Hindawi Publishing Corporation
VLSI Design
Volume 2013, Article ID 905493, 12 pages
http://dx.doi.org/10.1155/2013/905493

Research Article

Power-Driven Global Routing for Multisupply Voltage Domains

Tai-Hsuan Wu, Azadeh Davoodi, and Jeffrey T. Linderoth

University of Wisconsin, Madison, WI 53706, USA

Correspondence should be addressed to Azadeh Davoodi; adavoodi@wisc.edu

Received 22 June 2012; Revised 16 November 2012; Accepted 5 April 2013

Academic Editor: Shantanu Dutt

This work presents a method for global routing (GR) to minimize power associated with global nets. We consider routing in designs with multiple supply voltages. Level converters are added to nets that connect driver cells to sink cells of higher supply voltage and are modeled as additional terminals of the nets during GR. Given an initial GR solution obtained with the objective of minimizing wirelength, we propose a GR method to detour nets to further save the power of global nets. When detouring routes via this procedure, overflow is not increased, and the increase in wirelength is bounded. The power saving opportunities include (1) reducing the area capacitance of the routes by detouring from the higher metal layers to the lower ones, (2) reducing the coupling capacitance between adjacent routes by distributing the congestion, and (3) considering different power weights for each segment of a routed net with level converters (to capture its corresponding supply voltage and activity factor). We present a mathematical formulation to capture these power saving opportunities and solve it using integer programming techniques. In our simulations, we show considerable saving in a power metric for GR, without any wirelength degradation.

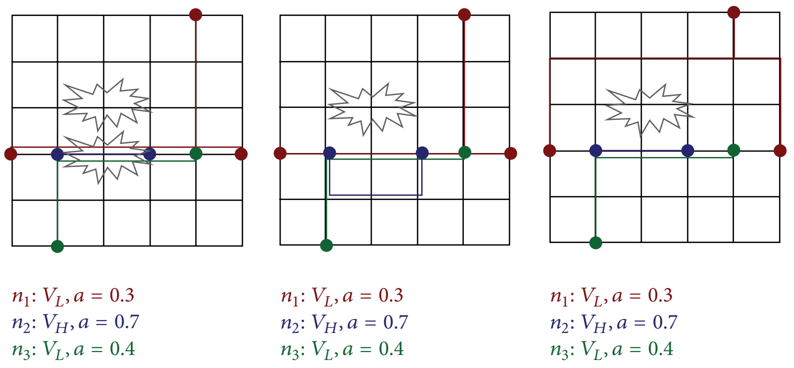
1. Introduction

Power consumption is a primary design objective in many application domains. Dynamic power still remains the dominant portion of the overall power spectrum. Design with multisupply voltage (MSV) allows significant reduction in dynamic power by taking advantage of its quadratic dependence on the supply voltage.

Dynamic power is dissipated in combinational and sequential logic cells, clock network, and the (remaining) local and global nets. We refer to the latter as the signal power. The signal power can take a significant portion of the dynamic power spectrum. For example, the contribution of the signal power is reported to be around 30% of dynamic power for a 45 nm high-performance microprocessor synthesized using a structured data paths design style and about 18% of the overall power spectrum [1].

The signals are complex structures in nanometer technologies that span over many metal layers. The power of a route segment depends on its width, metal layer, and spacing relative to its adjacent parallel-running routes. These factors determine the area, fringe, and coupling capacitances which impact power. Furthermore, in MSV designs, the power of a routed net depends on its corresponding supply voltage. For example, a route will have lower power if all its terminal cells have (the same) lower supply voltage. If a net connects a driver cell of lower voltage to a sink cell of higher voltage, its route includes a level converter (LC) and is decomposed into two segments of low and high supply voltages, corresponding to before and after the LC.

Figure 1 shows an example to motivate for power-aware global routing. Three nets are given in this example with a corresponding power supply (V_L low voltage or V_H high voltage). An activity factor is also given for each net. A net with higher power supply and activity factor consumes more power. Figure 1(a) shows that a shortest-length routing results in overflow in routing resources. The congested area is shown in the figure. Traditional GR is based on minimization overflow with minimal increase in wirelength. It is shown in this example in Figure 1(b), in which net n_2 is now 2 units longer; however, the congested area is eliminated. However, net n_2 has the highest power consumption (due to higher values of supply and activity factor). Making n_2 longer further increases its power consumption. Therefore, in power-aware GR which is shown in Figure 1(c), net n_2 has the shortest length, however, net n_1 instead is detoured to eliminate



(a) Shortest-length GR results in overflow (OF)

(b) Wirelength-based GR trades off increase in wirelength (WL) with decrease in OF

(c) Power-aware GR makes the high power consuming net shorter with minimal increase in WL and decrease in OF

Figure 1: Motivation for power-aware GR.

congestion. The wirelength of power-aware GR is higher than wirelength-based GR, but it has less signal power.

In this work, we propose a global routing (GR) method that optimizes the signal power in MSV designs. Figure 2 shows a generic design flow for a MSV-based GR. After placement and voltage assignment, the location and supply voltage of each cell are known. The supply voltage is determined either through voltage island generation [2, 3] or through a row-based assignment in a standard cell methodology. Furthermore, LCs are added to any net that connects a driver cell to a set of sink cells of higher supply voltage. Next, GR is applied to minimize the overall wirelength (WL), where the LCs are also included as terminals of a net.

For a given WL-optimized GR solution, we propose to further detour the nets in order to optimize the signal power. The signal power can be approximated during GR since at this stage the metal layers of each route segment are known. Furthermore, the spacing of parallel routes can be estimated from the routing congestion. Given a WL-optimized solution, the nets can be rerouted to trade off WL with power. For example, nets from higher metal layers can be routed to the lower ones for less wire widths and area capacitance. Nets can also be rerouted to spread the congestion, thereby increasing their spacing for less coupling capacitance. Activity factor and voltage can be incorporated as a power-weight for each route.

We present a mathematical formulation for MSV-based GR to minimize power and present integer programming-based techniques to solve the formulation. As part of power saving, our methods spread the routing congestion and ensure no additional overflow (of routing resources) and a bounded degradation in WL compared to the initial solution.

To the best of our knowledge, this is the first work of power-driven global routing in MSV designs. Recently, the work [4] discusses power-driven GR; however it does not consider the MSV case. Furthermore, it relies on the availability of *power-efficient candidate routes* for each net but generates such candidate routes quite heuristically. As part of the contributions of this work, we show a formal procedure to generate power-efficient candidate routes from the initial WL-optimized solution while taking into account the overall WL degradation and power saving. Also, recently the work [5] studies the GR problem for MSV domains, but it does not focus on routing for power minimization.

2. New Algorithmic Techniques Used

Power-aware routing can be considered as a new EDA problem. This is because the power of global interconnects (or signals) are starting to show nonnegligible contribution to the overall power spectrum for advanced technology nodes [1]. This issue is further exacerbated for multisupply voltage domains for which the power of a net dramatically changes depending on the voltage domain(s) that it (fully or partially) falls in. This work is the first to propose and formulate the power-aware routing for multi-supply voltage domains.

Furthermore, from an algorithmic perspective, the techniques offered in this work are a combination of integer programming with parallel processing based on problem decomposition. Integer programming allows obtaining a higher-quality solution compared to using heuristics as shown in [6]. However, it is not considered a suitable algorithmic venue for large-sized industry circuits. This work relies on decomposition of the routing problem intro smaller-sized and parallel-processed subproblems in order to make the use of integer programming possible for large-sized circuits.

