Short Papers

An Interconnect Reliability-Driven Routing Technique for Electromigration Failure Avoidance

Xiaodao Chen, *Student Member*, *IEEE*,
Chen Liao, Tongquan Wei, *Member*, *IEEE*,
and Shiyan Hu, *Senior Member*, *IEEE*

Abstract—As VLSI technology enters the nanoscale regime, design reliability is becoming increasingly important. A major design reliability concern arises from electromigration which refers to the transport of material caused by ion movement in interconnects. Since the lifetime of an interconnect drastically depends on the current flowing through it, the electromigration problem aggravates with increasingly growing thinner wires. Further, the current-density-induced interconnect thermal issue becomes much more severe with larger current. To mitigate the electromigration and the current-density-induced thermal effects, interconnect current density needs to be reduced. Assigning wires to thick metals increases wire volume, and thus, reduces the current density. However, overstretching thick-metal assignment may hurt routability. Thus, it is highly desirable to minimize the thick-metal usage, or total wire cost, subject to the reliability constraint. In this paper, the minimum cost reliability-driven routing, which consists of Steiner tree construction and layer assignment, is considered. The problem is proven to be NP-hard and a highly effective iterative rounding-based integer linear programming algorithm is proposed. In addition, a unified routing technique is proposed to directly handle multiple current levels, which is critical in analog VLSI design. Further, the new algorithm is extended to handle blockage. Our experiments on 450 nets demonstrate that the new algorithm significantly outperforms the state-of-the-art work [1] with up to 14.7 percent wire reduction. In addition, the new algorithm can save 11.4 percent wires over a heuristic algorithm for handling multiple currents.

Index Terms—VLSI circuit computer-aided design, interconnect reliability, electromigration, Steiner tree construction, integer linear programming.

1 Introduction

Design reliability becomes increasingly important in nanoscale circuit design. A major reliability concern is due to the electromigration which refers to the transport of material caused by ion movement in interconnects. The lifetime of an interconnect drastically depends on the current flowing through according to Black's Law [2]. Consequently, the problem aggravates when the wire becomes increasingly thinner. In addition, the current-density-induced interconnect thermal issue becomes more severe with larger current density. To mitigate the electromigration effect and the induced thermal effect, it is in a great demand to reduce the current density along wires. Assigning wires to thick metals improves reliability due to their tolerance of large current density. However, overstretching thick-metal assignment may make the design unroutable [3]. Thus, it is highly desirable to minimize the thick-metal usage, or total wire cost, subject to the constraint on Mean Time To Failure (MTTF) which is a commonly used metric of reliability.

There are some previous works addressing this reliability-driven minimum cost Steiner routing and layer assignment problem [1], [4], [5], [6], [7]. However, they are mainly focused on applying the optimizations in the global routing stage. Since timing closure is already very difficult to achieve in advanced technology, it is not desirable to perform simultaneous timing and reliability-driven global routing. This may necessarily put more burden on the already oversqueezed global routing stage. Thus, our application target is on the detailed routing stage in a design flow for interconnect lifetime improvement. Note that previous works [1], [4], [8] perform wire sizing to approximate the layer assignment. In reality, only discrete routing layers are presented and continuous wire sizing may not be practical. There are also works (e.g., [3]) on layer assignment for timing optimization; however, they do not consider the reliability issue.

In this paper, a new reliability-driven minimum cost Steiner routing and layer assignment approach is proposed. In contrast to all the previous work that designs heuristics, our new algorithm is based on the integer linear programming (ILP) formulation. This allows us to compute the optimal routing on small nets and greatly improve the solution quality on large nets when compared to the state-of-the-art schemes presented in [1]. A striking feature for the new algorithm is its capability to handle multiple currents. This is particularly important in analog VLSI design due to the existence of a large multitude of current levels. In [1], a single current value is used to approximate the effects of multiple current values. This technique cannot be extended to handle multiple current values due to its underlying limitation. In contrast, the proposed routing technique directly handles the multiple current values by a novel *unified multiple current routing* technique.

In this paper, the reliability-driven routing problem is formulated as simultaneous Steiner tree construction and layer assignment. The main contribution of this paper is summarized as follows:

- The minimum cost reliability-driven Steiner routing and layer assignment problem is proven to be NP-hard.
- A highly effective iterative rounding-based integer linear programming algorithm is proposed, allowing us to compute optimal solutions on small nets and well approximate the optimal solutions on large nets.
- Multiple currents are handled by the proposed unified multiple current routing technique, which is critical in analog VLSI design.
- The new algorithm is extended to handle blockage, which makes it ready for practical use.
- Our experiments on 450 test cases demonstrate that the new algorithm significantly outperforms the state-of-the-art work [1] with up to 14.7 percent wire reduction. In addition, the new algorithm can save 11.4 percent wire as compared to the heuristic algorithm for handling multiple currents.

The rest of the paper is organized as follows: Section 2 formulates the reliability-driven routing problem. Section 3 describes the iterative rounding-based integer linear programming approach for the problem. Section 4 describes the algorithm for handling multiple currents. Section 5 presents the experimental results with analysis. A summary of work is given in Section 6.

