Low Power Discrete Voltage Assignment Under Clock Skew Scheduling

Li Li¹, Jian Sun², Yinghai Lu¹, Hai Zhou¹, Xuan Zeng²

¹Electrical Engineering and Computer Science Department, Northwestern University, U.S.A.

²State Key Lab. of ASIC & System, Microelectronics Department, Fudan University, China

Abstract—Multiple Supply Voltage (MSV) assignment has emerged as an appealing technique in low power IC design, due to its flexibility in balancing power and performance. However, clock skew scheduling, which has great impact on criticality of combinational paths in sequential circuit, has not been explored in the merit of MSV assignment. In this paper, we propose a discrete voltage assignment algorithm for sequential circuit under clock scheduling. The sequential MSV assignment problem is first formulated as a convex cost dual network flow problem, which can be optimally solved in polynomial time assuming delay of each gate can be chosen in continuous domain. Then a mincut-based heuristic is designed to convert the unfeasible continuous solution into feasible discrete solution while largely preserving the global optimality. Besides, we revisit the hardness of the general discrete voltage assignment problem and point out some misunderstandings on the approximability of this problem in previous related work. Benchmark test for our algorithm shows 9.2% reduction in power consumption on average, in compared with combinational MSV assignment. Referring to the continuous solution obtained from network flow as the lower bound, the gap between our solution and the lower bound is only 1.77%.

I. INTRODUCTION

Power consumption has become one of the major concerns in modern IC design, especially for portable devices. To control the trade-off between power and performance, low power design techniques such as Multiple Supply Voltage (MSV) [1] and Multiple Threshold Voltage (MTV) [2] assignment have drawn much attention of the EDA community. In MSV design, high supply voltage is assigned to timing critical cells to guarantee performance, while low supply voltage is assigned to other cells to save power consumption. In MTV design, low threshold voltage is assigned to timing critical cells in a similar way. Since MSV assignment and MTV assignment problem share very similar structure, an algorithm that applies to one can usually apply to the other as well. Therefore, both MSV and MTV assignment can be classified as the general *voltage assignment* problem, which is to minimize total power consumption by assigning different voltages to the circuit elements without violating the given timing constraint. This problem has become intensively studied in the literature [3], [4], [5], [6], [7], [2]. In [3], the authors proposed a heuristic to minimize power consumption and to simplify the power network resource under timing constraints simultaneously. In [4], the authors tackled the voltage assignment problem by formulating it into an *Integer Linear Programming* (ILP) with inequalities representing timing requirements. Dynamic programming is used in [5] to solve this problem. However, all of these approaches suffer from either non-optimality or long run-time.

In [6], Ma et al. formulated the MSV assignment problem as a convex cost integer dual network flow problem [8] after relaxing the discreteness requirement of supply voltages. The convex cost dual problem can be solved by a cost-scaling algorithm, which is computationally much more efficient than solving the general ILP problem. Straightforward rounding is then used to obtain the discrete solution, which has no optimality guarantee even though the continuous solution is optimal. In order to resolve the unfeasibility of the continuous solution, a branch and bound based algorithm is proposed in [7], which uses the method in [6] as tight bound estimation. While the proposed method is able to obtain optimal discrete solution, the computational complexity of branch and bound is not guaranteed. Recently, Feng et al. [2] claimed an ϵ -approximation to threshold voltage assignment problem under timing constraints. However, we

find out that previous work [9], [10] has ruled out the possibility of existence of any ϵ -approximation to the discrete voltage assignment problem unless $P = NP$. After our careful examination, we found out that there are some flaws in their approach. We will discuss these issues in Section V.

Despite of good efforts from aforementioned work, the voltage assignment is still restricted in combinational circuit. It is well known that the criticality of the combinational paths can be magnificently changed by sequential adjustments such as retiming and clock skew scheduling. Both retiming and clock skew scheduling have been utilized in various optimization problems such as gate sizing [11], threshold voltage assignment [12] and slack budgeting [13]. These works have motivated us to incorporate appropriate sequential optimization techniques when handling the voltage assignment problem.

In this paper, we propose an algorithm for the discrete voltage assignment problem in sequential circuit under clock skew scheduling. The main contributions of this paper include:

- The discrete voltage assignment problem under clock skew scheduling is properly formulated for the first time as a convex cost integer dual network flow problem which has optimal continuous solution;
- A novel mincut-based heuristic is devised to obtain a feasible solution from the optimal unfeasible solution. Its effectiveness and efficiency is justified by theoretical analysis as well as verified by experimental results;
- Serious mistakes are identified in an ϵ -approximation approach to the MTV assignment problem [2] and the impossibility of existence of any ϵ -approximation to this problem is pointed out.
- Our approach is highly practical. According to experimental results on all benchmarks of ISCAS89, our method can achieve 9.2% power saving improvement in comparison with combinational MSV assignment [6]. It also turns out that our solution is nearly optimal. Referring to the optimal unfeasible solution obtained from network flow problem as the theoretical lower bound of optimal feasible solution, the gap between our final solution and the lower bound is only 1.77%.

The rest of this paper is organized as follows. The definition and formulation of sequential voltage assignment problem are given in Section II. The hardness and approximability of the general voltage assignment problem is discussed in Section III. Section IV describes the reduction to convex cost flow problem and the corresponding cost-scaling algorithm. The mincut-based heuristic is described in Section V, and the flaws of [2] are also discussed in this section. Section VI presents the experimental results. Finally, Section VII concludes the paper.

