MSV-Driven Floorplanning

Qiang Ma, Zaichen Qian, Evangeline F.Y. Young, and Hai Zhou

***Abstract*—Power consumption has become a crucial problem in modern circuit design. Multiple supply voltage (MSV) design is introduced to provide higher flexibility in controlling the power and performance tradeoff. One important requirement of MSV design is that timing constraints of the circuit must be satisfied after voltage assignment of the cells. In this article, we develop two algorithms to solve the voltage assignment problem under timing constraints, namely, min-cost flow (MCF) and value-oriented branch-and-bound (VOBB). In the MCF algorithm, the voltage assignment problem is formulated as a convex cost dual network flow problem, and can be solved optimally in polynomial time under certain conditions by calling a MCF solver. The VOBB algorithm, which is a VOBB-based searching method, solves the voltage assignment problem optimally in general cases by employing the MCF algorithm and a linear programming solver as subroutines. At last, we propose a MSV-driven floorplanning framework that optimizes power consumption and physical layout of a circuit simultaneously during the floorplanning stage, by embedding the MCF algorithm into a simulated annealing-based floorplanner and applying the VOBB algorithm as a postprocessing step. We compared our approach with the latest works on this problem, and the experimental results show that, using our approach, significant improvement on power saving can be achieved in much less running time, which confirms the effectiveness and efficiency of our method.**

***Index Terms*—Algorithms, floorplanning, linear programming, min-cost flow, multi-supply voltage.**

I. Introduction

POWER optimization has become one of the most important issues to be addressed in modern circuit design because of the increasing power density and the wide use of portable systems. Multiple supply voltage (MSV) design [9] was introduced as an effective mean to reduce both dynamic and static power. In MSV designs, the timing slacks of the cells are traded for power saving by assigning appropriate supply voltage levels to the cells while still satisfying the timing constraints. Basically, high voltage level is assigned to critical cells and low voltage level is assigned to noncritical cells, so that power can be saved without violating the timing constraints. Therefore, an effective voltage assignment method is desirable to minimize power consumption under timing requirements. Moreover, it is beneficial to consider the problem of voltage assignment, voltage island generation, and floorplanning simultaneously under the timing, power, area, and other physical constraints, since these steps will significantly affect each other.

A number of previous papers addressed the voltage assignment and island generation problem in floorplanning and placement. In these previous papers, MSV is considered at various design stages, including the floorplanning and placement stages [7], [8], [10], [12], [13], as well as the post-floorplanning and post-placement stages [4], [11], [15], [19], [20]. For the latter category, Wu *et al.* [19] and Ching *et al* [4] targeted at minimizing the number of voltage islands generated, and Mak *et al.* [15] formulated the voltage assignment and island generation problem as an integer linear program (ILP), but none of them had taken timing constraints into consideration explicitly. In [20], Wu *et al.* proposed an approximation algorithm based on a *zero slack algorithm* to minimize the power cost and to simplify the power network resource under timing constraints; Lee *et al.* [11] tackled the voltage assignment problem using an ILP-based approach with a set of linear inequalities representing the timing requirements. An inevitable deficiency of all these post-floorplanning and post-placement works is that physical layout and power optimization are done separately, hence the solution quality is restricted to a certain extent. Concurrent physical layout and power optimization will be much more favorable. Hu *et al.* [7] and Ma *et al.* [13] considered this simultaneous voltage assignment, voltage island generation, and floorplanning problem, however, neither of them explicitly considered timing in their problem formulation. In one recent paper [10], Lee *et al.* handled the voltage assignment problem under timing constraints with a dynamic programming heuristic, and applies a power network aware floorplanner afterward. However, physical information is not taken into account in their voltage assignment step; also, the length of an interconnect is restricted to stay within a certain range during floorplanning, which will degrade the solution quality.

In this paper, we propose two algorithms to solve the voltage assignment problem under timing constraints, namely, the min-cost flow (MCF) algorithm and the value-oriented branch-and-bound (VOBB) algorithm. MCF is a fast approximation algorithm while VOBB is a slower but optimal algorithm.

Manuscript received July 26, 2010; revised October 16, 2010 and January 7, 2011; accepted February 22, 2011. Date of current version July 20, 2011. This work was supported in part by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China, under Project 4184/07. The preliminary versions of this article appeared in the Proceedings of ICCAD'08 [14] and ISPD'09 [16]. This paper was recommended by Associate Editor P. Saxena.

Q. Ma is with the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61821 USA (e-mail: qiangma1@illinois.edu).

Z. Qian and E. F.Y. Young are with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Shatin, Hong Kong (e-mail: zcqian@cse.cuhk.edu.hk; fyyoung@cse.cuhk.edu.hk).

H. Zhou is with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208 USA (e-mail: haizhou@eecs.northwestern.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2011.2131890

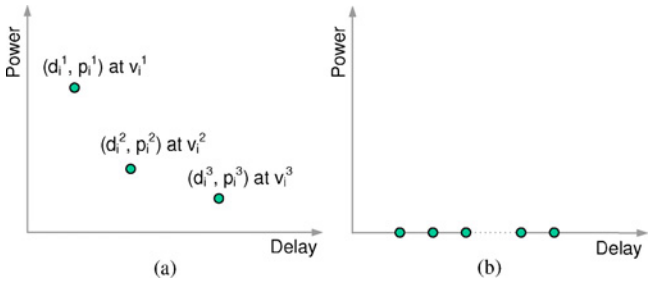


Fig. 1. DP Curve of (a) a vertex and (b) an edge in G.

We then propose a MSV-driven floorplanning framework that optimizes power consumption and physical layout of a circuit simultaneously during the floorplanning stage, by embedding the faster MCF algorithm into a simulated annealing (SA)-based floorplanner and applying the slower but optimal VOBB algorithm as a postprocessing step. Our approach is compared with the most updated previous works, and the results show that significant improvement on power saving can be achieved in much less running time.

The remainder of this paper is organized as follows. We define the multiple voltage assignment (MVA) problem and the MSV-driven floorplanning problem in Section II. The MCF algorithm and VOBB algorithm for the MVA problem are proposed in Sections III and IV, respectively. The SA-based MSV-driven floorplanning framework that integrates the two algorithms is introduced in Section V. Our approach is tested and compared with previous works, and the experimental results will be reported in Section VI. We conclude our paper in the last section.

II. Preliminaries and Problem Formulation

We are given a set of n modules $m_1, m_2, \ldots, m_n$ with areas and aspect ratio bounds. Each module m_i is given k_i choices of supply voltages v_i^q, for $1 \le q \le k_i$, and the power-delay tradeoff in m_i is represented by a delay-power curve (DP Curve), $\{(d_i^1, p_i^1), (d_i^2, p_i^2), \ldots, (d_i^{k_i}, p_i^{k_i})\}$, where each pair (d_i^q, p_i^q) is the corresponding delay and power consumption when m_i is operated at v_i^q. Note that each d_i^q is in Z^+ (positive integer set), since the delay of a module is measured in terms of the number of clock cycles. Fig. 1(a) is an example of a DP Curve, which contains three choices of supply voltages for this module. Moreover, we assume that for each module, power is a convex function of delay when each point (d_i^q, p_i^q) is connected to its neighboring point(s) by a linear segment in the DP Curve. We are also given a timing requirement T_{cycle}, called clock cycle time, of this circuit and require that the critical path delay is at most T_{cycle}.

We denote a netlist by a directed acyclic graph (DAG), $G = (V, E)$. Each vertex $i \in V$ denotes a module m_i, while each directed edge $e(i, j) \in E$ denotes an interconnect through which a signal is sent from m_i to m_j. Fig. 2 is an example of the DAG representation of a netlist, where each vertex is associated with a DP Curve, while each edge is attributed with a wire delay. Thus, both the vertices and the edges have delay.

To facilitate our problem formulation, we desire to associate delay with only edges, thus we transform $G = (V, E)$ into

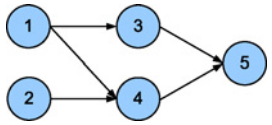


Fig. 2. DAG representation of a netlist.

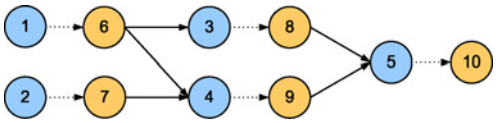


Fig. 3. Transformed DAG $\bar{G}$ of the graph G in Fig. 2.

