Non-tree Routing for Reliability and Yield Improvement*

Andrew B. Kahng, Bao Liu, and Ion I. Măndoiu

CSE and ECE Departments, UCSD, La Jolla, CA 92093-0114
{abk,bliu,mandoiu}@cs.ucsd.edu

Abstract

We propose to introduce redundant interconnects for manufacturing yield and reliability improvement. By introducing redundant interconnects, the potential for open faults is reduced at the cost of increased potential for short faults; overall, manufacturing yield and fault tolerance can be improved. We focus on a post-processing, tree augmentation approach which can be easily integrated in current physical design flows. Our contributions are as follows:

- We formulate the problem as a variant of the classical 2-edge-connectivity augmentation problem in which we take into account such practical issues as wirelength increase budget, routing obstacles, and use of Steiner points.
- We show that an optimum solution can always be found on the Hanan grid defined by the terminals and the corners of the feasible routing region.
- We give a compact integer program formulation which, for up to 100 terminal nets, is solved in practical runtime by the commercial optimization package CPLEX.
- We give a well-scaling greedy algorithm which has practical runtime up to 1,000 terminals, and comes on the average within 1-2% of the optimum computed by CPLEX.
- We give a comprehensive experimental study comparing the solution quality and runtime of our methods with the best reported methods for 2-edge-connectivity augmentation, including a sophisticated heuristic based on minimum-weight branchings [9] and a recent genetic algorithm [14].

Experiments on randomly generated and industry testcases show that our greedy augmentation method achieves significant increase in reliability (as measured by the percentage of biconnected tree edges) with very small increase in wirelength. For example, on 1,000 terminal nets the average percentage of biconnected tree edges is 34.19% for a wirelength increase of only 1%, and 87.73% for a wirelength increase of 20%. SPICE simulations on industry routed nets show that non-tree routing has the additional benefit of reducing maximum sink delay by an average of 28.26% compared to Steiner routing, and by an average of 3.72% compared to timing optimized routing. SPICE simulations further imply that non-tree routing has smaller delay variation due to process variability.

1 Introduction

Ever-decreasing feature sizes allow integration of millions of gates on a single chip. This integration has been enabled in part by low defect density. However, continued reductions in defect density cannot be expected to continue in the near future. Sensitivity of parametric (performance) yield to variability also increases as a result of performance optimization (sizing, etc.) design flows. New design techniques must be applied to improve manufacturing yield of large-area chips, as yield becomes an ever-greater determinant of design viability [2].

In nanometer technologies, likelihood of back-end-of-line (BEOL) defects (i.e., high-resistance via or interconnect defects) has increased relative to that of front-end-of-line (FEOL) defects (i.e., device defects). Interconnects are now more likely to cause circuit malfunction and/or performance or reliability degradation. Two types of catastrophic faults arise due to BEOL defects: open circuit faults or broken interconnects due to loss of mass, and short circuit faults or unintended bridgings between interconnects due to augmentation of mass (Fig. 1). Manufacturing yield is directly affected by the size of the *critical area*, which is the union of all centers of fixed size defects that induce IC faulty behavior. A typical figure of merit that measures the layout's robustness is obtained as the ratio of the total critical area to the layout area [3].

In this paper we propose to introduce layout level redundancy by constructing non-tree interconnect topologies for manufacturing yield and reliability improvement. For easier integration within existing flows we emphasize a *post-processing, tree augmentation* approach, rather than monolithic non-tree routing construction during global or detailed routing. In adding redundant wiring, the extra wire creates more critical area for short faults, but the redundancy makes some wires immune to open defects and thus reduces open-fault critical area. Overall, critical area and manufacturing yield could improve. We observe that:

- The existing tradeoff between short- and open-fault critical area in the BEOL is conducive to the approach we propose. For current design methodologies and manufacturing processes, the probability of failure (POF) due to open defects of any given size is > 3× higher than the POF due to short defects of identical size [3].
- Previous methods that improve manufacturability or reliability by "decompaction" (see, e.g., [2]) will not be as useful in the future due to heavy restrictions on allowed spacings and pitches in nanometer-scale ($\leq$100nm) processes. On the other hand, our approach would work well even with restricted spacings and pitches (but would require incremental detailed routing capability). In fact, it can be speculated that introduction of interconnect redundancy improves uniformity of routing resource utilization by forming "functional fill" as opposed to the present "dummy fill" methodologies.
- Tree augmentation schemes have been previously proposed in the context of clock routing for delay and skew reduction [16] and critical net routing for delay optimization [11]. However, previous algorithms do not work well in our context, since tree augmentation for manufacturability and reliability improvement involves different tradeoffs than tree augmentation for delay or skew optimization.

Our *Manhattan Routing Tree Augmentation (MRTA)* formulation resembles the classical *edge connectivity augmentation problem* [7], which a given subgraph must be augmented at minimum cost into a 2-edge (or more generally k-edge) connected graph.[1] Finding a minimum

*This work was partially supported by Cadence Design Systems, Inc., the MARCO Gigascale Silicon Research Center and NSF Grant CCR-9988331.

[1] A graph is k-edge connected if it cannot be separated by removing less than k edges.

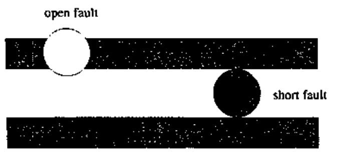

Figure 1: An open fault is formed due to loss of mass; a short fault is formed due to augmentation of mass.

cost k-edge-connected augmentation is NP-hard even for $k = 2$ [4], and much work has been devoted to finding good heuristics and approximation algorithms, see, e.g., [6] and the references therein. The MRTA formulation differs from the 2-edge-connectivity augmentation (E2AUG) problem in several respects:

- While E2AUG is typically formulated for graphs, MRTA has a strong geometric flavor. We consider routing trees embedded in the Manhattan plane, and allow augmenting paths between any two points on the embedded tree (however, augmenting paths must be fully contained in the feasible routing region defined by routing obstacles and design spacing rules). In particular, we allow augmenting paths that are "parallel" to (fragments of) tree edges.
- To ensure the optimal balance between vulnerability to short- and open-faults, our formulation imposes a budget on the total length of augmentation paths and requires maximizing 2-edge connectivity subject to this constraint. In contrast, E2AUG requires 100% 2-edge connectivity regardless of the wirelength increase.
- To enable higher quality MRTA solutions, we allow augmenting paths with one or both ends on other augmenting paths, i.e., at newly created Steiner points (see Figure 2). The existing literature on 2-edge-connectivity augmentation focuses almost exclusively on the spanning subgraph formulation of the problem, in which the use of Steiner points is disallowed.