II. PROBLEM FORMULATION

The input to voltage assignment problem includes a sequential circuit consisting of a set of properly connected combinational gates and flip-flops (FF). Each gate has several supply/threshold voltage candidates with fixed (*delay*, *power*) pair, which can be computed offline. All (*delay*, *power*) pairs of a gate compose a *delay-power curve* (DP curve), which is typically a convex function representing the relation of delay-power tradeoff. The objective of voltage assignment problem is to minimize the total power consumption of

the circuit while subjecting to various timing constraints, which is explained as follows.

Given a sequential circuit, a directed graph $G = (V, E)$ can be constructed as follows. Node set V consists of two subsets V_i, V_o and two dummy nodes PI, PO . PI and PO are connected to all primary inputs and outputs respectively. Each gate or FF is represented by an input vertex $v_i \in V_i$ and an output vertex $v_o \in V_o$, and the input vertex is connected to the output vertex by a directed edge. Edge set E is composed of three subsets E_{gate}, E_{net} and E_{FF} . E_{gate} and E_{FF} is the set of all edges between the input vertex and output vertex of a gate and FF, respectively. A directed edge $(i, j) \in E_{net}$ denotes an interconnect between the gates and FFs corresponding to vertex i and j . An example of G is as shown in Figure 1.

Fig. 1. An example of sequential circuit and its constraint graph G

For each $(i, j) \in E$, we use d'_{ij} to denote gate delay, net delay or FF delay on edge (i, j) . Moreover, we use $\bar{P}_{ij}$ to represent the function that maps delay to power on (i, j) . For $(i, j) \in E_{gate}$, $\bar{P}_{ij}(d'_{ij}) = p_{ij}^k$ for $1 \leq k \leq k_{ij}$ where k_{ij} is the number of possible voltage options of each gate, and $\bar{P}_{ij} = 0$ for $(i, j) \in E_{net} \cup E_{FF}$.

On combinational paths, the arrival time t_i of a node i satisfies $t_i + d_{ij} \leq t_j$, $\forall (i, j) \in E_{net} \cup E_{gate}$. In addition, for $(i, j) \in E_{FF}$, the clock skew on each FF in sequential circuit needs to be bounded in $[0, s_{max}]$, where s_{max} is the maximum acceptable clock skew. The influence of clock skew can be represented by the following inequalities:

$$t_j \geq s_{ij} + d'_{ij} \wedge t_i \leq s_{ij} + T \wedge 0 \leq s_{ij} \leq s_{max},$$

where s_{ij} stands for clock skew of the FF embedded in $(i, j) \in E_{FF}$ and T is the clock period. In order to facilitate the problem formulation, we transform the above inequalities into its equivalent form to eliminate variable s_{ij} :

$$t_j \geq t_i - T + d'_{ij} \wedge t_j \geq d'_{ij} \wedge t_i \leq T + s_{max}$$

After the transformation, the timing constraint on each edge $(i, j) \in E$ can be uniformly stated as $t_i \leq t_j - d_{ij}$, where

$$d_{ij} = \begin{cases} d'_{ij}, (i, j) \in E_{gate} \cup E_{net} \\ d'_{ij} - T, (i, j) \in E_{FF} \end{cases}$$

Based on deduction, the *Discrete Voltage Assignment Problem under Clock Skew Scheduling* problem can be formulated into the

following mathematical program:

$$\begin{aligned} \text{Min} \quad & \sum_{(i,j) \in E_{gate}} \bar{P}_{ij}(d_{ij}) \\ \text{s.t.} \quad & t_i \geq 0, \quad i = PI \quad (1a) \\ & t_i \leq T, \quad i = PO \quad (1b) \\ & t_i \leq t_j - d_{ij}, \quad \forall (i, j) \in E \quad (1c) \\ & t_j \geq d'_{ij}, \quad \forall (i, j) \in E_{FF} \quad (1d) \\ & t_i \leq T + s_{max}, \quad \forall (i, j) \in E_{FF} \quad (1e) \\ & d_{ij} \in \{d_{ij}^1, \dots, d_{ij}^{k_{ij}}\}, \quad \forall (i, j) \in E_{gate} \quad (1f) \end{aligned}$$

In Problem 1, the first two inequities assert the boundary constraints of the whole circuit. The third one is the general arrival time constraint on each timing edge. The fourth and fifth inequities are the clock skew constraints for flip-flops. Finally, the last constraint states the discreteness of the valid voltage candidate.

III. HARDNESS OF DISCRETE VOLTAGE ASSIGNMENT

Discrete voltage assignment is well-known as a NP-hard problem. Feng et al. have studied the ϵ -approximation to the MTV assignment problem in [2]. Unfortunately, their solution turns out to be flawed after our careful analysis. In fact, some early research in relative areas has revealed impossibility of any ϵ -approximation to this problem unless $P = NP$. The discrete voltage assignment is actually a specific application of the general *Discrete Time-Cost Tradeoff* (DTCT) problem, which has been widely studied in the area of operation research and management science. De et al. has proved in [10] that the DTCT problem is strongly NP hard and can not be solved in pseudo-polynomial time unless $P = NP$. [9] further show that there does not exist a polynomial $(1 + \alpha, \frac{5}{4} - \beta)$ -approximation of DTCT with some $\alpha, \beta > 0$ unless $P = NP$. So far the best approximation algorithm [14] has been found to the DTCT problem can only guarantee a performance of $O(\log l)$, where l is the ratio of the maximum duration ("delay") of any activity ("gate") to the minimum nonzero duration of any activity. No constant-approximation to the DTCT problem has been found except for some special cases.

