Minimum-Cost Bounded-Skew Clock Routing*

Jason Cong and Cheng-Kok Koh

UCLA Computer Science Department

310-206-2775 (tel) 310-825-2273 (fax) {cong,kohck}@cs.ucla.edu

Abstract

In this paper, we present a new clock routing algorithm which minimizes total wirelength under any given path-length skew bound. The algorithm constructs a bounded-skew tree (BST) in two steps: (i) a bottom-up phase to construct a binary tree of *shortest-distance feasible regions* which represent the loci of possible placements of clock entry points, and (ii) a top-down phase to determine the exact locations of clock entry points. Experimental results show that our clock routing algorithm, named BST/DME, can produce a set of routing solutions with skew and wirelength trade-off.

1 Introduction

Clock skew minimization is an important issue in the design of high performance circuits. Over the past few years, a number of clock routing algorithms have been proposed, including the H-tree construction for regular systolic arrays [1], the method of means and medians (MMM) by [10], the recursive geometric matching method by [6], and exact zero skew routing under the Elmore delay model by [17]. Recently, the problem of embedding a given *topology* on a Manhattan plane with zero path-length skew is solved optimally by [2, 7] using the *Deferred-Merge Embedding* (DME) algorithm. The algorithm can be either applied to a given clock topology [2] or combined with a clock topology generation algorithm to achieve zero skew with smaller wirelength [7]. Currently, research on clock routing is moving along a few directions. Zero-skew planar routing was first proposed by [18] using Max-Min operations, followed up by [12, 13] using single-phase DME algorithm. Other work includes buffer insertion [9, 3], process-variation-tolerant skew minimization [15, 4, 14], and a clock router that accomplished specified pin-to-pin delay [16].

The emphasis of most of the current clock routing algorithms is on achieving zero-skew at the expense of longer wirelength, resulting in high power dissipation. In practice, circuits still operate correctly within a tolerable skew bound. To reduce clock net power dissipation, we believe that the clock routing algorithm should consider bounded-skew trees (BST) instead of zero-skew trees (ZST). In this paper, we propose an algorithm to construct BST based on the DME approach. Our *BST/DME algorithm* first computes shortest-distance feasible regions (as opposed to the merging segment in the original DME algorithm) for the roots of

*This work is partially supported by ARPA/CSTO under Contract J-FBI-93-112, the National Science Foundation Young Investigator Award MIP9357582, a grant from Intel Corporation and a Tan Kah Kee Foundation Postgraduate Scholarship.

recursively merged subtrees in a bottom-up fashion, followed by a top-down phase to determine the exact embedding of the clock entry points. Experimental results show that as the skew bound increases, we generally see a decrease in total wirelength.

The rest of the paper is organized as follows: In Section 2, we formulate the minimum-cost bounded-skew clock routing problem. In Section 3, we present the BST/DME algorithm under the path-length delay model. Section 4 shows the experimental results obtained by our DME-BST algorithm. Section 5 concludes the paper.

2 Problem Formulation

Assume that we are given a set of m synchronizing components or sinks $\mathcal{N} = \{N_1, N_2, \cdots, N_m\}$ and their locations, denoted $l(\mathcal{N}) = \{l(N_1), l(N_2), \cdots, l(N_m)\}$, on a Manhattan plane. The location of the clock source N_0 may be given. A *routing topology*, $R(\mathcal{N})$, is a rooted binary tree with m leaves, each corresponding to a sink. Consider any two nodes, say u and v, with a common parent node $w = p(u) = p(w)$, then w corresponds to the *clock entry point* that the clock signal from the source has to pass through before reaching u and v (and their descendants).

A clock tree $T(R(\mathcal{N}))$ is an embedding of the routing topology in the Manhattan plane, i.e. it maps each internal node $v \in R(\mathcal{N})$ to a location, denoted $l(v)$, on the Manhattan plane. Since each node has a unique parent, we denote the edge from any node, say u to its parent $p(u)$ uniquely by e_u. The cost of edge e_u is its wirelength, denoted $|e_u|$. Note that $|e_u|$ is at least as large as the Manhattan distance between $l(u)$ and $l(p(u))$. The cost of the tree $T(R(\mathcal{N}))$ is the total wirelength of the edges in $T(R(\mathcal{N}))$.

Given a routing tree $T(R(\mathcal{N}))$, let $P(u, v)$ denote the unique path from u to v where u is a ancestor of v in $R(\mathcal{N})$. For a node v in $T(R(\mathcal{N}))$, we use $T(v)$ to denote the subtree rooted at v. Under the linear delay model, the signal propagation time from u to v is the sum of the wirelengths of the edges in the path $P(u, v)$, i.e. $\sum_{e \in P(u,v)} |e|$. Let $PL(u, v)$ denote the signal propagation delay time (more precisely, path-length) from u to v. The *skew* of $T(R(\mathcal{N}))$, denoted $skew(T(R(\mathcal{N}))$, is defined to be the maximum value of $|PL(N_0, N_i) - PL(N_0, N_j)|$ over all $N_i, N_j \in \mathcal{N}$. Given the above definitions, we can formally define the *Minimum-Cost Bounded-Skew Clock Routing* problem as follows:

Minimum-Cost Bounded Skew Clock Routing Problem (MCBS Problem): Given a set of sinks $\mathcal{N}$ with locations $l(\mathcal{N})$ and a skew bound B, find a routing topology $R(\mathcal{N})$ such that a bounded skew tree $T(R(\mathcal{N}))$ can be constructed with minimum total wirelength and $skew(T(R(\mathcal{N}))) \leq B$.

3 The BST/DME Algorithm

Our BST/DME algorithm computes a routing tree in two steps. The bottom-up process constructs a tree of *shortest-distance feasible regions* (SDFR) which contain possible locations of the internal nodes in the BST. The top-down process then determines the exact locations of all internal nodes (similar to the DME algorithm). First, we define the following terminology.

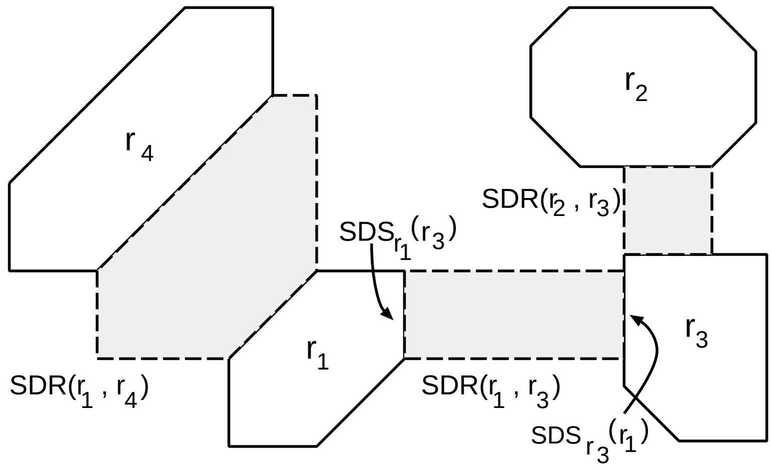


Figure 1: SDRs between regions r_1 and r_3, r_1 and r_4, and r_2 and r_3.

3.1 Definition of SDFR

The *distance* between two points u and v, denoted $d(u, v)$, is the Manhattan distance between the points. We define the distance between two regions r_i and r_j, denoted $d(r_i, r_j)$, to be $\min_{u \in r_i, v \in r_j} d(u, v)$. The *shortest distance region* (SDR) between two regions r_i and r_j, denoted $SDR(r_i, r_j)$, is defined to be the set of points $\{p \mid d(p, r_i) + d(p, r_j) = d(r_i, r_j)\}$. Figure 1 shows some SDRs between regions. We refer to the segments $SDS_{r_i}(r_j) = SDR(r_i, r_j) \cap r_i$ and $SDS_{r_j}(r_i) = SDR(r_i, r_j) \cap r_j$ as the *shortest distance segments* (SDS) that define $SDR(r_i, r_j)$ (see Figure 1).

Given a routing topology $R(\mathcal{N})$, a SDFR of each clock entry point v, denoted $SDFR(v)$ is defined recursively as follows:

(i) For each sink N_i, $SDFR(N_i) = \{l(N_i)\}$.

(ii) For an internal node in v_k in $R(\mathcal{N})$ with children v_i and v_j, $SDFR(v_k)$ is the region of possible placement of v_k within $SDR(SDFR(v_i), SDFR(v_j))$ such that the path-length difference from v_k to any pair of sinks in $T(v_k)$ is within the skew bound and the *merging cost* $|e_{v_i}| + |e_{v_j}|$ is minimized, under the condition that a point $p \in SDFR(v_i)$ can merge with a point $q \in SDFR(v_j)$ only if $d(p, q) = d(SDFR(v_i), SDFR(v_j))$.

