NETLIST DECOMPOSITION AND CANDIDATE GENERATION FOR ANALOG IC ROUTING

A Thesis

by

VAISHNAVI VENKATESH

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Jiang Hu
Committee Members,	Jose Silva-Martinez
	Hank Walker
Head of Department,	Miroslav M. Begovic

August 2019

Major Subject: Computer Engineering

ABSTRACT

Netlist decomposition and candidate generation is a non-conventional approach in the routing stage of the place and route (PnR) flow. While there has been significant research and advancement in the digital domain for automation with respect to this as well as other techniques, very little work has been done in the analog domain due to its complex constraints and specific requirements. With this proposed method, the most common requirements of Analog circuits are taken into consideration to provide candidate routes for netlists of analog Integrated Chips (IC).

Netlist decomposition is an important stage of breaking down multi-pin nets into two-pin nets by adding additional nodes for each net. The proposed method takes into account blockages and constraints such as symmetry and bends to develop a new algorithm using Steiner trees and Hanan grids to generate optimal Steiner points. This method also breaks down multi-pin nets to 3-pin nets which reduces the wirelength and computations significantly. The decomposed net segments are run through Dijkstra algorithm to generate multiple candidates and an Integer Linear programming (ILP) solver is used to pick the best candidates that follow all the constraints and design rules.

The experimental results show that overall wirelength is reduced by 5.16% while using 3-pin net decomposition when compared to 2-pin net decomposition. There is also a reduction in the number of metal layers used and the number of Steiner points generated. The method shows lesser computations when compared to other decomposition techniques as it avoids multiple reroutes to obtain Design Rule Check (DRC) clean routes.

DEDICATION

To my parents.

ACKNOWLEDGMENTS

My journey in pursuing my Master's degree at Texas A&M University has been exciting and life changing. I am very grateful for the various opportunities and experiences throughout the course of this degree. I would like to express my gratitude towards my advisor, Dr. Jiang Hu, whom I worked under for the last year and a half. He provided me with great research opportunities and patiently guided me throughout. I received great support and mentoring from him as he challenged me to try and learn new things. I am forever indebted to him for the help and suggestions that he gave me and for trusting me through the good and difficult times from the day I joined his research group. I would also like to thank Dr. Hank Walker and Dr. Jose Silva-Martinez who are on my committee and gave me useful insights throughout my thesis.

I would like to thank my fellow teammates Jinhyun So, Wenbin Xu and Yaguang Li with whom I worked closely throughout and learnt a lot from their respective areas of expertise. I would also like to thank the teammates from University of Minnesota and from Intel who also worked with me on this project and providing helpful feedback on my work. I would also like to thank former student Chia-Yu Wu, who provided valuable inputs based on his thesis work while I worked on my research.

I would like to thank my mother whose life and career inspired me to explore the field of research and has been my unwavering support. I also thank my father who has been my pillar of support throughout and encouraged me to challenge myself everyday. I thank my friends in College Station Sanjana, Megha and Pranitha who are my family here and have been by my side through everything. I also thank my dearest friends Nupur, Ramya, Mahathy and Siddharth who have been my greatest emotional support through this journey.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Professor Jiang Hu and Professor Jose Silva-Martinez of the Department of Electrical and Computer Engineering and Professor Hank Walker of the Department of Computer Science and Engineering.

The work related to placement was carried out by Wenbin Xu and Yaguang Li. The work on Candidate selection and ILP solver was done by Jinhyun So,

All other work conducted for the thesis was completed by the student independently.

Funding Sources

Graduate study was supported by a scholarship from the Department of Electrical and Computer Engineering at Texas A&M University.

NOMENCLATURE

PnR	Place and Route
IC	Integrated Chips
ILP	Integer Linear Programming
DRC	Design Rule Check
LVS	Layout vs Schematic
RST	Rectilinear Steiner Tree
RSMT	Rectilinear Steiner Minimum Tree
MST	Minimum Spanning Tree
LUT	Look up Table
FLUTE	Fast Look up Table Estimation
VLSI	Very Large Scale Integration
EDA	Electronic Design Automation
CAD	Computer Aided Design
PDK	Physical Design Kit
CN	Critical Nets
WM	Wirelength Matching
MS	Mirror Symmetry
BM	Bend Matching
OM	Orientation Matching
TM	Topology Matching

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Analog circuit design has been a challenging field over years. Due to the specific requirements and uniqueness of each type of analog circuit, it is very difficult to automate the PnR flow for them. There are many challenges to be taken into account such as multi-pin connections, matching with respect to symmetry, bends, wirelengths as well as parasitics and timing. This especially gets complex when the blocks are of varied sizes and the wires have multiple destinations. This leads to the necessity for a powerful algorithm to decompose the nets and generate candidates.

Unlike the digital domain, there is no straightforward objectives to be met such as the meeting timing constraints and minimizing wirelength. These are not the top priorities for analog circuits. The routing constraints have greater importance. Another big difference is that analog circuits tend to have very few blocks that need to be placed and routed whereas digital circuits have thousands of blocks. This implies that data size for analog circuits are much lesser and hence the run-time of the tool is not very crucial.

Focusing more on the routing phase of the PnR flow, it can be categorized into the following stages as shown in Figure 1.1.

- Placed blocks, net connection and constraints are usually the main inputs required for routing.
- Grids are generated based on the design rules of metal and via spacing for different metal layers to obtain tracks on which the nets can be routed.
- The nets are then decomposed from multiple pins to 2-pin "segments" which has a single source and destination that can be used for shortest path algorithms.
- Candidates are generated for each segment which gives multiple viable route options for each segment based on shortest path algorithms and constraints.

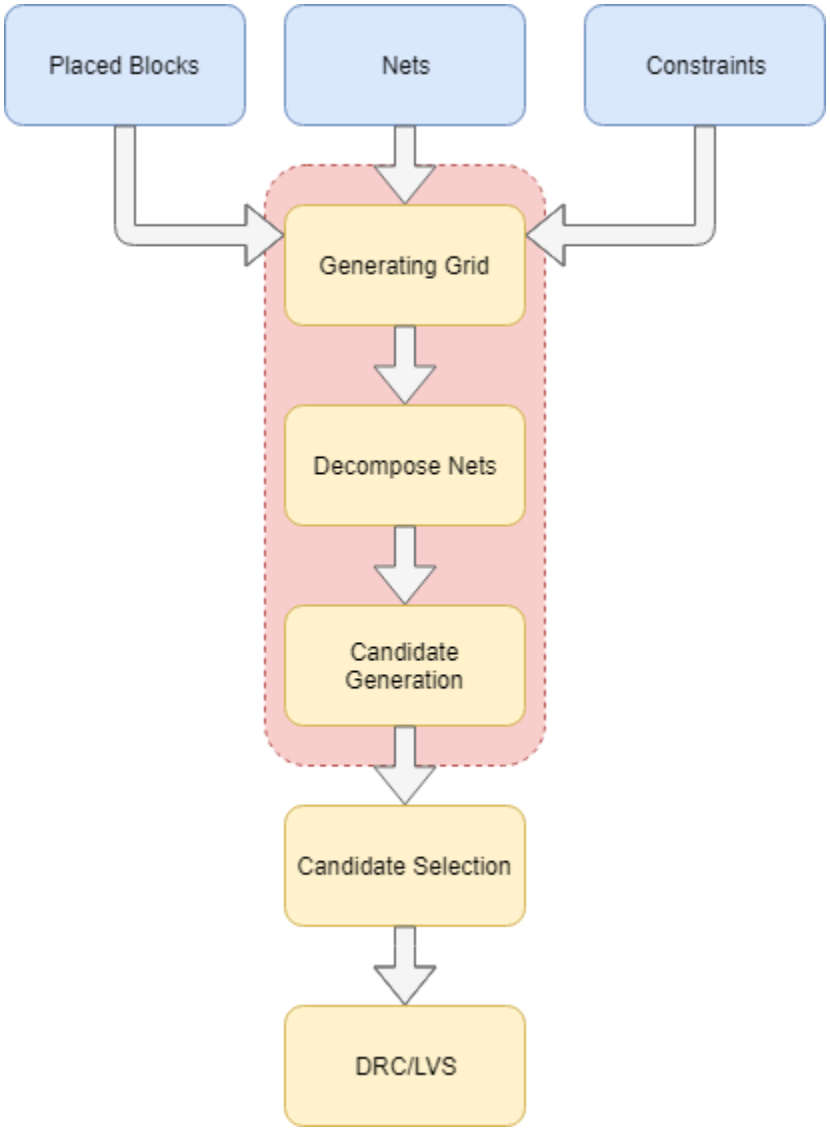

Figure 1.1: Design flow for routing.

- Metal and Via annotation is performed for each candidate and the optimal candidate is selected.
- The final routing is the sent to Design Rule Check (DRC) and Layout vs Schematic (LVS) to verify if the results are satisfied.

The main focus in this design flow is done on the grid generation, netlist decomposition and candidate generation stages. The innovation in this work that is different from other work that

have used various different PnR flow to find an optimal solution for routing nets lies in the decomposition phase. The improvements in this work is a new form of netlist decomposition that decomposes large degree netlists to 3-pin nets while being mindful of the design constraints to provide a simplified solution for candidate generation.

