Low-Cost Single-Layer Clock Trees With Exact Zero Elmore Delay Skew *

Andrew B. Kahng and Chung-Wen Albert Tsao

UCLA Computer Science Dept., Los Angeles, CA 90024-1596 USA

Abstract

We give the first single-layer clock tree construction with exact zero skew according to the Elmore delay model. The previous Linear-Planar-DME method [11] guarantees a planar solution under the linear delay model. In this paper, we use a Linear-Planar-DME variant connection topology to construct a low-cost zero skew tree (ZST) according to the Elmore delay model. While a linear-delay ZST is trivially converted to an Elmore-delay ZST by "detouring" wires, the key idea is to defer this detouring as much as possible to reduce tree cost. Costs of our planar ZST solutions are comparable to those of the best previous non-planar ZST solutions, and substantially improve over previous planar clock routing methods.

1 Introduction

Given a set $S = \{s_1, s_2, \ldots, s_n\} \subset \Re^2$ of sink locations, and a connection topology G (a rooted binary tree with n leaves corresponding to the sinks in S), we seek a zero-skew tree $T(S)$ which embeds G in the Manhattan plane. Several previous works achieve exact zero-skew routing. Tsay [14] recursively combines pairs of zero-skew trees at "tapping points" to induce the topology G as it is being embedded; exact zero Elmore delay skew is maintained by elongating wires as necessary. The DME algorithm [2, 3, 7] embeds internal nodes of G via (i) bottom-up construction of a *tree of merging segments*, or *merging tree*, representing loci of possible placements of internal nodes in the ZST; and (ii) top-down determination of exact locations for the internal nodes of G. Since DME requires an input topology, several works have studied topology constructions that lead to low-cost solutions when DME is applied [2, 3, 8].

Clock tree solutions given by the above algorithms may not be easily embedded in the layout plane. However, in practice clock nets are often routed with a single preferred layer, e.g., to reduce delay and attenuation through vias. Single-layer routing also has more uniform electrical parameters, reducing sensitivity to process variation and simplifying buffering optimizations. Zhu and Dai [16] gave the first planar ZST construction; their $O(n^2)$ solution has minimum possible source-sink pathlength. Khan et al. [13] hybridized the top-down partitioning approach of [10] with the Zhu-Dai construction. Both [16, 13] rely on the linear delay model to achieve their results.

1.1 Single-Pass DME

In [11], we showed that under linear delay the two-phase DME algorithm can be emulated by a top-down "Single-Pass DME". More precisely, the tree of merging segments constructed in the bottom-up DME phase can be generated *during* the top-down phase.[1] The enabling result is that the root of the minimum-pathlength zero-skew subtree over any sink set $S' \subseteq S$ must always be located at $center(S')$. In other words, the merging segment for any node v is always the core of the minimum TRR that contains all sinks in the subtree rooted at v – independent of the subtree connection topology. Thus, Single-Pass DME allows the connection topology to be determined dynamically in a top-down fashion, and at the same time still finds a minimum-pathlength, minimum-cost embedding of whatever topology is eventually determined.[2]

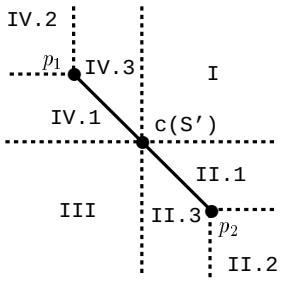
The Linear-Planar-DME algorithm [11] is Single-Pass DME with the top-down determination of node embeddings and connection topology guided by the *embedding rules* and *partitioning rules* illustrated in Figure 1 (reproduced from [11]). These rules correspond to Steps 4 and 6 of the procedure Linear-Planar-

*Partial support for this work was provided by NSF MIP-9223740 and MIP-9257982. We thank Kenneth Boese and Masato Edahiro for use of their DME codes.

[1] Recall that the *merging segment* for node v is the locus of possible embedding points for v that are compatible with a minimum-cost ZST solution. If v has children s and t, the *merging cost* at node v is the sum of the wirelengths of edges $\overline{sv}$ and $\overline{tv}$ in such a minimum-cost ZST solution. Note that this does not necessarily equal the sum of Manhattan distances $d(s,v)+d(t,v)$, since sink delays must be balanced. The following discussion uses the same terminology as in [2, 4, 11] (e.g., Figures 1 and 2). In brief: A *Manhattan arc* is a line segment oriented at a 45-degree angle from the coordinate axes. A *tilted rectangular region*, or TRR, consists of all points within some fixed distance r of a *core* Manhattan arc. $c(S')$ is the midpoint of the Manhattan arc $center(S')$ that lies at the center of pointset S' (i.e., at the core of the minimum TRR containing S'). When the core of a TRR is a single point u, the TRR is called a *Manhattan disk* and is denoted $MD(u,r)$. The embedding point of a node $v \in G$ is denoted by $pl(v)$. Finally, $t_{LD}(v)$ denotes the linear delay from node v to each sink in the subtree rooted at v.