3. Interconnect Modeling

In this section, we discuss an MSV-based GR model. We assume that the level converters (LCs) are placed for some of nets and the supply voltage of each cell is known.

3.1. Interconnect Modeling in MSV Designs. We are given a grid-graph $G = (\mathscr{V}, \mathscr{E})$ model of the GR problem, where each

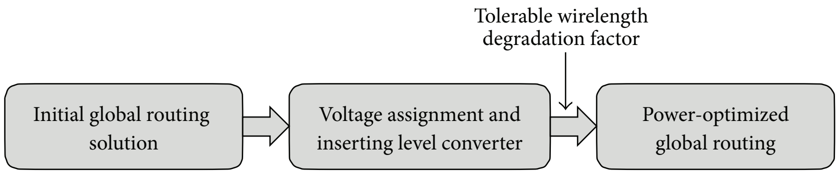

Tolerable wirelength degradation factor
Initial global routing solution
Voltage assignment and inserting level converter
Power-optimized global routing

Figure 2: Overview of GR with MSV.

vertex $v \in \mathscr{V}$ corresponds to a global bin containing a number of cells. Each edge $e \in \mathscr{E}$ represents the boundary of two adjacent bins. A capacity r_e is associated with each edge e, reflecting the maximum number of routes that can pass between two adjacent bins. A net $i \in \{1, \ldots, N\}$ is identified by its terminal cells, which are a subset of the vertices $\mathscr{V}$. In MSV-based GR, the terminals of a net may also be the LCs. During GR, a Steiner tree t_i in G is found for each net i to connect its terminals. The length of t_i is taken to be its wirelength (WL).

Figure 3(a) shows an example. The chip is divided into regions. Each region has either a low (V_L) or high (V_H) supply voltage. A routed net is specified in the figure. The net has one driver terminal with V_L voltage and three sink terminals of V_H voltage. The route includes two LCs which are also considered as additional terminals of the net.

For power-driven MSV-based GR, we first decompose a net which contains a LC into a set of subnets. We reroute each subnet as an individual net during power optimization. Consequently, we have $N_d > N$ number of nets after decomposition. For example, in Figure 3(b), the initial route is shown with its LCs. The net is decomposed into three subnets, each of which will be rerouted. The first subnet connects the driver terminal in V_L to the two LCs. The second one connects one LC to one V_H terminal. The third one connects the other LC to the other two V_H terminals.

Figures 3(c)–3(e) illustrate our net decomposition procedure. The decomposition of each net is done using its initial route and the location(s) of its level converter(s), assuming they are determined before this stage. For a net containing level converters, starting from its driver terminal, a subnet corresponding to a low supply voltage is formed that connects the driver terminal to a set of level converters and/or a set of sink terminals of the same supply voltage. Next, one or more subnets are formed that connect the level converters to the sink terminals of the same (and higher) voltage level. The BFS algorithm is utilized to traverse the initial route in our implementation. For example, in Figure 3(d), we start traversing from the source node until reaching the two level converters. All the touched edges form the first subnet n_1 which has a low supply voltage. Next, we continue traversing from each of the level converters individually until reaching all the sink nodes, using which the subnets n_2 and n_3 with high supply voltage are then identified.

Our net decomposition procedure is able to find a minimum number of subnets for each net that contains a level converter such that each subnet has only one corresponding supply voltage. Note that after rerouting the subnets, it is possible that these subnets may pass through the same edge(s) as shown in Figure 3(e). If the subnets which pass through the same edges have the same voltage level, (e.g., the subnets n_2 and n_3 in Figure 3(e)), then we can merge these subnets to release the overutilized routing resources. The above procedure is given for the case when two supply voltages V_L and V_H exist, which is also the case considered in this work. For higher number of voltage domains, the procedure can be extended in a similar way.

3.2. Power Modeling. Each decomposed net $i \in \{1, \ldots, N_d\}$ has a corresponding supply voltage V_i and switching activity α_i. The required interconnect power for a GR solution is estimated as

$$P = f_{clk} \times \left(\sum_{i=1}^{N_d} \alpha_i V_i^2 \left(C_i^{\text{sink}} + C_i^{\text{route}} \right) \right), \tag{1}$$

where f_{clk} is the frequency. As seen in (1), the capacitance of routed net i is the sum of the capacitances of its sink cells (denoted by C_i^{sink}) and of its route (denoted by C_i^{route}). Here C_i^{sink} is a constant that does not depend on the rerouting, so it is excluded from the optimization. Note that the power of the LCs are considered fixed and thus also not considered as part of the interconnect power optimization. The capacitance C_i^{route} for a routed net i is the sum of the capacitances of its unit-length edges that are contained in route t_i (given by notation $e \ni t_i$):

$$C_i^{\text{route}} = \sum_{e \ni t_i} C_e^u. \tag{2}$$

The parameter C_e^u is the capacitance of one routed edge $e \in \mathscr{E}$. This capacitance is a function of the metal layer l_e, wire width w_e, and wire spacing s_e of the edge e. Specifically,

$$C_e^u = Ca\left(l_e, w_e\right) + 2Cf\left(l_e, w_e, s_e\right) + 2Cc\left(l_e, w_e, s_e\right), \tag{3}$$

where Ca and Cf are the area and fringe capacitances with respect to substrate, and Cc is the coupling capacitance. As indicated, these capacitances are functions of wire length, width, and spacing and are provided by the technology library through a lookup table.

In this work, we assume that only one (and a different) wire width is associated with each metal layer, so we exclude the parameter w_e, and for each edge $e \in \mathscr{E}$, its metal layer l_e is known. The spacing for edge e is estimated from the edge utilization u_e in a GR solution. Given the utilization u_e and the length of edge e (computed from the chip dimension and the routing grid granularity), the spacing s_e is calculated to allow maximum spacing between its corresponding routes. Figure 4 shows an example for $u_e = 3$. This simple *averaging* strategy may be adjusted if more information is available at

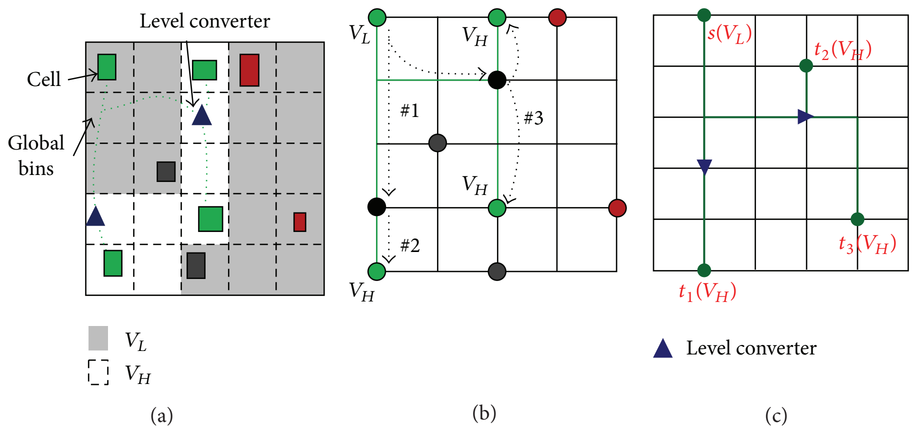
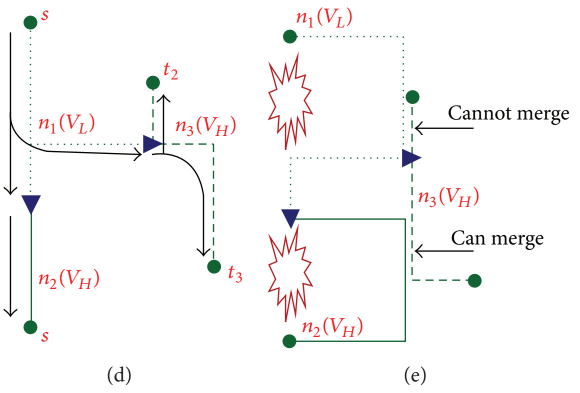



Figure 3: Graph modeling and net decomposition in global routing with level converters.