$\bar{G} = (\bar{V}, \bar{E})$ in such a way that each vertex $i \in V$ is split into an input vertex and an output vertex, and the input vertex is connected to the output vertex by a directed edge. We denote this set of newly created edges and the set of original edges by E_1 and E_2, respectively. Hence, $\bar{E} = E_1 \cup E_2$. Fig. 3 illustrates the resultant DAG of the graph in Fig. 2 after this transformation, where dashed edges are in E_1 and solid edges are in E_2. After this transformation, the DP Curve of a vertex in G will be associated with its corresponding edge in E_1.[1] For example, after this transformation, vertex 1 and vertex 6 in $\bar{G}$ of Fig. 3 are the input and output vertex of vertex 1 in G of Fig. 2, respectively, thus the DP Curve of vertex 1 in G is now associated with the edge $e(1, 6)$ in $\bar{G}$, and we will use $k_{1,6}$ to denote the number of voltage level choices of this DP Curve. We can assume that each edge $e(i, j) \in E_2$ (corresponding to a wire) is also associated with a DP Curve in which the delay is an integer ranging from the wire's minimum delay to the maximum allowed delay T_{cycle}, and the power consumption is zero at all points. Fig. 1(b) is an example DP Curve corresponding to an edge $e(i, j) \in E_2$. Therefore, each edge $e(i, j) \in \bar{E}$ of $\bar{G}$ is now associated with one and only one DP Curve, while the vertices in $\bar{G}$ simply represent the starting or ending point of a time interval and are not associated with any DP Curves. For each edge $e(i, j) \in \bar{E}$, we use d_{ij} to denote the delay on it, and use $\bar{P}_{ij}$ to represent the function that maps delay to power consumption on edge $e(i, j)$, i.e., $\bar{P}_{ij}(d_{ij}^q) = p_{ij}^q$ for $1 \le q \le k_{ij}$ if $e(i, j) \in E_1$, and $\bar{P}_{ij}(d_{ij}) = 0$ if $e(i, j) \in E_2$. Moreover, we use l_{ij} and u_{ij} to denote the lower and upper bound of the delay d_{ij} on edge $e(i, j)$, i.e., $l_{ij} \le d_{ij} \le u_{ij}, \forall e(i, j) \in \bar{E}$. For each edge $e(i, j) \in E_1$, $l_{ij} = d_{ij}^1$ and $u_{ij} = d_{ij}^{k_{ij}}$. For each edge $e(i, j) \in E_2$, u_{ij} is T_{cycle}, while l_{ij} is the minimum wire delay of the corresponding connection which is estimated by scaling wire length to delay according to (1) when physical information is available as follows:

$$l_{ij} = \delta \cdot w_{ij} \tag{1}$$

where w_{ij} is the estimated length of the wire connecting module i and module j, and δ is a constant scaling factor. In

[1]We assume that a module has a fixed delay value given a working voltage level. In practice, there are multi-input and multi-output modules, where each input–output combination has different delay values. We can extend our approach to tackle this issue, by splitting the vertex representing a multi-input multi-output module into more vertices with more sophisticated connections, instead of splitting the vertex into two connected vertices.

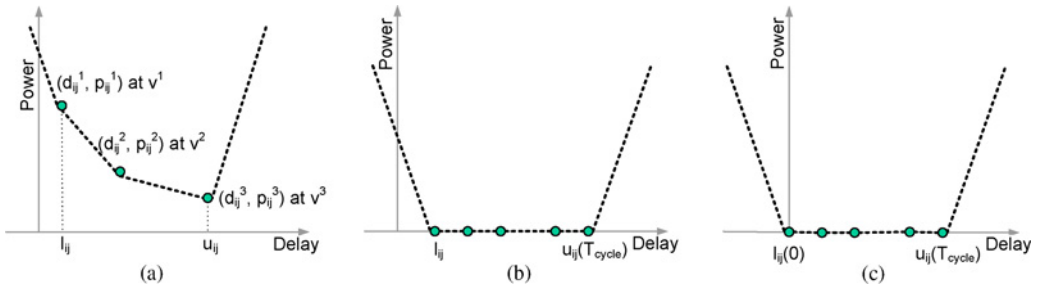


Fig. 4. DP Curve $P_{ij}(d_{ij})$ of (a) an edge $e(i, j) \in E_1$, (b) an edge $e(i, j) \in E_2$, and (c) an edge $e(i, j) \in E_3$.

our implementation, w_{ij} is computed as the Manhattan distance between module i and module j, which is a commonly used estimation of interconnection wire length at floorplanning stage. Let $t_i \in Z^+$ denote the arrival time at vertex i in $\bar{V}$, then it follows that

$$t_j - t_i \geq d_{ij}, \quad \forall e(i, j) \in \bar{E} \tag{2}$$

$$0 \leq t_i \leq T_{cycle}, \quad \forall i \in \bar{V}. \tag{3}$$

The static timing constraint and the voltage assignment problem can be defined as follows.

Definition 1: Static timing constraint: Given a clock cycle time T_{cycle} and a DAG, $\bar{G} = (\bar{V}, \bar{E})$ corresponding to a netlist, the static timing constraint of the netlist is that the arrival time t_i satisfies (2) and (3), $\forall i \in \bar{V}$.

Definition 2: MVA: Given a clock cycle time T_{cycle} and a DAG, $\bar{G} = (\bar{V}, \bar{E})$ where each edge e is associated with a DP Curve, select a delay-power pair for each edge in its DP Curve such that the sum of power consumption is minimized while the static timing constraint is satisfied.

Our ultimate goal in this paper is to solve the following MSV-driven floorplanning problem.

Definition 3: MSV-driven floorplanning: Given a netlist, a timing constraint, and a set of modules, each of which has multiple choices of supply voltage, generate a floorplan in which each module is assigned to work at a specific voltage level such that the timing constraint is satisfied and a weighted sum of power consumption, power network resource, area, and interconnect length is minimized.

III. MCF Algorithm for MVA

Chang and Pedram [3] prove that the MVA problem is an NP-hard problem, even if each module had only two voltage choices. In the following, we show that the MVA problem can be formulated as a convex cost dual network flow problem, and can be transformed into a MCF problem by using the Lagrangian relaxation technique. The MCF can be computed by calling the cost-scaling algorithm, after which the voltage assignment solution is obtained from the flow value on the network. When the delay choices of each module are continuous in the real or integer domain, the resultant voltage assignment solution is optimal. In general cases when the delay choices are not continuous, we are able to obtain a feasible voltage assignment with power consumption approximating the optimal solution.

A. Mathematical Program Formulation for MVA

According to the definitions and notations introduced in Section II, the MVA problem can be easily formulated into the following mathematical program.

Problem (1)

$$\begin{aligned}
&\text{Minimize} \quad \textstyle\sum_{e(i,j)\in\bar{E}} \bar{P}_{ij}(d_{ij}) && && \text{(1a)}\\
&\text{Subject to}\\
&\quad t_j - t_i \geq d_{ij}, && \forall e(i, j) \in \bar{E} && \text{(1b)}\\
&\quad 0 \leq t_i \leq T_{cycle}, && \forall i \in \bar{V} && \text{(1c)}\\
&\quad l_{ij} \leq d_{ij} \leq u_{ij}, && \forall e(i, j) \in \bar{E} && \text{(1d)}\\
&\quad d_{ij} \in \{d_{ij}^1, d_{ij}^2, \ldots, d_{ij}^{k_{ij}}\}, && \forall e(i, j) \in E_1 && \text{(1e)}\\
&\quad d_{ij} \in Z^+, && \forall e(i, j) \in E_2 && \text{(1f)}\\
&\quad t_i \in Z^+ && \forall i \in \bar{V}. && \text{(1g)}
\end{aligned}$$

Problem (1) without the discrete choice constraint (1e) is a convex cost integer dual network flow problem (or simply a *dual network flow problem*) [1], since its dual can be transformed to a MCF problem.

B. Simplification of the Mathematical Program

To simplify the problem, we will first make some modifications to the DP Curves. First of all, we connect each pair of neighboring points in a DP Curve by a linear segment, resulting in a piecewise linear convex function of power versus delay. It can be shown that there always exists an optimal solution to the dual network flow problem [problem (1) without (1e)] with integral variables [1]. Therefore, (1f) and (1g) can be removed.

In the second step of simplification, the lower and upper bounds on variables d_{ij} and t_i in (1c) and (1d) are eliminated. We first eliminate bounds on d_{ij} by defining a new power delay function for each edge $e(i, j) \in \bar{E}$ as follows:

$$P_{ij}(d_{ij}) = \begin{cases} \bar{P}_{ij}(u_{ij}) + M \times (d_{ij} - u_{ij}), & \text{if} \quad d_{ij} > u_{ij} \\ \bar{P}_{ij}(d_{ij}), & \text{if} \quad l_{ij} \leq d_{ij} \leq u_{ij} \\ \bar{P}_{ij}(l_{ij}) - M \times (d_{ij} - l_{ij}), & \text{if} \quad d_{ij} < l_{ij} \end{cases}$$

where M is a sufficiently large number, and here, we set $M = D + 1$, where $D = \sum_{e(i,j)\in E_1} \bar{P}_{ij}(l_{ij}) - \sum_{e(i,j)\in E_1} \bar{P}_{ij}(u_{ij})$. Actually, M can be viewed as a penalty for violating the lower and upper bounds of the variables, and we set the penalty to be large enough to guarantee that a solution with variables violating the bounds is impossible to be an optimal one, if a feasible solution of problem (1) exists. Therefore, by replacing $\bar{P}_{ij}(d_{ij})$ with $P_{ij}(d_{ij})$, we can remove (1c) and

(1d) with variables in the optimal feasible solution guaranteed to stay in the range specified by (1c) and (1d). An example transformed DP Curve of an edge in E_1 and that of an edge in E_2 are illustrated in Fig. 4(a) and (b), respectively.

