Adaptive Wire Adjustment for Bounded Skew Clock Distribution Network

H. Saaied, D. Al-Khalili[†]*, A. Al-Khalili and M. Nekili*

Concordia University, Montreal, CANADA,[†]Royal Military College, Kingston, CANADA
haydar@ece.concordia.ca,alkhalili-d@rmc.ca,asim@ece.concordia.ca,mnekili@ece.concordia.ca

Abstract - In this paper, we suggest an adaptive approach for the Clock Distribution Network (CDN) to cope with a modification in the VLSI system design. The CDN's wires are adjusted iteratively to reduce the skew that is resulting from a minor modification in the clock pins of a complex VLSI system. Such skew can be remedied by selecting a Balancing Node (BN) and adjust its edges so that the skew gets smaller. The required edge adjustments are determined using the Elmore delay model. The performance of the algorithm is investigated using different random sets of clock pins. Also, the algorithm is tested by altering some clock pins in a zero skew CDN. For small modifications in a large number of nodes in the CDN, our algorithm can achieve zero skew with less iterations than linear order algorithms.

I. Introduction

The growth in System on Chip (**SoC**) complexity presents a new challenge to the design of Clock Distribution Network (**CDN**). The challenge stems from the decrease in the gate delay compared to the increase of the interconnection delay [2]. Sylvester et al. showed that the delays on interconnects that span the chip will extend longer than the clock period [3], and hence, controlling the skew will be difficult. In addition to that, final locations of clock pins cannot be known till later in the design process. Thus, the time closure problem of SoC will be more difficult due to the unavoidable need for the CDN modification to satisfy the system skew requirements. Last decade witnessed many approaches that tackle the skew problem by devising CDN to deliver the clock signal with zero skew (**ZSCDN**) or bounded skew (**BSCDN**) [1]. However, an Engineering Change (**EC**) in the locations or the load capacitance of the clock pins would require a recalculation of the whole CDN solution as in Deferred Merging Embedded algorithm (**DME**) [1]. In fact, it is often that IPs are inserted or removed, and that would change clock pins placement, capacitance and CDN's topology. Moreover, the IP cores have different clock delays, and each IP should be treated as a black box. The multiple redesigning would become very painful with pressing of shortest time to market. Presence of minor modifications in SoC design process mountains the need for an incremental algorithm to tune the CDN in order to eliminate repetitive redesign of the CDN. We propose a novel algorithm, called Adaptive Wire Adjustment (**AWA**), with a main goal to enable a quick EC into a CDN that capable of integrating IP cores into SoC. A secondary goal is to reduce the CDN wire length to help with routing and power consumption. The reminder of the paper is organized as follows. Section 2 reviews some preliminaries and Elmore delay. Section 3 presents the proposed algorithm. Section 4 proves the convergence of the algorithm. Section 5 presents experimental results supporting the effectiveness of the proposed algorithm for small modifications.

II. Preliminary

A. Definitions

Consider a CDN that connects a set of clock pins R={1, 2,., N}, for example, see Figure-1(a). Each clock pin $i \in$ R, is associated with a capacitance load c_i and a location $l(r_i)$ in the Manhattan plane. If there is a unique path between any two nodes in the CDN, then the CDN can be represented by a tree, ***T***, as shown in Figure-1(b). Any node $v \in$ ***T*** is connected to its parent by an edge e_v and the cost of e_v is its rectilinear length in the Manhattan plane. The rectilinear distance between two points in the Manhattan plane, say *u* and *v*, is denoted ***Rect(u,v).*** For any two nodes *w,v*$\in$ ***T*** where *w* is the ancestor of *v*, let ***Path(w,v)*** be the unique path from *w* to *v* in ***T***.

