NORTHWESTERN
UNIVERSITY

Electrical Engineering and Computer Science Department

Technical Report
NWU-EECS-09-07
April 08, 2009

Fast Voltage Assignment by Convex-cost Flow

Stephen P. Tarzia **Hai Zhou** **Robert P. Dick**

Abstract

In this work, we cast the continuous voltage assignment problem as a minimum convex-cost network flow problem to solve it both optimally and efficiently. Experiments show that, on real circuits, our algorithm is two orders of magnitude (100X) faster than previous work [1]. Another research group independently developed and published the continuous voltage assignment algorithm that we describe [2]. The purpose of this technical report is to present an alternative mathematical derivation and experimental evaluation of the same algorithm. We also present a new and elegant dynamic-programming discrete voltage assignment heuristic.

This work was supported in part by the Semiconductor Research Corporation under award 2007-HJ-1593 and in part by the National Science Foundation under awards CCF-0702761 and CNS-0347941. Stephen P. Tarzia is supported by a Dr. John N. Nicholson fellowship.

Keywords: voltage assignment, network flow, convex programming, optimization, VLSI-CAD

I. Introduction

In Multiple Supply Voltage (MSV) digital circuits, we can slow portions of the circuit at design-time by powering them with reduced supply voltages. Voltage assignment is a dual-objective optimization problem; there is a tradeoff between delay and energy. In the *deadline* problem, the maximum circuit delay is given and energy is minimized. In the *budget* problem, the maximum circuit energy (or power) is given and delay is minimized. We focus on the deadline problem.

A. Related work

Other problem definitions: A great deal of work has been done on voltage assignment due to its huge energy saving potential. However, most of these works use very different problem definitions [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20]. Our problem definition is in many ways a simplification of these; we ignore both voltage-level shifter costs and floorplanning considerations. We believe that these complexities can be handled in a multi-stage design flow based on our basic voltage assignment algorithm. However, we do handle timing deadlines that span circuits with reconverging paths, so our timing model is among the most sophisticated. In addition, we do not presume a given set of available voltage levels as many work does. Instead, we show in Section IV how a set of discrete voltage levels can be chosen for each circuit. This added flexibility leaves more options available for energy reduction while staying within the limitations of practical power delivery.

Delay budgeting: The voltage assignment problem that we solve is based on *static timing analysis*. The circuit's connections form a directed acyclic graph of precedence constraints upon which we are to distribute timing slack. An optimal flow-based solution to linear-cost delay budgeting was introduced to the CAD community by Ghiasi et al. [21] and to a broader audience even earlier by Boros et al. [22]. Previous delay budgeting algorithms were path-based and suboptimal; typically a variant of the Critical Path Method called the Zero Slack Algorithm [23] has been used [24]. Lin et al. [25] give a flow formulation for an interconnect-delay budgeting problem with convex, rather than linear, costs. We use the same underlying network flow problem to solve a different CAD problem.

The continuous voltage assignment problem is analogous to a classical problem in Operations Research; it is an instance of the well-studied time-cost tradeoff in project management problem [26] [27] with non-linear costs [28]. Our main insight has been recognizing the voltage assignment problem as such. The discrete version of the problem that we solve in Section IV has not been considered in the project management literature, as far as we know.

Direct relatives: Niyogi and Marculescu [29] optimally solve the same continuous voltage assignment problem, but their formulation has an exponential number of constraints, one for each path in the circuit. Dabiri et al. [1] give a polynomial-time formulation for same problem. Our main contribution is an algorithm that is much faster in practice. The recent work of Ma et al. [2] presents essentially the same fast algorithm as our work, but was completed simultaneously and without our knowledge. Ma et al.go a step further to build a complete voltage assignment and voltage-island floorplanning design flow.

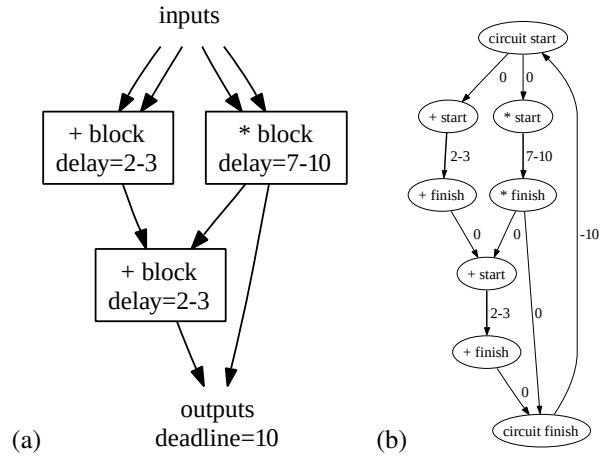

Fig. 1. (a) is an example data flow graph with two addition blocks and a multiplication block. (b) is the activity-on-arc timing graph representation of (a). Edge labels are delays, either a range of operating points or a fixed value.