The above definition is based on the observation that the merging cost of v_i and v_j is at least as large as $d(SDFR(v_i), SDFR(v_j))$. Therefore, the merging cost is minimized if we can find suitable placement of v_k in $SDR(SDFR(v_i), SDFR(v_j))$ such that $|e_{v_i}| + |e_{v_j}| = d(SDFR(v_i), SDFR(v_j))$.

Each location p in a SDFR is associated with two delays (or path-lengths), namely $SPL(p)$ and $LPL(p)$ which correspond to the *shortest path-length* and *longest path-length* from p to the set of sinks rooted under p, respectively. We define $skew(p) = LPL(p) - SPL(p)$. These numbers are used for computing SDFR of its parent node.

3.2 Overview of the BST/DME Algorithm

Bottom-Up Phase: Topology Construction

Given a set of unconnected sink locations $l(\mathcal{N})$, the original topology corresponds to a forest $\mathcal{F}$ of m single-node trees. The

Topology Construction Algorithm

$\mathcal{F} \leftarrow \emptyset$

for each sink $N_i \in \mathcal{N}$ **do**

Create node v_i

$\mathcal{F} \leftarrow \mathcal{F} \cup \{v_i\}$

$SDFR(v_i) \leftarrow l(v_i)$

end for

while $|\mathcal{F}| > 1$ **do**

Construct the nearest-neighbor graph G of $\mathcal{F}$

For each edge (v_i, v_j) in the first $s(1, |\mathcal{F}|/k, |\mathcal{F}| - 1)$

edges of G in non-decreasing order **do**

if v_i and v_j are unmatched nodes **then**

Mark (v_i, v_j) as matched

Create new node v_k

v_k(children) $\leftarrow \{v_i, v_j\}$

Compute $SDFR(v_k) \leftarrow Merge(SDFR(v_i), SDFR(v_j))$

$\mathcal{F} \leftarrow \mathcal{F} - \{T(v_i), T(v_j)\} \cup \{T(v_k)\}$

end if

end for

end while

Figure 2: The bottom up phase to construct a tree of feasible regions.

SDFR of each sink corresponds to its given location, and $SPL(l(N_i)) = LPL(l(N_i)) = 0$ for $i = 1...m$. The bottom-up phase constructs the topology and the tree of SDFRs by repeatedly merging pairs of trees until $\mathcal{F}$ contains only a single rooted binary tree, which is the routing topology $R(\mathcal{N})$. The approach is similar to the clustering-based algorithm in [8].

During each merging step, a *nearest neighbor graph* [8] is constructed. The nodes in the nearest neighbor graph correspond to the roots of trees in the forest. Two nodes v_i and v_j are connected in the graph if v_j is nearest to v_i or v_i is nearest to v_j. The weight of the edge connecting v_i and v_j is the distance between $SDFR(v_i)$ and $SDFR(v_j)$. A matching is then obtained by inspecting the first $s(1, |\mathcal{F}|/k, |\mathcal{F}| - 1)$ edges in the nearest neighbor graph in non-decreasing order, where k is a parameter > 1 and the function $s(a, b, c)$, with $a \leq c$, is defined by [8]

$$s(a, b, c) = \begin{cases} a; & a \geq b, \\ b; & a < b < c, \\ c; & b \geq c. \end{cases}$$

For each pair of matched nodes, say v_i and v_j, a new clock entry point v_k which corresponds to the parent node of v_i and v_j in the routing topology is introduced. The function $Merge(SDFR(v_i), SDFR(v_j))$ computes the FDSR of the parent node v_k. The details of the function is given in Section 3.4. The outline of the bottom-up phase is given in Figure 2.

Top-Down Phase: Topology Embedding

The details of the top-down phase is given in Figure 3. Let v_0 be the root of the routing topology. If the physical location

Topology Embedding Algorithm

$v_0 \leftarrow$ root of $R(\mathcal{N})$
if $l(N_0)$ is given **then**
  Choose $q \in SDFR(v_0)$ such that
    $d(l(N_0), q) = \min_{p \in SDFR(v_0)} d(l(N_0), p)$
else
  Choose any $q \in SDFR(v_0)$
  end if
$l(v_0) \leftarrow q$
for each internal node $v_i \neq v_0$ in $R(\mathcal{N})$ (top-down order) **do**
  Let v_k be the parent node of v_i
  Choose $q \in SDFR(v_i)$ s.t.
    $d(l(v_k), q) = \min_{r \in SDFR(v_i)} d(l(v_k), r)$
  $l(v_i) \leftarrow q$
end for

Figure 3: The top-down phase to determine the exact locations of clock entry points.

of the clock source N_0 is given, we place v_0 at the nearest location in $SDFR(v_0)$ from $l(N_0)$. Otherwise, we assign v_0 with an arbitrary location (or a location with best skew) in $SDFR(v_0)$. We process the rest of the internal nodes of the routing topology in a top-down order. Consider an internal node $v_i \neq v_0$. Let $v_k = p(v_i)$ be the parent of v_i. Select a point $q \in SDFR(v_i)$ such that the distance $d(l(v_k), q)$ is minimized. Note that $|e_{v_i}|$ may not be equal to $d(l(v_k), q)$.

3.3 Properties of BST/DME Algorithm

We can show the following results for our BST/DME algorithm. We define an octilinear line segment to be either a Manhattan arc (line segment with slope ± 1 [2]) or a horizontal line segment or a vertical line segment. The following properties of feasible region will be proved after we present the merging of two feasible region.

Theorem 1 *Each feasible region exhibits the following two properties:*

1. **Octilinear Property:** *Each feasible region is an* octilinear *convex polygon, i.e. the boundary of the region is defined by octilinear line segments.*
2. **Path-Length Property:** *Consider any two points p and q on the same line segment on the boundary of a SDFR,*
   (a) *If the line segment is a Manhattan arc, $SPL(p) = SPL(q)$ and $LPL(p) = LPL(q)$.*
   (b) *If the line segment is horizontal or vertical, the changes of skew (due to interaction of different SPLs and LPLs) along the segment divide the line segment into three contiguous intervals (one or two intervals may be empty): skew decreases, skew remains equal and skew increases. Along each interval, the SPLs or LPLs are strictly increasing or strictly decreasing. The difference between the SPLs (or LPLs) between two points on the same interval is equal to the distance between the points.*

□

Graphically, the path-length property states that along a vertical or horizontal line segment of the boundary, the SPL and LPL behave as in Figure 4.

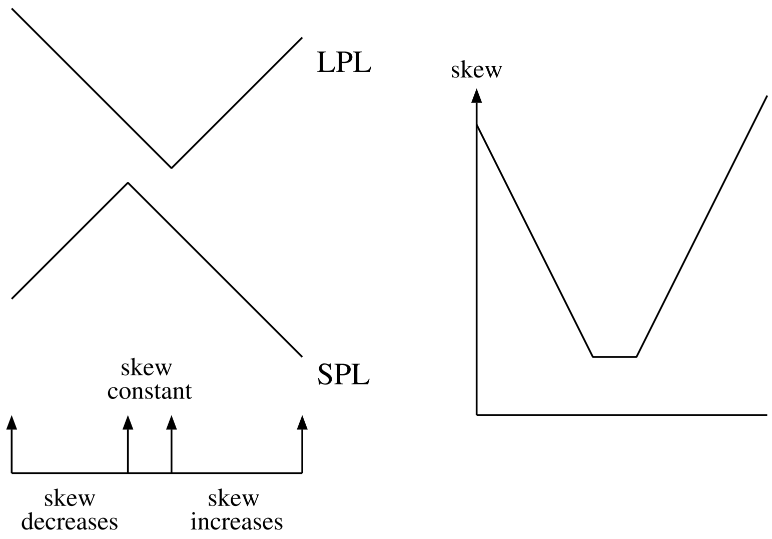


Figure 4: Path-length Property

Theorem 2 Non-overlapping Property: *In each iteration, the SDFRs of the roots of the trees are non-overlapping, and only the boundaries of the SDFRs may touch each other.*

Proof: This is trivially true at the beginning of the algorithm when the roots of the trees are the pins. Consider merging of two roots u and v to yield the new root w at the k-th iteration. Let V_k denote the collection of SDFRs of roots at the end of the k-th iteration. We assume that u is the nearest neighbor of v. Note that $SDR(u, v)$ does not overlap with any SDFR in $V_{k-1} - \{SDFR(u), SDFR(v)\}$. Otherwise, u cannot be the nearest neighbor of v. Since $SDFR(w)$ lies completely within $SDR(u, v)$, $SDFR(w)$ does not overlap with SDFRs in $V_{k-1} - \{SDFR(u), SDFR(v)\}$. We only have to show that $SDFR(w)$ does not overlap with SDFRs of other new roots created at the k-th iteration.