Our main contributions are as follows:

- We show that an optimum MRTA solution can always be found on the Hanan grid defined by the terminals and the corners of the feasible routing region.
- We give integer program formulations for the MRTA problem with and without Steiner points. The compact integer program for MRTA without Steiner points is solved in practical runtime by the commercial optimization package CPLEX for testcases with up to 100 terminals.
- We give a well-scaling greedy algorithm which has practical runtime up to 1,000 terminals, and comes on the average within 1-2% of the optimum computed by CPLEX. The runtime of our algorithm is $O(ND+N^2K)$), where D is the runtime of Dijkstra's algorithm on the Hanan grid for the terminals and the corners of the feasible routing region, N is the number of Hanan grid vertices on the given routing tree, and K is the number of augmenting paths (typically a small fraction of N). Without routing obstacles the running time reduces to $O(N^2K)$.
- We give a comprehensive experimental study comparing the solution quality and runtime of our methods with the best reported methods for 2-edge-connectivity augmentation, including a sophisticated heuristic based on minimum-weight branchings [9] and a recent genetic algorithm [14].

Experiments on randomly generated and industry testcases show that our greedy augmentation method achieves significant increase in reliability (as measured by the percentage of biconnected tree edges) with very small increase in wirelength. For example, on 1,000 terminal nets the average percentage of biconnected tree edges is 34.19% for a wirelength increase of only 1%, and 87.73% for a wirelength increase of 20%. SPICE simulations on industry routed nets show that non-tree routing has the additional benefit of reducing maximum sink delay by an average of 28.26% compared to Steiner routing, and by an average of 3.72% compared to timing optimized routing. SPICE simulations further imply that non-tree routing has smaller delay variation due to process variability.

The rest of the paper is organized as follows. Section 2 gives the defect model, notations, and the problem formulation. Section 3 gives the reduction to a Hanan grid, integer program formulations, and the greedy MRTA algorithm. Section 4 presents experimental results comparing the solution quality and runtime of our methods with two of the best existing methods for 2-edge-connectivity augmentation, as well as results of SPICE simulations showing that non-tree routing has improved interconnect delay and process variation robustness. Finally, Section 5 gives directions for future research.

2 Problem Formulation

Our problem formulation is based on the following defect model:

- **Uniform defect distribution.** We assume that manufacturing defects are uniformly distributed across the die area. In particular, since we are concerned with routing reliability of large global nets, the uniform defect distribution allows us to ignore defects at the nodes of the routing (which have negligible probability of occurrence), and consider only defects that affect its edges.
- **Single-defect faults.** The occurrence probability of a fault caused by multiple defects is orders of magnitude smaller than the occurrence probability of a fault caused by a single defect. We therefore concentrate on reducing routing vulnerability to single defect faults, and measure routing vulnerability to open faults by the total length of *bridges* (i.e., routing edges whose removal disconnects the net) of the routing.
- **Open faults only.** For current design methodologies and manufacturing processes (e.g., damascene copper electroplating) the prevailing error mechanism is void formation (open faults). As noted in [3], the probability of failure (POF) due to open defects of a given size is $> 3\times$ higher than the POF due to short defects of identical size. In particular, design rule correctness guarantees that there will be no short fault induced by a defect of size smaller than the minimum spacing between interconnects and routing obstacles. In this paper we concentrate on reducing vulnerability to open faults by adding redundant wires. Vulnerability to short faults is maintained within desired limits by imposing an upper-bound on the amount of added wires.

We use the following notations throughout the paper :

1. P = set of terminals for the given net
2. $T(P)$ = given routing tree over terminals of P
3. $p_T(u,v)$ = the unique path in tree T between $u,v \in T$
4. $a(u,v)$ = augmenting path, assumed to be a shortest path between u and v within the feasible routing region (i.e., avoiding the given routing obstacles)
5. A = set of augmenting paths
6. $G = T \cup A$ = augmented routing graph
7. $bridges(G)$ = set of all bridge edges of G; an edge (u,v) is a bridge of G if its removal disconnects G
8. $l(G)$ = total length of routing graph G
9. $l(A)$ = total length of augmenting paths
10. $lbridges(G)$ = total length of bridge edges of G; in our defect model the probability of failure due to open faults for the routing G is proportional to $lbridges(G)$
11. $l_a(u,v)$ = length of augmenting path $a(u,v)$
12. $lbridges_G(u,v) = l(p_T(u,v) \cap bridges(G))$ = total length of bridge edges on path $p_T(u,v)$
13. FRR = rectilinear feasible routing region[2]

[2]The feasible routing region FRR is formed by enlarging the neighboring wires and routing obstacles by the minimum design spacing rules. The initial routing T as well as the augmenting paths A must be within the FRR to guarantee design rule correctness.

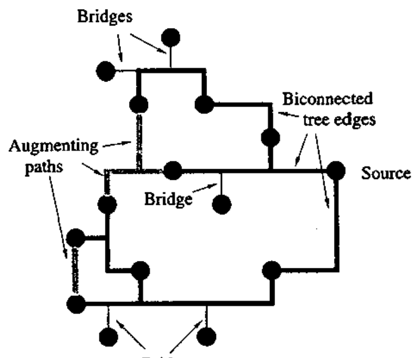

Figure 2: Initial routing tree T and a set of augmenting paths. Remaining bridges are shown as thin lines.

