Nontree Routing for Reliability and Yield Improvement

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

***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 postprocessing, tree-augmentation approach, which can be easily integrated in current physical design flows. Our contributions are as follows. 1) We formulate the problem as a variant of the classical two-edge-connectivity augmentation problem in which we take into account such practical issues as wirelength increase budget, routing obstacles, and the use of Steiner points.2) 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. 3) We give a compact integer program formulation which is solved in practical runtime by the commercial optimization package CPLEX for nets with up to 100 terminals. 4) We give a well-scaling greedy algorithm which has a practical runtime up to 1000 terminals, and comes on the average within 1%–2% of the optimum computed by CPLEX. 5) We give a comprehensive experimental study comparing the solution quality and runtime of our methods with the best methods reported in the literature for the related two-edge-connectivity augmentation problem, including a sophisticated heuristic based on minimum-weight branchings [11] and a recent genetic algorithm [17]. 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 a small increase in wirelength. For example, on 1000 terminal nets, the average percentage of biconnected tree edges is 34.19% for a wirelength increase of 1%, and 87.73% for a wirelength increase of 20%. SPICE simulations on industry-routed nets show that nontree routing has the additional benefit of reducing maximum sink delay by an average of 28.26% compared with Steiner routing, and by an average of 3.72% compared to timing optimized routing. SPICE simulations further imply that nontree routing has smaller delay variation due to process variability.**

***Index Terms*—Algorithms, design for manufacturability, integer programming, integrated circuit interconnections, integrated circuit layout, integrated circuit reliability, nontree routing, yield optimization.**

NOMENCLATURE

P	Set of terminals for the given net.
$T(P)$	Given routing tree over terminals of P.
$p_T(u, v)$	The unique path in tree T between u, $v \in T$.
$a(u, v)$	Augmenting path, assumed to be a shortest path between u and v within the feasible routing region (i.e., avoiding the given routing obstacles).
A	Set of augmenting paths.
$G = T \cup A$	Augmented routing graph.
$bridges(G)$	Set of all of the bridge edges of G; an edge (u, v) is a bridge of G if its removal disconnects G.
$l(G)$	Total length of routing graph G.
$l(A)$	Total length of augmenting paths.
$lbridges(G)$	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 $lbridges(G)$.
$l_a(u, v)$	Length of augmenting path $a(u, v)$.
$lbridges_G(u, v) = l(p_T(u, v) \cap bridges(G))$	Total length of bridge edges on path $p_T(u, v)$.
FRR	Rectilinear feasible routing region.[1]
F	Set of corner vertices of the rectilinear feasible routing region FRR.
$H(P \cup F)$	Hanan grid over the points in $P \cup F$, obtained by taking the union of vertical and horizontal lines through the points.
$HV(P \cup F)$	Set of vertices of $H(P \cup F)$.
$N = \lvert T \cap HV(P \cup F) \rvert$	Number of Hanan grid points on the routing tree T.

I. 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 the manufacturing yield of large-area chips, as yield becomes an ever-greater determinant of design viability [2].

In nanometer-scale manufacturing technologies, the 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 of the 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 [4].

Previous works [2], [10], [12] have addressed manufacturing reliability of interconnect exclusively under the assumption that tree topologies are used for routing all of the nets. In this paper, we propose to introduce layout-level redundancy by constructing nontree interconnect topologies for manufacturing yield and reliability improvement. For easier integration with existing flows, we emphasize a *postprocessing* tree-augmentation approach, rather than monolithic nontree 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 the following.

- The existing tradeoff between short- and open-fault critical areas in the BEOL is conducive to the approach we propose. For current

[1]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.

Manuscript received November 27, 2002; revised April 29, 2003. This work was supported in part by Cadence Design Systems, Inc., in part by the MARCO Gigascale Silicon Research Center, and in part by the National Science Foundation under Grant CCR-9988331. This paper was recommended by Associate Editor M. D. F. Wong.