[2] Under the Elmore delay model, the location of the merging segment for the root of the subtree over $S' \subseteq S$ is no longer independent of the subtree connection topology. Hence, the bottom-up DME phase cannot be eliminated, i.e., Single-Pass DME cannot be applied to the Elmore delay model.

DME-Sub in Figure 2, and are what distinguish Linear-Planar-DME from generic Single-Pass DME. The main idea is that (Euclidean) convex polygons can guide the top-down partitioning of both the routing area and the set of sinks. Given $S' \subseteq S$ and a convex polygon $P_{S'}$ containing S', we can recursively divide $P_{S'}$ into two smaller convex polygons, such that routing inside one convex polygon cannot interfere with routing inside the other convex polygon or on the boundary between the polygons. The noninterfering property of the wiring immediately implies a planar solution.



Embedding Rules

Location of p	$pl(v)$
Regions I, III	$c(S')$
Regions II.1, IV.1	intersection of $\overline{p_1 p_2}$ with horizontal line through p
Regions II.3, IV.3	intersection of $\overline{p_1 p_2}$ with vertical line through p
Region II.2 (IV.2)	p_2 (p_1)

Partitioning Rules

Locations of p, v	Splitting line
$p \neq v$	$\overleftrightarrow{pv}$
$p = v$; $p \neq c(S')$	$\overleftrightarrow{p_1 p_2}$
$p = v$; $p = c(S')$	Vertical line through p

Figure 1: Rules used by Linear-Planar-DME: (i) to choose the embedding point of v (the root of the subtree over sink set $S' \subseteq S$ in *any* minimum-radius ZST), and (ii) to choose the splitting line to partition the sink set S' based on the relative positions of v's parent p and $center(S') = \overline{p_1 p_2}$.

1.2 Overview of Our Approach

Given sink locations S and connection topology G, our new *Elmore-Planar-DME* method returns a low-cost planar-embedded clock tree with exact zero Elmore delay skew. Two interesting issues are addressed by our work: (i) choice of the topology G, and (ii) embedding to achieve zero Elmore delay skew.

First, any connection topology can be trivially embedded with exact zero skew onto a single routing layer; however, re-embedding the topology of a non-planar ZST (e.g., from [8]) onto a single layer can drastically increase the tree cost. The correspondence between linear delay and Elmore delay suggests that the (optimum) Linear-Planar-DME solution can be re-embedded with zero Elmore delay skew and very little increase in tree cost. Thus, Linear-Planar-DME is a natural choice for generating the connection topology within our approach.[3]

[3] Interestingly, relaxing the planar-embeddable constraint in variations of Linear-Planar-DME leads to improved solutions (see Section 3 below). This is possible because our method of achieving exact zero Elmore delay skew does not depend on an initial planar-embedded solution.

Algorithm Linear-Planar-DME (S,clk)
Input: Set of sinks S; clock location clk in P_S
Output: Planar ZST $T(S)$ with root s_0; $cost(T)$
1. $r = radius(S)$
2. Build $TRR(u) = MD(u,r)$ for all sinks $u \in S$
3. $center(S) = core(\bigcap_{u \in S} TRR(u))$
4. **if** clk not specified
5. Embed s_0 at $c(S)$ (i.e., $pl(s_0) = c(S)$)
   else
6. Embed s_0 at clk (i.e., $pl(s_0) = clk$)
7. $t_{LD}(s_0) = r + d(pl(s_0), center(S))$
8. P_S = a rectangle containing S and clk
9. Linear-Planar-DME-Sub(S,P_S,s_0)
10. $cost(T) = \sum_{v \in T} |e_v|$

