Applied Mathematics and Computation 255 (2015) 92–104

Contents lists available at ScienceDirect

Applied Mathematics and Computation

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

A provably tight delay-driven concurrently congestion mitigating global routing algorithm ☆

Radhamanjari Samanta [a,*], Adil I. Erzin [b,c], Soumyendu Raha [a], Yuriy V. Shamardin [b], Ivan I. Takhonov [c], Vyacheslav V. Zalyubovskiy [b]

[a] Supercomputer Education and Research Center, Indian Institute of Science, Bangalore, India
[b] Sobolev Institute of Mathematics, Siberian Branch, Russian Academy of Sciences, Novosibirsk, Russia
[c] Novosibirsk State University, Novosibirsk, Russia

ARTICLE INFO

Keywords:
Steiner tree
Elmore delay
Global routing
Gradient method

ABSTRACT

Routing is a very important step in VLSI physical design. A set of nets are routed under delay and resource constraints in multi-net global routing. In this paper a delay-driven congestion-aware global routing algorithm is developed, which is a heuristic based method to solve a multi-objective NP-hard optimization problem. The proposed delay-driven Steiner tree construction method is of $O(n^2 \log n)$ complexity, where n is the number of terminal points and it provides n-approximation solution of the critical time minimization problem for a certain class of grid graphs. The existing timing-driven method (Hu and Sapatnekar, 2002) has a complexity $O(n^4)$ and is implemented on nets with small number of sinks. Next we propose a FPTAS Gradient algorithm for minimizing the total overflow. This is a concurrent approach considering all the nets simultaneously contrary to the existing approaches of sequential rip-up and reroute. The algorithms are implemented on ISPD98 derived benchmarks and the drastic reduction of overflow is observed.

1. Introduction

The Global Routing Problem (GRP) in VLSI design is a problem of routing a set of nets (multi-net global routing) subject to limited resources and delay constraints. There are various recent approaches for solving GRP [2,5,20,15,18,23] available, but the referred methods are not timing-driven. Most of these modern routers generate Steiner trees with highly optimized wirelength and then use rip-up and reroute iteratively for reducing the congestion. But merely optimizing the wirelength and then minimizing the overflow will not produce a feasible routing because they will not necessarily meet timing at the sinks. A simple example is shown in Fig. 1 to demonstrate that optimum wirelength does not necessarily mean optimum delay and vice versa.

The computation of delay is heavily dependent on pendant subtrees. Therefore, optimizing delay and congestion is a multi-objective constraint, which is the focus of this work. In the example, Fig. 1(a) shows that, given 3 pins, 1(source), 2(sink), and 3(sink), we first draw the Hanan grid by drawing horizontal and vertical lines through them. The intersecting

☆ The reported study was partially supported by RFBR, Russia, research project No. 13-07-00139, and by Basic Science Research Program (NRF-2013R1A1A2064302) through NRF, Korea.

* Corresponding author.
E-mail addresses: samanta@ssl.serc.iisc.in, radhamanjari@gmail.com (R. Samanta), adilerzin@math.nsc.ru (A.I. Erzin), raha@serc.iisc.in (S. Raha), orlab@math.nsc.ru (Y.V. Shamardin), takhonov@gmail.com (I.I. Takhonov), slava@math.nsc.ru (V.V. Zalyubovskiy).

http://dx.doi.org/10.1016/j.amc.2014.11.062
0096-3003/

points are the generated Steiner points. Fig. 1(b) shows the minimum wire-length(WL = 3 units) tree configuration and Fig. 1(c) shows another tree configuration for the net, whose wirelength, WL = 4 units. Now, we compute the delay of the sink nodes for both the trees. In the figure, all $R_i = R$ and all $C_i = C$. Though the tree in Fig. 1(b) has minimum WL(3 units), it has $3RC$ delay whereas tree in Fig. 1(c) has 4 units of WL but $2RC$ delays at the sinks as can be seen from the following equations. For, Fig. 1(b),