1.1 Grid Generation

This is the first stage of the routing flow. Here, taking into consideration all the design rules of metal and via spacing for each layer, routing pitch is calculated. Based on the estimated die size, grid vertices are calculated for each layer and edges are added. Also, based on the internal metal information obtained from the placed block metal overlaps are checked and certain edges are prohibited from routing. From the values of grid pitches for each metal layer a minimal Hanan grid pitch is calculated which is required if additional Hanan points needed to avoid blockages. Hanan grid is defined as a grid that contains coordinates of all combinations of x and y coordinates of the pins that need to be connected. This is a necessary feature for netlist decomposition.

1.2 Netlist Decomposition

Netlist decomposition has been used in VLSI routing over the last three decades using a very popular NP- complete problem called Steiner trees. Unlike other minimum spanning tress problems that connect all nodes using the shortest path possible, Steiner trees create additional nodes in the given region called Steiner nodes or Steiner points that is added to the original nodes in the tree such that the overall length is minimized.

This is the key idea required for converting the multi-pin connections into 2-pin connections. Once each segment of the net contains a source and a destination coordinate, it is simplified into a simple shortest path problem. The grids created in the previous stage act as the nodes and edges of the graph G=(V,E). In this case, the rectilinear distance between the vertices act as the weights to the E = (u,v).

The proposed method decomposes multi-pin nets into 3-pin nets groups which simplifies the constraints and the complexity of candidate generation. It also reduces the number of segments that

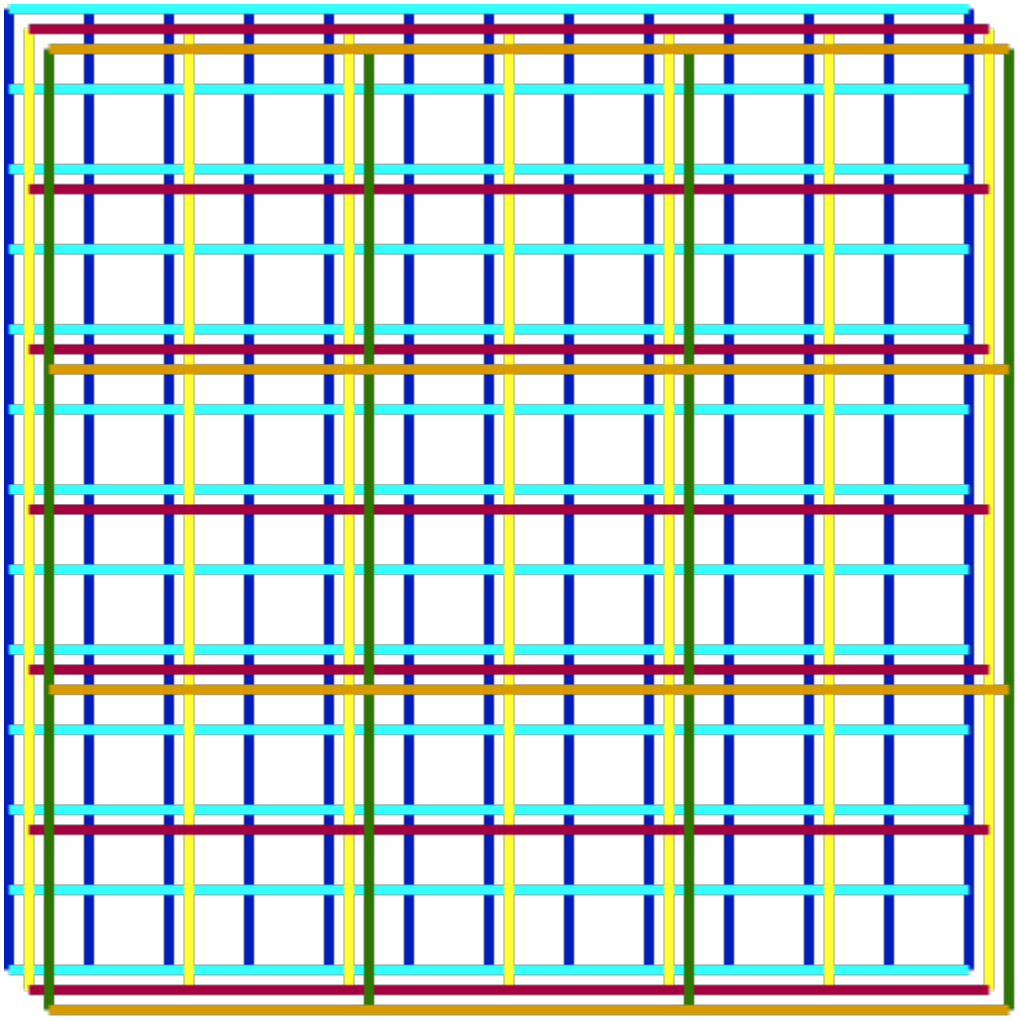

Figure 1.2: Grid Generation.

need to be re routed. This form of decomposition is also capable of blockage avoidance and hence the final routes of certain parts of the net are already estimated. This helps in tackling complex constraints such as symmetry and bend matching by breaking it down into simpler constraints in this stage and then in the candidate generation stage.

1.3 Candidate Generation

This is a novel methodology in determining the routes of nets. Instead of the standardized method of using global and detailed router, candidate generation and selection is used. Usually in most commercial tools, global router estimates the arbitrary path for the nets by only taking into

consideration the placement of the blocks and pins. Then this is followed by the detailed router which takes all the physical information such as metal layers, widths and parasitics to make actual connections in the layout.

On the other hand, for candidate generation, all the metal and physical information is taken into consideration along with the location of the decomposed net segments and the constraints available for each net. Based on this information, a number of viable candidates for each net is generated. Each of these candidates satisfy all the constraints and can be used as a final route.

Followed by this is candidate selection where the candidate information is fed into an Integer Linear Programming (ILP) solver such that it picks the best of each candidate and there is no overlap or mismatch between the nets. This methodology proves to be faster and efficient than using global and detailed router simply because the application is for analog circuits. As opposed to digital circuits, the main objective is not to minimize only wirelength but also to satisfy the complex constraints. If this was implemented using global router there would be a lot of incorrect estimations and the detailed router will have to iteratively correct each route such that the constraints are satisfied. The trade off is that this method would be more complex than the general approach but due to analog circuits containing significantly lesser blocks and nets, the complexity is manageable.

1.4 Our Contribution

The main motivation of this thesis is to develop an algorithm that decomposes netlists suitable for analog circuit routing which is inclusive of the matching constraints to get suitable candidates for each net. To achieve this, the problem can be broken down into the following:

- To implement the Rectilinear Steiner Minimum tree using the iterative 1-Steiner algorithm which generates the RSMT to minimize the wirelength with the help of Hanan grids to reduce the decomposition from multi-pin nets to three-pin nets.

- To develop a method for Steiner tree generation that is aware of blockages and can be avoided using Hanan grids with the help of penalty in the cost function.

- To create routing grids for various metal layers that honors the design rules of the PDK, the matching and symmetry constraints and implement the Dijkstra algorithm.
- To obtain candidates for each net that can be used by an ILP solver to get final routes for the layout.

The rest of the thesis is organized according to this. Section 2 is a background study of previous work and other methodologies in this work. Section 3 is a detailed explanation of the Netlist Decomposition stage of the PnR flow. Section 4 focuses on the details of Candidate Generation and Selection. Section 5 goes into the details of our methodology of 3-pin netlist decomposition and the new simplified PnR flow. Section 6 shows the experimental results and the performance of this methodology. Section 7 gives the conclusion of the thesis and presents some ideas for future work on this topic.

2. PREVIOUS WORK

Netlist decomposition is an important step in routing flow which is used to simplify multi-pin nets into simpler nets. These nets that are generated from decomposition are fed into any shortest path algorithm for achieving shortest distance between each set of pins[1]. Unlike digital routing, analog routing has many objectives apart from finding the shortest distance. Constraints such as length matching, symmetry, common centroid as well as topology are some of the common constraints that are seen in analog layouts[2][3]. The most efficient methodology for netlist decomposition is using Steiner trees. Steiner trees can be extended for this application by trying to avoid blockages[4]. When a Steiner node that is generated by the Steiner algorithm is on a blockage such as active regions or internal metals, using a penalty or slack the node is moved outside the blockage. The node on the Hanan grid with the least penalty is the chosen as the new Steiner node. This re-adjusts the Steiner tree generated for decomposition.

The first paper solely devoted to RSMT problem was written by Hanan [5] in 1966. In addition to characterizing optimal solutions for small instances of the problem, Hanan gave the fundamental structural definition. Draw horizontal and vertical lines through every terminal. The obtained grid is called the Hanan grid. In rectilinear Steiner minimum tree with obstacles (RSMTO) problem[6], the Hanan grid is modified as the extended Hanan grid. The proposition of extended Hanan grid transforms the routing problem into a graph problem, and the weighted Hanan grid transforms the computing scale from routing area into the input size of terminals and obstacles.