B. The need for speed

An important premise of this work is that very fast voltage assignment algorithms are needed. We believe that in many design flows this is the case. For example, the high-level synthesis work by Gu et al. [24] runs a voltage assignment procedure in it's inner loop to evaluate the energy cost of each candidate design. A floorplanner might also run a voltage assignment procedure on each candidate design since varying interconnect delays would affect the voltage assignment. We present the first optimal continuous voltage assignment algorithm that is fast enough to run within such an inner loop.

In Section II we define the continuous voltage assignment subproblem and present for it a fast polynomial-time algorithm. We demonstrate its efficiency relative to previous work in Section III. Building on that, we present an elegant discrete voltage selection and assignment heuristic in Section IV and demonstrate its closeness to optimality in Section V.

II. Continuous voltage assignment

A. Problem definition

Informally, the continuous voltage assignment problem is to assign voltages to circuit blocks in a data flow graph (DFG) to minimize total energy consumption while still meeting a given timing deadline. This continuous assignment problem is impractical in that it ignores the costs and limitations of the power delivery network; we assume that any voltage can be delivered anywhere at no cost. In Section IV we round the voltages to several discrete levels for realistic designs.

A DFG with timing requirements is modeled by a graph $G = (V, E)$. Without loss of generality, we assume that blocks in the DFG are represented by edges rather than vertices; this

is known as the *activity-on-arc* model. The vertices V are I/O ports or "events" representing the completion of a set of (incoming arc) tasks. For each vertex $i \in V$, the labelling t_i represents the arrival time of i. The transformation from DFG to timing graph G is shown in Figure 1.

For each edge $(i,j) \in E$, the variable d_{ij} represents the delay from i to j and is bounded by the constants l_{ij} and u_{ij}. The edge set E is composed of three subsets: $E = E_{blk} \cup E_{int} \cup E_{tim}$. E_{blk} is the set of circuit blocks; for $(i,j) \in E_{blk}$, d_{ij} is the computational delay of the block, where l_{ij} and u_{ij} are determined by the operating voltage range. E_{int} is the set of the interconnect edges; for $(i,j) \in E_{int}$, $d_{ij} = l_{ij} = u_{ij}$ is the constant interconnect delay. E_{tim} is the set of the timing requirement edges; for $(i,j) \in E_{tim}$, we set $l_{ij} = u_{ij} = -R_{ij}$ where the maximal path delay from j to i is required to be no more than the constant R_{ij}. In most cases there is only one edge in E_{tim}, from outputs to inputs, representing the desired computation period of the circuit.

Let $\varepsilon_{ij}(d_{ij})$ be the energy consumption for the edge (i,j). Note that the energy, ε, is expressed as a continuous function of delay rather than of voltage. In our problem formulation we are actually doing *delay assignment* but this is clearly equivalent to voltage assignment. We thus have the following problem for finding the lowest feasible total energy consumption $\mathcal{E}$:

$$\mathcal{E} = \min_{t,d} \sum_{(i,j)\in E} \varepsilon_{ij}(d_{ij}) \tag{1}$$

$$\text{s.t.} \quad t_i + d_{ij} \le t_j, \quad \forall (i,j) \in E \tag{2}$$

$$l_{ij} \le d_{ij} \le u_{ij}, \quad \forall (i,j) \in E \tag{3}$$

We must minimize total energy (1) while enforcing precedence constraints (2) and honoring voltage bounds (3).

B. Minimum convex-cost flow formulation

In this section, we will see how to solve above nonlinear mathematical program efficiently by minimum convex-cost flow. We draw heavily from the work of Ahuja, Hochbaum, and Orlin [28]. Instead, we could have proven the convexity of the problem and entered equations 1, 2 and 3 directly into a general-purpose convex programming solver, as done by Dabiri et al. [1]. However, we will see in Section III that our convex-cost flow code is much faster than the general solvers.