Similarly, we define $B_i(t_i)$ as follows and put it into the objective function so that the lower and upper bounds on t_i can also be eliminated as follows:

$$B_i(t_i) = \begin{cases} M \times (t_i - T_{cycle}), & \text{if} \quad t_i > T_{cycle} \\ 0, & \text{if} \quad 0 \le t_i \le T_{cycle} \\ -M \times (t_i), & \text{if} \quad t_i < 0. \end{cases}$$

After all the above simplifications, problem (1) without (1e) can be transformed into problem (2) as follows.

Problem (2)

$$\text{Minimize} \quad \sum_{e(i,j)\in\bar{E}} P_{ij}(d_{ij}) + \sum_{i\in\bar{V}} B_i(t_i) \quad (2a)$$

$$\text{Subject to} \quad t_j - t_i \ge d_{ij}, \quad \forall e(i,j) \in \bar{E}. \quad (2b)$$

In the following section, problem (2) is further simplified by using the Lagrangian relaxation technique.

C. Lagrangian Relaxation

The Lagrangian relaxation technique is useful to eliminate difficult constraints. Here, we want to eliminate the constraints $t_j - t_i \ge d_{ij}$ in (2b). We apply the Lagrangian relaxation technique to problem (2), by introducing a nonnegative Lagrangian multiplier vector $\vec{x}$, the corresponding Lagrangian subproblem is to compute

$$L(\vec{x}) = \min_{d,t} \sum_{e(i,j)\in\bar{E}} P_{ij}(d_{ij}) + \sum_{i\in\bar{V}} B_i(t_i) - \sum_{e(i,j)\in\bar{E}} (t_j - t_i - d_{ij})x_{ij}$$

where x_{ij} is the nonnegative Lagrangian multiplier associated with the constraint $t_j - t_i \ge d_{ij}$. Note that

$$\sum_{e(i,j)\in\bar{E}} (t_i - t_j)x_{ij} = \sum_{i\in\bar{V}} \Big(\sum_{j:e(i,j)\in\bar{E}} x_{ij} - \sum_{j:e(j,i)\in\bar{E}} x_{ji} \Big) t_i.$$

We define

$$x_{0i} = \sum_{j:e(i,j)\in\bar{E}} x_{ij} - \sum_{j:e(j,i)\in\bar{E}} x_{ji}, \quad \forall i \in \bar{V}.$$

Then we can rewrite $L(x)$ as follows:

$$L(\vec{x}) = \min_{d,t} \sum_{e(i,j)\in\bar{E}} (P_{ij}(d_{ij}) + x_{ij}d_{ij}) + \sum_{i\in\bar{V}} (B_i(t_i) + x_{0i}t_i).$$

In order to convert this problem to a network flow problem, we add an extra vertex 0 into $\bar{G}$, and add an edge $e(0, i)$ for each vertex $i \in \bar{V}$. Let us denote this set of newly added edges by E_3 and denote the resultant DAG by $G^* = (V^*, E^*)$, so that $V^* = \{0\} \cup \bar{V}$ and $E^* = E_1 \cup E_2 \cup E_3$. Particularly, for each newly added edge $e(0, i)$, we put $d_{0i} = t_i$, $l_{0i} = 0$, $u_{0i} = T_{cycle}$, and $P_{0i} = B_i$. The delay power curve $P_{ij}(d_{ij})$ of an edge $e(i, j) \in E_3$ is shown in Fig. 4(c). Upon these notations, the Lagrangian subproblem can be restated as problem (3).

Problem (3)

$$L(\vec{x}) = \min_{d} \sum_{e(i,j)\in E^*} \{P_{ij}(d_{ij}) + x_{ij}d_{ij}\} \quad (3a)$$

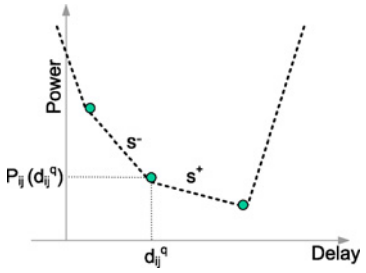


Fig. 5. Left slope s^- and right slope s^+ at $(d_{ij}^q, P_{ij}(d_{ij}^q))$ in a DP Curve.

subject to

$$\sum_{j:e(i,j)\in E^*} x_{ij} - \sum_{j:e(j,i)\in E^*} x_{ji} = 0 \qquad \forall i \in V^* \quad (3b)$$

$$x_{ij} \ge 0, \quad \forall e(i,j) \in E_1 \cup E_2. \quad (3c)$$

Since each $P_{ij}(d_{ij})$ is a piecewise linear convex function, $P_{ij}(d_{ij}) + x_{ij}d_{ij}$ is also a convex function of d_{ij} for a given value of x_{ij}. It is important to note that, for a given value of the vector $\vec{x}$, each term $\min_{d_{ij}}\{P_{ij}(d_{ij})+x_{ij}d_{ij}\}$ can be optimized separately, since d_{ij} is now independent of each other among all the edges $e(i, j) \in E^*$. We define function $H_{ij}(x_{ij})$ for each $e(i, j) \in E^*$ as follows:

$$H_{ij}(x_{ij}) = \min_{d_{ij}}\{P_{ij}(d_{ij}) + x_{ij}d_{ij}.\}$$

Then problem (3) can be rewritten as to compute

$$L(\vec{x}) = \sum_{e(i,j)\in E^*} H_{ij}(x_{ij}) \quad \text{subject to (3b) and (3c).}$$

The Lagrange dual problem is to determine $\vec{x}^*$ as follows.

Problem (4)

$$L(\vec{x}^*) = \max_{\vec{x}} L(\vec{x}) = \max_{\vec{x}} \sum_{e(i,j)\in E^*} H_{ij}(x_{ij}) \quad (4a)$$

subject to

$$\sum_{j:e(i,j)\in E^*} x_{ij} - \sum_{j:e(j,i)\in E^*} x_{ji} = 0 \qquad \forall i \in V^* \quad (4b)$$

$$x_{ij} \ge 0, \quad \forall e(i,j) \in E_1 \cup E_2. \quad (4c)$$

The following well-known theorem [2] establishes a connection between problems (2) and (4) as follows:

Theorem 1: Let $\vec{x}^*$ be a solution to the Lagrange dual problem, then $L(\vec{x}^*)$ equals the optimal objective value of the original convex dual network flow problem.

D. Transformation into MCF Problem

We first derive an explicit expression for $H_{ij}(x_{ij})$. As shown in Fig. 5, $P_{ij}(d_{ij})$ is a piecewise linear convex function. Let $(d_{ij}^q, P_{ij}(d_{ij}^q))$ be a break point (where the slope changes) on this function, and let s^- and s^+ denote the left slope and the right slope at this point, respectively. We can have the following two lemmas.

Lemma 1: $H_{ij}(x_{ij}) = P_{ij}(d_{ij}^q) + x_{ij}d_{ij}^q$, if $-s^+ \le x_{ij} \le -s^-$.

Proof: When $d_{ij} \le d_{ij}^q$, let $f(d_{ij}) = s^- d_{ij} + c$ (c is a constant) be the equation of the straight line on the left-hand side of d_{ij}^q (with slope s^-). Due to the convexity of P_{ij}, $P_{ij}(d_{ij}) \ge f(d_{ij})$, for $d_{ij} \le d_{ij}^q$. Thus $P_{ij}(d_{ij}) + x_{ij}d_{ij} \ge f(d_{ij}) + x_{ij}d_{ij} = (s^- + x_{ij})d_{ij} + c$. Note that $s^- + x_{ij} \le 0$, so $(s^- + x_{ij})d_{ij} + c$ is monotonically decreasing. Therefore, for $d_{ij} \le d_{ij}^q$, $P_{ij}(d_{ij}) + x_{ij}d_{ij} \ge (s^- + x_{ij})d_{ij} + c \ge (s^- + x_{ij})d_{ij}^q + c = P_{ij}(d_{ij}^q) + x_{ij}d_{ij}^q$. Similarly, when $d_{ij} \ge d_{ij}^q$, let $f(d_{ij}) = s^+ d_{ij} + c'$ (c' is a constant) be the equation of the straight line on the right-hand side of d_{ij}^q (with slope s^+). Then $P_{ij}(d_{ij}) + x_{ij}d_{ij} \ge f(d_{ij}) + x_{ij}d_{ij} = (s^+ + x_{ij})d_{ij} + c'$. Since $s^+ + x_{ij} \ge 0$, $(s^+ + x_{ij})d_{ij} + c'$ is monotonically increasing. Therefore, $P_{ij}(d_{ij}) + x_{ij}d_{ij} \ge (s^+ + x_{ij})d_{ij} + c' \ge (s^+ + x_{ij})d_{ij}^q + c' = P_{ij}(d_{ij}^q) + x_{ij}d_{ij}^q$, for $d_{ij} \ge d_{ij}^q$. Combining the two cases completes the proof. ■

Lemma 2: $H_{ij}(x_{ij}) = -\infty$ if $x_{ij} < -M$ or $x_{ij} > M$.

