AN IMPROVED LAGRANGIAN RELAXATION METHOD FOR VLSI COMBINATIONAL CIRCUIT OPTIMIZATION

A Thesis

by

YI-LE HUANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2010

Major Subject: Electrical Engineering

AN IMPROVED LAGRANGIAN RELAXATION METHOD FOR VLSI COMBINATIONAL CIRCUIT OPTIMIZATION

A Thesis

by

YI-LE HUANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Co-Chairs of Committee,	Jiang Hu
	Weiping Shi
Committee Members,	Duncan M. Walker
Head of Department,	Costas N. Georghiades

December 2010

Major Subject: Electrical Engineering

ABSTRACT

An Improved Lagrangian Relaxation Method for VLSI Combinational Circuit Optimization. (December 2010)

Yi-Le Huang, B.S., National Tsing Hua University;
M.S., National Tsing Hua University
Co-Chairs of Advisory Committee: Dr. Jiang Hu
Dr. Weiping Shi

Gate sizing and threshold voltage (Vt) assignment are very popular and useful techniques in current very large scale integration (VLSI) design flow for timing and power optimization. Lagrangian relaxation (LR) is a common method for handling multi-objectives and proven to reach optimal solution under continuous solution space. However, it is more complex to use Lagrangian relaxation under discrete solution space. The Lagrangian dual problem is non-convex and previously a sub-gradient method was used to solve it. The sub-gradient method is a greedy approach for substituting gradient method in the deepest descent method, and has room for further improvement. In addition, Lagrangian sub-problem cannot be solved directly by mathematical approaches under discrete solution space. Here we propose a new Lagrangian relaxation-based method for simultaneous gate sizing and Vt assignment under discrete solution space. In this work, some new approaches are provided to solve the Lagrangian dual problem considering not only slack but also the relationship between Lagrangian multipliers and circuit timing. We want to solve the Lagrangian dual problem more precisely than did previous methods, such as the sub-gradient method. In addition, a table-lookup method is provided to replace mathematical approaches for solving the Lagrangian sub-problem under discrete size and Vt options. The experimental results show that our method can lead to about 50% and 58% power reduction subject to the same timing constraints compared with a Lagrangian relaxation method using sub-gradient method and a state-of-the-art previous work. These two methods are implemented by us for comparison. Our method also results in better circuit timing subject to tight timing constraints.

TABLE OF CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

CHAPTER I

INTRODUCTION

In deep sub-micron technologies, minimization of leakage power becomes the dominant concern as we try to combat the increase in the overall circuit power consumption. In addition, power consumption also directly relates to battery life, reliability, packaging and heat removal cost. Therefore, how to efficiently handle trade-off between circuit performance and power consumption becomes a big issue in current design flow.

In the past, due to chip area cost issue, people focused on area/timing optimization to minimize total chip area subject to circuit timing constraints. Gate sizing [1] is one of the most popular methods people used to perform area/timing, or area/timing/power, optimization for circuit designs. In recent years, leakage power becomes more and more important due to battery life, reliability, packaging and heat removal cost. People start to put attention on leakage power and try to optimize combinational circuit for leakage power reduction by using different threshold voltage (Vt) levels [2-4]. A gate with a higher Vt level will decrease performance but reduce power. Oppositely, a gate with a lower Vt level will result in better performance but more leakage power. Therefore, we can use gates with higher Vt level in non-critical paths to reduce leakage power and then keep gates in critical paths lower Vt level for

This thesis follows the style of *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.*

retaining desired performance. It is obvious that there are many similarities between traditional gate sizing and Vt assignment, and hence they can be easily combined with each other for combinational circuit optimization [5-10].

Most of simultaneous gate sizing and Vt assignment methods are either sensitivity-based heuristics [2,8] or mathematical programming methods [3,9,11]. In the sensitivity-based heuristics, people use their own sensitivity function to evaluate gates in the circuit and choose gate size and Vt level according to function value. Usually, the sensitivity function only considers about some local information, like the efficiency of trading power for performance for single gate. Hence, the timing critical gate which can speed up itself more and cost less power than others is the best candidate for bigger sizes or lower Vt levels. Unfortunately, the sensitivity function only takes local information into consideration, only caring about effect for current gate not whole circuit. The greedy nature makes sensitivity-based methods easily to fall into local optimal.

In [1], continues gate/transistor sizing is formulated as geometric programming and [11,12] solves it by Lagrangian relaxation. In [12], constraints are defined on circuit components rather than circuit paths. Therefore, the number of timing constraints is only linearly proportional to the number of circuit components rather than an exponential number of circuit paths. Then, the Lagrangian relaxation method solves the constrained optimization problem by relaxing constraints into objective function with Lagrangian multipliers and the constraint-reduced problem can be solved easier than original one. By iteratively changing Lagrangian multipliers with decreasing step size, this work is proved to converge and guarantee optimality. Therefore, the mathematical method becomes one of the main approaches to solve circuit optimization problems.

Nowadays, circuits are implemented by gates in standard library provide by foundries. Sizes and threshold voltage options of logic gates are limited and discrete specified in the standard library. Usually people use traditional continuous optimization methods for circuit optimization at first and then solutions are rounded to the nearest feasible options. Existing rounding continuous optimization methods [13,14] result in remarkable rounding errors and the errors can be significant if options in standard library are highly discrete [15].

The continuous gate sizing method by Lagrangian relaxation [12] gives us a good starting point for simultaneous gate sizing and Vt assignment. Even though the Lagrangian relaxation method is proved to converge and guarantee optimality, there are still many practical implementation details which weaken this work [16] due to the nature of greedy approaches when solving Lagrangian dual problem. For instance, different step size method and initial Lagrangian multipliers affect solution quality and make run time much longer.

Besides, the shape of Lagrangian dual problem under discrete solution space becomes sharper and non-convex, and the greedy nature makes sub-gradient method used in [12] struggle when applied on discrete solution space. Sub-gradient method tends to oscillate and those oscillating iterations have no benefit for approaching the optimal solution of Lagrangian dual problem. Therefore, sub-gradient method using in continuous solution space is not powerful enough to guide Lagrangian relaxation and guarantee optimality under discrete solution space anymore.

In this work, we consider simultaneous discrete gate sizing and Vt assignment for general very large scale integrated (VLSI) circuits by Lagrangian relaxation. Timing constraints reside in circuit components and objective function is to minimize total circuit power consumption. Due to the weakness of sub-gradient method under discrete solution space, we propose some new approaches for solving Lagrangian dual problem. We also solve the Lagrangian sub-problem directly under discrete solution space by a table lookup method without rounding errors. Compared with those sensitivity based heuristics, our method using Lagrangian relaxation is more systematic and therefore leads into better solution quality. Our new approach for solving Lagrangian dual problem distinguishes our method from those previous Lagrangian relaxation based methods using sub-gradient method. Experimental results show that our dual problem approach gives Lagrangian relaxation better direction toward optimal solution and results in faster convergence than sub-gradient method.

The rest of this thesis is organized as follows. In Chapter II, we introduce some notations and terminology that we use in this paper. In Chapter III, we write down the detailed problem formulation of this work. In Chapter IV, we briefly present how Lagrangian relaxation solves constrained optimization problem. In Chapter V, we show how to improve Lagrangian relaxation with our algorithm for dual problem under discrete sizes and Vt levels and our algorithm for sub-problem is in Chapter VI. In Chapter VII, we show experimental results compared with the dual problem method in

[12] and sub-problem method in [17] under two different timing requirements. The convergence of our method is demonstrated by results of an ISCAS85 benchmark.

CHAPTER II

PRELIMINARIES

Given a combinational logic circuit, usually it can be described by a directed acyclic graph (DAG) $G(V, E)$, where V is a set of nodes representing circuit components, including logic gates X, input drivers S and output loads T, $V = X \cup S \cup T$, and E is a set of edges indicating the wire connection between components. Each edge $e_{ij} \in E$ indicates the connection between v_i and $v_j \in V$ and logic signal propagates from v_i to v_j. Each logic gate $v_i \in V$ has a size W_i and a Vt level U_i, total $|W_i| \times |U_i|$ possible options. The simultaneous gate sizing and Vt assignment problem is to assign $w_i \in W_i$ and $u_i \in U_i$ for all $v_i \in X$ such that the total power consumption is minimized subject to timing constraints. For example, for the circuit in Figure 1, v_i, v_j and v_k are logic gates and e_{ij} and e_{ik} are edges between v_i and v_j, and v_i and v_k, respectively. The gates v_j and v_k are called the fan-in gates of v_i, $fanin(v_i)$. On the other hand, v_i is called the fan-out gates of v_j and v_k, $fanout(v_j)$.

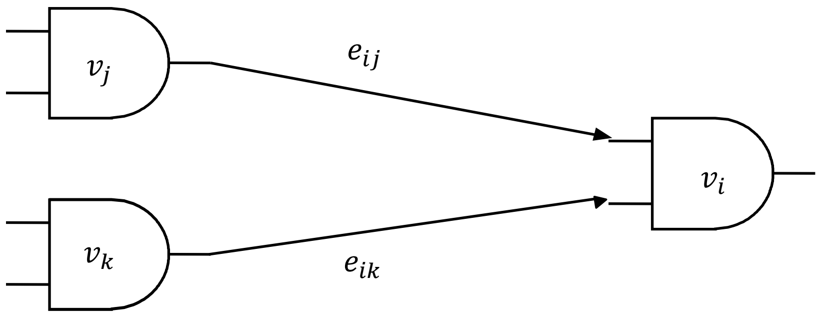


Figure 1. A circuit example with three logic gates

Here we only consider two kinds of power consumption, dynamic and leakage power. Dynamic power $\mathrm{P_{dyn}}$ consumption is proportional to switching factor β, clock frequency $\mathrm{f_{clk}}$, load capacitance $\mathrm{C_{load}}$ and square of supply voltage level V_{dd}. Detailed equation of dynamic power is $P_{dyn} = (\beta V_{dd}^2 f_{clk} C_{load})/2$. On the other hand, leakage power $P_{leakage}$ consumption is related to supply voltage and off current I_{off} of gates, where I_{off} is given by cell library changed with size and Vt level. Detailed equation of leakage power consumption is $P_{leak} = V_{dd} \times I_{off}$. There are still some other types of power consumption, like short circuit power, and they are relatively small so we ignore them in this work. However, they can be taken into consideration easily when they are significant.