Extending the concept of Hanan grids[5], the large netlists are decomposed to three-pin netlists instead two-pin netlists. This greatly reduces the number of segments that the net is broken down to which directly impacts the run time. The implementation of Steiner trees which uses buffer insertion for blockages is proposed in [4]. This uses a Steiner-tree heuristic generation and uses maze routing methodology. With an iterative approach a new subpath is created for the generated Steiner tree and is then reconnecting using maze routing. A grid graph is created for sparsification which impacts the computational efficiency of the algorithm. Arora [7] found a polynomial time

approximation scheme (PTAS). The PTAS is mainly based on dynamic programming, and provides a balance between the computation time and performance ratio. Another PTAS was proposed [8], but it still cannot satisfy the demand of integrated circuit production. So people turn to seek efficient heuristic algorithms to get the sub-optimal solutions. Many algorithms have been proposed focusing on RSMTO problem [9], but there is little consideration of boundary. The boundary cannot be regarded as an obstacle, because the wires cannot pass through the boundary on both sides. It brings new hardness to the problem, but the maze routing method such as Lee algorithm could overcome this difficulty. Lee [10] presented maze algorithm to route two-terminal nets optimally. Some improvements on Lee algorithm proposed later. Lee algorithm can be applied not only on the grid of rectilinear plane (i.e., there are no more than 4 neighbors for each grid), but also can deal with a graph in the general sense of graph theory.

The iterative approach of Rectilinear Steiner Minimum Tree (RSMT) generation from Minimum Spanning Tree (MST) is proposed in [11][12][13]. It optimally finds out each Steiner point that needs to be added to the tree to arrive at the right RSMT. This algorithm also limits the number of Steiner points that are added to a value k which is arbitrarily calculated in the cost function. The Hanan grid created for each point in every iteration is used to find the optimal set of points keeping in mind the cost of adding vias and worst-case scenarios. Thus, several fast algorithms have been proposed in the literature to construct an RSMT for a given set of pin locations [14], [9], [10]. However, the RSMT construction problem is NP-hard[8], so several papers also proposed Rectilinear Spanning Tree or RST construction algorithms for practical use.

An ILP based analog circuit routing [15] proposes sequential routing with the help of integer linear programming (ILP) solver. Here, a number of possible candidates are generated using the A* search algorithm for each net in the analog circuit. The candidate decomposition is then performed using FLUTE[16] to obtain two-pin nets. The available candidates are the fed into an ILP solver with constraints and the objective function is minimized. This work also features bend matching and weighted grids.

A highly scalable Steiner tree problem is proposed in[17]. This work uses a greedy triple

contraction algorithm or the Zelikovsky algorithm[18]. It has also been implemented as the Fast-SteinerUM software available at the VLSI GSRC bookshelf [19][20]. It solves the Steiner tree problem for 100 terminals. The algorithm is an extension of [11] which uses a set size of O(n log n) in pre-processing. It makes the process cheaper by decomposing the nets into 3-restricted Steiner tree instead of 2-restricted Steiner tree. One of the applications heavily using RSMT construction is global routing in which RSMTs are used for routing topologies. For example, BoxRouter [21], DpRouter [22], Archer[23], MaizeRouter[24], FastRoute [25], GRIP [26], and NTHU-Route [27] use FLUTE for routing topology generation. However, FLUTE constructs only one RSMT for a net. [28] prposes an efficient algorithm to construct all RSMTs on the Hanan grid for given pin locations. The algorithm builds a database (called ARSMT DB) of all potentially optimal Steiner trees (POST) on the Hanan grid for each potentially optimal wirelength vector (POWV) so that applications can quickly obtain all RSMTs from the ARSMT DB.

3. NETLIST DECOMPOSITION

Decomposing nets into smaller segments is an important process in determining the routes of nets in the layout. There are multiple approaches to achieving this but the most common approach is by constructing Steiner trees.

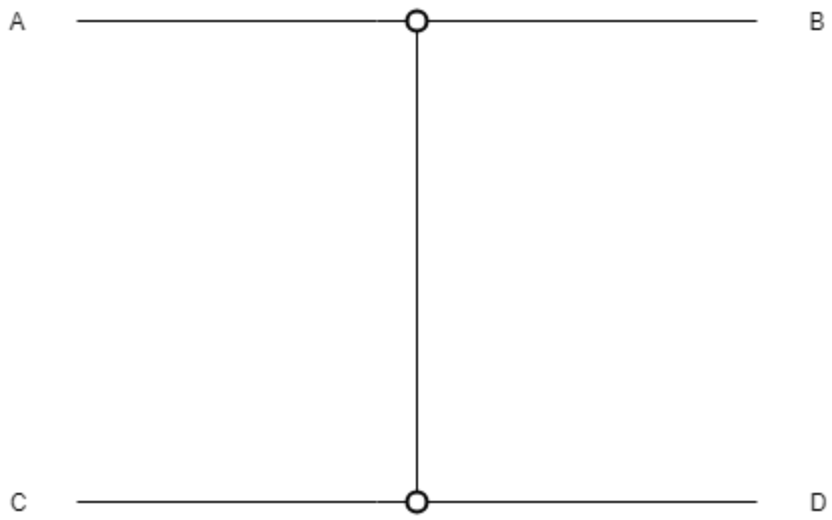

Figure 3.1: Example of Steiner Node.

3.1 Background

Steiner Tree problems in graphs are NP- complete in nature. As they are usually simplified into smaller variants, they can be estimated and solved in polynomial time using simple modifications such as look up tables (LUT) and so on. For the case of VLSI routing, the class of Steiner trees that are used are Rectilinear Steiner Trees (RST) which uses the rectilinear geometric space rather than euclidian. Rectilinear or Manhattan distances calculated are minimized to obtain Steiner points and can be defined as Minimum Rectilinear Steiner Trees or (RSMT) as shown in an example in Figure 3.1. The scope for obtaining Steiner trees are limited to the region of the Hanan Grids.

3.2 Hanan Grid

Hanan Grid is defined as H(n) which is a finite set of points obtained by drawing horizontal and vertical lines across each vertex n. Therefore, the Hanan Grid always contains n^2 points as shown in Figure 3.2. This proves to be the key region of interest for each net where the routing occurs. The Hanan grid created must also match the routing grids that are created which have the routing tracks. Also, the Hanan Grid is modified and flexible to accommodate blockages that require rerouting.

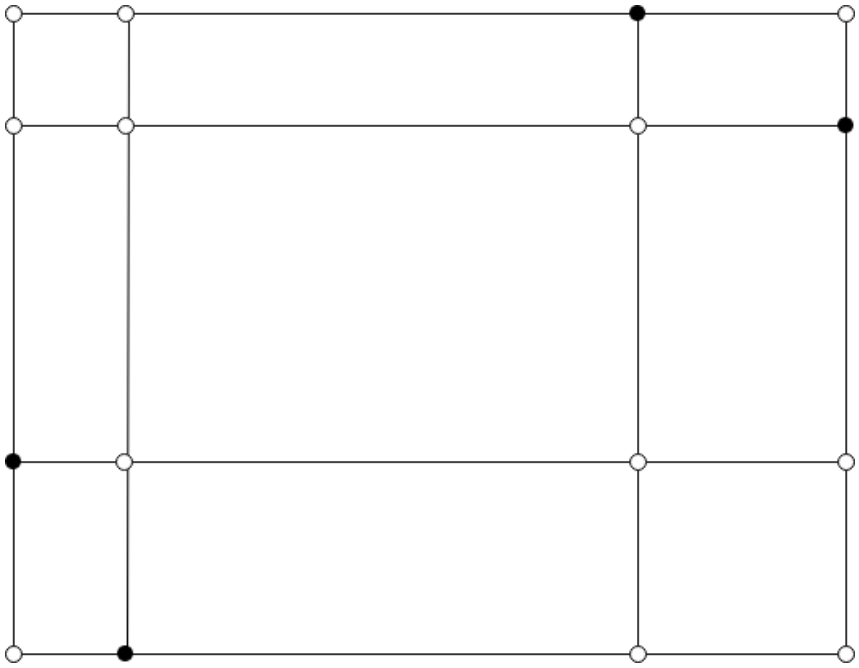

Figure 3.2: Hanan Grid for n=4. The black circles represent the location of pins on the Hanan grid and the remaining white circles are the possible locations for a Steiner node to be generated.

3.3 Construction of Steiner tree

For the construction of Steiner tree, the points are grown from an initial Minimum Spanning tree (MST). This is done using the iterative 1-Steiner tree approach. A Steiner tree is defined as an MST on the union of the original pins of the net in a set P and a set of Steiner points S[1].

This approach is to iteratively calculate optimum 1-Steiner points and include them into S. Let c be the cost of the MST which is the length in this case. The length of the MST over $P \cup S$ will decrease with each additional point, and we terminate the construction if there is no x such that $c(MST(P \cup S \cup \{x\})) < c(MST(P \cup S))$. The figure illustrates the execution of iterative 1-Steiner on a four-point example.

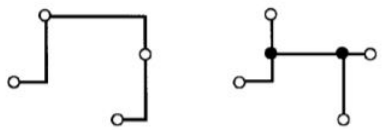

Figure 3.3: Steiner Tree Construction.

By the result of Hanan, we can find a 1-Steiner point by constructing a new MST on n + 1 points for each element in the Steiner candidate set, then picking the candidate which results in the shortest MST. Each MST computation can be performed in $O(nlogn)$ time[15], yielding an $O(n^3logn)$ time bound. This is the time required to find just one 1-Steiner point, and that the Steiner tree may contain up to n - 2 Steiner points [7]. As it turns out, a new 1-Steiner point may be added in $O(n^2)$ time. A linear number of Steiner points can thus be found efficiently with a total of $O(n^3)$ effort, and finding heuristic solutions with k Steiner points, in a region with high via costs, can be accomplished in $O(kn^2)$ time. There are also 4 rules that govern the selection of each Steiner point iteratively. They are:

- A point p cannot have two neighbors in the MST which lie in the same octant of the plane with respect to p. Thus eight "orientations" at 45°intervals can be fixed, each of which induces a Voronoi-like partition (the oriented Dirichlet cells) of the plane.

