INTEGRATION, the VLSI journal 47 (2014) 347–355

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Optimal gate sizing using a self-tuning multi-objective framework

Amin Farshidi *, Logan Rakai, Laleh Behjat, David Westwick

Department of Electrical and Computer Engineering, University of Calgary, Calgary, Canada

ARTICLE INFO

Available online 6 November 2013

Keywords:
Gate sizing
Clock networks
Multi-objective optimization
Geometric programming

ABSTRACT

In this paper, we present a self-tuning multi-objective framework for geometric programming that provides a fine trade-off between the competing objectives. The significance of this framework is that the designer does not need to perform any tuning of weights of objectives. The proposed framework is applied to gate sizing and clock network buffer sizing problems. In gate sizing application, power consumption is reduced on average by 86% while delay sees only an increase of 34 ns. In clock network butter sizing application, our framework results in a significant reduction in power, 57%, and an improvement of 31 ps in skew.

1. Introduction

Many problems encountered in engineering these days require optimizing two or more competing objectives. This means that all objectives cannot be minimized simultaneously as a decrease in one can result in an increase in the others. For example, in the gate sizing problem in physical design of digital integrated circuits, power, area and delay are desired to be minimized simultaneously [1]. However, decreasing area and power normally results in increased delay.

There are three common approaches for solving multi-objective problems. In one approach, one of the objectives is targeted for minimization and the others are considered as the constraints of the optimization problem [2–4]. An example of this approach can be found in [2], where the gate sizing problem is formulated as a delay optimization problem that minimizes the maximum delay of the circuit while power consumption and area are dealt with as constraints.

In the second approach, a weighted sum of all the objectives is considered as the target for optimization [5–7], where the weights used in the objective are constant numbers. An example of this approach can be found in [5], where the gate sizing problem is solved by optimizing a scalarized weighted sum of power and delay objectives.

In the third approach, evolutionary algorithms are used to solve the multi-objective optimization problems. An example of this approach is presented in [8], where genetic algorithm is employed for solving multi-objective problems.

All of these approaches require the designers to determine an acceptable trade-off between the objectives. However, finding this trade-off requires parameter tuning. This is often a difficult task since a designer may need to generate the Pareto surface of the problem which is a time consuming process.

In this paper, we present a new technique in which the multi-objective weights are included as variables in the optimization problem. Hence, the multi-objective optimization problem is turned into a self-tuning multi-objective framework. We have shown that the proposed multi-objective framework is a convex Geometric Programming (GP) problem which can be solved efficiently. The solution of the multi-objective framework not only optimizes the sizes of the gates but also finds the weights by minimizing a combined objective. We show that by using the proposed self-tuning multi-objective framework, the need for solving several optimization problems to generate a Pareto surface, a very time consuming operation, is eliminated.

To show the efficacy of our technique, we have applied the proposed self-tuning multi-objective framework to two types of problems. The first problem is the gate sizing problem that was originally formulated as a GP in [2]. This problem is reformulated using the proposed self-tuning multi-objective framework where the maximum circuit delay and power consumption are minimized simultaneously. We show that using the proposed technique, we can reduce power consumption by 86% compared to the single objective solutions.

In the second problem, the clock network buffer sizing problem, which is formulated as a GP in [3], is also reformulated as a self-tuning multi-objective buffer sizing problem with the conflicting objectives of minimizing skew and power consumption. We show that solution of the proposed framework finds a good compromise by obtaining solutions that consume 57% less power on average and have on average 31 ps reduction in skew compared to the input clock networks.

The major contributions of this paper are:

- Proposal of a multi-objective framework for geometric programming that produces a fine trade-off between objectives without the need for designer tuning.

* Corresponding author.
E-mail address: afarshid@ucalgary.ca (A. Farshidi).

0167-9260/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.vlsi.2013.10.008

- Development of a self-tuning multi-objective gate sizing framework.
- Obtaining up to 90% improvement in power consumption in gate sizing instances.
- Development of a self-tuning multi-objective clock network buffer sizing framework.
- Obtaining on average 57% improvement in power and 31 ps improvement in skew for ISPD2009 clock networks.
- Experimental validation that the solutions of the proposed multi-objective framework provide a good trade-off between the objectives.
- Analysis of the single objective and multi-objective clock network frameworks performance in the presence of non-linearity and discretization.

The rest of this paper is organized as follows: In Section 2, previous works on gate sizing, clock network buffer sizing and the optimization techniques used in this paper are reviewed. The proposed self-tuning multi-objective framework for GP is formulated and applied to the gate sizing and clock network buffer sizing in Section 3. In Section 4, the experimental results to validate the proposed framework are presented. Finally, conclusions and future work are provided in Section 5.

2. Background

2.1. Gate sizing

In a digital circuit library, several versions of a gate that perform the same function but have different sizes exist. Each version of a gate in the library has a different driving strength, resulting in a different gate delay and power consumption [1]. An important part of designing a circuit is choosing the sizes of the gates, such that the maximum delay and the total power consumption are minimized. Therefore, the gate sizing problem is formulated using a single objective of minimizing the maximum delay or power consumption [2,9,10].