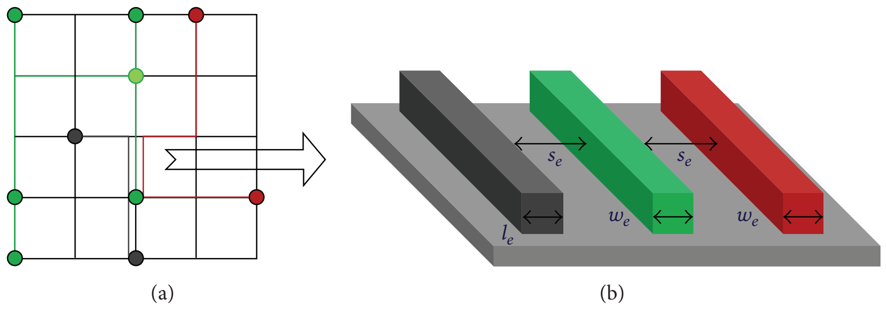


Figure 4: Modeling route capacitance on a GR edge.

the GR stage (e.g., the adjustment may be due to the fixed short nets which fall inside a single global routing bin). With this approximation, we can express the capacitance of a unit-length route edge in terms of the edge's metal layer and its utilization. The total capacitance of edge e is given by the product of the per-unit capacitance C_e^u and the utilization u_e: $C_e = C_e^u \times u_e$.

Figure 5(a) shows the curves representing area, fringe, and coupling capacitances for metal layer 1 with respect to edge utilization for a 45 nm library [7], assuming each GR edge is 2μ. The summation of the 3 capacitances (C_e^u) is shown in Figure 5(b).

4. Placement of Level Converters

The LCs may only be placed on the WL-optimized route, initially provided for each net. This ensures that the addition of LCs will not cause extra congestion; it allows connecting each LC to the initial route conveniently just by adding vias

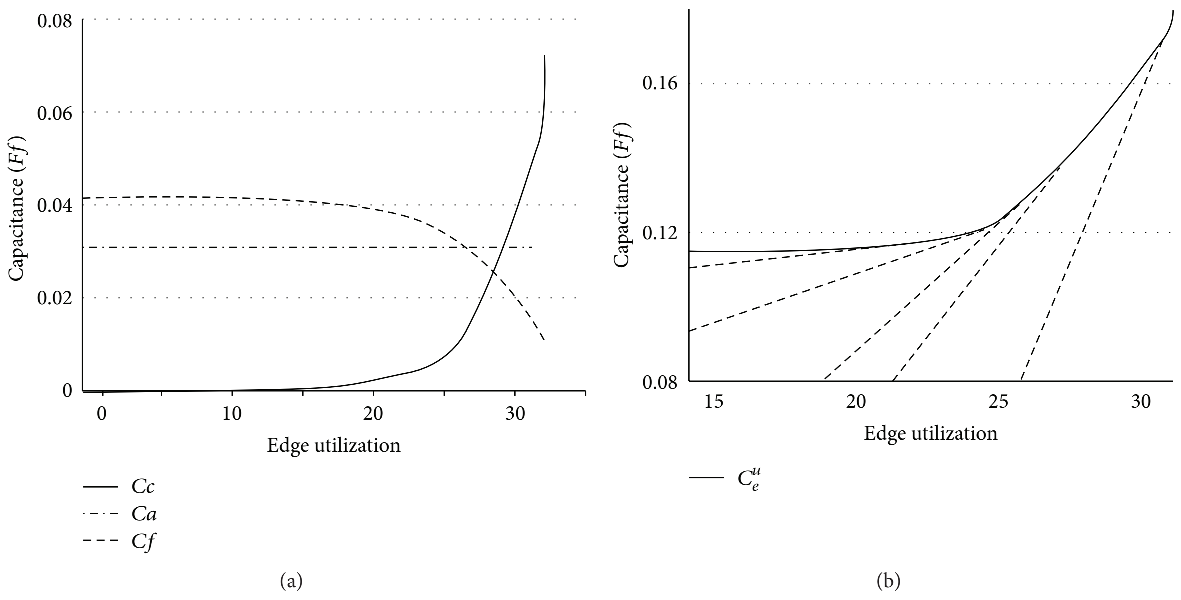


FIGURE 5: Dependence of three types of capacitance on edge utilization in metal layer 1.

from the LC to the initial route. Randomly placing the LCs may harm the GR congestion and degrade WL or overflow.

We list a set of requirements to identify valid LC insertion cases for a net i with given route t_i. We assume the net has a single source and may have multiple sink terminals.

(1) The location of LC is vertex v in $t_i (v \ni t_i)$.

(2) This vertex v should fall inside a V_H voltage island.

(3) The global bin corresponding to v should have enough space to add the LC. We denote the available space of v by A_v and compute it after placement (see Figure 2).

(4) For k vertices $v_1, \ldots, v_k$ satisfying the above 3 conditions, if all have the same distance to the source terminal (in terms of the number of edges on t_i), we require k LCs be added on these vertices simultaneously.

Figure 6 shows the set of potential LC locations of net i with initial route t_i. The source is the terminal in V_L island. Note that one vertex in t_i cannot be used because it is in the V_L island. We have four cases for valid LC insertion indicated by i_1, i_2, i_3, and i_4. In the latter case, two LCs should be placed on the net after the diverging point on the route to ensure that V_H is delivered to both sink terminals. For a single-source net i, we identify all the cases for valid LC insertion using a breadth first traversal on t_i and denote this set by $\mathscr{L}_i$. In this example $|\mathscr{L}_i| = 4$. For each case $l \in \mathscr{L}_i$, we further compute a corresponding power p_{il} using (1), where the edge utilization required to compute coupling capacitance is obtained from the initial WL-optimized solution. The power includes the interconnect portions on t_i and the LC(s).

To select one LC insertion case for each net, we define binary variable x_{il} to be equal to 1 if and only if case $l \in \mathscr{L}_i$ is selected for net i. The LC placement problem is expressed as the following integer program (IP) which can efficiently be solved using a solver, as we elaborate in our experiments:

$$\min_{x,s} \sum_{i=1}^{N} \sum_{l \in \mathscr{L}_i} p_{il} x_{il} + \sum_{i=1}^{N} M s_i, \qquad \text{(IP-LC)}$$

$$\begin{aligned} &\sum_{l \in \mathscr{L}_i} x_{il} + s_i = 1, \quad \forall i = 1, \ldots, N, \\ &\sum_{i=1}^{N} \sum_{l \in \mathscr{L}_i} a_{vl} x_{il} \le A_v, \quad \forall v \in V, \\ &s_i \ge 0, \quad \forall i = 1, \ldots, N, \\ &x_{il} = \{0, 1\}, \quad \forall i = 1, \ldots, N, \ \forall l \in \mathscr{L}_i, \end{aligned} \qquad (4)$$

where the parameter a_{vl} is equal to 1 if, in case l, an LC is placed at vertex v. The first set of constraints ensures at most one LC insertion case is selected for each net. The slack variable s_i will be positive if there is no available space for placing LCs for net i and is heavily penalized by positive M to maximize the number of placed LCs. The second constraints ensure LCs are placed in the free placement space.