Procedure Linear-Planar-DME-Sub (S',$P_{S'}$,p)
Input: Set of sinks $S' \subseteq S$; convex polygon $P_{S'}$ containing S'; parent node p lying inside $P_{S'}$
Output: Planar ZST $T(S')$ with root v
1. $t_{LD}(v) = radius(S')$
2. $center(S') = core(\bigcap_{u \in S'} TRR(u))$
3. $|e_v| = t_{LD}(p) - t_{LD}(v)$
4. Embed node v at $pl(v) \in center(S')$ by embedding rules in Figure 1
5. Connect a wire from $pl(p)$ to $pl(v)$
6. Divide S' and $P_{S'}$ into S'_1, S'_2 and $P_{S'_1}$, $P_{S'_2}$ by partitioning rules in Figure 1
7. $parent(v) = p$
8. **if** $|S'| = 1$ RETURN
9. Linear-Planar-DME-Sub(S'_1,$P_{S'_1}$,v)
10. Linear-Planar-DME-Sub(S'_2,$P_{S'_2}$,v)

Figure 2: The Linear-Planar-DME Algorithm.

Second, given a Linear-Planar-DME solution, it is simple to obtain a planar Elmore-ZST by elongating tree edges in a bottom-up fashion to balance differences in sink delays (see [14]). In the experimental comparisons of Section 4 below, we call such an approach "Naive-Elmore-Planar-DME". We find that unneeded detouring can be saved by iterating both the application of DME to the given topology, and the bottom-up modification of any resulting non-planar routing, based on a "principle of least commitment". Planarity is enforced in bottom-up order, with planar-embedded subtrees being retained and routing at higher levels being modified. Whenever any non-planar routing at some level of the ZST is changed, the merging tree for the ZST above this level is rebuilt, and top-down DME embedding is applied to the new merging tree. The complementary processes of merging tree reconstruction and top-down embedding are iterated until the entire ZST is planar.

Note that the DME algorithm cannot guarantee optimal tree cost under the Elmore model [2]. Thus, our approach only heuristically minimizes the cost of the output planar Elmore-ZST.

2 Elmore-Planar-DME

Let T be a ZST with topology G, and let T_v denote the subtree of T rooted at node $v \in G$. (For conve-

nience, we overload embedded points $v \in T$, and nodes $v \in G$ of the topology, since the correct meaning is usually clear.) Our method marks each point $v \in T$ as either planar or non-planar. An edge in T is called a *planar edge* if both of its endpoints are marked as planar. A path $s \rightsquigarrow t$ is a sequence of line segments from s to t; a *planar path* is a path that does not cross any edge of T. We use $cost(s \rightsquigarrow t)$ and $hops(s \rightsquigarrow t)$ to respectively denote the cost of a path and the number of segments in the path. Finally, $bbox(s,t)$ denotes the smallest rectangle containing points s and t.

The Elmore-Planar-DME algorithms is described in Figure 3.[4] Initially, a ZST T is obtained by applying DME to the given topology G and sink set S. Then, every sink is marked planar and all other nodes are marked non-planar. As long as the ZST T has a non-planar node, Elmore-Planar-DME iterates at Steps 7 and 8. Note that Step 7 constructs the merging tree TS only for non-planar nodes in the upper part of the ZST; Step 8 calls the top-down embedding phase of DME ("Find-Exact-Placements(TS)" in [2, 4]) to embed the shrinking set of non-planar nodes.

Algorithm Elmore-Planar-DME (G, S)
Input: Topology G; set of sinks S
Output: Planar ZST T having topology G
1. ZST T = DME(G,S)
2. **for** each $v \in T$
3. **if** v is a sink
4. mark v planar
else
5. mark v non-planar
6. **while** T still has a non-planar node **do**
7. Merging tree TS =
Rebuild-Tree-of-Segments(T,S)
8 T = Top-down embedding(TS) by DME

Procedure Rebuild-Tree-of-Segments(T, S)
Input: ZST T having topology G; set of sink S
Output: Merging tree TS
1. L = Deepest level in G containing non-planar nodes
2. A = { v | v is non-planar and at level L in G}
3. B = { v | v is non-planar and at level $< L$ in G}
4. **for** each node v (with embedding point w and children s, t) in A (increasing order of merging cost)
5. **if** $\overline{sw}$ and $\overline{tw}$ do not cross any planar edges
6. mark v as planar
7. **else**
8. $s \rightsquigarrow t$ = Find-Merging-Path(T, v)
9. **if** $cost(s \rightsquigarrow t) = d(s,t)$ and $hops(s \rightsquigarrow t) \leq 3$
10. u' = intersection of $ms(v)$ and path $s \rightsquigarrow t$
11. Planar path $s \rightsquigarrow t$ = Detour-1(T, v, u')
12. Detour-2(T, v, $s \rightsquigarrow t$)
13. **for** each node v in B (bottom-up order)
14. Construct merging segment $ms(v)$

Figure 3: The Elmore-Planar-DME Algorithm.

