Clock Tree Regeneration

Jan-ming Ho *

Institute of Information Sciences
Academia Sinica
Taipei, Taiwan
R. O. C.

Ren-Song Tsay

IBM
Thomas J. Watson Research Center
Yorktown Heights, New York 10598
U. S. A.

Abstract

We present in this paper a clock tree regeneration algorithm for improving both the wirability and performance of VLSI chip designs. After circuit placement, we modify the clock trees originally specified by logic designs utilizing the geometrical information derived from the placement. First, a bipartite bottleneck matching approach is applied to minimize the longest driver to clock pin length. Then a linear assignment approach is used to optimize the total driver-pin length. The experimental results are extremely encouraging.

1 Introduction

Wirability and performance optimization are the main reasons for clock tree regeneration. Usually, this regeneration step is processed during the physical design stage, after circuit placement but before routing phase.

We first define what is a clock tree. A clock tree is a set consisting of a clock source, a number of clock drivers, and a number of clock pins on Flip-Flop's. The clock pins are driven by the intermediate clock drivers, and the drivers are driven by other clock drivers or the clock source. We consider in this paper a clock tree regeneration problem of assigning clock pins to the last level of clock drivers optimally.

The first consideration for clock tree regeneration is the wirability issue. Without circuit placement information, a clock tree specified in a logic design can be quite arbitrary. A typical example is shown in Figure 1(a) and (b) to illustrate the difference of a clock tree before and after optimization. If no optimization is done, the clock tree forms a massive network, which obviously has severe impact on wirability. Thus, one of the criteria for clock tree regeneration is to optimally assign the clock pins to drivers such that the chip is most wirable. As a common practice, this is implemented by minimizing the total wire length.

The other reason for clock tree regeneration is to optimize the circuit performance by reducing clock skew. To minimize the skew of a clock tree, we have to balance the total resistance and capacitance of each group of clock driver and associated clock pins. This criterion is usually implemented by the maximum fan-out constraint, which restricts the number of clock pins that each clock driver can drive.

Another criterion related to the skew problem requires that the delay times from all clock drivers to the associated clock pins have to be as close as possible. This is the most important criterion for clock tree regeneration. It will be implemented by minimizing the longest delay. If we assume all clock drivers and clock pins have the same circuit characteristics, i.e. same driving resistance and/or loading capacitance, then the longest delay is proportional to the longest wire length from clock driver to clock pin. Hence, one optimization objective is to minimize the longest length.

Edahiro [1] has studied a similar problem that minimizes the total spanning tree length subject to a fan-out constraint. Note that the minimization of total spanning tree length is quite different from the minimization of the maximum signal path as demonstrated by the example shown in Figure 2. Instead, as explained earlier, in our approach we will use the longest length as our primary objective and the total wire length of the clock tree as a secondary objective, and all are subject to a fan-out constraint. We will present optimum algorithms for solving these problems.

This paper is organized as follows. We first formally define related problems in the next section. Algorithms for clock tree regeneration with minimum

*This work has been supported in part by the National Science Council, R.O.C., under Grant NSC 79-0404-E-001-01.