B. Delay Model

The clock signal delay along a path in ***T*** can be determined under Elmore model by modeling each edge of length *e* as a π-type circuit with a resistor er_0 and two capacitors $ec_0/2$, where r_0 and c_0 are the resistance and capacitance per unit length of the interconnect respectively. Using this model, the tree ***T*** can be modeled as an RC tree as shown in Figure-1(c), and, the signal delay

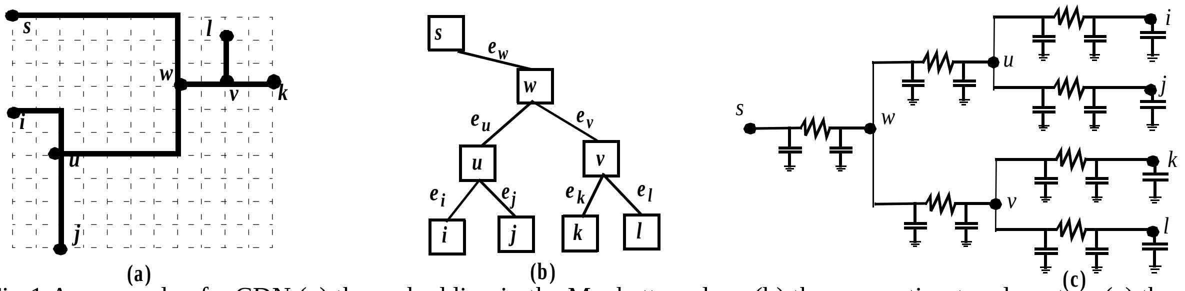

Fig 1 An example of a CDN (a) the embedding in the Manhattan plane (b) the connection topology tree (c) the RC-Tree model

from node w to node v (w is the ancestor of v) is given by [4]:

$$\mathrm{Delay}(w, v) = \sum_{e_\alpha \in \mathrm{Path}(w,v)} e_\alpha r_0\left(\frac{e_\alpha c_0}{2} + C_\alpha\right) \tag{1}$$

where, C_α is the total capacitance of the sub-tree rooted at node α

The skew between two leaves in T, say i and k, can be calculated as following:

$$s(i, k) = \mathrm{Delay}(s_0, i) - \mathrm{Delay}(s_0, k) \tag{2}$$

where, s_0 is the root of the tree T.

C. Problem Definition

The skew constraint on any pair of clock pins, $i, k \in$ R, is given as the minimum, $s_{min}(i,k)$, and maximum, $s_{max}(i,k)$. If the skew between a pair of leaves is out of the permissible skew range, then ***T*** must be modified, otherwise, there will be a failure in the system. The problem can be formulated as follows: for any pair of leaves $i,k \in \boldsymbol{T}$, adjust edges of ***T*** so that $s_{\min}(i,k) < s(i,k) < s_{\max}(i,k)$. A common approach to control the skew is to design a BSCDN with the assumption that a certain bounded skew, B, between all clock pins would satisfy the skew requirements. In this paper, we present a novel approach that can minimize the skew to any required bound B for an initial CDN by adjusting wires of the CDN iteratively. The proposed algorithm is called adaptive wire adjustment.

III. Adaptive Wire Adjustment Algorithm

The Adaptive Wire Adjustment (AWA) algorithm takes as an input a CDN in a form of a linked tree data structure and checks for the maximum skew in the tree then minimizes it by adjusting wires of the tree. Whenever a skew exists between two leaves, the paths to these two leaves must be adjusted so that the skew becomes smaller. To adjust these two paths, the algorithm finds the node that can be shifted, by adjusting its child edges, so that the skew is minimized. Such a node, called the **Balancing Node**, **BN**, is the first ancestor of the same two leaves. For example consider the CDN shown in Figure-1, to minimize the skew between leaves i and k, their BN, w, is shifted by adjusting its child edges e_u and e_v. If the leaf i suffers higher delay than leaf k, then w must be shifted by Δ towards node u and away from node v. Thus, the edge e_u decreases and edge e_v increases as following:

$$e'_u = e_u - \Delta \quad , \quad e'_v = e_v + \Delta \tag{3}$$

where, e'_u and e'_v are the new value of e_u and e_v.