First, we dualize constraint 2 using vector $\vec{x}$ to obtain the following *Lagrangian Subproblem*:

$$L(\vec{x}) = \min_{\vec{t},\vec{d}} \Big\{ \sum_{(i,j)\in E} \varepsilon_{ij}(d_{ij}) + \sum_{(i,j)\in E} (t_i + d_{ij} - t_j) x_{ij} \Big\}$$
$$\text{s.t.} \quad l_{ij} \le d_{ij} \le u_{ij}, \quad \forall (i,j) \in E \tag{4}$$

Since the Lagrangian subproblem is a relaxation of the original problem

$$L(\vec{x}) \le \mathcal{E}, \quad \forall \vec{x} \in \mathbb{R}^+. \tag{5}$$

An optimal solution to the original problem will also be a solution to the Lagrangian subproblem for some $\vec{x}^*$; specifically, it will be the largest such solution by equation 5. Thus, the following *Lagrangian Multiplier Problem* is equivalent to equations 1, 2, and 3.

$$\mathcal{E} = \max_{\vec{x}\in\mathbb{R}^{|E|} \ \wedge \ \vec{x}\ge\vec{0}} L(\vec{x}) \tag{6}$$

Now we coax the above Lagrangian multiplier problem into a convex cost flow problem. Observe that by expanding and recombining the second summation in equation 4 we have

$$L(\vec{x}) = \min_{\substack{\vec{t}\in\mathbb{R}^{|V|} \\ d_{ij}\in[l_{ij},u_{ij}]}} \Big\{ \sum_{(i,j)\in E} \varepsilon_{ij}(d_{ij}) + \sum_{(i,j)\in\ E} d_{ij}x_{ij} + \sum_{(i,j)\in\ E} (t_j - t_i)x_{ij} \Big\} \tag{7}$$

$$= \min_{d_{ij}\in[l_{ij},u_{ij}]} \sum_{(i,j)\in E} \big(\varepsilon_{ij}(d_{ij}) + d_{ij}x_{ij}\big) + \min_{\vec{t}} \sum_{i\in V} \Big(\sum_{(j,i)\in E} x_{ji} - \sum_{(i,j)\in\ E} x_{ij} \Big) t_i \tag{8}$$

$$= \min_{d_{ij}\in[l_{ij},u_{ij}]} \sum_{(i,j)\in E} \big(\varepsilon_{ij}(d_{ij}) + d_{ij}x_{ij}\big) + \min_{\vec{t}} \sum_{i\in V} e_i t_i \tag{9}$$

where

$$e_i \triangleq \sum_{(j,i)\in E} x_{ji} - \sum_{(i,j)\in\ E} x_{ij} \tag{10}$$

We can simplify this formulation as follows. We introduce a new vertex to the DFG, called *node zero*, and add $|V|$ edges $(i,0)$ to it, one from each existing vertex. After extending vectors $\vec{d}$ and $\vec{t}$ by renaming the variables $t_i \to d_{i0}$ and $e_i \to x_{i0}$ and defining $l_{i0} \triangleq -\infty$, $u_{i0} \triangleq \infty$, and $\varepsilon_{i0}(d_{i0}) \triangleq 0$ we get

$$L(\vec{x}) = \min_{d_{ij}\in[l_{ij},u_{ij}]} \sum_{(i,j)\in E'} \big(\varepsilon_{ij}(d_{ij}) + d_{ij}x_{ij}\big) \tag{11}$$

$$\text{s.t.} \qquad x_{i0} = \sum_{(j,i)\in E} x_{ji} - \sum_{(i,j)\in\ E} x_{ij}, \quad \forall i \in V \tag{12}$$

Observe that objective function 11 is separable; we can minimize with each d_{ij} independently. Now equation 6 can be expanded into

$$\mathcal{E} = \max_{\vec{x}\in\mathbb{R}^+} \Big\{ \sum_{(i,j)\in E'} \min_{d_{ij}\in[l_{ij},u_{ij}]} \big\{\varepsilon_{ij}(d_{ij}) + d_{ij}x_{ij}\big\} \Big\} \tag{13}$$

s.t. constraint 12. If we define

$$c_{ij}(\chi) \triangleq -\min_{\vec{d}:d_{ij}\in[l_{ij},u_{ij}]} \big\{\varepsilon_{ij}(d) + d\chi\big\} \tag{14}$$

we get

$$\mathcal{E} = \min_{\vec{x}\in\mathbb{R}^+} \sum_{(j,i)\in E'} c_{ij}(x_{ij}) \quad \text{s.t. constraint 12} \tag{15}$$