In addition, it may not be possible to place LCs on any vertex v on the GR grid because its corresponding global bin is highly congested. We therefore associate for each vertex v a constant parameter A_v, indicating its available placement space. In our experiments, we calculate this available space for each global bin according to the placement density.

With this assumption, after adding an LC, the initial route can connect to the LC by extending through a set of vias at the LC location. Furthermore, for the WL-optimized tree t_i of net i, the potential locations of LCs are only allowed to be those vertices $v \in t_i$ which fall inside the V_H voltage islands as the LC should get connected to V_H voltage.

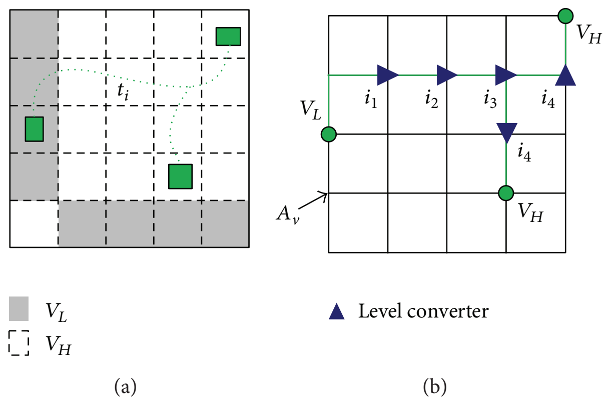


Figure 6: Valid LC locations for one net.

To enumerate all the possible LC insertion cases in a given route, consider a single-source net with WL-optimized route t_i. We enumerate and systematically identify all the cases for *valid* LC insertion according to the distance of vertex v from the source vertex on the route. The distance is measured in terms of the number of edges on the route between v and the source and obtained using breadth first traversal on tree t_i. We count each LC location $v \in t_i$ and in the V_H island as one possibility for LC placement on that route. However, if multiple vertices have the same distance to the source, we consider adding LCs on all such vertices simultaneously and count them as one possibility. For example, in case i_4 in Figure 6 we insert two LCs simultaneously. We then define a set $\mathscr{L}_i$ for each net i, indicating all the possibilities for its valid LC insertion. In the given example $|\mathscr{L}_i| = 4$. Furthermore, we compute the power p_{ij} for LC insertion case j for route i. The power is computed according to (1) where the edge utilizations used to compute coupling capacitance are computed from the provided WL-optimized solution.

5. Power-Driven MSV-Based GR

In this section, we first present a mathematical formulation of power-driven MSV-based GR. We then discuss integer programming-based techniques to obtain high-quality solutions to the formulation.

5.1. Mathematical Formulation. As described in Section 3.2, the per-unit capacitance of an edge $e(C_e^u)$ is a function of its metal layer and the edge utilization. Typically, this function is a convex increasing function, as depicted in Figure 5. We represent the function C_e^u by a set of line segments denoted by Q_e^u. For example, the set Q_e^u is composed of 7 line segments in the library used in this work [7]. Each line segment $q \in Q_e^u$ is of the form $m_q^u + r_q^u u_e$, for a given range of u_e, where m_q^u and r_q^u are derived from the library for that range. For each of the 8 metal layers in our library, the curve C_e^u is represented as 7 piecewise linear segments.

Since the per-unit capacitance is convex, its value may be expressed in our mathematical optimization problem for GR with the following set of linear inequalities:

$$m_q^u u_e + r_q^u \le C_e^u, \quad \forall q \in Q_e^u. \tag{5}$$

For a given edge utilization u_e, the corresponding C_e^u is obtained from the line equation that gives the largest value of $m_q^u u_e + r_q^u$ for $q \in Q_e^u$.

To model GR, we are given a routing grid graph $G = (\mathscr{V}, \mathscr{E})$, a set of decomposed multiterminal nets denoted by N_d, and edge capacities r_e. Let $\mathscr{T}_i$ be a collection of all Steiner trees that can route net i. We later discuss how to approximate $\mathscr{T}_i$ by generating a set of power-efficient candidate trees with consideration of WL degradation. Each tree $t \in \mathscr{T}_i$ is associated with a binary decision variable x_{it} which is equal to 1 if and only if it is selected to route net i. Let the parameter a_{te} be equal to 1 if tree t contains edge e (if $e \ni t$). The GR problem for power minimization is given by

$$\min_{x,s,C^u} \sum_{i=1}^{N_d} \sum_{t\in\mathscr{T}_i} \alpha_i V_i^2 \left(\sum_{e\ni t} C_e^u \right) x_{it} + \sum_{i=1}^{N_d} M s_i, \tag{PGR}$$

$$\sum_{t\in\mathscr{T}_i} x_{it} + s_i = 1, \quad \forall i = 1, \ldots, N_d,$$

$$\sum_{i=1}^{N_d} \sum_{t\in\mathscr{T}_i} a_{te} x_{it} \le r_e, \quad \forall e \in \mathscr{E},$$

$$m_q^u \left(\sum_{i=1}^{N_d} \sum_{t\in\mathscr{T}_i} a_{te} x_{it} \right) + b_q^u \le C_e^u, \quad \forall e \in \mathscr{E}, \ \forall q \in Q_e^u, \tag{6}$$

$$\sum_{i=1}^{N_d} \sum_{t\in\mathscr{T}_i} w_{it} x_{it} \le W_0 (1 + \beta),$$

$$s_i \ge 0, \quad \forall i = 1, \ldots, N_d,$$

$$x_{it} = \{0, 1\}, \quad \forall i = 1, \ldots, N_d, \ \forall t \in \mathscr{T}_i.$$

The first term in the expression of the objective function is the interconnect power as explained in Section 3.2. It includes activity α_i and voltage V_i of net i. The capacitance of a route t of net i is obtained by adding the unit edge capacitances C_e^u

for all the edges $e \ni t$. Here, the route $t \in \mathcal{T}_i$ will be selected for net i only if $x_{it} = 1$.

The first set of constraints selects at most one route for each net. The slack variable s_i is equal to 1 if net i cannot be routed, and the variable is penalized in the objective function by a large parameter M to maximize the number of routed nets. The term $\sum_{i=1}^{N_d} \sum_{t\in\mathcal{T}_i} a_{te}x_{it}$ represents the edge utilizations u_e. The second set of constraints ensures that the edge utilizations are within the given edge capacities. The third set of constraints determines the per-unit edge capacitance C_e^u for each edge e from its utilization, using the discussed piecewise linear model. The fourth constraint ensures the new wirelength is within a factor β of the initially-provided wirelength W_0. Here w_{it} denotes the wirelength of route t of net i.

The constraints of formulation (PGR) are all linear. However, the objective expression is nonlinear (due to the multiplication of variables x_{it} and C_e^u). We handle the nonlinearity in a heuristic manner using a two-phase approach. First, we choose a rerouting that attempts to minimize the total capacitance of all edges. Next, per-unit capacitances are estimated (and fixed) based on the solution of the first phase, and a rerouting is sought that minimizes the total estimated power. Each of these two phases becomes integer *linear* programs (IPs) which are discussed in the next sections.