Because non-planar nodes are made planar in bottom-up order, the procedure Rebuild-Tree-of-Segments identifies the lowest non-planar nodes in the tree, i.e., the node set A at level L of the tree. Nodes in A have planar children and will be made planar in the current iteration. Even though there may be other non-planar nodes whose children are all planar, their processing is deferred since subtrees at lower levels of the ZST tend to contain shorter tree edges, and it is easier for longer edges to detour around shorter edges than vice-versa. This same reason suggests processing the nodes of A in order of increasing merging cost. For each non-planar node v in Set B, we construct a new merging segment $ms(v)$ as in the bottom-up phase of the DME algorithm.

[4] For simplicity, the template assumes that no clock source location has been prescribed. Accommodating a fixed clock source is simple, as seen from Figure 2.

2.1 Merge Paths and Detouring

Consider any node v with parent p and children s and t. The DME algorithm is based on two facts: (i) the two subtrees of node v can be merged with minimum cost $d(s,t)$ anywhere on $ms(v)$; and (ii) the merging cost for node p and its sibling will depend on the embedding point of node v. Within our methodology, suppose that DME has chosen v's embedding point w on $ms(v)$, and that edge $\overline{sw}$ or edge $\overline{tw}$ crosses an existing planar edge. We now discuss how Rebuild-Tree-of-Segments determines an embedding point and associated planar routing, while heuristically minimizing both the merging cost for v and the merging cost for p.

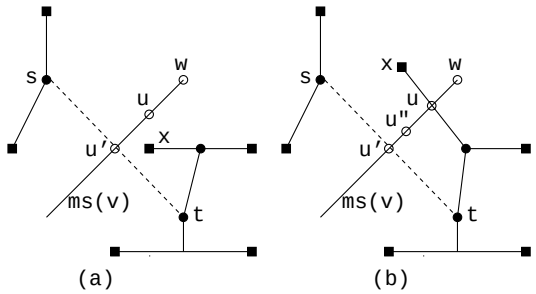


Figure 4: Selection of an embedding point for node v which reduces merging cost for v's parent while keeping minimum merging cost (= $d(s,t)$) for node v. Point w is the embedding point for v returned by DME. $ms(v)$ indicates the merging segment for v.

When $\overline{sw}$ or $\overline{tw}$ crosses a planar edge, Step 8 (Find-Merging-Path) seeks a planar detouring path $s \rightsquigarrow t$ with low merging cost at both v and p. If the $s \rightsquigarrow t$ path has minimum possible pathlength (= $d(s,t)$), the merging cost for v can be minimized at the same time that v is made planar. For instance, we could embed v at the point u' where $s \rightsquigarrow t$ intersects the merging segment $ms(v)$ (refer to Figure 4). Recall, however, the original embedding point w that was returned by DME: if we can shift u' to another point u that is closer to w, while $s \rightsquigarrow u \rightsquigarrow t$ remains a shortest planar path, then we can reduce the merging cost at v's parent. Step 11 (Detour-1 subroutine call) performs this shifting task. In practice, we apply Detour-1 only when s and t are very close, e.g., $hops(s \rightsquigarrow t) \leq 3$. Then, Detour-2 at Step 12 gradually brings the roots of the two subtrees below v closer together, using as little detouring as possible.

2.2 Details of the Subroutines

Some details of Find-Merging-Path are given in Figure 6. Note that finding a shortest path over all pos-

sible detouring points may not minimize the merging cost at p, and that slightly greater merging cost at v may result in much lower merging cost at p. Figure 5 shows an example in which path P_1 is slightly longer than path P_2, but is a better choice since it passes much closer to the DME embedding point w. To balance between efficiency and solution quality, Find-Merging-Path gradually increases the set of possible detouring points, in the hope that a feasible path will be found early (i.e., when the problem size is small).