Note that, assuming ε_{ij} are convex, $c_{ij}(x_{ij})$ are convex functions which we can evaluate in logarithmic time [28]. We can now see that minimization problem 15 is a minimum

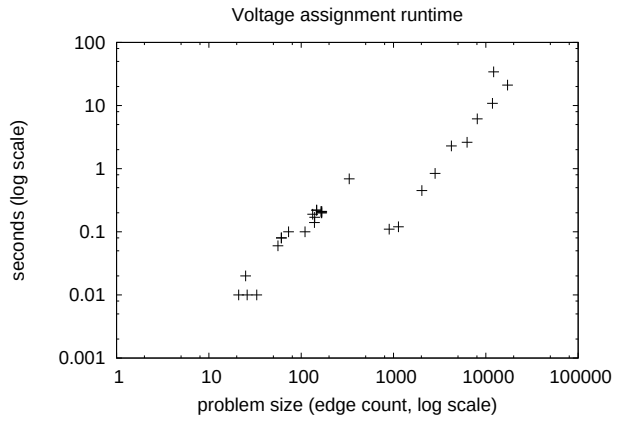

Fig. 2. Runtimes of our flow-based continuous voltage assignment code for problems of various size. Scalability is evident.

convex-cost circulation problem. $\vec{x}$ is the edge-flow vector, $c(x_{ij})$ are the flow cost functions, and constraint 12 enforces mass balance (the flow into a node is equal to the flow out of the node). A circulation problem is simply a flow problem without any flow sources or sinks. None of the edges have any capacity restrictions. Solving this flow problem gives us the Lagrangian multipliers $\vec{x}$ and the optimal energy value $\mathcal{E}$. In addition, if we solve the dual flow problem with a dual method (like the cost-scaling method described in [28]) we get the dual-dual variables as well; that is, the primal variables $\vec{d}$ and $\vec{t}$, The voltage of each block is simply a function of the assigned delay: $v_{ij}(d_{ij})$.

Note that we have made no assumptions about the energy functions ε_{ij} other than convexity. Convexity, in this case, means that continually decreasing a circuit block's delay by Δd creates increasingly higher energy consumption changes $\Delta\varepsilon$. Formally, the derivative of energy with respect to delay is nondecreasing.

Time complexity: Using the cost-scaling algorithm described by Ahuja et al. [28], the above minimum convex-cost flow problem can be solved in $O(|V||E|b\log(|V|b))$ time where V and E are the vertex and edge sets from flow graph (or equivalently the DFG) and b is the bit-precision. Our runtime complexity has an extra factor of b compared to the basic cost-scaling convex cost flow algorithm to account for the time needed to evaluate the edge cost function $c_{ij}(\chi)$ by binary search.

III. Experiment 1: continuous voltage assignment

We implemented a cost-scaling minimum convex-cost flow code in C++ following the algorithm of Ahuja et al. [28]. Our code solves the *integer* flow problem. However, the Lagrangian multipliers $\vec{x}$ are fractional, so we adjusted the edge cost functions to get an fixpoint fractional flow solution.

Figure 2 and Table I show the efficiency and scalability of our continuous voltage assignment algorithm. Dabiri et al.solve their convex program using Matlab and MOSEK with a runtime of about two minutes for their largest problem (84 nodes and 140 edges) [30]. We solve problems of the same size in under 0.5 seconds, so we conservatively (allowing for possible differences in machine speed) report a $100\times$ speedup. In fact, we expect that a better-optimized flow code can perform even faster.

TABLE I
Continuous voltage assignment results

benchmark	nodes	edges	runtime (s)
jj1	4	7	0.00
ex1	6	13	0.00
paulin	8	21	0.01
diffeq_new	10	26	0.01
mac	10	25	0.02
tseng	13	33	0.01
chemical	20	56	0.06
ewf-hyper	22	61	0.08
elliptic	22	61	0.08
iir77	25	73	0.10
dct_dif	40	110	0.10
wdf	48	133	0.19
dct_lee	50	139	0.14
jacobi_sm	51	147	0.22
pr1	51	140	0.17
pr2	60	165	0.20
dct_wang	60	164	0.21
dct_ijpeg	61	167	0.20
dct	61	167	0.21
jacobi	115	331	0.69
c432	359	897	0.11
c499	448	1130	0.12
c880	829	2026	0.45
c1355	1136	2818	0.84
c1908	1796	4231	2.29
c2670	2622	6263	2.62
c3540	3391	8070	6.16
c5315	4795	11788	10.84
c6288	4867	12146	34.15
c7552	7234	17204	21.11