2 Preliminaries

2.1 Interconnect Reliability

In this paper, the interconnect reliability is measured by a commonly used metric, namely, MTTF. According to Black's Law [2], interconnect MTTF is given as

- *X. Chen, C. Liao, and S. Hu are with the Department of Electrical and Computer Engineering, Michigan Technological University, Houghton, MI 49931. E-mail: {cxiaodao, cliao, shiyan}@mtu.edu.*
- *T. Wei is with the Computer Science and Technology Department, East China Normal University, Shanghai 200241, China. E-mail: tqwei@cs.ecnu.edu.cn.*

Manuscript received 20 Apr. 2009; revised 12 Feb. 2010; accepted 16 June 2010; published online 21 Oct. 2010.
Recommended for acceptance by Y. Zorian.
For information on obtaining reprints of this article, please send e-mail to: tdsc@computer.org, and reference IEEECS Log Number TDSC-2009-04-0055.
Digital Object Identifier no. 10.1109/TDSC.2010.57.

$$MTTF = \frac{A}{J^2} e^{\frac{E}{KT}}, \tag{1}$$

where A denotes the cross-section area of the interconnect, J denotes the current density, E denotes the activation energy, K denotes the Boltzmann constant, and T denotes the temperature. The above Black's Law clearly indicates the significance of current density in the interconnect reliability. In the design, one needs to suppress maximum current density to increase the interconnect lifetime.

2.2 Problem Formulation

The technique presented in this paper is applied to net level. The input to the reliability-driven routing problem is a driver r and a set of n sinks, denoted by $S = \{s_1, s_2, \dots, s_n\}$, which belong to a net. We are to construct a Steiner tree (with layer assignment) over the driver and sinks such that the reliability constraint is satisfied. Due to layer assignment, we are also given a set of m routing layers as $L = \{l_1, l_2, \dots, l_m\}$. Given an edge e on a layer l, the width, length, height, cost, and delay of the edge are denoted by $w(e,l)$, $l(e,l)$, $h(e,l)$, $c(e,l)$, and $d(e,l)$, respectively.

Before defining wire cost and wire delay, note that the driver and sinks are associated with currents. Denote by I_a the value of the current associated with node a. If the current is injected into the node, I_a is positive. Otherwise, I_a is negative. Denote by I_e the current flowing through an edge e. Note that there can be a set of currents associated with each sink/driver that forms a time-varying series (e.g., a sinusoidal waveform) in analog VLSI design. The driver/sink currents can be computed by either circuit simulation or manually attaching current values by designers [1].

Let us now link the MTTF metric to the layer assignment. We follow the formulation in [1]. As shown in [1], for a wire to have reasonable MTTF, the width of the wire layer $w(e,l)$ needs to satisfy [1]

$$w(e,l) \cdot h(e,l) \geq \frac{I_e \cdot s}{J_{peak}}, \tag{2}$$

where I_e is the current flowing through the wire e, s is a guard-band factor to balance the MTTF and routing resource usage, J_{peak} is the maximum allowable peak current density determined by the technology, and $h(e,l)$ is the thickness of routing layer l. MTTF and s are highly correlated. Varying s, different trade-offs between routing resource usage and MTTF can be obtained. In practice, s is set to be within $[1.1, 1.2]$ as shown in [1].

Given a wire e at a layer l, the cost for a wire is generic. In this paper, to illustrate the effectiveness of the proposed technique, the wire cost is defined by wire volume, i.e., $c(e,l) = l(e,l) \cdot w(e,l) \cdot h(e,l)$. This is a commonly used metric in routing and layer assignment. Equation (2) says that $w(e,l) \cdot h(e,l) \geq \beta I_e$, where $\beta = \frac{s}{J_{peak}}$ is a constant with a given $MTTF$ target. For a routing tree T, define the total wire cost as the sum of the costs for all edges in T. Given an s which corresponds to a constraint on MTTF, one aims to compute a Steiner routing and layer assignment solution such that the total wire cost is minimized.

In this paper, since the reliability-driven routing technique needs to change the routing topology and the layer assignment, timing degradation may be introduced to the initial design. To evaluate the timing, the widely used Elmore delay model is adopted. That is, $d(e,l) = R_e \cdot (C_e/2 + C_l)$, where R_e, C_e, and C_l are edge resistance, edge capacitance, and load capacitance, respectively.

The problem is to meet an MTTF target (through fixing s) by Steiner tree construction and layer assignment with minimum total wire cost. It can be formulated as follows:

Reliability-driven minimum cost Steiner routing and layer assignment. Given a driver and a set of sinks, each of which is associated with a current vector, a set of routing layers L, and cost of each wire on each layer, to compute a Steiner tree with layer assignment such that reliability constraint is satisfied and the total wire cost is minimized.

The problem is NP-hard. This can be shown by reducing from minimum length Steiner tree problem which is known to be NP-hard [9]. Given an instance of a minimum Steiner tree problem, assign it as an instance for our problem where there is only one layer (with one size) and it can hold all possible currents. We are to compute a minimum volume Steiner tree, which is the same as a minimum length Steiner tree since wire width and wire height are constants in a single layer. Thus, this problem is NP-hard given that the minimum length Steiner tree problem is NP-hard. We have

Theorem 1. *The reliability-driven minimum cost Steiner routing and layer assignment problem is NP-hard.*

3 The Algorithm

3.1 Integer Linear Programming Formulation

In the proposed algorithm for interconnect reliability-driven routing, both Steiner tree construction and layer assignment will be performed. Each wire/edge at a layer is associated with an MTTF constraint as computed by (2). The problem is first formulated as an ILP problem, and then, solved by the iterative rounding technique. Previous works [10], [11] also focus on the ILP-based routing. However, none of them considers the reliability-driven current issues as handled in this paper.