Experimental results below use $V_1 = \{u \mid u \in T_q$, where edge connecting q and its parent intersects $\overline{sw}$ or $\overline{tw}\}$ and $V_2 = V_1 \cup \{\ u \mid u \in$ the triangle formed by s, t and $w\}$. (For the example in Figure 5, Find-Merging-Path will use $V_1 = \{a, b\}$ and $V_2 = \{a, b, c, d\}$.) These choices of V_1 and V_2 allow planar paths near w to be selected first. If Find-Merging-Path fails to discover a feasible path, the procedure considers a succession of larger point sets V_i, $i \geq 3$; in our experiments, these are simply increasing dilations of the bounding box $bbox(s,t)$.

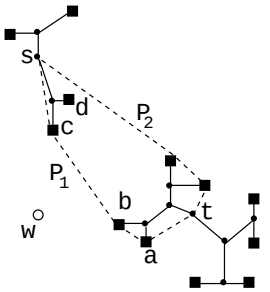


Figure 5: Selection of merging path $s \rightsquigarrow t$ to heuristically minimize the merging cost at both v and its parent p. Point w is the embedding point for node v returned by DME.

Procedure Find-Merging-Path(T, v)
Input: ZST T; node v with children s and t
Output: Planar path $s \rightsquigarrow t$
1. $i = 1$
2. **do**
3. Construct $V_i \equiv i^{th}$ set of candidate detour points
4. $s \rightsquigarrow t$ = Find-Shortest-Planar-Path(T,V_i,s,t)
5. $i = i + 1$
6. **while** ($s \rightsquigarrow t$ has not yet been found)

Figure 6: Procedure: Find-Merging-Path.

Procedure Detour-1 selects embedding points on $ms(v)$ which heuristically minimize the merging cost at p while keeping minimum merging cost $= d(s,t)$ at node v. The procedure first selects a set of candidate embedding points on $ms(v)$. Then, each selected point u (in increasing order of $d(u,w)$) is checked to see whether the shortest planar path $s \rightsquigarrow u \rightsquigarrow t$ has cost $= d(s,t)$. The shortest planar path $s \rightsquigarrow u \rightsquigarrow t$ is obtained by calling Find-Shortest-Planar-Path twice, i.e, by finding $s \rightsquigarrow u$ and $u \rightsquigarrow t$. Note that to find a minimum-cost path, say, $s \rightsquigarrow u$, we need only consider detouring points inside $bbox(s,u)$. The procedure terminates when the first $s \rightsquigarrow u \rightsquigarrow t$ path is found.

There are two types of candidate embedding points on $ms(v)$: the intersection of $ms(v)$ with any vertical or horizontal line through any point inside $bbox(s,t)$ (e.g., see Figure 4a), and (ii) the intersection of $ms(v)$ with any planar edge (e.g., see Figure 4b)[5]. Again, the key property of u in Figure 4 is that it is the point on $ms(v)$ closest to the DME embedding point w, such that the merging path through u still has minimum cost $= d(s,t)$.

Procedure Detour-2 uses a "principle of least commitment" whereby the distance between the two children of node v is shortened by one hop at each iteration. Consider that the current non-planar node v has children s and t, and we have a planar path $s \rightsquigarrow t = \{s, s', \ldots, t', t\}$. Without loss of generality, assume that $d(s,s') \leq d(t,t')$ or that s' is on $ms(v)$. Then, Detour-2 implements only the partial path $\overline{ss'}$, with s' replacing s as a child of v. In this way, v's children are "pulled closer" so that v can be more properly re-embedded by DME. This avoidance of "commitment" also allows Detour-2 to minimize the harmful effects of a suboptimal result from Find-Merging-Path.[6]

Finally, both Find-Merging-Path and Detour-1 call the procedure Find-Shortest-Planar-Path, which seeks a shortest path between two points s and t in the presence of obstacles (the obstacles are the planar edges comprising subtrees of the ZST T). Since the detouring points must be located at the endpoints of planar edges, a general approach based on visibility graphs (e.g., [1, 15]) can be used. Our current implementation uses Dijkstra's algorithm in the visibility graph, with edge weights computed on the fly; this does not cause excessive runtimes (see Section 4) since the size of the allowable detouring pointset is small in the most procedure calls.