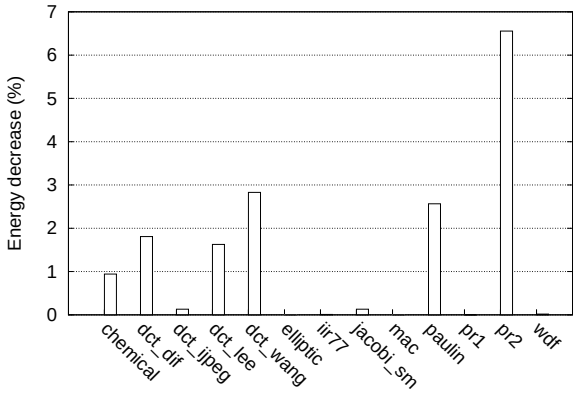

Fig. 3. Energy savings of our optimal continuous voltage assignment relative to critical path method [24].

Figure 3 shows the improvement of our optimal continuous voltage assignment algorithm compared to the heuristic path-based approach of Gu et al. [24]. The energy decrease we achieve peaks at 6.5% and comes at no cost.

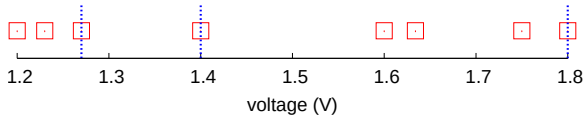

Fig. 4. Partitioning-based voltage discretization. All blocks in a partition are rounded up to the voltage at the right-hand side cutpoint, represented by a dashed line.

IV. Discrete voltage set selection

In practical multiple supply voltage designs, only a handful of voltages can be made available by the power distribution network. Each requires a separate off-chip voltage regulator and complicates power network design. One reasonable way to choose a set of discrete voltages is to partition the sorted voltage assignments and set each partition's voltage to its upper bound. This method is shown in Figure 4. In this case, we never decrease any voltages from the continuous voltage assignment, so we can be sure that the timing requirements are still met. This partitioning discretization method was introduced by Gu et al. [24] and also used by Dabiri et al. [1]; however, we contribute an algorithm which is both optimal and elegant. Gu et al.'s algorithm is optimal but very complex and Dabiri et al.'s algorithm is simple but heuristic. Note that we optimally solve the *partitioning* problem, which gives a heuristic solution to the more general *discretization* problem.

A. Dynamic programming (DP) algorithm

We can consider the *cost* of a partitioning to be the energy penalty incurred by increasing continuous voltages up to the discrete level chosen. This cost can be expressed with the recurrence

$$C[i,j,v] = \min_{k \in [i,j)} \left\{ C[i,k,1] + C[k+1,j,v-1] \right\} \qquad (16)$$

where $C[i,j,v]$ is the minimum cost of partitioning the sequence from positions i to j using v voltages. The integer variable k indicates the optimum endpoint for the first partition. Translating recurrence 16 into a dynamic programming algorithm is straightforward.

Gu *et at.*'s algorithm [24] has a runtime complexity of $O(n^2 2^v)$ while our DP algorithm is $O(n^3 v)$, where v is the number of voltage levels. Although exponential in the number of voltage levels, we expect their solution to be faster for practical designs with few voltage levels.

V. Experiment 2: discretization overhead

There is no known optimal polynomial-time algorithm for the discrete voltage set selection and assignment problem; the algorithm we present is suboptimal because timing slack is not redistributed after rounding. For example, in Figure 4, after rounding the first two blocks up to 1.27 V the rightmost block may only require 1.75 V to meet the timing deadline. However, we can use the continuous voltage assignment's energy as a lower-bound on the optimal discrete solution energy and thus measure the maximum error of our partitioning heuristic. Our experiments show that our DP discrete voltage selection and assignment solutions are near even this somewhat loose lower bound.

