Exploring the Scope of Unconstrained Via Minimization by Recursive Floorplan Bipartitioning

BAPI KAR, Indian Institute of Technology
SUSMITA SUR-KOLAY, Indian Statistical Institute
CHITTARANJAN MANDAL, Indian Institute of Technology

Random via failure is a major concern for post-fabrication reliability and poor manufacturing yield. A demanding solution to this problem is redundant via insertion during post-routing optimization. It becomes very critical when a multi-layer routing solution already incurs a large number of vias. Very few global routers addressed unconstrained via minimization (UVM) problem, while using minimal pattern routing and layer assignment of nets. It also includes a recent floorplan based early global routability assessment tool STAIRoute [15].

This work addresses an early version of unconstrained via minimization problem during early global routing by identifying a set of minimal bend routing regions in any floorplan, by a new recursive bipartitioning framework. These regions facilitate monotone pattern routing of a set of nets in the floorplan by STAIRoute. The area/number balanced floorplan bipartitioning is a multi-objective optimization problem and known to be NP-hard [25]. No existing approaches considered bend minimization as an objective and some of them incurred higher runtime overhead. In this paper, we present a Greedy as well as randomized neighbor search based staircase wave-front propagation methods for obtaining optimal bipartitioning results for minimal bend routing through multiple routing layers, for a balanced trade-off between routability, wirelength and congestion.

Experiments were conducted on MCNC/GSRC floorplanning benchmarks for studying the variation of early via count obtained by STAIRoute for different values of the trade-off parameters $(\gamma, \beta)$ in this multi-objective optimization problem, using 8 metal layers. We studied the impact of $(\gamma, \beta)$ values on each of the objectives as well as their linear combination function $Gain$ of these objectives.

Additional Key Words and Phrases: Recursive floorplan bipartitioning, minimal bend monotone staircase routing regions, randomized neighbor search, staircase wave-front propagation, unconstrained via minimization, early global routing.

ACM Reference format:
Bapi Kar, Susmita Sur-Kolay, and Chittaranjan Mandal. 2016. Exploring the Scope of Unconstrained Via Minimization by Recursive Floorplan Bipartitioning. 1, 1, Article 1 (January 2016), 25 pages.
DOI: 10.1145/nnnnnn.nnnnnnn

1 INTRODUCTION

With sustained advancement in IC fabrication technology, stringent design rules are evolving by exponentially large numbers. Straightforward routing solutions from a HPWL aware placement solution may not yield an acceptable physical design closure due to too many routing violations in subsequent global routing. If these violations are not resolved by the subsequent detailed routing
or by an iterative global and detailed routing, the placement (or floorplanning or even logic restructuring) of the design should be redone. In practice, several iterations in block placement (and floorplanning) are required for complex designs containing multi-million gates in order to attain a feasible routing solution (see Fig. 2 (a)). Therefore, it has been a mandate to consider different global routing metrics such as routability, wirelength, congestion [6, 22, 23] and even timing [31] as the prime objectives in a placement problem. Some placement engines, however, integrated faster global routing solutions for iterative improvement of the placement solution [13, 28, 29, 33, 34]. For fewer design iterations before successful routing closure, integrated global/detailed routing methods were also explored [37].

Modern VDSM fabrication processes, such as 65nm and below, continue to allow more routing layers with varying metal width/pitch for successful routing completion. A routing solution with excessive via count not only causes design for reliability issues due to random via failures [7], but also impacts the circuit performance due to increased resistance along the routing paths with more vias. Double via insertion during post-routing layout optimization or identifying a via-failure aware routing [7] as depicted in Fig. 1 or even redundant via aware ECO routing during mask optimization [8] for increased reliability and yield of the fabricated design are some of the known approaches to minimize these failures. Moreover, vias consume substantial routing area and pose as additional routing blockages in the routing regions impacting routability of the design. Therefore, via minimization [31] is a critical problem to handle in physical design flow. There are two approaches: (a) unconstrained via minimization (UVM), and (b) constrained via minimization (CVM). While UVM identifies a routing path of a net with minimal number of vias along it for a given number of routing (metal) layers, CVM approaches aims to minimize the number of vias while keeping the routing topology unchanged. This routing topology is obtained by planar routing solution during early phases of global routing. Although, both are known to be NP-hard.
Exploring the Scope of Unconstrained Via Minimization by Recursive Floorplan Bipartitioning

problems, UVM is much harder than CVM [14, 31]. Existing global routers [9, 28, 30, 36], except a few like [5, 24, 27, 38], used CVM based layer assignment approaches on a planar routing solution for reducing via count as well as mitigating congestion [21].

![Diagram of the Physical Design (PD) Flow: (a) Traditional [2, 31], and (b) New [15]]

Recently, an early global routing (EGR) method STAIRoute [15] was proposed for early routability assessment of a floorplanned layout, facilitated by a monotone staircase cut based recursive floorplan bipartitioning framework [18, 25, 26]. These bipartitioners work on any floorplan irrespective of their sliceability. As [12, 32] pointed out, the monotone staircase routing framework ensures an well defined routing order of the nets, based on the net cut information available with the nodes in the bipartitioning hierarchy [18, 25, 26]. As highlighted in Fig. 2 (b), STAIRoute works in two stages: (a) enumerating the monotone staircase routing regions in a floorplan by recursive bipartitioning using monotone staircase cuts [18, 25, 26], and (b) proposing an early global routing model for routing these nets through a number of metal layers, using these bipartitioning results.
The existing bipartitioning methods using monotone staircase cuts [11, 18, 25, 26] considered only two objectives: (i) the area (number) of the blocks in each bipartition to be maximized, and (ii) the number of nets cut by a bipartition be minimized. While the former objective is related to the height of the bipartition hierarchy (also known as MSC tree [18]), minimizing the number of nets being cut has several advantages like: (a) distributing the routing paths of the nets uniformly across the entire layout, (b) reducing routing violations due to congestion hot-spots (congestion > 100%), (c) achieving uniform wire distribution across the layout for minimal variation due to chemical mechanical polishing (CMP) process, and (d) minimizing cross-talk effect due to long (global) nets running through the longer staircases, specially those nets corresponding to the upper nodes in the bipartition tree. In global routing, routing path of a net using multi-bend monotone pattern routing and its variants L/Z patterns [5, 19] is confined within the net bounding boxes. Therefore, identification of a minimal bend monotone patterns can potentially yield fewer via counts, while L/Z patterns use minimum of one/two vias respectively for minimum layer change.

In this work, we propose a new recursive floorplan bipartitioning framework, for identifying minimal bend monotone staircase routing regions in a floorplan, in order to use fewer vias during early global routing of the nets in the floorplan. The key contributions of this paper are:

1. define a new objective of bend minimization in the existing multi-objective floorplan bipartitioning problem;
2. propose a greedy method for identifying minimal bend monotone staircase routing regions for early global routing with smaller via count (this is an early approach for unconstrained via minimization (UVM)); and
3. introduce a randomized neighbor search technique and staircase wave front propagation approach for exploring a larger solution space of potentially optimal minimal bend monotone staircase regions in a floorplan.