A Hannan grid is first constructed on the given set of the driver and sinks. Each grid point is associated with a node. Some of them are driver and sinks. Our Steiner tree will only use Hannan edges thus resulting in the commonly used Hannan routing. For simplicity, index each Hannan edge e by its two endpoints (a,b) which are Hannan grid points. In the ILP formulation, each Hannan edge is associated with a variable $I_{a,b}$ corresponding to the value of the current flowing through e.

By Kirchhoff's Law, for each node g, the sum of input currents is equal to the sum of output currents. That is, for a Steiner node s which is neither a driver nor a sink

$$\sum I_{s-in} = \sum I_{s-out}, \ \forall \text{ Steiner node s.} \tag{3}$$

For a driver d, denoting by $I_{d-drive}$ the current flowing from the driver, one has

$$\sum I_{d-in} + I_{d-drive} = \sum I_{d-out}, \ \forall \text{ driver d.} \tag{4}$$

For a sink s, denoting by I_{s-sink} the current flowing out of the sinks s, one has

$$\sum I_{g-in} = \sum I_{g-out} + I_{g-sink}, \ \forall \text{ sink g.} \tag{5}$$

According to the MTTF constraint computed by (2), each wire e needs to satisfy $w(e,l) \cdot h(e,l) \geq \beta \cdot I(e)$. Denote by v_e the *unit volume* of a wire e at layer l which is computed as $w(e,l) \cdot h(e,l)$. Since the wire volume needs to be larger than the current subject to the factor of β, we have

$$v_e \geq \beta I(e), \tag{6}$$

where β denotes $\frac{I_e \cdot s}{J_{peak}}$.

Note that $v_e = 0$ means that wire e is not used in the routing. In practice, there are only limited number of layers (e.g., eight in some advanced technology) and each wire layer has its own characteristics. Let m denote the number of layers. v_e cannot take arbitrary continuous values, and it needs to be chosen from the set $v_e \in \{V_1, V_2, \dots, V_m\}$, where V_i denotes the unit volume that corresponds to assigning wire e to layer i.

It is helpful to look at a simple example as shown in Fig. 1. If the node a is a Steiner node, then $I_{ba} + I_{ea} = I_{ad} + I_{ac}$. If the node a is a sink, then $I_{ba} + I_{ea} = I_{ad} + I_{ac} + I_{a-sink}$. If the node a is a drive,

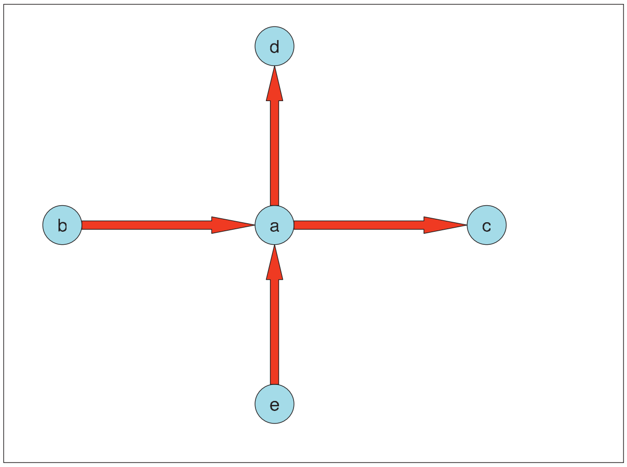

Fig. 1. Illustration of node current computation in LP formulation.

then $I_{ba} + I_{ea} + I_{d-drive} = I_{ad} + I_{ac}$. The MTTF constraints can be formulated as $v_{ba} \geq \beta I_{ba}, v_{ea} \geq \beta I_{ea}, v_{ad} \geq \beta I_{ad}$, and $v_{ac} \geq \beta I_{ac}$.

The objective of the routing is to minimize the total cost of wires. The cost function is generic. To illustrate the effectiveness of the proposed technique, wire volume is used as the cost which is a commonly used metric. Precisely, wire cost is computed as $v_e \cdot l_e$, where l_e denotes the wire length (which is a constant if the edge is given) and v_e is the unit volume. The reliability-constrained minimum cost Steiner routing and layer assignment problem can be formulated to an integer linear program as follows:

$$\begin{aligned} & min \quad \sum_e v_e \cdot l_e \\ & s.t. \\ & \qquad v_e \geq \beta I_e, \ \forall e \\ & \qquad \sum I_{s-in} = \sum I_{s-out}, \ \forall \text{ Steiner node s} \\ & \qquad \sum I_{d-in} + I_{d-drive} = \sum I_{d-out}, \ \forall \text{ driver d} \\ & \qquad \sum I_{g-in} = \sum I_{g-out} + I_{g-sink}, \ \forall \text{ sink g} \\ & \qquad v_e \in \{V_1, V_2, \ldots, V_m\}, \ \forall e. \end{aligned} \tag{7}$$

After solving the ILP using standard LP solver, routing tree and layer assignment can be constructed by v_e. However, it is well known that the large-scale ILP cannot be efficiently solved due to the integer constraints. Thus, the classic iterative rounding technique is used in Section 3.3 to efficiently compute the integer solution.