- These eight partitions can be overlaid into a "coarsest common partition" within $O(n^2)$ time. The resulting $O(n^2)$ regions of this partition are isodendral and introducing any point from within a given region will result in a constant MST topology.

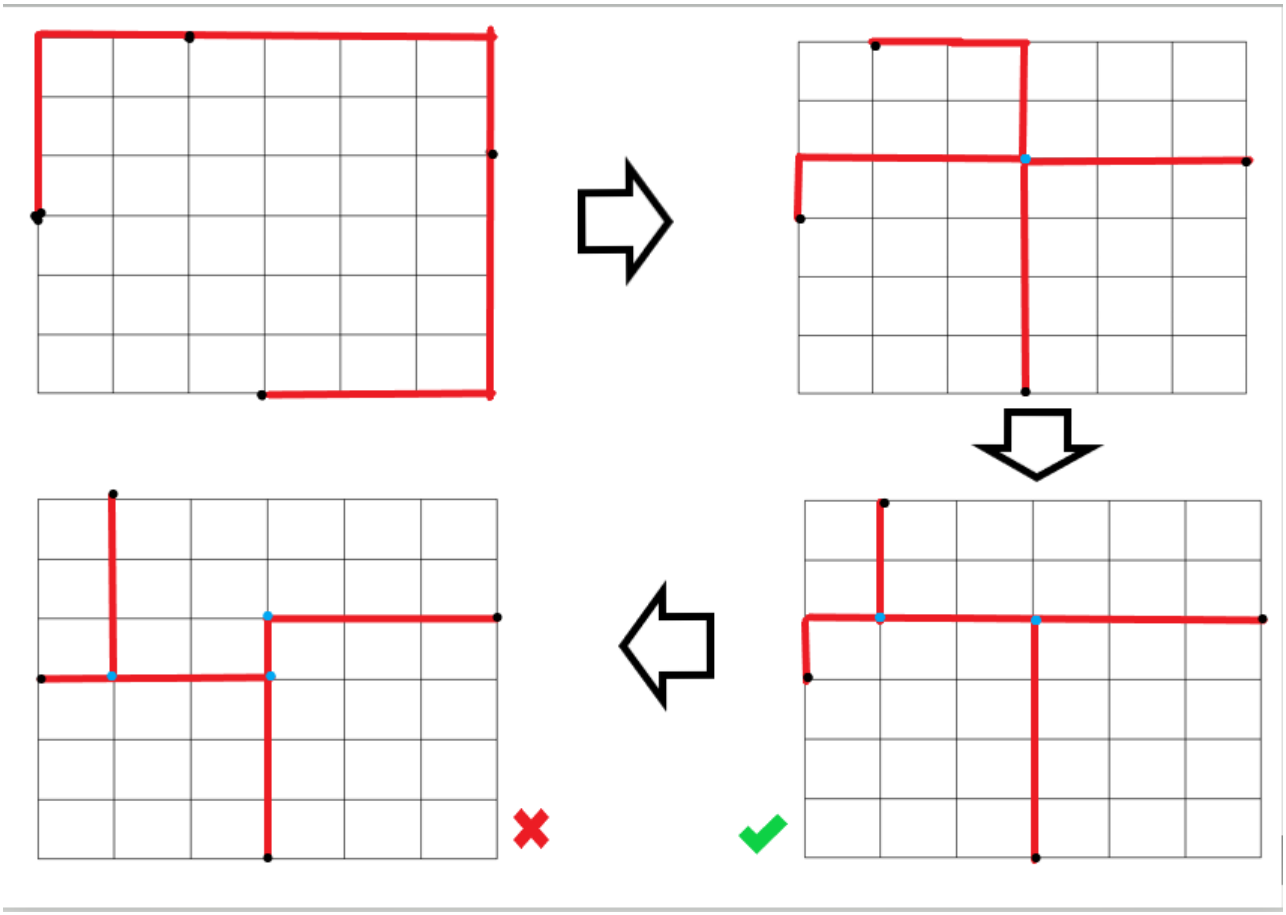

Figure 3.4: Iterative 1-Steiner method.

- The minimum spanning tree on the n points is constructed, and preprocessing is performed in $O(n^2)$ time such that whenever a new point is added to the point set, updating the MST to include the new point requires constant time.

- Iterate through the $O(n^2)$ regions of the overlaid partitions and determine, in constant time per region, the optimal Steiner point in each region. Each such point will induce an MST on n + 1 points that can be computed in constant time using the information obtained from the preprocessing. Comparing the costs of these trees and selecting the smallest one will give the minimum-length MST on n + I points. The total time for all phases is $O(n^2)$.

The performance ratio of $c(MST)/c(RSMT) \leq 3/2$. Now, another key feature to take into consideration is the active regions and internal metals due to block placement which leads to block-

ages in the area of the Hanan Grid. This is another issue that must be tackled in the stage of netlist decomposition.

3.4 Blockage Avoidance

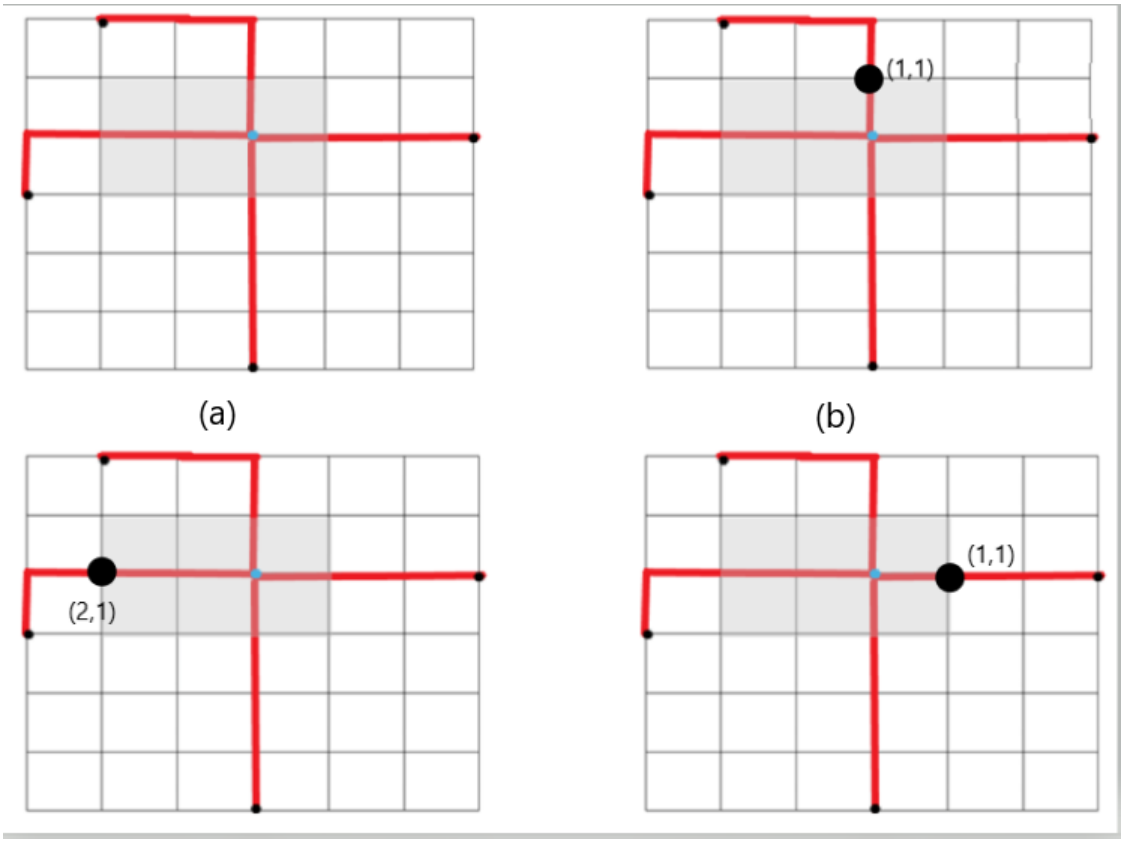

Figure 3.5: Blockage Avoidance.

When the Steiner point is formed in a forbidden region which can be due to active regions or internal pins and metals in layout, the Steiner tree is readjusted by accounting for the blockages and adding a penalty to the cost function. These algorithms are implemented on routing grids which are not uniform but take into account the design rules of the Process Design Kit (PDK) used for the layout. They consider the minimum width and length for each metal layer along with the worst case spacing between two metals side to side and end to end to create the grid pitch in the x and y-axes.

Thus, the Hanan grids are mapped onto the routing grids and if any edge or vertex is blocked due to a certain blockage, then the vertices and edges surrounding the blockage is activated and is translated to the Hanan grid to get a modified Hanan. Now, this is a new space that is input for the Iterative 1-Steiner to find new points and move the original Steiner.

4. CANDIDATE GENERATION

Candidate generation is a novel approach to routing in the PnR flow when compared to the standard global and detailed router. After the stage of candidate decomposition, the Steiner points are obtained. Now, this goes through various steps to find the final routing for each net as shown in Figure 4.1.