0-8186-2610-0/92 $3.00 © 1992 IEEE

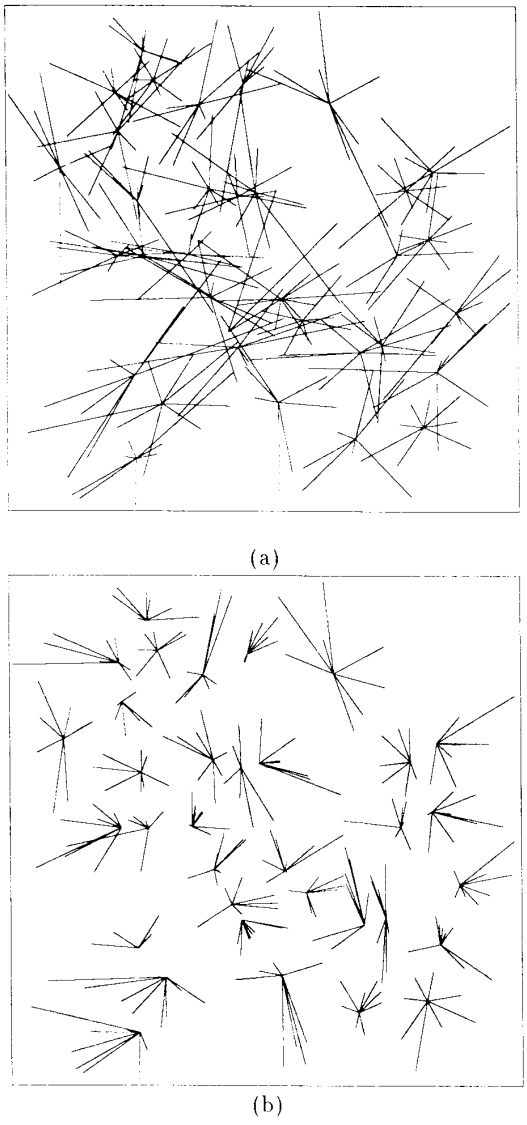

Figure 1: (a) A clock tree specified from a logic design. (b) An optimized clock tree.

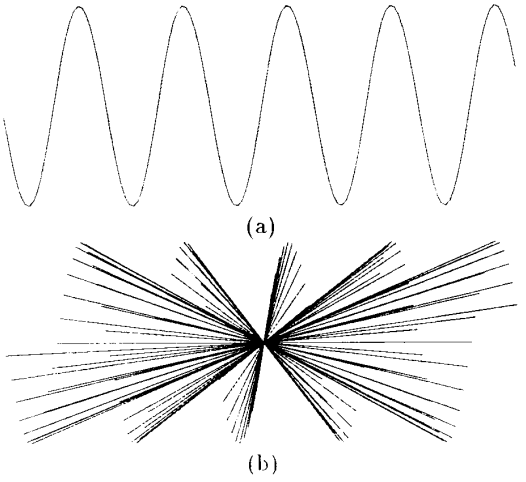

Figure 2: The longest path of the minimum spanning tree (a) of the above set of points is $\sqrt{n}$ times that of the star tree (b) illustrated.

longest length and minimum total length are presented in sections 3 and 4. Experimental results are shown in section 5. Finally, some concluding remarks are given in section 6.

2 Problem Formulation

In this section, we shall give a formal formulation of the clock tree regeneration problem. We assume that the clock tree contains a clock source, a set D of k clock drivers and a set P of n clock pins. Each clock driver redistributes the signal generated by the clock source and drives the subset of clock pins assigned to it. The number of fan-outs of each clock driver is limited by an integer constant $B > 0$. We assume $kB \geq n$, which is the only case the problem is meaningful. For simplicity, we assume that the clock source, drivers, and pins are points on a two-dimensional plane. Note that the algorithms presented in this paper can be extended to clock tree problems with multi-stage clock drivers, problems that have clock pins distributed on higher dimensions and problems that the fan-out bound of each driver differs from others.

We say that the clock tree regeneration problem is the problem of finding an optimum partition of the clock pins P into k disjoint subsets $P_1, \ldots, P_k$, such that $\cup_{i=1}^{k} P_i = P$ and $|P_i| \leq B$. There are two objectives of this optimization problem: the primary objective is to minimize the longest length from clock

drivers to the associated clock pins; and the secondary objective is to minimize the total length.

The distance of a clock pin p to a clock driver d is denoted as $|p-d|$. Define radius r_i as the maximum distance (or the longest length) of the pins in P_i to the driver d_i, or $r_i = max_{p \in P_i} |p - d_i|$. The distance measure can be the Manhattan distance (or L_1 distance) or other measure system.

We divide the clock tree problem into the following two subproblems. The first subproblem is focused on minimizing the maximum radius (the longest length) of the partitions, as required by the primary objective. Then the second subproblem is to minimize the total length with the maximum radius as a constraint. The two subproblems are formally defined as follows.