Consider $SDS(u)$ and $SDS(v)$ that define $SDR(u, v)$. Let p and q be any two points on $SDS(u)$ and $SDS(v)$, respectively, such that the $d(p, q) = d(u, v)$. Let $BB(p, q)$ be the smallest bounding box containing p and q. Clearly, $BB(p, q)$ is within $SDR(u, v)$. Since u and v cannot merge with other vertices, any new SDFRs that may overlap with $SDFR(w)$ are due to merging of other roots, say u' and v'. Similarly, let p' and q' be any two points on $SDS(u')$ and $SDS(v')$ such that $d(u', v') = d(p', q')$.

Clearly, the smallest bounding box $BB(p', q')$ cannot contain p or q. Otherwise, $p'q'$ cannot be an edge in the nearest neighbor graph. Without loss of generality, there are only three possible cases as shown in Figure 5(a)–(c) for the new SDFRs to overlap, assuming that $p = (0, 0)$ and $q = (x_q, y_q)$ with $0 \leq y_q \leq x_q$. Since p is the nearest neighbor of q, $p' = (x_{p'}, y_{p'})$ and $q' = (x_{q'}, y_{q'})$ can only be on or outside the rectilinear circles (dash-line squares rotated 90°), each with radius $d(p, q)$, centered at p and q, respectively.

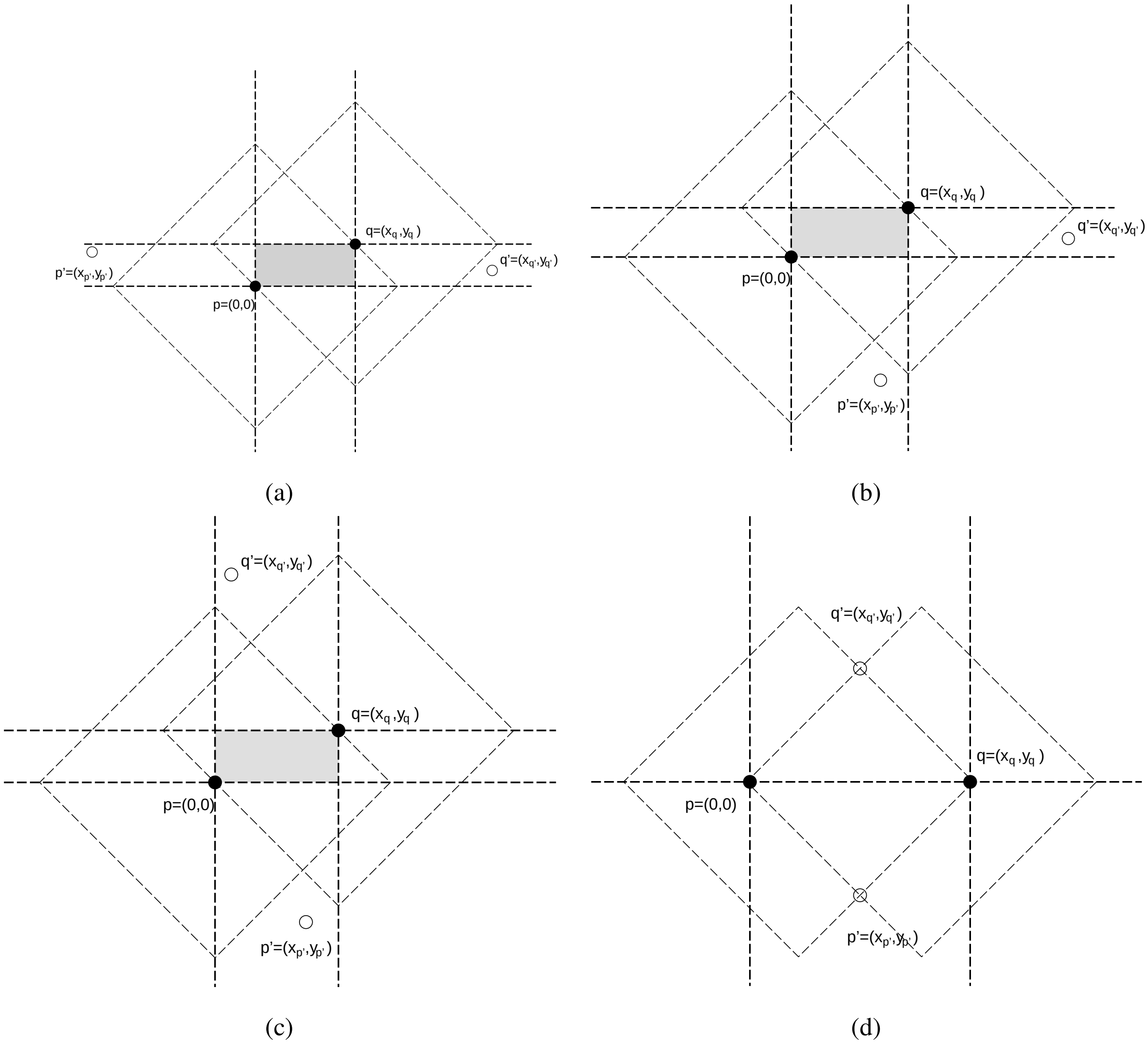


Figure 5: Overlapping of two new SDFR's

In Figure 5(a), $d(q, q') = x_{q'} - x_q + y_q - y_{q'} \leq d(p, q') < d(p', q')$. Similarly, $d(p, p') < d(p', q')$. Therefore, $u'v'$ cannot be an edge in the nearest neighbor graph, contradicting the fact that they are merged.

In Figure 5(b), if $y_{q'} = 0$ or $x_{p'} = x_q$, the non-overlapping property is satisfied since the SDRs only touch each other. Consider $y_{q'} > 0$ and $x_{p'} < x_q$. Since $BB(p', q')$ contains the point $r = (x_q, 0)$, $d(p', q') = d(p', r) + d(r, q')$. Since $d(p', r) \geq x_q$,

$$
\begin{aligned}
d(p', q') &\geq x_q + (x_{q'} - x_q + y_{q'}) \\
&= x_{q'} + y_{q'}.
\end{aligned}
$$

However, since $x_q \geq y_q$,

$$
d(q, q') = x_{q'} - x_q + y_q - y_{q'}
$$

$$\begin{aligned} &< x_{q'} + y_{q'} \\ &\leq d(p', q'). \end{aligned}$$

Similarly, since $d(r, q') > x_q$,

$$\begin{aligned} d(p', q') &= d(p', r) + d(r, q') \\ &> (x_q - x_{p'} - y_{p'}) + x_q \\ &> x_q - y_{p'}. \end{aligned}$$

However, since $x_q > x_{p'}$,

$$\begin{aligned} d(p, p') &= x_{p'} - y_{p'} \\ &< d(p', q'). \end{aligned}$$

Again, $u'v'$ is not a nearest neighbor edge.

Consider the configuration in Figure 5(c). We have

$$\begin{aligned} d(p', (x_{p'}, 0)) &\geq d(p, q)/2 \\ d(q', (x_{q'}, y_q)) &\geq d(p, q)/2 \\ \min(d(p, (x_{p'}, 0)), d(q, (x_{p'}, 0))) &\leq d(p, q)/2 \\ \min(d(q, (x_{q'}, y_q)), d(p, (x_{q'}, y_q))) &\leq d(p, q)/2 \end{aligned}$$

Therefore, $d(p', q') = d(p', (x_{p'}, 0)) + d(q', (x_{q'}, y_q)) + y_q + |x_{q'} - x_{p'}|$ which is strictly greater than $d(p, p')$ and $d(q, q')$, unless $y_q = 0$ and $x_{q'} = x_{p'}$ as shown in Figure 5(d).

Hence, except for the case in Figure 5(d) (a special case of Figure 5(c)), $p'q'$ is not a nearest neighbor edge if $BB(p', q')$ and $BB(p, q)$ overlap (not just touch). In the special case of Figure 5(d), the intersection of the SDRs (a vertical line segment $p'q'$ with a horizontal line segment pq) is a single point and it happens only if u, v, u' and v' are single-point SDSs, i.e. $u = \{p\}$, $v = \{q\}$, $u' = \{p'\}$ and $v' = \{q'\}$. Therefore, the boundaries of the SDFRs may only touch each other.

□

Suppose we modify the BST/DME algorithm slightly such that the roots are merged according to a given routing topology, i.e. selection of roots to merge is fixed instead of determined during execution of the algorithm. Hence, it is possible that for the SDFRs to overlap. We first observe the followings:

1. The SDFRs are rectangular: This can be proved easily by induction. The SDFR of two pins, say u and v, is the smallest rectilinear bounding box $BB(u, v)$ of the two points. Consider merging at a higher level involving SDFRs which are rectangular in shape. If the two SDFRs $SDFR(u)$ and $SDFR(v)$ overlap (Figure 6(a)), the new SDFR $SDFR(w)$ is formed by intersecting $SDFR(u)$ and $SDFR(v)$ which is rectangular. Otherwise, the $SDFR(w) = SDR(u, v)$ which is again rectangular (Figures 6(b) and (c)).