In order to determine the edge adjustment, Δ, the delay from node w to its leaves have to be considered. In fact, the leaf delays of the subtrees rooted at u and v span ranges in the time domain as shown in Figure-2 (leaf delays are calculated from the root of the tree). Thus, to minimize the skew between the two sub-trees, the ranges of u and v must be shifted in opposite directions. Let us define the range of a sub-tree rooted at a node, say α, by the maximum and the minimum delays in that range calculated from the node α as following:

$$Max_\alpha = \mathrm{Maximum}\{\mathrm{Delay}(\alpha,\beta)\} \quad \text{for all leaf } \beta \in \text{ subtree } \alpha$$

$$Min_\alpha = \mathrm{Minimum}\{\mathrm{Delay}(\alpha,\beta)\} \quad \text{for all leaf } \beta \in \text{ subtree } \alpha \tag{4}$$

Also, let us define the **center of the range** of a sub-tree rooted at α, $\boldsymbol{Centre_\alpha}$, to be the average of maximum and minimum delays of that range:

$$\mathrm{Centre}_\alpha = \frac{\mathrm{Max}_\alpha + \mathrm{Min}_\alpha}{2} \tag{5}$$

Using the range centre as a time reference, the skew between a pair of internal nodes, u and v, can be defined as:

$$S(u, v) = \left[e_u r_0\left(\frac{e_u c_0}{2} + C_u\right) + \mathrm{Centre}_u\right] - \left[e_v r_0\left(\frac{e_v c_0}{2} + C_v\right) + \mathrm{Centre}_v\right] \tag{6}$$

Equation-6 can be used to determine a BN's edge adjustment, Δ, so that the skew between the BN's children, u and v, becomes zero by substituting e_u and e_v by e'_u and e'_v respectively in Equation-6 and equating it to zero as following:

$$\Delta = \frac{\frac{r_0 c_0}{2}\left(e_u^2 - e_v^2\right) + (\mathrm{Centre}_u - \mathrm{Centre}_v) + r_0(e_u C_u - e_v C_v)}{r_0(c_0(e_u + e_v) + C_u + C_v)} \tag{7}$$

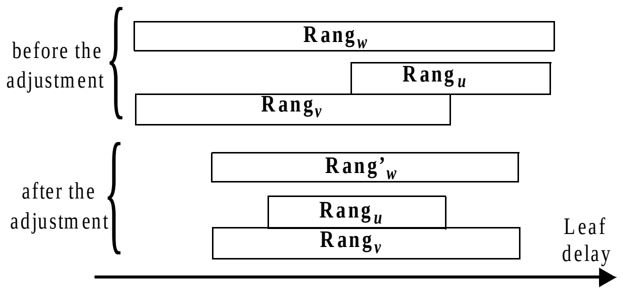

Fig 2 The leaf delays ranges of the sub-trees rooted at nodes u, v and w in Fig. 1 (leaf delays are calculated from the root of the tree).

Adjusting e_u and e_v by Δ would shift the BN, where the possible locations of the BN form a segment that will be called the **Balancing Segment, *BS***. The BS of u and v, $\boldsymbol{BS_{uv}}$, can be determined from the intersection of two tilted rectangles, $\mathbf{TR_u}$ and $\mathbf{TR_v}$, whose centers are nodes u and v and whose radii are e'_u and e'_v respectively as shown in Figure-3(a). Note that this is true only when $e'_u + e'_v = \boldsymbol{Rect_{uv}}$. If $e'_u + e'_v > \boldsymbol{Rect_{uv}}$, then the intersection of $\mathbf{TR_u}$ and $\mathbf{TR_v}$ is a **TR** as shown in Figure-3(b). In such a case, it is possible to shorten both e'_u and e'_v while minimizing the skew between u and v. The final location of the BN, w, is determined by the intersection of $\mathbf{BS_{uv}}$ and $\mathbf{TR_s}$ whose centre is s and its radius is the BN edge e_w as shown in Figure-3(a). The BN's edge, e_w, has not been considered yet, and its length would affect the total wire length. In fact, the total wire length can be minimized by determining the BN's edge as the minimum length between the BS and the BN's parent.

