST when virtual terminal exist.

For FST concatenation a new formulation is proposed with blockages. In branch and cut search, new separation algorithm is developed to adapt to the present of virtual terminals.

This approach can handle problems with hundreds of terminals in the presence of multiple obstacles, generating optimal solution in reasonable amount of time.

Continuing the work of [18][19] proposed Obsteiner[20] to construct OARSMT among complex obstacles based on geosteiner approach.

OARSMT is partitioned into a set of FST by splitting at real terminals or virtual terminals of degree more them one. Generation of FST is similar to [19] as the FST structures are same.

The Pruning algorithms works by growing an FST f to larger trees and testing whether these larger trees can exist in the optimal solution. Virtual terminals in OARSMT must have degree 2,3 or 4. The growth of the tree is by combining FST at virtual terminals.

Concatenation of FST is similar to [19]. An ILP formulation is developed where each FST and virtual terminal is associated with a binary variable to indicate whether it is selected as a part of OARSMT. Comparing with the previous works[18][19]for 200 rectilinear obstacles, Obsteiner can solve problems upto 2000 complex obstacles.

F**.** *Algorithms based on Maze Routing method*

The main problem with Maze routing based approaches is that it is believed that it can handle only small scale problems and lack of effective multiterminal variant to handle multipin nets. Also the time complexity and memory usage can grow very large as the routing ares expands.

Liang Li and Young proposed a heuristic on maze routing based approach which can handle large scale OARSMT problems and multipin nets. The efficiency of this routing engine is improved by use of heap data structure and by propagation of simplified Hanan[1] grid. A fast post processing step will be applied to scan the paths connected to each pin once to further minimize the total wire length. Handling of Multipin nets algorithm to choose the pin whose position is nearest to the boundary, propagate from one pin p1 to another p2 through shortest Manhattan distance. Multiple paths of this length will be recorded in Maze router. Propagate from all these paths to find third pin. This process is repeated until all pins in P are reached. After this process, steiner points lying on the paths are found and record Manhattan distance between pins and steiner points. Then MST algorithm is used to connect all points. Garbage collection step to remove dangling connection which are connected to steiner points only.

Comparing the results with those of the previous works and can show that this work can out-perform the best previous results on this problem [28] by giving an OARSMT with 2.01% less wire length on average and can make a 27.04% improvement in wire length in comparison with a lower bound of the optimal solution on average, while the running times are all very short and comparable to those in [28].

G. *Algorithms based on Steiner point selection*

Concept of steiner point locations

Hanan grid[1] and Escape graph[8][7][6] generate vertices on intersection of line segments expended from pin vertices or obstacle corners. But n segments will generate n*(n-1)/2 intersections which are too many for steiner point selection which lower the efficiency of steiner point selection. Hence a new steiner point location concept provides more efficient way to generate desirable Steiner point candidates.

Consider a 2-pin vertex on a plane and shortest path region(SPR) is the union of all the shortest paths between P and Q. For a 3-pin instance if the new pin vertex is inside SPR of original pin vertices, the new pin vertex is directly the best steiner point of 3 pin instance and no steiner point, else the best steiner point must be a boundary corner of one SPR among the 3-pin vertices.

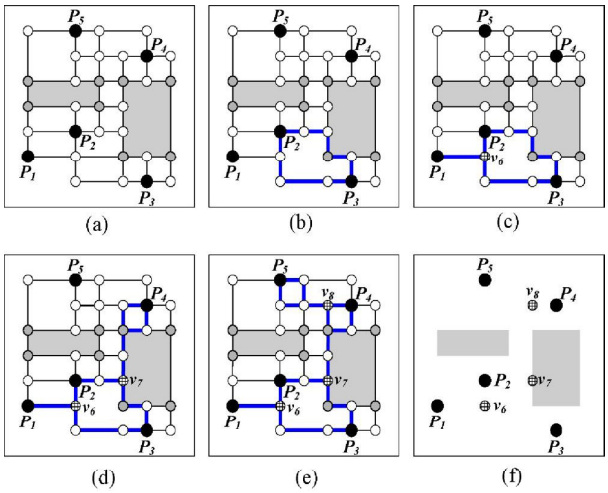


Fig. 7. Steiner point selection[23]

Steiner point selection, where the existing connected component is represented by bold segments. (a) OAVG.
(b) Boundary of SPR(*P*2, *P*3) isthe initial existing connected component. (c) SPR(*P*1, *v*6) is constructed and *v*6 is selected as a Steiner point. (d) SPR(*P*4, *v*7) is constructed and *v*7 is selected as a Steiner point. (e) SPR(*P*5, *v*8) is constructed and *v*8 is selected as a Steiner point. (f) Selected Steiner points are *v*6, *v*7, and *v*8.