3 Linear-Planar-DME Variants

As noted above, Elmore-Planar-DME does not actually require a planar-embedded ZST as input. Thus, while we use the topology G obtained from the Linear-Planar-DME solution, this construction can be modified. We have considered modifications to the partitioning rules of Section 1.1 which change the splitting line to a splitting *path* of two or more line segments. In other words, rather than draw a straight line through points p and v, we draw a line segment $\overline{pv}$ and a ray $\vec{v}$ emanating from v to separate the polygons P_{S_1} and P_{S_2}. Since we no longer have a straight splitting line, one of the new smaller regions may be non-convex, and more case analysis is required to maintain guaranteed planarity of the output ZST. From a theoretical perspective, such Linear-Planar-DME variants are unappealing: we lose guaranteed-planarity, and the worst-case time complexity increases. However, all ZST's obtained in our experiments remain planar, with non-convex polygons becoming further divided into smaller convex polygons within the succeeding two or three levels. Furthermore, such variants can achieve averages of

[5] In this figure, all points on $\overline{uu''}$ actually yield the same merging cost savings at v's parent. For such reasons as crosstalk minimization, point u'' may actually be a better choice than u. However, in our present implementation we simply select u.

[6] Thus, Procedure Rebuild-Tree-of-Segments may iterate several times at each level. In our experience, no more than 56 iterations in total were necessary for any of the benchmarks tested.

up to 10.9% wirelength reduction versus our previous results in [11]. We briefly describe two possible Linear-Planar-DME variants.

Using a Splitting Path

Consider a subset of sinks $S' \subseteq S$ that is being partitioned. Recall that the splitting path consists of line segment $\overline{pv}$ and $\vec{v}$, a ray emanating from v. The line segment $\overline{pv}$ has been determined, but there are $|S'|-1$ different choices of $\vec{v}$. To determine all possible choices of $\vec{v}$, our Linear-Planar-DME-2 variant sorts the sinks of S' in clockwise order around point v; each pair of consecutive sinks determines a splitting path. A given splitting path will partition S' into S'_1 and S'_2. To choose among the possible splitting paths, Linear-Planar-DME-2 uses a heuristic estimate of the cost of the ZSTs over S'_1 and S'_2. We have experimentally determined two such estimates:

- $r_1 \times |S'_1| + r_2 \times |S'_2|$, where r_1 and r_2 are the respective radii of the sink sets S'_1 and S'_2; and
- $r_1 \times |S'_1| + r_2 \times |S'_2| + 0.5r(|c_1 - c_2|/c)^2$, where r is the radius of S' and c, c_1 and c_2 are the respective total capacitances of the sink sets S', S'_1 and S'_2.

The latter estimate considers load balance when bipartitioning the sinks, and yields slightly better results (it is also the estimate used in the experiments reported below). More useful cost functions for sink partitioning are still under investigation.

In the Manhattan plane, computing the radii of all pairs of sink subsets (corresponding to bipartitions of S') can be accomplished in $O(|S'|)$ time. Thus, the sorting operation dominates the time complexity, and the overall Linear-Planar-DME-2 complexity is $O(l \cdot n \lg n)$, where l is the number of levels in the output ZST and $n = |S|$. In practice, l is very close to $\lg n$.

Using a Splitting Path and Lookahead

Our Linear-Planar-DME-3 variant is similar to Linear-Planar-DME-2, but chooses splitting paths more carefully based on lookahead. After determining a set of candidate bipartitions of S', we estimate the cost of each by actually constructing the ZST that will be output by Linear-Planar-DME-2. To maintain practical runtimes, the number of candidate bipartitions considered is limited to a small constant (≈ 16 in the experiments reported below). Given this constraint, our Linear-Planar-DME-3 implementation has worst-case runtime of $O(l^2 \cdot n \lg n)$; as reported in the next section, $l \approx 1.3 \cdot \lg n$ in our experiments.

4 Experimental Results

We implemented the Linear-Planar-DME and Elmore-Planar-DME algorithms using Sun Sparc-10 workstations and the C/Unix environment. The same seven examples as in [4, 8, 16] were studied. Table 1 shows that our Elmore-Planar-DME implementation is relatively efficient, with runtimes dominated by the generation of a good topology in the call to Linear-Planar-DME-3. Note that the relatively high runtimes for the P2 and r4 test cases are due to more uneven distribution of sink locations, which leads to more detouring.

benchmark (#pins)	Lin-Planar DME	Elm-Planar DME	ZST height ($l/\lg n$)
P1(269)	115	1	9 (1.1)
P2(603)	362	60	11 (1.2)
r1(267)	114	2	11 (1.4)
r2(598)	382	7	12 (1.3)
r3(862)	664	6	12 (1.2)
r4(1,903)	2512	90	14 (1.3)
r5(3,101)	5557	85	15 (1.3)

Table 1: Sun Sparc-10 CPU time (seconds) for our Elmore-Planar-DME implementation. Note that the topology generation via Linear-Planar-DME-3 requires much more time than the embedding by Elmore-Planar-DME. In the last column, we also show the ZST height as a multiple of the minimum possible tree height, $\lg n$.