14. F = set of corner vertices of the rectilinear feasible routing region FRR
15. $H(P \cup F)$ = Hanan grid over the points in $P \cup F$, obtained by taking the union of vertical and horizontal lines through the points
16. $HV(P \cup F)$ = the set of vertices of $H(P \cup F)$
17. $N = |T \cap HV(P \cup F)|$ = number of Hanan grid points on the routing tree T

The problem of maximally increasing the reliability to open faults by adding a bounded amount of wire redundancy to an already routed net is formulated as follows:

Manhattan Routing Tree Augmentation (MRTA) Problem
Given:

1. Rectilinear feasible routing region FRR,
2. Rectilinear Steiner routing tree T within FRR, and
3. Wirelength budget W.

Find: Set of augmenting paths A within the FRR such that:

(a) Total length of augmenting paths is at most W, i.e., $l(A) \leq W$, and
(b) Total length $l(T) - lbridges(G)$ of edges of T which are non-bridges in $G = T \cup A$ is maximum.

3 Exact and Heuristic Algorithms for the MRTA Problem

We begin this section by showing that, despite the seemingly continuous solution space (due to the flexibility in choosing endpoints of augmenting paths), an optimum solution can always be found on the Hanan grid defined by the terminals and the corners of the feasible routing region. Based on this result, in Section 3.2 we give a compact integer linear program (ILP) formulation for the MRTA problem in which Steiner points are disallowed, and an ILP with exponentially many constraints for the MRTA problem with Steiner points. Finally, in Section 3.3 we describe an efficient greedy MRTA heuristic.

3.1 Reduction to Hanan Grid

Theorem 1 *There exists an optimum MRTA solution with all augmenting paths embedded on the Hanan grid defined by terminals and corners of the rectilinear feasible routing region.*

Proof. If both ends of an augmenting path are Hanan grid vertices then clearly the whole augmenting path (which is a shortest path within the given rectilinear feasible routing region) can be routed along the Hanan grid. Assume that an optimum MRTA solution $G = T \cup A$ has a non-Hanan augmenting path a ending at a point p which is not a Hanan grid vertex. Let l and r be the vertical Hanan grid lines immediately to the left and right of p, and let h be the horizontal line through p. There are two cases to consider:

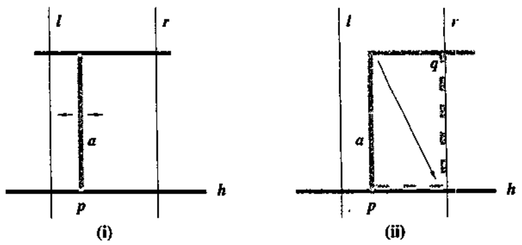

Figure 3: (i) If augmenting path a connects two parallel edges, then $l(G)$ remains constant when a slides horizontally, and for at least one direction $lbridges(G)$ does not increase. (ii) Otherwise, a can be re-embedded along the Hanan grid lines.

(i) If the augmenting path a has the other end strictly between l and r on a horizontal edge of G, then, by optimality, it follows that a is a straight line segment. Sliding a horizontally does not change the total length $l(G)$, and there exists a horizontal sliding direction (either sliding towards l or sliding towards r) which does not increase the bridge length $lbridges(G)$. By sliding a in this direction we obtain another optimum solution with strictly shorter augmenting path length not embedded on the Hanan grid (Fig. 3(i)).

(ii) Otherwise, let q be the first point, starting from p, where a crosses again $l \cup h \cup r$ (such a point must exist, since every augmenting path creates a cycle with the rest of G). The portion of a between p and q can be re-embedded as an L shape along the Hanan grid (Fig. 3(ii)). Again, this gives an optimum solution with strictly shorter augmenting path length not embedded on the Hanan grid.

The theorem follows by repeating the above transformations (and the symmetrical vertical transformations) until all augmenting edges are completely embedded on the grid. □

Remark. Notice that Theorem 1 does not guarantee that all augmenting path endpoints are Hanan grid vertices. It is easy to see that, due to the wirelength budget, endpoints which are not Hanan grid vertices may be required. However, Theorem 1 implies that this happens only when the augmenting path is parallel to an edge of the augmented tree T.

3.2 Integer Program Formulations

Theorem 1 implies that the optimum MRTA can be found by considering only augmenting paths which are shortest paths between arbitrary vertices on the Hanan grid $H(P \cup F)$, plus, possibly, augmenting paths which are parallel to (partial) tree edges.

We first give the ILP formulation for MRTA without Steiner points. In this version of the problem augmenting paths $a(u,v)$ which are not parallel to tree edges connect points $u, v \in T \cap HV(P \cup F)$. To formulate the MRTA problem as an integer program, we first assign to each pair of vertices $u, v \in T \cap HV(P \cup F)$ a 0/1 variable $x_{u,v}$ indicating whether or not $a(u,v)$ is an augmenting path, i.e.,

$$x_{u,v} = \begin{cases} 1 & : \quad a(u,v) \in A \\ 0 & : \quad a(u,v) \notin A \end{cases} \tag{1}$$

Also, we assign to each edge $(u,v) \in T$ a 0/1 variable $y_{u,v}$ indicating whether or not (u,v) is contained in a cycle of $T \cup A$, i.e.,

$$y_{u,v} = \begin{cases} 1 & : \quad (u,v) \notin bridges(G) \\ 0 & : \quad (u,v) \in bridges(G). \end{cases} \tag{2}$$

Removing any edge $(u,v) \in T$ separates T into two subtrees, T_u and T_v, such that $T = T_u \cup T_v \cup (u,v)$. Since Steiner points are disallowed, it follows that (u,v) is contained in a cycle of $T \cup A$ if and only if A contains an augmenting path between T_u and T_v, i.e., iff there exist $i \in T_u$ and $j \in T_v$ such that $a(i,j) \in A$. Using this observation, we can reformulate MRTA as the following integer program:

$$\text{maximize} \sum_{(u,v)\in T} y_{u,v} l(u,v) + \left(W - \sum_{u,v\in T\cap HV(P\cup F)} x_{u,v} l_a(u,v) \right) \quad (3)$$

such that

$$\begin{aligned}
\sum_{u,v\in T\cap HV(P\cup F)} x_{u,v} l_a(u,v) &\leq W \\
\sum_{i\in T_u, j\in T_v} x_{i,j} &\geq y_{u,v}, \forall (u,v)\in T \\
x_{u,v} &\in \{0,1\}, \forall u,v \in T\cap HV(P\cup F) \\
y_{u,v} &\in \{0,1\}, \forall (u,v)\in T
\end{aligned}$$

where $l(u,v)$ is the length of edge $(u,v)\in T$ and $l_a(u,v)$ is the length of the augmenting path between u and v.