The power consumption of a gate is not easily modeled by a mathematical formula. In [2], it is shown that area is well correlated to power consumption. Therefore, in most formulations of gate sizing, the power is represented by the total area of the gates. The gate sizing is then formulated as follows:

$$\begin{aligned}&\min\ \upsilon_{\text{delay}}\\ &\text{s.t. } d_p(\mathbf{x}) < \upsilon_{\text{delay}}\ \ \forall p \in P\\ &\quad \text{Area}(\mathbf{x}) \le \max_{\text{area}}\\ &\quad l_{\min_g} \le x_{l_g}\ \ \forall g \in G\\ &\quad w_{\min_g} \le x_{w_g}\ \ \forall g \in G\end{aligned} \tag{1}$$

where $\mathbf{x}$ is the vector of all lengths and widths of the gates. $[x_{w_g}, x_{l_g}]$ are the width and length of the gate g, which belongs to the set of all gates, G, in the circuit. The minimum length and width of the gates are represented by $l_{\min_g}$ and $w_{\min_g}$, respectively, which depend on the type of gate and the technology process. Area($\mathbf{x}$) is the total gate area of the circuit while the maximum allowed gate area is represented by $\max_{\text{area}}$. The delay of a path p in the set of all paths P from an input of the circuit to an output of the circuit is shown by $d_p(\cdot)$. The target of optimization, υ_{delay}, is a variable representing the maximum circuit delay. In [2], it is shown that if the problem in (1) is formulated as a GP, it can be solved using convex optimization techniques [11]. GP formulation will be discussed in Section 2.3.

2.2. Clock network buffer sizing

Clock networks distribute the clock signal to all the clock sinks. Ideally, the clock signal is desired to arrive simultaneously at all nodes. However, the clock signal quality degrades as it goes through the circuit. Buffers are added to the clock network to maintain signal quality, but these buffers also consume large amounts of power. As in the gate sizing problem, there are several buffer sizes provided in a digital circuit library. A designer's tasks is to choose the appropriate size for each buffer such that the total power consumption of the circuit is minimized and at the same time, the clock signal arrives at all nodes at the same time [12]. The buffer sizing problem for clock networks, where the objective is to minimize the total power, or the total area, can be written as follows [3]:

$$\begin{aligned}&\min \text{Area}(\mathbf{x}) = \sum_{b \in B} x_{w_b} x_{l_b}\\ &\text{s.t. } \max\{d_i(\mathbf{x}) - d_j(\mathbf{x})\} \le t_{\text{skew}}\ \ \forall i,j \in S,\ i \ne j\\ &\quad \text{slew}_k(\mathbf{x}) \le t_{\text{slew}}\ \ \forall k \in B \cup S\\ &\quad l_{\min} \le x_{l_b},\ \ b \in B\\ &\quad w_{\min} \le x_{w_b},\ \ b \in B\end{aligned} \tag{2}$$

where $\mathbf{x} = [\mathbf{x_w}; \mathbf{x_l}]$, and $\mathbf{x_w}$ and $\mathbf{x_l}$ are vectors of buffer widths and lengths of the set of all buffers, B, in the clock tree. Total buffer area is represented by Area($\mathbf{x}$). $d_i(\mathbf{x})$ is the delay at each sink i, i.e. the time it takes for the clock signal to get from the clock source to the clock sink i in the set of all sinks S. The slew at each node k in the set of all nodes of the clock tree, i.e. clock sinks and buffers, is shown by $\text{slew}_k(\mathbf{x})$. t_{skew} and t_{slew} are the target clock skew and maximum allowed slew, respectively. $l_{\min}$ and $w_{\min}$ are the minimum length and widths of a buffer.

The formulation in (2) is a non-linear and non-convex problem. Therefore, its solution significantly depends on the initial solution and cannot be guaranteed to be a global optimum. In [3], the clock network buffer sizing problem is formulated as GP. Although, a problem in GP format is not originally convex, it can be transformed to a convex form and solved efficiently using the convex optimization techniques. The GP formulation of the buffer sizing problem can be written as

$$\begin{aligned}&\min \text{Area}(\mathbf{x}) = \sum_{b \in B} x_{w_b} x_{l_b}\\ &\text{s.t. } \mu(t_{\text{skew}} + d_{\min})^{-1} \le 1\\ &\quad d_i(\mathbf{x})\mu^{-1} \le 1\ \ \forall i \in S\\ &\quad \text{slew}_k(\mathbf{x}) t_{\text{slew}}^{-1} \le 1\ \ \forall k \in B \cup S\\ &\quad l_{\min} x_{l_b}^{-1} \le 1,\ \ b \in B\\ &\quad w_{\min} x_{w_b}^{-1} \le 1,\ \ b \in B\end{aligned} \tag{3}$$

where minimum sink delay is represented by $d_{\min}$ and μ is a dummy variable for relaxing the problem into GP format. The resultant problem is in GP format and can be turned into a convex problem [3]. The problem can then be solved using convex optimization techniques where every local minimum is global minimum.

The main focus of (3) is to minimize the total area and is referred to as the Area Objective (AO) problem in the rest of this paper. The disadvantage of this formulation is that area/power can be over-optimized at the cost of large increases in skew, which can significantly degrade the circuit's performance.

In [3], the GP formulation is further developed to address process variations in buffer lengths and widths. The resultant robust formulation improves the skew at the cost of increased area. However, since skew is still considered as a constraint, the designer needs to dedicate a significant amount of time to choose the power–skew trade-off. In addition, the clock networks sized using this technique either experience increased skew or increased