In this work, we take Elmore delay model as our delay model and model circuit components as resistance-capacitance (RC) circuits [18]. A logic gate $v_i \in X$ is modeled as input capacitance c_i and output resistance r_i plus intrinsic delay D_{int_i}, as shown in Figure 2. A wire segment is modeled as a π-type RC circuit. The wire model for a wire $e_{ij} \in E$ is shown in Figure 3 where $l_{e_{ij}}$ is length of e_{ij} and r_{unit} and c_{unit} are the unit

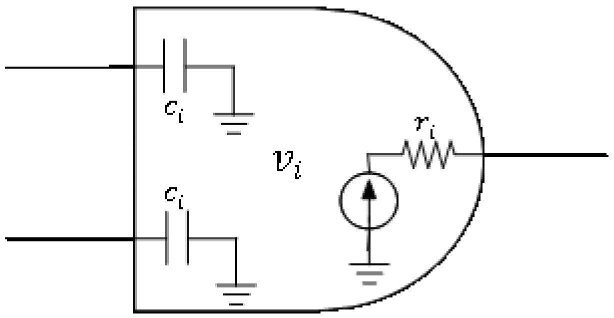


Figure 2. The RC model for a logic gate

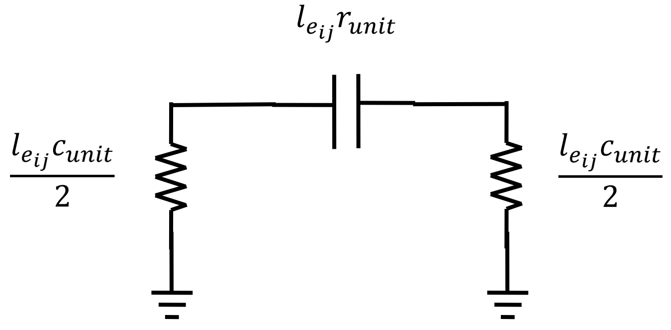

$l_{e_{ij}} r_{unit}$
$\frac{l_{e_{ij}} c_{unit}}{2}$
$\frac{l_{e_{ij}} c_{unit}}{2}$

Figure 3. The RC model for a wire segment

length wire resistance and capacitance, respectively. The delay associated with a resistor is calculated by its resistance times its downstream capacitance. The delay for a path is the sum of the delay on resistors which it passes through. The Elmore delay model is relatively easy and proven to be applicable to distributed network of resistors and capacitors. However, our work can also deal with more complex delay model easily. Here we assume that the arrival time a at input resistors and required arrival time q at output capacitors are given. Then, arrival time for $v_j \in V$ is calculated by $a_j = max_{v_i \in S}(a_i + max_{p_{ij} \in paths\ from\ v_i\ to\ v_j} D(p_{ij}))$ and required arrival time is calculated by $q_j = min_{v_i \in T}(q_i - max_{p_{ij} \in paths\ from\ v_i\ to\ v_j} D(p_{ij}))$. The timing information can be obtained easily by static timing analysis [19]. The arrival time is propagated in topological order and oppositely required arrival time is obtained in reversed topological order. In Figure 4, arrival time of gate 3, a_3, is propagated from gate 1 and 2, and the detailed equation is that $\mathrm{a_3 = max(a_1, a_2) + D_3}$ where D_3 is delay of gate 3. Required arrival time of gate 3, q_3, is propagated from gate 4 and 5, and the detailed equation is that $\mathrm{q_3 = min(q_4 - D_4, q_5 - D_5)}$ where D_4 and D_5 are delay of gate 4 and 5,

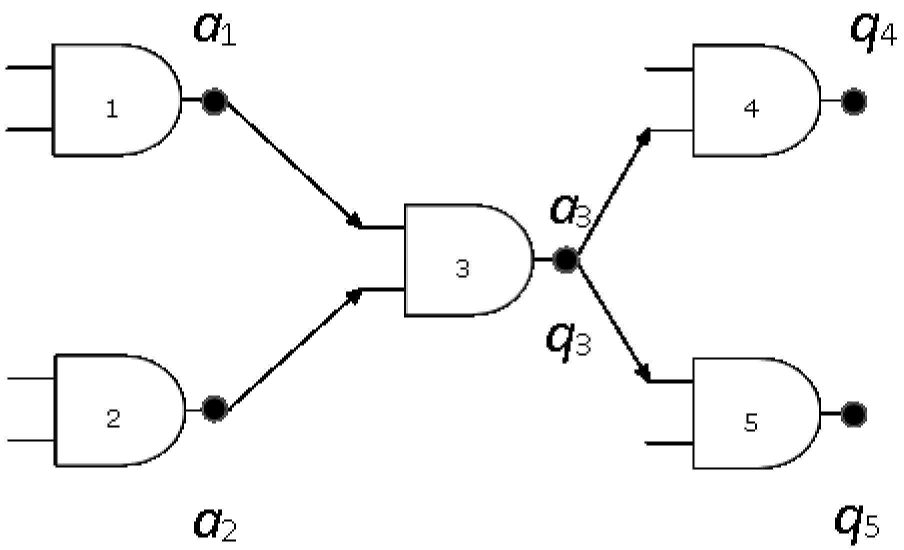


Figure 4. An example of static timing analysis

respectively. Then, slack is used to indicate the timing criticality of components given by $q - a$. If the slack of a component is negative, it means that the timing constraint on this component is violated. The overall circuit timing is characterized by the minimum slack among components.

The size of a gate $v_i \in X$ affects some of its parasitic elements, including intrinsic delay, input capacitance, output resistance, dynamic and leakage power consumption. The Vt level of a gate also changes some of its parasitic elements, like intrinsic delay and dynamic power consumption. We use $r_i^{w_i u_i}$to represent output resistance of $v_i \in X$ under $w_i \in W_i$ and $u_i \in U_i$ and other parasitic elements also symbolized by the same rule. All the parasitic information of logic gates is defined in a cell library. A gate with a bigger size results in smaller input capacitance, bigger output resistance, bigger intrinsic delay and higher power consumption. A gate with a higher Vt level will be converted to a gate model with bigger output resistance and lower power consumption.

CHAPTER III

PROBLEM FORMULATION

Given a combinational circuit, we want to solve the problem of minimizing the total circuit power consumption with respect to gate size and Vt level subject to the timing constraints, no negative slack in the circuit. We formulate the problem as a constrained optimization problem with a polynomial number of constraints. These constraints are formulated with arrival time a and only applied on components rather than paths to reduce the number of constraints. We call the constrained optimization problem primal problem, pp.

$$pp : \; Minimize \sum_{v_i \in X} \alpha_i p_i$$

$$a_j \le A_j \qquad v_j \in T$$

$$a_j + D_i \le a_i \qquad \forall v_j \in fanin(v_i), \qquad v_i \in X$$

$$D_i \le a_i \qquad v_j \in S$$

$$w_i \in W_i, u_i \in U_i \qquad v_i \in X$$

A_i, p_i, $\alpha_i$and D_i are given required arrival time, power consumption, weighting factor and gate delay of v_i, respectively.

CHAPTER IV

LAGRANGIAN RELAXATION

Lagrangian relaxation method is a well-known approach for solving constrained optimization problem. In [12], a timing constrained area optimization problem is solved by Lagrangian relaxation under continuous gate sizes. In Lagrangian relaxation method, constraints are relaxed and incorporated into the objective function by multiplying with Lagrangian multipliers $\vec{\lambda}$. A Lagrangian multiplier λ is a non-negative value for each constraint. A Lagrangian multiplier λ_{ij} is associated on the arrival time constraint of the wire connection from v_j to v_i, where v_j is a fan-in gate of v_i. Then the new objective function becomes

$$L_\lambda = \sum_{v_i \in X} \alpha_i p_i + \sum_{\substack{v_j \in fanin(v_i) \\ v_i \in T}} \lambda_{ij} \left(a_j - A_i\right) + \sum_{\substack{v_j \in fanin(v_i) \\ v_i \in X}} \lambda_{ij} \left(a_j + D_i - a_i\right)$$

$$+ \sum_{v_i \in S} \lambda_{i0} \left(D_i - a_i\right)$$

For a given vector $\vec{\lambda}$, we have a new optimization problem only with size and Vt constraints and the constraint-reduced optimization problem only with size and Vt level constraints is called Lagrangian sub-problem LRS/λ.

$$LRS/\lambda : \; Minimize\ L_\lambda$$

$$subject\ to\ w_i \in W_i, u_i \in U_i, \forall v_i \in X$$

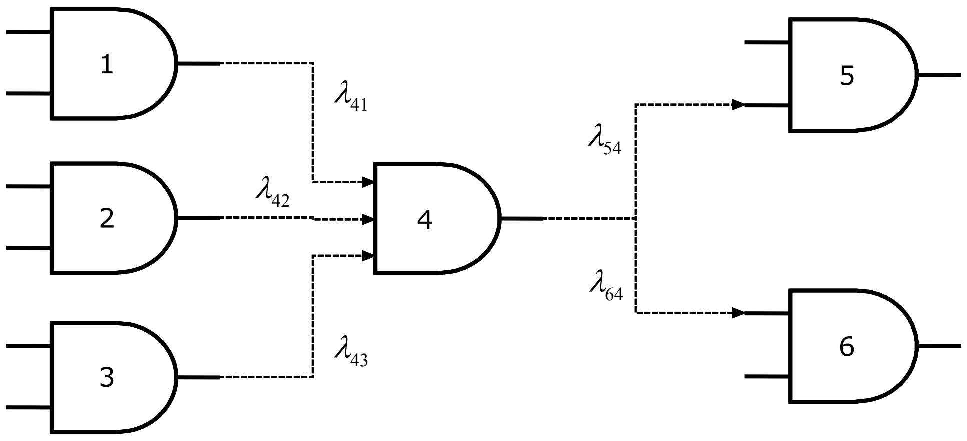


Figure 5. An example of KKT condition

It can be shown that there exists a vector of Lagrangian multipliers $\overrightarrow{\lambda_{\mathrm{opt}}}$ such that the optimal solution of Lagrangian sub-problem is also the optimal solution of original constrained optimization problem. How to find the $\overrightarrow{\lambda_{\mathrm{opt}}}$ is called Lagrangian dual problem LDP. To reduce the complexity of L_λ, we apply Kuhn-Tucker (KKT) conditions to it, requiring $\partial L_\lambda / \partial a_i = 0$ at the optimal solution for $v_i \in X$. Applying $\partial L_\lambda / \partial a_i = 0$, we can get the optimal conditions for $\vec{\lambda}$,

$$\sum_{v_k \in \mathrm{fanout}(v_i)} \lambda_{ki} = \sum_{v_j \in \mathrm{fanin}(v_i)} \lambda_{ij} \quad , \forall v_i \in V - T$$