Chih-Hung Liu et,al [22] proposed a steiner point based algorithm to achieve best practical performance in wirelength and runtime. More focus is on usage of steiner point as steiner points cause NP-completeness of OARSMT problem. A Obstacle Avoiding Vornoi graph(OAVG) is constructed which contain many desirable steiner point besides, pin vertices and obstacles corners but has only O(n) vertices and edges. Using prim's algorithm and SPR we effectively select good steiner points to minimize wirelength. Based on OAVG and selected steiner points, integrate existing schemes to construct OARST and reduce the redundant segments of OARST.
This algorithm achieves best solution quality in O(n log n) time. This can be extended to ML-OARSMT and OAPD-ST. Compared to Maze routing[21] this gives best solution quality and performs mush faster.
The previous work was continued by Chih- Hung Liu et.al[23] with a new concept of steiner point locations creating a linear space routing graph to resolve the bottleneck of existing heuristics. A steiner point based framework is given to get a solution for OARST.
OAVG construction-A routing graph called OAVG G(V,E) which contains satisfactory steiner points and uses only O(n) space is constructed. The first type of vertices are pin vertices and obstacle corners i,e PUC. Second type of vertices are obtained by projecting from each vertex in PUC to the boundary of its closest obstacle in each axis parallel direction. Third type of vertices are generated based on our concept of steiner point location.
Steiner point selection- To minimize the wirelength, the closest pin vertex from an existing connected component, we select the farthest pin-vertex as a starting vertex, which is a pin vertex whose distance to its nearest pin vertex is the largest among those between any other pin vertex and its pin vertex.
OARSMT construction- Specify all the pin vertices and all the selected steiner points as the terminals in OAVG and construct an MTST based on OAVG. After MTST construction, recursively remove non pin vertex leaves from the resulting tree and obtain OARSMT.
Refinement- A O(nlogn) time effective refinement proposed by Liu [24].
The average time complexity of this algorithm Is O(n log m log n) and worst case in O(mn log m) where m is the number of pin vertices and n is input size.

H. *LookUp Table based*

Gaurav Ajvani, Chrish Chu[26] proposed FOARS which applies Top down approach by partitioning the set of pins inti several subsets uncluttered by obstacles. Then an obstacle avoiding steiner tree is generated for each subset by an obstacle aware version of RSMT FLUTE. Finally the trees are merged and refined to form OARSMT.
It is found that the OASG generation algorithm in [14] has a few shortcomings.The nearest neighbor for any vertex in a quadrant is contingent upon the direction of scanning which means they have to scan along all four quadrants of a vertex in order to capture its connectivity information. In the absence of obstacles, algorithm cannot guarantee the presence of at least one minimum spanning tree in their spanning graph. Algorithm cannot handle abutting obstacles due to minor mistakes in theinequality conditions.
OASG generation algorithm in [30] which takes connectivity information between pins and obstacle corner vertices using a octant OASG generation algorithm.
OPMST is constructed for any corner vertex v, find the nearest neighbouring pin vertex u, we connect all the pin vertices originally connected with v to u and delete v. update their weights as their weight + weight of e(u,v) i,e edge would be penalized for the obstacles in its path. Partition the pin vertices based on OPMST and then pass the subproblems to OAFLUTE, which calls FLUTE recursively to construct an OAST. Every pin to pin connection avoiding obstacles is rectilinearized to generate an OARSMT

Sl. No	**Algorithm**	**Underlying Concept**	**Time Complexity**	**Advantages**	**Disadvantages**
1	Escape Graph based [6],[7],[8]	Ganley *et al* [6][7] introduced a strong connection graph called e*scape graph*, and proved that all Steiner points in the optimal solution can be found in this graph	O(nm)	It works well when terminals are less than 7 and obstacles are convex	the number of edges and vertices in escape graph is much larger than that in track graph. So, finding a solution in escape graph is time consuming
2	Track Graph	Wu *et al* [3] introduced *track graph*, a grid-like structure, which		An OARSMan takes about 0.1, 3, and	Takes longer time for large scale design.