Note that there can be multiple solutions for a net and the solution can be a nontree. Consider the example shown in Fig. 3. There are at least two solutions for this net (Figs. 3a and 3b). In the net, there are one driver, one sink, and two Hanan nodes. For simplicity, we assume that the width of layer 1 is one, the width of layer 2 is two, and the lengths of all Hanan edges are equal to one. Our LP program may produce either solution which has the same objective value four. Therefore, for a net, there might be more than one solution, and those solutions may be tree-based (Fig. 3b) or non-tree-based (Fig. 3a) because of the reconvergent path.

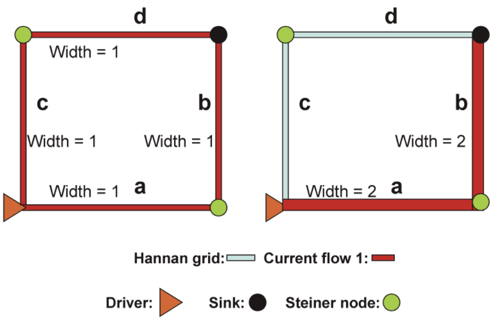

Fig. 3. Two solutions with the same objective solution.

3.2 Handling Blockage

In VLSI design, there are often routing blockages (such as big macros) which need to be considered in constructing the route. Suppose that two blockages are introduced in Fig. 2, which results in Fig. 4.

The blockage needs to be avoided in routing. To handle this, a weighting factor τ_e for each edge is introduced. Given an edge e, if it is blocked, set τ_e to infinity. Otherwise, set τ_e to 1. In practice, a large number is used to serve as infinity. The objective function is changed to $min \sum_e \tau_e \cdot v_e \cdot l_e$, where τ_e is the weighting factor on the edge e, while the remaining ILP is the same as (7). The new ILP is as follows:

$$\begin{aligned} & min \quad \sum_e \tau_e \cdot v_e \cdot l_e \\ & s.t. \\ & \qquad v_e \geq \beta I_e, \forall e \\ & \qquad \sum I_{s-in} = \sum I_{s-out}, \ \forall \text{ Steiner node s} \\ & \qquad \sum I_{d-in} + I_{d-drive} = \sum I_{d-out}, \ \forall \text{ driver d} \\ & \qquad \sum I_{g-in} = \sum I_{g-out} + I_{g-sink}, \ \forall \text{ sink g} \\ & \qquad v_e \in \{V_1, V_2, \ldots, V_m\}, \ \forall e. \end{aligned} \tag{8}$$

Fig. 5 shows an example to illustrate a rerouted Steiner tree with blockage avoidance. Blocked edges are with large weighting factors. The part of the objective corresponding to edges (A, B)

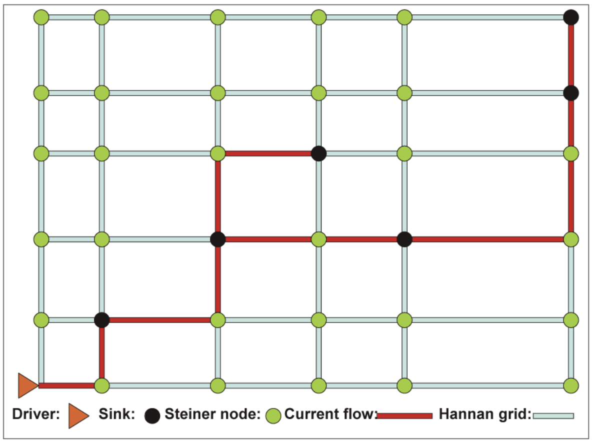

Fig. 2. A routing solution.

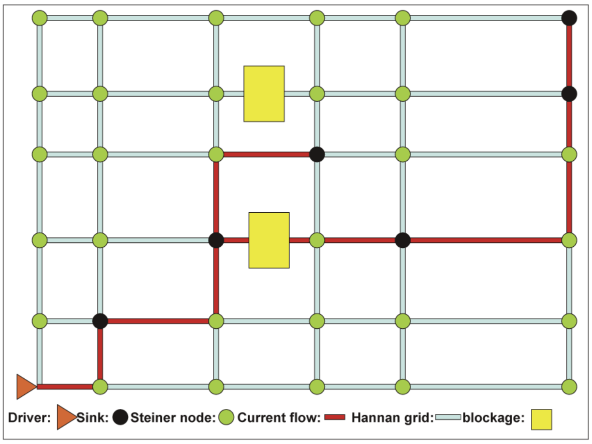

Fig. 4. An example with blockage.

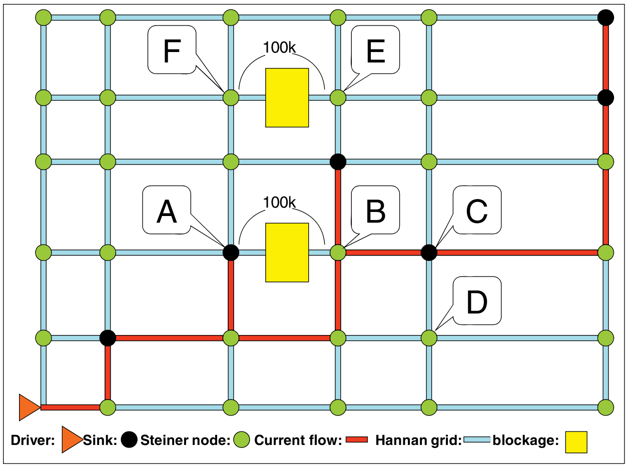

Fig. 5. A routing solution with blockage. The weighting factor of edges with blockage is set to 100k.

and (E, F) can be written as $\tau_{AB} \cdot v_{AB} \cdot l_{AB}$ and $\tau_{EF} \cdot v_{EF} \cdot l_{EF}$, where $\tau_{AB} = 100$ k and $\tau_{EF} = 100$ k. In contrast, the weighting factor of τ_e of unblocked edges is set to 1. For example, the part of the objective corresponding to edges (B, C) and (C, D) can be written as $\tau_{BC} \cdot v_{BC} \cdot l_{BC}$ and $\tau_{CD} \cdot v_{CD} \cdot l_{CD}$, where $\tau_{BC} = 1$ and $\tau_{CD} = 1$. The large τ_e prevents the LP solver from choosing those edges in routing tree.