The organization of this paper is as follows: in Section 2, we discuss the background on monotone staircase routing region definition in a floorplan. The proposed floorplan bipartitioning method, for identifying a set of monotone staircases with minimal number of bends for the entire floorplan, is presented in Section 3. Section 4 discusses the basis for an extension of this greedy bipartitioning method and illustrates a new randomized neighbor search technique and the corresponding staircase wave-front propagation approach. Experimental results and relevant discussions are covered in Section 5, followed by the summary of this work in Section 6.

2 BACKGROUND ON MONOTONE STAIRCASE CUTS

Before discussing the proposed recursive floorplan bipartitioning method, we revisit the formulation of an unweighted directed graph $G_b(V_b, E_b)$, namely block adjacency graph (BAG) [18, 26], used to define the adjacency relation of a set of $n$ blocks $B = \{b_i\}$ in a given floorplan $F$. The graph $G_b(V_b, E_b)$ is defined as follows: the vertex set $V_b = \{v_i\}$ where $v_i$ corresponds to block $b_i$ and the edge set $E_b = \{e_{ij}\}$ where $e_{ij} = \{(v_i, v_j) \mid \text{block } b_i \text{ is on the left of (above) an adjacent block } b_j \text{ in } F\}$. The vertices corresponding to the top-left and bottom-right corner blocks are designated as the source and the sink vertices respectively, with zero in-degree and out-degree respectively. This definition yields a monotonically increasing staircase (MIS) $C_I$ (see Fig. 3 (a)).

The definition of BAG for obtaining a monotonically decreasing staircase (MDS) is as follows: edge $e_{ij} = \{(v_i, v_j) \}$ for a pair of adjacent blocks $(b_i, b_j)$ such that $b_i$ is to the left of (below) $b_j$. The source and sink vertices are identified as the vertices pertaining to the bottom-left and top-right corner blocks respectively. This scenario is captured in Fig. 3 (b) along with the MDS cut $C_D$. In the rest of the paper we refer an MIS/MDS cut as a $ms$-cut unless stated explicitly. It is to be noted that, unlike in [25, 26], this graph based framework does not consider any netlist information.
while constructing BAG for faster bipartitioning results. The netlist information is solely used to identify the cut nets and the uncut nets that fall on either side of the bipartition. These uncut nets and the respective parts of the cut nets with at least two pins in each part. In this method, net cut information in each level of the bipartition hierarchy is a measure of the optimality of each ms-cut obtained, and is referred to as min-cut balanced floorplan partitioning [25, 26].

In order to ensure each cut in BAG is an ms-cut, we refer to the following lemma given in [26], commonly known as monotone staircase property.

**Lemma 2.1.** If $e_{ij} \in E_b$ is an arc in $G_b$, then there exists at least one monotone staircase in the floorplan such that the blocks $b_i$ and $b_j$ appear in the left and right partitions respectively, and there exists no staircase with $b_i$ in the right partition and $b_j$ in the left partition.

**Proof.** In [26]. □

In Fig. 4, we illustrate the working of Lemma 2.1 for an MIS cut, which is equally applicable for an MDS cut. It shows that all the cut edges in the BAG are forward edges, i.e., directed from the left partition containing the source vertex $A$ towards the right partition containing the sink vertex $J$ yielding a valid monotone staircase cut. However, in Fig. 4 (b), the highlighted edge $(B,E)$ in the BAG is directed from the right partition to the left partition. This cut leads to a non-monotone staircase cut. From this illustration and Lemma 2.1, we observe that it requires at least one back edge directed from the right to left partition to generate a non-monotone staircase cut.

**Corollary 2.2.** Given a BAG formulated for obtaining a MIS (MDS) cut, any cut which has at least one back edge results in a non-monotone staircase cut.

**Proof.** From Lemma 2.1 and Fig. 4 (b). □

In order to study the advantage of early global routing using monotone staircase patterns over non-monotone staircases, we consider the example in Fig. 5 for two different routing instances of a two pin net $n$ having terminal pins $(A, B)$. Wirelength for the monotone routing path is equal to half of the bounding box length of the net, i.e., half perimeter wirelength (HPWL), while that of...
the non-monotone path yields extra wirelength beyond HPWL. This eventually consumes more routing area and hence increases the congestion in the routing regions, impacting the routability of the nets. A non-monotone pattern may also require more number of vias depending the number of bends in it. On the other hand, a suitably chosen monotone staircase pattern with fewer bends in it may yield fewer vias. Therefore, pattern routing using non-monotone staircases is not beneficial for identifying a shortest routing path, as well as fewer via counts. Nevertheless, non-monotone routing [38] or maze routing [20, 31] can be effective when monotone or L/Z [5, 19] patterns can not be used due to heavy congestion and more routing blockages due to already routed nets within the bounding box of a net. This leads to a detoured routing path with increased wirelength and possibly higher via count, identified using non-monotone or maze routing.

Our study also shows that a very large number of monotone routing paths with varying number of bends are possible within the bounding box of a net, L/Z patterns being a subset of all those possible patterns with only one/two via overhead. An optimal monotone pattern is the one which takes minimal number of (bends) vias to complete the routing between a pair of pins through a set of metal layers, thus motivating this work. In this paper, the proposed recursive bipartitioning framework identifies a set of optimal monotone staircases with minimal number of bends in a given floorplan, for early global routing of the nets with minimal wirelength and via count. In this paper, we used only STAIRoute as the early global routing tool, by preferred directional routing in different metal layers.

3 MONOTONE STAIRCASE BIPARTITIONING WITH MINIMAL BENDS

In this section, we discuss the proposed recursive floorplan bipartitioning method in order to identify a set of minimal bend monotone staircase routing regions in a floorplan, for obtaining the shortest routing paths of a set of nets in floorplan, by an early global routing framework such as STAIRoute [15]. Before that, we study the impact of a number of bends in a monotone staircase...
routing region on the number of vias when a net is routed through it, using reserved layer model for layer assignment of the net segments in different routing layers. In this routing model, horizontal and vertical segments of a net are routed through designated metal layers, say $M_1$ and $M_2$ respectively (see Fig. 5). This requires inter-layer metal interconnects, called vias, to establish electrical connections between the wire segments of a net running in different layers. For the sake of simplicity, we assume routing with two routing layers ($M_1, M_2$), although it can be extended to any number of permissible layers in the fabrication processes.

