A Fast Algorithm for Context-Aware Buffer Insertion *

Ashok Jagannathan, Sung-Woo Hur and John Lillis
Dept. of EECS, University of Illinois at Chicago
{ajaganna,shur,jlillis}@eecs.uic.edu

Abstract

We study the problem of performing buffer insertion in the context of a given layout. In a practical situation, there are restrictions on where buffers may be inserted while routing over such regions may be possible (e.g., due to the presence of macro cells). As a result, it is desirable to perform route planning and buffer insertion simultaneously. Further it is necessary that such an algorithm be aware of the tradeoff between cost (e.g. total capacitance) and delay. In this context we propose the Delay Reduction to Cost Ratio (DRCR) problem and present a fast algorithm for the same. Solutions identified by the algorithm are characterized with respect to the overall cost vs. performance tradeoff curve. Computational experiments demonstrate the viability of the approach.

1 Introduction

In the Deep Submicron era, delay optimization for high performance interconnects has become of fundamental interest. In this context, buffer insertion has been proven to be a powerful technique. Much of the past work (e.g., [2]), on buffer insertion, while of fundamental interest, has focused on idealized situations, where buffers can be inserted at arbitrary positions on the routing area. However, as pointed out in the recent work of Zhou and Wong [1], in practice such optimizations must occur in the context of, for example, a floorplan where there may be pre-placed macro cells which can be routed over, but which preclude the insertion of buffers in that region.

This is illustrated in Figure 1. The point illustrated in the figure is that it now seems critical to consider both the global route of a signal and the buffer insertion problem simultaneously since we may, for instance, have to detour not merely around congested routing regions, but also to "pick up" a buffer if necessary.

This kind of context-aware buffer insertion problem was studied by Zhou and Wong in [1] in the context of the two-pin problem. Their main result was a labeling algorithm which finds the minimum delay buffered source to sink path. They also discussed several natural and more general formulations which capture a cost vs. performance tradeoff (e.g. minimizing congestion subject to a delay constraint). Such formulations were shown to be NP-hard and could in fact be viewed as instances of the classical shortest weight-constrained path problem [3]. A pseudo-polynomial algorithm for such formulations was also presented, which finds the set of all source-to-sink paths that lie on the cost vs. delay tradeoff curve (a similar algorithm appears in [4]).

*This work was supported by the Design Automation Conference Scholarship Program.

DAC 2000, Los Angeles, California

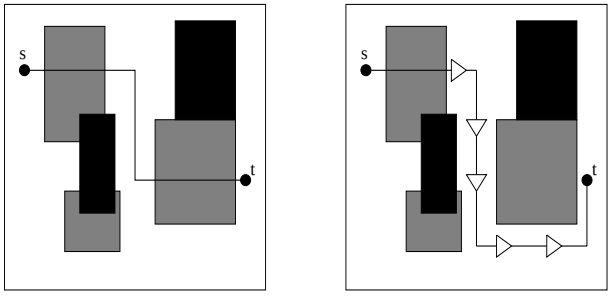

Figure 1: *An illustration of buffer insertion taking pre-placed macro cells into consideration; the black boxes represent portions of the routing area where neither buffering nor wiring is possible; the grey boxes represent areas where wires can be routed, but buffers cannot be inserted.*

This paper also focuses on the two-pin problem with an emphasis on the tradeoffs between cost (e.g., total capacitance) and delay. The ability to capture such tradeoffs is crucial in practice since the cost overhead of min-delay solution tends to be excessive.

Toward this end we propose and characterize a new formulation, called the *Delay Reduction to Cost Ratio Maximization* **(DRCR)** problem. Given a set of candidate buffer insertion locations and their candidate connections modeled as a directed graph, and a reference delay value D_{ref}, we wish to maximize the ratio $\frac{D_{ref}-d(p)}{g(p)}$ over all source-to-sink paths p, where $g(p)$ and $d(p)$ are the path cost and delay respectively – i.e., we maximize the ratio of the reduction in delay to the corresponding cost.