Next, we would like to point out the misunderstandings in [2]. In the first step, the authors transform the original problem into a convex min-max resource-sharing LP, which is Formula (7) in [2] :

$$\begin{aligned} \text{Min} \quad & \lambda \\ \text{s.t.} \quad & \sum_{k, (i,j) \in E_{gate}} p_{ij}^k x_{ij}^k - P + 1 \leq \lambda, \\ & t_i + \sum_k d_{ij}^k x_{ij}^k - t_j + 1 \leq \lambda, \quad \forall (i, j) \in E_{gate} \quad (2) \\ & \sum_k x_{ij}^k = 1, \quad \forall (i, j) \in E_{gate} \\ & x_{ij}^k \in [0, 1], \quad \forall 1 \leq k \leq k_{ij} \end{aligned}$$

In Problem 2, x_{ij}^k indicates the voltage choice of gate (i, j) . When the k th voltage is assigned, x_{ij}^k equals to 1. Otherwise, it is set to 0. d_{ij}^k, p_{ij}^k are the gate delay and power consumption corresponding to the k th voltage. P is the estimated value of the optimal total power consumption. Notice that the introduction of x, P and λ does not change the nature of this problem. At a high level, their algorithm strives to get an ϵ -approximation of P by iteratively solving the ϵ -approximation to Problem 2 [15]. The key point of their argument is that the ϵ -approximation to Problem 2 is also the ϵ -approximation to binary integer programming version of the same problem with all x_{ij}^k constrained to binary values. We will demonstrate that this claim is incorrect by a simple example. Considering $\lambda^* = \min\{\lambda | x_1 \leq \lambda, x_2 \leq \lambda, x_1 + x_2 = 1, x_1 \in [0, 1], x_2 \in [0, 1]\}$, it is easy to find $\lambda^* = 1/2$, at this time $x_1 = x_2 = 1/2$. If we constrain x_1 and x_2 to be binary values, the optimal solution will be either x_1 or x_2 to be 1. Obviously, no ϵ -approximation to the continuous problem will also be ϵ -approximate to the discrete problem. Another flaw in [2]

is that the authors tackle all t_i as constants when solving Problem 2 as a LP about x . They claim that it is easy to know t_i once x is determined. This is an inversion of consequence because fixing t_i at any constants is equal to constraining the slack and the range of $x_{ij}^{k_{ij}}$.

In this paper, we do not seek for approximation to the discrete voltage assignment problem. Instead, we will first solve the optimal solution of its continuous form and use it as a good starting point for finding the feasible solution. Based on the continuous solution, we further propose an efficient heuristic to obtain the discrete solution while striving for maintaining the global optimality. The two stages of the proposed approach will be described in the following sections.

IV. CONTINUOUS RELAXATION

With the last constraint relaxed to the continuous domain $d_{ij} \in [d_{ij}^1, d_{ij}^{k_{ij}}]$, Problem (1) is a *Convex Cost Dual Network Flow* problem [8], which can be optimally solved in polynomial time. In this section, we describe the details of the continuous algorithm based on convex cost flow.

First, the neighboring points in DP curve are connected with line segments, resulting in a linear-piecewise convex function of power versus delay. Next, in order to eliminate the lower and upper bound of the delay, we generate a new power-delay function as follows.

$$P_{ij}(d_{ij}) = \begin{cases} \bar{P}_{ij}(d_{ij}^{k_{ij}}), & d_{ij} \geq d_{ij}^{k_{ij}} \\ \bar{P}_{ij}(d_{ij}^q) + b_{ij}^q(d_{ij} - d_{ij}^q), & d_{ij}^q \leq d_{ij} \leq d_{ij}^{q+1} \\ \vdots & \\ \bar{P}_{ij}(d_{ij}^1) - M(d_{ij} - d_{ij}^1), & d_{ij} \leq d_{ij}^1 \end{cases}$$

where $b_{ij}^q = \frac{P_{ij}(d_{ij}^q) - P_{ij}(d_{ij}^{q-1})}{d_{ij}^q - d_{ij}^{q-1}}$ is the slope of power-delay function between d_{ij}^q and d_{ij}^{q-1} . M is a sufficiently large number which can be viewed as a penalty factor for violating the bound constraints.