TABLE II
Voltage discretization overhead versus optimal continuous voltage assignment, in percent.

benchmark	single voltage	2 voltage	3 voltage	4 voltage
tseng	62.10%	1.10%	0.00%	0.00%
paulin	76.33	11.74	5.57	1.24
diffeq_new	76.94	13.74	2.77	1.15
mac	67.01	0.12	0.03	0.01
dct_dif	101.60	8.37	0.13	0.03
wdf	77.65	12.36	4.68	1.62
ewf-hyper	98.37	15.29	0.27	0.06
elliptic	98.37	15.29	0.27	0.06
pr1	105.30	6.10	1.74	0.07
chemical	117.56	11.25	0.11	0.03
dct_lee	97.45	8.13	2.12	0.02
pr2	68.40	22.85	7.42	4.40
dct_wang	82.91	20.22	5.12	1.73
dct_ijpeg	96.28	12.84	0.89	0.06
dct	96.28	12.84	0.89	0.06
iir77	118.84	8.60	0.10	0.05
jacobi_sm	95.17	0.04	0.02	0.01
average	**90.38%**	**10.64%**	**1.89%**	**0.62%**

Table II gives the energy overhead of our discrete voltage selection and assignment algorithm compared to the baseline continuous voltage assignment. Note that we are comparing to an optimal solution that uses up to n different voltages. In reality, the optimal 2, 3, or 4 voltage energy will be higher, so the results of Table II are an upper-bound on our true overhead. For two-voltage assignment, worst-case energy overhead is bounded by 23%, for three-voltage assignment by 8%, and for four-voltage assignment by 5%. Average-case overhead is lower still.

Similar experiments have been done by Dabiri et al. [1] and Hua and Qu [4]. Their results also indicate that a few discrete voltage levels can provide almost the same energy savings as an infinite range of voltages. Due to the closeness of our discretization energies and the theoretical lower bound, we omit a comparison of competing discretization approaches.

The single-voltage column in Table II indicates the energy consumption overhead of the circuit before applying the Multiple Supply Voltage technique. We observe, as others have in the past, that the energy gains acheived by moving from a single to two or three voltages are significant.

VI. Experimental setup

The benchmark problems we used are real circuits used as high-level synthesis benchmarks by Gu et al. [24]; we adopt their nonlinear power-delay model using $\alpha = 1.5$. No interconnect delay was included, although this would have been possible, as described in Section II. We allow voltages to vary from 1.8 V down to 1.2 V. Under our device model this yields a maximum block slowdown of 22% corresponding to an energy reduction of 56%. The deadlines were halfway between the fastest and slowest possible execution time.

To evaluate the runtime on large problem instances we also used the ISCAS85 combinational logic benchmarks. In these experiments all gates were assumed to be of the same type. All

experiments were run on an Intel Core2 duo E6300 1.86 GHz Linux workstation.

VII. Conclusions and future work

Our fast flow-based voltage assignment algorithm provides the same impressive energy savings as previous work but much more efficiently. This enables accurate energy cost evaluations to be made in the inner loop of design flows which solve complex CAD problems [24].

One of the strengths of our work is its potential for extension and adaptation; the edge cost functions can be replaced with any convex function. In addition, our flow problem derivation can be followed to solve silimar convex budgeting problems.

References

[1] F. Dabiri, R. Jafari, A. Nahapetian, and M. Sarrafzadeh, "A unified optimal votage selection methodology for low-power systems," in *Proc. Int. Symp. Quality of Electronic Design*, 2007.

[2] Q. Ma and E. Young, "Network flow-based power optimization under timing constraints in msv-driven floorplanning," in *Proc. Int. Conf. Computer-Aided Design*, 2008, pp. 1–8.

[3] H.-Y. Liu, W.-P. Lee, and Y.-W. Chang, "A provably good approximation algorithm for power optimization using multiple supply voltages," in *Proc. Design Automation Conf.*, 2007, pp. 887–890.

[4] S. Hua and G. Qu, "Approaching the maximum energy savings on embedded systems with multiple voltages," in *Proc. Int. Conf. Computer-Aided Design*, 2003, pp. 26–29.

[5] W.-P. Lee, H.-Y. Liu, and Y.-W. Chang, "An ILP algorithm for post-floorplanning voltage-island generation considering power-netowrk planning," in *Proc. Int. Conf. Computer-Aided Design*, 2007, pp. 650–655.

[6] W.-K. Mak and J.-W. Chen, "Voltage island generation under performance requirement for soc designs," in *Proc. Asia & South Pacific Design Automation Conf.*, 2007, pp. 798–803.

[7] L. Guo, Y. Cai, Q. Zhou, and X. Hong, "Logic and layout aware voltage island generation for low power design," in *Proc. Asia & South Pacific Design Automation Conf.*, 2007, pp. 666–671.