Problem 1 (Minimize Longest Length) *We will refer to this problem in a more theoretical term as the* bounded fan-out minimum-radius partition problem *or the BMP for short. For this problem, we are given a set $D = \{d_i | 1 \leq i \leq k\}$ of k points called the clock driver vertices, a set $P = \{p_i | 1 \leq i \leq n\}$ of n points called the clock pin vertices, and a constant $B > 0$ for fan-out limit, and $kB \geq n$. The question is to find an optimum partition that divides the set P into k disjoint subsets $P_1, \ldots, P_k \subseteq P$ such that $\cup_{i=1}^{k} P_i = P$, $|P_i| \leq B$, and the objective $\max_{i=1}^{k} r_i$ is minimized.*

Problem 2 (Minimize Total Length) *We will refer to this problem as the* bounded fan-out bounded radius minimum weighted partition problem, *or the BBMP for short. For this problem, we are given a set $D = \{d_i | 1 \leq i \leq k\}$ of k points called the clock driver vertices, a set $P = \{p_i | 1 \leq i \leq n\}$ of n points called the clock pin vertices, constants $B, R^* > 0$, and $kB \geq n$. The question is to find an optimum partition that divides the set P into k disjoint subsets $P_1, \ldots, P_k \subseteq P$ such that $\cup_{i=1}^{k} P_i = P$, $|P_i| \leq B$, $|p - d_i| \leq R^* \ \ \forall p \in P_i$ and the objective $\sum_{i=1}^{k} \sum_{p \in P_i} |p - d_i|$ is minimized. Note that the radius constraint R^* is determined from the solution of the previous problem, BMP.*

In the next section we will present an $O(n^{2.5}\sqrt{\log n})$ time algorithm for solving the first problem, BMP, and an $O(n^3)$ time algorithm for the second problem, BBMP. The total runtime complexity of this clock tree regeneration problem is thus $O(n^3)$.

3 Minimize Longest Length

We propose two techniques for solving the problem BMP, minimization of the longest length with fan-out constraint. One of the techniques utilizes bipartite bottleneck matching algorithm. The other technique uses maximum-flow algorithm, which maybe better when the fan-out limit B is large.

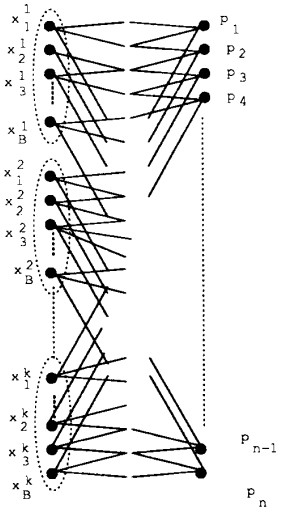

Figure 3: Illustration of the graph G^b.

3.1 A Bipartite Bottleneck Matching Approach

Now, we show how a BMP can be reduced to a bipartite bottleneck matching problem.

We first introduce a few related terms. A *maximum matching* of a graph is a selection of a maximum number of edges in which no two vertices appear more than once. A *bottleneck matching* is a maximum matching whose maximum edge cost is a minimum [2]. A *minimum weighted matching* is a maximum matching whose total edge cost is a minimum. We shall refer to the last two matchings on a bipartite graph as *bipartite bottleneck matching* and *bipartite weighted matching* respectively. A *bipartite graph* is usually denoted as $G = (X, Y, E)$, where $X \cup Y$ is the set of vertices and each edge $e = (x, y)$ in E connects a vertex x in X and a vertex y in Y. Each edge $e \in E$ is associated with a real-valued weight $w(e)$.

To solve BMP, we construct a weighted complete bipartite graph $G^b = (X^b, Y^b, E^b)$ (see Figure 3), where $Y^b = P$, $X^b = \cup_{i=1}^{k} X_i$ and $X_i = \{x_j^i = d_i | 1 \leq j \leq B\}, 1 \leq i \leq k$, in which each driver d_i is duplicated B times. Each edge $e = (x_j^i, p)$ in E^b is given a weight $w(e)$ equals to the distance between the driver d_i and the pin p. Note that the set X^b contains B duplica-

tions of each driver vertex d_i and thus its cardinality is kB. We can prove that the bottleneck matching problem on the bipartite graph G^b given above is equivalent to the BMP.