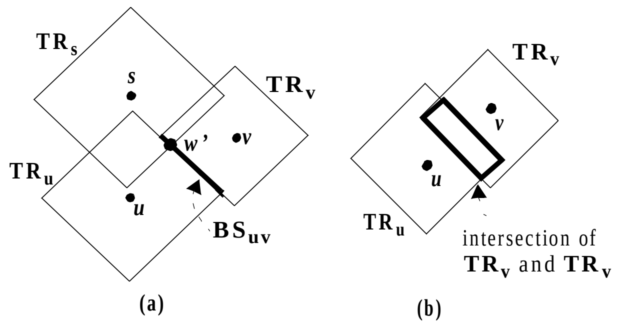

Fig 3 (a) Determination of $\mathbf{BS_{uv}}$ and the new location of the **BN** w. (b) The intersection of $\mathbf{TR_u}$ and $\mathbf{TR_v}$ is a **TR** also when $e_u + e_v > \boldsymbol{Rect_{uv}}$

Adjusting the BN's child edges, as described before, would minimize the maximum skew between the BN's leaves. In fact, the BN's range gets smaller as the child ranges are shifted in opposite directions so that their centres coincide on each other as shown in Figure-2. However, if the centres of the child ranges are already coincide, then it is impossible to shorten the BN's range by adjusting its child edges. Also, if the BN's range is equal to one of its child ranges, then shifting the child ranges would not help to minimize the BN's range. In such a case, the BN's child that have greater range have to be selected as a BN. In general, the BN can be defined as the node that have the maximum skew and belongs to the lowest level in the tree. From previous discussion, we can draw the AWA algorithm to achieve a bounded skew for a given tree by minimizing the maximum skew in the tree iteratively. At each iteration, a BN is selected and its child edges are adjusted as in Equations-3-7. The formal description of AWA algorithm is as following:

```
Input:  Initial CDN for a set of clock pins R
Output:  Bounded Skew CDN by B
Generate T
Procedure AWA
while maximum skew >B:
    Find the BN that has the maximum skew in T, and it
        belongs to the lowest level in T
    Calculate the required adjustment, $\Delta$, to balance the
        child nodes of the BN, u and v.
    Construct $TR_u$ and $TR_v$ as following:
        core($TR_u$) = location of u, radius(TR )=$e'_u$
        core($TR_v$) = location of v, radius($TR_v$)=$e'_v$
    Calculate the $BS_{uv}$ as $TR_u \cap TR_v$
    Calculate $e'_w$ as the minimum distance between $BS_{uv}$
        and parent of BN, s.
    Construct $TR_s$ as following:
        core($TR_s$) = location of s, radius($TR_s$)=$e'_w$
    Locate BN at the intersection of $TR_s$ with $BS_{uv}$
    Find the maximum skew in T
```

The AWA algorithm can narrow the skew in a given tree iteratively, and hence the whole solution needs not to be recalculated when the skew needs to be smaller. In fact, the iterative nature of the AWA differentiates it from other algorithms, such as Tsay approach or DME algorithm. This feature of AWA is important when there is a minor modification in the clock pins during the design process, especially if the number of the clock pins is huge. Furthermore, the AWA can save the calculation time required for CDN determination while managing a small modification during the back and forth between different steps of digital system design process.

Unfortunately, minimizing the BN's range not necessarily leads immediately to a smaller skew in the tree due to the alteration in the skew between the leaves that belong to the BN and other leaves in the whole tree. However, selecting an ancestor node as a BN in later iterations will ensure to minimize the skew further, and AWA can achieve the desired bound of skew ultimately as will described in the next section.

IV. Convergence of AWA Algorithm

As described before, shifting the child ranges would not help to minimize the BN's range if the centres of the child ranges are already coincide, or if the BN's range is equal to one of its child ranges. This can be stated as following:

Lemma 1: Let $Rang_u$, $Rang_v$ and $Rang_w$ be the ranges of u, v and w respectively such that $Rang_w > Rang_u$, and $Rang_w > Rang_v$, where u and v are the children of w. If $Centre_u$ and $Centre_v$ are not coincided on each other, then shifting $Rang_u$ and $Rang_v$ in opposite direction so that $Centre_u$ and $Centre_v$ get closer would result in $Rang'_w < Rang_w$, where $Rang'_w$ is the range of w after the shifting.