![Fig. 5. Illustrating two routing instances of a 2-pin net $n = (A,B)$ using a: (a) monotone staircase path, and (b) non-monotone staircase path.](image)

We consider two different routing instances between the terminal points (pins) $x$ and $y$ of a net segment $n$ as depicted in Fig. 6. These routes, denoted as $n_{xy}$ and $n'_{xy}$ respectively, use different
monotone staircase paths with different bend counts. While routing path $n_{x,y}$ uses five vias, $n'_{x,y}$ requires only three vias. From this example, we infer that a monotone staircase with fewer bends can potentially reduce the number of vias when a net is routed through it using different metal layers and hence serves as the motivation of this work.

The problem definition in this work is augmented over the existing bipartitioning methods such [11, 18, 25, 26], considering a new objective of bend minimization. We enlist the objectives of this new multi-objective optimization problem as below:

1. balance ratio $\text{balr} = \min(A_l, A_r)/\max(A_l, A_r)$ be maximized
2. the number of cut nets ($k_c$) be minimized, and
3. the number of bends ($z$) in the monotone staircase be minimized

where $A_{l(r)} = \sum_{b_i \in B_{l(r)}} \text{Area}(b_i)$, the area of the left (right) partition and $B_{l(r)}$ denotes the set of blocks in the left (right) partition. The number balanced bipartition problem can be seen as a restricted version of the area balanced bipartitioning problem when the area of each block is almost equal, i.e., having negligible variance in block area such that they can be normalized to unity. In this case, $\text{balr}$ is defined as $\min(n_l, n_r)/\max(n_l, n_r)$, where $n_l$ ($n_r$) denotes the number of blocks in the left (right) partition.

A linear combination function of these objectives, with a pair of trade-off parameters ($\gamma, \beta$), is defined as below:

$$\text{Gain} = \gamma \cdot \text{balr} + (1 - \gamma - \beta)(1 - k_c/k) + \beta(1 - z/z_{\text{max}})$$

(1)

where $z_{\text{max}}$ is the maximum possible number of bends if the constituent rectilinear segments in the corresponding monotone staircase had alternating (vertical or horizontal) orientation. It is computed as one fewer than the number of segments in it. Notably, Eqn. 1 is similar to that defined in [18] when $\beta = 0.0$. Careful selection of ($\gamma, \beta$) pair may yield an optimal balance among these objectives, not necessarily a global optimum. Since the area balanced bipartitioning is an NP-hard problem [25], the optimum balance among these objectives is hard to obtain in polynomial time. Instead, for a given ($\gamma, \beta$) pair, an optimal monotone staircase with maximum $\text{Gain}$ is chosen out of those with $\text{Gain}$ values in the sequence of $n - 1$ bipartitions of a floorplan of $n$ blocks [18] (see Fig. 7), at a given bipartition hierarchy. In Section 5, we study the bipartitioning results with a range of ($\gamma, \beta$) values on a set of floorplan benchmark circuits.

Now we refer to Fig. 7 for the working of this bipartitioning framework while maximizing the area in each partition and assessing the corresponding bends in the resulting monotone staircase. In this study, we do not consider minimal net cut for the sake of simplicity and restrict only to area balance and minimal bend count. The bipartition instance in Fig. 7 (a) and (h) gives minimum number of bends ($z = 3$), but with poor area balance. The area balance between the partitions keeps on improving through the instances depicted in Fig. 7 (b)-(e) with varying number of bends, while it declines for instances shown in Fig. 7 (f)-(h). The best possible area balance may be attained in case of the bipartition in Fig. 7 (e), but yields the worst bend count ($z = 6$) among all others. Therefore, a suitable trade-off between area balance and bend count has to be made based on ($\gamma, \beta$) values. The bipartition instance with $z = 4$ in Fig. 7 (d) appears to be a good choice among all the other instances. The following lemma gives a measure of the number of bends in a monotone staircase.

**Lemma 3.1.** Given a floorplan with $n$ blocks, the number of bends in a monotone staircase routing region is $O(n)$.

**Proof.** The number of bends in a monotone staircase can be at most one fewer than the number of cut edges in BAG $G_b$ due to alternate orientation of the contiguous cut edges. Since $G_b$ is a planar
Fig. 7. A sequence of $n - 1$ monotone staircases with varying number of bends (denoted as ■) in a given floorplan: (a) 3, (b) 5, (c) 5, (d) 4, (e) 6, (f) 5, (g) 5 and (h) 3.
graph \([18, 26]\) and \(|E_b|\) is \(O(n)\), the number of cut edges (a subset of \(E_b\)) that constitutes a monotone staircase is also \(O(n)\). \(\square\)

### 3.1 The Algorithm: BFS based Greedy Approach

The pseudo-code for the proposed monotone staircase partitioning method with minimal bends, namely \(MSCut\_Bend\_BFS\), is presented in Algorithm 1. The inputs to this method are the BAG \(G_b\) obtained from a given floorplan \(F\) of a set of blocks \(B\), a set of nets \(N\), the trade-off parameters \((\gamma, \beta)\) such that \(\gamma, \beta \in [0, 1]\). The balance type \(baltype\) dictates either an area or a number balanced bipartitioning \([25, 26]\). Unlike the previous works, we focus on area balanced partitioning only, since number balanced mode is a special case of it. The key differences between \(MSCut\_Bend\_BFS\) and the bipartitioning method in \([18]\) are: (i) bend minimization considered as an additional objective, and (ii) no restriction on the convergence within user-defined area bounds. In rare floorplan instances, these area bounds in \([18]\) may lead to exploration of a sequence of \(n - 1\) monotone staircases. On contrary, our method is able to explore a sequence of \(n - 1\) staircases without any such constraints on any floorplan of \(n\) blocks.

**Algorithm 1:**

```
input : \(G_b, N, \gamma, \beta, baltype\)
output: An optimal monotone staircase for a given \((\gamma, \beta)\) with maximal area balance, minimal net cut and minimal number of bends

1. Initialize a Queue \(Q\) and the left partition \(L = \emptyset\)
2. Enqueue the source vertex of \(G_b\) in \(Q\) as (BFS) level 0 vertex, and include it in \(L\) (right partition \(R = V_b \setminus L\))

   /* A vertex once enqueued always remains in \(L\) \([18]\)*/
3. Also enqueue \(\emptyset\) as BFS level indicator

4. while \(Q\) is not empty do
5.   Let \(v_l\) be the dequeued vertex
6.   if \((v_l \neq \emptyset)\) then
7.     for \((v_j \in adj(v_l))\) do
8.       if \((v_l, v_j)\) results in a valid ms-cut (see Lemma 2.1) then
9.         Enqueue the vertex \(v_j\) and include it in \(L\)
10.        Compute the parameters for the \((L, R)\) partition (see Eqn. 1) and store them in a list \(\lambda\)
11.       end
12.     end
13.   else
14.     Increment BFS level
15.     Enqueue \(\emptyset\) as next BFS level indicator
16.   end
17. end
18. Return an optimal monotone staircase with the maximum \(Gain\) value \(C_{max} \in \lambda\)
```