Similarly, in order to eliminate the bound constraints on the arrival time t_i , we define $B_i(t_i)$ as follows and put it into the objective function.

$$B_i(t_i) = \begin{cases} M \times (t_i - u_i), & t_i > u_i \\ 0, & 0 \leq t_i \leq T \\ -M \times (t_i - l_i), & t_i < l_i \end{cases}$$

where l_i and u_i are the lower and upper bounds of t_i . Without the bound constraints and discretization constraints Problem (1) can be transformed into the following form:

$$\begin{aligned} \text{Min} \quad & \sum_{(i,j) \in E} P_{ij}(d_{ij}) + \sum_{i \in V} B_i(t_i) \\ \text{s.t.} \quad & t_i \leq t_j - d_{ij}, \quad \forall (i,j) \in E \end{aligned} \quad (3)$$

It is well know that the dual of Problem 3 [8] is a convex cost problem.

$$\begin{aligned} \text{Min} \quad & \sum_{(i,j) \in E^*} C_{ij}(x_{ij}) \\ \text{s.t.} \quad & \sum_{(i,j) \in E^*} x_{ij} - \sum_{(j,i) \in E^*} x_{ji} = 0 \quad \forall i \in V^* \\ & 0 \leq x_{ij} \leq M \quad \forall (i,j) \in E^* \end{aligned} \quad (4)$$

In Problem 4, we add an extra *ground* node 0 and its connections to nodes in V to facilitate the problem. For each node $i \in V$ with a lower bound on arrival time t_i (PI and outputs of all flip-flops), an edge $(0, i)$ is introduced. Moreover, for each node $i \in V$ with an upper bound on arrival time t_i (PO and inputs of all flip-flops), an edge $(i, 0)$ is introduced. We denote the set of newly added edges by E_0 and the new graph by $G^* = (V^*, E^*)$ so that $V^* = \{0\} \cup V$ and $E^* = E_{gate} \cup E_{net} \cup E_{FF} \cup E_0$.

The cost function of each type of edges can be derived from the power functions and arrival time constraints using Lagrangian

relaxation [8], [6]. For each $(i, j) \in E_{gate}$, the cost function is

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

For each $(i, j) \in E_{net}$:

$$C_{ij}(x_{ij}) = -d'_{ij} x_{ij}, \quad 0 \leq x_{ij} \leq M.$$

For each $(i, j) \in E_{FF}$:

$$C_{ij}(x_{ij}) = (T - d'_{ij}) x_{ij}, \quad 0 \leq x_{ij} \leq M.$$

And finally for each $(i, j) \in E_0$:

$$C_{ij}(x_{ij}) = \begin{cases} -d'_{0i} x_{0i}, & \forall (k, i) \in E_{FF} \\ (T + s_{max}) x_{i0}, & \forall (i, k) \in E_{FF} \\ 0, & i = 0, j = PI \\ T, & i = PO, j = 0 \end{cases}$$

The convex-cost scaling algorithm in [8] can be applied to obtain the optimal solution to Problem 4. The general flow of the algorithm is as shown in Algorithm 1.

Algorithm 1 Convex Cost-Scaling Algorithm

```

1: Initialize  $\pi \leftarrow 0$  and  $\epsilon \leftarrow \max_{(i,j) \in E} |b_{ij}^{k_{ij}}|$ 
2: Initialize flow  $x \leftarrow 0$ 
3: while  $\epsilon \geq 1/|V|$  do
4:   for For each admissible edge  $(i, j) \in E(x)$  do
5:     Push  $q(i, j)$  flow through  $(i, j)$ 
6:   end for
7:   Examine the node for excess
8:   while there is a node  $i$  with excess flow do
9:     if there is an admissible edge  $(i, j)$  then
10:      push  $\min(X(i), q(i, j))$  flow through  $(i, j)$ 
11:     else
12:       $\pi(i) \leftarrow \pi(i) + \epsilon/2$ 
13:     end if
14:   end while
15: end while
16: return  $x$  and  $\pi$ 

```

In the convex-cost algorithm, a pseudoflow x that satisfies upper and lower capacity constraints for all edge but might violate the flow conservation constraints is maintained. The excess of a node i is defined as $X(i) = \sum_{(j,i) \in E^*} x_{ji} - \sum_{(i,j) \in E^*} x_{ij}$ for all $i \in V^*$. Node i has excess if $X(i) > 0$. A potential $\pi(i)$ is attached to each node i , which is intrinsically the arrival time t_i . The algorithm proceeds by pushing flows in a residual network $G_r^*(x)$, in which the residual capacity $q(i, j)$ of each edge $(i, j) \in E^*$ is defined as

$$q(i, j) = -\mathcal{P}_{ij}^-(\lfloor \pi(j) - \pi(i) \rfloor) - x_{ij}$$

for forward edge and

$$q(j, i) = \mathcal{P}_{ij}^+(\lfloor \pi(j) - \pi(i) \rfloor) + x_{ij}$$

for backward edge, where $\mathcal{P}_{ij}^-$ and $\mathcal{P}_{ij}^+$ are the left and right slope of power function P_{ij} . For a given residual network $G_r^*(x)$, the residual cost of each edge $(i, j) \in E^*$ is defined as

$$c_{ij}^\pi = C_{ij}^+(x_{ij}) - \pi(i) + \pi(j)$$

for forward edge and

$$c_{ji}^\pi = -C_{ij}^-(x_{ij}) - \pi(j) + \pi(i)$$

for backward edge, where C_{ij}^- and C_{ij}^+ are the left slope and right slope of cost function C_{ij} . We define a pseudoflow to be ϵ -optimal for $\epsilon > 0$ if $\exists \pi$ satisfies that $c_{ij}^\pi \geq -\epsilon$ for every $(i, j) \in G_r^*(x)$. Initially, ϵ is large enough to make any feasible flow ϵ -optimal. Then the algorithm gradually transforms the ϵ -optimal solution into $\epsilon/2$ -optimal solution by pushing as much as possible flow through admissible edges, which satisfy the condition $-\epsilon \leq c_{ij}^\pi < 0$. The algorithm terminates when $\epsilon < 1/|V|$.