Proof: Let D and D' be the difference between $\textbf{\textit{Centre}}_u$ and $\textbf{\textit{Centre}}_v$ before and after shifting the ranges respectively. Shifting $\textbf{\textit{Rang}}_u$ and $\textbf{\textit{Rang}}_v$ so that $\textbf{\textit{Centre}}_u$ and $\textbf{\textit{Centre}}_v$ get closer implicitly gives D'<D. Thus, the case that $\textbf{\textit{Rang'}}_w>\textbf{\textit{Rang}}_w$ is impossible as it implies that D'>D. Furthermore, the case that D'<D but $\textbf{\textit{Rang'}}_w=\textbf{\textit{Rang}}_w$ would happen only if $\textbf{\textit{Rang}}_w$ is equal to either $\textbf{\textit{Rang}}_u$ or $\textbf{\textit{Rang}}_v$, which is contradictory to the statement of the Lemma. Hence $\textbf{\textit{Rang'}}_w<\textbf{\textit{Rang}}_w$ must be true.

Indeed, the minimum range of a BN can be achieved only by shifting the two BN's child ranges in opposite directions till their centres coincide as stated below:

Lemma 2: Let $\textbf{\textit{Rang}}_u$, $\textbf{\textit{Rang}}_v$ *and* $\textbf{\textit{Rang}}_w$ *be the ranges of u, v and w respectively, such that u and v are the children of w. If* $\textbf{\textit{Centre}}_u$ *and* $\textbf{\textit{Centre}}_v$ *coincide on each other, then* $\textbf{\textit{Rang}}_w=\textit{Maximum}(\textbf{\textit{Rang}}_u,\textbf{\textit{Rang}}_v)$.

Proof: Let $\textbf{\textit{Rang}}_u>\textbf{\textit{Rang}}_v$. The case that $\textbf{\textit{Rang}}_w>\textbf{\textit{Rang}}_u$ implies that either $\textbf{\textit{Rang}}_u<\textbf{\textit{Rang}}_v$ or $\textbf{\textit{Centre}}_u$ and $\textbf{\textit{Centre}}_v$ have not coincided on each other, which contradicts the given statement. Also, it is impossible that $\textbf{\textit{Rang}}_w<\textbf{\textit{Rang}}_u$ since *u* is the child of *w*. Hence $\textbf{\textit{Rang}}_w=\textbf{\textit{Rang}}_u$ must be true. A similar argument can be deduced when $\textbf{\textit{Rang}}_v>\textbf{\textit{Rang}}_u$.

Previous discussion is centered on getting the BN's range smaller, which implies the minimization of the maximum skew between the BN's leaves. Unfortunately, getting the BN's range smaller may result in a greater skew between the BN's leaves and other leaves in the tree as described before. However, minimizing the range of a BN in a tree will lead to minimize the range of the BN's parent in a later adjustment as stated in the followings two lemmas:

Lemma 3: Let $\textbf{\textit{Rang}}_w$ *be the range of w when its child ranges are* $\textbf{\textit{Rang}}_u$ *and* $\textbf{\textit{Rang}}_v$*, and they are coincided on each other. Also, let* $\textbf{\textit{Rang'}}_w$ *be the range of w when its child ranges are* $\textbf{\textit{Rang'}}_u$ *and* $\textbf{\textit{Rang'}}_v$*, and they are coincided on each other. If* $\textbf{\textit{Rang'}}_u<\textbf{\textit{Rang}}_u$ *and* $\textbf{\textit{Rang'}}_v<\textbf{\textit{Rang}}_v$ *then* $\textbf{\textit{Rang'}}_w<\textbf{\textit{Rang}}_w$.

Proof: Since the child ranges are coincided on each other, then, from Lemma 2, $\textbf{\textit{Rang}}_w=\text{Maximum}(\textbf{\textit{Rang}}_u,\textbf{\textit{Rang}}_v)$ and $\textbf{\textit{Rang'}}_w=\text{Maximum}(\textbf{\textit{Rang'}}_u,\textbf{\textit{Rang'}}_v)$. Thus, $\textbf{\textit{Rang'}}_w<\textbf{\textit{Rang}}_w$, since $\textbf{\textit{Rang'}}_u<\textbf{\textit{Rang}}_u$ and $\textbf{\textit{Rang'}}_v<\textbf{\textit{Rang}}_v$.