5.2. Phase 1: Minimizing Total Capacitance. Using the piecewise linear approximation for the per-unit capacitance C_e^u given by (5), we may also approximate the total capacitance as

$$C_e = C_e^u \times u_e \geq m_q^u u_e^2 + r_q^u u_e, \quad \forall q \in Q_e^u. \tag{7}$$

This (convex) nonlinear expression may be relinearized, resulting in another piecewise linear expression for the total edge capacitance that may be used in our linear integer program for minimizing the total capacitance:

$$C_e \geq m_q u_e + r_q, \quad \forall q \in Q_e. \tag{8}$$

5.2.1. Formulation. The formulation of phase 1 is given by the following IP:

$$\min_{x,s,C} \sum_{\forall e\in\mathcal{E}} C_e + \sum_{i=1}^{N_d} M s_i, \qquad \text{(PGR-P1)}$$

$$\sum_{t\in\mathcal{T}_i} x_{it} + s_i = 1, \quad \forall i = 1,\ldots,N_d,$$

$$\sum_{i=1}^{N_d}\sum_{t\in\mathcal{T}_i} a_{te}x_{it} \leq r_e, \quad \forall e \in \mathcal{E},$$

$$m_q\left(\sum_{i=1}^{N_d}\sum_{t\in\mathcal{T}_i} a_{te}x_{it}\right) + b_q \leq C_e, \quad \forall e \in \mathcal{E},\ \forall q \in Q_e, \tag{9}$$

$$\sum_{i=1}^{N_d}\sum_{t\in\mathcal{T}_i} w_{it}x_{it} \leq W_0\left(1+\beta\right),$$

$$x_{it} = \{0,1\}, \quad \forall i = 1,\ldots,N_d,\ \forall t \in \mathcal{T}_i,$$

$$s_i \geq 0, \quad \forall i = 1,\ldots,N_d.$$

The objective expression is similar to formulation (PGR) but the first term is replaced by $\sum_{\forall e\in\mathcal{E}} C_e$ which represents an estimate of the total interconnect capacitance. The third set of constraints is also updated; the variable C_e replaces C_e^u in the previous formulation, and the coefficients in the piecewise linear model are updated to use (8).

5.2.2. A Price-and-Branch Solution Procedure. We approximately solve the (PGR-P1) using the two-step heuristics. First, a *pricing* procedure is used to generate a set of candidate routes for each net that are power-efficient while considering the WL degradation. The pricing step approximates $\mathcal{T}_i$ in the formulation to contain a small set of power-efficient candidate routes, instead of all the potential routes of net i. Second, *branch-and-bound* is applied to solve (PGR-P1), selecting one route for each net from the set of generated candidate routes. The standard branch and bound algorithm can be carried out using a commercial solver. This two-step procedure of generating candidate routes and then running branch and bound is commonly known as price and branch [8, 9]. The price and branch procedure was recently applied to solve the GR problem for WL improvement [6]. We apply the same procedure for power improvement. The major technical difference in our procedure is in the pricing step to find power-efficient candidate routes, which we next discuss in detail.

5.2.3. Overview of Pricing for Route Generation. We solve a linear-programming relaxation of (PGR-P1) by replacing the binary requirements on the variables x_{it} with constraints $0 \leq x_{it} \leq 1$ for all i, for all t. The linear program is solved by an iterative procedure known as column-generation [10]. In column generation, we start by replacing $\mathcal{T}_i$ (set of all possible routes of net i) in formulation (PGR-P1) by subset $\mathcal{S}_i \subset \mathcal{T}_i$, initially containing one candidate route per net. We then gradually expand $\mathcal{S}_i$, adding new routes that may decrease the objective function. Adding the new candidate routes is via a *power-aware pricing condition* for each net.

Before explaining the procedure in more detail, we first give the following notations:

(1) we refer to the LP relaxation of (PGR-P1) in which $\mathcal{T}_i$ is replaced by $\mathcal{S}_i$ and $0 \leq x_{it} \leq 1$ by the "restricted master problem" denoted by (RMLP-P1); the solution of (RMLP-P1) for a given $\mathcal{S}_i$ is denoted by $(\widehat{x}, \widehat{s}, \widehat{C})$;

(2) we refer to the dual of the restricted master problem by (D-RMLP-P1). The solution of (D-RMLP-P1) consists of ($\widehat{\lambda} \leq M$, $\widehat{\pi} \leq 0$, $\widehat{\mu} \geq 0$, $\widehat{\theta} \leq 0$), corresponding to the dual variables for the first, second, and third set of constraints in the relaxed (PGR-P1), respectively.

The iterative column generation procedure including the pricing condition is enumerated below.

(1) For each net $i = \{1,\ldots,N_d\}$, initialize $\mathcal{S}_i$ with one route. (In this work we start with the solution of [11].)

(2) Solve (RMLP-P1), yielding a primal solution $(\widehat{x}, \widehat{s}, \widehat{C})$ and dual values $(\widehat{\lambda}, \widehat{\pi}, \widehat{\mu}, \widehat{\theta})$ in (D-RMLP-P1).

(3) Generate a new route t^* for net $i = \{1,\ldots,N_d\}$. Using the solution of step 2, evaluate the pricing condition:

if $\widehat{\lambda}_i > \sum_{e\in t^*}\sum_{q\in Q_e} m_q\widehat{\mu}_{eq} - \sum_{e\ni t^*}(\widehat{\pi}_e + \widehat{\theta})$, then $\mathcal{S}_i = \mathcal{S}_i \cup \{t^*\}$.

(4) If an improving route for some net i was found in step 3, return to step 1. Otherwise, stop—*the solution* $(\widehat{x}, \widehat{s}, \widehat{C})$ *is an optimal solution to* (RMLP-P1).

Step 3 gives the pricing condition in terms of the solution of the dual problem (D-RMLP-P1) obtained at the current iteration. This step can determine for a given new route t^* if it should be added to the set $\mathcal{S}_i$ to reduce the objective of (RMLP-P1). However, it does not specify how a new route should be found such that the pricing condition gets satisfied. We discuss a convenient graph-based procedure to generate new route t^* which satisfies the pricing condition.

5.2.4. Route Generation for One Net. To find improving routes for net i, we associate a weight w_e for edge e in the GR grid as

$$w_e = \max_{q\in Q_e}\left(m_q\widehat{\mu}_{eq}\right) - \widehat{\pi}_e - \widehat{\theta}. \tag{10}$$

By the theory of linear programming, for each edge e, at most one dual variable μ_{eq}, $q \in Q_e$ will be positive in an optimal solution to (D-RMLP-P1). Thus, considering route t^*, we can compute the pricing condition as $\widehat{\lambda}_i > \sum_{\forall e\ni t^*} w_e$. We take advantage of this interpretation to identify promising route t^* which satisfies the pricing condition. Given a route $t \in \mathcal{S}_i$ obtained from previous iterations, we obtain t^* by rerouting branches of t with the updated edge weights so that the overall weights of rerouted branches are reduced.