The optimal conditions say that for all logic gates and input resistors, the sum of Lagrangian multipliers on the wire connected from its fan-in gates must be equal to the sum of Lagrangian multipliers on the wire connected to its fan-out gates. We use Figure 5 to illustrate the KKT conditions where $\lambda_{41} + \lambda_{42} + \lambda_{43}$ must be equal to $\lambda_{64} + \lambda_{54}$. Using the result of KKT conditions, the problem can be further reduced to

$$L_\lambda = \sum_{v_i \in X} \alpha_i p_i + \sum_{v_i \in V-T} \Big(\sum_{v_j \in fanin(v_i)} \lambda_{ij} \Big) \; D_i$$

$$subject\ to\ w_i \in W_i\ u_i \in U_i, \forall v_i \in X$$

Replacing $\sum_{\mathrm{v_j \in fanin(v_i)}} \lambda_{\mathrm{ij}}$ by μ_{i} for $\forall \mathrm{v_i} \in \mathrm{V} - \mathrm{T_i}$, the problem can be written as

$$L_\lambda = \sum_{v_i \in X} \alpha_i p_i + \sum_{v_i \in V-T} \mu_i D_i$$

$$subject\ to\ w_i \in W_i\ u_i \in U_i, \forall v_i \in X$$

We can see that there is only power and component delay left in L_λ without arrival time. The part of L_λ affected by a component is independent from that affected by others. Therefore, the Lagrangian sub-problem can be solved much easier than the original objective function with arrival time.

In [12], for a given $\overrightarrow{\lambda_e}$, the Lagrangian sub-problem LRS/λ can be solved optimally by a greedy algorithm when the gate sizes are continuous. The algorithm iteratively performs local optimal sizing for a gate while the other gates are fixed. For $v_i \in X$, the local optimal size is $\sqrt{\mu_i r_i C_i / (R_i c_i + \alpha_i)}$, where $R_i = \sum_{v_j \in fanin(v_i)} \mu_j r_j$ and C_i is the downstream capacitance of v_i. For the convenience of presentation, we ignore gate size constraints in this equation.

The problem of finding the optimal set of Lagrangian multipliers $\overrightarrow{\lambda_{\mathrm{opt}}}$ is called Lagrangian dual problem LDP.

$$LDP: Maximize\ Q(\vec{\lambda})$$

$$where\ Q(\vec{\lambda}) = inf\left(\sum_{v_i \in X} \alpha_i p_i + \sum_{v_i \in V-T} \Big(\sum_{v_j \in fanin(v_i)} \lambda_{ij}\Big)\ D_i\right): w_i \in W_i\ u_i \in U_i, \forall v_i \in X$$

In LDP, we try to find a vector $\vec{\lambda}$ such that $Q(\vec{\lambda})$ is the maximum.

In general, $Q(\lambda)$ is not differentiable, so the gradient direction used in steepest descent method does not work and therefore is replaced by sub-gradient direction here. For each arrival time, its sub-gradient is defined as the arrival time constraint and then evaluated with current situation. The sub-gradient direction $\Delta\vec{\lambda}$ is the vector of all the sub-gradients. Next, the sub-gradient direction $\Delta\vec{\lambda}$ is multiplied by a step size ρ. Then, the next point for Lagrangian multipliers $\vec{\lambda}$ is obtained by current point plus sub-gradient direction. Given that the step size satisfies the following conditions: $\lim_{n\to\infty} \rho_n = 0$ and $\sum_{n=1}^{\infty} \rho_n = \infty$, the sub-gradient method will converge to the optimum under continuous solution space. The equations for updating Lagrangian multipliers associated with a gate by corresponding sub-gradients are shown as followed.

$$for\ all\ v_j \in fanin(v_i), \qquad \lambda_{ij} = \begin{cases} \lambda_{ij} + \rho_n(a_j - A_i), & v_i \in T \\ \lambda_{ij} + \rho_n(a_j + D_i - a_i), & v_i \in X \\ \lambda_{ij} + \rho_n(D_i - a_i), & v_i \in S \end{cases}$$

CHAPTER V

IMPROVED ALGORITHM FOR SOLVING *LDP*

As we mentioned before, Lagrangian relaxation methods are easily weakened by the practical implementation details. The sub-gradient direction does not always guide Lagrangian relaxation method to the optimal solution for discrete cases. Here we want to find a better way to solve LDP to make Lagrangian relaxation method to converge faster with better solution quality under discrete solution space.

The importance of KKT conditions

Here we want to spend some time to look into KKT conditions. In Lagrangian relaxation method, the Lagrangian multipliers work like weight of constraints. If a constraint is violated, its Lagrangian multiplier will be increased according to its slack. Then, gates with bigger μ are allowed trading more power for timing by using bigger gate size or lower Vt level. Unfortunately, it does not make sense to choose gate size and Vt level only considering about local information of a gate. If that, Lagrangian relaxation

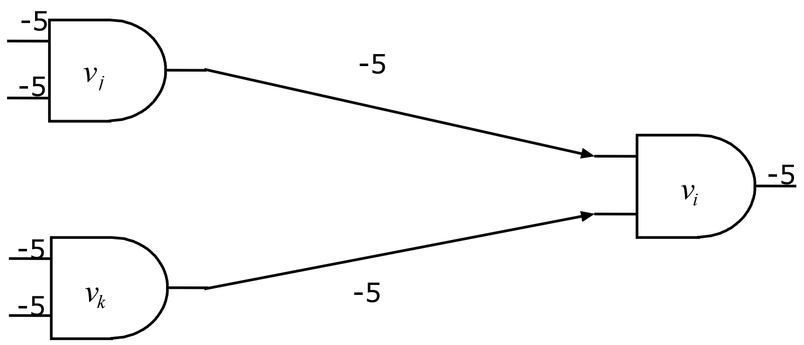


Figure 6. An example of circuit with -5 slack on all the components

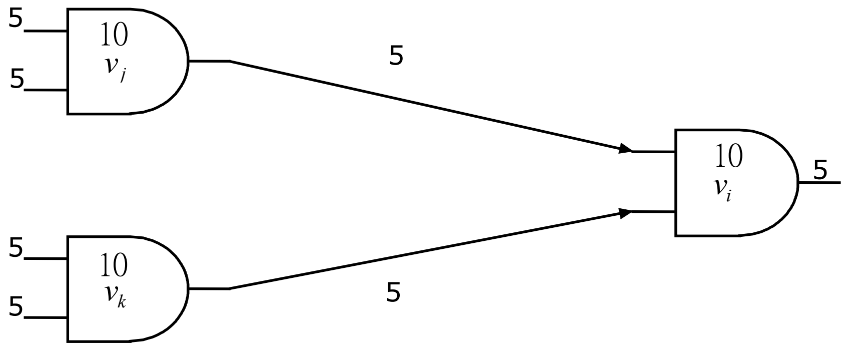


Figure 7. The distribution of Lagrangian multipliers of circuit in Figure 6 without KKT condition

will fall into local optimal easily due to lack of global information. In Figure 6, there are three gates in the circuit and the slack on all components are the same, -5. We assume that the initial value of Lagrangian multiplier is zero. Without KKT conditions, the sub-gradient $\Delta\lambda$ for each constraint is 5, assuming that $\rho = 1$ for simplicity. Then Lagrangian multiplier λ is 5 and μ is 10 when we calculate μ by the sum of λ from fan-in gates. We can see that the weights for these three gates are the same, as shown in Figure 7. It means that the power allowed to trade for timing on these three gates is the same. For simplicity, we assume that the efficiency of trading power for timing among these three gates is the same. If we change the solution of v_i to make constraints on v_i satisfied, then the constraints on v_j and v_k will also be satisfied. However, if the change is made on either v_j or v_k, the constraints on v_i is still violated unless the change is made on v_j and v_k concurrently. The power consumption of changing v_j and v_k will be twice than that of changing v_i. However, KKT conditions provide us a very good way to avoid the above situation. KKT conditions can help us to solve LDP with global information. The

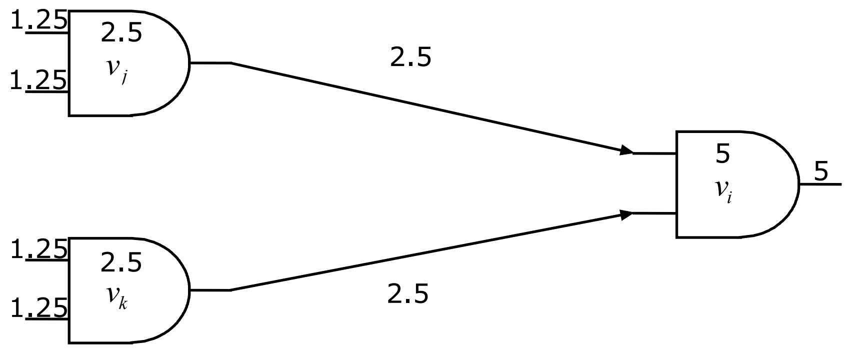


Figure 8. The distribution of Lagrangian multipliers of circuit in Figure 6 with KKT condition

distribution of λ and μ with KKT conditions for the circuit in Figure 6 is shown in Figure 8. Both of μ_j and μ_k are only half of μ_i. Hence only when the efficiency to trade power for timing of v_j and v_k are both twice better than that of v_i, the situation of changing v_j and v_k concurrently, instead of v_i will happen. By the example, we can see the importance of KKT conditions when we solve LDP.