Lemma 4: Let u and v be the children of w, and the child edges of u and v are adjusted separately such that $\textbf{\textit{Rang'}}_u<\textbf{\textit{Rang}}_u$ *and* $\textbf{\textit{Rang'}}_v<\textbf{\textit{Rang}}_v$*, where* $\textbf{\textit{Rang'}}_u$ *and* $\textbf{\textit{Rang'}}_v$ *are the ranges of u and v after the adjustments respectively. If the new range of w,* $\textbf{\textit{Rang'}}_w$*, increases due to the previous adjustments, then the node w can be selected as the BN to adjust its child edges so that* $\textbf{\textit{Rang''}}_w<\textbf{\textit{Rang}}_w$*, where* $\textbf{\textit{Rang''}}_w$ *is the range of w after adjusting child edges of w.*

Proof: In this statement, there are three individual adjustments such that the adjustments for nodes *u* and *v* lead the adjustment for node *w*. Before any adjustment, and according to Lemma 2, the minimum possible value of $\textbf{\textit{Rang}}_w$ is $\text{Maximum}(\textbf{\textit{Rang}}_u,\textbf{\textit{Rang}}_v)$ which would happen when $\textbf{\textit{Rang}}_u$ and $\textbf{\textit{Rang}}_v$ are coincided on each other. If the last adjustment, for node *w*, shifts the child ranges of *w*, $\textbf{\textit{Rang'}}_u$ and $\textbf{\textit{Rang'}}_v$, till they coincide on each other, then $\textbf{\textit{Rang''}}_w=\text{Maximum}(\textbf{\textit{Rang'}}_u,\textbf{\textit{Rang'}}_v)$ according to lemma 2. Hence, according to Lemma 3, $\textbf{\textit{Rang''}}_w$ will be less than the minimum possible value of $\textbf{\textit{Rang}}_w$ since $\textbf{\textit{Rang'}}_u<\textbf{\textit{Rang}}_u$ and $\textbf{\textit{Rang'}}_v<\textbf{\textit{Rang}}_v$.

Note that the condition that the BN belongs to the lowest level is intended to satisfy Lemma 1 as the BN's range would not equal to any of its child ranges. Further, the fact that BN's range is greater than its child ranges implies that the child centres are not coincide on each other. Also, note that Lemma 4 implies that the maximum skew in the tree would bounce, and hence it would slow down the convergence of the algorithm. However, the algorithm would converge ultimately to the required bounded skew as stated in the following theorem

Theorem: applying the AWA algorithm to any given tree of a CDN would ultimately minimize the maximum skew to any bound.

Proof: The algorithm, at each iteration, selects a BN that have the maximum skew in the tree. According to Lemma 1 and Lemma 2, adjusting the BN's child edges would minimize the maximum skew in the subtree rooted at the BN. If the adjustment results in a greater skew in the whole tree, then at a later iteration, an ancestor of that BN will be selected as the BN and the new maximum skew will get smaller according to Lemma 4. Getting a greater skew would stop when the BN is the root of the tree according to Lemma 1. This can be visualize as the BN bouncing between different levels of the CDN's tree.

Such bouncing results in spikes in the convergence of AWA due to the bouncing in the maximum skew of the whole tree **T**. The maximum skew in **T** can be a result of any combination of two leaves, where N leaves have N(N-

1)/2 combinations. As a tree of N leaves has $\log_2 N$ levels, the BN may bounce $\log_2 N$ times for each maximum skew in **T**. Thus, the order of AWA is $N^2 \log_2 N$ which is not linear as the order of DME.

V. Results

A package has been built using C++ in Windows environment to study the performance of the proposed algorithm. Various benchmarks were tested for various scenarios to provide assessment of the proposed algorithm such as the number of iterations.

Each set is generated by distributing N clock pins randomly on a Manhattan plane using a uniform random function. The capacitance of each clock pin is set between 2fF and 3fF by the same uniform random function. The resistance and the capacitance of the interconnect are assumed to be 300 $\Omega.cm^{-1}$ and 2.6 $pF.cm^{-1}$ respectively. The connection topology of each set is generated by the Method of Means and Median (MMM), where the leaves stand for the clock pins [1]. Each node in the tree is represented by a tuple of the following 8 parameters: *x*,*y* coordinate of the node in the Manhattan plane, edge and detour length that connect the node to its parent, the capacitance of the subtree connected to the node, two pointers to the left and right children and the clock signal delay from the root to the node.