A nice property of this formulation is that it is completely independent of the cost and delay models used to estimate the cost and delay associated with the candidate edges in terconnecting the buffers. For example, we are not restricted to using the total capacitance as the cost and Elmore delay model [6] for the interconnect delay (though for simplicity in our experiments we have used Elmore). By the same token, the cost associated with a particular candidate connection and buffer is also flexible; while total estimated capacitance is a natural measure, heuristic measures relating to congestion and buffer availability are also plausible.

It is suggested that the DRCR is a natural composite objective function capturing the tradeoff between cost and delay. Our main contribution is a fast polynomial time algorithm for this problem. It is then natural to consider the relation between solutions of the DRCR problem and other formulations (in particular cost minimization subject to a delay constraint). Toward this end the problem is characterized with respect to all source-to-sink paths that lie on the cost vs. delay tradeoff curve (i.e., non-dominated paths). A subset of these paths forms the *Lower Convex Hull* (LCH); the LCH is essentially the points on the lower-left of the tradeoff curve. It is shown that a variant of the algorithm can efficiently identify any point on the LCH. Thus we have a tradeoff: the expense of using the fast algorithm presented is that we are no longer able

to identify paths which lie *off* the LCH while a comparatively slow pseudo-polynomial algorithm is able to identify such points. We argue that in practice this is not a major sacrifice since paths off the LCH tend to make less sound engineering choices. Computational experiments show the algorithms to be extremely efficient. Thus we believe the proposed algorithm will become a valuable tool in the early stages of design where buffers must be allocated and topological buffer-to-buffer routes determined.

2 Preliminaries

2.1 Delay Models

Though the graph model we utilize allows any desired technique to estimate the delay from the input of a buffer to the input of the next, we review some of the basic RC delay models for the purpose of discussion.

Let r_0 and c_0 represent the unit length resistance and capacitance respectively of a wire. Then for a wire e of length l_e, the resistance r_e and the capacitance c_e are given by

$$r_e = r_0 \cdot l_e \quad \text{and} \quad c_e = c_0 \cdot l_e$$

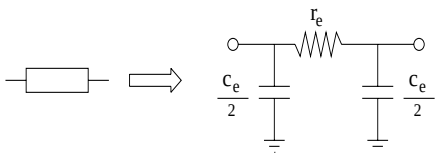

Figure 2: *RC delay modeling for a wire.*

Figure 2 shows the common RC model of a wire. In the Elmore delay model [6], the delay D_e of a wire e driving a load C is given by

$$D_e = r_e \cdot \left(\frac{c_e}{2} + C\right)$$

Similarly, if b is a buffer with intrinsic delay d_b, output resistance r_b and a loading capacitance C, then the delay of the buffer D_b is given by

$$D_b = d_b + r_b \cdot C$$

2.2 Dominance Property

Since paths are characterized by two parameters g and d, they may be compared with a partial order. A path $p : s \rightsquigarrow t$ is said to be non-dominated, if all other paths $p' : s \rightsquigarrow t$, have $g' \geq g$ or $d' \geq d$. – i.e., the set of non-dominated paths are those on the cost vs. delay tradeoff curve intrinsic in the problem.

3 Buffer Graph

We model the context-aware buffer insertion problem by a directed graph in which nodes represent buffers and edges represent the candidate connections between buffers. A path in such a graph represents a sequence of buffers inserted by virtue of the nodes on the path. To avoid confusion, we emphasize that the buffer selection is *implicit* in the node – there is no need to explicitly determine the type of buffer inserted at a node; this is determined by the graph itself (see below).

Each edge e is annotated with two labels: g_e is the cost associated with taking the edge (perhaps including the routing cost and the cost of the destination buffer) and d_e is the delay from the input of the source buffer to the input of the destination. Figure 3 shows two buffers a and b and their candidate interconnection modeled as a graph, where the cost and the delay of the edges are computed under the Elmore delay model. Recall that the cost and delay of the edge are computed from the input of buffer a to the input of buffer b.