Proof: Consider the DP Curve $P_{ij}(d_{ij})$ of an edge $e(i, j)$ in E^*. If $x_{ij} < -M$, $P_{ij}(d_{ij}) + x_{ij}d_{ij}$ is minimum (with a value of $-\infty$) when d_{ij} tends to ∞ since $x_{ij}d_{ij}$ dominates $P_{ij}(d_{ij})$ when d_{ij} tends to ∞ (the slope of the rightmost segment of P_{ij} is M). Similarly, if $x_{ij} > M$, $P_{ij}(d_{ij}) + x_{ij}d_{ij}$ is minimum (with a value of $-\infty$) when d_{ij} approaches $-\infty$ since $x_{ij}d_{ij}$ dominates $P_{ij}(d_{ij})$ as d_{ij} tends to $-\infty$ (the slope of the leftmost segment of P_{ij} is $-M$). Therefore, $H_{ij}(x_{ij}) = \min_{d_{ij}}\{P_{ij}(d_{ij}) + x_{ij}d_{ij}\} = -\infty$, if $x_{ij} < -M$ or $x_{ij} > M$. ■

Recall that the objective of the Lagrange dual problem is to maximize $\sum_{e(i,j)\in E^*} H_{ij}(x_{ij})$ over $\vec{x}$, so we can safely add the constraint $-M \le x_{ij} \le M$ $\forall e(i, j) \in E^*$, according to Lemma 2.

Note that, for the function $H_{ij}(x_{ij})$ corresponding to an edge $e(i, j) \in E_1$, the Lagrangian multiplier x_{ij} must be nonnegative, therefore, it can be further bounded as $0 \le x_{ij} \le M$ $\forall e(i, j) \in E_1$. Together with Lemma 1 and the DP Curve in Fig. 4(a), we are able to express $H_{ij}(x_{ij})$ of each edge $e(i, j) \in E_1$ as follows:

$$H_{ij}(x_{ij}) = \begin{cases} P_{ij}(d_{ij}^{k_{ij}}) + d_{ij}^{k_{ij}} x_{ij}, & 0 \le x_{ij} \le b_{ij}(k_{ij}) \\ P_{ij}(d_{ij}^{k_{ij}-1}) + d_{ij}^{k_{ij}-1} x_{ij}, & b_{ij}(k_{ij}) \le x_{ij} \le b_{ij}(k_{ij}-1) \\ \vdots & \\ P_{ij}(d_{ij}^{q}) + d_{ij}^{q} x_{ij}, & b_{ij}(q+1) \le x_{ij} \le b_{ij}(q) \\ \vdots & \\ P_{ij}(d_{ij}^{1}) + d_{ij}^{1} x_{ij}, & b_{ij}(2) \le x_{ij} \le M \end{cases}$$

where $b_{ij}(q) = \frac{P_{ij}(d_{ij}^{q-1}) - P_{ij}(d_{ij}^{q})}{d_{ij}^{q} - d_{ij}^{q-1}}$. Note that it directly follows from the convexity of $P_{ij}(d_{ij})$ that $b_{ij}(q+1) \le b_{ij}(q)$.

Similarly, the constraint $0 \le x_{ij} \le M$ can also be added for each function $H_{ij}(x_{ij})$ that corresponds to an edge $e(i, j) \in E_2$, since the variable x_{ij} of an edge $e(i, j) \in E_2$ is also a nonnegative Lagrangian multiplier. According to Lemma 1 and the power delay curve $P_{ij}(d_{ij})$ in Fig. 4(b), the corresponding $H_{ij}(x_{ij})$ of an edge $e(i, j) \in E_2$ can be written as follows:

$$H_{ij}(x_{ij}) = l_{ij}x_{ij}, \quad 0 \le x_{ij} \le M.$$

Note that this is because the slope of the DP Curve of an edge $e(i, j) \in E_2$ from l_{ij} to u_{ij} is 0.

For the function $H_{ij}(x_{ij})$ corresponding to an edge $e(i, j) \in E_3$, the variable x_{ij} is not a Lagrangian multiplier, and it is bounded by $-M \le x_{ij} \le M$, according to Lemma 2. According to Lemma 1 and the DP Curve in Fig. 4(c), the corresponding function $H_{ij}(x_{ij})$ can be written as follows:

$$H_{ij}(x_{ij}) = \begin{cases} T_{cycle}x_{ij}, & -M \le x_{ij} \le 0 \\ 0, & 0 \le x_{ij} \le M. \end{cases}$$

Similarly, this is because the slope of the DP Curve of an edge $e(i, j) \in E_3$ from l_{ij} to u_{ij} is 0. Note that unlike the above cases, the variable x_{ij} for an edge $e(i, j) \in E_3$ is not a Lagrangian multiplier, thus it can be negative.

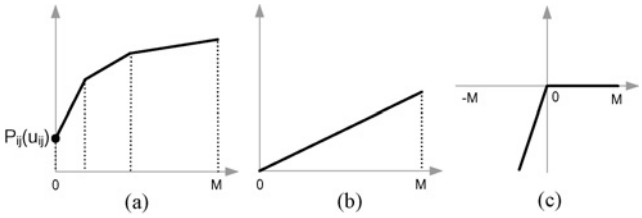


Fig. 6. Function $H_{ij}(x_{ij})$ of (a) an edge in E_1, (b) an edge in E_2, and (c) an edge in E_3.

Fig. 6 shows the bounds and shapes of all these functions $H_{ij}(x_{ij})$. It is easy to observe that these functions $H_{ij}(x_{ij})$ are piecewise linear concave. Then, we let $C_{ij}(x_{ij}) = -H_{ij}(x_{ij})$, so that $C_{ij}(x_{ij})$ is a piecewise linear convex function. We can subsequently transform problem (4) into problem (5) as shown below, by replacing $H_{ij}(x_{ij})$ with $-C_{ij}(x_{ij})$.

Problem (5)

$$\text{Minimize} \quad \sum_{e(i,j)\in E^*} C_{ij}(x_{ij}) \quad (5a)$$

Subject to

$$\sum_{j:e(i,j)\in E^*} x_{ij} - \sum_{j:e(j,i)\in E^*} x_{ji} = 0, \qquad \forall i \in V^* \quad (5b)$$

$$-M \le x_{ij} \le M \qquad \forall e(i, j) \in E_3 \quad (5c)$$

$$0 \le x_{ij} \le M \quad \forall e(i, j) \in E_1 \cup E_2. \quad (5d)$$

E. Cost-Scaling Algorithm

Problem (5) is a convex cost flow problem in which the cost associated with the flow on an edge $e(i, j)$ is a piecewise linear convex function containing a number of linear segments. We can transform problem (5) into a MCF problem in an expanded network $G' = (V', E')$, and solve it using a cost-scaling algorithm [5].

In the transformation from G^* to G', each edge $e(i, j)$ in G^* will be replaced by one or more edges, depending on the number of linear segments in the corresponding function $C_{ij}(x_{ij})$. There are three categories of edges to consider, namely, E_1, E_2, and E_3.

1) *Edge in* E_1: An edge in E_1 represents an input and output relationship of a module. For each edge $e(i, j) \in E_1$, $k = k_{i,j}$ edges are inserted into G', with one edge corresponding to one linear segment in the cost function C_{ij}. These edges have costs of $-d_{ij}^k$, $-d_{ij}^{k-1}$, $-d_{ij}^{k-2}$, …, $-d_{ij}^1$, upper capacities of $b_{ij}(k)$, $b_{ij}(k-1) - b_{ij}(k)$, $b_{ij}(k-2) - b_{ij}(k-1)$, …, $M - b_{ij}(2)$, respectively, and lower capacities of zero for all of them.
2) *Edge in* E_2: An edge in E_2 represents an interconnect. Each edge $e(i, j) \in E_2$ is directly copied to G', with

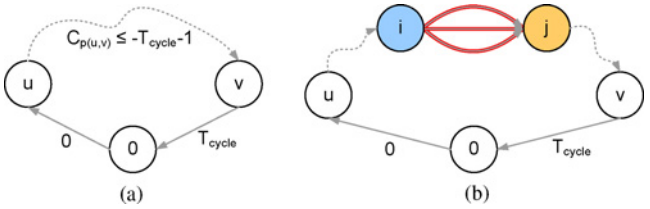


Fig. 7. Paths in the proof of Lemma 3. (a) The "only if" direction of the proof. (b) The "if" direction of the proof.

its cost, lower and upper capacity set as $-l_{ij}$, 0 and M, respectively.

3) *Edge in* E_3: Edges belonging to this category emanate from vertex 0 in G^*. For each of them, two edges are introduced in G'. The cost, lower capacity, and upper capacity are $-T_{cycle}$, $-M$ and 0 for one edge, and 0, 0, and M for the other edge, respectively.

It is well known that solving the convex cost flow problem in G^* is equivalent to solving the MCF problem in G'.[2] We can directly apply the cost-scaling algorithm ([5] and [6]) on G' to find a MCF. The time complexity of using cost-scaling algorithm to solve our MVA problem is $O(nk(m+nk)log(k^2n^2/m)log(knT_{cycle}))$.