Initial CDNs were generated by MMM for three random sets of 1024, 4096 and 8192 clock pins, and each set was used as an input for the AWA to generate ZSCDN by setting B=0. Figure-3 shows the skew convergence for 1024 clock pins (or leaves). It is obvious that the number of required iterations decreases as the required skew bound, B, increases. The spikes shown in Figure-3 are resulting from the bouncing of the BN between levels of T for each maximum skew. Figure-4 shows the relationship between the number of leaves for different random sets and the number of required iteration to reach ZS. Though the order of AWA is not linear as the order of DME, the main advantage of AWA is its iterative approach of minimizing the skew whereas a small modification in the input data would not require the recalculation of the whole solution as in DME. This feature is important when size of T is very large. To show this advantage, different ZSCDNs of size N=1024,4096 and 8192 where tested by altering leaf locations then tested with AWA. The bounded skew was set to zero, B=0, in order to compare number of the iterations to the DME required iterations, where DME is a linear zero skew algorithm. As one would expect, the number of iterations increases as the number of altered leaves increases. Figure-5 shows the relationship between the number of iterations and number of altered leaves for the mentioned sets. The number of iterations for each number of altered leaves is the average of numbers of the iterations, where each number of iteration corresponds to altering different leaves but of the same number of leaves. The alteration in the leaf locations was 6.25% of the whole Manhattan plane. The benefit of AWA is elevated as the size of the tree increases. For example, when the number of leaves N=1024, and number of altered leaves is 9, the AWA's iteration is 1693 while using DME requires 1024 iterations only. However, when the number of leaves is N=8192, and number of altered leaves is 9, the AWA's iteration is 2410 while using DME requires 8192 iterations.

Finally, the sensitivity of the AWA for the amount of alteration in the leaf location was tested. Again, three sets of ZSCDN of sizes N=1024, 4096 and 8192 were tested by altering location of four leaves by different amounts. Also, for more accurate results, different leaves were selected for alteration for each set of ZSCDN. Figure-6 shows the relationship between the amount of alteration in the leaves location and the required iterations to achieve ZS. From Figure-5 and Figure-6, it is obvious that the number of iterations is less sensitive to the alteration value than to the number of altered leaves.

VI. Conclusion

The topic of small modification of the CDN is an important concept due to the increase in the complexity of synchronous systems and the need for an algorithm to cope with small modification in CDN. In this paper, we showed that whenever there is a skew between any two clock pins, the proposed algorithm can adjust wires of the CDN and relocate the branching points so that the skew becomes smaller. Zero skew sometimes is an over constraint; and the algorithm can achieve any skew bound and it would take less iterations as the skew bound, B, increases. Simulation performed on CDNs have shown that using AWA algorithm can:

(vii) reduce the bounded skew requirement further without recalculating the whole solution.

(viii) achieve ZS for a given initial CDN.

(ix) achieve ZS for a modified ZSCDN at less computation compare to linear methods

The AWA algorithm can identify the node that requires a long wire elongation to minimize the skew. This feature can help to guide future works based on topology modification. Another direction for the future work is to modify the AWA with objective of minimizing the perturbation to some existing edges in the CDN by adjusting other edges.

References

[1] A. Kahng and G. Robins "On Optimal Interconnections for VLSI", Kluwer Academic Publishers 1995

[2] E. Friedman, "Clock Distribution Network in Synchronous Digital Integrated Circuits", Proceedings of the IEEE, vol. 89, NO. 5, May 2001, pp. 665-692.

[3] D. Sylvester and K. Keutzer, "A Global wiring Paradigm for Deep-Submicrometer Design", IEEE Trans. CAD/ICAS, Feb. 2000, pp. 242-252.