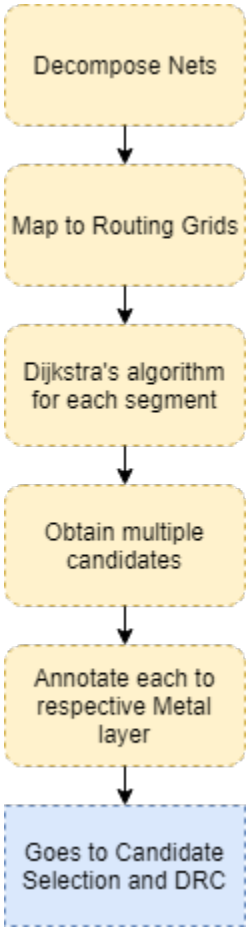

Figure 4.1: Candidate Generation.

4.1 Mapping Pins, Terminals and Steiner points to Routing Grids

This is the stage immediately after netlist decomposition. Here, the nets are mapped onto the routing grids. Essentially, the minimal grid pitch is calculated for both vertical and horizontal metal layers. It is also a common rule from most commercial PDKs that metal layers are strictly used either for vertical or horizontal routing. Using the design rules of ASAP7 PDK (Arizona State Predictive PDK) [29]as reference, it can be determined that metals M1, M3, M5 and M7 are used for vertical routing and metals M2, M4, M6, M8 are used for horizontal routing.

The pins or Steiner points obtained need not coincide with the grid pitch. Hence, it is moved onto the closest location and that node is selected as the source or node of each 2-pin segment which can be used for Dijkstra's algorithm. Also, the internal metal information need to be taken into account and certain edges and or nodes are marked as forbidden such that it doesn't cause DRC violations.

4.2 Dijkstra's Algorithm

Dijkstra's algorithm is a very popular shortest path finding algorithm which can be used for estimating routes. The routing grids are converted to a two dimensional graph G=(V,E) and the region of each segment is extracted. The edges are weighted and for each candidate when the edge is selected the weight is incremented and a penalty is added such that various options of candidates can be generated for the ILP solver.

4.3 Obtaining Multiple Candidates

Based on the area of each segment and the rectilinear distance of the the source and destination the number of candidates are calculated. In some cases, the distance between source and destination is a few grid points away and there can be only a few possible candidates possibly generated. But there are also cases where there can be multiple candidates generated. Based on these values, a simple estimation is conducted to get multiple Dijkstra solutions as candidates. These contain information of the grid points being used and the direction of each the path to get information regarding the number of bends as well as symmetry which are later necessary for the constraints.

4.4 Annotation of Candidates

So far, the physical distance and metal information have been considered while finding the candidate routes but the candidate themselves are yet to be annotated. The annotation occurs in this stage. Based on the direction and grid point, the metal layer to be used can be found out easily. Then the physical information of the metal is mapped onto the candidate. The following information is added:

1 Length of metal based on coordinates.

2 Minimum width of metal as per design rule.

3 Vias at the vertex to connect to next metal pices.

4 Via enclosures as per design rule.

5 Additional vias and metal segments to complete connection to original location of pins and Steiner points.

4.5 Candidate Selection

Once, the candidates are refined and translated they are introduced as inputs to an ILP solver that needs to pick a candidate for each segment of the net that satisfies all the symmetry and bend matching constraints as well as the overlap between different segments of nets. Once the final selected candidates are obtained it can be written in GDS format to obtain an actual layout. Though all the design rule constraints are met, it can be verified if the obtained layout is DRC clean and if the LVS is matched. This is usually the last stage of a PnR flow that completes the routing.

5. 3-PIN NET DECOMPOSITION

3-pin net decomposition is a novel variant to the original flow of netlist decomposition and candidate generation. The main ideology for this form of decomposition is to take care of the various routing constraints in the initial stages such that those segments of the nets are fixed. Also, by including the blockage avoidance feature in this stage, the 3-pin segments will not require candidates. Only the additional segments that are required to complete the routing of the net will go through candidate generation. This makes all the symmetry and matching constraints easily solvable and reduces the runtime and complexity of the flow drastically.

5.1 Methodology

This follows the original flow of generating routing grids followed by the iterative 1-Steiner method of constructing the RSMT. Once, the RSMT is created, the blockage avoidance is implemented and the constraints are added. If there is any matching or symmetry constraint with another net then both nets are parallely solved. The constraints of both nets say $N = (n_1, n_2)$ are considered at the same time where n_1 and n_2 are the pair of nets in the netlist with a matching constraint.

Based on the constraints the Steiner points are readjusted and the Hanan grid is modified. Now, to avoid multiple 2-pin segments, the net region for 3-pin that is of minimal area and doesn't fall into the region of other nets are grouped as a 3-pin net and is fixed. This 3-pin net segment is essentially pre-processed with all the constraints and other possible DRC violations, it is mapped onto the routing grids, metal and via annotated and, stored as a final route.

Now, the remaining routing is completed and then checked if all the constraints are met. If there is any issue with matching or possible overlap violation, then these 2-pin segments of the nets are sent for candidate generation. Else, the segment is directly sent to the ILP solver as a segment with just one candidate that is guaranteed to be picked. This naturally eliminates many unnecessary iterations in candidate generation and multiple estimations of the routes in more than

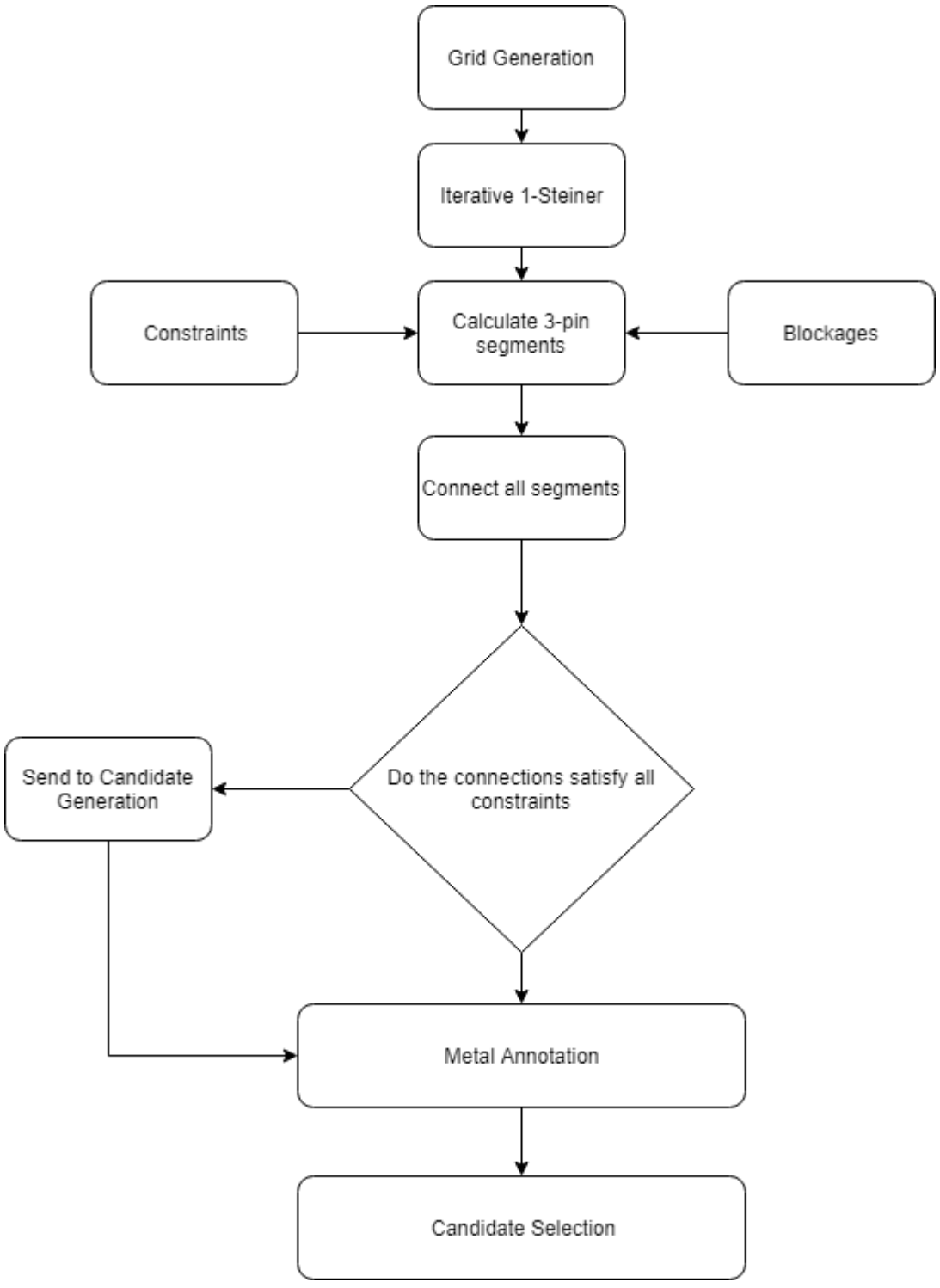

Figure 5.1: Simplified Flow when 3-pin decomposition is used.

1 stages of the PnR flow.