We explain the procedure with the example of Figure 7. Considering two nets a and b, suppose we are initially given the routes t_a and t_b for these two nets. After step 2 at the first iteration of column generation, we obtain edge weights which are given in Figure 7(a). To obtain a new route t_a^* for net a, we reroute different branches of t_a. For each terminal, we identify a branch as the segment connecting it to the first Steiner point on t_a. We then reroute this branch by solving Dijkstra's single-source shortest path algorithm [12] on the weighted graph with the weights of the first iteration, similar to [13, 14]. The route t_a^* is shown in Figure 7(b). After adding t_a^* to $\mathcal{S}_a$, we proceed to the second iteration and obtain new edge weights which are shown in Figure 7(b).

The discussed pricing procedure is similar to [6]. However, it differs in the pricing condition and the way edge weights are set up. For solving (RMLP-P1) and its dual at each iteration, we use the solver CPLEX 12.0. After obtaining the final set $\mathcal{S}_i$, again we use CPLEX 12.0 for the branch and bound step to get the final solution. We further accelerate the process by applying a simple problem decomposition that we will discuss in Section 5.4.

5.3. Phase 2: Considering Activity and Voltage. At phase 2, we approximate the per-unit edge capacitances using the solution from phase 1 and reroute the nets to minimize an approximation of the total power. Since the utilization (and hence capacitance) corresponding to the routing solution of phase 2 may be different from phase 1, we heavily penalize any mismatch in our optimization.

5.3.1. Formulation. We compute the following quantities after phase 1.

(1) We define a new "effective" capacity for each edge e as $\widetilde{r}_e = \sum_{i=1}^{N_d}\sum_{t\in\mathcal{T}_i} a_{te}\widetilde{x}_{it}$, where $\widetilde{x}_{it}$ is the value of the routing solution from phase 1.

(2) We define the new per-unit capacitance as $\widetilde{C}_e^u = \widetilde{C}_e/\widetilde{r}_e$, where $\widetilde{C}_e$ is the value of the edge capacitance from the solution found in phase 1.

With these definitions, the formulation of phase 2 is the following integer linear program:

$$\min_{x,s,\epsilon}\sum_{i=1}^{N_d}\sum_{t\in\mathcal{T}_i}\alpha_i V_i^2\left(\sum_{e\ni t}\widetilde{C}_e^u\right)x_{it} + \sum_{i=1}^{N_d} M_1 s_i + \sum_{\forall e\in\mathcal{E}} M_2\epsilon_e, \tag{PGR-P2}$$

$$\begin{gathered}\sum_{t\in\mathcal{T}_i} x_{it} + s_i = 1, \quad \forall i = 1,\dots,N_d,\\ \sum_{i=1}^{N_d}\sum_{t\in\mathcal{T}_i} a_{te}x_{it} \le \widetilde{r}_e + \epsilon_e, \quad \forall e \in \mathcal{E},\\ \sum_{i=1}^{N_d}\sum_{t\in\mathcal{T}_i} w_{it}x_{it} \le W_0\left(1+\beta\right),\\ 0 \le \epsilon_e \le r_e - \widetilde{r}_e, \quad \forall e \in \mathcal{E},\\ x_{it} = \{0,1\}, \quad \forall i = 1,\dots,N_d,\ \forall t \in \mathcal{T}_i,\\ s_i \ge 0, \quad \forall i = 1,\dots,N_d.\end{gathered} \tag{11}$$

The first term in the objective expression is summation of an estimate of the power of the nets, where $(\sum_{e\ni t}\widetilde{C}_e^u)$ is the fixed approximate per-unit capacitance of edge e which contains route t and is obtained using the solution of phase 1 as discussed before. The first set of constraints ensures that at most one route is selected per net; otherwise, a heavy penalty of M_1 is associated if $s_i \ne 0$, and this is reflected in the second term of the objective function. The second set of constraints enforces the new utilization of each edge to be $\widetilde{r}_e + \epsilon_e$, where ϵ_e is a new variable which is heavily penalized by a large factor M_2 in the objective function if $\epsilon_e \ne 0$. In other words, we highly penalize if the rerouting of a net causes a larger edge utilization compared to phase 1. This in effect forces the routing process to keep the *mismatch* in the edge utilizations as small as possible which translates in the capacitance (which is function of utilization) to remain close to phase 1. We also enforce $\epsilon_e + \widetilde{r}_e \le r_e$ to ensure that the edge utilization is not beyond its actual capacity r_e in the fourth set of constraints. Finally, the third set of constraints ensures that the increase in wirelength is bounded by factor β.

5.3.2. Solving Using Price and Branch. The solution procedure is quite similar to the one explained in the previous Section 5.2 for phase 1. Here, we just note the differences. We denote the restricted master problem by (RMLP-P2) and its solution by $(\widehat{x}, \widehat{s}, \widehat{\epsilon})$. The dual of the restricted master is denoted by (D-RMLP-P2) and its solution is $(\widehat{\lambda}, \widehat{\pi}, \widehat{\theta})$,

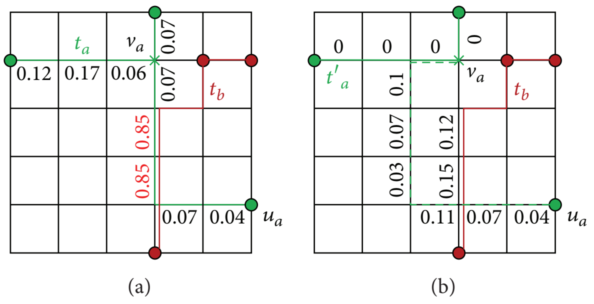


FIGURE 7: Power-aware route generation.

corresponding to the first, second, and third set of inequalities in relaxed (PGR-P2), respectively.

The initial set $\mathcal{S}_i$ is set to *all the candidate routes generated from phase 1*. This helps to quickly generate a high-quality solution for phase 2. It also ensures that the solution of phase 1 is included as a *feasible* solution in phase 2.

The pricing condition is given by the following inequality $\widehat{\lambda}_i > \alpha_i V_i^2(\sum_{e\ni t} C_e^u) - \sum_{e\in t}(\widehat{\pi}_e + \widehat{\theta})$ and is used to define the edge weights given by $w_e = \alpha_i V_i C_e^u - \widehat{\pi}_e - \widehat{\theta}$, for all $e \in \mathcal{E}$.

5.4. Decomposition. To accelerate solving the two-phase formulation, we apply a simple problem decomposition. We recursively divide the chip into a set of rectangular subregions while balancing the total number of nets that fall inside each subregion. We use the initial WL-optimized solution of [11] to guide this process. We stop when the number of nets at each subregion is at most 3000, which we empirically determined for our experimented benchmarks from the ISPD2008 suite [15].

Each subproblem is then defined as one rectangular subregion with the set of nets assigned to it. If a net passes from multiple subregions, we force the terminal location on the subregion boundary to be fixed from the initial WL-optimized solution. This allows independent solving of each subproblem without the hassle of later connecting the segments of a route in adjacent subregions. The subproblems are then (one-time) parallel-solved to get the final solution. Figure 8 shows an example.

Even though in our decomposition each subproblem in effect is assigned a low or high voltage level, it is possible that the nets assigned to it have different supply levels. For example, a high voltage net may just pass from a subproblem in a low voltage island, or a net with level converter (which will have portions of high and low voltage levels after net decomposition) may fall in a high voltage island.

Please note, the main difference between our decomposition procedure and [6] is the use of the initial WL-optimized solution to fix the terminal locations on the subregion boundaries and thus avoid later connecting adjacent subproblems.