The recursive procedure for obtaining a set of minimal bend monotone staircases for the entire floorplan is presented in Algorithm 2, by recursively calling \(MSCut\_Bend\_BFS\) with a set of required inputs. Here, \(stype\) dictate the output staircase type, either an MIS or MDS (see Fig. 3). In this procedure, the root node of the bipartition hierarchy starts with a particular type e.g. MIS, followed by alternating types in the subsequent levels of the hierarchy. An example of a bipartition (MSC) tree in Fig. 8 illustrates a set of optimal monotone staircases (MIS/MDS) with minimal bends are overlaid on an input floorplan of 17 blocks.
Exploring the Scope of Unconstrained Via Minimization by Recursive Floorplan Bipartitioning

**Algorithm 2: Hier_MSCut_Bend**

- **Input**: $B, N, F, stype, \gamma, \beta, balytpe$
- **Output**: A bipartition hierarchy (MSC tree) with increasing (decreasing) monotone staircases MIS (MDS) at alternate level

1. if (Root_node || TreeLevel%2 = 0) then
   2. $stype = 1$
   3. end
   4. else
   5. $stype = 0$
   6. end
7. $G_b = \text{ConstructBAG}(B, F, stype)$ /* (see Fig. 3) */
8. Node.cut = MSCut_Bend_BFS($G_b, N, \gamma, \beta, balytpe$)
9. Node.Level = TreeLevel; increment TreeLevel
10. if ($|B_l| \geq 2$) then
11. Node.left = Hier_MSCut_Bend($B_l, N_l, F_l, stype, \gamma, \beta, balytpe$)
12. end
13. if ($|B_r| \geq 2$) then
14. Node.right = Hier_MSCut_Bend($B_r, N_r, F_r, stype, \gamma, \beta, balytpe$)
15. end
16. Return Node.

**Fig. 8.** A floorplan of 17 blocks (a) with monotone increasing/decreasing staircases (MIS/MDS), and (b) a (nearly) balanced bipartition tree (MSC tree [18]) for a ($\gamma, \beta$) pair

**Theorem 3.2.** Given a floorplan with $n$ blocks and $k$ nets, Hier_MSCut_Bend takes $O((n^2+nk) \log n)$ time to generate a hierarchy of minimal bend monotone staircases in it.

**Proof.** Since the block adjacency graph $G_b$ of a given floorplan instance $F$ for $n$ blocks is a planar graph, its construction takes $O(n)$ time. By Lemma 3.1, each while loop in Algori/th 1...
(MSCut\_Bend\_BFS) takes $O(n)$ for identifying $O(n)$ bends and $O(k)$ for net bipartition. Thus, at any recursion level, each call to MSCut\_Bend\_BFS takes $O(n + n^2 + nk)$, i.e., $O(n^2 + nk)$. Since, MSCut\_Bend\_BFS yields a (nearly) balanced bipartition of the (sub)floorplans at each recursion, the number of levels in the bipartition hierarchy (called MSC tree [18]) is $O(\log n)$. Therefore, for the entire bipartition hierarchy of $O(\log n)$ levels, the recursive procedure Hier\_MSCut\_Bend takes $O((n^2+nk)\log n)$ time to identify a set of minimal bend monotone staircases for the entire floorplan $F$. □

In Section 5, we provide a few experimental results to show that the bipartition hierarchy, i.e., MSC tree has $O(\log n)$ height for any floorplan instance $F$ of a circuit containing $n$ blocks with any area distribution.

4 A NEW RANDOMIZED NEIGHBOR SEARCH APPROACH

Given floorplan $F$ for a set of $n$ blocks, the number of all possible monotone staircases in $F$ is exponentially large. Hence, the problem of finding the optimum monotone staircase is known to be NP-Hard [11, 25]. As discussed in Section 3, an (near) optimal solution of monotone staircase bipartition implies a suitable trade-off between the constituent objectives: (a) maximizing the area of each bipartition, (b) minimizing the number of bends in the corresponding monotone staircase, and (c) the number of cut nets by this bipartition. Since the area balanced bipartitioning is an NP-Hard problem [25], no polynomial time algorithm exists. Hence, several greedy heuristic approaches have been proposed in [11, 18, 25, 26] and in Section 3. In all cases, a monotone staircase cut with the maximum Gain value pertaining to a given trade-off among the objectives is considered as an optimal bipartition. As stated in Section 3, we pick an (nearly) optimal monotone staircase among a sequence of $n−1$ monotone staircases for a given $(\gamma, \beta)$ pair. Intuitively, different $(\gamma, \beta)$ pairs may potentially yield different optimal solution(s) and even a different sequence.

Given a set $B$ of $n$ blocks for a given floorplan $F$, a monotone staircase bipartition $(L, R)$ represents a proper subset of $B$. In other words, the blocks in the left partition $L$ (hence the right partition $R = B \setminus L$) constitute a proper subset of $B$, while obeying the monotone staircase property (refer to Lemma 2.1 [26]). Thus, $(L, R)$ represents a valid monotone staircase cut on the block adjacency graph (BAG) for $F$. In summary, the set of all possible monotone staircases $S_m$ in $F$ is a subset of power set of $B$ ($S_m \subseteq \text{power}(B)$). Notably, $S_m$ is a partially ordered set by inclusion $\subseteq$ operation on all the monotone staircases in $F$ that can be identified in exponential time. A staircase $s_m \in S_m$ covers a set of one or more staircases $\{s_n\} \subseteq S_m$ if $s_m$ can be obtained from $\{s_n\}$. Based on this, we construct the corresponding hasse diagram [1] pertaining to $S_m$. An example hasse diagram for a floorplan of $n = 12$ (and $|S_m| = 56$) is illustrated in Fig. 9.

In Section 3 and also in [18], we studied that a sequence of $n − 1$ monotone staircases can be identified greedily at any level of bipartition hierarchy by the respective bipartitioning methods. An optimal solution is identified from this sequence based on a given trade-off $(\gamma, \beta)$. There is a scope of obtaining an improved solution if more than $n−1$ staircases can be explored, with proportionally higher runtime overhead. In this section, we present a new technique for exploring the neighbors of a block (vertex) in the BAG $G_b$ for a potentially better optimal monotone staircase (obeying Lemma 2.1 in terms of the objectives considered. We study how selection of a neighbor is done based on random indexing of the neighbors of vertex $v_i$ in $G_b$. Alike the BFS based method (see Algorithm 1), the proposed bipartitioning method also adopts BFS on $G_b$. However, this method can more aptly resemble with an wave-front propagation in Ether. This may lead to different sequences of (not necessarily disjoint) monotone staircase cuts on the BAG. In Fig. 9 (b), an example of these sequences are highlighted by different paths, one with black and other by blue color, from START to STOP node in the hasse diagram. In this diagram, each node represent a
Exploring the Scope of Unconstrained Via Minimization by Recursive Floorplan Bipartitioning