[8] 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. Asia & South Pacific Design Automation Conf.*, 2006, pp. 582–587.

[9] H. Wu, I.-M. Liu, M. D. F. Wong, and Y. Wang, "Post-placement voltage island generation under performance requirement," in *Proc. Int. Conf. Computer-Aided Design*, 2005, pp. 309–316.

[10] H. Wu, M. D. F. Wong, and I.-M. Liu, "Timing-constrained and voltage-island-aware voltage assignment," in *Proc. Design Automation Conf.*, 2006, pp. 429–432.

[11] Q. Ma and E. F. Y. Young, "Voltage island-driven floorplanning," in *Proc. Int. Conf. Computer-Aided Design*, 2007, pp. 644–649.

[12] W.-P. Lee, H.-Y. Liu, and Y.-W. Chang, "Voltage island aware floorplanning for power and timing optimization," in *Proc. Int. Conf. Computer-Aided Design*, 2006, pp. 389–394.

[13] K. Usami and M. Horowitz, "Clustered voltage scaling technique for low-power design," in *Proc. Int. Symp. Low Power Electronics & Design*, 1995, pp. 3–8.

[14] R. Puri, L. Stok, J. Cohn, D. Kung, D. Pan, D. Sylvester, A. Srivastava, and S. Kulkarni, "Pushing ASIC performance in a power envelope," in *Proc. Design Automation Conf.*, 2003, pp. 788–793.

[15] R. Puri, L. Stok, and S. Bhattacharya, "Keeping hot chips cool," in *Proc. Design Automation Conf.*, 2005, pp. 285–288.

[16] J.-M. Chang and M. Pedram, "Energy minimization using multiple supply voltages," *IEEE Trans. VLSI Systems*, vol. 5, no. 4, pp. 436–443, Dec 1997.

[17] C. Chen, A. Srivastava, and M. Sarrafzadeh, "On gate level power optimization using dual-supply voltages," *IEEE Trans. VLSI Systems*, vol. 9, no. 5, pp. 616–629, 2001.

[18] U. Y. Ogras, R. Marculescu, P. Choudhary, and D. Marculescu, "Voltage-frequency idland partitioning for GALS-based newtorks-on-chip," in *Proc. Design Automation Conf.*, 2007, pp. 110–115.

[19] Y. Cho, N. Chang, C. Chakrabarti, and S. Vrudhula, "High-level power management of embedded systems with application-specific energy cost functions," in *Proc. Design Automation Conf.*, 1996, pp. 568–573.

[20] R. Jejurikar and R. Gupta, "Dynamic voltage scaling for systemwide energy minimization in real-time embedded systems," in *Proc. Int. Symp. Low Power Electronics & Design*, 2004, pp. 76–81.

[21] S. Ghiasi, E. Bozorgzadeh, P.-K. Huang, R. Jafari, and M. Sarrafzadeh, "A unified theory of timing budget management," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 11, pp. 2364–2375, Nov. 2006.

[22] E. Boros, P. L. Hammer, and R. Shamir, "A polynomial algorithm for balancing acyclic data flow graphs," *IEEE Trans. Computers*, vol. 41, no. 11, pp. 1380–1385, 1992.

[23] R. Nair, C. Berman, P. Hauge, and E. Yoffa, "Generation of performance constraints for layout," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 8, no. 8, pp. 860–874, Aug 1989.

[24] Z. P. Gu, Y. Yang, J. Wang, R. P. Dick, and L. Shang, "TAPHS: thermal-aware unified physical-level and high-level synthesis," in *Proc. Asia & South Pacific Design Automation Conf.*, 2006, pp. 879–885.

[25] C. Lin, A. Xie, and H. Zhou, "Design closure driven delay relaxation based on convex cost network flow," in *Proc. Design, Automation & Test in Europe Conf.*, 2007, pp. 1–6.

[26] D. R. Fulkerson, "A network flow computation for project cost curves," *Management Science*, vol. 7, no. 2, pp. 167–178, Jan. 1961.

[27] R. K. Ahuja, T. L. Magnanti, and J. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

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

[29] K. Niyogi and D. Marculescu, "Speed and voltage selection for GALS systems based on voltage/frequency islands," in *Proc. Asia & South Pacific Design Automation Conf.*, 2005.

[30] F. Dabiri, personal communication, Jul. 2008.