Overall this decomposition is extended from PGRIP [16], but we make use of our initially provided global routing solution for more effective decomposition to determine the fixed terminal locations on the boundaries for independent and parallel processing of the subproblems.

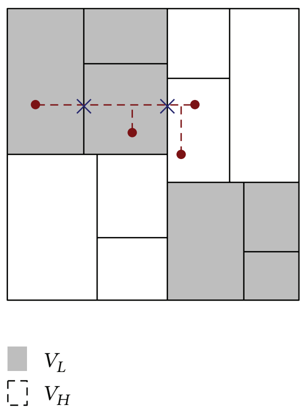


FIGURE 8: Decomposition into independent subproblems.

6. Simulation Results

6.1. Benchmark Instances. In order to test our solution procedure and determine whether or not significant power savings were possible without increasing wirelength, we modified known benchmarks to include multisupply voltages. Modifying the benchmarks required us to generate timing data and power data and place level converters. We implemented the procedure of [2] to generate voltage islands for two voltage levels of $V_L = 0.9$ V and $V_H = 1.1$ V. The procedure required a sequential netlist with gate-level delay and power models.

Timing Modeling. We assumed the locations of the sequential elements in the ISPD 2008 benchmarks using the following procedure. First, we obtained a directed acyclic graph (DAG) representation of the benchmarks from the variation provided by the ISPD 2006 placement benchmarks [17]. Using the placement benchmarks, we obtained a DAG by starting from the designated primary inputs and traversing in forward direction until reaching the primary outputs. We also assumed the nets with more than 50 terminals to be clock trees to identify sequential elements.

We then assumed that the delay of each cell (or node in the DAG) is proportional to its size (for unit load) where the unit delay was assumed to be of the inverter of the 45 nm library [7] used in this work. We considered loading in our cell delay modeling to be proportional to the cell size which was also given in the placement benchmarks.

Power Modeling. We randomly and uniformly generated the activity factors of each net to be between 0.1 and 0.9. The 45 nm library used in this work contained information about the total capacitance (area, fringe, and coupling) for each of the 8 metal layers. We used the method described in Section 5 to extract piecewise linear model for C_e and C_e^u for each of the 8 metal layers. For each metal layer, we considered the minimum wire size given in the library. To map edge utilization to spacing, we assumed the length of each edge of the GR grid to be 2μ; for a given utilization we assumed the maximum spacing between the routes mapped to the same GR edge.

Level Converter Placement. After voltage island generation, we needed to decide the locations of the level converters (LCs). (The procedure in [2] did not specify these locations.) For simplicity, we inserted the LCs on the initial WL-optimized solution that was taken from [11]. The LCs were inserted for any net that had a source terminal driving one or more sink terminals. The procedure minimized the number of LCs and placed them as close as possible to the sink terminals, subject to the available whitespace. The whitespace inside each global bin was derived by evaluating (both) the placement and GR variations of the ISPD benchmarks.

6.2. Level Converter Placement. In our first experiment, we report the result from our level converter placement algorithm for the nets that contained a level converter (had a source terminal in V_L island with fanout terminals in V_H islands). We consider the following case in our experiment: we routed all the nets using the initial wirelength-optimized solution of NTHU-Route2.0 [11]. We solve our formulation (IP-LC) to obtain the level converter locations subject to the area density constraints. We consider the obtained results as the base case for power comparison in our second experiment.

Recall the placement of level converters can impact the power of each route by decomposing it into multiple segments where each segment has a high or low supply level. Using (1), we compute the total power of the nets which need level conversion. This includes the power of level converters and the different routes segments of the decomposed nets after inserting the level converters.

Table 1 reports our power comparison results. We report the total number of nets and the number of nets which require level conversion in columns 2 and 3, respectively, for each benchmark. The total number of level converters in our case is given in column 4. The number of level converters is larger than column 3, indicating that for some nets it may be better to add extra LCs but place them closer to the sink terminals to reduce the route portion that is driven by high voltage and save power. In column 5, we report the power of ([11] + LC) for the nets including the ones with level conversion. We use these power numbers as the base case for our next experiment. Finally, the wall clock time of the level converter placement (indicated by WCPU) is given in column 6. As can be seen this step is done very quickly.

6.3. Power-Aware Global Routing. In this experiment, we used the initial WL-optimized solution of [11], and after fixing the locations of LCs, we applied net decomposition (as described in Section 3.1). We then solved two IPs corresponding to the formulations given in phase 1 and phase 2 for each subproblem using CPLEX 12.0 [18]. The number of subproblems is listed in Table 2 column 5 (indicated by SP number) which ranged from 130 to 670 among the benchmarks. These IPs were solved on the computer-aided engineering (CAE) grid at the University of Wisconsin Madison. Each machine had 2 GB of memory. All IPs were submitted to HTCondor [19] which manages the computers in a shared environment. HTCondor then assigned the jobs for parallel processing to the available machines.

Table 2 reports the number of nets, decomposed nets, and LCs in columns 2, 3, 4, and respectively. We then applied our power-driven GR procedure using a wirelength degradation factor of $\beta = 0$, so *no* wirelength degradation was allowed.

We then compared three routing solutions:

(i) the initial WL-optimized solution of [11];

(ii) the solution after applying phase 1, obtained by solving the formulation (PGR-P1);

(iii) the solution by further applying phase 2, obtained by solving (PGR-P1) followed by (PGR-P2).

For each case, we report the wirelength (WL), the total capacitance (C) ($\sum_{i=1}^{N_d} C_i^{\text{route}}$, where C_i^{route} is defined in (2)), given in units fF, and the GR power metric P from (1), excluding the constant portions of the expression.

The results are reported in Table 2 in columns 6 to 14. For the initial solution, we report the wirelength (W_0) of the NTHU-R2.0 routes that have been augmented with the extra via-only segment(s) to connect the LC(s) to the original routes. (As a result, there is slight increase in wirelength compared to the numbers reported in the work [11].) For the solutions of phase 1 and phase 2, we report only the percentage *improvement* in WL, C, and P, all with respect to the initial solution.

As can be seen, applying phase 1 of the power-reduction heuristic results in significant saving of 8.77% in P. Recall, the savings are solely due to capacitance reduction (as can be seen from the higher improvement rate in C compared to P). By further applying phase 2, we see additional improvement in P (on average 16.70%). The improvement in C is slightly larger than phase 1, even though phase 1 solely focuses on optimizing C. This is because we start phase 2 by including all the candidate routes generated from phase 1. Notice that in both phase 1 and phase 2 there is an improvement (reduction) in WL compared to W_0. It is important to note that no extra overflow was introduced in the power-optimized solutions.

In our simulations, we explicitly bounded the runtime for phase 1 and phase 2. The wall clock runtime of all benchmarks for phase 1 and phase 2 was set to 30 min and 40 min,

Table 1: Results of the level converter placement for the ISPD 2008 benchmarks.

Bench	Net number	Net_{LC} number	LC number	Power	WCPU (min)
Adaptec1	177K	9K	20K	432242	5
Adaptec2	208K	8K	17K	336881	7
Adaptec3	368K	17K	43K	1056778	8
Adaptec4	401K	16K	36K	751120	13
Adaptec5	548K	32K	85K	1199591	11
Newblue1	271K	75K	16K	318922	10
Newblue2	374K	22K	47K	453234	17
Newblue4	531K	38K	79K	927712	9
Newblue5	892K	26K	84K	1469859	14
Newblue6	835K	31K	91K	1367000	17
Newblue7	1647K	28K	72K	2201835	21
Bigblue1	197K	9K	26K	619321	6
Bigblue2	429K	15K	43K	560723	13
Bigblue3	666K	23K	60K	814957	12
Bigblue4	1134K	17K	51K	1254323	15

Table 2: Results for ISPD 2008 benchmarks. The WL is scaled to 10^5. Power and cap. are scaled to 10^3.