Fig. 9. A floorplan of 12 blocks (a) overlaid with an optimal monotone staircase \{1,2,3,4,5,9,10\} and a near optimal staircase \{1,2,5,6,9,10\}, and (b) the corresponding Hasse diagram [1] for exponentially large number of sequences (paths) of monotone staircases: the blue path containing \{1,2,3,4,5,9,10\} and black path containing \{1,2,3,4,5,9,10\}.
distinct monotone staircase and edges represent their possible transition to another distinct monotone staircase. In other words, these edges represent the inclusion operation. While the directed search method in Section 3 identifies only one sequence marked by the bold black line in Fig. 9 (b), the randomized method under discussion identifies different sequences during different trials of the proposed randomized neighbor search technique. Notably, the number of sequences obtained by the random method can be more than one, but are not necessarily maximally disjoint. It is also evident that the length of such a path \((START \rightsquigarrow STOP)\) is always \(n - 1\) as stated in Lemma of [18]. However, the number of such paths grow exponentially with \(n\) and the sequences (hence the Hasse diagram) also differ due to different floorplan topology for the same set of blocks \(B\). A comparative study of staircase wave-front propagation using greedy and randomized neighbor search technique is presented in Appendix 7.1.

In Fig. 9 (b), we consider an example of two different sequences of \(n - 1\) monotone staircases marked by the blue and black lines, out of exponentially large number of possible sequences between \(START\) and \(STOP\) nodes. Here \(START\) and \(STOP\) nodes denote trivial monotone staircases containing only one block in the left (right) partition. If one path does not contain an optimal monotone staircase, another path may be explored in a hope to identify an optimal one. Since area balanced monotone staircase bipartitioning is a NP-hard problem, there is no method that verifies such a scenario, unless we apply the brute force method to explore all possible sequences. However, a random transition from one node to another may lead to traversing a new path either completely or partially. Careful study of Fig. 9 (b) shows that randomization at the suitable node, say in this case \(\{1, 2\}\), choosing the block 3 randomly instead of 5 (by greedy approach) may guide to a different sequence leading to a potentially optimal solution \(S_{opt} = \{1, 2, 3, 4, 5, 9, 10\}\), for a given \((\gamma, \beta)\) pair. In summary, several such randomized selections (while traversing from \(START \rightsquigarrow STOP\)) may yield an optimal solution or a scope of obtaining a better solution than the previously found optimal solution. A number of such trials may be exercised in order to explore partially/completely different sequences and thus obtain a potentially better solution. However, in order to contain the run time within the same bound as in Section 3, a large number of such trials cannot be afforded. Instead, we restrict the number of trials to a reasonably small number and use random seeds for each trial. After all such trials, an optimal monotone staircase is identified as the one, with maximum \(Gain\) value, among all the staircases explored along different paths in \(START \rightsquigarrow STOP\).

![Diagram](image-url)

Fig. 10. Exploring the neighbors of \(v_i\) in \(G_b\) based on their indexing.
In the proposed randomized neighbor search method [17], the underlying process of randomly indexing the neighbors of a vertex $v_i$ of $G_b$ brings in the difference with the greedy methods [16, 18]. Unlike greedy indexing approach, from left to right used in [18] and also in Section 3 (see Fig. 10 (a)), a neighbor $v_j$ of $v_i$ with out-degree $p$ is indexed with randomly chosen number $j \in [1, p]$ (see Fig. 10 (b)). As in [18] (also Section 3), identifying a set of monotone staircases while exploring all adjacent vertices of $v_i$ takes $p$ time for exploring all the neighbors. The following lemma shows that the average runtime improves.

**Lemma 4.1.** For a given vertex $v_i$ with out-degree $p$ in $G_b$, the expected time $E[t_{adj}]$ to search its adjacent list to identify one or more distinct monotone staircases is $(p + 1)/2$.

**Proof.** Since all the vertices in the neighborhood of $v_i$ are equally probable to be picked, with a probability of $1/p$, the expected runtime to search a particular neighbor $v_j$ with random indexing $j \in [1, p]$ is:

$$E[t_{adj}] = \sum_{j=1}^{p} \left(\frac{1}{p}\right) . j$$

$$= (p + 1)/2$$

□

Alike the greedy method in [18] and Section 3, the best case scenario occurs when all the $p$ edges emanating from $v_i$ obey the monotone staircase property (Lemma 2.1), thus giving $p$ distinct monotone staircases. The worst case scenario occurs when the number of such edges is only 1, resulting in only one monotone staircase. The following lemma gives the average number of staircases can be explored by a single vertex $v_i$.

**Lemma 4.2.** For a given vertex $v_i$ with out-degree $p$ in $G_b$, $O(p)$ distinct monotone staircases can be identified while obeying Lemma 2.1.

**Proof.** Since, all the $p$ edges emanating from $v_i$ have $1/2$ probability of obeying Lemma 2.1, the average case

$$= 1/p(1 + 2 + ....... + (p - 1) + p)$$

$$= (p + 1)/2$$

Hence, $O(p)$ distinct monotone staircases can be identified. □

### 4.1 The Pseudo-code for the proposed randomized bipartitioner

In this section, we present the pseudo-code for the proposed randomized floorplan bipartitioning method MSCut_Bend_RAND in Algorithm 3, in order to identify a minimal bend monotone staircase in a given floorplan at a given level of bipartition hierarchy. Alike Algorithm 1, this algorithm is called at any level of the bipartitioning hierarchy. The bipartition hierarchy is obtained by the same recursive framework presented in Algorithm 2.

**Lemma 4.3.** The proposed randomized bipartitioning method MSCut_Bend_RAND takes $O(n^2 + nk)$ time for obtaining an optimal monotone staircase with minimal bend count on BAG of a given floorplan $F$.

**Proof.** Since the number of edges $|E_b|$ in $G_b$ is $O(n)$ and $E_b = \sum_{i=1}^{n} p_i$, where $p_i$ is the out-degree of $v_i$, it takes $O(n)$ time for searching distinct monotone staircases. In this method, we use 3 trials in order to obtain a different sequence of monotone staircases in each trial, but possibly not disjoint. Also the net partitioning procedure takes $O(k)$, while finding the number of bends account for $O(n)$ time (see Lemma 3.1). Thus, the overall time taken by the proposed bipartitioning method is $O(n(n + k))$, i.e., $O(n^2 + nk)$. □
input : $G_b, N, \gamma, \beta, \text{baltype}$
output : An optimal monotone staircase for a given $(\gamma, \beta)$ with maximal area balance, minimal net cut and minimal number of bends