A. B. Kahng is with the Departments of Computer Science and Engineering and of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA 92093-0114 USA (e-mail: abk@ucsd.edu).

B. Liu was with the Computer Science and Engineering Department, University of California at San Diego, La Jolla, CA 92093-0114 USA. He is now with Incentia, Inc., Santa Clara, CA 95054 (e-mail: bobliu@incentia.com).

I. I. Măndoiu was with the Electrical and Computer Engineering Department, University of California at San Diego, La Jolla, CA 92093-0114 USA. He is now with the Computer Science and Engineering Department, University of Connecticut, Storrs, CT 06269–3155 USA (e-mail: ion@engr.uconn.edu).

Digital Object Identifier 10.1109/TCAD.2003.819426

0278-0070/04$20.00 © 2004 IEEE

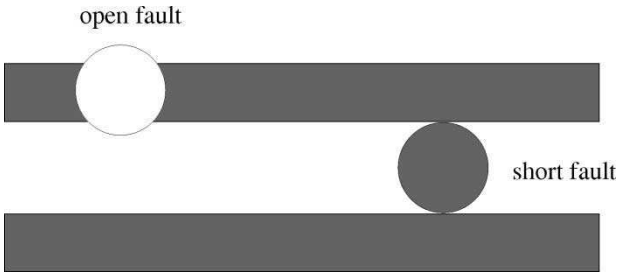

Fig. 1. Open fault is formed due to loss of mass. Short fault is formed due to augmentation of mass.

design methodologies and manufacturing processes, the probability of failure (POF) due to open defects of any given size is more than 3× higher than the POF due to short defects of identical size [4].

- 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$100 nm) 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 [21] and critical net routing for delay optimization [14]. 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* [8], in which a given subgraph must be augmented at minimum cost into a two-edge (or, more generally, k-edge) connected graph.[2] Finding a minimum cost k-edge-connected augmentation is NP-hard even for $k = 2$ [5], and much work has been devoted to finding good heuristics and approximation algorithms, see, e.g., [7] and the references therein. The MRTA formulation differs from the two-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 two-edge connectivity subject to this constraint. In contrast, E2AUG requires 100% two-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 Fig. 2). The existing literature on two-edge-connectivity augmentation focuses almost exclusively on the spanning subgraph formulation of the problem, in which the use of Steiner points is disallowed.

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

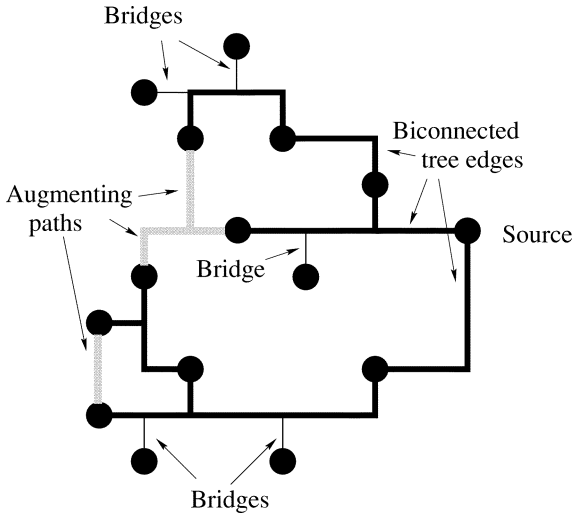

Fig. 2. Initial routing tree T and a set of augmenting paths. Remaining bridges are shown as thin lines.

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 a practical runtime for testcases with up to 1000 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 two-edge-connectivity augmentation, including a sophisticated heuristic based on minimum-weight branchings [11] and a recent genetic algorithm [17].

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 1000 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 nontree 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 nontree routing has smaller delay variation due to process variability.