Theorem 1 *The BMP problem is equivalent to the bipartite bottleneck matching problem on the bipartite graph G^b as defined.*

Proof: First, we show that an optimal solution of the BMP implies an optimal solution of the bipartite bottleneck matching problem. Consider an optimal partition $P_1^*, \ldots, P_k^*$ of the clock pin set P such that the maximum radius (longest length) $r^* = \max_{i=1}^k r_i$ of the BMP achieves its minimum for all possible partition of the set P, where $r_i = \max_{p \in P_i} |p - d_i|$ is the radius of the set P_i with respect to d_i. Let $m_i = |P_i|$, and let the j-th pin in the set P_i be p_j^i. It then follows that the set M of edges is an optimal bottleneck matching of G^b, where $M = \cup_{i=1}^k E_i$ and $E_i = \{(x_j^i, p_j^i) | 1 \leq j \leq m_i \text{ and } p_j^i \text{ is the } j\text{-th pin in } P_i\}$. Note that r^* equals to the maximum edge weight of edges in M.

Assuming otherwise, i.e., there exists an optimal bipartite bottleneck matching M^b of G^b such that the maximum of the weights of the edges in M^b is $r^b < r^*$. Let P_i^b denote the set of vertices in Y^b matching vertices in X_i. Obviously, $|P_i^b| \leq |X_i| = B$. Then $\{P_i^b | 1 \leq i \leq k\}$ is a feasible partition of P with $\max_{i=1}^k r_i = r^b < r^*$. Which contradicts the fact that r^* is an optimal solution of the BMP.

Similarly, it can be shown that an optimal solution of the bipartite bottleneck matching problem also implies an optimal solution of the BMP. Note that the BMP problem does not have a solution if and only if there are some vertices in Y^b which are not covered by the bipartite bottleneck matching. It is not difficult to prove that this case will not occur and the proof is left for interested readers.

◇

Note that the best known algorithm for the bipartite bottleneck matching problem takes time $O((|V| \log |V|)^{1/2}|E|)$ [2], where $|V|$ denotes the number of vertices of the bipartite graph G^b and $|E|$ denotes the number of edges of G^b. Since the reductions as introduced above takes $O(Bkn)$ time, $|V| = kB + n$ and $|E| = kBn$, the BMP is solvable in $O(Bkn\sqrt{(kB+n)\log(kB+n)})$ time.

Lemma 1 *The BMP is solvable in $O(Bkn\sqrt{(kB+n)\log(kB+n)})$ time.*

For small constant value B, and thus $k = O(n)$, this complexity is also given as $O(n^{2.5}\sqrt{\log n})$, while it becomes $O(n^{3.5}\sqrt{\log n})$ if both B and k are $O(n)$.

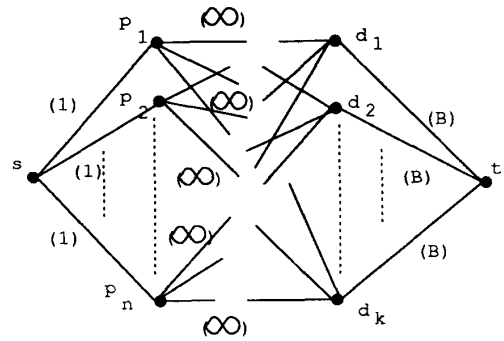

Figure 4: Illustration of the maximum-flow model.

3.2 A Maximum-Flow Approach

In the following, we propose a second technique for solving the BMP based on the maximum-flow algorithm. Let us consider a decision version of the BMP. Given a fan-out bound $B > 0$ and a radius bound $R > 0$, the decision problem is to find whether there exists a partition $\Psi = \{P_i | 1 \leq i \leq k\}$ of P such that $P_i \cap P_j = \phi$ for $i \neq j$, $\cup_{i=1}^k P_i = P$, and both the fan-out bound and the radius bound are satisfied, i.e., $|P_i| \leq B$ and $|p - d_i| \leq R, \forall p \in P_i, 1 \leq i \leq k$. This problem is called the *bounded fan-out bounded-radius partition problem* (BBP for short).