The optimal constraints of KKT conditions for $\vec{\lambda}$ of gates are very similar to the flow constraints for nodes in flow network, in flow must being equal to out flow. In order to keep KKT conditions hold when solving LDP, we calculate sub-gradient direction by using the same idea of distributing flows in flow network. At first, we calculate flows, $\Delta\vec{\lambda}$, at source nodes and then distribute flows toward sinks in circuit, $G(V, E)$, without edge capacity limits. With that, we can guarantee that the KKT conditions are always met.

The disadvantage of sub-gradient method

In the sub-gradient method, $\Delta\vec{\lambda}$ is decided by slack. Under discrete solution space, circuit timing on some sensitive paths changes drastically and oscillates between big number of positive and negative slack. Sub-gradients on those paths are relatively large and that makes Lagrangian relaxation jumping over the optimal solution. Those oscillations will result in stall iterations and make Lagrangian relaxation hard to reach the optimal solution. In the other hands, if small slack happen on less sensitive paths, sub-gradients on those paths are small. The small steps for those paths cause us a lot of iterations to approach optimal solution. Hence, in this work we want to find $\Delta\vec{\lambda}$ considering about slack and sensitivity of paths, or sub-circuits, concurrently.

Furthermore, in some cases the sum of sub-gradients on fan-in edges of a gate is far away from that on fan-out edges. It is hard to keep KKT constraints satisfied under above situation. Here we take gate 3 in Figure 9 for example. The slack on e_{31}, e_{32} and

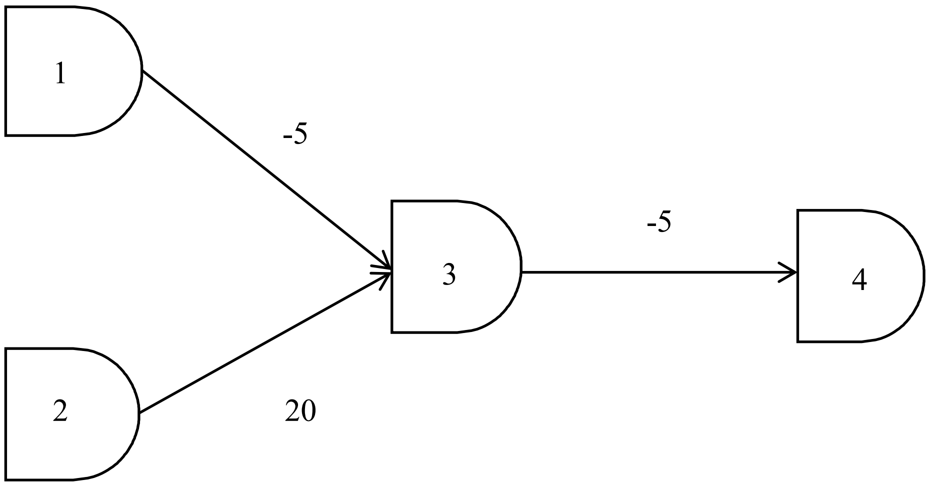


Figure 9. An example circuit with different values of Lagrangian multiplier on fan-in and fan-out edges

e_{32} is -5, 20 and -5, respectively, and sub-gradients on those edges are 5, -20 and 5. Therefore, the sum of sub-gradients on fan-in edges is -15 and that on fan-out edges is 5. It is confusing that how to make KKT conditions satisfied on gate 3 and which number $\Delta\mu_3$ should be. Generally, value of Lagrangian multiplier on a gate with negative slack should keep increasing until its slack become positive. When solving LDP, it should be avoided to decrease value of Lagrangian multipliers, applying negative $\Delta\mu$, on those gates with negative slack. However, the sub-gradient method cannot deal with the situations we mention above well because the sub-gradient direction is only related to slack.

Sensitivity of Lagrangian multiplier and arrival time

In the following chapters, we will introduce our methods to find directions for Lagrangian relaxation better than the sub-gradient method. When solving LDP, if $\Delta\lambda$ is distributed to a component, it will keep being distributed in the sub-circuit rooted by the

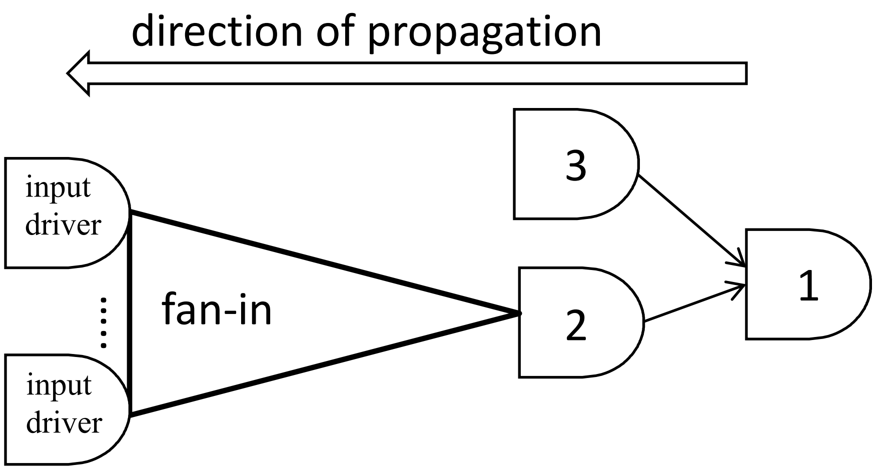


Figure 10. The effect of distributing $\Delta\lambda$

component, fan-in cone, as shown in Figure 10. Therefore the effect of distributing $\Delta\lambda$ to the component is equal to that of distributing $\Delta\lambda$ to the sub-circuit. Then, due to the distribution of $\Delta\lambda$ in the sub-circuit, timing inside the sub-circuit may change and arrival time of the component also change. Thus, the arrival time of a component will change along with its Lagrangian multiplier and there must be a kind of relationship between Lagrangian multiplier and arrival time of a component. Here we assume that the relationship can be modeled as a function η for each component, where $a = \eta(\lambda)$. Arrival time a of a component is a function of its Lagrangian multiplier value λ. Each component has its own η function. Then the first order differentiation of η function means the ratio of trading Lagrangian multiplier for arrival time, $\Delta a = \eta'(\lambda)\Delta\lambda$. With the known η' function, it is more reasonable to calculate $\Delta\lambda$ for a component using its slack divided by $\eta'(\lambda_{cur})$, $\Delta\lambda = (a - q)/\eta'(\lambda_{cur})$, where λ_{cur} is its current value of Lagrangian multiplier. The $\Delta\lambda$ is what the component needs now to make its constraint satisfied. We think that considering about ratio of trading Lagrangian multiplier for arrival time can assign Lagrangian multipliers for constraints more accurately than sub-gradient method.

Unfortunately, it is impossible to directly figure out the equation of $\eta(\lambda)$ because sizes and Vt levels are not continuous, resulting in non-smooth Lagrangian multiplier and arrival time curve. The curve shown in Figure 11 is extracted from simulation results of a component in C432 benchmark. It shows that the curve of Lagrangian multiplier and arrival time is not only non-smooth but complex so that we cannot use either a linear

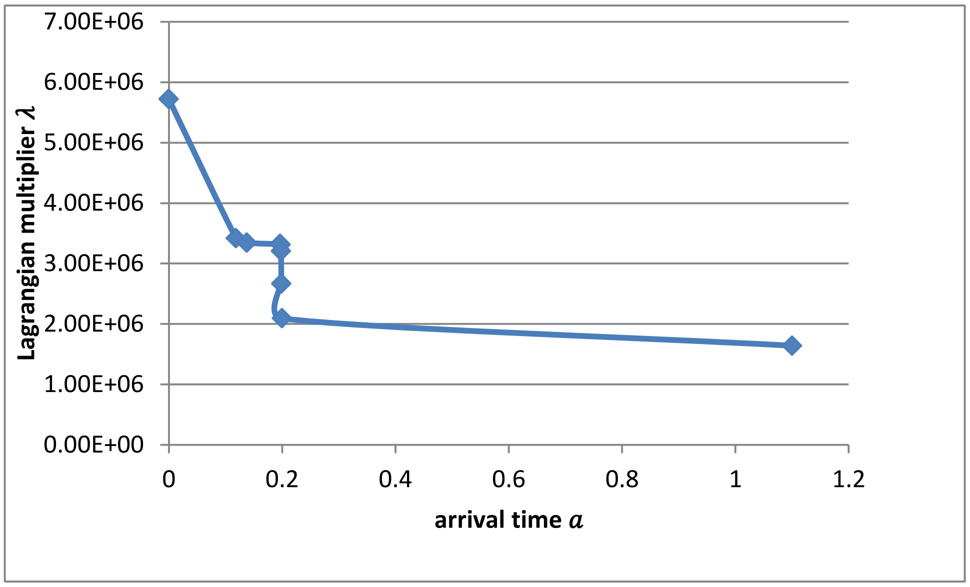


Figure 11. Lagrangian multiplier and arrival time curve

function, or even quadratic function, to model it. Therefore, the well-known curve-fitting method cannot be applied here.