Though the convex-cost scaling algorithm is based on integer costs, the optimal flow x^* and node potential π^* returned by Algorithm 1 may be non-integer. But since all edge costs are integers, there must exist some integral optimal node potential π . Such π can be obtained by computing the shortest distance $sp(i)$ from node 0 to every other node $i \in V^*$ in the residual graph $G_r^*(x^*)$ and set $\pi(i) = -sp(i)$. The optimal solution to Problem 3 can be then obtained by assigning $t_i = \pi(i)$ and $d_{ij} = t_j - t_i$ for each $(i, j) \in E_{gate}$. Additionally, clock skew s_{ij} for FF embedded in edge (i, j) is feasible with any value in $[t_i - T, t_j - d'_{ij}]$. The time complexity of Algorithm 1 is $O(|V||E|\log(|V|^2/|E|)\log(|V|T))$, where $|V|$, $|E|$, T is the number of nodes, the number of edges and the cycle time of circuit, respectively.

Note that since some d_{ij} may not be on the discrete levels $\{d_{ij}^1, \dots, d_{ij}^{k_{ij}}\}$, this solution may be unfeasible and therefore can not be directly implemented in the circuits. The continuous solution requires further discretization processing, which will be discussed in the next section.

V. DISCRETE ALGORITHMS

Most of the previous LP-based or network flow-based approaches to the discrete voltage assignment problem [8], [16], [17] simply floor the optimal continuous delay d_{ij} to largest possible delay option $d_{ij}^{q_{ij}}$ to obtain a feasible solution. However, it is not guaranteed that the optimal continuous solutions will automatically fall on the expected discrete levels. In some cases, naive rounding may produce extra slacks that can be reused to further reduce power. In this section, we will present a mincut-based heuristic which can fully utilize the extra slacks while maintaining low power consumption.

Assume that the optimal continuous solution is already available, Problem 1 can be further reduced by removing flip-flop edges from the graph. The new voltage assignment problem can be stated as follows.

$$\text{Min} \quad \sum_{(i,j) \in E_{gate}} P(d_{ij}) \quad (5a)$$

$$\text{s.t.} \quad t_i + d_{ij} \leq t_j, \quad \forall (i, j) \in E' \quad (5b)$$

$$d_{ij} \in \{d_{ij}^1, \dots, d_{ij}^{k_{ij}}\}, \quad \forall (i, j) \in E_{gate} \quad (5c)$$

The new graph $G' = (V, E')$ is slightly different from the original graph G . All edges $(i, j) \in E_{FF}$ are canceled and for each FF two new edges (PI, j) and (i, PO) are introduced with $d(PI, j) = s_{ij}^*$ and $d(i, PO) = -s_{ij}^*$, in which s_{ij}^* is the optimal clock skew of flip-flop corresponding to edge (i, j) . Additionally, we introduce one edge from PO to PI with $d(PO, PI) = -T$ to eliminate the arrival time constraints on single nodes. Let's denote the new added edges by E_{nFF} so that $E' = E_{net} \cup E_{gate} \cup E_{nFF}$. Notice the new graph G' is a DAG without (PO, PI) , canceling cycles in the graph will allow us to better explore the special structure of this problem.

Let us first look back at the relaxed version of Problem 5 from a different perspective. According to [18], the *Karush-Kuhn-Tucker* (KKT) optimality condition for relaxed Problem 5 is:

$$t_i + d_{ij} \leq t_j \quad \forall (i, j) \in E' \quad (6a)$$

$$-\mathcal{P}_{ij}^+(d_{ij}) \leq x_{ij} \leq -\mathcal{P}_{ij}^-(d_{ij}) \quad \forall (i, j) \in E' \quad (6b)$$

$$\sum_{(i,j) \in E'} x_{ij} - \sum_{(j,i) \in E'} x_{ji} = 0 \quad \forall i \in V' \quad (6c)$$

$$x_{ij} \geq 0 \quad \forall (i, j) \in E' \quad (6d)$$

$$x_{ij}(t_j - t_i - d_{ij}) = 0 \quad \forall (i, j) \in E' \quad (6e)$$

The above KKT condition actually describes a network flow satisfying certain properties. Initially, if we set all $x_{ij} = 0$, $d_{ij} = d_{ij}^{k_{ij}}$, $t_{PI} = 0$ and all other t_i as maximum delay from PI to i , the only violated constraint is timing constraint (6a) on edge (PO, PI) . Since the KKT condition is close to be satisfied in the beginning, it is natural to develop a way of decreasing delay while maintaining flow constraints. Constraints (6b) (6c) (6d) can be easily maintained by setting upper/lower capacity of each edge (i, j) as $-\mathcal{P}_{ij}^-(d_{ij})/-\mathcal{P}_{ij}^+(d_{ij})$ and pushing flow along cycles from PI to PO and back to PI . The dual constraint (6e) specifies that if there is positive flow on an edge (i, j) then equation $t_j = t_i + d_{ij}$ must hold, which means (i, j) must be on the longest path from PI to PO . In the following text we will refer the maximum delay from PI to PO as $MaxDelay(PI, PO)$ and the set of all edges in the longest paths as critical network. In order to satisfy constraint (6e), we have to push flows in the critical network. Once flow x_{ij} reaches its upper/lower bound $-\mathcal{P}_{ij}^-(d_{ij})/-\mathcal{P}_{ij}^+(d_{ij})$, we will be able to adjust d_{ij} without violating constraint (6b). However, the edges with positive flow may not be in the critical network after changing. In order to maintain dual constraint (6e), we expect the critical network to satisfy an important property called *Monotonic Grow*. *Monotonic Grow* ensures that once an edge is in the critical network, it will always be in the critical network. The way we use here is to change delay of the edges that on the mincut of the residual critical network. All details are as shown in Algorithm 2.