1 Define a Queue $Q$, a list $\lambda$ and iterator $r = 0$
2 while ($r < 3$) do
3     Initialize left partition $L = \emptyset$ (right partition $R = V_b \setminus L$)
4     Enqueue the source vertex of $G_b$ in $Q$ as (BFS) level 0 vertex, and include it in $L$
5     Also enqueue $\emptyset$ as BFS level indicator
6     while (NOT EMPTY($Q$)) do
7         Let $v_i$ be the dequeued vertex
8         Define a wavefront $V_{list} = \emptyset$
9         if ($v_i \neq \emptyset$) then
10            $V_{list} \leftarrow \{v_i\}$
11            end
12         else
13            while There exists at least one cut edge in $E_b$ emanating the vertex front $V_{list}$ and terminating on $R$ do
14                Generate a random seed to choose a cut edge $(v_i, v_j)$, such that $v_i \in V_{list}$ and $v_j \in R$
15                if $(v_i, v_j)$ yields a valid ms-cut (see Lemma 2.1) then
16                    Enqueue the vertex $v_j$ and include it in $L$
17                    Mark the edge $(v_i, v_j)$ as explored
18                    Compute the parameters for the $(L, R)$ partition (see Eqn. 1) and store them in a list $\lambda$
19                end
20            end
21            Increment BFS level
22            Enqueue $\emptyset$ as next BFS level indicator
23        end
24    end
25    Increment $r$
26 Return optimal monotone staircase with maximum $Gain C_{max} \in \lambda$

Algorithm 3: MSCut_Bend_RAND

Note that Algorithm 3 has the same $O(n^2 + nk)$ time complexity as Algorithm 1, but only a constant times higher due to multiple trials conducted for obtaining different sequences. In order to obtain a set of optimal monotone staircase cuts with minimal bend count for the entire floorplan, the same recursive bipartitioning framework presented in Algorithm 2 can be used. Therefore, the recursive procedure considering the proposed randomized technique takes $O((n^2 + nk) \log n)$ time to generate a hierarchy of monotone staircase cuts for a given floorplan topology.

5 EXPERIMENTAL RESULTS
In order to verify the correctness and efficiency the proposed bipartitioning methods, we ran them on MCNC/GSRC floorplanning benchmark circuits [3] (see Table 1). Different floorplan instances of a circuit were generated using Parquet floorplacement tool [3, 4] using random seeds. In order to observe different bipartitioning scenarios for the same circuit, we generated four different floorplan instances for each circuit. The algorithms were implemented in C programming language and run on a Linux platform (2.8GHz, 16GB RAM).
Exploring the Scope of Unconstrained Via Minimization by Recursive Floorplan Bipartitioning

Table 1. Floorplanning Benchmarks [3]

| Suite | Circuit | #Blocks | #Nets | Avg. Net Degree |
|-------|---------|---------|-------|-----------------|
| MCNC  | apte    | 9       | 44    | 3.500           |
|       | hp      | 11      | 44    | 3.545           |
|       | xerox   | 10      | 183   | 2.508           |
|       | ami33   | 33      | 84    | 4.154           |
|       | ami49   | 49      | 377   | 2.337           |
| GSRC  | n10     | 10      | 54    | 2.129           |
|       | n30     | 30      | 147   | 2.102           |
|       | n50     | 50      | 320   | 2.112           |
|       | n100    | 100     | 576   | 2.135           |
|       | n200    | 200     | 1274  | 2.138           |
|       | n300    | 300     | 1632  | 2.161           |

5.1 Bipartitioning Results

In our experimental setup, we ran the proposed monotone staircase bipartitioning methods with minimal bends, BFS (see Algorithm 1) and randomized (RAND) version (refer to Algorithm 3) that works in breadth-first traversal (BFS) fashion at any node of the bipartition hierarchy (see Algorithm 2). For experimental purpose, we also came up with a variant of the BFS based greedy method by adopting depth-first search (DFS) on the BAG. Due to lack of space, we are unable to present its pseudo-code. An example showing the working of these bipartitioning methods (BFS, DFS, RAND) is presented in Appendix 7.1.

These experiments were conducted with $\gamma \in [0.1, 0.7]$ and $\beta \in [0.0, 0.3]$, both varying in steps of 0.1 such that $\gamma + \beta \leq 1$. The corresponding bipartitioning results for BFS, DFS, and RAND methods are presented in Fig. 11 for: (a) area balance ratio ($balr$), (b) normalized bend count ($z/z_{max}$), (c) normalized net cut ($k/k_c$), and (d) Gain (see Eqn. 1) respectively. The corresponding values were computed as an average of the respective parameters over the specified ($\gamma, \beta$) pairs and all 4 instances of a given circuit. We compare these results with an earlier BFS based directed search method [18] which did not consider bend minimization (BFS-NB). It is also important to note that the results presented in [18] is for $\gamma = 0.4$ only which is similar to the results for $\gamma = 0.4$ and $\beta = 0.0$ case in BFS mode. Moreover, they did not report the individual objective values in their paper. For fair comparison, we ran their code [18] for obtaining the results for each of the objectives other than Gain in BFS-NB mode, including runtime.

The results on area balance in Fig. 11 (a) show that BFS-NB [18] outperforms all other modes {BFS,DFS,RAND} that used bend minimization objective, by focusing on area balance and net cut only. Among the proposed methods, DFS has the worst area balance values for most of the circuits. For net cut, BFS-NB mode performs well only for a few circuits although the net cut objective has more weight of 0.6 for $\gamma = 0.4$. BFS and RAND have better net cut results for most of the circuits. Likewise, DFS mode continues to give higher net cut values for all the circuits. Regarding the number of bends, RAND mode is consistently better for most of the circuits compared to BFS and DFS. Due to certain floorplan topologies in specific circuits, DFS mode had better average values of bend counts for smaller circuits such as apte, hp, xerox, n10 with around 10 blocks and large circuit n300. Lastly, BFS-NB consistently yielded the worst (highest) bend counts over other modes. Overall, the Gain values reported for each circuit show that BFS-NB is the best for circuits up to n50, followed by RAND mode which dominates the Gain values over BFS and DFS modes for the
remaining circuits. For larger circuits like $n50$ and above, RAND mode is seen to supersede BFS-NB with the maximum $Gain$ values.

Due to balanced partitioning at each node of the bipartition hierarchy (MSC tree [18]), the height of the bipartition (MSC) tree is stated to be $O(\log n)$, where $n$ is the number of blocks in a floorplan. The results presented in Fig. 12 for each circuit shows that the average height of the MSC tree taken over the generated floorplan instances and $(\gamma, \beta)$ values, is contained within the tight bounds of $\log n$ and $2 \log n$, thus establishing the claim in Theorem 3.2.

Table 2 presents the runtime results for the proposed recursive floorplan bipartitioners (BFS, RAND and DFS) as well as [18] (BFS-NB). As stated in Section 4, RAND mode is merely a constant times higher than the other two modes and is more prominent with larger circuits such as $n100$, $n200$ and $n300$, while BFS/DFS report similar runtime for all the circuits. But, none of these methods can match the runtime values obtained by the faster method BFS-NB as claimed by [18] even for the larger circuits.