F. Solution Transformation

The convex cost-scaling algorithm upon termination gives an optimal flow x^* and the corresponding optimal node potentials π^*. Both the solutions x^* and π^* may be non-integer. Since all the edge costs are integers, there always exist integral optimal vertex potentials π. To determine them, we construct $G'(x^*)$ and determine the shortest path distance $sd(i)$ from vertex 0 to every other vertex $i \in V'$. Since all edge costs in $G'(x^*)$ are integers, each $sd(i)$ is also an integer. Then $\pi(i) = -sd(i)$ for each $i \in V'$ gives an integral optimal set of vertex potentials for problem (5). According to [1], an optimal solution to problem (2) can be obtained by first assigning $t_i = \pi_i$, then set $d_{ij} = t_j - t_i$ for each $e(i, j) \in E_1$. Note that we do not need to assign the value of d_{ij} for each $e(i, j) \in E_2$, since the power consumption on a wire is zero, and the timing constraints will be automatically satisfied if there exists a feasible solution to the original voltage assignment problem [problem (1)]. If no feasible solutions of the voltage assignment problem exist, the total flow passing through vertex 0 will be larger than D, as stated in the following lemma.

Lemma 3: Given an instance of the MVA problem, there exists a feasible solution to this instance if and only if the flow value through vertex 0 in the min-cost network flow problem in G' as constructed above is such that $f_0 \le D$, where $D = M - 1 = \sum_{e(i,j)\in E_1} \bar{P}_{ij}(l_{ij}) - \sum_{e(i,j)\in E_1} \bar{P}_{ij}(u_{ij})$.

Proof: *(Only if part:)* Suppose a feasible solution exists for the voltage assignment problem. Let C be the total cost of the flow network. Initially, $C = 0$ when the flow on each edge is zero, and this is automatically the MCF if there is no negative cycle. Note that every negative cycle must pass vertex 0, since the original netlist is a DAG. Now suppose, without loss of generality, the edge $e(0, u)$, the path from vertex u to vertex v (denoted by $p(u, v)$) and the edge $e(v, 0)$ form a negative cycle, as shown if Fig. 7(a). Recall that the edge cost are all integers, thus the cost through $p(u, v)$ [denoted by $C_{p(u,v)}$] must be integral, i.e., $C_{p(u,v)} \le -T_{cycle} - 1$, otherwise the above-mentioned negative cycle cannot be formed. This yields that one unit flow through vertex 0 will decrease the total cost C by at least one, and hence $C \le -f_0$, where f_0 denotes the flow value through vertex 0. Note that the optimal objective value of the Lagrange dual problem [problem (4)] $L(x^*) = (-C) + \sum_{e(i,j)\in E_1} \bar{P}_{ij}(u_{ij})$. If $f_0 > D$, then $C < -D$, thus $L(x^*) > D + \sum_{e(i,j)\in E_1} \bar{P}_{ij}(u_{ij}) = \sum_{e(i,j)\in E_1} \bar{P}_{ij}(l_{ij})$. According to Theorem 1, $L(x^*)$ equals the optimal objective value of problem (2) (the dual network flow problem), so the objective value of problem (2) is also greater than $\sum_{e(i,j)\in E_1} \bar{P}_{ij}(l_{ij})$, which corresponds to the highest possible total power consumption of all the modules. This is impossible and we can thus conclude that no feasible solutions exist for the original voltage assignment problem exists, contradictory to our original assumption.

(If part:) Suppose the flow through vertex 0 is less than or equal to D, i.e., $f_0 \le D$. Again, we prove by contradiction. If there is no feasible solution to the original voltage assignment problem, it means that the timing constraint T_{cycle} will be violated by some critical path $p(u, v)$ even if each module on $p(u, v)$ is assigned to work at its highest legal voltage level. Given this, there must be at least one module with its input vertex as i and output vertex as j in the expanded network G', such that all the edges between vertex i and vertex j are saturated, since a negative cycle will be formed with vertex 0 otherwise [as shown in Fig. 7(b), all the red edges are saturated]. Note that the sum of the capacities of these thickened red edges is $M = D + 1$ and the flow through these edges must go through vertex 0. Therefore, we can conclude that $f_0 > D$, contradictory to our original assumption. ■

If the flow through vertex 0 is not more than D, we can construct a feasible solution for the original problem [problem (1)] with discretization constraint (1e) from this optimal solution of problem (2), by taking the largest possible delay choice d_{ij}^q smaller than or equal to d_{ij} for each edge $e(i, j) \in E_1$ (this is the legalization step of the voltage assignment). It can be easily shown that this solution is feasible to our original problem, and we can see from the experimental results that this transformation method can approximate the optimal power consumption well and gives large power savings.

G. Handling Level Shifters

Whenever there is a net through which a signal is sent from a module at a low voltage level to a module at a high voltage level, a level shifter will be inserted into this net. Each level shifter is associated with a fixed delay and power values, and will affect the timing and power consumption of the whole circuit. In order to take level shifters into consideration, we extend our approach in a conservative way. When the DAG representation of the input netlist is given, we will first compute the longest path L (in terms of the number of edges on a path) from a primary input to a primary output in the DAG. It

[2]There is a constant difference of $\sum_{e(i,j)\in E_1} P_{ij}(u_{ij})$ between the cost of the MCF in G^* and the cost of the MCF in G'.

is trivial that at most L level shifters will be inserted on any path of the DAG after voltage assignment. This indicates that after level shifter insertion, the delay of the whole circuit will be increased by $L \cdot d_{ls}$ units in the worst case where d_{ls} denotes the delay of a level shifter. Therefore, in order to guarantee that the timing constraint can still be satisfied after level shifter insertion, we simply put $T_{cycle} - L \cdot d_{ls}$ as the timing constraint of the circuit while solving the voltage assignment problem, after which the necessary level shifters are inserted.

H. Summary of the MCF Algorithm

Algorithm *MCF* (MVA Instance $\mathcal{S}$).

1) Construct DAG $G = (V, E)$ for the input netlist.
2) Compute longest path L of G.
3) $T'_{cycle} \leftarrow T_{cycle} - L \cdot d_{ls}$.
4) Compute the capacity bound M.
5) Convert $G = (V, E)$ into flow network $G' = (V', E')$.
6) Assign capacity and cost to each edge $e \in E'$ of G'.
7) Run cost-scaling algorithm on $G' = (V', E')$.
8) Transform the flow into voltage assignment.
9) Legalize the voltage assignment for each module.

IV. VOBB Algorithm for MVA

The MCF algorithm proposed in the last section is a polynomial time algorithm, however, it may not produce the optimal voltage assignment solution in general cases when the delay choices on each edge are not continuous. In this section, we present a branch-and-bound-based searching method called VOBB that is able to obtain the optimal voltage assignment for a candidate floorplan. This VOBB algorithm employs the MCF algorithm as a subroutine, and is used to further optimize the voltage assignment solution on a candidate floorplan generated by the MCF algorithm. However, there is no guarantee that this searching based method will terminate within polynomial time, which makes it inappropriate to be embedded into a SA procedure. It will thus be used as a postprocessing step in our MSV-driven floorplanning framework.

A. Integer Program for MVA

In order to facilitate the presentation of the optimal algorithm, we first introduce some extra notations. Consider the DAG $\bar{G} = (\bar{V}, \bar{E})$ transformed from the DAG $G = (V, E)$ representing the netlist of a circuit, where each vertex $v \in V$ is split into an input vertex and an output vertex, and the input vertex is connected to the output vertex by a directed edge (refer to Figs. 2 and 3 for illustration). Recall that $\bar{E} = E_1 \cup E_2$, where E_1 denotes the set of newly added edges and E_2 denotes the set of original edges in G. Each edge $e(i, j) \in E_1$ is associated with a DP Curve $\{(d_{ij}^1, p_{ij}^1), (d_{ij}^2, p_{ij}^2), \ldots, (d_{ij}^{k_{ij}}, p_{ij}^{k_{ij}})\}$, where each pair (d_{ij}^q, p_{ij}^q) (for $q = 1, 2, \ldots, k_{ij}$) is the delay and power consumption when the corresponding module of $e(i, j)$ operates at the qth working voltage level (in the following, we denote this voltage level by v_{ij}^q). We assign a set of binary variables $\{x_{ij}(1), x_{ij}(2), \ldots, x_{ij}(k_{ij})\}$ to each edge $e(i, j) \in E_1$, such that $x_{ij}^q = 1$ if the module corresponding to $e(i, j)$ operates at the qth working voltage level, and $x_{ij}^q = 0$ otherwise. Therefore, $\sum_{q=1}^{k_{ij}} x_{ij}(q) = 1$. Each edge $e(i, j) \in E_2$ corresponds to an interconnection wire and is attributed with an interconnection delay d_{ij}. The interconnection delay is calculated as $d_{ij} = \delta \cdot w_{ij}$, where δ is a constant scaling factor, and w_{ij} is the estimated wire length. Note that on an edge $e(i, j) \in E_2$, a level shifter needs to be inserted if the working voltage level at the source is lower than the working voltage level at the sink. Let the binary variable y_{ij} denote whether a level shifter is needed to be inserted on edge $e(i, j)$: $y_{ij} = 1$ if a level shifter is needed, and $y_{ij} = 0$ otherwise. We use φ and ρ to denote the power consumption and delay of a level shifter, respectively. We use t_i to denote that arrival time at vertex i and use T_{cycle} to denote the clock cycle time. In addition, for each edge $e(i, j) \in E_1$, we say that vertex i is the predecessor of vertex j (denoted by $pred(j)$), and we say that vertex j is the successor of vertex i (denoted by $succ(i)$). Note that each edge $e(i, j) \in E_1$ comes from the splitting of an original vertex in G, so vertex i is the unique predecessor of vertex j and vertex j is the unique successor of vertex i.