The rest of the paper is organized as follows. Section II gives the defect model and the problem formulation. Section III gives the reduction to a Hanan grid, integer program formulations, and the greedy

MRTA algorithm. Section IV presents experimental results comparing the solution quality and runtime of our methods with two of the best existing methods for two-edge-connectivity augmentation, as well as results of SPICE simulations showing that nontree routing has improved interconnect delay and process variation robustness. Finally, Section V gives directions for future research.

II. 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.
- **Stapper's defect size distribution**. We assume Stapper's $1/x^3$ distribution of defect sizes [4], [18]. Under this assumption, the probability that a net N fails due to a short defect is proportional to the length of wires routed parallel to N at minimum spacing, and the probability that N fails due to an open defect is proportional to the length [10]. For both types of failures, the proportionality constants are determined by process parameters and design rules.
- **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 [4], the POF due to open defects of a given size is more than 3× 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 a 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.

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:

MRTA Problem

Given:

1) Rectilinear feasible routing region FRR;
2) Rectilinear Steiner routing tree T within FRR;
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., $l(A) \leq W$;
b) Total length $l(T) - lbridges(G)$ of edges of T which are nonbridges in $G = T \cup A$ is maximum.

III. 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 III-B, 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 III-C, we describe an efficient greedy MRTA heuristic.

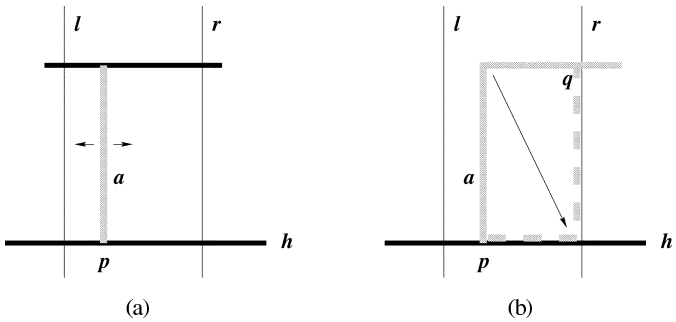

Fig. 3. (a) If augmenting path a connects two parallel edges, then $l(G)$ remains constant when a slides horizontally, and for at least one direction $lbridges(G)$ does not increase. (b) Otherwise, a can be reembedded along the Hanan grid lines.

A. Reduction to Hanan Grid

Theorem 1: There exists an optimum MRTA solution with all of the 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.

1) 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 toward l or sliding toward r) which does not increase the bridge length $lbridges(G)$ [see Fig. 3(a)]. By sliding a in this direction until it reaches the closest grid line, we obtain another optimum solution for which the length of augmenting paths not embedded on the Hanan grid is reduced by an amount equal to the length of a.
2) 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 reembedded as an L shape along the Hanan grid [Fig. 3(b)]. 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 of the augmenting edges are completely embedded on the grid. ■

Remark: Notice that Theorem 1 does not guarantee that all of the 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.

B. 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 fragments of tree edges.

We first give an integer linear program (ILP) formulation for MRTA without Steiner points and without augmenting paths parallel to fragments of tree edges. To formulate this version of the problem as an integer program, we 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 the augmenting path $a(u, v)$ connecting u and v is used in the augmentation, i.e.,

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

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.,

$$y_{u,v} = \begin{cases} 1, & (u,v) \in T - bridges(T \cup A) \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

Let T_u and T_v denote the two subtrees obtained from T, after removing edge $(u, v) \in T$. 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., if and only if there exist $i \in T_u$ and $j \in T_v$, such that $a(i, j) \in A$. Thus, $y_{u,v} = 1$, if and only if $\sum_{i \in T_u, j \in T_v} x_{i,j} \geq 1$. The following integer program captures the MRTA without Steiner points and without augmenting paths parallel to fragments of tree edges. Here, $l(u, v)$ denotes the length of edge $(u, v) \in T$, and $l_a(u, v)$ denotes the length of the augmenting path between u and v.

$$\text{maximize} \sum_{(u,v) \in T} y_{u,v} l(u,v) \quad (3)$$

such that

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

In ILP (3), the objective is to maximize the total length of biconnected tree edges. The first constraint enforces the wirelength budget, and the following set of constraints ensures (based on the previous observation) that only edges of T contained in cycles of $T \cup A$ are counted in the objective function.