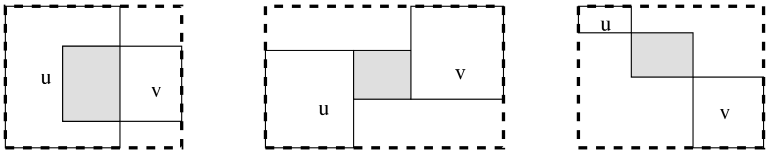
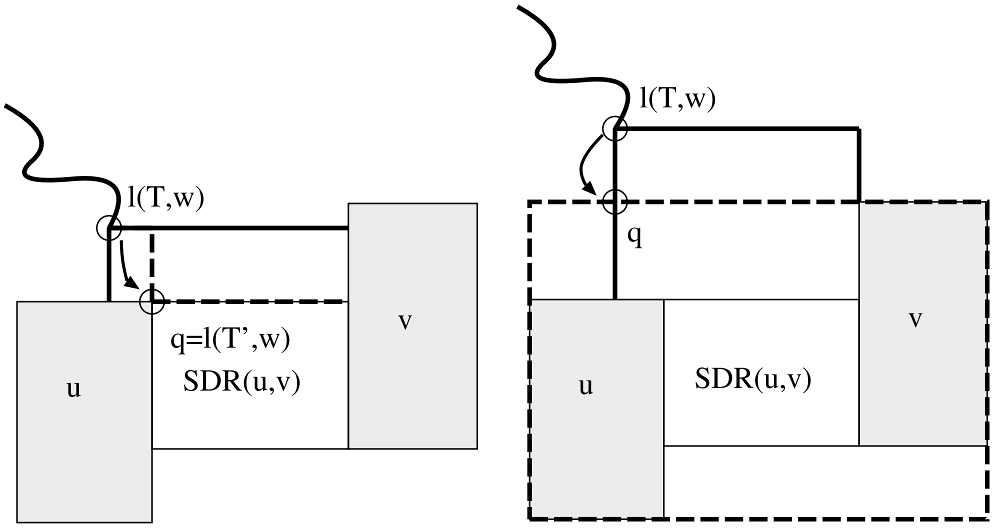


Figure 6: Possible configurations $SDFR(u)$ and $SDFR(v)$



(a)

(b)

Figure 7: (a) $l(T, w)$ lies within $BB(SDFR(u), SDFR(v))$, (b) $l(T, w)$ lies outside $BB(SDFR(u), SDFR(v))$, (b)

2. Given $SDFR(w)$, the minimum merging cost for w $(= d(SDFR(u), SDFR(v))$ where u and v are children of w) is always achievable for any placement $l(w) \in SDFR(w)$ by applying the embedding rule such that $d(l(u), l(w))$ and $d(l(v), l(w))$ are minimal with the condition that $l(u) \in SDFR(u)$ and $l(v) \in SDFR(v)$.

Lemma 1 *Given a set of sinks $\mathcal{N}$ with locations $l(\mathcal{N})$, let $R(\mathcal{N})$ be the given routing topology. Let T be the minimum-cost BST defined by the given topology. Let w be an internal node with children u and v and $l(T, w)$ is the placement of w in T. Suppose the subtrees T_u and T_v can be generated optimally by the DME/BST algorithm. Then another tree T' such that $l(T', w) \in SDR(u, v)$ can be constructed without incurring extra cost.*

Proof: Without loss of generality, there are three possible configurations of $SDFR(u)$ and $SDFR(v)$ as shown in Figure 6. The smallest bounding box containing $SDFR(u)$ and $SDFR(v)$ are depicted as bold dash-line. It can be verified that if $l(T, w) \notin SDFR(w)$ lies within the smallest bounding box $BB(SDFR(u), SDFR(v))$, then we can move $l(T, w)$ to another location $q \in SDFR(w)$ such that $d(l(T, w), q) = \min_{r \in SDFR(w)} d(l(T, w), r)$ (Figure 7(a)) without incurring extra cost. Effectively, we have constructed a new tree T' with $l(T', w) \in SDR(u, v)$ without incurring extra cost.

If $l(T, w)$ lies outside $BB(SDFR(u), SDFR(v))$ (Figure 7(b)), then we can move $l(T, w)$ to the nearest point q on the boundary of $BB(SDFR(u), SDFR(v))$ and reduce the cost by the distance $d(q, l(T, w))$ we move (Figure 7(b)). Therefore, T

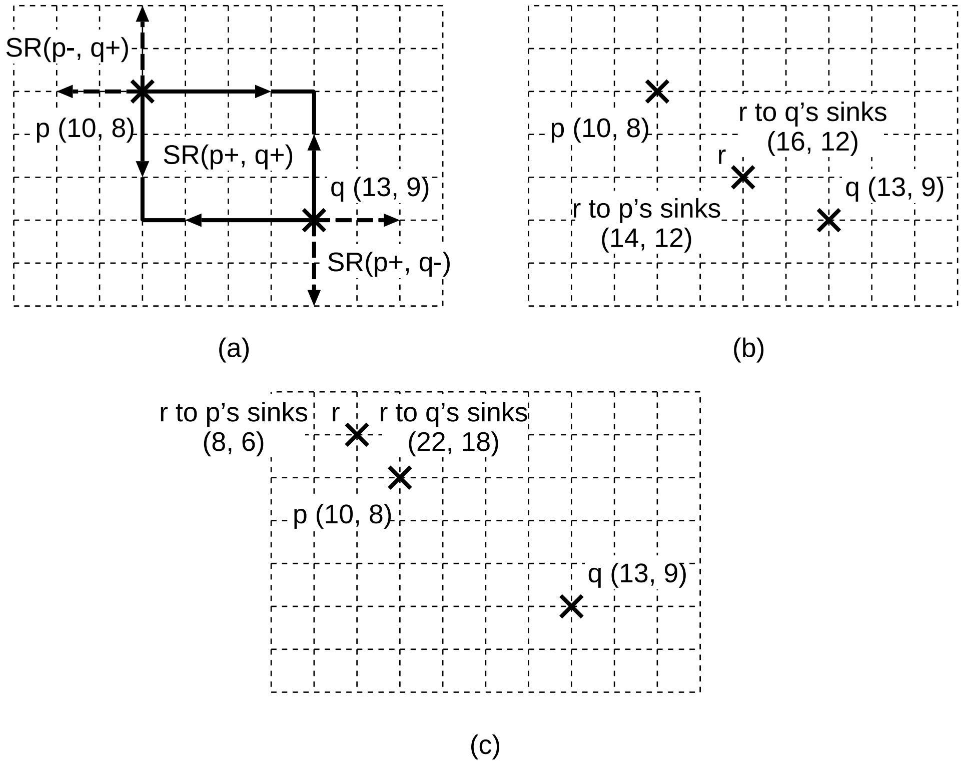


Figure 8: (a) The signed regions defined by p and q. (b) $sd(p, r) = d(p, r)$ and $sd(q, r) = d(q, r)$. (c) $sd(p, r) = -d(p, r)$ and $sd(q, r) = d(q, r)$. Each pair of co-ordinates represents (LPL, SPL).

is not an optimal tree.

□

By induction and using Lemma 1, we can now state the following result:

Theorem 3 Optimality Property: *The BST/DME algorithm is optimal for unbounded path-length skew for a given routing topology.*

3.4 Merging of SDFRs

The SDR between two SDFRs can be defined by the corresponding pair of SDSs (defined in Section 3.1). Due to the octilinear property of the SDFRs, the pair of SDSs are either a pair of parallel Manhattan arcs, or horizontal line segments or vertical line segments. For simplicity and clarity, we will first illustrate merging of two points. Moreover, we introduce the notion of *signed distance* between two points.