3.3 Iterative Rounding to Solve ILP

Since there are many integer constraints in the linear programming formulation, the problem of the reliability-driven minimum cost Steiner routing and layer assignment cannot be efficiently solved on large nets. To handle this, a commonly used iterative rounding technique is adopted. This technique has been used in a number of works in VLSI CAD such as [12].

As integer constraints $v_e \in \{V_1, V_2, \ldots, V_m\}$ in (7) are the most difficult constraints, they are relaxed as $0 \leq v_e \leq v_{max}$, where v_{max} denotes the largest value in $\{V_1, V_2, \ldots, V_m\}$. Consequently, the problem is translated into a *relaxed linear programming* problem which is well known to be much more efficient to solve. The iterative rounding is accomplished by iteratively adding new constraints to the problem and solving the problem until all the variables have integer values.

First, the original relaxed linear program is solved, where the solution may contain fractional numbers. Define the feasible unit volumes as the unit volumes in $\{V_1, V_2, \ldots, V_m\}$. All variables in the solution that are equal to the corresponding feasible unit volume v_t, where $v_t \in \{V_1, V_2, \ldots, V_m\}$, will be first fixed to v_t. This is accomplished by adding new constraints such as $v_e = v_t$, to (7) if v_e is equal to v_t in the solution. If there is no variable equal to any feasible unit volume, those close to v_t in the current solution will be chosen and fixed. Precisely, a *round up and fix* strategy will be used. That is, the variable with the rounding error (when being rounded up) smaller than δ, which is a user input, will be rounded up. If there is no variable having rounding error smaller than δ, the variable with the smallest rounding error (when being rounded up) will be chosen, rounded up, and fixed during the following iterations. In this way, the rounding procedure will converge and no oscillation may occur. The new linear program will be solved by LP again to obtain a new solution. This process is iterated until all variables have feasible unit volumes. These strategies guarantee that in each iteration, there will be at least one new rounded variable which is fixed during following iterations and ensure that an integer solution will be obtained.

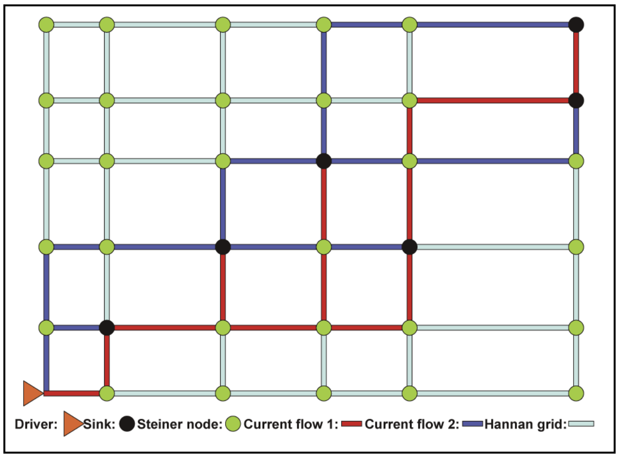

Fig. 6. Routes with different topologies by different sets of current levels.

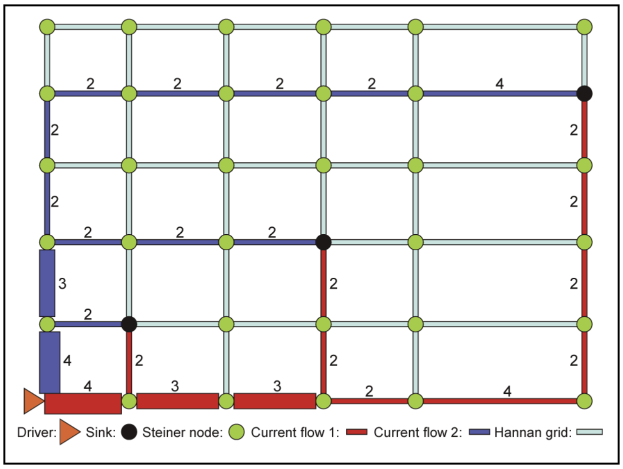

Fig. 7. Illustration of different topologies in a multicurrent problem.

4 The Algorithm for Multiple Current Levels

The continuous-valued current in analog VLSI design is often quantized to discrete current values. Since the current flowing through wires of an analog VLSI design varies at different instants, the discrete current values associated with sinks and the driver change with time. Let I_{t_i} denote the discrete current value at instant t_i for $1 \leq i \leq k$. The driver and sinks are associated with $n+1$ discrete current values as $I_{t_i} = \{I_{1t_i}, I_{2t_i}, \ldots, I_{nt_i}, I_{(n+1)t_i}\}$, where the driver is indexed as 1 and the sinks are indexed from two to $n+1$. Since the topologies derived at each single current level are not guaranteed to be the same, directly applying the single current value technique described in Section 3 may not generate a feasible solution in the multicurrent case. For example, the topology produced using ILP at the current level I_{t_1} may be different from the topology produced using ILP at the current level I_{t_2}. This can be clearly seen from Fig. 6 by comparing blue route (for I_{t_1}) and red route (for I_{t_2}).