We now modify ILP (3) to allow the use of augmenting paths parallel to fragments of tree edges. Such augmenting paths can further increase the total biconnected length, e.g., by fully utilizing the wirelength budget. The key observation is that we do not need to know the exact location of the parallel augmenting paths and, hence, we do not need individual 0/1 variables for each such a path. It suffices to know their total length, represented in the following program by the variable P. This length is now added to the left-hand side of the wirelength budget constraint, and also contributes directly to the objective function (i.e., to the total length of biconnected tree edges). The resulting integer program formulation is

$$\text{maximize } P + \sum_{(u,v) \in T} y_{u,v} l(u,v) \quad (4)$$

such that

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

Similar to ILP's (3) and (4), the ILP formulation for MRTA with Steiner points uses 0/1 variables $y_{u,v}$, indicating whether or not tree edges (u, v) are biconnected, and a variable P representing the total length of augmenting paths parallel to (fragments of) tree edges. In order to capture the possible use of Steiner points as ends of augmenting paths, we now use 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

$$\text{maximize} P + \sum_{(u,v) \in T} y_{u,v} l(u,v) \quad (5)$$

such that

$$P + \sum_{e \in H(P \cup F)} x_e l(e) = W$$
$$\sum_{e \in X} x_e \geq y_{u,v}, \quad \forall (u,v) \in T, X \in X_{u,v}$$
$$x_e \in \{0,1\}, \quad \forall e \in H(P \cup F)$$
$$y_{u,v} \in \{0,1\}, \quad \forall (u,v) \in T$$
$$P \geq 0.$$

Here, $l(u, v)$ 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 of the Hanan grid cuts separating the two connected components, T_u and T_v, of $T - (u, v)$.

As in (3) and (4), the objective is to maximize the total length of the biconnected tree edges (including those biconnected by adding augmenting paths parallel to fragments of tree edges). The first constraint of (5) enforces the wirelength budget. Ensuring that $y_{u,v}$ is set to 1 only if edge $(u, v) \in T$ is biconnected now requires an exponential number of constraints. The formulation is based on the classic 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.

Remark: We note that, despite its exponential size, the fractional relaxation of ILP (5) can still be solved in polynomial time, e.g., by using the Ellipsoid algorithm [6] with a separation oracle that runs a min-cut algorithm for each $(u, v) \in T$ to check feasibility of any given solution. This can be combined with a standard branch-and-bound procedure for finding the optimum integer solution. However, the resulting runtime is likely to be much higher than that required for solving ILP's (3) and (4).

C. Greedy MRTA Algorithm

In this section, we propose a greedy algorithm for the MRTA problem. Our algorithm (see Fig. 4) iteratively adds an augmenting path $a(u, v)$ that maximizes the ratio $lbridges_G(u, v)/l_a(u, v)$ between the length of the 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).

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
   $A = A \cup a(u^*,v^*), \quad G = G \cup a(u^*,v^*)$
   $bridges(G) = bridges(G) - p_T(u^*,v^*)$
   $V = V \cup (a(u^*,v^*) \cap HV(P \cup F))$
   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

Fig. 4. Greedy MRTA algorithm.

In order to efficiently compute the best augmenting path in each greedy iteration, we precompute the length of all of the 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 from 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 grid vertices on a, and compute the lengths of all of the 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 $H(P \cup F)$[3] 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.

IV. Experimental Results

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

- A greedy best-drop heuristic in [11], 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.[4]
- 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 [17].

We implement three versions of the greedy MRTA algorithm:

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

The c) version of the greedy MRTA algorithm gave almost identical results to the b) version in experiments with 1–20% wirelength budget and 5–20 terminals, and therefore we omit its results.