Consider any two points p and q. The pair of points defines three *signed regions*: $SR(p^+, q^+)$, $SR(p^+, q^-)$ and $SR(p^-, q^+)$, collectively referred to as $SR(p, q)$ (see Figure 8(a)). Now, consider a third point r. We denote the signed distance of r from p and q by $sd(p, r)$ and $sd(q, r)$, respectively. If r is in the $SR(p^+, q^+)$ region, then r is of positive distance from both p and q,

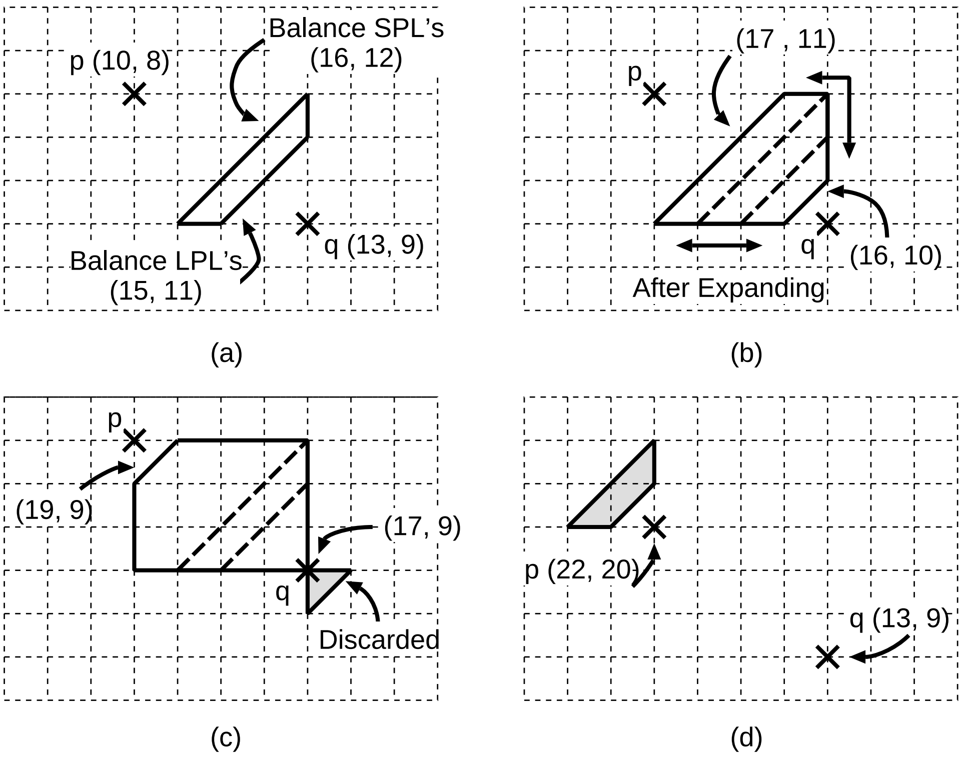


Figure 9: Finding FR of two points p and q: (a) Compute $CFR(r)$ by balancing the SPLs and LPLs. (b) Expand $CFR(r)$ for a skew bound of 6 units. (c) Expand $CFR(r)$ for a skew bound of 10 units. (d) FR between p and q for a skew bound of 4 units lies outside of the SDR. Each pair of co-ordinates represents (LPL, SPL).

i.e. $sd(p, r) = d(p, r)$ and $sd(q, r) = d(q, r)$ (Figure 8(b)). If r is in the $SR(p^-, q^+)$, then r is of negative distance from p and positive distance from q, i.e. $sd(p, r) = -d(p, r)$ and $sd(q, r) = d(q, r)$ (Figure 8(c)).

Suppose r is the clock entry point to p and q. Then the *signed* LPL (or SPL) from r to any sinks of p is $LPL(p) + sd(p, r)$ (or $SPL(p) + sd(p, r)$). We say that signed $LPL(r)$, denoted $SLPL(r)$, is $\max(LPL(p) + sd(p, r), LPL(q) + sd(q, r))$ and signed $SPL(r)$, denoted $SSPL(r)$, is $\min(SPL(p) + sd(p, r), SPL(q) + sd(q, r))$. The *feasible region* (FR) of r under the signed distance metric is defined to be $FR(r) = \{s \mid s \in SR(p, q), SLPL(s) - SSPL(s) \leq B\}$. The *core* of the FR (or CFR), denoted $CFR(r)$, is defined to be $CFR(r) = \{s \mid s \in SR(p, q), SLPL(s) - SSPL(s) = \max(skew(p), skew(q))\}$.

$FR(r)$ is computed in two steps:

(i) Compute $CFR(r)$ which is bounded by two line segments within $SR(p, q)$ such that the two equalities $sd(p, r) + LPL(p) = sd(q, r) + LPL(q)$ and $sd(p, r) + SPL(p) = sd(q, r) + SPL(q)$ hold.

(ii) Define $slack(p, q) = B - \max(skew(p), skew(q))$. Expand $CFR(r)$ by $slack(p, q)/2$ units towards both p and q.

For example, Figure 9(a) shows $CFR(r)$ after merging p and q. If $B = \max(skew(p), skew(q))$, then $CFR(r) = FR(r)$. In Figure 9(b), the skew bound B is 6 units and $slack(p, q) = 2$. We can therefore expand $CFR(r)$ by one unit towards both p and q. It is possible to have part or all of the FR outside of the SDR (Figure 9(c) and (d)).

We shall now present the computation of the FR due to merging of two SDFRs (or effectively, SDSs). Note that we place a restriction on the merging operation such that a point p on the first SDS, say $SDS(u)$, can only be merged with another q on the second SDS, say $SDS(v)$, if $d(p,q) = SDS(u,v)$. Since points along a Manhattan arc have the same SPL and LPL (path-length property), computation of FR for a pair of parallel Manhattan arcs is almost identical to that for a pair of points. Therefore, we shall focus on the merging of a pair of horizontal line segments (since merging of vertical line segments is symmetrical to that for horizontal line segments).

Due to the presence of intervals in a line segment (path-length property), we compute the FR (using the 2-step computation) between an end-point of an interval and the point directly opposite it on the other segment for all interval end-points on both segments. For example, Figure 10(a) shows the 5 FRs due to the end-points of the two horizontal line segments. Subsequently, we perform a walk to join the vertices of these FRs to produce the FR of the two horizontal line segments (Figure 10(b)). This is possible due to the path-length property along an interval.

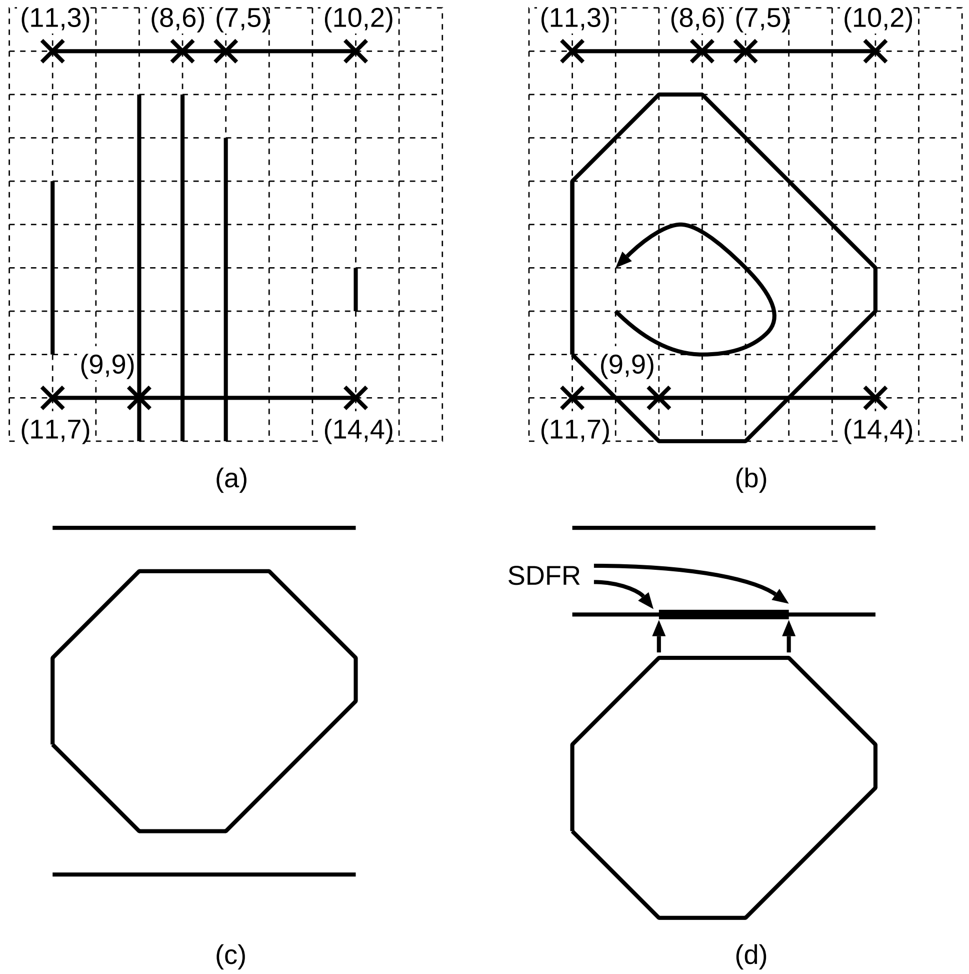


Figure 10: Finding FR of two horizontal line segments: (a) FR of all interval end-points. (b) A walk to produce the resultant FR. (c) FR strictly within the SDR. (d) FR lies outside of the SDR. Note that each pair of co-ordinates represents (LPL, SPL).