5.2 Via Count in Early Global Routing by STAIRoute

In this section, we present the experimental results on early via estimation by performing early global routing of the corresponding floorplan level netlist using STAIRoute [15] and the bipartitioning results presented in earlier subsection for BFS, DFS, and RAND modes. A maximum of
8 metal layers were used by STAIRoute using preferred routing directions. We present the corresponding results for the largest benchmark circuit n300 in Fig. 13 and 14 for $\beta \in \{0.0, 0.1, 0.2, 0.3\}$ and $\gamma \in [0.1, 0.7]$ in steps of 0.1. This experimental setup does not apply to BFS-NB mode since the corresponding values of $(\gamma, \beta)$ is not applicable for it. However, our study confirmed that the via count for BFS-NB mode resembles that with BFS mode for $\gamma = 0.4$ and $\beta = 0.0$.

We also study the variation of via count for two different floorplan instances of n300, the best-case instance with smaller HPWL (Instance#1) and the worst-case instance with larger HPWL (Instance#2) in Fig. 13 and 14. In case of instance#1, DFS mode dominates over BFS and RAND modes only for $\beta = 0.0$. However, $\beta > 0$ cases show that RAND mode dominates DFS for upto some $\gamma$ values, such as 0.3, 0.5 and 0.4 respectively, for the respective $\beta \in \{0.1, 0.2, 0.3\}$. Beyond these $\gamma$ values, DFS yields the best via count for this floorplan instance of n300. In a very small range of $\gamma$ and $\beta$ values, i.e., $\beta = 0.3$ and $0.4 < \gamma \leq 0.5$, BFS appears to dominate over DFS and RAND modes.

For the worst case instance, RAND gives smallest via count as compared to other modes for $\beta = 0.0$ and $\gamma \geq 0.4$ for $\beta > 0.0$. As $\beta$ increases, BFS dominates in lower values of $\gamma$, while RAND dominates for the remaining $\gamma$ values with fewer via counts. For all $\gamma$ values and the respective
Fig. 13. Via count vs. $\gamma$ for $n=300$ and $\beta$ values: for Instance #1

$\beta$ values, via count due to DFS mode is almost constant, with some variations near $\gamma$ value of 0.6 and 0.7.

The experiments on all benchmark circuits for different floorplan instances showed that there was no significant variation in routed netlength obtained for BFS, DFS and RAND modes, but are better than that obtained in BFS-NB mode. Due to lack of space, we are not able to put the relevant details obtained by STAIRoute. These netlength values as normalized with respect to no-blockage aware steiner length (computed by FLUTE [10]) ratio and their geometric mean values were obtained as 1.207, 1.201 and 1.208 for BFS, DFS and RAND modes respectively, while BFS-NB mode yields a value of 1.287. Using the approach in [35], the average worst case congestion, defined as the ratio of routing demand and routing capacity, for different floorplan instances of all the circuits in all the modes and for all $(\gamma, \beta)$ pairs, remained 85% ensuring 100% routability, using up to 8 metal layers as per the congestion model proposed in STAIRoute [15]. However, the maximum average congestion [35] in any of the floorplan instances for any mode and $(\gamma, \beta)$ values was seen to be 99%. This shows that no monotone staircase routing region had a congestion over 100% in any routing layer as claimed by [15].
6 CONCLUSION

In this paper, we proposed an early version of unconstrained via minimization in floorplan based early global routing, by a new recursive floorplan bipartitioning framework. This bipartitioning framework identifies, for a given floorplan topology, a set of monotone staircase routing regions with minimal number of bends, by: (a) a greedy method employing BFS/DFS based graph search techniques, and (b) a randomized neighbor search technique for staircase wavefront propagation on BAG of the given floorplan. In this work, we first introduce the bend minimization objective in the multi-objective floorplan bipartitioning problem using monotone staircase cuts and used a pair of trade-off parameters $(\gamma, \beta)$. The solution of this optimization yields a minimal bend monotone staircase routing which impacts the via count during floorplan based early global routing.

Experimental results show the impact of the results of the proposed minimal bend monotone staircase bipartitioning methods on via count during early global routing for varying $(\gamma, \beta)$ pairs and yield fewer via counts. This framework can potentially assess the quality of the floorplan in terms of these via counts.
7 Appendix

7.1 Staircase Wave-front Propagation in a Floorplan

We consider an example of monotone staircase wave-front propagation in a floorplan instance for 9 blocks, as depicted in Fig. 15. In this example, we study how different monotone staircase cuts on BAG can sequentially be obtained by the proposed DFS, BFS and randomized partitioning methods (RAND). This helps in exploring different sequences of monotone staircases with increased solution space for identifying an optimal monotone staircase for a given \((\gamma, \beta)\) pair.

Due to space limitation, only first few steps for identifying a sequence of monotone staircases obtained by BFS/DFS based partitioning are illustrated in Fig. 15 (a) and (b). It shows that both the methods greedily search the neighborhood of a vertex (block) in the BAG \(G_b\) (see Fig. 10 (a)) for propagating the respective wave-fronts. Fig. 15 (c)-(e), illustrates three different trials of Algorithm 3 employing the proposed randomized neighbor search (see Fig. 10 (b)). The trials in RAND yield different wave-front propagation instances, as monotone staircase cuts on the BAG.

It is important to note that BFS/DFS explores a fixed sequence of 9 distinct staircases (see Lemma in [18]) for the same \((\gamma, \beta)\) value, irrespective of the number of trials. On the other hand, RAND yields different sequences during different trials, by the proposed random neighbor indexing of the vertices. It is not necessary for the sequences to be fully disjoint as evident from Fig. 15 (c)-(e). Despite that, an increased solution space of different monotone staircases (a union of all of them obtained during different trials in RAND mode) facilitates us to identify an optimal monotone staircase with minimal number of bends for a given \((\gamma, \beta)\), implied by the maximum Gain value.

7.2 Potential cross-talk minimization

This part discusses potential cross-talk minimization by minimizing the number of cut nets at any level of the partition hierarchy, MSC tree, by suitably choosing \((\gamma, \beta)\) pair, as illustrated in Fig. 16. In this example, we consider two instances of monotone staircases: (a) with more bends and net cut, and (b) with less bend and net cut, as depicted in Fig. 16 (a) and (b) respectively. In the former case, we see that two nets \(a\) and \(b\) are routed through the same monotone staircase routing region (MIS here) using same metal layer and therefore may results in signal cross talk among themselves. The latter case, however, shows that two different staircases are used to route nets \(a\) and \(b\); although net \(b\) partly uses the same staircase (MIS), rest of its routing is done through a different staircase (MDS here). Therefore, both \(a\) and \(b\) will have minimal scope of signal interference between them.