An intuitive approach to the multicurrent problem is to derive the routing topology at a certain current level, and then, adjust the layer assignment to satisfy the constraints corresponding to other current levels. However, this heuristic approach does not globally optimize the volume. Refer to Fig. 7. Two sets of current levels are given and they are shown as (I_{t_1}, I_{t_2}), where I_{t_1} corresponds to the first set of currents and I_{t_2} corresponds to the second set of currents. The computed topology of the first set of currents is shown in blue edges. In order to satisfy the constraint on the second set of currents, the heuristic algorithm needs to increase the layers on some blue edges as shown in Fig. 7, e.g., from layer 1 to layer 2, and from layer 2 to layer 3. Blue edges will then satisfy the constraints of both sets of currents and the total cost is 33. However, if we handle both sets of current levels simultaneously, the total cost is only 29, as shown in red edges. One can see that the topology and the

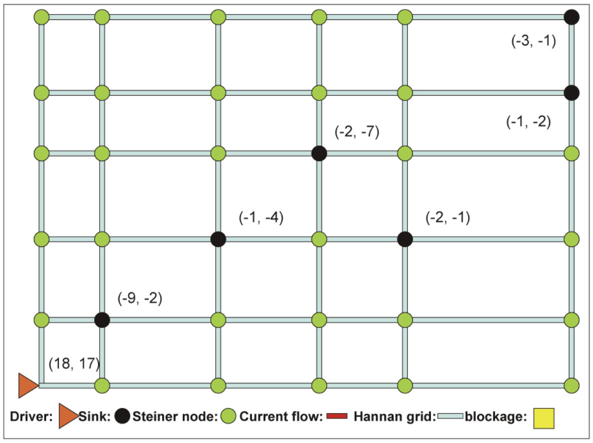

Fig. 8. An example with multicurrent.

layer assignment are significantly different between blue edges and red edges. This clearly demonstrates that the heuristic algorithm cannot compute the optimal routing solution.

In this paper, a global optimization method is proposed for the multiple current problem. In the new approach, all constraints at k current levels are incorporated in a unified fashion for any node including the driver, Steiner nodes, and sinks. This is why our approach is called *unified multiple current routing*.

The example as shown in Fig. 8 illustrates the proposed algorithm. Assume that each sink and driver have two current values. For the sink g with current values of -1 and -4, its constraints are given by

$$\begin{aligned} \sum I_{in\ t_1} &= \sum I_{out\ t_1} + 1, \\ \sum I_{in\ t_2} &= \sum I_{out\ t_2} + 4. \end{aligned} \tag{9}$$

The constraints of the Steiner node s are given by

$$\begin{aligned} \sum I_{in\ t_1} &= \sum I_{out\ t_1}, \\ \sum I_{in\ t_2} &= \sum I_{out\ t_2}. \end{aligned} \tag{10}$$

Similarly, the constraints on the driver d with current values 17 and 18 are given by

$$\begin{aligned} \sum I_{in\ t_1} + 17 &= \sum I_{out\ t_1}, \\ \sum I_{in\ t_2} + 18 &= \sum I_{out\ t_2}. \end{aligned} \tag{11}$$

The constraints on edges are the same as those in the single current value case, that is, $v_e \geq \beta I_{t_i}$.

The objective function of the multicurrent optimization problem is the same as the single current case. The problem formulation based on integer linear program is as follows:

$$\begin{aligned} &min \quad \sum_e v_e \cdot l_e \\ &s.t. \\ &\qquad v_e \geq \beta I_e,\ \forall e \\ &\qquad \textstyle\sum_{\text{I flows in s}} I_{t_i} = \sum_{\text{I flows out of s}} I_{t_i},\ \forall t_i,\ \text{Steiner node s} \\ &\qquad \textstyle\sum_{\text{I flows in d}} I_{t_i} + I_{drivert_i} = \sum_{\text{I flows out of g}} I_{t_i},\ \forall t_i,\ \text{driver d} \\ &\qquad \textstyle\sum_{\text{I flows in g}} I_{t_i} = \sum_{\text{I flows outof g}} I_{t_i} + I_{sink\ t_i},\ \forall t_i,\ \text{sink g} \\ &\qquad v_e \in \{V_1, V_2, \ldots, V_m\}, \forall e. \end{aligned} \tag{12}$$

TABLE 1
Optimality Study on 50 Small Test Cases

Net	Algorithm	Cost	CPU(s)
Net1	Integer Linear Programming	35	0.002
	NEW (LP+Iterative Rounding)	36	0.002
Net2	Integer Linear Programming	40	0.002
	NEW (LP+Iterative Rounding)	41	0.002
Net3	Integer Linear Programming	43	0.002
	NEW (LP+Iterative Rounding)	43	0.002
Net4	Integer Linear Programming	29	0.002
	NEW (LP+Iterative Rounding)	30	0.002
Net5	Integer Linear Programming	47	0.002
	NEW (LP+Iterative Rounding)	48	0.002
AVG. of 50 nets	Integer Linear Programming	41	0.002
	NEW (LP+Iterative Rounding)	42	0.002

AVG. of 50 nets refers to averaged results on 50 nets. Results on five representative nets are also shown. CPU refers to the runtime in seconds.