The idea for solving the BBP is the following. We first identify a set P_i' of the pin vertices which are within a distance R from a driver vertex d_i. If there exists a vertex p which does not belong to any set $P_i', 1 \leq i \leq k$, then we immediately conclude that the BBP is unsolvable. On the other hand, if every pin vertex belongs to exactly one of the sets P_i' and $|P_i'| \leq B$, then the problem is solved. Otherwise, we have to carefully assign each pin vertex to a driver vertex within the distance R, subject to the fan-out constraint. This decision problem can be solved using a maximum-flow algorithm as described in the following.

In the maximum-flow model (see Figure 4), four types of vertices are used: a source s, a destination t, a set of vertices D corresponding to the clock drivers, and a set of vertices P corresponding to the clock pins. The source s connects to each pin vertex p_j by an directed edge with a capacity 1. Pin vertex p_j connects to each driver vertex d_i by an edge of unbounded capacity if the distance between p_j and d_i is no greater than R. Note that, the set of pin vertices connecting to d_j is exactly the set of clock pins that can be assigned to the clock driver d_i. By assigning capacity 1

to each edge (s, p_j), it guarantees that every clock pin is assigned to at most one clock driver.

The third class of directed edges connect each vertex d_i to the sink vertex t, with capacities B. This arrangement is made to guarantee that no more than B pin vertices are associated with a driver vertex d_i. It is obvious that the BBP is solvable if and only if the corresponding maximum-flow problem described above has a maximum flow of n, which means no clock pins can be left unassigned.

This problem has an interesting monotonic property. Note that as we increase the radius bound R of this decision problem, the only change to the maximum-flow model is the addition of edges. In other words, there is a threshold value R^* such that the corresponding BBP has a solution if $R \geq R^*$, and has no solution if $R < R^*$. This property enables us for an efficient binary search for this threshold value R^*. It can be easily shown that R^* is exactly the solution of BMP, i.e. R^* is the minimum longest length.

Since R^* is one of the kn possible combinations of pin-driver pairs, the BMP can be solved in the following way. First, the distances of all driver-pin pairs are computed and sorted into non-decreasing order, and the sorted array is denoted as A. Then we perform a binary search in the array A for an optimal solution of the BMP. At each step, the current array element $A[q]$ is used as the radius bound of the BBP and the feasibility of BBP is tested. Note that it takes $O(kn)$ time to transform BBP to a maximum-flow model. The best known maximum-flow algorithm runs in $O(|V|^3)$ time [4], where $|V|$ denotes the number of vertices in the model and equals $k + n$ in the BBP. As a result, this algorithm takes $O((k+n)^3 \log n)$ time. Note that the complexity of this algorithm does not depend on the value of the fan-out bound B. Summarizing the results of this section, we have the following theorem:

Theorem 2 *The problem BMP can be solved in* $O(\min\{Bkn\sqrt{(kB+n)\log(kB+n)}, (k+n)^3 \log n\})$ *time.*

4 Minimize Total Length

The problem BBMP, minimization of total length with fan-out and maximum radius constraint, can be reduced to a bipartite weighted matching problem. We construct a bipartite graph $G^w = (X^w, Y^w, E^w)$ as follows. The vertex sets X^w and Y^w are defined the same way as X^b and Y^b in the bipartite graph $G^b = (X^b, Y^b, E^b)$ from last section. Let the edge weight of (x, y) represent the distance between vertex x (a clock driver) and vertex y (a clock pin). The edge set E^w contains only the edges in E^b whose weights are no larger than R^*. By an argument similar to that we used in the proof of Theorem 1, the equivalence between the BBMP and the bipartite weighted matching problem on G^w is established.

Theorem 3 *The BBMP is equivalent to the weighted matching problem on the bipartite graph* G^w *as defined above.*

Proof: Omitted.