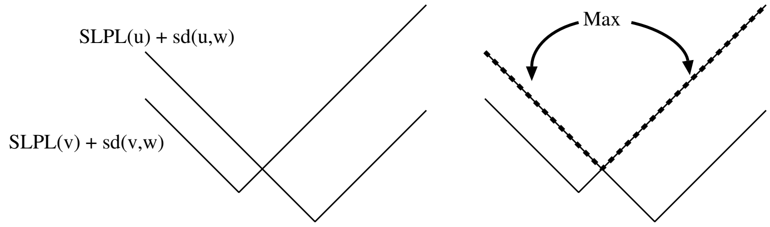


Figure 11: LPL of a horizontal line segment in the signed regions obeys the path-length property.

It is possible that the FR may (i) overlap with the SDR (Figure 9(a)–(c) and 10(b)–(c)), or (ii) lie outside of the SDR (Figure 9(d) and 10(d)). In case (i), we take the intersection of the FR and the SDR as the new SDFR. In case (ii), we take the segment of SDSs that is closest to the FR as the new SDFR. For example, p in Figure 9(d) is chosen to be the SDFR, and the bold horizontal line segment in Figure 10(d) is the new SDFR.

We shall now present the proof of Theorem 1. First, assuming that the FRs (and therefore, SDFRs) satisfy the pathlength properties. We want to show the following lemmas hold. Therefore, by induction, Theorem 1 holds.

Lemma 2 *Consider merging of two SDSs u and v. All horizontal and vertical segments within the signed regions defined by the SDSs satisfy the path-length property.*

Proof: Suppose we are merging two parallel diagonal line segments, say u and v. Let pq be a horizonal line segment in a signed region. Clearly, $sd(u, p) + sd(v, p) = d(u, v)$ and $sd(u, q) + sd(v, q) = d(u, v)$. Without loss of generality, let p be closer to u than v (or equivalently, q is closer to v than u). Consider any point r on pq, if $d(x, r) = \delta$, then the signed shortest delay from r to a sink in the subtree rooted at u is $sd(u, p) + \delta + SPL(u)$. Similarly, the signed longest delay from r to a sink in the subtree rooted at v is $sd(u, p) + \delta + LPL(u)$. Therefore, the signed longest and shortest delays from pq to a sink in the subtree rooted at u is increasing at a constant rate along the direction of $\vec{pq}$. In other words, if we plot the pq as the horizontal axis, then the signed delays are of slope $+1$. Similarly, the delays due to sinks in v are of slope -1. Taking the maximum of the two signed longest delay gives $SLPL(pq)$ which is, in general, $\vee$-shaped. Similarly, $SSPL(pq)$ is of $\wedge$-shape. Therefore, they satisfy the path-length property as shown in Figure 4.

Now, consider merging two horizontal (or vertical) line segments. Let u be the lower line segment and v be the upper line segment. We have pointed out that a point in one SDS will only merge with points in the other SDS if their distance is equal to the distance between the two SDSs. Therefore, a point on u will only merge with the point on v which is directly above it. Consider such a pair of points; if we draw a vertical line through the pair of points, it is obvious that the vertical line satisfies the path-length property since it is the same as merging two parallel Manhattan arcs. Now consider any horizonal line segment within the range of u and v. Let this line segment be w. Therefore, $SLPL(w) = \max(SLPL(u) + sd(u, w), SLPL(v) + sd(v, w))$ which still remains as $\vee$ (Figure 11). Similarly, $SSPL(w)$ remains as $\wedge$. Therefore, the horizontal line segment w also satisfies the path-length property.

□

We recall the notation of *merging segment* which was introduced in [2] for zero-skew routing. We define the merging segment of node v recursively as follows:

(i) If v is a pin, then $ms(v) = l(v)$.

(ii) If v is an internal node, then $ms(v)$ is the set of all placements $l(v)$ within the signed region which allow minimum merging cost (in term of signed distance metric).

Since we are concerned with merging of SDS only, we only have to consider merging of two parallel Manhattan arcs, two horizontal line segments, or two vertical line segments. It is therefore sufficient to prove only the first two cases since merging of vertical line segments is identical to merging of horizontal line segments.

Lemma 3 *The construction of the FR for merging of two parallel Manhattan arcs is correct. The resultant FR satisfies the octilinear and path-length properties.*

Proof: Let the $SDS(u)$ and $SDS(v)$ be two parallel Manhattan arcs to be merged. Clearly, the skew of w, the parent of u and v after merging, is at least the maximum of $skew(u)$ and $skew(v)$. We first show that the core is computed correctly. Let $ms_{SPL}(w)$ be the merging segment such that $sd(u, ms_{SPL}(w)) + SPL(u) = sd(v, ms_{SPL}(w)) + SPL(v)$ holds. For the equality to holds, any point p on $ms_{SPL}(w)$ must satisfy the following equation:

$$sd(u,p) \;=\; \frac{d(u,v) - (SPL(u) - SPL(v))}{2} \tag{1}$$

Similarly, $ms_{LPL}(w)$ is the merging segment such that $sd(u, ms_{LPL}(w)) + LPL(u) = sd(v, ms_{LPL}(w)) + LPL(v)$ holds. Note that to construct the locus of points, say p, of signed distance $sd(p, u)$ away from u is to take the intersection of the boundary of the tilted rectangle of radius $|sd(p, u)|$ with either the signed region $SR(u^+, \cdot)$ if $sd(p, u) >= 0$ or the signed region $SR(u^-, \cdot)$ if $sd(p, u) < 0$. Note that p is a Manhattan arc and it is parallel to both u and v. Also note that it is $sd(p, v) = d(u, v) - sd(p, u)$ away from v. Therefore, both $ms_{SPL}(w)$ and $ms_{LPL}(w)$ are Manhattan arcs parallel to each other.

Without loss of generality, we assume that $sd(u, ms_{SPL}(w)) \leq sd(u, ms_{LPL}(w))$ (or equivalently, $sd(v, ms_{SPL}(w)) \geq sd(v, ms_{LPL}(w))$). We want to show that for any point p such that $sd(u, ms_{SPL}(w)) \leq sd(u, p) \leq sd(u, ms_{LPL}(w))$, $skew(p) = \max(skew(u), skew(v))$. Now,

$$\begin{aligned}
SSPL(p) &= \min(sd(u,p) + SPL(u), sd(v,p) + SPL(v)) \\
&= SPL(v) + sd(v,p), \\
SLPL(p) &= \max(sd(u,p) + LPL(u), sd(v,p) + LPL(v)) \\
&= LPL(v) + sd(v,p).
\end{aligned}$$

Therefore, $skew(p) = SLPL(p) - SSPL(p) = skew(v)$. Since $skew(p) \geq \max(skew(u), skew(v))$, $skew(v) = \max(skew(u), skew(v))$. Therefore, the skew within the core is equal to the maximum skew of u and v.

Also, we know from Lemma 2 that $SSPL(p)$ decreases from $SPL(v) + sd(v, ms_{SPL}(w))$ to $SPL(v) + sd(v, ms_{LPL}(w))$ uniformly as we slide along a vertical or horizontal line from $ms_{SPL}(w)$ to $ms_{LPL}(w)$. Similarly, $SLPL(p)$ decreases from $LPL(v) + sd(v, ms_{SPL}(w))$ to $SPL(v) + sd(v, ms_{LPL}(w))$ uniformly.

Now, consider $sd(u, ms_{SPL}(w)) > sd(u, p)$. Let $sd(u, p) = sd(u, ms_{SPL}(w)) - \delta$.

$$\begin{aligned}
SSPL(p) &= sd(u,p) + SPL(u) \\
&= sd(u, ms_{SPL}(w)) - \delta, \\
SLPL(p) &= sd(v,p) + LPL(v) \\
&= sd(u, ms_{SPL}(w)) + SPL(u) + skew(v) + \delta.
\end{aligned}$$

Therefore, $skew(p) = skew(v) + 2\delta$. Hence, for $skew(p) \leq B$, $\delta \leq (B - skew(v))/2 = slack(u, v)$.

Similarly, consider $sd(u, ms_{LPL}(w)) < sd(u, p)$. Let $sd(u, p) = sd(u, ms_{LPL}(w)) + \delta$.

$$\begin{aligned}
SSPL(p) &= sd(v,p) + SPL(v) \\
&= sd(v, ms_{LPL}(w) - \delta + SPL(v), \\
SLPL(p) &= sd(u,p) + LPL(u) \\
&= sd(v, ms_{LPL}(w)) + LPL(v) + \delta.
\end{aligned}$$

Therefore, for $skew(p) \leq B$, $\delta \leq (B - skew(v))/2 = slack(u, v)$.

Therefore, construction of new FR for merging a pair of Manhattan arcs is correct as outlined and the contructed FR satisfies both path-length and octilinear properties.