My *slope* function

Due to complex curves of Lagrangian multiplier and arrival time of components, it is impossible to directly formulate η function and then calculate η' function we mention in previous chapter. Here, we use a history-based method to substitute the η' function. A table *T* associated with each component is used to record pairs of its Lagrangian multiplier and arrival time, (λ, a), and updated iteratively. Those pairs stored in *T* are sorted by λ value. We assume that the value of η' function at a given point is similar to that at its neighbors. Therefore, we use a function called *slope* to replace the η' function for calculating $\Delta\lambda$. The *slope* function has two parameters, *T* and λ, where *T* is the table associated with a component and λ is the Lagrangian multiplier value of this

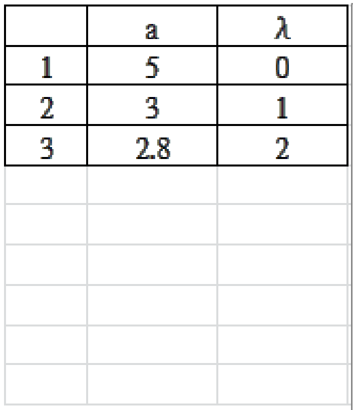
```
ALGORITHM slope
Input : $a\ table\ T\ containing\ \lambda[]\ and\ a[],\ \lambda_{current}$
Output : $slope\_now$

For j = 1 to size(T)
        $weight_j = 1/(\ fabs((\lambda[j]\ +\ \lambda[j-1])/2 - \lambda_{current})\ );$
        total_weight+=$weight_j$;
        $slope_j = (a[j-1] - a[j])/(\lambda[j-1] - \lambda[j]);$
        total_slope+=$slope_j \times weight_j$;
$slope\_now$ = total_slope/ total_weight;
```

Figure 12. The pseudo code of calculating $slope$ function

component. The *slope* function uses historic information stored in table *T* to predict the ratio of trading Lagrangian multiplier for arrival time at current λ value, slope at current point. We assume that the correlation between ratios is inversely proportional to the difference between values of their Lagrangian multipliers. Here, we use weight variables *w* to present the correlations and the sum of all the weight variables is one. Then, the return value of $slope$ function is the sum of product of each slope between two consecutive pairs in the table *T* and its corresponding weight variable. The function $slope(T, \lambda_{cur})$ can be written as

$slope(T, \lambda_{cur}) = \sum_{(\lambda_{j-1},a_{j-1}),(\lambda_j,a_j)\in T}\left(w_j\,(a_{j-1} - a_j)/(\lambda_{j-1} - \lambda_j)\right)$, where $w_j \propto \left|1/\left((\lambda_{j-1} + \lambda_j)/2 - \lambda_{cur}\right)\right|$. The pseudo code of our history based method for calculating $slope$ function is shown in Figure 12. Here we use a simple example in Figure 13 to show the calculation of *slope* function. There are four pairs in a table *T* and λ is 2. Then the return value of $slope(T, 2)$ is calculated by

$(|1/((0+1)/2-2)|\,(5-3)/(0-1) + |1/((1+2)/2-2)|\,(3-2.8)/(1-2))/$



	a	λ
1	5	0
2	3	1
3	2.8	2

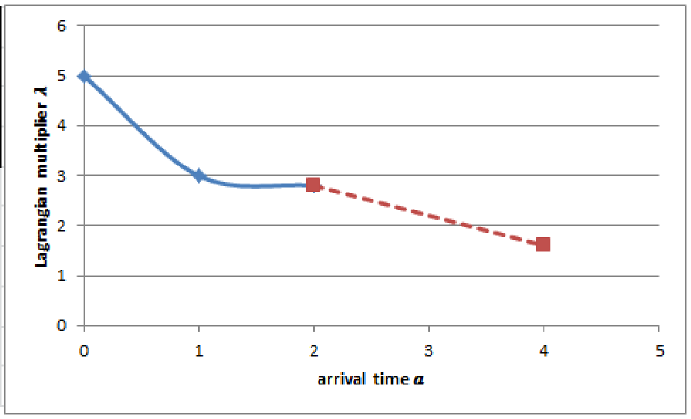


Figure 13. An example of calculation of *slope* function

$(|1/((0+1)/2-2)| + |1/((1+2)/2-2)|) = -1.5$

The solid curve is formed by the data in *T* and the dashed line is the prediction based on the return value of $slope(T, 2)$. The slope of the dashed line is -1.5.

In order to keep history data accurate, we perform pruning in each table. Generally, a larger value of Lagrangian multiplier allows more power for timing, resulting in smaller arrival time for a component. Therefore, if a bigger Lagrangian multiplier results in larger arrival time, it means that the distribution of Lagrangian multiplier in the fan-in cone of the component is not good enough. Some power is spent to speed up some unnecessary paths and those timing improvements have no benefit for the timing of the fan-in cone. We define that (λ_i, a_i) is inferior to (λ_j, a_j) if $\lambda_i \geq \lambda_j$ and $a_i \geq a_j$ and those interior pairs will be pruned.

On the other hand, size of table keeps growing with the increase of simulation iterations. To prevent from memory explosion, we need to remove some data which is less useful to reduce table size. We know that finally the vector of Lagrangian multipliers will converge so that the Lagrangian multipliers only change in a small region. The information which is far away from current operating region has little influence when calculating *slope* function. Therefore, the pair (λ_i, a_i) with biggest value of Lagrangian multiplier difference, $|\lambda_i - \lambda_{current}|$, is the least useful data so it will be removed when the size of the table exceeds a user-defined limit.

Distribution of Lagrangian multipliers

With the *slope* function for each component, we can start to distribute Lagrangian multipliers in $G(V, E)$. Here, we distribute Lagrangian multipliers in reversed topological order, from output loads to input resistors, working with arrival time information. For a circuit, output loads are treated as sources and input resistors work as sinks. The process of distribution can also be performed in topological order with required arrival time information.

In each iteration, we calculate the change of Lagrangian multipliers Δλ at sources and the Δλ for each source is equal to $(A_i - a_i)/slope(T_i, \lambda_i)$ where $v_i \in T$. The symbol a_i, A_i, T_i λ_i is arrival time, given required arrival time, data table and Lagrangian multiplier of v_i, respectively. Then those Δλs are propagated toward sinks in reversed topological order. For each logic gate, it receives Δλs from its fan-out gates and then propagate the sum of Δλ it receives, Δμ, to its fan-in gates. The difference between Δλ propagation and common flow propagation in flow network is that Δλ can be negative. In general, Δλ is negative if the constraint on the component is met, positive slack.

When distributing Δλ, it is not always possible to assign as much Δλ as a component needs to satisfy its constraint without violating KKT conditions. In static timing analysis, the arrival time of a gate is the maximum arrival time among its fan-in gates. It is not useful to reduce arrival time on those non-timing critical fan-in gates. Thus for a gate, when distributing its Δμ for fan-in gates, the main goal is to make arrival time of them equal. Therefore, we will not waste power on the non-timing critical sub-circuits by assigning too large number of Lagrangian multiplier to them. At first, we calculate the expected arrival time a_{exp} for all the fan-in gates such that the sum of Δλs on them is equal to that on fan-in gates. The equation for calculating a_{exp} for $v_i \in V$ is $\sum_{v_j \in fanin(v_i)} (a_{exp} - a_j)/slope(T_j, \mu_j) = \sum_{v_k \in fanout(v_i)} \Delta\lambda_{ki} = \Delta\mu_i$. With a_{exp}, we

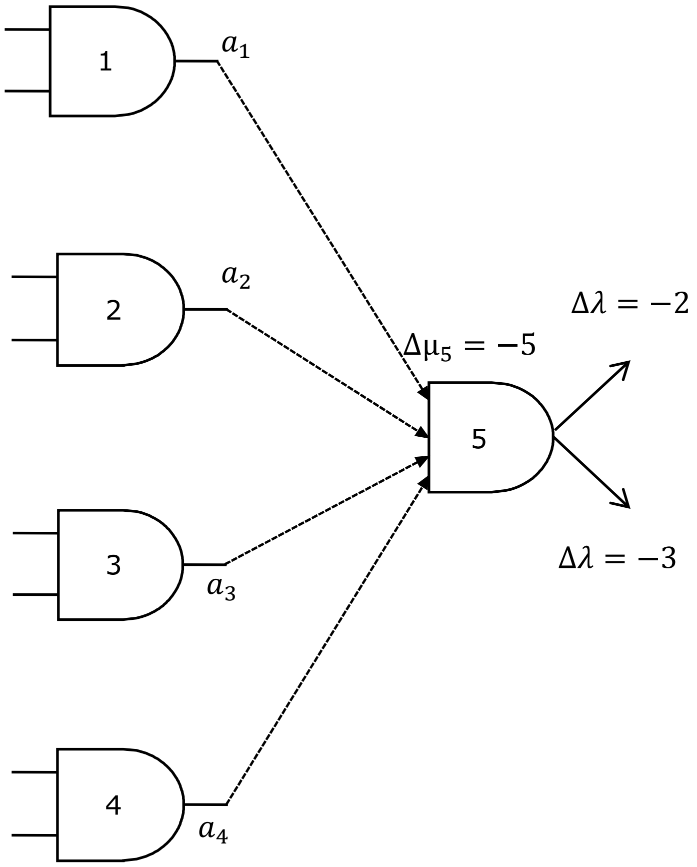


Figure 14. A circuit example with five logic gates and $\Delta\mu_5$ is -5

can calculate $\Delta\lambda$ for each fan-in gate and also guarantee KKT satisfied. The equation for calculating $\Delta\lambda$ is For all $v_j \in fanin(v_i)$, $\Delta\lambda_{ij} = (a_{exp} - a_j)/slope(T_j, \mu_j)$. Here we use an example to illustrate the distribution of Lagrangian multipliers in our work. When distributing $\Delta\mu$ of a gate, we regard $\Delta\mu$ as flows passing among the gate and its fan-in gates. If $\Delta\mu$ of a gate is negative/positive, it means the gate work as a sink/source and all its fan-in gates are treated as either sources or sinks to satisfy the flow constraints. We take the circuit in Figure 14 for example. There are five gates and $\Delta\mu_5$ is -5. So gate 5 is treated as a sink and we need to generate flows toward it from its fan-in gates and the sum of coming flows is 5. By the equations we mention above, the result of distributing $\Delta\mu$as flows is shown in Figure 15 where the number on an arrow presents flows going on the corresponding edge. We take gate 1 for example. Due to its arrival time, a_{exp} and value of *slope* function, $\Delta\mu_1$ is 2. Hence gate 1 works as a sink to receive flows from other gates and the total number of flows is 2. We can see that the total flows going to gates 5 are 5 so the flow constraint on gate 5 is satisfied.