The bipartite weighted matching problem is also known as the linear assignment problem. It can be formulated as a linear programming problem and thus solved by the primal-dual method, in this particular case, called the Hungarian method. Time complexity of the Hungarian method is $O(|V|^3)$ [4], where $|V|$ denotes the total number of vertices in the bipartite graph. The reduction of the BBMP to the bipartite weighted matching problem as introduced above takes $O(Bkn)$ time and the bipartite weighted matching algorithm runs in $O((kB + n)^3)$. The BBMP can thus be solved in $O((kB + n)^3)$ time.

Lemma 2 *The BBMP is solvable in* $O((kB + n)^3)$ *time.*

In conclusion, we have the following theorem.

Theorem 4 *The clock tree regeneration can be solved in*

$$O(\min\{Bkn\sqrt{(kB+n)\log(kB+n)}, (k+n)^3 \log n\} + (kB+n)^3)$$

time.

Note that $kB = O(n)$ in practice. Thus, the time complexity of BMP is approximately $O(n^{2.5}\sqrt{\log n})$ and that of the BBMP is $O(n^3)$. Thus the total time complexity of the tree regeneration problem is approximately $O(n^3)$. But, in the worst case $k = B = O(n)$, thus the complexity of our clock tree regeneration problem becomes $O(n^3 \log n)$.

5 Experimental Results

We tested the algorithms on various sized examples. The statistics of the examples are listed in Table 1. The units of chip width and height are both in 1/10 microns. The locations of the latches (clock pins) and the clock drivers are determined along with other circuits by a placement program which takes whatever net list specified from the logic designs. Then we optimized the maximum radius and total length of each

Example	r1	r2	r3	r4	R1	R2	R3
No. clock drivers	34	75	108	191	25	100	92
No. clock pins	267	598	862	1903	100	400	1840
fan-out limit	8	8	8	10	4	4	20
chip width	69984	94016	97000	126970	59112	100000	122500
chip height	70000	93134	98500	126988	59700	100000	121000

Table 1: The statistics of the testing examples.

Example	r1	r2	r3	r4	R1	R2	R3
initial max radius	30235	51222	43560	73527	89313	155280	24873
initial total length	2811126	8393133	12536291	39686765	4161540	26103240	22899321
final max radius	18464	19658	26701	38390	21257	18150	23354
improvement	63.8%	160.6%	63.1%	91.5%	320.2%	755.5%	6.5%
final total length	1658169	4025115	5376763	13940534	929714	3468900	11730929
improvement	69.5%	108.5%	133.2%	184.7%	347.7%	652.5%	95.2%
runtime(s)	73	732	2137	25180	5	345	18673

Table 2: A comparison between the initial clock trees from logic designs and the clock trees after optimization.

example using the algorithms described in this paper. Depending on how well the logical information was matched with the physical information, we have improvements on maximum radii ranging from 6.5% up to 755.5%, and have improvements on the total lengths ranging from 69.5% up to 652.2%. It is needless to say that the results help to improve the wirability and performance of all examples tremendously. The two clock trees of the example r1 before and after optimization are illustrated in Figure 1(a) and (b).

6 Conclusions

After clock tree regeneration, a special clock routing algorithm should follow up to actually interconnect each clock driver to the clock pins assigned to it. To maintain the objective, minimum longest length, achieved by the tree regeneration step, we may use a *rectilinear multicast tree* (RMT) algorithm [3] to connect each pin group $P_i \cup \{d_i\}$. The rectilinear multicast tree of the set of clock pins P_i and clock driver d_i is a minimum length Steiner tree with longest length constraint. The constraint requires that the Steiner wire length of each clock pin to the clock driver has to be smaller than a given number R^*, for example, as can be decided from BMP.

References

[1] M. Edahiro. A clock net reassignment algorithm using Voronoi diagram. In *IEEE International Conference on Computer-Aided Design, Degist of Technical Papers*, pages 420–423, November 1990.

[2] H.N. Gabow and R.E. Tarjan. Algorithms for two bottleneck optimization problems. *Journal of Algorithms*, 9:411–417, 1988.

[3] J.-M. Ho, M.T. Ko, and T.Y. Sung. Optimal multicast trees. in preparation.

[4] C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Inc., 1982. p. 247.