As indicated by our experimental results, the new approach achieves area savings of up to 11.4 percent compared to the heuristic approach described in the beginning of this section. In addition, our technique is extended to deal with blockage, as is described in Section 3.2.

5 Experimental Results

The proposed algorithm for the reliability-constrained Steiner routing problem is implemented in C, and tested on a Pentium IV machine with 2.4 GHz CPU and 8 GB memory. Experiments are performed on a set of 450 nets at various scales. Due to the lack of access to industrial analog nets, we randomly generated these 450 nets. The generated nets have varying numbers of sinks, current values, and coordinates. The number of sinks is in the range from 1 to 30. The number of Hanan edges varies from 4 to 3,000, and the current values vary from 1 to 40 mA. Since blockage is crucial in practical applications, we also investigate the algorithm performance with blockage. For those nets with blockage, we randomly generate 1-20 blockages on each net. In this paper, to illustrate the effectiveness of the proposed technique, the wire cost is defined by wire volume. However, our approach can also be extended to handle other cost metrics. The proposed algorithm in Section 3 is denoted by NEW and the proposed algorithm for handling multiple current levels in Section 4 is denoted by NEW w/multicurrent. Note that the shown Cost and Delay values are averaged over 450 nets.

5.1 Optimality Study

An optimality study is performed to validate the proposed algorithm. NEW takes randomly generated small nets as the input without considering blockage. Various experiments have been performed and 50 nets with 3-5 pins are chosen. Refer to Table 1 for the results. The following observations are made:

- Due to the small size of the net, we can afford to solve ILP exactly and the optimal average cost is 41.
- NEW also works well on small nets. One can see the average cost by NEW is 42, which implies that NEW can almost optimally solve the problem on our small nets. No significant discrepancy in the runtime has been found. This is primarily due to the fact that both algorithms can run very fast on small nets. However, for large nets, ILP becomes computationally prohibitive while NEW can still solve them efficiently, which will be demonstrated in the next two sections.

5.2 Performance on Single Current Case

We would compare our NEW algorithm to ILP, as described in the above section, on large nets. However, ILP becomes computationally prohibitive for handling most of the nets. Thus, we choose to compare NEW with the state-of-the-art algorithm in [1]. The

TABLE 2
Comparison of Previous Work [1] and the New LP Algorithm without Blockage

	Lower Bound			Previous Work [1]			NEW			Ratio	
Sink number	Cost	Delay	CPU(s)	Cost	Delay	CPU(s)	Cost	Delay	CPU(s)	Cost	Delay
1-10	2626	29.6	0.01	2938	34.6	0.0002	2826	37.5	32.2	3.8%	-8.4%
11-20	7442	30.0	0.08	8719	64.6	0.001	7784	38.0	613.3	10.7%	41.2%
21-30	12169	26.3	0.21	14680	65.4	0.001	12822	36.2	2314.1	12.7%	44.6%

CPU refers to the runtime in seconds. Cost and delay refer to average value per net.

TABLE 3
Comparison of Previous Work [1] and the New LP Algorithm with Blockage

	Lower Bound			Previous Work [1]			NEW			Ratio	
Sink number	Cost	Delay	CPU(s)	Cost	Delay	CPU(s)	Cost	Delay	CPU(s)	Cost	Delay
1-10	2634	29.8	0.02	3103	36.4	0.0003	2890	36.9	0.221	6.9%	-1.4%
11-20	7444	30.0	0.09	9145	57.0	0.001	7855	38.2	4.491	14.1%	33.0%
21-30	12315	26.7	0.21	15128	68.0	0.001	12900	37.5	15.897	14.7%	44.9%

CPU refers to the runtime in seconds. Cost and delay refer to average value per net.

experimental results without blockage are shown in Table 2 and the results with blockage are shown in Table 3. Ratio is computed as 1 minus the ratio of the cost or delay of NEW to that of [1]. The following observations are made:

- Without considering blockage, NEW exhibits better performance in terms of cost as compared to [1] (refer to Table 2), especially for the large nets. For example, for the net with 21-30 sinks, NEW achieves $1 - 12{,}822/14{,}680 = 12.7\%$ reduction in cost compared to [1].
- With blockage, NEW also outperforms [1] (refer to Table 3) in cost. For example, for large nets, NEW achieves up to $1 - 12{,}900/15{,}128 = 14.7\%$ reduction in cost.
- NEW exhibits better performance in the delay for most nets compared to [1]. For large nets, NEW achieves up to 44.9 percent saving in delay. Although the results of NEW for the small nets are not as good as those of [1], there are only slight differences. For instance, the difference is only 1.4 percent for the nets with blockage. NEW significantly outperforms [1] in delay when the size of the net increases.
- Although the optimal ILP solutions cannot be computed, the lower bound of the optimal solutions can be obtained by solving the relaxed LP. NEW can compute solutions much closer to the optima in both the cases with and without blockage compared to [1].
- There exists a trade-off between the solution quality and runtime. It is true that [1] saves some runtime over NEW. However, its solution quality is much worse. As discussed above, the reduction in the cost by NEW is up to 14.7 percent and the reduction in delay by NEW is up to 44.9 percent. The significant improvement in solution quality by NEW clearly outweighs the runtime slowdown.