	based[9]	consists of rectilinear tracks defined by the obstacles and the terminals		32 seconds respectively to construct a tree with 10, 50, and 100 terminals among over 9 rectangle obstacles.	
3	Spanning Graph based[11],[12],	a spanning graph is an undirected graph over the points that contains at least one minimal spanning tree. The number of edges in the graph is called the cardinality of the graph. Thealgorithm of Shen *et [11]al.* involves constructing a complete graph whose vertex set contains all the pins and corner vertices	$O(n^2 \log n)$	spanning graph is able to describe the relative geometrical relationship between vertices in the plane using O(n) edges. Moreover, by using an O(n log n) sweep line algorithm, the construction of a spanning graph can be done in O(n log n) time	Misses out on essential edges which gives better solution for OASG
4	[13],[14]	construct an OASG with "essential" edges and prove the existence of a rectilinear shortest path between any two pins, which is not guaranteed in the OASG constructed by Shen *et al.* [11]	O(n log n)	This approach has a more global view of both pins and obstacles. Consequently, this approach can often obtain a much better solution quality.	
5	[15]	generate $O(n)$ critical paths as essential solution components, and prove that those paths guarantee the existence of desirable solutions. The path-based framework neither generates invalid initial solutions nor constructs connected routing graphs, and thus provides a new way to deal with the OARSMT problem	O(n log n)	Gives 50.1 times speedup than [21] with wire length 1.1% longer on average.Cosidering the large number of obstacles the improve ment in runtime is very significant	
6	Full Steiner tree based approach [16]	It uses Geosteiner approach which is one of the best RSMT construction techniques for OARSMT, by adding 4 virtual terminals to each obstacle. It is prove d that FST are simple and can be constructed efficiently(uses ILP and Branch-cut technique)		Can handle hundreds of pins with multiple blockages, generating an optimal solution in a reasonable amount of time	This algorithm is affected by number of virtual terminals required.Most of the cases contain only 10 obstacles but running time is still very expensive
	[17]	Only two Virtual terminals are introduced for each obstacle and it adopts incremental way to handle obstacles.		The incremental way to handle obstacles effectively reduces the running time for large scale problems.	
	[19]	This is based on Geosteiner modified and enhanced to allow non-overlappng rectangular blockages in the routing region.		It generates optimal solution in reasonable time.	It can Handle only rectangular Obstacles and less than 100 obstacles. cannot be applied when there are complex rectilinear obstacles in the routing region

	[20]	To construct OARSMTs among rectilinear obstacles of both convex and concave shapes. the OARSMT problem with an empty list of obstacles is solved resulting in an RSMT. Then check for obstacles that overlap with the solution. For each FST used to build the current solution, decompose it into line segments. For each line segment, check whether it intersects with any obstacles. Among all overlapping obstacles, choose the dominating one. For example, for a vertical segment, choose an obstacle that has the largest width. All chosen obstacles are added to the obstacle list. A new iteration then starts again by solving the OARSMT problem with the obstacles in the renewed list		It is 31 times faster than [19] for small cases and can solve up to 2000 obstacles for rectangular obstacles	
	Maze Routing based[21]	To handle multi-pin nets, multiple paths between the pins are kept until all the pins are reached, then a MST based method is used to select between those paths to create an OARSMT. A post-processing step is then performed to further reduce the total wire length		Show maze routing based approaches can also handle large scale OARSMT problems effectively, with best solution qualitywith good wire length	Performs much slower
	Based on steiner point selection[22]	The Steiner-point based framework focuses more on the usage of Steiner points instead of the handling of obstacles as the usage of Steiner points causes the NP-completeness ofthe OARSMT problem. To integrate the effectiveness and the efficiency of routing graphs, a new concept of Steiner point locations to efficiently generate desirable Steiner point candidates	$\Theta(n^{1.17})$	an OAVG, which contains desirable Steiner points and has both only $O(n)$ vertices and edges. All these help the algorithm to achieve the best practical performance in both wirelength and run time,	
	[23]	Uses the concept of steiner point selection with preferred direction	$O(mn \log n)$.	Achieves excellent solution quality and speed performance	
	Flute Based [26][27]	FOARS applies a top-down approach which first partitions the set of pins into several subsets uncluttered by obstacles. Then an obstacle-avoiding Steiner tree is generated for each subset by an obstacle aware version of the rectilinear Steiner minimal tree algorithm FLUTE. Finally, the trees are merged and refined to form the OARSMT	$O(n \log n)$	an efficient algorithm to construct OARSMT and RSMT based on extremely fast and high-quality Steiner tree generation tool called FLUTE. proposed a novel OASG algorithm with a linear number of edges, also an obstacle aware version of FLUTE, which generates OAST	