The objective of the MVA problem is to select a delay-power pair for each module such that the total power consumption is minimized while the static timing constraint is satisfied. Using the notations above, the MVA problem can be formulated into an integer program as stated in problem (6). The objective function (6a) is the sum of all modules' and level shifters' power consumption. Constraints (6b)–(6d) are to ensure that the timing requirement is satisfied, while (6e) and (6f) make sure that exactly one working voltage level is selected for each module. Constraint (6g) specifies the values of the variables representing the level shifters: $\forall e(i, j) \in E_2$, a level shifter needs to be inserted ($y_{ij} = 1$) if the module at the source works at a lower voltage level than the module at the sink, i.e., $\sum_{q=1}^{k_{\text{pred}(i),i}} v_{\text{pred}(i),i}^q \times x_{\text{pred}(i),i}(q) \leq \sum_{q=1}^{k_{j,\text{succ}(j)}} v_{j,\text{succ}(j)}^q \times x_{j,\text{succ(j)}}(q)$; otherwise, there is no need to insert a level shifter.

Problem (6)

$$\text{Minimize} \sum_{e(i,j)\in E_1} \sum_{q=1}^{k_{ij}} p_{ij}^q x_{ij}(q) + \sum_{e(i,j)\in E_2} y_{ij} \cdot \varphi \tag{6a}$$

Subject to

$$t_j - t_i \geq \sum_{q=1}^{k_{ij}} d_{ij}^q x_{ij}(q), \quad \forall e(i, j) \in E_1 \tag{6b}$$

$$t_j - t_i \geq d_{ij} + y_{ij} \cdot \rho, \quad \forall e(i, j) \in E_2 \tag{6c}$$

$$0 \leq t_i \leq T_{cycle}, \quad \forall i \in \bar{V} \tag{6d}$$

$$\sum_{q=1}^{k_{ij}} x_{ij}(q) = 1, \quad \forall e(i, j) \in E_1 \tag{6e}$$

$$x_{ij}(q) \in \{0, 1\}, \quad q = 1, 2, \ldots, k_{ij}, \forall e(i, j) \in E_1 \tag{6f}$$

$$y_{ij} = \begin{cases} 1, & \sum_{q=1}^{k_{\text{pred}(i),i}} v_{\text{pred}(i),i}^q \times x_{\text{pred}(i),i}(q) \leq \\ & \sum_{q=1}^{k_{j,\text{succ}(j)}} v_{j,\text{succ}(j)}^q \times x_{j,\text{succ(j)}}(q) \\ 0, & \text{otherwise} \end{cases} \quad \forall e(i, j) \in E_2. \tag{6g}$$

B. VOBB-Based Search

In this section, we will show how to solve the MVA problem optimally. We will first describe the basic VOBB algorithm and then explain how to compute the upper and lower bounds. The branch-and-bound approach solves problems by searching successive partitions of the solution space. As usual, we will describe the searching process by a search tree in which an internal node represent a set of solutions while a leaf node represent a single solution.

1) *Branching Rules:* In our searching approach, we visit different parts of the solution space by confining the values of some selected variables. We use the set of binary variables $\{x_i(1), x_i(2), \ldots, x_i(k_i)\}$ to denote the voltage assignment for module m_i. In each iteration, we may select a module m_i (we will describe the selection criteria in the following) and branch to k_i children nodes, which represent the solution space of k_i different subproblems where k_i is the number of supply voltage choices of module m_i. Without loss of generality, we assume that we choose m_1 in the first iteration and choose m_2 in the second iteration. We branch into the first subproblem by assigning m_1 to operate at its first working voltage level, so the subproblem is obtained by adding the set constraints to problem (6) as follows:

$$x_1(1) = 1, x_1(2) = 0, \ldots, x_1(k_1) = 0.$$

Similarly, we branch into the subproblem in the next level by assigning m_2 to operate at its first working voltage level, so this subproblem is obtained by adding another set of constraints as follows:

$$x_2(1) = 1, x_2(2) = 0, \ldots, x_2(k_2) = 0.$$

In this way, we will branch out to many subproblems and search the whole solution space of the original problem. Next, we will discuss selection criteria for the branching steps. Given a candidate floorplan, we first solve the MVA problem using the approach in Section III (without the legalization step) and obtain a solution $\{(d_1^*, p_1^*), (d_2^*, p_2^*), \ldots, (d_n^*, p_n^*)\}$ for module $m_1, m_2, \ldots, m_n$, we will check starting from $i = 1$ to n whether module m_i can work at (d_i^*, p_i^*), and select the first module whose solution is infeasible to branch out to the children nodes. After selecting one module m_i, we will construct k_i subproblems with additional constraints to confine the voltage level of m_i where k_i is the number of voltage choices for m_i.

2) *Upper Bounds and Lower Bounds Computation:* For a branch-and-bound-based algorithm, tight upper and lower bounds are important for efficiency. A good upper bound can be obtained by solving the MVA problem with the MCF algorithm. The sum of the power consumptions of modules and the level shifters gives the upper bound. This is a good upper bound as our experiments show that the number of cells with infeasible voltage assignments before the legalization step of the MCF algorithm is relatively small. The lower bound can be easily obtained by solving a linear relaxation of the problem (6) (the detail is omitted). Note that we will do the upper bound and lower bound computations whenever a new node in the search tree is visited.

3) *Pruning Rules and Value-Oriented Searching Rules:* With the upper and lower bounds obtained above, we can construct our pruning rules. Some nodes with their subtrees together will never contribute an optimal solution, and good pruning rules help to find such nodes to reduce runtime. If a node is infeasible, i.e., we cannot find a feasible solution satisfying the timing constraint even assuming a continuous domain for the module voltages (this can be verified by invoking the MCF algorithm), it will be pruned. Besides, if a node's lower bound is greater than or equal to the global upper bound, it will also be pruned. This is correct since we can never find a better solution by traversing into its subtree.

With the above bounds and pruning rules, we can search the solution space faster. Our VOBB searching algorithm is divided into rounds. In each round, we set a *target* which is used as a threshold to decide whether a node should be visited in this round. For example, in the first round, the *target* will be computed as follows:

$$target = \alpha(O_{ub} + O_{lb})$$

where $0 \leq \alpha \leq 1$ is a constant (set to 0.6 in our experiments), and O_{ub} and O_{lb} are the upper and lower bounds of the original problem. When we reach a node S, we will check whether $S_{lb} \leq target$ where S_{lb} is the lower bound at S. If so, we will continue to visit the subtree of S; otherwise, this node S will be marked and will not be visited in this round (may be visited in the following rounds). After one round, the value of *target* will be updated as $target + C$ where C is a constant (set to 5% of O_{ub} in our experiments). Note that a subtree which is explored in a previous round will not be repeatedly visited in any subsequent rounds. In general, this value-oriented searching strategy will search nodes with smaller lower bounds in earlier rounds and we can upper bound the deviation of a solution obtained from such value-oriented search from the optimal solution. This idea is derived from the target-oriented branch-and-bound algorithm in [17]. Actually, this searching scheme is general and is independent of how the upper and lower bounds are computed, i.e., other upper and lower bounds computation methods can be applied and integrated into this searching scheme.

In our VOBB algorithm, there are two conditions under which we can stop searching.

1) A feasible solution is found at a node of which the objective function value is equal to the smallest lower bound (obtained by the linear relaxation) ever seen during the search.
2) Current round is finished and the global upper bound O_{ub} is less than or equal to *target*.
3) All possible nodes are visited and searched.

V. MSV-Driven Floorplanning Framework

We integrate the MCF algorithm and the VOBB algorithm with a SA-based floorplanner to develop a MSV-driven floorplanning framework. This framework can be divided into two stages: floorplanning stage and postprocessing stage.

A. Floorplanning Stage

In the floorplanning stage, the MCF algorithm for voltage assignment is embedded into a SA-based FAST-SP [18] floorplanner to explore the solution space. After each random move, the length of some interconnects may be changed which will cause changes in the minimum delay of the corresponding wires. Thus, the operating voltage level of each module is reassigned by solving the updated dual network flow problem, after which the necessary level shifters are inserted and the total power consumption is calculated. The power network resource is evaluated afterwards. When the annealing process terminates, the best found solution will be returned.

We use the cost function to evaluate a candidate floorplan as follows:

$$\psi = A + \lambda_w W + \lambda_{pn} \Phi + \lambda_p P. \quad (4)$$

In this function, A is the area of the floorplan, W is the total wire length estimated by the half perimeter bounding box method, Φ represents the power network resource, which is estimated as the sum of half-perimeters of the bounding boxes of the blocks working at the same voltage levels (the blocks assigned with the same voltage level are preferred to stay close with this term), P denotes the total power consumption, and particularly, P is set to a large number to give a heavy penalty when the candidate floorplan violates timing. The parameters λ_w, λ_{pn}, and λ_p can be used to adjust the relative weighting between the contributing factors, and they are determined in a random walk with 1000 random moves before the annealing process starts, in such a way that the four factors, area, wire length, power network resource, and power cost, are contributing similarly to the cost function. This cost function can be modified to consider the fixed-outline constraint by replacing the area term A (since we are not minimizing the area) by $\lambda_\infty(\max\{0, w - W'\} + \max\{0, h - H'\})$ where λ_∞ is a large positive constant, w and h are the width and height of the candidate floorplan solution, and W' and H' are the fixed width and height required for the final floorplan design.