Bench	Net number	Net_d number	LC number	SP number	Initial solution ([11] + LC)			Phase 1			Phase 1 + phase 2		
					W_0	C	P	$-$WL (%)	$-C$ (%)	$-P$ (%)	$-$WL (%)	$-C$ (%)	$-P$ (%)
Adaptec1	177K	197K	20K	130	54.2	953.3	432.2	0.05	11.70	8.57	0.07	15.48	16.17
Adaptec2	208K	224K	17K	195	53.0	750.0	336.9	0.12	10.34	6.93	0.14	14.57	15.13
Adaptec3	368K	411K	43K	359	132.7	2187.0	1056.8	0.01	11.51	8.67	0.34	13.55	13.94
Adaptec4	401K	437K	36K	296	123.0	1613.8	751.1	0.02	12.16	8.46	0.04	16.92	17.20
Adaptec5	548K	632K	85K	454	158.7	2543.0	1199.6	0.38	8.60	6.08	0.43	10.23	10.88
Newblue1	271K	287K	16K	195	47.0	612.2	318.9	0.11	13.39	9.87	0.22	17.45	18.40
Newblue2	374K	421K	47K	312	77.6	894.9	453.2	0.04	14.19	7.87	0.09	19.20	19.34
Newblue4	531K	610K	79K	462	133.7	1955.4	927.7	0.02	13.39	9.61	0.54	17.45	17.61
Newblue5	892K	975K	84K	658	234.7	3405.3	1469.9	0.89	11.55	6.75	0.86	14.00	13.47
Newblue6	835K	926K	91K	532	180.2	2834.9	1367.0	0.62	12.56	9.35	0.57	16.35	17.80
Newblue7	1647K	1719K	72K	670	360.2	5004.4	2201.8	0.01	15.12	11.20	0.17	19.63	20.93
Bigblue1	197K	222K	26K	152	57.0	1110.4	619.3	0.23	10.68	7.20	0.16	12.17	12.56
Bigblue2	429K	472K	43K	275	92.4	1283.8	560.7	0.14	11.64	7.86	0.10	14.76	14.33
Bigblue3	666K	725K	60K	453	133.0	1664.6	815.0	0.91	15.22	10.99	0.93	20.25	20.31
Bigblue4	1134K	1184K	51K	509	233.0	3006.6	1254.3	0.18	16.03	12.12	0.28	22.31	22.46
Avg.								0.25	12.54	8.77	0.34	16.29	16.70

respectively. The number of processors used for parallel processing of the subproblems was upper bounded by the number of subproblems, for example, up to 130 simultaneously processed jobs in benchmark `adaptec1`; the exact number of parallel jobs is not known and depended on the number of free machines in our computational grid (which depended on the number of users of the grid when the simulations ran) as well as HTCondor's internal procedure to schedule jobs to available resources which considers factors such as user priority and past usage history. Furthermore, HTCondor resource management policy ensured that each machine ran at most one job at each time, so the machines were solely dedicated to solving the subproblems when utilized by us.

In an ideal situation (i.e., a grid which can support simultaneous runs of all the subproblems), the wall clock time of our tool will be 30 min and 40 min (for phases 1 and 2, resp.) for a total sum of 70 min for each of the benchmarks. We note, in this work unlike PGRIP [16], our decomposition procedure creates *independent* subproblems so there will not be any communication between the subproblems.

7. Conclusions

We proposed a formulation for minimizing an interconnect power metric for global routing for design with multi-supply voltage. Power minimization is after an initial wirelength-optimized solution is obtained. We presented a mathematical formulation which considered power saving opportunities by reducing the area, fringe, and congestion-dependent coupling capacitances at each metal layer, while accounting for the activity and supply voltage of each route segment. We showed significant savings in the power metric for global routing without any degradation in wirelength or overflow.

References

[1] R. S. Shelar and M. Patyra, "Impact of local interconnects on timing and power in a high performance processor," in *ACM International Symposium on Physical Design*, pp. 145–152, 2010.

[2] R. L. S. Ching, E. F. Y. Young, K. C. K. Leung, and C. C. N. Chu, "Post-placement voltage island generation," in *Proceedings of the International Conference on Computer-Aided Design (ICCAD '06)*, pp. 641–646, November 2006.

[3] L. Guo, Y. Cai, Q. Zhou, and X. Hong, "Logic and layout aware voltage island generation for low power design," in *Proceedings of the Asia and South Pacific Design Automation Conference*, pp. 666–671, January 2007.

[4] H. Shojaei, T.-H. Wu, A. Davoodi, and T. Basten, "A Pareto-algebraic framework for signal power optimization in global routing," in *Proceedings of the 16th ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED '10)*, pp. 407–412, August 2010.

[5] W.-H. Liu, Y.-L. Li, and K.-Y. Chao, "High-quality global routing for multiple dynamic supply voltage designs," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD '11)*, pp. 263–269, November 2011.

[6] T.-H. Wu, A. Davoodi, and J. T. Linderoth, "GRIP: scalable 3D global routing using integer programming," in *Proceedings of the 46th ACM/IEEE Design Automation Conference (DAC '09)*, pp. 320–325, July 2009.

[7] Nangate 45 nm open cell library, 2008, http://www.nangate.com/.

[8] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance, "Branch-and-price: column generation for solving huge integer programs," *Operations Research*, vol. 46, no. 3, pp. 316–329, 1998.

[9] D. G. Jørgensen and M. Meyling, "A branch-and-price algorithm for switch-box routing," *Networks*, vol. 40, no. 1, pp. 13–26, 2002.

[10] G. Dantzig and P. Wolfe, "Decomposition principle for linear programs," *Operations Research*, vol. 8, pp. 101–111, 1960.

[11] Y.-J. Chang, Y.-T. Lee, and T.-C. Wang, "NTHU-route 2.0: a fast and stable global router," in *Proceedings of the International Conference on Computer-Aided Design (ICCAD '08)*, pp. 338–343, November 2008.

[12] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[13] M. Pan and C. C. N. Chu, "FastRoute 2.0: a high-quality and efficient global router," in *Proceedings of the Asia and South Pacific Design Automation Conference*, pp. 250–255, January 2007.

[14] J. A. Roy and I. L. Markov, "High-performance routing at the nanometer scale," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 6, pp. 1066–1077, 2008.

[15] "ISPD 2008 global routing contest and benchmark suite," http://www.sigda.org/ispd2008/contests/ispd08rc.html.

[16] T.-H. Wu, A. Davoodi, and J. T. Linderoth, "A parallel integer programming approach to global routing," in *Proceedings of the 47th Design Automation Conference (DAC '10)*, pp. 194–199, June 2010.

[17] ISPD 2006 placement contest and benchmark suite.

[18] CPLEX Optimization, *Using the CPLEX Callable Library, Version 9*, Incline Village, Nev, USA, 2005.

[19] M. J. Litzkow, M. Livny, and M. W. Mutka, "Condor—a hunter of idle workstations," in *Proceedings of the 8th International Conference on Distributed Computing Systems*, pp. 104–111, 1998.