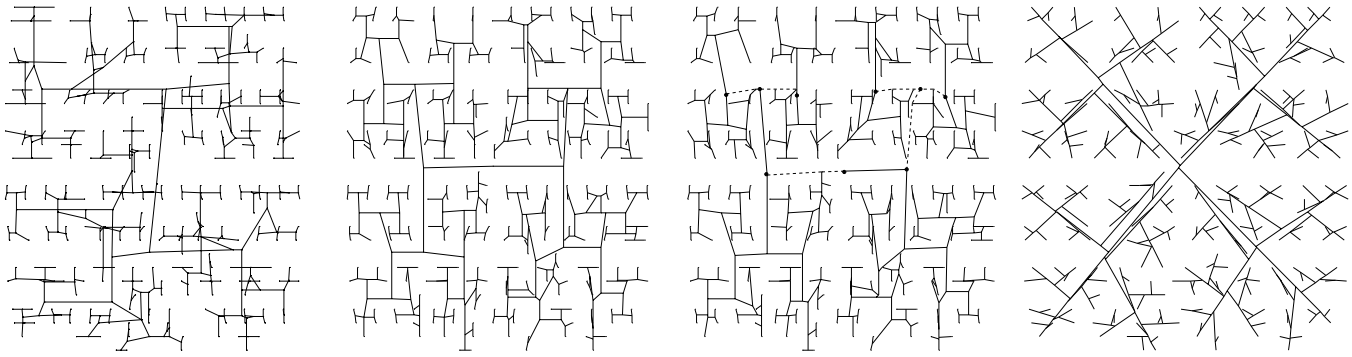
Table 2 compares our algorithms with two leading non-planar ZST algorithms in the literature – Greedy-DME [8] and KCR+DME [12] – as well as the previous planar routing method of Zhu and Dai [16]. Our new planar ZST solutions are competitive with the best non-planar ZST solutions of Greedy-DME (having 10% greater wiring cost), and are superior to KCR+DME solutions in all cases. Elmore-Planar-DME also uses 22.5% less wire than the (linear delay based) method of [16]. It is interesting to note that the cost of our Elmore-Planar-DME solutions is only slightly increased from the cost of the starting Linear-Planar-DME ZSTs. We believe this implies that better solutions can be obtained as we continue to improve Linear-Planar-DME. Finally, Figure 7 shows ZSTs for the Primary1 benchmark constructed by Greedy-DME, Linear-Planar-DME, Elmore-Planar-DME, and the method of Zhu and Dai.

5 Future Work

We have considered several improvements to our current work. First, the output of our Planar-DME approach may be viewed as a *planar routing sketch* for a ZST. Currently, we do not take routing capacity, cross-talk constraints, etc. into consideration (recall the example of Figure 4b). We hope to use such computational geometry techniques as those of Dai et al. [6] to enhance our current approach. Second, Find-Shortest-Planar-Path provides the main computational bottleneck in practice. Various heuristic speedups are possible, e.g., obstacles (planar edges) are actually connected as subtrees, and each subtree can be replaced by its convex hull to reduce the complexity of the path-finding instance. Third, Linear-Planar-DME itself can be improved to yield better connection topologies for input to Elmore-Planar-DME, e.g., by using more sophisticated partitioning rules (e.g., using splitting paths with more than two segments; clustering sinks before partitioning) and embedding rules (e.g., embed the root of zero-skew subtree over S' at a more appropriate place than $c(S')$ for the Elmore delay model). Finally, we are pursuing methods which construct single-layer clock routing trees with *bounded*, rather than exactly zero, skew.