□

Lemma 4 *The construction of the FR for merging of two parallel horizontal line segments is correct. The resultant FR satisfies the octilinear and path-length properties.*

Proof: Consider merging of two horizontal line segments, say u and v, both of which start at x-coordinate x_l and end at x-coordinate x_r. Let $PL_{u(x_l)}$ be a path length delay of the u at x_l and $PL_{v(x_l)}$ be a path length delay of v at x_l. If the path-length delays along the horizontal line segments are monotone, i.e. strictly increasing or decreasing, then pathlength delay of u at x-coordinate $x_l + \delta$, denoted $PL_{u(x_l+\delta)}$, is either $PL_{u(x_l)} + \delta$ or $PL_{u(x_l)} - \delta$ with $\delta \leq x_r - x_l$. Let d_{x_l} be the signed distance from u to balance $PL_{u(x_l)}$ and $PL_{v(x_l)}$, i.e. $d_{x_l} = (d(u, v) - (PL_{u(x_l)} - PL_{v(x_l)}))/2$ at x_l. Now, consider balancing the two montone path length delays at $x_l + \delta$, we have the four possible scenarios for the balancing signed distance at $x_l + \delta$, denoted $d_{x_l+\delta}$:

$$\begin{aligned}
PL_{u(x_l+\delta)} = PL_{u(x_l)} + \delta, PL_{v(x_l+\delta)} = PL_{v(x_l)} + \delta &\Rightarrow d_{l(x_l+\delta)} = d_{x_l} \\
PL_{u(x_l+\delta)} = PL_{u(x_l)} + \delta, PL_{v(x_l+\delta)} = PL_{v(x_l)} - \delta &\Rightarrow d_{l(x_l+\delta)} = d_{x_l} - \delta \\
PL_{u(x_l+\delta)} = PL_{u(x_l)} - \delta, PL_{v(x_l+\delta)} = PL_{v(x_l)} + \delta &\Rightarrow d_{l(x_l+\delta)} = d_{x_l} + \delta \\
PL_{u(x_l+\delta)} = PL_{u(x_l)} - \delta, PL_{v(x_l+\delta)} = PL_{v(x_l)} - \delta &\Rightarrow d_{l(x_l+\delta)} = d_{x_l}
\end{aligned} \tag{2}$$

Now suppose both two horizontal lines have monotone SPLs and LPLs. Then some possible configurations of FR are given in Figure 12. The remaining configurations can be obtained by reflecting the appropriate configuration in Figure 12 along the vertical axis. We will show the reasoning for the construction of Figure 12(c) in Figure 13. The rest of the configurations can be obtained in a similar fashion.

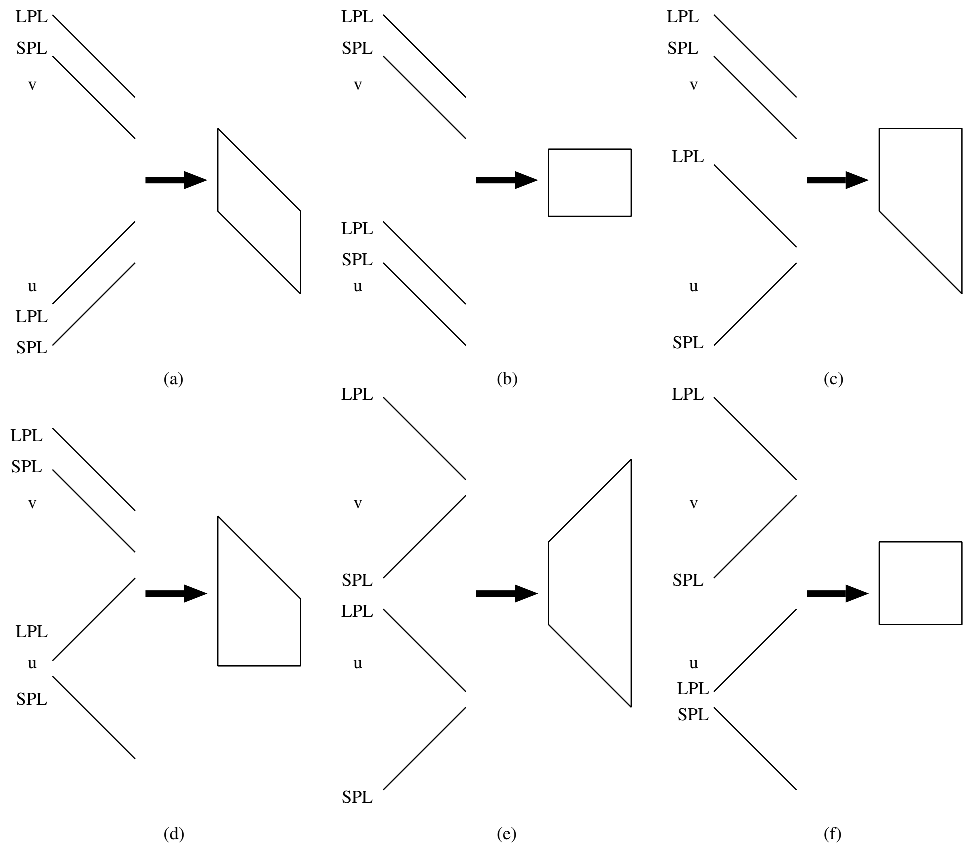


Figure 12: Merging of two horizontal segments u and v with monotone LPLs and SPLs.

We will "abuse" the notation of "CFR" in the rest of the proof. Let the "CFR" of two horizontal segments be the union of all the actual CFRs formed by merging points on u with the corresponding points on v. As before, the CFR of two points is formed by balancing the SPLs and LPLs of the two points. From Eqn. (2), the horizontal bound of the "CFR" is due to balancing of LPLs and the -1 slope of the "CFR" is due to balancing of SPLs. There are three possible cases:

(a) $skew(v) = LPL_v - SPL_v \geq LPL_u - SPL_u$: $slack(u, v)$ is constant (Figure 13(a)).

(b) $LPL_u - SPL_u > skew(v) \geq LPL_{u(x_r)} - SPL_{u(x_r)}$: Let q be the point with x-coordinate $x_l + \delta_q$ such that $skew(v) = LPL_{u(x_l+\delta)} - SPL_{u(x_l+\delta)}$. Since the delays are monotone, there is only one such point. From x_l to $x_l + \delta_q$, the slack $slack(u, v)$ increases monotonically at half the rate that the skew of u is decreasing. To the right of $x_l + \delta_q$, the slack remains constant.

(c) $skew(v) < LPL_{u(x_l+\delta)} - SPL_{u(x_l+\delta)}$ for $0 \leq \delta \leq x_r - x_l$: This is the same situation as for the interval between x_l and q in (b); the slack $slack(u, v)$ increases monotonically at half the rate that the skew of u is decreasing.

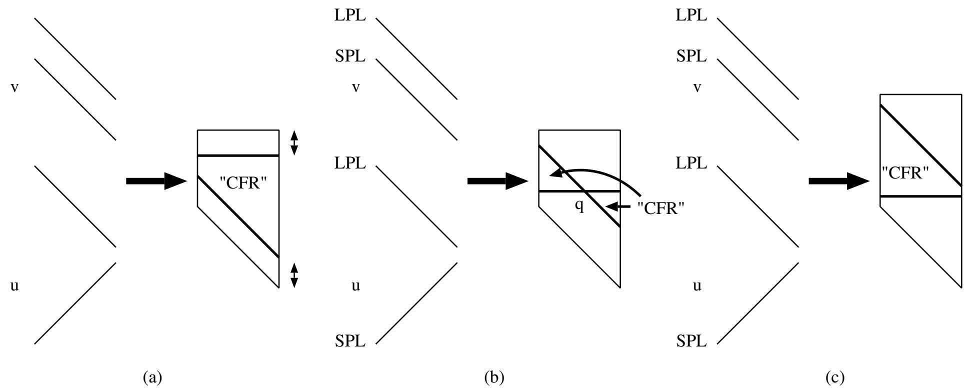


Figure 13: Merging of two horizontal segments u and v. Segment u has increasing LPL and decreasing SPL and segment v has decreasing LPL and SPL.

In all cases, the LPL along the Manhattan arc of the boundary is due to the $LPL(v)$ plus the signed distance of v from the Manhattan arc. Therefore, it is constant. Similarly, the SPL along the Manhattan arc of the boundary is due to $SPL(u)$ plus the signed distance of u from the Manhattan arc. Again, it is constant.

Note that the merging operation breaks the horizontal line segments into segments where the delays are monotone. Therefore, construction of new FR is correct.