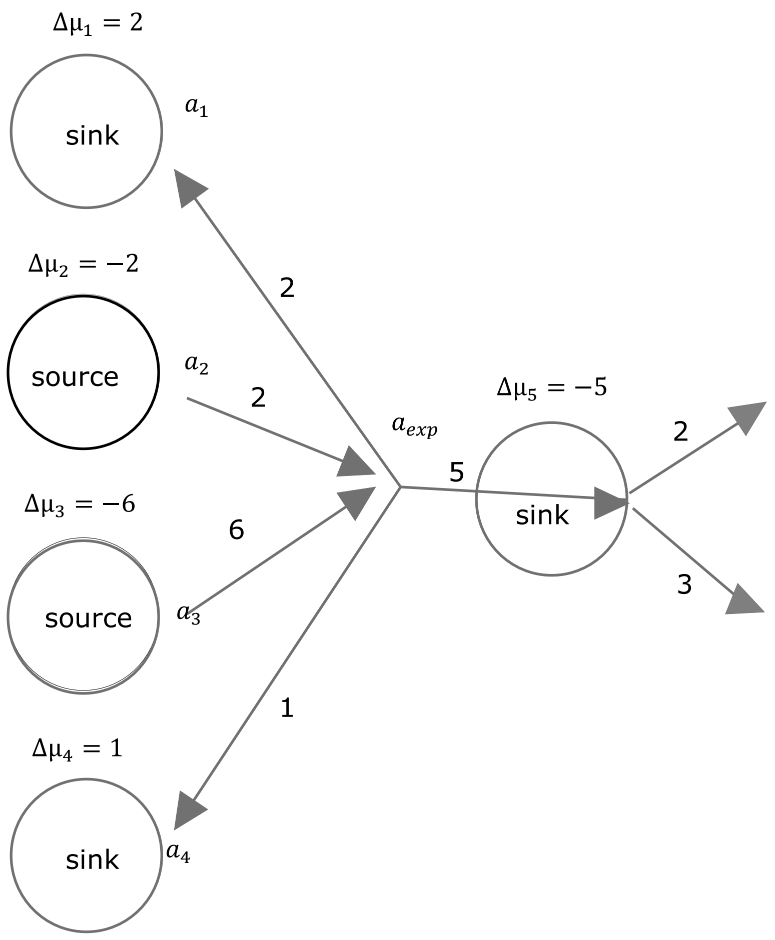


Figure 15. The result of distributing Δμ as flows

CHAPTER VI

IMPROVED ALGORITHM FOR SOLVING *LRS/λ*

For Lagrangian sub-problem, we apply the local optimal sizing method to solve gates individually. We find size and Vt level for a gate while keeping all other gates fixed. For a gate $v_i \in X$, the L_λ can be written as

$$L_\lambda = \alpha_i p_i^{w_i u_i} + \left(\sum_{v_j \in fanin(v_i)} \mu_j r_j\right) c_i^{w_i u_i} + \mu_i r_i^{w_i u_i} C_i + terms\ independent\ of\ v_i,$$

where $w_i \in W_i, u_i \in U_i$. Due to discrete sizes and Vt levels, we cannot solve L_λ by any mathematical method. Here we use a table look-up method to find the solution for each gate. For a gate $v_i \in X$, we have known $r_i^{w_i u_i}$ and $c_i^{w_i u_i}$ for different combination of size

```
ALGORITHM solve_sub_problem
Input : combinational circuit G and cell library L
Output : size and Vt level for all gates in G

For all the gates $v_i \in X$ in G
    For $j \in sizes$ and $k \in Vt\ levels\ in\ L$
        now_l = $\alpha_i p_i^{jk} + \mu_i r_i^{jk} C_i$
        For $v_y \in fanin(v_i)$
            now_l +=$\mu_y r_y c_i^{jk}$
    $if(now\_l < min\_l)$
        $min\_l$ = now_l
        $size_i = j$
        $vt_i = k$
For all the gates $v_i \in X$ in G
    Implement $v_i$ by $size_i$ and $vt_i$
```

Figure 16. The algorithm for solving Lagrangian sub-problem

$w_i \in W_i$ and Vt level $u_i \in U_i$. Then we evaluate L_λ with all the combinations of size and Vt level to find the size and Vt level combination resulting in minimum L_λ. We show the algorithm for solving sub-problem in Figure 16. With that, we can avoid the remarkable errors to round continuous solutions into the nearest feasible size and Vt level. Due to the nature of greedy method, we need to iteratively solve the Lagrangian sub-problem until converge. However, sometimes $slope$ function may be not accurate enough due to some suddenly huge arrival time change when size or Vt level of gates change inside the sub-circuit. Therefore, we need to do some adjustment for $\Delta\lambda s$ when iteratively solving Lagrangian sub-problem. Here no new $\Delta\lambda$ comes into $G(V, E)$ at output loads, only redistributing $\Delta\lambda$ by $slope$ function with updated table data of each gate. Our result shows that the minor change for Lagrangian multipliers will not delay convergence too much.

CHAPTER VII

EXPERIMENTAL RESULTS

We compare our method with a Lagrangian relaxation based method using sub-gradient for dual problem. The initial Lagrangian multipliers and method for Lagrangian sub-problem are the same. Our experiments focus on the comparison between our Lagrangian dual problem method using $slope$ function and sub-gradient method in [12]. Besides, we also compare our method with a state-of-the-art Lagrangian relaxation based method under discrete solution space [17]. The core idea of [17] is that the Lagrangian sub-problem is solved by a DP-like method including consistency relaxation and coupled bi-directional search. In this work, ISCAS85 benchmarks are used for comparison, synthesized by SIS [20] and placed by mPL [21]. The cell library is based on $70nm$ technology. There are eight size options, 0.25X, 0.5X, 1X, 2X, 3X, 4X, 6X and 8X of original size, and three Vt levels for each logic gate. There are totally 24 different implementations for each logic gate. The V_{dd} is set to 0.9V. Elmore delay model and an analytical leakage power model [5] are used for delay and power calculation in our experiments. In addition, wire delay is also included in circuit delay. At first, we set all the gates to minimum power consumption implementation, smallest size and highest Vt level, and λ for all constraints is zero. Table 1 shows the experimental results of our method, sub-gradient method [12] and [17] under loose timing constraints. The experimental results show that our method reduces 50% power consumption on average with only 35% run time overhead compared with sub-gradient method and 58% power

Table 1. Experimental results with loose timing constraints

	# of gates	initial setup		Sub-gradient method			Our method			[17]		
testcase		power	slack	power	slack	time	power	slack	time	power	slack	time
chain	11	9.3369	-215.5	27.231	5	0.04	27.231	5	0.05	27.231	5	0.85
c432	289	221.75	-8033	249.81	17	0.8	238.67	18	1.14	257.31	0.58	19.61
c499	539	418.78	-4198	874.48	614	1.56	498.76	7	2.36	815.52	24	40.86
c880	340	259.4	-3219	327.77	231	0.94	279.81	1	1.37	326.08	59	22.02
c1355	579	426.6	-4084	736.42	38	1.7	522.11	3	2.71	713.41	20	40.57
c1908	722	582.84	-5716	878.05	22	2.2	666.72	70	3.45	1039.6	876	58.93
c2670	1082	725.1	-12970	759.99	711	2.97	734.22	115	4.54	788.08	163	79.99
c3540	1208	994.53	-5873	2012.5	718	3.85	1147.6	7	6.11	2863.3	1233	108.4
c5315	2440	1941.7	-8156	3165.6	1033	8.06	2171.9	17	12	4140.7	1091	201.2
C6288	2342	1819.7	-7787	3951.3	310	8.1	2518.8	13	13	2909.2	21	224.5
c7552	3115	2390	-16900	3897.8	1386	10.7	2445.5	107	16.2	3929.6	1386	275.9
				1.5004		0.65	1		1	1.5829		17.04

Table 2. Experimental results with tight timing constraints

	# of gates	initial setup		Sub-gradient method		Our method		[17]	
testcase		power	slack	power	slack	power	slack	power	slack
chain	11	9.33686	-295.62	60.8433	-13.8726	104.428	0.095859	-18.7871	55.6521
c432	289	221.745	-10379.8	832.352	-33.1426	803.687	0.6065	818.577	-79.6789
c499	539	418.777	-5389.68	1545.37	-11.4963	1522.83	1.70575	1552.93	-22.4923
c880	340	259.399	-4239.06	515.522	-31.7455	549.426	15.5728	520.756	-38.0014
c1355	579	426.595	-5353.66	1470	-5.34412	1403.9	7.59088	1468.18	-9.525
c1908	722	582.84	-7286.43	1452.71	-12.8188	1402.68	5.8775	1397.21	-10.1248
c2670	1082	725.101	-16177.1	1465.94	-32.7513	1312.64	9.08075	1549.78	-49.9214
c3540	1208	994.532	-7369.01	2650.47	-116.61	3016.61	20.0094	2658.32	-123.507
c5315	2440	1941.65	-9956.27	3627.44	-198.951	4088.71	7.8015	3667.3	-238.416
C6288	2342	1819.71	-10476.1	6305.54	-29.436	5382.37	3.432	6142.56	-205.8
c7552	3115	2389.95	-21197.9	6875.74	-97.2085	5433.92	20.577	5202.18	-75.3418

consumption on average with faster run time than [17]. Then we run experiments with tight timing constraints to change the main objective to circuit performance. The results in Table 2 show that our method can find the solution with positive slack for all the benchmarks but sub-gradient method and [17] cannot. Therefore, the experimental results demonstrate the robustness of our method for either power reduction or circuit performance.