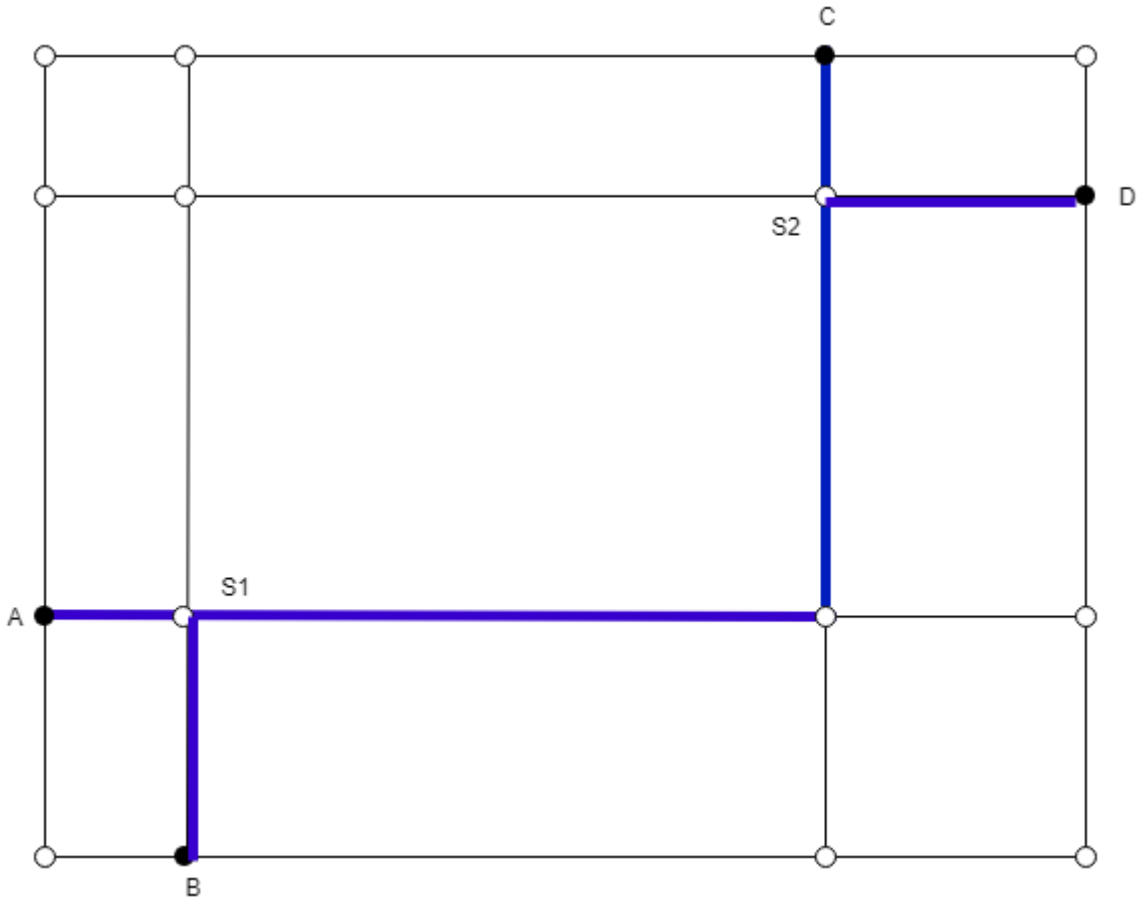

Figure 5.2: Example of 3-pin decomposition.

As illustrated in Figure 5.2, the highlighted path is the Steiner tree constructed on the Hanan grid. After iterative 1-Steiner construction, there are five 2-pin segments (A,S1), (B,S1), (S1,S2), (C,S2) and (D,S2). The candidate generation tool needs to be run five times. After grouping into 3-pin groups (A,B,S1), (C,D,S2) and a remaining segment (S1,S2), the 3-pin groups are mapped and fixed. The tool needs to be run at most once for (S1,S2) if the constraints are not satisfied.

The main advantage of this method is satisfying the main geometric and analog constraints that are crucial to the circuits. There are six main constraints that are taken into consideration for this method. They are:

1 Critical Nets

2 Wirelength Matching

3 Mirror Symmetry

4 Orientation Matching

5 Bend Matching

6 Topology Matching

Each of these constraints are explained in detail in the next sections.

5.2 Critical Nets

This is a constraint usually given to nets that connect cascades or those that carry signals and need to be minimized as much as possible. It is defined as CN where the key function is to minimize wirelength above everything else. During netlist decomposition, it neglects all blockages and other possible violations as it can simply use higher metal layers to achieve the shortest distance.

When the candidates are generated, there is very high penalty given to candidates with longer wirelength. Another important consideration is that in the ILP solver stage, the shortest candidates are given least cost and very high priority such that it is guaranteed to be picked unless it is infeasible to do so.

5.3 Wirelength Matching

This is a constraint that is usually given to clocking circuits or heavily time sensitive nets. It is represented as WM. The pair of nets need to have similar wirelengths with only 10 mismatch allowed overall. It uses a wirelength balance between the pair. As each segment, is constructed in the Steiner and offset is indicated for the other net which needs to be met. The objective is to maintain the wirelength balance at 0 when there is no mismatch at all as shown in the example in Figure 5.3.

When the candidates are generated, the paths in the Dijkstra algorithm that match wirelength are given reduced costs while the others are given a penalty. These candidates have a much higher priority than the rest and are more probable to be picked. Usually, these nets have a much longer wirelength than the minimal feasible wirelength.

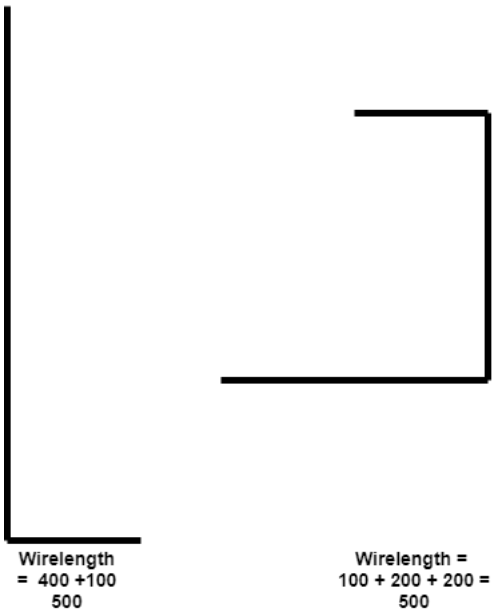

Figure 5.3: Wirelength Matching.

5.4 Mirror Symmetry

This type of symmetry is usually for symmetric circuits with components such as differential pairs. It is represented as MS. Here, the distance between each pin and the length of each segment during decomposition is perfectly matched. The die is figuratively folded along the line of symmetry and the blockages and internal metal information is mapped onto each side. After this, the net is decomposed and is replicated and flipped onto the other side to get matching pairs.

In the stage of candidate generation the similar process is followed to get symmetric candidate pairs. In the candidate selection, if any candidate is not selected due to violations or overlaps, the corresponding symmetry candidate is also eliminated to guarantee a feasible solution.

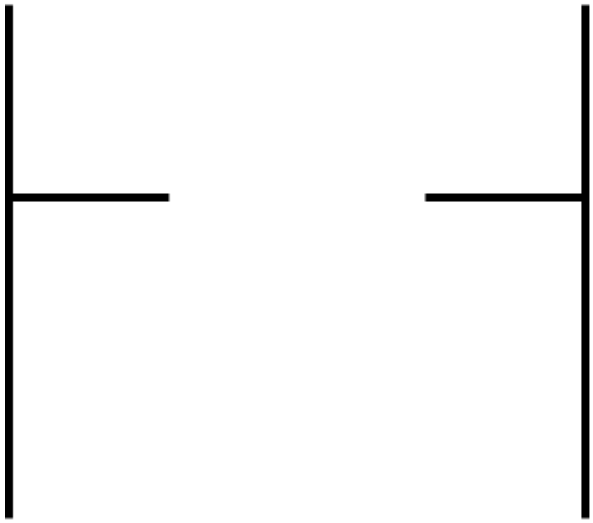

Figure 5.4: Mirror symmetry of a 3-pin segment.

5.5 Bend Matching

This is a constraint to maintain the same amount of signal reflection in both the nets. It is represented as BM. Here, the wirelength need not be minimized or equal. The only constraint is that the number of bends in the overall net should be matched. Similar to wirelength matching, a bend balance parameter is introduced. When a bend occurs in a net during decomposition, an offset is indicated on the other net. In this case, thee is no relaxation. The bend balance must result to a 0 for the condition to be satisfied.

In the candidate generation, the edges that are perpendicular to the current path have reduced costs such that they are selected to equalize the number of bends for both nets. Similar to MS, the pairs sent to ILP solver, but if a candidate for a net is eliminated its pair is eliminated only if no other candidate of that same number of bends exist.

5.6 Orientation Matching

This is a constraint to maintain the same direction of signal in both the nets or parallel routing like common mode inputs or outputs and opamps. It is represented as OM. Here, the wirelength need not be minimized or equal. The only constraint is that the bends are in same direction.

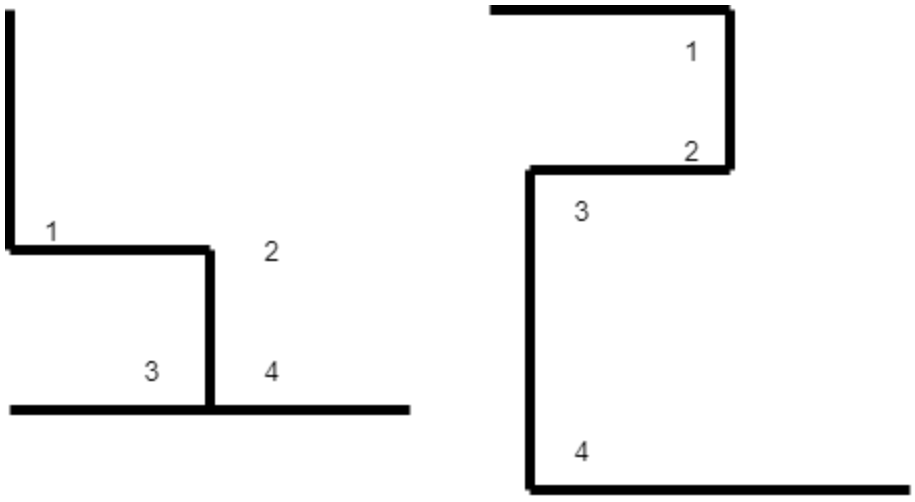

Figure 5.5: Bend Matching.