B. Postprocessing Stage

We use the VOBB voltage assignment algorithm as a postprocessing step. After an initial floorplan is generated by the SA-based floorplanner with MCF algorithm, we perform our VOBB algorithm to obtain the optimal voltage assignment. However, the resultant power network routing resources may be worsened, as the modules assigned with the same working voltage level may not stay together. Thus, we apply an ordinary SA-based floorplanner whose objective function is a weighted sum of area, wire length, and power network routing resources (the voltage assignment for the modules are fixed). After this fast floorplanning step, the VOBB algorithm is performed once again to assign a final working voltage to each module to ensure that the timing constraint is satisfied.

VI. Experimental Results

In this section, we perform several sets of experiments to validate our proposed approach. We first compare the performance of the MCF algorithm and the VOBB algorithm, we then compare VOBB with an existing ILP-based approach [11] for voltage assignment. Finally, we evaluate our MSV-driven floorplanning framework by comparing with the best previous paper [10] on this problem. Throughout the experiments, we use six test cases which are obtained from the authors of [10]. The number of blocks is ranging from 10 to 300, and some of the blocks are soft blocks. These test cases are based on the GSRC benchmarks, with power and delay specifications added for each block. Every block has two legal working voltage levels, namely, VDDH and VDDL, each of which is associated with a power consumption value and a delay value. In our test cases, VDDH is 1.5 V and VDDL is 1.0 V. The power consumption associated with VDDH (VDDL) is high (low), and the delay value associated with VDDH (VDDL) is low (high). The values of power consumption and delay values are approximately proportional to the area of the corresponding block. For example, for block b_1 (with area of 24 045-unit) in test case $n10$, the delay value and power consumption are 21 640-unit and 24 045-unit at VDDH, respectively, while the delay value and power consumption are 57 706-unit and 10 686-unit at VDDL, respectively. For block b_1 (with area of 432-unit) in test case $n300$, the delay value and power consumption are 388-unit and 432-unit at VDDH, respectively, while the delay value and power consumption are 1034-unit and 192-unit at VDDL, respectively. In general, the area of the blocks in the test cases is ranging from 150 to 40 000. In our

TABLE I
VOLTAGE ASSIGNMENT COMPARISONS OF MCF AND VOBB

Test Cases	Power		Ratio	Runtime (s)		No.
	MCF (x)	VOBB (y)	y/x (%)	MCF	VOBB	Difference
n10	202 709	185 270	91.4	1.09	1.2	1.7
n30	162 534	155 853	95.9	9.31	12.1	2.9
n50	166 931	157 163	94.1	25.34	35.0	7.8
n100	137 608	126 855	92.2	121.6	600	9.9

TABLE II
RESULTS OF OUR FIXED-OUTLINE VERSION SA-BASED FLOORPLANNER WITH MCF ALGORITHM

Data Set	Power Cost with LS	Power Saving (%)	No. LS	Runtime (s)	Success Rate (%)
n10	189 942	12.40	4	1.13	100
n30	151 483	26.34	25	8.45	100
n50	152 684	21.76	34	23.23	100
n100	120 450	33.09	77	123.2	100
n200	130 537	26.52	134	435.5	100
n300	159 431	41.71	87	1045.2	80
Average	150754.5	26.97	60	272.8	96.67

TABLE III
COMPARISONS BETWEEN VOBB AND PREVIOUS WORK [11]

Test Cases	Power			Runtime		
	MCF	VOBB	[11]	MCF	VOBB	[11]
n10	189 942	169 058	169 058	1.09 s	1.2 s	0.0 s
n30	153 526	143 460	143 460	9.31 s	12.1 s	10 h
n50	152 684	138 983	138 983	25.34 s	35.0 s	11.1 m
n100	120 450	113 231	117 761*	121.6 s	10.0 m	10 h
n200	135 250	119 229*	116 341*	454.6 s	10 h	10 h
n300	188 113	142 641	143 041*	1027 s	32.4 m	10 h
Average	156 661	137 767	138 107	–	–	–
Difference	1.0	0.879	0.882	–	–	–

*Indicates that the test case is not finished within the 10 h time limit.

TABLE IV
COMPARISON BETWEEN OUR MSV-DRIVEN FLOORPLANNING FRAMEWORK AND PREVIOUS WORK [10]

Data Set	Power Cost with LS		Power Saving (%)		Power Network Routing Resources		Level Shifter No.		Dead Space (%)		Runtime	
	Ours	[10]	Ours	[10]	Ours	[10]	Ours	[10]	Ours	[10]	Ours (MCF+VOBB)	[10]
n10	169 058	216 841	22.04	0	1373	965	8	0	2.12	4.87	4.49 s (1.09 s+3.4 s)	6.001 s
n30	143 460	190 717	30.24	7.26	1354	1369	21	57	7.05	9.03	41.84 s (9.31 s+32.53 s)	115.069 s
n50	138 983	172 884	28.78	11.40	1662	1514	32	119	10.82	21.10	120.18 s (25.34 s+94.84 s)	569.360 s
n100	113 231	179 876	37.10	0.10	1446	1671	50	92	9.59	34.07	1424.77 s (121.6 s+1303.17 s)	1768 s
n200*	121 222	174 818	31.76	1.58	1626	2040	94	399	14.30	46.52	20.7 h (454.6 s+20.5 h)	4212 s
n300	142 641	219 492	47.85	19.75	1690	2147	30	452	12.52	44.10	5710.11 s (1027 s+4683.11 s)	4800 s
Average	138 099	192 438	32.96	6.68	1525	1618	39	186	9.46	26.61	–	–

*Indicates that the corresponding test case is not finished within the 10 h time limit.

experiments, the clock cycle time T_{cycle} for each test case is set to be 1.5 times of the critical path delay when all the modules are working in VDDH, and the delay of a level shifter d_{ls} is set to be 200-unit. Our algorithms are implemented in the C programming language and all the experiments are performed on a Linux machine with a 3.20 GHz CPU and 2 GB RAM.

We found that the voltage assignment solution produced by the MCF algorithm was fairly close to the optimal one (by VOBB), which can be verified by the following set of experiments. We randomly generate ten floorplans for each benchmark and compare the two algorithms in terms of average power consumption and the number of modules with different voltage assignments. Results show that, for each test case, only about 10% of the modules are assigned with different voltage levels by the two algorithms (see Table I). Note that the runtime for MCF is the execution time of the SA process that integrates the MCF algorithm (a single iteration of the MCF algorithm terminates in less than 1 s).

As we mentioned before, our SA-based floorplanner with MCF algorithm can be easily adapted to consider fixed-outline constraints by modifying the cost function. In the following, we test the fixed-outline version of our SA-based floorplanner. The fixed-outline is set to be $W \times H$ in our experiment, where $W = H$ and $W \times H$ equals 1.2 times the total area of all the modules. The results are listed in Table II. We can see that our MCF algorithm works well within the fixed-outline context. An average of 26.97% power saving can be achieved. The fixed-outline floorplanner is performed ten times on each test case, and the success rate is also reported in Table II.

Besides, we further demonstrated the effectiveness of the MCF approach by performing another set of experiments in which the same set of circuits is used but the number of legal working voltage levels for each module is augmented to either three or four. The resultant floorplans are displayed in Fig. 8. An average of 28.62% power saving can be achieved.