Algorithm 2 Mincut-Based Continuous Algorithm

- 1: Initialize $d_{ij} \leftarrow d_{ij}^{k_{ij}}$ $x_{ij} \leftarrow 0 \quad \forall (i, j) \in E'$
  - 2: Initialize $c_{ij} \leftarrow -\mathcal{P}_{ij}^-(d_{ij})$ $c_{ji} = 0 \quad \forall (i, j) \in E'$
  - 3: **while** $MaxDelay(PI, PO) > T$ **do**
  - 4:   Identify critical network G^0
  - 5:   Solve maximum flow in the residual graph of G^0 using s as source and t as sink
  - 6:   Select a mincut M and maximum possible δd
  - 7:   **for each** $(i, j) \in M$ **do**
  - 8:   **if** (i, j) is a forward edge **then**
  - 9: $d_{ij} = d_{ij} - \delta d$
  - 10:   **else**
  - 11: $d_{ij} = d_{ij} + \delta d$
  - 12:   **end if**
  - 13:   **end for**
  - 14: $c_{ij} \leftarrow -\mathcal{P}_{ij}^-(d_{ij}) - x_{ij} \quad \forall (i, j) \in G^0$
  - 15: $c_{ji} \leftarrow \mathcal{P}_{ij}^+(d_{ij}) + x_{ij} \quad \forall (i, j) \in G^0$
  - 16: **end while**
-

The basic idea behind our mincut-based continuous algorithm is similar as [19], but [19] only explained the concept of mincut on a price-oriented perspective and didn't further explore the deeper relation between the flow and the delay. It is also worth point out that computationally Algorithm 2 may be not as efficient as Algorithm 1 for the power/delay tradeoff problem with a specific T . However, Algorithm 2 provides us with a theoretical foundation to develop an advanced discrete algorithm. After several modifications, we immediately get a mincut-based heuristic as shown in Algorithm 3.

There are four main changes in our mincut discrete heuristics:

- 1) d_{ij} can only switch from one discrete level to another rather than changing continuously. If an edge (i, j) is on the mincut and its current delay is $d_{ij}^{q_{ij}}$, then we will directly reduce the delay to the next smaller discrete level $d_{ij}^{q_{ij}-1}$.
- 2) d_{ij} will be initialized as the smallest delay option $d_{ij}^{m_{ij}}$ that is not less than optimal continuous delay, instead of the largest delay option $d_{ij}^{k_{ij}}$. Such initialization will not only be able to take full advantage of the global optimality of the optimal continuous solution, but also significantly shorten the run-time of the algorithm.

Algorithm 3 Mincut-based Discrete Heuristics

```

1: Initialize  $q_{ij} \leftarrow m_{ij}$   $x_{ij} \leftarrow 0 \forall (i, j) \in E'$ 
2: Initialize  $c_{ij} \leftarrow -\mathcal{P}_{ij}^-(d_{ij}^{q_{ij}}) \forall (i, j) \in E'$ 
3: while  $MaxDelay(PI, PO) > T$  do
4:   Identify approximate critical network  $G^0$ 
5:   Solve maximum flow in the residual graph of  $G^0$  using  $s$  as
     source and  $t$  as sink
6:   Select a mincut  $M$ 
7:    $q_{ij} \leftarrow q_{ij} - 1 \forall \text{forward edge } (i, j) \in M$ 
8:    $c_{ij} \leftarrow -\mathcal{P}_{ij}^-(d_{ij}^{q_{ij}}) - x_{ij} \forall (i, j) \in E'$ 
9: end while

```

- 3) The backward edges on the mincut will not be taken into account. Based on our observation and experiments, we find backward mincut edges rarely appear in the circuit network in our problem. Actually whether omitting the backward edges have very little impact on our results.
- 4) The range of critical network will be expanded. In our experiments, the approximate critical network is defined as the set of all paths with delay larger than 95% of the maximum delay.

The runtime of Algorithm 3 is dominated by the time of solving maximum flow. The upper bound of the runtime is $O(knM)$, where k, n, M is the number of voltage options, the number of gates and the average time of solving maximum flow, respectively. In each while loop, at least one gate will reduce delay. Since the delay of each gate can be reduced for at most k times, there are at most kn while loops. However, since we start from the rounding solution and the mincut usually contains more than one gate, the number of while loops is much less than the order of kn in practice. Experimental results have shown that our discrete heuristic almost run in the same time order with the convex-cost scaling algorithm in the first stage.