□

Lemma 5 *All FRs are convex.*

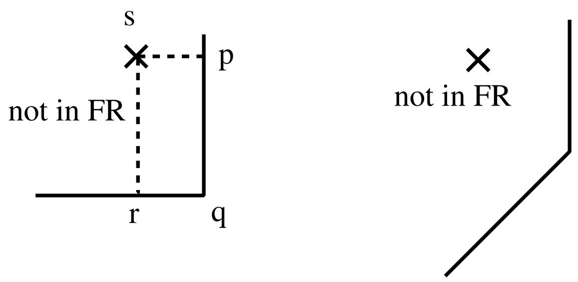


Figure 14: Convexity of the SDFRs

Proof: Suppose it is not, the configurations shown in Figure 14 are possible. We show only the proof for the configuration in Figure 14(a). The rest can be derived in a similar fashion. We can identify a rectangle formed by p, q, r and s as in Figure 14(a) with the property that $d(s,p) \neq d(p,q)$. Assume that $d(s,p) < d(p,q)$. Consider the line segment sp, it must satisfy the path-length property. Since p is in the FR, the LPL and SPL for points along sp must be increasing and decreasing monotonically, respectively, along the direction $\vec{ps}$, i.e. $SPL(s) = SPL(p) - d(s,p)$ and $LPL(s) = LPL(p) + d(s,p)$. Also, p has a skew which is equal to the bound B.

Similarly, consider any point along the line segment pq except q. Since all points to the left of pq except q are not in FR, they all have skew equals to B. Based on the path-length property along a vertical line, the SPL and LPL at q is either $SPL(p)+d(p,q)$ and $LPL(p)+d(p,q)$, respectively, or $SPL(p)-d(p,q)$ and $LPL(p)-d(p,q)$, respectively. By similar argument, $skew(r)$ also equals to B. Therefore $SPL(r)=SPL(q)+d(s,p)$ and $LPL(r)=LPL(q)+d(s,p)$, or $SPL(r)=SPL(q)-d(s,p)$ and $LPL(r)=LPL(q)-d(s,p)$.

Using the same arguments as before, the LPL and SPL for points along rs must be increasing and decreasing monotonically along the direction $\vec{rs}$, i.e. $SPL(s)=SPL(r)-d(p,q)$ and $LPL(s)=LPL(r)+d(p,q)$. We therefore obtain two different LPLs and SPLs for point s. This is a contradiction since the LPL and SPL for s is obtained by merging exactly one point in u and one point in v (both points and s lie on a vertical line).

□

4 Experimental Results

We have implemented the BST/DME algorithm in ANSI C for the Sun SPARC station environment. In our experiments, we tested the BST/DME algorithm on benchmark data prim1–prim2 [10] and r1–r5 [17] for k ranging from 1.5 to 4.0 in the bottom-up phase of the algorithm. Table 1 compares the ZST routing costs by the NN (Nearest Neighbor) algorithm from [7] with the routing costs of our BST/DME algorithm for different skew bounds. The reason that we only compare with [7] is because it outperforms other clock routing algorithms including [2, 6, 17]. Note that the NN algorithm can be improved slightly using the MD and ME algorithms in [7]. Moreover, [8] showed that wirelength can be further reduced by changing the topology after an initial topology is obtained. We are currently incorporating these enhancements in our BST/DME algorithm.

Skew Bound	Circuits						
	prim1 cost / skew	prim2 cost / skew	r1 cost / skew	r2 cost / skew	r3 cost / skew	r4 cost / skew	r5 cost / skew
0 ([7])	131210 / 0	312430 / 0	1331867 / 0	2590670 / 0	3317598 / 0	6779690 / 0	9889688 / 0
0 (BST/DME)	129105 / 0	311350 / 0	1288715 / 0	2556887 / 0	3316250 / 0	6714451 / 0	9874719 / 0
100	125480 / 100	301380 / 100	1287908 / 100	2547415 / 100	3308659 / 100	6648148 / 100	9857184 / 100
200	123695 / 200	292550 / 200	1279592 / 200	2526300 / 200	3289013 / 200	6619472 / 200	9808148 / 200
500	117905 / 500	276970 / 500	1267831 / 500	2513952 / 500	3232975 / 500	6536240 / 500	9649783 / 500
1000	112185 / 1000	267020 / 1000	1264303 / 1000	2483180 / 1000	3213398 / 1000	6395835 / 1000	9498362 / 1000
2000	108225 / 2000	256380 / 2000	1242024 / 2000	2430848 / 2000	3143449 / 2000	6267746 / 2000	9307250 / 2000
5000	106990 / 4920	253350 / 5000	1186527 / 5000	2337273 / 5000	2963036 / 5000	5972735 / 5000	8801182 / 5000
∞	105960 / 8050	249240 / 9980	1069802 / 61449	2096325 / 100260	2710362 / 122391	5382237 / 196276	8022978 / 162786

Table 1: Total wirelengths and skews of the clock routings generated by the BST/DME algorithm for benchmark circuits prim1-2 [10] and r1-5 [17].

In general, we see a decrease in total wirelengths as the skew increases. However, our results do not compare favorably with [7] when it comes to large circuits with small skew. We believe this is due to the computation of new SDFRs when the FRs lie outside of the SDRs (case (ii)). It is also due to this limitation that the algorithm is sub-optimal (for a given topology) when this algorithm is used for ZST routing. However, this approach is optimal (for a given topology) for unbounded skew.

5 Conclusion and Future Work

This paper presents a generalized DME algorithm, named BST/DME, which constructs BST by a bottom-up phase which creates a tree of SDFRs, followed by a top-down phase which determines the exact location of the clock entry points. Our clock routing algorithm can produce a set of routing solutions with skew and wirelength trade-off. We learned recently that an independent study of the bounded-skew clock routing problem will be reported in [11].

Most of the current clock routing algorithms first compute the clock routing tree topology and then carry out buffer insertion and wire sizing independently. Also, their emphasis is on achieving zero-skew at the expense of very high power dissipation. Our future plan is to develop a practical clock routing algorithm which carries out simultaneous topology generation, buffer insertion, and wiresizing for achieving bounded skew with minimum power dissipation.

References

[1] H. Bakoglu, J. T. Walker and J. D. Meindl, "A symmetric clock-distribution tree and optimized high-speed interconnections for reduced clock skew in ULSI and WSI circuits," *Proc. IEEE ICCD*, Port Chester, Oct. 1986, pp. 118–122.

[2] 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), Nov. 1992, pp. 799–814.

[3] J. D. Cho and M. Sarrafzadeh, "A buffer distribution algorithm for high-speed clock routing," *Proc. ACM/IEEE Design Automation Conf.*, Jun. 1993, pp. 537–543.

[4] J. Chung and C.-K. Cheng, "Skew sensitivity minimization of buffered clock tree," *Proc. Int'l Conf. on Computer-Aided Design*, 1994, pp. 280–283.

[5] J. Cong and C.-K. Koh, "Minimum-Cost Bounded-Skew Clock Routing," *UCLA Computer Science Department Technical Report #950003*, January 1995.

[6] 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.

[7] M. Edahiro, "Minimum path-length equi-distant routing," *IEEE Asia-Pacific Conference on circuits and Systems*, Dec. 1992, pp. 41–46.

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

[9] L. P. P. P. van Ginneken, "Buffer placement in distributed RC-tree networks for minimal Elmore delay,", *Proc. Int'l Symposium on Circuits and Systems*, 1990, pp 865–868.

[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] J. H. Huang, A. B. Kahng and C.-W. A Tsao, "On the Bounded-Skew Routing Tree Problem", to appear in *Proc. ACM/IEEE Design Automation Conf.*, San Francisco, June 1995.

[12] A. B. Kahng and C.-W. A. Tsao, "Planar-DME: Improved planar zero-skew clock routing with minimum pathlength delay," *Proc. European Design Automation Conference*, 1994.

[13] A. B. Kahng and C.-W. A. Tsao, "Low-cost single-layer clock trees with exact zero Elmore delay skew," *Proc. IEEE Int'l Conf. on Computer-Aided Design*, 1994, pp. 213–218.

[14] S. Lin and C. K. Wong, "Process-variation-tolerant clock skew minimization," *Proc. IEEE Int'l Conf. on Computer-Aided Design*, 1994, pp. 284–288.

[15] S. Pullela, N. Menezes, J. Omar, and L. Pillage, "Skew and delay optimization for reliable buffered clock trees," *Proc. IEEE Int'l Conf. on Computer-Aided Design*, 1993, pp. 556–562.

[16] M. Seki, K. Inoue, K. Kato, K. Tsurusaki, S. Fukasawa, H. Sasaki, and M. Aizawa, "A Specified Delay Accomplishing Clock Router Using Multiple Layers," *Proc. IEEE Int'l Conference on Computer-Aided Design*, 1994, pp. 289–292.

[17] R. S. Tsay, "Exact zero skew," *Proc. IEEE Int'l Conference on Computer-Aided Design*, 1991, pp. 336-339.

[18] Q. Zhu and W. W.-M. Dai, "Perfect-balance planar clock routing with minimal path-length," *IEEE/ACM Int'l Conference on Computer-Aided Design*, 1992, pp 473–476.