We then compare our VOBB algorithm for the MVA problem with the latest previous paper [11] that also tried to solve the voltage assignment problem on a candidate floorplan. Reference [11] is an ILP-based approach. The results are shown in Table V-B. The runtime limit for each data set is 10 h, and the "∗" indicates that the program does not run to the end after 10 h (so, the solution is suboptimal). VOBB can obtain optimal solutions in a reasonable amount of time for most of the test cases and has out-performed [11] quite a lot. The results obtained by the SA-based floorplanner with

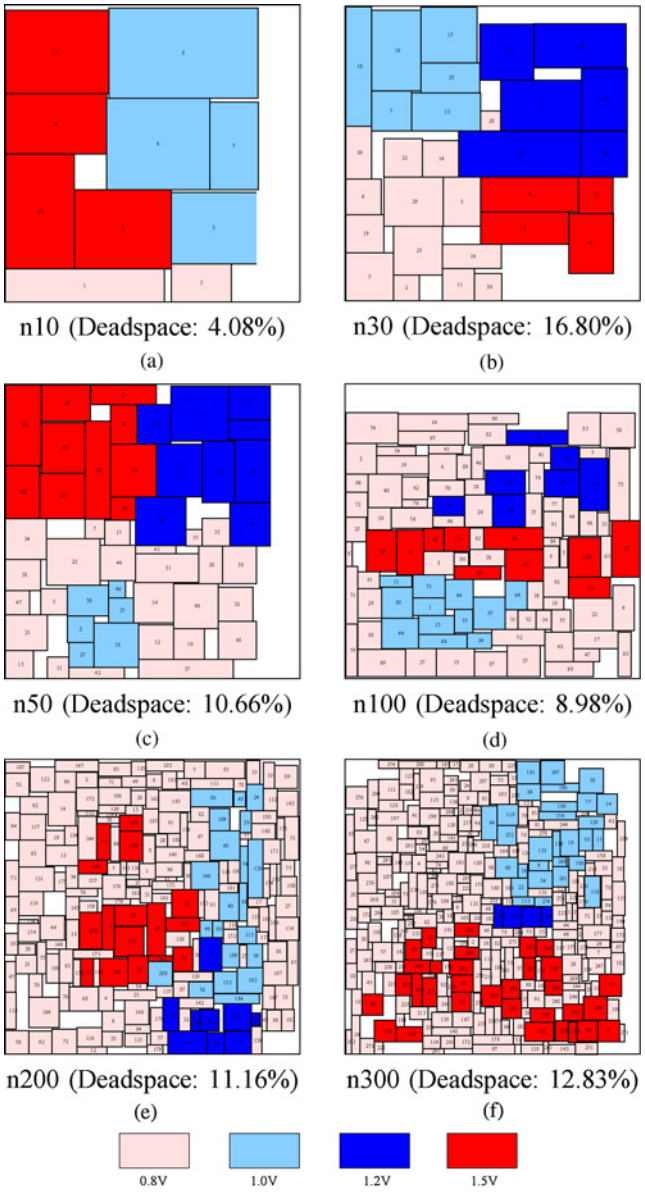


Fig. 8. Resultant floorplans with more working voltage levels. (a) n10 (Deadspace: 4.08%). (b) n30 (Deadspace: 16.80%). (c) n50 (Deadspace: 10.66%). (d) n100 (Deadspace: 8.98%). (e) n200 (Deadspace: 11.16%). (f) n300 (Deadspace: 12.83%).

MCF algorithm is also listed for comparison. Compared with the MCF approach, we can see that VOBB contributes 12.1% improvement on power saving on average, while [11] improves the power saving by 11.8%. During the experiments, we found that the runtime depends more on the complexity of the circuit rather than the number of modules. The word "complexity" refers to the total number of constraints. We can see that the runtime for n200 is even longer than that for n300.

At last, we report the results of a comparison between our MSV-driven floorplanning framework (as described in Section V) and the previous paper [10]. The results are displayed in Table V-B. Again, the runtime limit for each attempt of the VOBB procedure is 10 h, and "$*$" indicates that the VOBB procedure does not finish within this time limit. We can see that our approach achieves much more power saving (an average of 32.96% versus 6.68% by [10]) with much less level shifters (an average of 39 versus 186 by [10]).

VII. Conclusion

In this paper, we proposed two algorithms to solve the voltage assignment problem under timing constraints, namely, MCF and VOBB. The two algorithms are integrated into a MSV-driven floorplanning framework that simultaneously optimizes power consumption and physical layout of a circuit during the floorplanning stage. We compared our approach with the latest previous papers, and the experimental results show that, using our approach, significant improvement on power saving can be achieved in less running time, which confirms the effectiveness and efficiency of our proposed approach.

References

[1] R. K. Ahuja, D. S. Hochbaum, and J. B. Orlin, " Solving the convex cost integer dual network flow problem," *Manag. Sci.*, vol. 49, no. 7, pp. 950–964, 2003.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Application*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[3] J.-M. Chang and M. Pedram, "Energy minimization using multiple supply voltages," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 5, no. 4, pp. 436–443, Dec. 1997.

[4] R. L. S. Ching, E. F. Y. Young, K. C. K. Leung, and C. Chu, "Post-placement voltage island generation," in *Proc. IEEE/ACM ICCAD*, Nov. 2006, pp. 641–646.

[5] A. Goldberg and R. Tarjan, "Solving minimum-cost flow problems by successive approximation," in *Proc. 19th Annu. ACM STOC*, 1987, pp. 7–18.

[6] A. V. Goldberg, "An efficient implementation of a scaling minimum-cost flow algorithm," *J. Algorithms*, vol. 22, no. 1, pp. 1–29, 1997.

[7] J. Hu, Y. Shin, N. Dhanwada, and R. Marculescu, "Architecting voltage islands in core-based system-on-a-chip designs," in *Proc. ISLPED*, 2004, pp. 180–185.

[8] W.-L. Hung, G. M. Link, Y. Xie, N. Vijaykrishnan, N. Dhanwad, and J. Conner, "Temperature-aware voltage islands architecting in system-on-chip design," in *Proc. ICCD*, 2005, pp. 689–696.

[9] D. E. Lackey, P. S. Zuchowski, T. R. Bednar, D. W. Stout, S. W. Gould, and J. M. Cohn, "Managing power and performance for system-on-chip designs using voltage islands," in *Proc. IEEE/ACM ICCAD*, Nov. 2002, pp. 195–202.

[10] W.-P. Lee, H.-Y. Liu, and Y.-W. Chang, "Voltage island aware floorplanning for power and timing optimization," in *Proc. IEEE/ACM ICCAD*, Nov. 2006, pp. 389–394.

[11] W.-P. Lee, H.-Y. Liu, and Y.-W. Chang, "An ILP algorithm for post-floorplanning voltage-island generation considering power-network planning," in *Proc. IEEE/ACM ICCAD*, Nov. 2007, pp. 650–655.

[12] B. Liu, Y. Cai, Q. Zhou, and X. Hong, "Power driven placement with layout aware supply voltage assignment for voltage island generation in dual-Vdd designs," in *Proc. ASP-DAC*, 2006, pp. 582–587.

[13] Q. Ma and E. F. Y. Young, "Voltage island-driven floorplanning," in *Proc. IEEE/ACM ICCAD*, Nov. 2007, pp. 644–649.

[14] Q. Ma and E. F. Y. Young, "Network flow-based power optimization under timing constraints in MSV-driven floorplanning," in *Proc. IEEE/ACM ICCAD*, Nov. 2008, pp. 1–8.

[15] W.-K. Mak and J.-W. Chen, "Voltage island generation under performance requirement for SoC designs," in *Proc. ASP-DAC*, 2007, pp. 798–803.

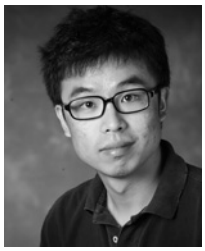
[16] Z. Qian and E. F. Y. Young, "Multi-voltage floorplan design with optimal voltage assignment," in *Proc. ISPD*, 2009, pp. 13–18.

[17] V. Stix, "Target-oriented branch and bound method for global optimization," *J. Global Optimiz.*, vol. 26, no. 3, pp. 261–277, 2003.

[18] X. Tang and D. F. Wong, "Fast-sp: A fast algorithm for block placement based on sequence pair," in *Proc. ASP-DAC*, 2001, pp. 521–526.

[19] H. Wu, I.-M. Liu, M. D. F. Wong, and Y. Wang, "Post-placement voltage island generation under performance requirement," in *Proc. IEEE/ACM ICCAD*, Nov. 2005, pp. 309–316.

[20] H. Wu, M. D. F. Wong, and I.-M. Liu, "Timing-constrained and voltage-island-aware voltage assignment," in *Proc. 43rd Annu. DAC*, 2006, pp. 429–432.



Qiang Ma received the B.E. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2006, and the M.Phil. degree in computer science from the Chinese University of Hong Kong, Shatin, Hong Kong, in 2008. He is currently pursuing the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, Urbana.

His current research interests include physical design of chips, packages, and printed circuit boards.

Zaichen Qian received the B.E. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2007, and the M.Phil. degree in computer science from the Chinese University of Hong Kong, Shatin, Hong Kong, in 2010.

Evangeline F. Y. Young received the B.S. and M.Phil. degrees in computer science from the Chinese University of Hong Kong (CUHK), Shatin, Hong Kong, and the Ph.D. degree from the University of Texas at Austin, Austin, in 1999.

She is currently an Associate Professor with the Department of Computer Science and Engineering, CUHK. Her current research interests include algorithms and computer-aided design of very large scale integration circuits. She is now working actively on floorplanning, placement, routing, and algorithmic designs.

Dr. Young has served on the technical program committees of several major conferences, including ICCAD, ASP-DAC, ISPD, and GLSVLSI, and also the Editorial Board of IEEE Transactions on Computer-Aided Design.

Hai Zhou received the B.S. and M.S. degrees in computer science and technology from Tsinghua University, Beijing, China, in 1992 and 1994, respectively, and the Ph.D. degree in computer sciences from the University of Texas at Austin, Austin, in 1999.

He is currently an Associate Professor with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL. His current research interests include very large scale integrated computer-aided design, algorithm design, and formal methods. He has published more than 70 technical papers in prestigious journals and conferences in these areas.

Prof. Zhou was a recipient of the CAREER Award from the National Science Foundation in 2003. He served on the Technical Program Committees of ACM Design Automation Conference, IEEE International Conference on Computer-Aided Design, and many other conferences.