References

[1] [n. d.]. Hasse Diagram, Wikipedia. ([n. d.]). https://en.wikipedia.org/wiki/Hasse_diagram
[2] [n. d.]. Olympus-SoC tool, Mentor Graphics Inc. ([n. d.]). https://www.mentor.com/products/ic\_nanometer\_design/place-route/olympus
[3] [n. d.]. Parquet Floorplanner and MCNC/GSRC Floorplanning Benchmarks. ([n. d.]). https://vlsicad.eecs.umich.edu/BK/parquet
[4] S. N. Adya and I. L. Markov. 2003. Fixed-outline floorplanning: Enabling hierarchical design. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 11, 6 (Dec 2003), 1120–1135. https://doi.org/10.1109/TVLSI.2003.817546
[5] Z. Cao, T. T. Jing, J. Xiong, Y. Hu, Z Feng, L. He, and X. L. Hong. 2008. Fashion: A Fast and Accurate Solution to Global Routing Problem. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 27, 4 (April 2008), 726–737. https://doi.org/10.1109/TCAD.2008.917590
[6] C. C. Chang, J. Cong, Z. Pan, and X. Yuan. 2003. Multilevel global placement with congestion control. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 22, 4 (Apr 2003), 395–409. https://doi.org/10.1109/TCAD.2003.809661
[7] H. Y. Chen and Y. W. Chang. 2009. Routing for manufacturability and reliability. *IEEE Circuits and Systems Magazine* 9, 3 (Third 2009), 20–31. https://doi.org/10.1109/MCAS.2009.933855
[8] H. A. Chien and T. C. Wang. 2014. Redundant-via-aware ECO routing. In 2014 19th Asia and South Pacific Design Automation Conference (ASP-DAC). 418–423. https://doi.org/10.1109/ASPDAC.2014.6742927
Fig. 15. Illustrating initial trail of sequences of monotone staircase wave-fronts: (a) BFS, (b) DFS, and (c) 1st, (d) 2nd, and (e) 3rd trial of RAND

[9] M. Cho, K. Lu, K. Yuan, and D. Z. Pan. 2009. BoxRouter 2.0: A Hybrid and Robust Global Router with Layer Assignment for Routability. *ACM Trans. Des. Autom. Electron. Syst.* 14, 2, Article 32 (April 2009), 21 pages.
Fig. 16. Scope of cross-talk between a pair of nets a and b with respect to minimal bends and cut nets in a monotone staircase with: (a) with more bends and cut nets, and (b) less bends and cut nets.
Exploring the Scope of Unconstrained Via Minimization by Recursive Floorplan Bipartitioning

[23] W. H. Liu, C. K. Koh, and Y. L. Li. 2013. Optimization of placement solutions for routability. In Design Automation Conference (DAC), 2013 50th ACM/EDAC/IEEE. 1–9.

[24] J. Lu and C. W. Sham. 2013. LMgr: A low-Memory global router with dynamic topology update and bending-aware optimum path search. In Quality Electronic Design (ISQED), 2013 14th International Symposium on. 231–238. https://doi.org/10.1109/ISQED.2013.6523615

[25] S. Majumder, S. Sur-Kolay, B. B. Bhattacharya, and S. K. Das. 2007. Hierarchical partitioning of VLSI floorplans by staircases. ACM Trans. Design Autom. Electr. Syst. 12, 1 (2007). https://doi.org/10.1145/1217088.1217095

[26] S. Majumder, S. Sur-Kolay, S. C. Nandy, and B. B. Bhattacharya. 2004. On Finding a Staircase Channel with Minimum Crossing Nets in a VLSI Floorplan. Journal of Circuits, Systems and Computers 13, 05 (2004), 1019–1038. https://doi.org/10.1142/S0218126604001854 arXiv: http://www.worldscientific.com/doi/pdf/10.1142/S0218126604001854

[27] M. Marek-Sadowska. 1984. An Unconstrained Topological Via Minimization Problem for Two-Layer Routing. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 3, 3 (July 1984), 184–190. https://doi.org/10.1109/TCAD.1984.1270074

[28] M. Pan and C. Chu. 2006. FastRoute: A Step to Integrate Global Routing into Placement. In Computer-Aided Design, 2006. ICCAD '06. IEEE/ACM International Conference on. 464–471. https://doi.org/10.1109/ICCAD.2006.320159

[29] M. Pan and C. Chu. 2007. IPR: An Integrated Placement and Routing Algorithm. In Design Automation Conference, 2007. DAC '07. 44th ACM/IEEE. 59–62.

[30] J.A. Roy and I.L. Markov. 2008. High-Performance Routing at the Nanometer Scale. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 27, 6 (June 2008), 1066–1077. https://doi.org/10.1109/TCAD.2008.923255

[31] N. A. Sherwani. 1995. Algorithms for VLSI Physical Design Automation (2nd ed.). Kluwer Academic Publishers, Norwell, MA, USA.

[32] S. Sur-Kolay and B. B. Bhattacharya. 1991. The cycle structure of channel graphs in nonsliceable floorplans and a unified algorithm for feasible routing order. In IEEE International Conference on Computer Design: VLSI in Computers and Processors. 524–527. https://doi.org/10.1109/ICCD.1991.139964

[33] N. Viswanathan and C. Chu. 2005. FastPlace: efficient analytical placement using cell shifting, iterative local refinement, and a hybrid net model. Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 24, 5 (May 2005), 722–733. https://doi.org/10.1109/TCAD.2005.846365

[34] N. Viswanathan, M. Pan, and C. Chu. 2006. FastPlace 2.0: an efficient analytical placer for mixed-mode designs. In Design Automation, 2006. Asia and South Pacific Conference on. 6 pp.–. https://doi.org/10.1109/ASPDAC.2006.1594681

[35] Y. Wei, C. Sze, N. Viswanathan, Z. Li, C.J. Alpert, L. Reddy, A.D. Huber, G.E. Tellez, D. Keller, and S.S. Sapatnekar. 2012. GLARE: Global and local wiring aware routability evaluation. In Design Automation Conference (DAC), 2012 49th ACM/IEEE. 768–773.

[36] Y. Xu, Y. Zhang, and C. Chu. 2009. FastRoute 4.0: Global router with efficient via minimization. In Design Automation Conference, 2009. ASP-DAC 2009. Asia and South Pacific. 576–581. https://doi.org/10.1109/ASPDAC.2009.4796542

[37] Y. Zhang and C. Chu. 2012. GDRouter: Interleaved global routing and detailed routing for ultimate routability. In Design Automation Conference (DAC), 2012 49th ACM/EDAC/IEEE. 597–602.

[38] Y. Zhang, Y. Xu, and C. Chu. 2008. FastRoute3.0: A fast and high quality global router based on virtual capacity. In Computer-Aided Design, 2008. ICCAD 2008. IEEE/ACM International Conference on. 344–349. https://doi.org/10.1109/ICCAD.2008.4681596