TABLE 4
Result of Multicurrent without Blockage

	Heuristic algorithm		NEW w/ multi-current		Ratio
sink number	Cost	CPU(s)	Cost	CPU(s)	Cost
1-10	2432	21	2303	43	5.3%
11-20	7320	62	6507	84	11.1%
21-30	12132	232	10820	271	10.8%

TABLE 5
Result of Multicurrent with Blockage

	Heuristic algorithm		NEW w/ multi-current		Ratio
Sink number	Cost	CPU(s)	Cost	CPU(s)	Cost
1-10	2485	23	2379	46	4.3%
11-20	7391	63	6548	85	11.4%
21-30	12697	246	11241	288	11.5%

CPU refers to the runtime in seconds.

5.3 Performance on Multiple Current Case

The proposed algorithm NEW w/multicurrent is then validated for the multicurrent case, which is important in analog VLSI design. To the best of our knowledge, this is the first work which directly handles the multiple current levels. NEW w/multicurrent is compared to the heuristic algorithm as described in the beginning of Section 4. The results are summarized in Tables 4 and 5. We make the following observations:

- As shown in Table 4, without blockage, NEW significantly outperforms the heuristic algorithm. For example, for the net with 11-20 sinks, NEW achieves the cost reduction by $1 - 6{,}507/7{,}320 = 11.1\%$ compared to the heuristic algorithm.
- With blockage, NEW is also much better than the heuristic algorithm, especially in large nets. For example, for the nets with 11-20 sinks, NEW achieves the cost reduction of 11.4 percent.

6 Conclusion

With fast technology scaling, a major design reliability concern arises from electromigration. In this paper, the minimum cost reliability-driven routing, which consists of Steiner tree construction and layer assignment, is considered. The problem is proven to be NP-hard and a highly effective iterative rounding-based integer linear programming algorithm is proposed. In particular, the first algorithm to directly handle multiple currents is designed, which is critical in analog VLSI design due to the existence of various current levels. Further, the new algorithm is extended to handle blockage, which makes it ready for practical use. Our experiments on 450 nets demonstrate that the new algorithm significantly outperforms the state-of-the-art work [1] with up to 14.7 percent wire reduction. In addition, the new algorithm can save 11.4 percent wires over a heuristic algorithm for handling multiple currents.

Acknowledgments

This work was supported in part by the Fundamental Research Funds for the Central Universities of China under the grant No. 78220021.

References

[1] J. Lienig and G. Jerke, "Current-Driven Wire Planning for Electromigration Avoidance in Analog Circuits," *Proc. IEEE Asia and South Pacific Design Automation Conf. (ASP-DAC),* pp. 783-788, 2003.

[2] J. Black, "Electromigration—A Brief Survey and Some Recent Results," *Proc. IEEE Int'l Reliability Physics Symp.,* pp. 338-347, 1968.

[3] S. Hu, Z. Li, and C. Alpert, “A Polynomial Time Approximation Scheme for Timing Constrained Minimum Cost Layer Assignment,” *Proc. IEEE/ACM Int'l Conf. Computer-Aided Design (ICCAD),* pp. 112-115, 2008.

[4] J. Lienig, G. Jerke, and T. Adler, “Electromigration Avoidance in Analog Circuits: Two Methodologies for Current-Driven Routing,” *Proc. IEEE Asia and South Pacific Design Automation Conf. (ASP-DAC),* pp. 372-380, 2002.

[5] G. Jerke, J. Lienig, and J. Scheible, “Reliability-Driven Layout Decompaction for Electromigration Failure Avoidance in Complex Mixed-Signal IC Designs,” *Proc. IEEE/ACM Design Automation Conf. (DAC),* pp. 181-184, 2004.

[6] G. Jerke and J. Lienig, “Hierarchical Current Density Verification in Arbitrarily Shaped Metallization Patterns of Analog Circuits,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems,* vol. 23, no. 1, pp. 80-90, Jan. 2004.

[7] G. Jerke and J. Lienig, “Hierarchical Current Density Verification for Electromigration Analysis in Arbitrarily Shaped Metallization Patterns of Analog Circuits,” *Proc. Design, Automation and Test in Europe (DATE) Conf. and Exhibition,* pp. 464-469, 2002.

[8] J. Lienig, “Introduction to Electromigration-Aware Physical Design,” *Proc. Int'l Symp. Physical Design (ISPD),* vol. 23, no. 1, pp. 80-90, 2004.

[9] M. Garey and D. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness.* W.H. Freeman, 1979.

[10] L. Behjat and A. Chiang, “Fast Integer Linear Programming Based Models for VLSI Global Routing,” *Proc. IEEE Int'l Symp. Circuits and Systems (ISCAS),* pp. 6238-6243, 2005.

[11] T. Terlaky, A. Vannelli, and H. Zhang, “On Routing in VLSI Design and Communication Networks,” *Discrete Applied Math.,* vol. 156, no. 11, pp. 2178-2194, 2008.

[12] S. Shah, A. Srivastava, D. Sharma, D. Sylvester, D. Blaauw, and V. Zolotov, “Discrete Vt Assignment and Gate Sizing Using a Self-Snapping Continuous Formulation,” *Proc. IEEE/ACM Int'l Conf. Computer-Aided Design (ICCAD),* pp. 705-712, 2005.