$$\text{Delay at node2}, d_{12} = R_1 * C_3 + (R_1 + R_2) * C_2 \simeq 3RC;$$
$$\text{Delay at node3}, d_{13} = R_1 * C_2 + (R_1 + R_3) * C_3 \simeq 3RC;$$

and for, Fig. 1(c),

$$\text{Delay at node2}, d_{12} = (R_1 + R_2) * C_2 \simeq 2RC;$$
$$\text{Delay at node3}, d_{13} = (R_4 + R_3) * C_3 \simeq 2RC.$$

As suggested by Moffitt et al. [21], there is increasing demand of timing-driven routing algorithms and there are not many works focused in this area. There are a few global routing algorithms [14,30,29] based on MVERT [13], which consider timing but they are of complexity $O(n^4)$ (n is the number of sinks), and implemented on nets with small number of sinks. Also GRP was formulated as a multi-commodity flow Problem [12] as well. With each net a certain flow of unit size is associated. Each edge has a flow capacity. With respect to the objective function we may get a min-cost multi-commodity flow problem or concurrent multi-commodity flow problem [27,3,1,10]. Meta-heuristics to solve GRP can be found in Timber-Wolf [8] (simulated annealing), [4] (evolution algorithm), [9] (genetic algorithm) and [31] (tabu search). Fault tolerant routing method for network-on-chip is described in [17].We propose an $O(n^2 \log n)$ method [26] for constructing delay-driven Steiner trees. Another contribution of our work is a Gradient based approach for minimizing the overflow. The novelty of this algorithm is that, it considers all the nets concurrently and provides a fully polynomial time approximation scheme(FPTAS).

In Section 2, the problem is formulated, and Section 3 describes our proposed algorithm MAD (Modified Algorithm of Dijkstra) and its iterative modifications (IMAD). IMAD is applied to create minimum critical delay Steiner trees for each net. History based IMAD described in Section 5 is used to create congestion-aware trees for each net. Also we have used a router FLUTE [7] to generate another set of Steiner candidate trees for each net. Finally a gradient algorithm is used to pick one tree for each net from its candidate set of trees, such that the total congestion/overflow is minimal. The Gradient method is described in Section 4.

Since there are no recent timing-driven router available, we run MAD without Gradient on IBM/ISPD98 benchmarks and this gives us the initial congestion of the chip. Then we run IMAD with Gradient and show how effectively it reduces the congestion. The benchmarks are modified by assigning resistance, capacitance values to the wires. We show that 66.4% trees

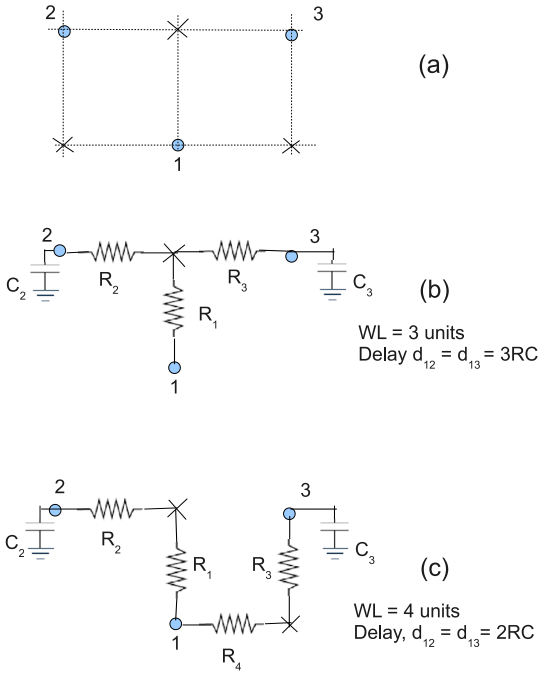

Fig. 1. Minimum wirelength does not necessarily mean minimum delay and vice versa.