VI. EXPERIMENTAL RESULTS

Our Voltage Assignment Clock Skew Scheduling algorithm SeqVA is implemented in C++. All of the experiments are performed on a Linux workstation with 3.0GHz CPU and 2.0GB memory. The proposed algorithm is tested on ISCAS89 benchmarks. The delay-power curve of each cell is simulated by HSPICE_U-2003.03-SP1 with supply voltage set as 0.8V, 1.0V, 1.2V and 1.4V respectively. After HSPICE simulation, we find that the assumption of convexity of delay-power curve is correct for all of the standard cells. An example of the delay-power curve of standard cell *buf1x* is shown in Figure 2.

Fig. 2. Delay-Power Curve of *buf1x*

Previous works on low power voltage assignment all focus on combinational circuit. To examine the performance of our algorithm, the combinational circuit voltage assignment algorithm in [6] is implemented for comparison, denoted as CombVA. The results are listed in Table I. Some statistics of each test case are listed including the number of gates and flip-flops, and the minimum possible period when all voltages are set to the largest level 1.4V. In our experiment,

clock period T is set to 1.1 times of the minimum period and maximum acceptable clock skew s_{max} is set to T . Total power consumption and running time of both CombVA and SeqVA are listed in columns titled “Power” and “CPUTime”. The power consumption for the continuous solution is also presented as a reference of the lower bound of all possible optimal results. Column “Gap” shows the difference between our result and lower bound.

As Table I shows, SeqVA can achieve 9.2% additional power reduction compared to CombVA. Besides that, since redundant arrival time constraints are eliminated on most of the combinational gates, the complexity of the graph construction is reduced dramatically, leading to less running time in solving continuous solution and less total running time. It turns out our solution is almost optimal, the achieved power consumption is only 1.77% larger than the lower bound on average.

It is mentioned in Section II that VACSS problem is equivalent to combinational voltage assignment problem when s_{max} is set to 0. In Figure 3 we show how the power ratio between SeqVA and CombVA changes as s_{max} changes from 0 to T . As the maximum acceptable clock skew increases, more power can be saved through clock skew scheduling. Similar properties can be observed on all test cases.

Fig. 3. Power Reduction vs. s_{max}

We also observe that the impact of clock skew scheduling is more significant under tight timing constraints. As shown in Figure 4, the power consumption of SeqVA is very close to the lower limit even when the timing constraints are very tight. If SeqVA sets T to $1.0T_{min}$ and $1.1T_{min}$, then CombVA needs to set T to $1.4T_{min}$ and $1.7T_{min}$ accordingly to achieve similar power consumption.

Fig. 4. Power Consumption vs. T

VII. CONCLUSIONS

In this paper, we proposed a low power discrete voltage assignment algorithm under clock skew scheduling. The problem is formulated in a continuous manner as a convex-cost network flow and optimally solved. Unlike previous work where the optimal continuous delay is directly floored to the nearest discrete level, an elaborate mincut-based heuristic is proposed to obtain a feasible solution from the

TABLE I
EXPERIMENT RESULTS COMPARISON OF SEQVA WITH COMBVA[6]

Original Circuits			CombVA[6]		SeqVA				
					Discrete			Continuous	
Cases	#Gate/#FF	Min-Period (ns)	Power (μ W)	CPUTime (s)	Power (μ W)	Reduction (%)	CPUTime (s)	Power (μ W)	Gap (%)
s27	9/3	0.20	11.78	<0.01	8.13	30.86	0.01	7.72	5.31
s208.1	51/5	0.34	54.05	0.01	49.68	8.09	0.01	47.25	5.14
s208	55/8	0.47	47.10	0.02	43.33	8.00	0.01	43.14	0.44
s298	76/14	0.31	52.09	0.02	42.75	17.93	0.02	41.89	2.05
s344	89/15	0.49	63.83	0.03	51.04	20.03	0.03	50.70	0.67
s349	91/15	0.49	65.37	0.03	52.13	20.25	0.03	51.67	0.89
s382	96/21	0.33	86.21	0.02	69.41	19.49	0.03	68.20	1.77
s386	80/6	0.26	90.81	0.02	86.79	4.43	0.04	82.90	4.69
s400	101/21	0.35	93.20	0.03	75.85	18.62	0.04	74.81	1.39
s420.1	112/16	0.55	92.57	0.03	83.54	9.75	0.05	81.69	1.25
s420	113/16	0.99	95.01	0.03	95.01	0.63	0.05	94.81	0.21
s444	123/21	0.37	112.61	0.04	83.07	26.23	0.05	81.67	1.71
s510	154/6	0.34	146.57	0.05	142.41	2.84	0.06	137.81	3.34
s526	133/21	0.31	94.92	0.04	83.11	12.44	0.06	81.56	1.90
s526n	134/21	0.31	93.76	0.04	82.41	12.10	0.07	80.39	2.51
s641	121/19	0.65	101.98	0.04	96.75	5.13	0.07	95.17	1.66
s713	126/19	0.65	108.88	0.04	101.10	7.15	0.08	99.23	1.88
s820	178/5	0.34	166.19	0.05	160.61	3.36	0.08	156.86	2.39
s832	186/5	0.34	169.98	0.05	163.71	3.69	0.08	159.94	2.36
s838	233/32	0.88	180.07	0.09	165.30	8.20	0.09	164.15	0.70
s953	318/29	0.63	242.73	0.11	212.42	12.49	0.12	210.62	0.85
s1196	318/18	0.64	253.37	0.13	246.69	2.64	0.17	242.33	1.80
s1238	343/18	0.70	262.20	0.14	257.41	2.95	0.19	252.08	2.11
s1423	357/74	2.63	259.18	0.19	230.01	11.25	0.25	229.21	0.35
s1488	331/6	0.35	257.66	0.12	255.61	0.80	0.26	246.62	3.64
s1494	331/6	0.34	259.01	0.11	255.53	1.34	0.26	244.39	4.56
s5378	889/164	0.50	681.97	0.57	661.52	3.00	0.61	648.22	2.05
s9234	1613/192	1.25	1082.00	1.54	985.03	8.96	1.25	984.41	0.06
s13207	2666/511	1.33	1678.71	3.96	1617.88	3.62	2.19	1615.44	0.15
s15850	3092/525	1.96	1961.96	4.88	1831.36	6.66	2.85	1830.62	0.04
s35932	12126/1728	0.84	13786.40	28.01	13628.4	1.22	20.83	13627.50	0.01
s38417	7039/1535	1.04	4621.69	13.10	4248.32	8.08	7.77	4236.32	0.28
s38584	8612/1426	1.35	5015.59	17.84	4963.07	1.05	8.66	4954.54	0.17
average	-	-	-	-	-	9.20	-	-	1.77