In ILP (3), the first constraint enforces the wirelength budget, while the following constraints ensure that only edges of T that are biconnected are counted in the objective function. Augmenting paths parallel to (fragments of) tree edges are handled implicitly by the addition to the objective function of the term $W - \sum_{u,v\in T\cap HV(P\cup F)} x_{u,v} l_a(u,v)$, which represents the wirelength budget left unused after adding "regular" augmenting paths.

Similar to ILP (3), the ILP formulation for MRTA with Steiner points uses 0/1 variables $y_{u,v}$ indicating whether or not $(u,v)\in T$ are biconnected. In order to capture the possible use of Steiner points as ends of augmenting paths, we now need a 0/1 variable x_e for each edge e in the Hanan grid $H(P\cup F)$. The ILP sets x_e to 1 if any augmenting path uses e, and to 0 otherwise. The ILP formulation for MRTA with Steiner points is:

$$\text{maximize} \sum_{(u,v)\in T} y_{u,v} l(u,v) + \left(W - \sum_{e\in H(P\cup F)} x_e l(e) \right) \quad (4)$$

such that

$$\begin{aligned}
\sum_{e\in H(P\cup F)} x_e l(e) &\leq W \\
\sum_{e\in X} x_e &\geq y_{u,v}, \forall (u,v)\in T, X\in X_{u,v} \\
x_e &\in \{0,1\}, \forall e\in H(P\cup F) \\
y_{u,v} &\in \{0,1\}, \forall (u,v)\in T
\end{aligned}$$

where $l(u,v)$ is the length of edge $(u,v)\in T$, $l(e)$ is the length of Hanan grid edge e, and, for every $(u,v)\in T$, $X_{u,v}$ is the set of all Hanan grid cuts separating the two connected components, T_u and T_v, of $T\setminus(u,v)$.

The first constraint of (4) is again enforcing the wirelength budget. Ensuring that $y_{u,v}$ is set to 1 only if edge $(u,v)\in T$ is biconnected requires now an exponential number of constraints. The formulation is based on the Max-Flow Min-Cut theorem, which guarantees that there exists a path between T_u and T_v consisting solely of edges e with x_e set to 1 if every cut separating T_u from T_v contains at least one such edge. Finally, augmenting paths parallel to tree edges are handled using the same method as in (3).

We note that, despite its exponential size, the fractional relaxation of (4) can be solved, e.g., using the Ellipsoid algorithm [5] with a separation oracle that runs a min-cut algorithm for each $(u,v)\in T$ to check feasibility of any given solution.

3.3 The Greedy MRTA Algorithm

In this section we propose a greedy algorithm for the MRTA problem. Our algorithm (see Algorithm 1) iteratively adds an augmenting path $a(u,v)$ that maximizes the ratio $lbridges_G(u,v)/l_a(u,v)$ between the length of bridge edges between u and v and the length of the augmenting path. In every step the algorithm considers only augmenting paths $a(u,v)$ that fit within the remaining wirelength budget and have $lbridges_G(u,v)/l_a(u,v) \geq 1$ (since otherwise it is better to simply use augmenting paths parallel to tree edges).

In order to efficiently compute the best augmenting path in each greedy iteration, we precompute the length of all shortest augmenting paths $a(u,v)$ by running Dijkstra's algorithm with each $u\in T\cap HV(P\cup F)$ as the source. Further, we compute bridge edge lengths $lbridges_G(u,v)$ by executing one depth-first search traversal of T for each $u\in T\cap HV(P\cup F)$. Whenever an augmenting path a is added to G, we update the set of possible augmenting path endpoints to include Hanan vertices on a, and compute the lengths of all shortest augmenting paths originating at these points with $|a\cap HV(P\cup F)|$ more runs of Dijkstra's algorithm. It can be checked that the number of possible augmenting path endpoints does not exceed $2N$ throughout the algorithm, where $N = |T\cap HV(P\cup F)|$. Thus, with this implementation, the greedy algorithm runs in $O(ND + N^2K))$ time, where D is the runtime of Dijkstra's algorithm on $H(P\cup F)$ and K is the number of augmenting paths (typically a small fraction of N). Without routing obstacles Dijkstra's algorithm becomes unnecessary since $l_a(u,v)$ is given by the rectilinear distance between u and v. In this case the greedy algorithm runs in $O(N^2K)$ time.