Test Case	CL+I6[8]	Lin-Pln-DME	Elm-Pln-DME	KCR+DME [4]	Naive Elmore-Planar-DME	Zhu-Dai [16]
P1	129.2	130.2	132.9	140.1	146.1	167.9
P2	304.0	320.1	330.2	345.2	391.6	422.5
r1	1,253.3	1,351.3	1,393.2	1,487	1,686.2	1,778.3
r2	2,483.8	2,725.1	2,789.3	3,020	3,315.2	3,580.1
r3	3,193.8	3,501.4	3,556.2	3,867	3,916.2	4,635.9
r4	6,499.7	7,073.4	7,261.7	7,713	8,582.5	9,577.1
r5	9,723.7	10,556.2	10,808.1	11,606	12,823.3	14,119.4
Cost (ave)	(+0.0%)	+7.2%	+9.9%	+17.3%	+28.1%	+39.4%
Planar	No	Yes	Yes	No	Yes	Yes
Delay Model	Elmore	Linear	Elmore	Elmore	Elmore	Linear

Table 2: Comparison of Elmore-Planar-DME with other algorithms in terms of total wirelength, using the same benchmarks studied in [8, 4, 16]. No prescribed clock source location was assumed. Cost (ave) indicates the average percentage increase in wire length versus the results of CL+I6 [8].



(a) Greedy-DME [8] (b) Linear-Planar-DME (c) Elmore-Planar-DME (d) Zhu-Dai [16]

Figure 7: Zero-skew clock routing solutions for the Primary1 benchmark. Six instances of detour routing in (c) are highlighted with dotted lines.

References

[1] T. Asano, T, Asano, L. Guibas, J. Hershberger, and H. Imai, "Visibility-polygon search and Euclidean shortest paths", *Proc. IEEE Symp. on Foundations of Computer Science*, 1985, pp. 155-164.

[2] K. D. Boese and A. B. Kahng, "Zero-Skew Clock Routing Trees With Minimum Wirelength," *Proc. IEEE Intl. Conf. on ASIC*, 1992, pp. 1.1.1 - 1.1.5.

[3] T.-H. Chao, Y.-C. Hsu and J.-M. Ho, "Zero Skew Clock Net Routing," *Proc. ACM/IEEE Design Automation Conf.*, 1992, pp. 518-523.

[4] T.-H. Chao, Y. C. Hsu, J. M. Ho, K. D. Boese and A. B. Kahng, "Zero Skew Clock Routing With Minimum Wirelength", *IEEE Trans. on Circuits and Systems* 39(11) (1992), pp. 799-814.

[5] J. Cong, A. B. Kahng and G. Robins, "Matching-Based Methods for High-Performance Clock Routing", *IEEE Trans. on CAD* 12(8), August 1993, pp. 1157-1169.

[6] W. M. Dai, R. Kong, J. Jue, and M. Sato, "Rubber band routing and dynamic data representation", *Proc. IEEE Intl. Conf. on Computer-Aided Design*, 1990, pp. 52-55.

[7] M. Edahiro, "Minimum Skew and Minimum Path Length Routing in VLSI Layout Design", *NEC Research and Development* 32(4), October 1991, pp. 569-575.

[8] M. Edahiro, "Clustering-Based Optimization Algorithm in Zero-Skew Routings", *Proc. ACM/IEEE Design Automation Conf.*, June 1993, pp. 612-616.

[9] W. C. Elmore, "The Transient Response of Damped Linear Networks With Particular Regard to Wide-Band Amplifiers," *J. Applied Physics* 19(1) (1948), pp. 55-63.

[10] M. A. B. Jackson, A. Srinivasan and E. S. Kuh, "Clock Routing for High Performance ICs," *Proc. ACM/IEEE Design Automation Conf.*, 1990, pp. 573-579.

[11] A. B. Kahng and C.-W. A. Tsao. "Planar-DME: Improved Planar Zero-Skew Clock Routing With Minimum Pathlength Delay," *Proc. ACM/IEEE European Design Automation Conf.*, September 1994.

[12] A. B. Kahng, J. Cong, and G. Robins, "High-Performance Clock Routing Based on Recursive Geometric Matching," *Proc. ACM/IEEE Design Automation Conf.*, 1991, pp. 322-327.

[13] W. Khan, X. He, L. Bangaru and N. Sherwani, "Combat: Zero Skew Minimal Delay Planar Clock Routing for High Performance Systems", *Western Michigan Univ. Computer Science Technical Report TR/93-08*, April 15, 1993

[14] R. S. Tsay, "Exact Zero Skew", *Proc. IEEE Intl. Conference on Computer-Aided Design*, 1991, pp. 336-339.

[15] E. Welzl, "Constructing the Visibility Graph for n Line Segments in $O(n^2)$ Time", *Information Processing Letters* 20 (1985), pp. 167-171.

[16] Q. Zhu and W. W.-M. Dai, "Perfect-Balance Planar Clock Routing With Minimal Path-Length", *Proc. IEEE Intl. Conf. on Computer-Aided Design*, Nov. 1992, pp. 473-476.