continuous solution while well preserving the global optimality. During the discussion, we also point out several misunderstandings in previous related works. Experiment results demonstrate the effectiveness and efficiency of our approach. Comparing to combinational supply voltage assignment [6], our method achieves more total power saving and faster running time. The gap between our final feasible solution and the lower bound of optimal solution is only 1.77%.

VIII. ACKNOWLEDGEMENT

This work was supported in part by the NSFC Research Projects 60976034 and 61076033, the National Basic Research Program of China under Grant 2005CB321701, the National Major Science and Technology Special Projects 2008ZX01035-001-06 and 2009ZX02023-4-3 of China during the 11th Five Year Plan Period, the Doctoral Program Foundation of the Ministry of Education of China 200802460068, the Program for Outstanding Academic Leader of Shanghai and NSF under CCF-0238484 and CCF-0811270.

REFERENCES

- [1] D. Lackey, P. Zuchowski, T. Bednar, D. Stout, S. Gould, and J. Cohn, "Managing power and performance for system-on-chip designs using voltage islands," in *ICCAD*, 2002.
- [2] Y. Feng and S. Hu, "The epsilon-approximation to discrete vt assignment for leakage power minimization," in *ICCAD*, 2009.
- [3] H. Wu, D. Wong, and I.-M. Liu, "Timing-constrained and voltage-island-aware voltage assignment," in *DAC*, 2006.
- [4] W.-P. Lee, H.-Y. Liu, and Y.-W. Chang, "Voltage island aware floorplanning for power and timing optimization," in *ICCAD*, 2006.
- [5] —, "An ILP algorithm for post-floorplanning voltage-island generation considering power-network planning," in *ICCAD*, 2007.
- [6] Q. Ma and F. Y. Young, "Network flow-based power optimization under timing constraints in MSV-driven floorplanning," in *ICCAD*, 2008.
- [7] Z. Qian and E. F. Young, "Multi-voltage floorplan design with optimal voltage assignment," in *ISPD*, 2009.
- [8] R. K. Ahuja, D. S. Hochbaum, and J. B. Orlin, "Solving the convex cost integer dual network flow problem," *Management Science*, vol. 49(7), pp. 950–964, July 2003.
- [9] V. G. Deineko and G. J. Woeginger, "Hardness of approximation of the discrete time-cost tradeoff problem," *Operation Research Letters*, vol. 29, pp. 207–210, 2001.
- [10] P. De, E. Dunne, J. Ghosh, and C. Wells, "Complexity of the discrete time-cost tradeoff problem for project networks," *Operation Research*, vol. 45, pp. 302–306, 1997.
- [11] J. Wang, D. Das, and H. Zhou, "Gate sizing by lagrangian relaxation revisited," in *ICCAD*, 2007.
- [12] M. Ni and S. O. Memik, "Leakage power-aware clock skew scheduling: Converting stolen time into leakage power reduction," in *DAC*, 2008.
- [13] S. Liu, Y. Ma, X.-L. Hong, and Y. Wang, "Simultaneous slack budgeting and retiming for synchronous circuits optimization," in *ASPCAD*, 2010.
- [14] M. Skutella, "Approximation algorithm for the discrete time-cost tradeoff problem," *Operation Research*, vol. 23, pp. 909–929, 1998.
- [15] K. Jansen and H. Zhang, "Approximation algorithms for general packing problems and their application to the multicast congestion problem," *Math Program*, vol. 114, pp. 183–206, 2008.
- [16] H. Chou, Y.-H. Wang, and C.-P. Chen, "Fast and effective gate sizing with multiple vt-assignment using generalized lagrangian relaxation," in *ASPDAC*, 2005.
- [17] S. Shah, A. Srivastava, D. Sharma, and D. Sylvester, "Discrete vt assignment and gate sizing using a self-snapping continuous formulation," in *DAC*, 2005.
- [18] R. Chen and H. Zhou, "An efficient algorithm for bugger insertion in general circuits based on network flow," in *ICCAD*, 2005.
- [19] S. Phillips and M. Dessouky, "Complexity of the discrete time-cost tradeoff problem for project networks," *Management Science*, vol. 24, pp. 393–400, 1977.