We also implemented versions a) and b) for the Best-Drop and ILP algorithms. Version b) of the ILP algorithm corresponds to solving exactly the integer program formulation (4); version a) is based on the same formulation written for the smaller set of allowed augmenting paths. Formulation (5), which allows the use of arbitrary paths on the Hanan grid and insertion of new Steiner points, was not used in the experiments due to the prohibitive running time required by its solution.

The integer program (4) 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 of the experiments were conducted on a Sun SPARC Ultra 10 workstation with 256 MB memory. For each instance size $n \in \{5, 10, 50, 100, 500, 1000\}$ we generated 100 instances uniformly at random from a 10 000 × 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 I shows the statistics on the number of nodes (including wire turns), number of leaves, and total wirelength of the initial trees.

TABLE I
Initial Routing Statistics (Averages Over 100 Random Instances of Each Size)

#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 II 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 the b)

[3]For a Fibonacci heap implementation of Dijkstra's algorithm [3], $D = O(|P \cup F|^2 \log |P \cup F|)$. Since the edges of $H(P \cup F)$ have positive integer weights, it is possible to improve the asymptotic bound on the running time by replacing Dijkstra's algorithm with Thorup's recent linear-time single-source shortest path algorithm [19]. For Thorup's algorithm, D becomes proportional to the number of vertices and edges in $H(P \cup F)$, i.e., $D = O(|P \cup F|^2)$.

[4]We have modified the code in [11] to enforce a specified wirelength budget.

TABLE II

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	#Sinks	Greedy(a)			Greedy(b)			Best-Drop(a)			Best-Drop(b)			ILP(a)			ILP(b)		
Incr.		#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	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

versions of the algorithms achieve significantly better solution quality than the a) versions for small wirelength increased budgets. In the case of the greedy MRTA algorithm, version b) is better than version a) by as much as 18.54%. Table III 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 1000 sinks. For 1–20% wire-

TABLE III
COMPARISON OF GREEDY MRTA, BEST-DROP, GENETIC, AND ILP ALGORITHMS FOR UNLIMITED WIRELENGTH BUDGET (AVERAGES OVER 100 RANDOM INSTANCES)