Algorithm 1: Greedy MRTA Algorithm

Input: Rectilinear feasible routing region FRR with corners F, routing tree T for P within FRR, wirelength budget W
Output: Set of augmenting paths A with $l(A) \leq W$

```
1. $bridges(G) = G = T, A = \emptyset, V = T\cap HV(P\cup F)$
2. For each $u\in V$, compute the lengths $l_a(u,v)$ of the shortest paths from $u$ to each
   $v\in V$ by running Dijkstra's algorithm with $u$ as the source
3. For each node $u\in V$, compute the length $lbridges_G(u,v)$ of the bridges between $u$
   and each $v\in V$ by a depth first search traversal of $T$ with $u$ as the source
4. Find, among paths $a(u,v)$ with $l(A) + l_a(u,v) \leq W$, an augmenting path $a(u^*,v^*)$
   maximizing $lbridges_G(u,v)/l_a(u,v)$
5. If $lbridges_G(u^*,v^*)/l_a(u^*,v^*) \geq 1$ then
        $A = A\cup a(u^*,v^*)$,    $G = G\cup a(u^*,v^*)$
        $bridges(G) = bridges(G)\setminus p_T(u^*,v^*)$
        $V = V\cup(a(u^*,v^*)\cap HV(P\cup F))$
        For each $u\in a(u^*,v^*)\cap HV(P\cup F)$, compute the lengths $l_a(u,v)$ of the
        shortest paths from $u$ to each $v\in V$ by running Dijkstra's algorithm with $u$
        as the source
        Go to Step 3
6. Else Exit
```

4 Experimental Results

We compare our integer program and the greedy MRTA algorithm with two existing algorithms:

- A greedy best-drop heuristic in [9] which selects augmentation edges by finding minimum-weight branching in an appropriately defined directed graph. The best-drop heuristic is reported to find high-quality solutions for a host of connectivity problems, including 2-edge-connectivity augmentation with no wirelength budget.[3]
- A genetic algorithm enhanced by using a compact edge set representation, problem specific variation operators and a stochastic local improvement algorithm to reduce solution space [14].

The integer program (3) was solved using the CPLEX 7.0 MIP optimizer, the other three algorithms were implemented in C/C++ and compiled with `g++` version `2.95`. All experiments were conducted on a SUN SPARC Ultra-10 workstation with 256MB memory. For each instance size $n\in\{5,10,50,100,500,1000\}$ we generated 100 instances uniformly at random from a $10,000\times 10,000$ grid. For each instance, a Steiner minimum tree is constructed using the ER heuristic [1] and then augmented (assuming no routing obstacles) by the four compared algorithms. Table 1 shows the statistics on the number of nodes (including wire turns), number of leaves, and total wirelength of the initial trees.

We implement three versions of the greedy MRTA algorithm:

[3]We have modified the code in [9] to enforce a specified wirelength budget.

#Sinks	#Nodes	#Leaves	Length
5	9.09	3.60	14801.50
10	19.68	6.32	23926.63
20	40.63	10.85	34660.11
50	103.77	25.38	54305.56
100	207.99	50.17	75806.31
200	417.24	98.19	106486.30
500	1045.48	241.68	167176.87
1000	2087.66	480.93	234839.26

Table 1: Initial routing statistics (averages over 100 random instances of each size).

(a) Considering all augmenting paths $a(u,v)$ for which u,v are either terminals, Steiner points, or corners of edges of T.

(b) Considering all augmenting paths $a(u,v)$ for which $u,v \in T \cap HV(P \cup F)$, i.e., all paths in (a) plus paths with one or both ends at the projection of a terminal on an edge of T.

(c) Considering all augmenting paths $a(u,v)$ for which $u,v \in (T \cup A) \cap VH(P \cup F)$, i.e., all paths in (b) plus paths with one or both ends at the projection of a terminal on an already added augmenting path.

We also implemented versions (a) and (b) for the Best-Drop and ILP algorithms. The (c) version of the greedy MRTA algorithm gives almost identical results to the (b) version in experiments with $1-20\%$ wirelength budget and $5-20$ terminals, and we omit its results.

Table 2 gives the number of augmenting paths, percentage of biconnected tree edges, and runtime for versions (a) and (b) of the greedy MRTA, Best-Drop, and ILP algorithms. The results show that versions (b) achieve better solution quality than versions (a); for the greedy MRTA algorithm version (b) is better than version (a) by as much as 18.54%. Table 3 gives the results obtained by the greedy MRTA, Best-Drop, Genetic, and ILP algorithms when there is no restriction on the added wirelength.

The results show that the greedy MRTA algorithm is the fastest of the compared algorithms, scaling up to 1,000 sinks. For $1-20\%$ wirelength budgets the greedy MRTA algorithm is also outperforming the other heuristics in solution quality, finding solutions within 1-2% of the optimum computed by CPLEX for all wirelength budgets. The much slower Best-Drop and Genetic heuristics outperform greedy MRTA only for unlimited wirelength budget and small number of sinks, and then by a very small amount.

As expected, MRTA biconnectivity increases with the wirelength budget, e.g., it increases from 34.19% under 1% wirelength budget to 87.73% under 20% wirelength budget for 1000 sink instances. Interestingly, the biconnectivity also increases significantly with the number of sinks, e.g., from 1.12% for 5 sinks to 34.19% for 1000 sinks for 1% wirelength budget. Table 4 shows statistics for the first augmenting path added by greedy MRTA. This path has a ratio between biconnected length and wirelength increase as large as 80 for 1,000 sinks, and already achieves a significant improvement in routing reliability at a very low wirelength increase cost.

The impact of non-tree augmentation on maximum delay and delay variation due to process variability was verified by running SPICE simulation on two sets of 14 instances each. The first set consisted of non-critical nets extracted from a recent industry design and routed by Cadence WarpRouter using minimum-area optimization, while the second set consisted of randomly generated nets routed using the timing-driven P-Tree algorithm [10] with buffer insertion disabled and identical sink required-arrival times. Each interconnect was represented by a Π model and driven by a $1.8V$ voltage source with a ramped input signal of $150ps$ slew time. 50% delay from the source to each sink was simulated based on $180nm$ ITRS predictive technology model beta version [8] with the following parameters: unit wire resistance $r = 0.040\Omega/\mu m$, unit wire capacitance $c = 0.259fF/\mu m$, sink capacitance $c_t = 63.358fF$ and source driving resistance $R_b = 139.434\Omega$. In computing robustness to process variation we assumed 100% wire width correlation. This models systematic variation sources such as lens aberrations which cover 5-10mm ranges [13, 12], i.e., ranges that are larger than those covered today by unbuffered interconnect. We uniformly varied wire width by