Table-1 Comparison of different OARST Algorithms

IV.CONCLUSION

This paper is a brief study of the existing algorithms for computing the Obstacle Avoiding Rectilinear Steiner Minimum Trees. Many heuristics have been proposed for the OARSMT problem. In order to deal with rectilinear obstacles, some heuristic approaches try to cut a rectilinear obstacle into several rectangular obstacles. But, the cutting lines on the rectilinear obstacles allow wires to go through, which may result in an infeasible solution. OARSMT construction based on the critical paths, which guarantees the existence of desirable solutions. A fast lookup table-based algorithm to route a multiterminal net in the presence of rectilinear obstacles. The algorithm uses the obstacle-avoiding spanning graph to guide the partitioning of the initial solution and constructs the final OARSMT based on an obstacle-aware version of fast lookup table. A Steiner-point-based framework to construct OARSMTs and showed that their algorithm can achieve best practical performance among existing heuristics.

In comparison with the heuristics, few exact algorithms have been proposed. The maze-routing approach constructs OARSMTs based on the extended Hanan grid, can handle complex obstacles, give an optimal solution to two-terminal nets but is computationally expensive for large scale designs. Extended GeoSteiner to an obstacle-aware version. Their algorithms are able to generate optimal OARSMTs for multiterminal nets in the presence of rectangular obstacles.

A strong connection graph called an escape graph for the OARSMT problem and proved that there is an optimal solution composed only of escape segments in the graph. Based on the escape graph optimal three-terminal and four-terminal OARSMTs can be constructed.

REFERENCES

[1] M. HANAN, *On Steiner's Problem With Rectilinear Distance,* SIAM J. Applied Math., **14 (1966),** pp. **255- 265.**

[2] M. GAREY AND D. *S.* JOHNSON, *The Rectilinear Steiner Problem is NP-complete,* SIAM *J .* Applied Math., **32 (1977),** pp. **826-834.**

[3] F. K. HWANG, *On Steiner Minimal Trees with Rectilinear Distance,* SIAM J. Applied Math., **30 (1976),** pp. **104-114.**

[4] D. W. Hightower. A solution to the line-routing problem on the continuous plane. In *Proceedings **of** the Sixth Design Automation Workshop,* pages 1-24, 1969.

[5] J. S. B. Mitchell, "An optimal algorithm for shortest rectilinear paths among obstacles," in *Proc. Abstracts 1st CCCG,* 1989, p. 22.

[6] J. L. Ganley and J. P. Cohoon, "Routing a multiterminal critical net: Steiner tree construction in the presence of obstacle," in *Proc. ISCAS,* 1994, pp. 113–116.

[7] Y.Yang, Q.Zhu, T.Jing, and X.L.Hong, "Rectilinear Steiner Minimal Tree among Obstacles, In:Proc. of IEEE ASICON, Beijing, China, 2003, pp.348-351.

[8] Y. Hu, T. Jing, X. Hong, Z. Feng, X. Hu, and G. Yan,"An-OARSMan: Obstacle-Avoiding Routing Tree Construction with Good Length Performance," in Proc. ofASP-DAC, 2005.

[9] Yung-Tai Chang*, Ya-Wen Tsai*, Jun-Cheng Chi** and Mely Chen Chi* "Obstacle-Avoiding Rectilinear Steiner Minimal Tree Construction"

[10] Z. Shen, C. Chu, and Y. Li, "Efficient rectilinear Steiner tree construction with rectilinear blockages," in *Proc. ICCD,* 2005, pp. 38–44.

[11] A Fast and Stable Algorithm for Obstacle-Avoiding Rectilinear SteinerMinimal Tree Construction, " in Proc. of ASP-DAC, pp.262-267,2007.

[12] J. Long, H. Zhou, and S. Memik, "EBOARST: An efficient edge-based obstacle-avoiding rectilinear Steiner tree construction algorithm," *IEEE Trans. Comput.-Aided Des.,* vol. 27, no. 12, pp. 2169–2182, Dec. 2008.

[13] C.-W. Lin, S.-Y. Chen, C.-F. Li, Y.-W. Chang, and C.-L. Yang, "Obstacle-avoiding rectilinear Steiner tree construction based on spanning graphs," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.,* vol. 27, no. 4, pp. 643–653, Apr. 2008.