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	-	-	-	-	-	-	-	-	-

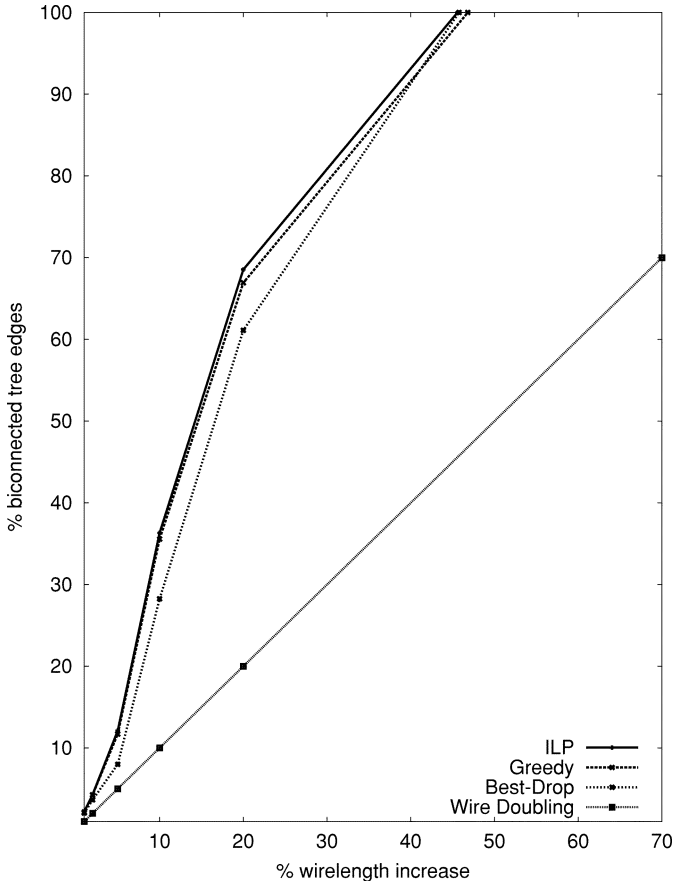

Fig. 5. Tradeoff curves between average biconnectivity and average wirelength increase (both measured as a percentage of the original tree length) for the ILP, Greedy, and Best-Drop algorithms on random nets with 20 sinks. The 1:1 tradeoff achieved by wire doubling is included as a reference.

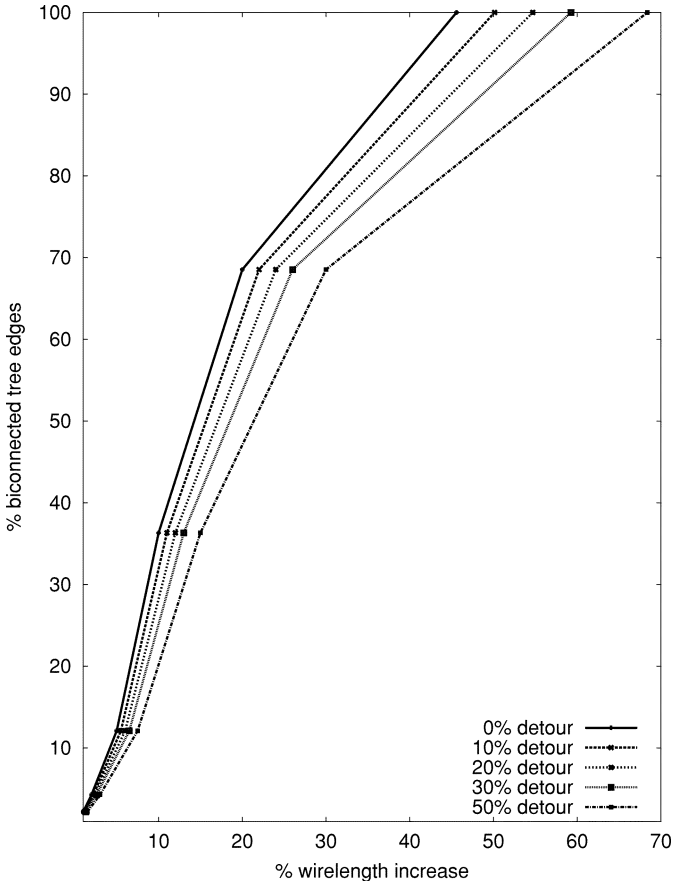

Fig. 6. Tradeoff curves between optimum biconnectivity and wirelength increase (both measured as a percentage of the original tree length) for varying average detours caused by obstacles (random nets with 20 sinks).

length 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 of the wirelength budgets. The much slower Best-Drop and Genetic heuristics outperform greedy MRTA only for unlimited wirelength budget and small number of sinks by a small amount.

The tradeoff curves between biconnectivity and wirelength increase achieved by the ILP, Greedy, and Best-Drop algorithms on random test-cases with 20 sinks are given in Fig. 5. The tradeoff curves are well above the 1:1 tradeoff achieved by wire doubling, which was included in the figure as a comparison term. For wirelength increase budgets of around 20%, the achieved biconnectivity is more than 3× larger than the increase in wirelength and, therefore, the decrease in open critical achieved by biconnectivity is very likely much greater than the increase in short critical area due to the extra wirelength. Fig. 6 shows the tradeoff curves between optimum biconnectivity and the higher wirelength increase caused by the presence of obstacles (again for random 20 sink nets). The tradeoff remains favorable to overall manufacturing

TABLE IV
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)

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

reliability improvements even when augmenting paths have an average detour length of as much as 50%.