[4] C. Cheng, J. Lillis, S. Lin and N. Ching, "Interconnect analysis and Synthesis", John Wiley & Sons, Inc. 2000

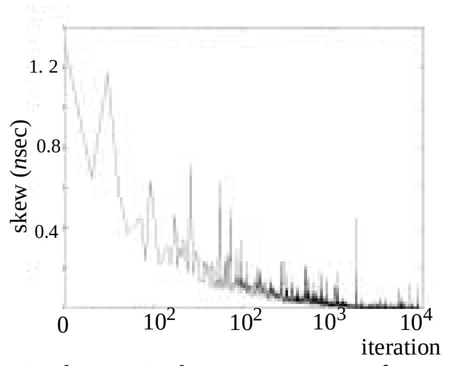

Fig 3 The AWA skew convergence for a set of 1024 clock pins.

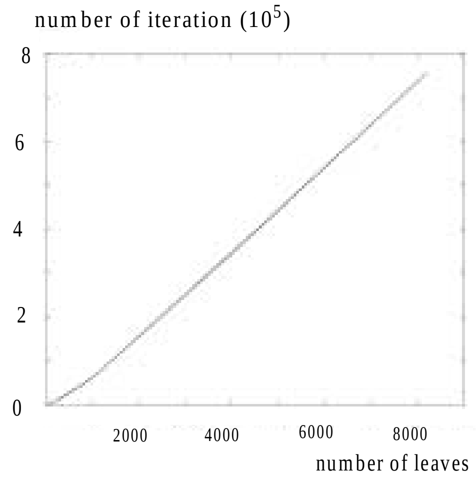

Fig 4 Relation between the number of leaves and the number of iteration to achieve ZS by AWA.

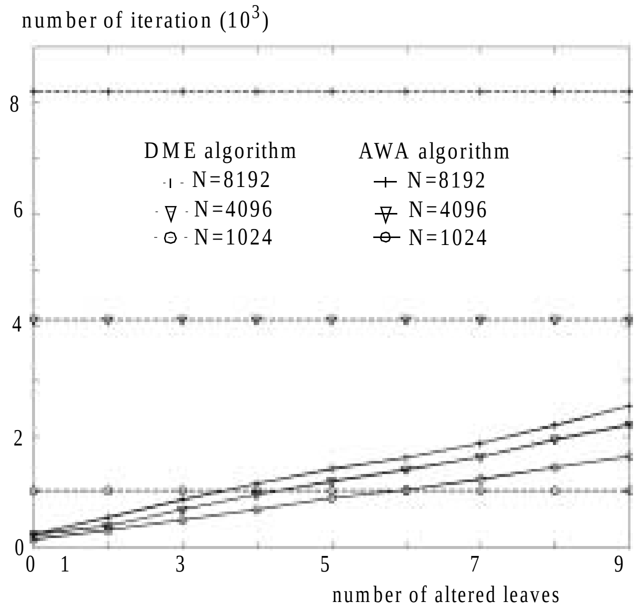

Fig 5 Relation between AWA's iteration and the number of altered leaves for ZSCDNs of different sets of

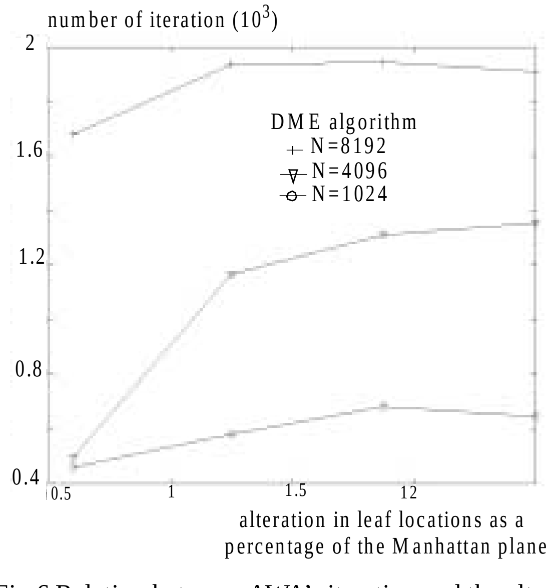

Fig 6 Relation between AWA's iteration and the alteration in the location of four leaves of different sets of clock pins.