In addition to ISCAS'85 benchmark, we also run experiments on a chain benchmark. The chain structure is a simple but special case. In a chain, almost all conditions for each gate are the same, including slack, value of Lagrangian multiplier, input resistance and output load. Once the value of Lagrangian multiplier reaches certain threshold values, a lot of gates will change at the same time. Because the circuit input

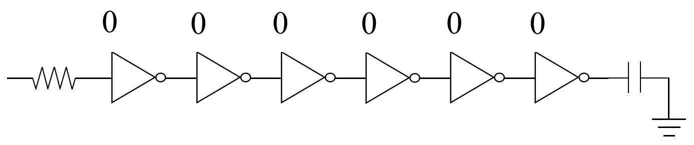


Figure 17. Initial setup of a chain

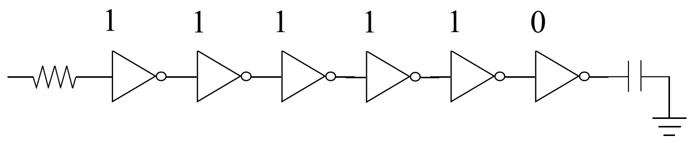


Figure 18. The first five gates size up to option 1

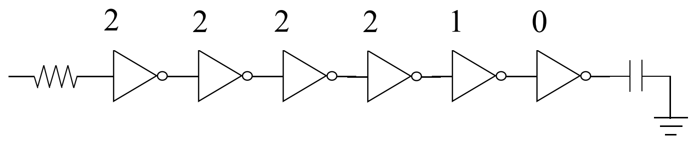


Figure 19. The first four gates size up to option 2

driver is always fixed and relatively small, so the first gate will size up at first, then the change will keep passing along the chain. If output resistance of a gate reduces, the reduction will facilitate the sizing up of the following gate. In other words, for a gate, once its input resistance becomes smaller, the gate can size up with smaller Lagrangian multiplier value. Oppositely, the last gate in a chain needs a bigger Lagrangian multiplier value to size up due to smaller output load than other gates. Generally, the implementation of gates in the middle will be the same unless the input resistances or output loads of certain gates change. We use an example to show how Lagrangian relaxation-based method works in a chain case. In Figure 17, all gates are initialized to size option 0, smallest size, and then we start to increase Lagrangian multipliers of gates. When reaching a certain value, the first five gates will size up to option 1 to speed up the chain. If we keep increasing values of Lagrangian multipliers, then the first four gates will size up to option 2 at the same time. Because the value of Lagrangian multipliers of all gates is the same, many gates will change their implementations at the same time. So it is impossible for Lagrangian relaxation to find the optimal solution of a chain for certain timing constraints under discrete solution space. In other words, there is no any vector of Lagrangian multipliers which can result in the optimal solution under certain timing constraints in a chain case. For example, we assume the circuit delay in Figure 18 is 10 and that in Figure 19 is 15. We set the timing constraint of this circuit to 13 and we assume both the circuit solutions in Figure 18 and Figure 19 are not the optimal solution. Then Lagrangian relaxation-based method will oscillate between them. Hence, it is the reason that our method cannot reach optimal solution even in a simple chain benchmark.

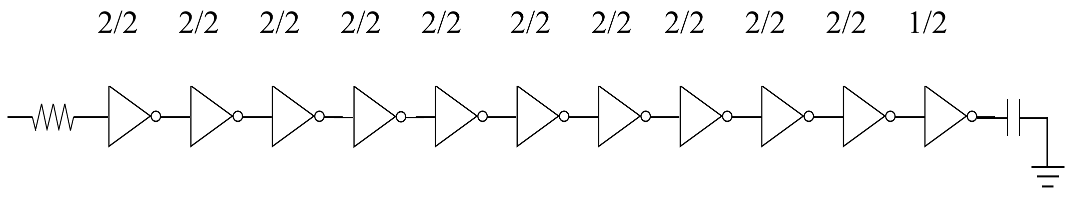


Figure 20. Solution of our method for the chain benchmark

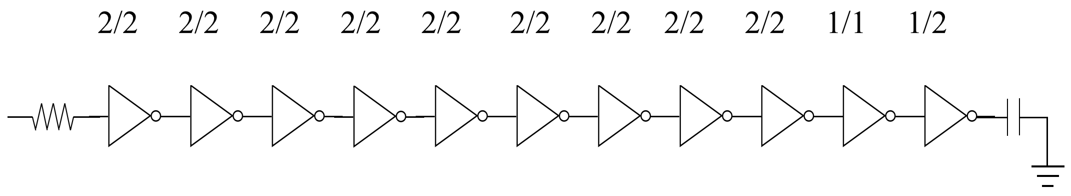


Figure 21. The optimal solution for the chain benchmark

In our chain benchmark, there are 11 identical gates and the distance between any two consecutive gates is the same. Under the setup we used in Table 1, the optimal power of chain benchmark is 26.6114 with 0.057 slack. Our method, sub-gradient method and [17] all cannot reach the optimal solution. The result of our method of the chain benchmark is shown in Figure 20 and the optimal solution is shown in Figure 21. The number above a gate means its implementation. The number 2/2 means size option 2 and Vt level 2.

In addition, we show the detailed slack and power information iteratively for C432 benchmark. In Figure 22, it shows that the power consumption of the circuit only change in small region close to 250. In Figure 23, we can see that the slack of the circuit only has minor change around 0. The results show the convergence of our method. However, in Figure 24 and 25, the slack and power information for each iteration show that sub-gradient method under discrete solution space oscilates and does not converge to a feasible solution. We can see that the timing information oscilate far away from feasible solution with slack greater than zero in several iterations. Therefore, our improved algorithm for solving dual problem make Lagrangain relaxation stable and converge faster to final result.