[14] C.-H. Liu, S.-Y. Yuan, S.-Y. Kuo, and Y.-H. Chou, "An $O(n \log n)$ path based obstacle-avoiding algorithm for rectilinear Steiner tree construction," in *Proc. DAC,* 2009, pp. 314–319.

[15] L. Li, Z. Qian, and E. F.-Y. Young, "Generation of optimal obstacle avoiding rectilinear Steiner minimum tree," in *Proc. ICCAD,* 2009, pp. 21–25.

[16] T. Huang and E. F.-Y. Young, "Obstacle-avoiding rectilinear Steiner minimum tree construction: An optimal approach," in *Proc. ICCAD,* 2010, pp. 610–613.

[17] T. Huang and E. F.-Y. Young, "An exact algorithm for the construction of rectilinear Steiner minimum trees among complex obstacles," in *Proc. DAC,* 2011, pp. 164–169.

[18] Chih-Hung Liu, Sy-Yen Kuo, *Fellow, IEEE,* D. T. Lee, *Fellow, IEEE,* Chun-Syun Lin, Jung-Hung Weng, and Shih-Yi Yuan " Obstacle-Avoiding Rectilinear Steiner Tree Construction: A Steiner-Point-Based Algorithm" IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, VOL. 31, NO. 7, JULY 2012

[19] Tao Huang and Evangeline F. Y. Young "ObSteiner: An Exact Algorithm for the Construction of Rectilinear Steiner Minimum Trees in the Presence of Complex Rectilinear Obstacles" IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, VOL. 32, NO. 6, JUNE 2013

[20] L. Li and E. F.-Y. Young, "Obstacle-avoiding rectilinear Steiner tree construction," in *Proc. ICCAD,* 2008, pp. 133–138

[21] C.-H. Liu, S.-Y. Yuan, S.-Y. Kuo, and J.-H. Weng, "Obstacle-avoiding rectilinear Steiner tree construction based on Steiner point selection," in *Proc. ICCAD,* 2009, pp. 26–32.

[22] C. H. Liu, S. Y. Kuo, D. T. Lee, C. S. Lin, J. H. Weng, and S. Y. Yuan, "Obstacle-avoiding rectilinear Steiner tree construction: A Steiner-pointbased algorithm," *IEEE Trans. Comput.-Aided Des.,* vol. 31, no. 7, pp.

[23] 1050–1060, Jul. 2012.
[24] C. H. Liu, S. Y. Yuan, S. Y. Kuo, and Y. H. Chou, "An *O*(*n* log *n*) path-based obstacle-avoiding algorithm for rectilinear Steiner tree construction,"in *Proc. Des. Automat. Conf.*, 2009, pp. 314–319
[25] J. Long, H. Zhou, and S. O. Memik, "An *O*(*n* log *n*) edge-based algorithm for obstacle-avoiding rectilinear Steiner tree construction," in *Proc. Int. Symp. Phys. Des.*, 2008, pp. 126–133.
[26] G. Ajwani, C. Chu, and W.-K. Mak, "FOARS: FLUTE based obstacleavoiding rectilinear Steiner tree construction," *IEEE Trans. Comput.-Aided Des.*, vol. 30, no. 2, pp. 194–203, Feb. 2011.
[27] C.W.Lin,S.Y.Chen,C.F.Li,Y.W.Chang,C.L.Yang,"Efficient obstacle-avoiding rectilinear Steiner tree construction", in:Proceedings of 1ISPD,2007.
[28] H. Zhou, N. V. Shenoy, and W. Nicholls, "Efficient minimum spanning tree construction without Delaunay triangulation," in *Proc. ASP-DAC*,2001, pp. 192–197.
[29] Y.F.Wu, P.Widmayer, M.D.F.Schlag, and C.K.Wong. "Rectilinear Shortest Paths and Minimum Spanning Trees in the Presence of Rectilinear Obstacles", IEEE Trans on Computers, 1987, 36(3): pp. 321-331.
[30] Yu Hu, Zhe Feng, Tong Jing, Xianlong Hong, Yang Yang Xiaodong Hu, Guiying Yan "FORSTer: An Efficient Heuristic for Obstacle-Avoiding Rectilinear Steiner Tree Construction"
[31] S.Das, S.V.Gosavi, W.H.Hsu, and S.A.Vaze, "An Ant Colony Approach for the Steiner Tree Problem", In Proc. of Genetic and Evolutionary Computing Conference, New York City, New York, 2002.