Similar to wirelength matching, a bend direction parameter is introduced. When a bend occurs in a net during decomposition, an enum is introduced (0,1,2,3) to represent the four possible bends which needs to be matched in the other net as well.

In the candidate generation, the edges are forced to move in a particular direction to match the bend to satisfy the constraint. Similar to MS, the pairs sent to ILP solver, if any candidate is not selected due to violations or overlaps, the corresponding matching candidate is also eliminated to guarantee a feasible solution.

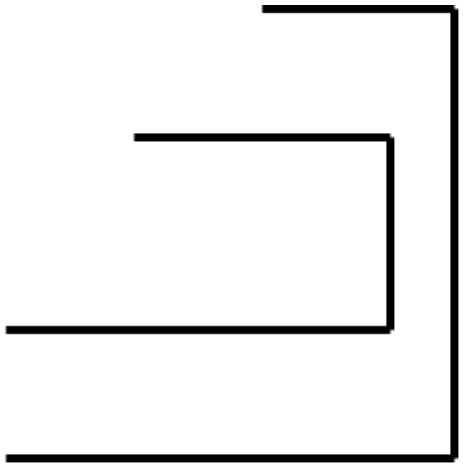

Figure 5.6: Orientation Matching.

5.7 Topology Matching

This is a constraint to maintain the parasitics of the nets equivalently. This means that the same type of metal layers need to be used which is crucial for electrical and RC delay sensitive circuits. It is represented as TM. Here, the wirelength needs to be equal with some relaxation similar to WM. The other constraint is that the metal segments are matched. Similar to wirelength matching, a metal parameter is introduced. When a routing grid is used, the same layer of grid is used for its counterpart pair. This is done to ensure that the same metal layers will be used.

In the candidate generation, the edges of other metal layers are set to infinity which forces the algorithm to use the same metal layer. Similar to MS, the pairs sent to ILP solver, if any candidate is not selected due to violations or overlaps, the corresponding matching candidate is also eliminated to guarantee a feasible solution.

6. EXPERIMENTAL RESULTS

For the implementation of this methodology, a Linux environment was used and run on the servers. The basic design flow of placement and routing was taken from the Analog Layouts using Intelligently Generated Netlists (ALIGN) project of the IDEA challenge by DARPA. The key idea of input files were created based on GSRC benchmarks to test. Over 115 nets were tested for different constraints and blockages. As a comparative study for the 3-pin net decomposition, Steiner tree softwares FLUTE[16] and FastSteinerUM[20] were used. LPSolve software was used as the ILP solver in the candidate selection stage of the design flow.

Based on the design rules of ASAP7 PDK, the following values are used to obtain grid pitch for each metal layer using the equation.

$$GridPitch = MinWidth + Max\{MinSpacingsidetoside, MinSpacingTiptoTip, MinViaSpacing\} + 2 \times Viaenclosure$$

Metal	Direction of Route	Grid Pitch
M1	Vertical	50
M2	Horizontal	50
M3	Vertical	100
M4	Horizontal	100
M5	Vertical	150
M6	Horizontal	150
M7	Vertical	200
M8	Horizontal	200

Table 6.1: Grid Pitch for each Layer

Based on these calculations, eight layers of routing grid was used for routing tracks. The 3-

pin net decomposition method with candidate generation was compared with the regular 2-pin net decomposition method. They are represented as 3P Decompose and 2P Decompose in the various results obtained.

6.1 3-pin Decomposition vs 2-pin Decomposition

For the test cases, multi-pin nets were used ranging from 3 pins to as high as 12 pin nets. The test cases also included different bends, wirelengths and blockages in various combinations to verify all the possible constraints. Based on the final candidates obtained the following results were obtained. One of the key objective of routing is the minimization of wirelength. Table 6.2.

Net Degree	3P Decompose	2P Decompose
3	333.33	372.4
4	564.8	588.27
5	590	612.5
6	645.3	649.67
7+	780.25	841.25

Table 6.2: Average Wirelength for 3P and 2P decompose for various net degrees

represents the average Wirelength for 3P and 2P decompose for various net degrees over the 115 test cases. It can be generally observed that for all degrees of nets, the average wirelength is lesser for 3P decompose when compared to 2P decompose as the constraints are taken into account and optimally fixed as 3-pin segments early on. This has a more evident impact for lower net degrees that have lesser rerouting segments such as nets with degree 3 and 4. This minimization of wirelength is less evident as the degree increases. This can also be observed in Figure 6.1. For very large net degrees of 7+ and extreme cases, the 3P decompose provides a better solution for wirelength minimization.

Another key observation that can be made is the number of Steiner points generated which is an important metric. There are multiple iterations in generating a Steiner point. The computations increase marginally in the whole flow when Steiner points increase. The results as shown in Figure

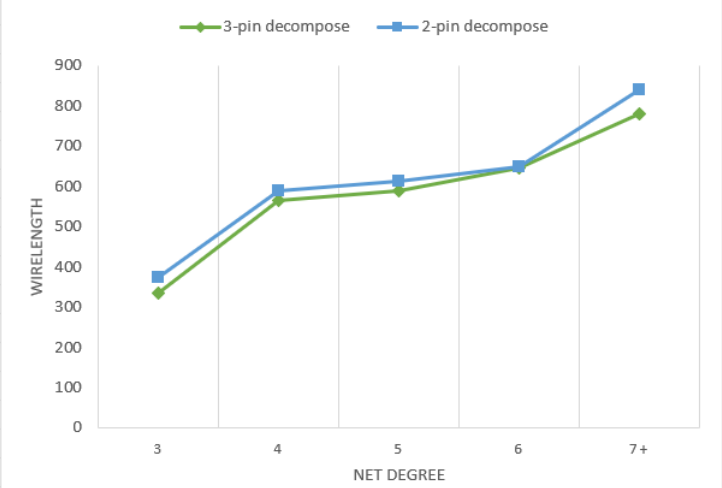

Figure 6.1: Average Wirelength for 3P and 2P decompose for various net degrees.

6.2 represent that the 3P decompose creates more Steiner points in average when compared to 2P decompose for various net degrees. This is because the 3P decompose method tries to solve constraints such as bend matching and symmetry while decomposing along with blockage avoidance. There are many cases where by choosing a Steiner point, there is a more rerouting to avoid blockages and to complete connections. This implies that the matching constraints cannot always be met and the wirelength on the whole net may not be minimized. In such cases, generating a sub optimal Steiner point along with another point solves the problem. The trade off to meeting all these constraints is the increase in computations which does not have a huge impact on performance of the router as the number of blocks and nets in analog circuits are very less.

As the bend and wirelength are matched and minimized much more in 3P decompose when compared to 2P decompose, the results have shown that lesser metal layers are used in 3P decompose for various net degrees. This proves to be an advantage as this minimizes the number of vias used that translates to lesser DRC errors. Also, the RC delay will be reduced as the lower metals layers have lesser resistance that higher metal layers. Generally, in industrial tools the higher metal layers are preferred to be exclusive for Shielding and Power routing which can be easily achieved with the 3P decompose method.

The main advantage of the 3P decompose method is that it reduces the number of segments

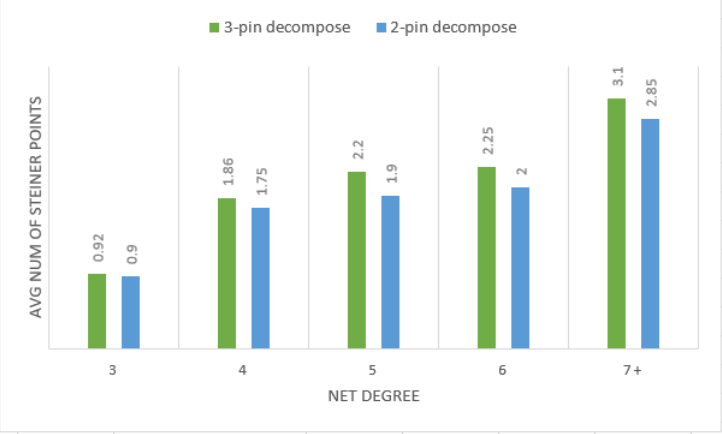

Figure 6.2: Average Number of Steiner Points generated for 3P and 2P decompose for various net degrees.