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 IV 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 1000 sinks, and already achieves a significant improvement in routing reliability at a very low wirelength increase cost.

TABLE V
MAXIMUM SPICE SINK DELAYS (ns) AND DELAY VARIATIONS (PERCENTS) UNDER NOMINAL AND UNIFORMLY 6.67% INCREASED/DECREASED WIRE WIDTHS 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	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%

The impact of nontree 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 noncritical 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 [13] with buffer insertion disabled and identical sink required-arrival times. Each interconnect was represented by a Π model and driven by a 1.8-V voltage source with a ramped input signal of 150 ps slew time. 50% delay from the source to each sink was simulated based on 180 nm ITRS predictive technology model beta version [9] with the following parameters: unit wire resistance $r = 0.040\ \Omega/\mu$m, unit wire capacitance $c = 0.259\ fF/\mu$m, sink capacitance $c_t = 63.358\ fF$ and source driving resistance $R_b = 139.434\ \Omega$.

For evaluating robustness to process variability, we used an experiment which models systematic variation sources such as lens aberrations which cover 5–10 mm ranges [15], [16], i.e., ranges that are larger than those covered today by unbuffered interconnect. Assuming 100% wire-width correlation, we ran our simulations under three scenarios: nominal wire width, wire width reduced by $dw = 6.67\%$, and wire width increased by $dw = 6.67\%$. For each scenario, we computed unit length wire capacitance using the formulas in [20] 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 V and VI. In these tables, the results under 0% wirelength budget correspond to the initial (area, respectively, timing-optimized) routing trees.

Results for noncritical nets (Table V) show that nontree augmentation continuously reduces maximum source-to-sink delays in most of the instances in our experiments (except $net10$). An average of 28.26%, and maximum of 62.15% delay reduction can be achieved (for $net12$) with a 20% wirelength budget. Nontree augmentation also decreases the effect of process variation in most noncritical instances (except

TABLE VI
MAXIMUM SPICE SINK DELAYS (ns) AND DELAY VARIATIONS (PERCENTS) UNDER NOMINAL AND UNIFORMLY 6.67% INCREASED/DECREASED WIRE WIDTHS FOR 14 RANDOMLY GENERATED NETS WITH 15 SINKS EACH. INITIAL TREES ARE CONSTRUCTED USING THE P-TREE ALGORITHM [13] WITH IDENTICAL SINK REQUIRED-ARRIVAL TIMES

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%

$net4$). An average of 13.79% and a maximum of 28.86% delay variation reduction is observed (for $net12$) when comparing nominal wire width w and $w - dw$ wire width. Results for timing-optimized interconnect trees (Table VI) show that nontree 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 nontree augmentation. Nontree augmentation decreases process variation in all of the timing-optimized instances, with an average of 3.24% and a maximum of 12.17%.

An explanation of the above results is that nontree augmenting paths can decrease interconnect source-to-sink delay by forming shorter connections, but can also increase interconnect delay due to increased capacitance.[5] The probability for nontree 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 nontree augmentation scheme achieves better improvements in interconnect delay and variability for noncritical area-optimized interconnects.

V. CONCLUSION AND FUTURE WORK

In this paper, we have proposed the introduction of redundant interconnect as a postrouting 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 augmentation and proposed both practical integer program formulations and a well-scaling greedy algorithm which comes within 1%–2% of the optimum on the average. Experiments on both randomly generated and industry testcases show that our methods achieve a significant increase in reliability (as measured by the percentage of biconnected tree edges) with very small increases in wirelength. Furthermore, SPICE simulations show that redundant interconnect also decreases maximum sink delay and delay variation due to process variability.