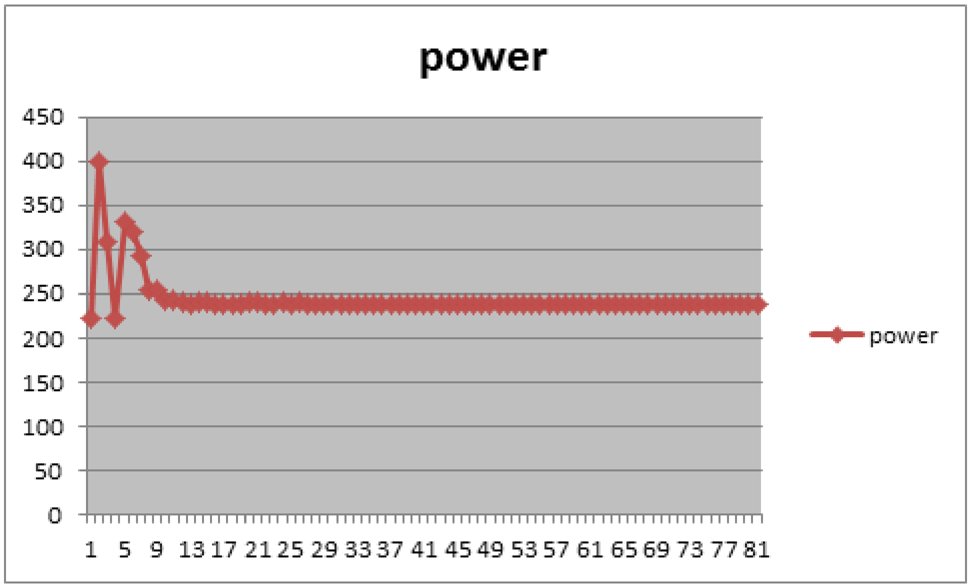


Figure 22. Power information of iterations by using our method

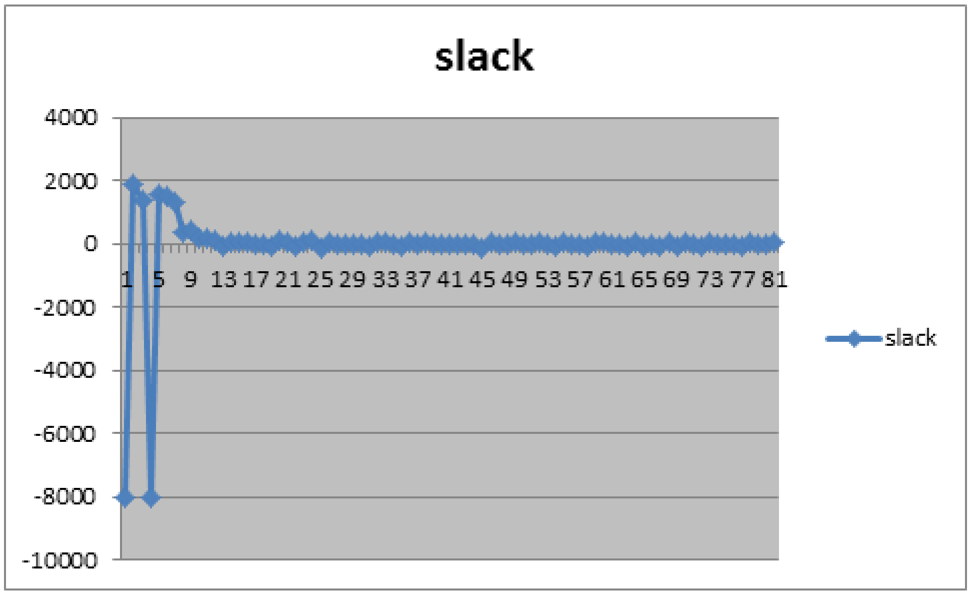


Figure 23. Slack information of iterations by using our method

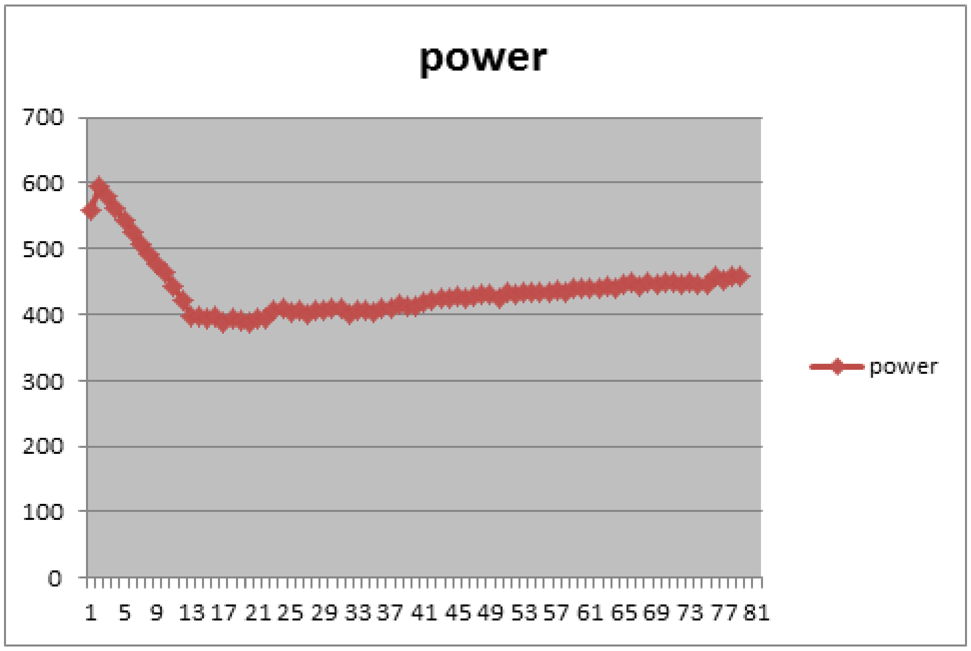


Figure 24. Power information of iterations by using sub-gradient method

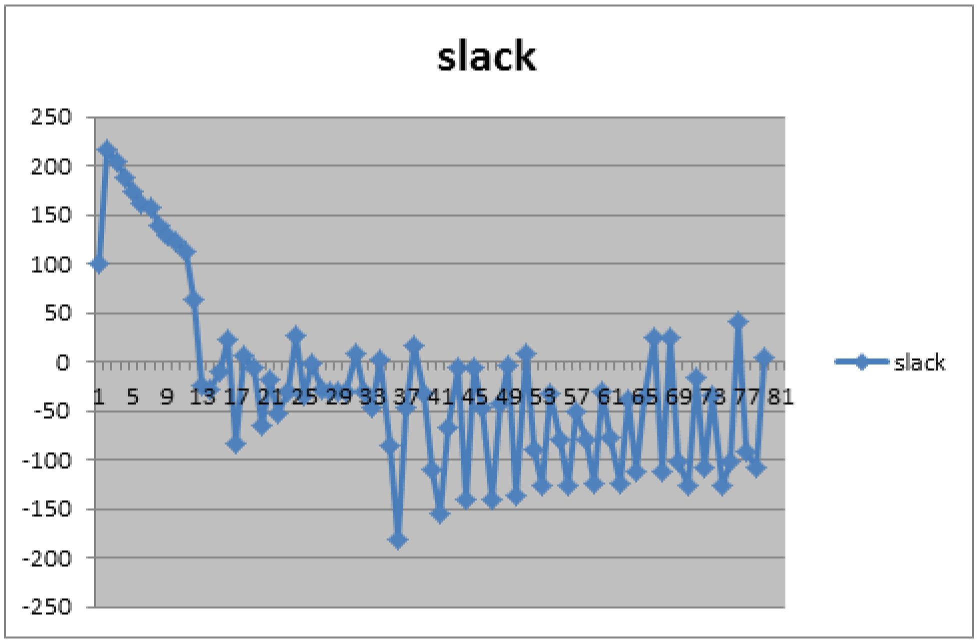


Figure 25. Slack information of iterations by using sub-gradient method

CHAPTER VIII

CONCLUSION

In this work, we propose an improved Lagrangian relaxation method for simultaneous discrete gate sizing and Vt assignment. The main idea of this work is that we distribute Lagrangian multipliers based on not only slack but sensitivity of timing and Lagrangian multipliers by our *slope* function. The Lagrangian multiplier distributed on each component is more accurate so that timing constraints will be just satisfied and no extra power will be wasted. We use a practical cell library in this work with discrete sizes and Vt levels. Therefore, our method can be applied to any industrial standard cell based designs for circuit optimization. The experimental results show that our method can improve 50% and 58% in power consumption under the same timing constraints than a Lagrangian relaxation method using sub-gradient method and [17]. In addition, our method can also find feasible solution but sub-gradient and [17] cannot when tight timing constraints are given. As a result, our improved Lagrangian relaxation method is powerful enough to handle discrete sizes and Vt levels with good solution quality and tolerable run time cost.

REFERENCES

[1] J. Fishburn and A. Dunlop, "TILOS: A posynomial programming approach to transistor sizing," in *Proc. Int. Conf. Comput.-Aided Design*, 1985, pp. 326–328.

[2] L. Wei, Z. Chen, K. Roy, and V. De, "Design and optimization of dual threshold circuits for low voltage low power application," *IEEE Trans. Very Large Scale Integr.*, vol. 7, no. 1, pp. 16–24, Mar. 1999.

[3] V. Sundararajan and K.K. Parhi, "Low power synthesis of dual threshold voltage CMOS circuits," in *Proc. Int. Symp. Low Power Electron. Design*, 1999, pp. 139–144.

[4] V. Khandelwal, A. Davoodi, and A. Srivastava, "Simultaneous Vt selection and assignment for leakage optimization," *IEEE Trans. Very Large Scale Integr.*, vol. 13, no. 6, pp. 762–765, 2005.

[5] M. Ketkar and S. S. Sapatnekar, "Standby power optimization via transistor sizing and dual threshold voltage assignment," in *Proc. Int. Conf. Comput.-Aided Design*, 2002, pp. 375–378.

[6] D. Nguyen, A. Davare, M. Orshansky, D. Chinnery, B. Thompson, and K. Keutzer, "Minimizion of dynamic and static power through joint assignment of threshold voltages and sizing optimization," in *Proc. Int. Symp. Low Power Electron. Design*, 2003, pp. 158–162.

[7] S. Shah, A. Srivastava, D. Sharma, D. Sylvester, D. Balaauw, and V. Zolotov, "Discrete Vt assignment and gate sizing using a self-snapping continuous formulation," in *Proc. Int. Conf. Comput.-Aided Design*, 2005, pp. 704–710.

[8] S. Sirichotiyakul, T. Edwards, C. Oh, J. Zuo, A. Dharchoudhury, R. Panda, and D. Blaauw, “Stand-by power minimization through simultaneous threshold voltage selection and circuit sizing,” in *Proc. Design Autom. Conf.*, 1999, pp. 436–441.

[9] H. Chou, Y. Wang, and C. Chen, “Fast and effective gate-sizing with multiple-Vt assignment using generalized Lagrangian relaxation,” in *Proc. Asia South Pacific Design Autom. Conf.*, 2005, pp. 381–386.

[10] T.-H. Wu, L. Xie, and A. Davoodi, “A parallel and randomized algorithm for large-scale dual-Vt assignment and continuous gate sizing,” in *Proc. Int. Symp. Low Power Electron. Design*, 2008, pp. 45–50.

[11] J. Wang, D. Das, and H. Zhou, “Gate sizing by Lagrangian relaxation revisited,” in *Proc. Int. Conf. Comput.-Aided Design*, 2007, pp. 111-118.

[12] C. Chen, C. C. N. Chu, and D. F. Wong, “Fast and exact simultaneous gate and wire sizing by Lagrangian relaxation,” *IEEE Trans. Compu.-Aided Design*, vol. 18, no. 7, pp. 1014–1025, 1999.

[13] K. Kasamsetty, M. Ketkar, and S.S. Sapatnekar, “A new class of convex functions for delay modeling and its application to the transistor sizing problem [CMOS gates],” *IEEE Trans. Compu.-Aided Design*, vol. 13, no. 6, pp. 779–788, 2000.

[14] S. Roy, W. Chen, and C.C. Chen, “ConvexFit: An optimal minimum-error convex fitting and smoothing algorithm with application to gate-sizing,” in *Proc. Int. Conf. Comput.-Aided Design*, 2005, pp. 196–204.

[15] S. Hu, M. Ketkar, and J. Hu, “Gate sizing for cell library based designs,” in *Proc. Design Autom. Conf.*, 2007, pp. 847–852.

[16] H. Tennakoon and C. Sechen, “Gate sizing using Lagrangian relaxation combined with a fast gradient-based pre-processing step,” in *Proc. Int. Conf. Comput.-Aided Design*, 2001, pp. 395-402.

[17] Y. Liu and J. Hu, “A new algorithm for simultaneous gate sizing and threshold voltage assignment,” in *Proc. Int. Symp.on Physical Design*, 2009, pp. 27-34.

[18] J. Shyu, J. P. Fishburn, A. E. Dunlop, and A. L. Sangiovanni-Vincentelli, “Optimization-based transistor sizing,” *IEEE J. Solid-State Circuits*, vol. 23, no. 2, pp. 400 - 409, April, 1988.

[19] R. Chadha and J. Bhasker. *Static Timing Analysis for Nanometer Designs*. Springer, New York, Jan. 2009

[20] E.M. Sentovich, K.J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P.R. Stephan, R.K. Brayton, and A.L. Snagiovanni. “SIS: A system for sequential circuit synthesis,” Electron. Res. Lab., Univ. California, Berkeley, CA, Mem. UCB/ERL M92/41, May 1992.

[21] *CPMO-constrained placement by multilevel optimization*. http://ballade.cs.ucla.edu/cpmo. Computer Science Department, UCLA, April, 2006.

VITA

Name:	Yi-Le Huang
Address:	Department of Electrical and Computer Engineering, Texas A&M University, 214 Zachry Engineering Center, TAMU 3128 College Station, Texas 77843-3128
Email Address:	kiwe@tamu.edu
Education:	B.S., Computer Science, National Tsing Hua University, 2003 M.S., Computer Science, National Tsing Hua University, 2005