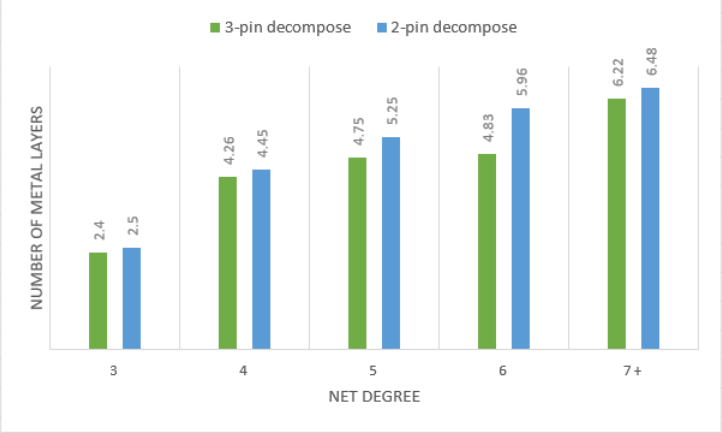

Figure 6.3: Average Number of Metal Layers used for 3P and 2P decompose for various net degrees.

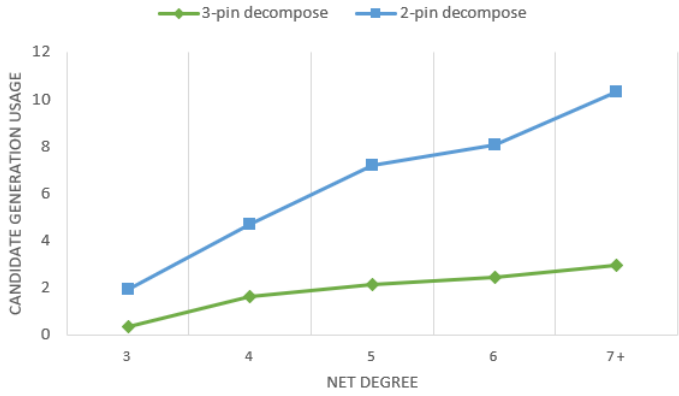

Figure 6.4: Average Number of times the Candidate Generation tool was used for 3P and 2P decompose for various net degrees.

Net Degree	3P Decompose	2P Decompose
3	0.35	1.93
4	1.6	4.66
5	2.15	7.18
6	2.46	8.05
7+	2.95	10.33

Table 6.3: Average Number of times the Candidate Generation tool was used for 3P and 2P decompose for various net degrees

that the net is decomposed to and fixes the 3-pin segments by considering all net constraints in the decomposition stage itself. By doing this, very few segments that connect the 3-pin segments need to be rerouted. Only these segments require the Candidate generation tool. Based on the results in Table 6.3, it is very clear that the Candidate Generation tool is called lesser in 3P decompose than 2P decompose as expected. By observing the trend in Figure 6.4, it can be said that instead of a linear increase in Candidate Generation usage with increase in degree as seen for 2P decompose the usage is almost constant with very little increase as the net degree increases. From this the main inference that can be made is that the number of computations is drastically reduced by eliminating the usage of the Candidate generation tool for many segments.

7. CONCLUSION AND FUTURE SCOPE

The 3-pin net decomposition and candidate generation is a novel approach to tackling the problems of Analog IC routing. By taking advantage of the analog domain and its small scale of netlists, Netlist Decomposition, Candidate Generation and Selection is a simple yet effective approach when compared to Global and Detailed routing. 3-pin net decomposition is a smart approximation that considers both blockages and constraints on nets to pre-process them while constructing the Steiner tree. Experimental results show that this method significantly reduces computations while minimizing the wire length and accurately satisfying the unique constraints of analog circuits.

This work has scope for other variations as well. This work considers only signal routing with critical net constraints and matching, symmetry and bend constraints for pairs. Work can be done to include shielding constraints and power routing as well. Also, another key factor is that analog blocks are of mixed sizes and hence a minimal grid pitch will create huge grids and increase the complexity exponentially. Work can be done to recognize the sizing constraints and create dynamic routing grids. Another variation that can be considered is multi-contact routing for nets with large pins.

REFERENCES

[1] B. M. Waxman, "Routing of multipoint connections," in *IEEE Journal on Selected Areas in Communications*, vol. 6, pp. 1617–1622, 1988.

[2] K. Sajid, J. D. Carothers, J. J. Rodriguez, and W. T. Holman, "Global routing methodology for analog and mixed-signal layout," in *IEEE International ASIC/SOC Conference*, pp. 442–446, 2001.

[3] C. Du, Y. Cai, X. Hong, and Q. Zhou, "A shortest-path-search algorithm with symmetric constraints for analog circuit routing," in *IEEE International Conference on ASIC*, vol. 2, pp. 844–847, 2005.

[4] J. Hu, C. J. Alpert, S. T. Quay, and G. Gandham, "Buffer insertion with adaptive blockage avoidance," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 492–498, 2003.

[5] M. Zachariasen, "A catalog of Hanan grid problems," *Networks: An International Journal*, vol. 38, no. 2, pp. 76–83, 2001.

[6] P.-H. Wu, S.-Y. Bai, and T.-Y. Ho, "A topology-based eco routing methodology for mask cost minimization," in *IEEE/ACM Asia and South Pacific Design Automation Conference*, pp. 507–512, 2014.

[7] M. W. Bern, "Two probabilistic results on rectilinear Steiner trees," *Algorithmica*, vol. 3, no. 1-4, p. 191, 1988.

[8] C. Ting-Hai and H. Y. Chin, "Rectilinear Steiner tree construction by local and global refinement," in *IEEE International Conference on Computer-Aided Design*, pp. 432–435, 1990.

[9] X. Li and Y. Luo, "Weighted lee algorithm on rectilinear Steiner tree with obstacles and boundary," in *World Congress on Computer Science and Information Engineering*, vol. 3, pp. 369–374, 2009.

[10] S.-E. D. Lin and D. H. Kim, "Construction of all rectilinear Steiner minimum trees on the Hanan grid," in *ACM International Symposium on Physical Design*, pp. 18–25, 2018.

[11] A. B. Kahng and G. Robins, "A new class of iterative Steiner tree heuristics with good performance," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 11, no. 7, pp. 893–902, 1992.

[12] T. Barrera, J. Griffith, S. A. McKee, G. Robins, and T. Zhang, "Toward a Steiner engine: enhanced serial and parallel implementations of the iterated 1-Steiner MRST algorithm," in *IEEE Design Automation of High Performance VLSI Systems*, pp. 90–94, 1993.

[13] A. B. Kahng, "A Steiner tree construction for VLSI routing," in *IEEE International Joint Conference on Neural Networks*, vol. 1, pp. 133–139, 1991.

[14] C. J. Alpert, G. Gandham, J. Hu, J. Neves, S. T. Quay, and S. S. Sapatnekar, "Steiner tree optimization for buffers, blockages, and bays," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 4, pp. 556–562, 2001.

[15] C.-Y. Wu, H. Graeb, and J. Hu, "A pre-search assisted ILP approach to analog integrated circuit routing," in *IEEE International Conference on Computer Design*, pp. 244–250, 2015.

[16] C. Chu and Y.-C. Wong, "FLUTE: Fast lookup table based rectilinear steiner minimal tree algorithm for vlsi design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 1, pp. 70–83, 2007.

[17] A. B. Kahng, I. I. Măndoiu, and A. Z. Zelikovsky, "Highly scalable algorithms for rectilinear and octilinear Steiner trees," in *IEEE/ACM Asia and South Pacific Design Automation Conference*, pp. 827–833, 2003.

[18] A. Z. Zelikovsky, "An 11/6-approximation algorithm for the network Steiner problem," *Algorithmica*, vol. 9, no. 5, pp. 463–470, 1993.

[19] W.-m. Hwu, K. Keutzer, and T. G. Mattson, "The concurrency challenge," *IEEE Design & Test of Computers*, vol. 25, no. 4, pp. 312–320, 2008.

[20] J. C. Zolper, "The GSRC's role in meeting tomorrow's design challenges," *IEEE Design & Test of Computers*, vol. 25, no. 4, pp. 366–367, 2008.

[21] M. Cho and D. Z. Pan, "BoxRouter: a new global router based on box expansion and progressive ILP," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 12, pp. 2130–2143, 2007.

[22] Z. Cao, T. Jing, J. Xiong, Y. Hu, L. He, and X. Hong, "DpRouter: A fast and accurate dynamic-pattern-based global routing algorithm," in *IEEE/ACM Asia and South Pacific Design Automation Conference*, pp. 256–261, 2007.

[23] M. M. Ozdal and M. D. Wong, "Archer: a history-driven global routing algorithm," in *IEEE International Conference on Computer-Aided Design*, pp. 488–495, 2007.

[24] M. D. Moffitt, "MaizeRouter: Engineering an effective global router," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 11, pp. 2017–2026, 2008.

[25] Y. Xu, Y. Zhang, and C. Chu, "FastRoute 4.0: global router with efficient via minimization," in *IEEE/ACM Asia and South Pacific Design Automation Conference*, pp. 576–581, 2009.

[26] T.-H. Wu, A. Davoodi, and J. T. Linderoth, "GRIP: scalable 3d global routing using integer programming," in *ACM Design Automation Conference*, pp. 320–325, 2009.

[27] Y.-J. Chang, Y.-T. Lee, J.-R. Gao, P.-C. Wu, and T.-C. Wang, "NTHU-route 2.0: a robust global router for modern designs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 12, pp. 1931–1944, 2010.

[28] N. Viswanathan and C.-N. Chu, "FastPlace: efficient analytical placement using cell shifting, iterative local refinement, and a hybrid net model," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 5, pp. 722–733, 2005.

[29] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthy, and G. Yeric, "ASAP7: A 7-nm finFET predictive process design kit," *Microelectronics Journal*, vol. 53, pp. 105–115, 2016.