WL Budget	#SINK	%WL INC	%BI CONN	RATIO	CPU SEC
1%	5	0.22	0.34	1.54545	0.00
	10	0.51	0.58	1.13725	0.00
	50	0.55	1.34	2.43636	0.02
	100	0.36	1.32	3.66667	0.07
	500	0.37	17.27	46.6757	1.81
	1000	0.18	14.32	79.5556	7.19
2%	5	0.69	0.81	1.17391	0.00
	10	1.13	1.34	1.18584	0.00
	50	1.06	3.13	2.95283	0.02
	100	1.48	16.34	11.0405	0.07
	500	0.37	17.27	46.6757	1.82
	1000	0.18	14.32	79.5556	7.25
5%	5	2.56	3.12	1.21875	0.00
	10	2.72	4.04	1.48529	0.00
	50	3.64	30.27	8.31593	0.02
	100	2.10	30.47	14.5095	0.07
	500	0.37	17.27	46.6757	1.85
	1000	0.18	14.32	79.5556	7.33
10%	5	4.89	6.18	1.2638	0.00
	10	5.69	9.74	1.71178	0.00
	50	4.09	36.20	8.85086	0.02
	100	2.10	30.47	14.5095	0.07
	500	0.37	17.27	46.6757	1.85
	1000	0.18	14.32	79.5556	7.34
20%	5	11.86	15.70	1.32378	0.00
	10	11.15	27.52	2.46816	0.00
	50	4.09	36.20	8.85086	0.02
	100	2.10	30.47	14.5095	0.07
	500	0.37	17.27	46.6757	1.85
	1000	0.18	14.32	79.5556	7.33
100%	5	33.70	53.78	1.59585	0.00
	10	16.76	46.93	2.80012	0.00
	50	4.09	36.20	8.85086	0.02
	100	2.10	30.47	14.5095	0.07
	500	0.37	17.27	46.6757	1.86
	1000	0.18	14.32	79.5556	7.33

Table 4: Wirelength increase, percentage of biconnected tree edges, $lbridges_G(u,v)/l_a(u,v)$ ratio, and CPU runtime due to first MRTA augmentation path (averages over 100 random instances).

$dw = 6.67\%$ from the nominal value, and computed unit length wire capacitance using the formulas in [15] for parallel lines between two planes, including area, fringe, and coupling capacitances. The maximum 50% sink delay and its variation in percents are reported for the two sets of test instances in Tables 5 and 6. In these tables, the results under 0% wirelength budget correspond to the initial (area, respectively timing-optimized) routing trees.

Results for non-critical nets (Tables 5) show that non-tree augmentation continuously reduces maximum source-to-sink delays in most of the instances in our experiments (except *net*10). An average of 28.26%, and maximum of 62.15% delay reduction can be achieved (for *net*12) with 20% wirelength budget. Non-tree augmentation also decreases process variation effect in most non-critical instances (except *net*4). An average of 13.79%, and maximum of 28.86% delay variation reduction is observed (for *net*12) when comparing nominal wire width w and $w-dw$ wire width. Results for timing-optimized interconnect trees (Table 6) show that non-tree augmentation still decreases the maximum source-to-sink delay by an average of 3.72% and a maximum of 39.04%. However, in some instances maximum source-to-sink delay can increase by as much as 6.47% due to non-tree augmentation. Non-tree augmentation decreases process variation in all the timing-optimized instances, with an average of 3.24%, maximum of 12.17% and minimum of 0.09%.

An explanation of the above results is that non-tree augmenting paths can decrease interconnect source-to-sink delay by forming shorter connections, but can also increase interconnect delay due to increased capacitance. The probability for non-tree augmentation to form a shorter connection between the source and a critical sink is smaller in timing-optimized interconnect, which results in smaller improvements in maximum delay and delay variability. In general our non-tree augmentation scheme achieves better improvements in interconnect delay and variability for non-critical, area-optimized interconnects.