Our ongoing research explores simultaneous consideration of both open and short critical area in the augmentation process and chip level evaluation of the proposed algorithms. In particular, we are experimenting with modified augmenting path costs that take into account 1) the increase in short critical area due to routing extra wire adjacent to existing nets and 2) routing congestion information. We are also studying optimal net-ordering heuristics to be used in conjunction with the greedy augmentation algorithm, as well as extensions of the greedy algorithm to simultaneous augmentation of an entire set of nets (e.g., extracted from a sliding window of fixed size).

[5]Since path delays may both increase and decrease during nontree augmentation, both setup and hold time violations may be introduced during this step. A simple solution to this problem is to follow nontree augmentation by timing verification—any detected violations can be easily corrected by removing the corresponding augmentation paths.

REFERENCES

[1] M. Borah, R. M. Owens, and M. J. Irwin, "A fast and simple Steiner routing heuristic," *Discrete Appl. Math.*, vol. 90, pp. 51–67, 1999.

[2] V. K. R. Chiluvuri, "Yield optimization in physical design: A review," in *Proc. 5th ACM/SIGDA Physical Design Workshop*, 1996, pp. 198–206.

[3] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, 2nd ed. Cambridge, MA: MIT Press, 2001.

[4] J. P. de Gyvez, "Yield modeling and beol fundamentals," in *Proc. Int. Workshop System-Level Interconnect Predition*, 2001, pp. 135–163.

[5] K. P. Eswaran and R. E. Tarjan, "Augmentation problems," *SIAM J. Comput.*, vol. 5, pp. 653–665, 1976.

[6] M. Grötschel, L. Lovász, and A. Schrijver, *Geometric Algorithms and Combinatorial Optimization*, 2nd ed. Berlin, Germany: Springer-Verlag, 1993.

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

[8] T.-S. Hsu, "Graph Augmentation and Related Problems: Theory and Practice," Ph.D. dissertation, Univ. of Texas, Austin, TX, 1993.

[9] Berkeley Predictive Technology Model Available: http://www-device.eecs.berkeley.edu/~ptm/ [Online]

[10] E. P. Huijbregts, H. Xue, and J. A. G. Jess, "Routing for reliable manufacturing," *IEEE Trans. Semiconduct. Manufact.*, vol. 8, pp. 188–194, May 1995.

[11] S. Khuller, B. Raghavachari, and A. Zhu, "A uniform framework for approximating weighted connectivity problems," in *Proc. 10th ACM-SIAM Annu. Symp. Discrete Algorithms*, 1999.

[12] S. Y. Kuo, "YOR: A yield optimizing routing algorithm by minimizing critical areas and vias," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 1303–1311, Sept. 1993.

[13] 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 *Proc. ACM/IEEE Design Automation Conf.*, 1996, pp. 395–400.

[14] B. A. McCoy and G. Robins, "Non-tree routing," *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 784–790, June 1995.

[15] M. Orshansky, Personal Communication, Apr. 2002.

[16] 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 Int. Conf. Computer-Aided Design*, 2000, pp. 62–67.

[17] G. R. Raidl and I. Ljubic, "Evolutionary local search for the edge-biconnectivity augmentation problem," *Inform. Process. Lett.*, vol. 82, pp. 39–45, 2002.

[18] C. H. Stapper, "Modeling of defects in integrated circuit photolithographic patterns," *IBM J. Res. Develop.*, vol. 28, no. 4, pp. 461–475, 1984.

[19] M. Thorup, "Undirected single-source shortest paths with positive integer weights in linear time," *J. ACM*, vol. 46, no. 3, pp. 362–394, 1999.

[20] S.-C. Wong, G.-Y. Lee, and D.-J. Ma, "Modeling of interconnect capacitance, delay, and crosstalk in VLSI," *IEEE Trans. Semiconduct. Manufact.*, vol. 13, no. 1, pp. 753–782, 2000.

[21] T. Xue and E. S. Kuh, "Post routing performance optimization via multi-link insertion and nonuniform wiresizing," in *Proc. IEEE-ACM Int. Conf. Computer-Aided Design*, 1995, pp. 575–580.