WL Incr.	#Sinks	Greedy(a)			Greedy(b)			Best-Drop(a)			Best-Drop(b)			ILP(a)			ILP(b)		
		#AUG EDGE	%BI CONN	CPU SEC	#AUG EDGE	%BI CONN	CPU SEC	#AUG EDGE	%BI CONN	CPU SEC	#AUG EDGE	%BI CONN	CPU SEC	#AUG EDGE	%BI CONN	CPU SEC	#AUG EDGE	%BI CONN	CPU SEC
	5	0.45	1.12	0.00	0.63	2.12	0.00	0.45	1.12	0.04	0.51	1.12	0.09	0.01	1.12	0.00	0.15	2.12	0.00
	10	1.66	1.07	0.00	2.76	1.85	0.00	1.56	1.01	2.88	2.03	1.01	27.71	0.06	1.07	0.00	0.40	1.85	0.00
	20	2.82	1.31	0.01	4.74	2.18	0.05	2.76	1.07	102.93	5.00	2.04	6976.65	0.28	1.32	0.01	1.25	2.19	0.18
	50	4.75	2.01	0.08	7.54	2.86	1.23	-	-	-	-	-	-	1.51	2.04	0.38	-	-	-
1%	100	6.04	2.79	0.42	9.46	3.84	11.59	-	-	-	-	-	-	2.43	2.86	2.97	-	-	-
	200	5.69	17.73	1.59	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	500	8.59	29.57	15.08	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	1000	11.26	34.19	78.12	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	5	0.79	2.12	0.00	1.07	3.14	0.00	0.76	2.12	0.07	0.90	2.31	0.16	0.01	2.12	0.00	0.16	3.14	0.00
	10	2.20	2.21	0.00	3.33	3.29	0.00	2.12	2.04	3.96	2.67	2.04	37.70	0.13	2.21	0.00	0.58	3.29	0.00
	20	3.42	2.92	0.01	5.16	4.26	0.06	3.46	2.14	129.56	4.00	3.64	5843.05	0.63	2.93	0.02	1.79	4.28	0.25
	50	5.45	4.81	0.09	7.85	6.11	1.27	-	-	-	-	-	-	1.94	4.91	0.42	-	-	-
2%	100	5.61	19.49	0.38	9.24	21.18	11.23	-	-	-	-	-	-	2.00	19.57	3.10	-	-	-
	200	7.15	33.77	1.97	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	500	10.23	42.34	17.79	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	1000	16.28	45.14	112.15	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	5	1.48	5.56	0.00	1.78	6.31	0.00	1.49	5.09	0.14	1.65	5.32	0.30	0.06	5.56	0.00	0.20	6.31	0.00
	10	2.89	6.35	0.00	3.96	8.29	0.00	2.99	5.37	5.40	3.46	5.52	48.37	0.41	6.36	0.00	1.10	8.32	0.0
	20	4.07	9.25	0.01	6.25	11.69	0.07	4.56	6.36	163.72	7.00	8.00	6691.95	1.42	9.56	0.03	2.37	12.10	0.29
	50	5.48	34.17	0.09	8.37	35.55	1.32	-	-	-	-	-	-	1.97	34.95	0.46	-	-	-
5%	100	7.23	46.95	0.48	11.48	48.16	13.67	-	-	-	-	-	-	3.35	47.90	3.52	-	-	-
	200	10.42	55.74	2.82	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	500	18.28	59.81	31.54	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	1000	31.91	61.29	218.11	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	5	2.12	11.30	0.00	2.31	12.24	0.00	2.11	9.89	0.20	2.26	10.44	0.41	0.13	11.30	0.00	0.31	12.30	0.00
	10	3.58	14.41	0.00	4.48	17.95	0.00	3.92	10.84	7.11	4.48	12.04	61.73	0.91	14.46	0.01	1.67	18.11	0.02
	20	4.26	30.53	0.01	5.98	35.56	0.06	4.98	23.68	173.12	6.00	28.25	5516.75	1.52	31.15	0.04	2.49	36.32	0.33
	50	6.60	56.95	0.11	9.76	58.58	1.51	-	-	-	-	-	-	2.96	58.04	0.58	-	-	-
10%	100	10.05	65.87	0.66	14.22	66.76	16.69	-	-	-	-	-	-	5.55	67.00	5.81	-	-	-
	200	15.99	71.33	4.28	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	500	33.28	73.56	56.77	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	1000	62.14	74.49	420.91	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	5	2.30	23.90	0.00	2.66	27.21	0.00	2.58	21.41	0.24	2.79	23.61	0.50	0.36	23.90	0.00	0.66	27.20	0.00
	10	3.70	38.13	0.00	5.07	43.96	0.01	4.11	32.87	7.19	4.58	35.97	58.41	1.32	39.01	0.01	1.93	45.10	0.03
	20	5.45	64.94	0.01	7.49	66.92	0.08	6.21	58.30	195.86	6.00	61.11	5879.08	2.15	66.64	0.05	3.36	68.54	0.47
	50	10.12	77.96	0.16	37.94	79.13	5.48	-	-	-	-	-	-	5.10	79.56	0.93	-	-	-
20%	100	17.05	83.29	1.10	123.99	83.88	133.99	-	-	-	-	-	-	10.56	84.44	8.62	-	-	-
	200	34.08	85.86	8.98	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	500	82.16	87.15	137.62	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	1000	164.74	87.73	1093.20	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 2: Comparison of greedy MRTA, Best-Drop, and CPLEX ILP (all results are averages over 100 random instances). (a) versions use only terminals or Steiner points of T as endpoints of augmentation paths, (b) versions can use all Hanan grid vertices that are on tree edges.

WL Budget	#SINKS	Greedy(a)			Best-Drop(a)			Genetic(a)			ILP(a)		
		#AUG EDGE	%WL INC	CPU SEC	#AUG EDGE	%WL INC	CPU SEC	#AUG EDGE	%WL INC	CPU SEC	#AUG EDGE	%WL INC	CPU SEC
	5	3.76	78.33	0.00	3.35	77.90	0.26	3.52	77.86	0.01	1.07	77.86	0.00
	10	6.46	59.70	0.00	5.97	58.19	7.54	6.10	57.97	0.52	1.79	57.97	0.01
	20	10.93	46.82	0.03	10.25	45.77	226.71	9.81	45.59	4.43	3.21	45.58	0.07
	50	26.20	40.16	0.41	-	-	-	20.85	39.44	37.35	7.33	38.64	0.98
100%	100	51.10	36.27	3.26	-	-	-	39.48	39.30	181.08	13.30	35.11	8.21
	200	100.49	34.10	26.12	-	-	-	-	-	-	-	-	-
	500	245.63	32.85	405.45	-	-	-	-	-	-	-	-	-
	1000	484.47	32.27	3160.24	-	-	-	-	-	-	-	-	-

Table 3: Comparison of greedy MRTA, Best-Drop, Genetic, and ILP algorithms for unlimited wirelength budget (averages over 100 random instances).

WLB	0%	1%	5%	20%
net1	1551.1 ± 4.13%	1564.3	1478.9	873.3 ± 3.78%
net2	366.3 ± 3.55%	374.7	327.2	345.1 ± 2.89%
net3	859.6 ± 3.96%	869.4	836.3	627.2 ± 3.67%
net4	282.9 ± 2.84%	282.5	306.7	262.9 ± 3.05%
net5	1002.0 ± 3.79%	1002.7	971.2	778.0 ± 3.47%
net6	787.5 ± 3.81%	794.3	520.8	442.8 ± 3.17%
net7	514.6 ± 3.50%	514.1	318.8	273.5 ± 2.93%
net8	235.2 ± 2.98%	236.5	228.3	185.2 ± 2.70%
net9	1602.9 ± 3.99%	1593.0	1633.4	1359.5 ± 3.83%
net10	888.3 ± 3.72%	873.4	889.1	944.7 ± 3.60%
net11	420.1 ± 3.81%	416.3	249.6	219.1 ± 2.28%
net12	642.8 ± 4.05%	648.0	605.4	243.6 ± 2.88%
net13	426.6 ± 3.76%	415.9	409.2	402.6 ± 3.48%
net14	562.5 ± 3.91%	558.4	545.5	263.4 ± 2.66%

Table 5: Maximum SPICE sink delays (*ns*) and delay variations (percents) under $dw = 6.67\%$ wire width variation for 14 nets with 52–56 sinks extracted from an industry design. Initial trees are constructed using Cadence WarpRouter using minimum-area optimization.

WLB	0%	1%	5%	20%
net1	495.6 ± 1.20%	498.5	502.8	454.2 ± 1.18%
net2	179.9 ± 0.98%	177.9	179.3	187.3 ± 0.98%
net3	298.9 ± 1.11%	290.8	292.9	293.2 ± 1.10%
net4	85.0 ± 0.70%	84.3	83.8	85.7 ± 0.66%
net5	492.3 ± 1.15%	502.7	491.9	524.2 ± 1.14%
net6	577.8 ± 1.20%	545.1	438.8	352.2 ± 1.05%
net7	259.6 ± 1.01%	257.2	257.4	254.7 ± 0.99%
net8	127.9 ± 0.82%	128.7	129.8	134.5 ± 0.82%
net9	499.9 ± 0.98%	463.5	465.0	422.5 ± 0.89%
net10	415.7 ± 1.03%	409.5	414.7	420.4 ± 1.02%
net11	69.3 ± 0.85%	69.8	66.3	66.1 ± 0.80%
net12	121.9 ± 1.14%	122.9	124.5	123.8 ± 1.12%
net13	196.4 ± 1.15%	197.9	199.7	203.7 ± 1.14%
net14	221.2 ± 1.01%	210.3	209.8	209.0 ± 0.97%

Table 6: Maximum SPICE sink delays (*ns*) and delay variations (percents) under $dw = 6.67\%$ wire width variation for 14 randomly generated nets with 15 sinks each. Initial trees are constructed using the P-Tree algorithm [10] with identical sink required-arrival times.

5 Conclusions and Future Work

In this paper we have proposed the introduction of redundant intercon nect as a post-routing optimization for manufacturing yield, reliability and process variation robustness improvement. We have formulated the problem as a variant of the classic NP-hard 2-edge connectivity augmen tation (in a Manhattan plane and under a given wirelength budget) and proposed both practical integer program formulations and a well-scaling greedy algorithm which comes within 1-2% of the optimum on the aver age. We have presented experimental results on both randomly generated and industry testcases with up to 1,000 terminals showing that:

- Our methods outperform best known 2-edge connectivity augmen tation algorithms in both solution quality and runtime for the prac tically relevant wirelength budgets.
- Significant increase in reliability (as measured by the percentage of biconnected tree edges) and simultaneous decreases in maximum sink delay and delay variation due to process variability can be achieved with very small increases in wirelength.

Our ongoing research efforts include algorithms for biconnectivity augmentation of buffered trees, simultaneous augmentation of multiple routed nets, and chip-level evaluation of the proposed augmentation al gorithms.

References

[1] M. Borah, R. M. Owens, and M. J. Irwin. A fast and simple Steiner routing heuristic. *Discrete Applied Mathematics*, 90:51–67, 1999.

[2] V.K.R. Chiluvuri. Yield optimization in physical design: a review. In *Proceedings of the Fifth ACM/SIGDA Physical Design Workshop*, pages 198–206, 1996.

[3] J. P. de Gyvez. Yield modeling and beol fundamentals. In *Proceedings of 2001 International Workshop on System-Level Interconnect Predition*, pages 135–163, 2001.

[4] K.P. Eswaran and R.E. Tarjan. Augmentation problems. *SIAM Journal on Computing*, 5:653–665, 1976.

[5] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*. Second edition. Springer-Verlag, Berlin, 1993.

[6] M. Grötschel, C.L. Monma, and M.Stoer. Design of survivable networks. In *Handbooks in Operations Research and Management Science, vol. 7: Network Models*, pages 617–672, Amsterdam, 1995. North-Holland.

[7] T.-S. Hsu. *Graph augmentation and related problems: theory and practice*. PhD thesis, University of Texas at Austin, 1993.

[8] `http://www-device.eecs.berkeley.edu/~ptm/`.

[9] S. Khuller, B. Raghavachari, and A. Zhu. A uniform framework for approximating weighted connectivity problems. In *Proc. 10th ACM-SIAM Annual Symposium on Discrete Algorithms*, 1999.

[10] J. Lillis, C.-K. Cheng, T.-T. Y. Lin, and C.-Y. Ho. New performance driven routing techniques with explicit area/delay tradeoff and simultaneous wire sizing. In *ACM/IEEE Design Automation Conference*, pages 395–400, 1996.

[11] B. A. McCoy and G. Robins. Non-tree routing. *IEEE Transactions on Computer-Aided Design*, 14(6):790–784, June 1995.

[12] M. Orshansky. *Personal Communication*. April 2002.

[13] M. Orshansky, L. Milor, P. Chen, K. Keutzer, and C. Hu. Impact of systematic spatial intra-chip gate length variability on performance of high-speed digital circuits. In *IEEE-ACM International Conference on Computer-Aided Design*, pages 62–67, 2000.

[14] G. R. Raidl and I. Ljubic. Evolutionary local search for the edge-biconnectivity augmentation problem. *Information Processing Letters*, 82:39–45, 2002.

[15] S.-C. Wong, G.-Y. Lee, and D.-J. Ma. Modeling of interconnect capacitance, delay, and crosstalk in vlsi. *IEEE Trans. on Semiconductor Manufacturing*, 13(1):753–782, 2000.

[16] T. Xue and E. S. Kuh. Post routing performance optimization via multi-link insertion and non-uniform wiresizin. In *IEEE-ACM International Conference on Computer-Aided Design*, pages 575–580, 1995.