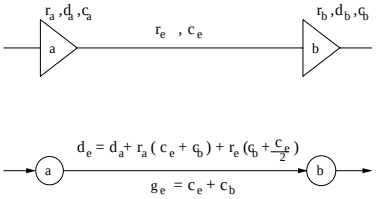

Figure 3: *Illustration of the transformation of a wire connecting two buffers into its corresponding graph model under the Elmore delay model.*

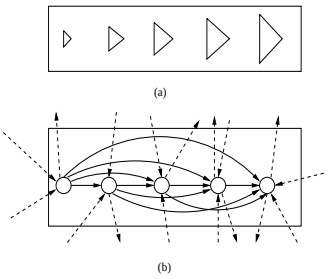

Figure 4: *Illustration of cascading buffers within a buffer station; solid lines represent connections within the buffer station and broken lines represent edges to and from outside the buffer station. (a) A buffer station with buffers of different sizes and (b) the corresponding graph model. Note that the edges in the graph are directed from smaller buffers to larger ones.*

It is often the case that each buffer station has a set of buffers of different sizes to facilitate cascading of buffers within the buffer station for improved performance. An illustration of a buffer station with multiple buffer sizes and its corresponding graph model is shown in Figure 4. Note that buffers within a buffer station are cascaded only in increasing order of their size – i.e., edges go only from smaller buffers to larger ones within the same buffer station. Thus the graph model naturally captures this situation.

Such an abstract graph model has several benefits. Of foremost importance is that it is completely independent of the models used to estimate the delay and cost of any edge in the graph – i.e., any desired means can be used to estimate the input-to-input delay and alternative cost measures can be applied depending on the situation.

For instance, more sophisticated interconnect and gate delay models can be used to model the delay associated with every edge. We are not limited for example to using Elmore delay or considering only the total capacitance as our cost measure.[1] Moreover, this graph model points out the intimate relationship between the context-aware buffer insertion problem and the shortest weight constrained problem in [3].

As stated earlier, a path in such a graph represents not only the wiring route to be taken, but by virtue of the vertices on the path, the buffers to be inserted. Figure 5 presents a complete example of a set of buffer stations, the corresponding graph model, and two $s \rightsquigarrow t$ paths in the graph.

[1] As presented, we do require that the delay be independent of the previous stage; however, if this is a serious issue, it can be modeled via a further transformation of the graph (at the expense of a large graph).

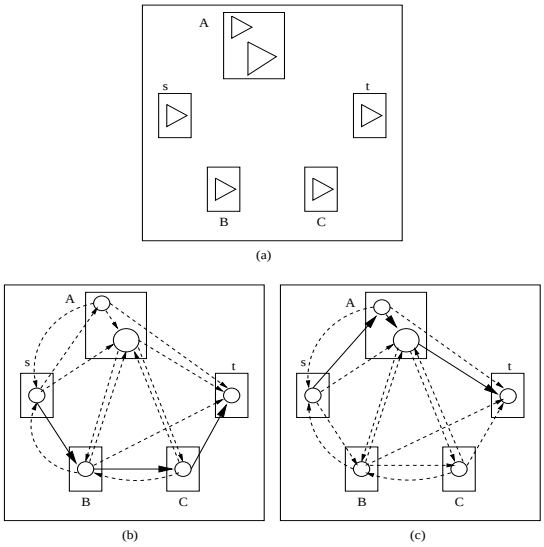

Figure 5: *Illustration of a set of buffer stations modeled as a graph ; all lines represent edges while the solid lines represent source-to-sink paths. (a) instance of a set of buffer stations with finite buffering resources (b) a simple $s \rightsquigarrow t$ path passing though various buffer stations (c) another $s \rightsquigarrow t$ path in which buffers are cascaded at the intermediate buffer station A.*

A naive graph construction method results in a complete graph – i.e., there is one vertex for each size of buffer inside every buffer station and an edge connecting every pair of vertices. However, in practice there is a threshold on the interconnect length beyond which a buffer must be inserted and thus it is sufficient if a buffer is connected to only its neighbors which lie within a specific technology dependent distance. By taking this factor into account during the construction process, the graph size can be reduced considerably.

4 Problem Formulations

Given such a graph theoretic interpretation of the problem, the traditional constrained optimization problem can be stated as follows (recall this problem is NP-hard).

Formulation 1 ***Given:*** *A directed graph $G = (V, E)$, where V represents the set of candidate buffer insertion locations, a buffer library B, each $e \in E$ is annotated with a cost g_e and delay d_e, a source terminal s with a driving resistance R_s, a sink terminal t and a delay bound d_{spec}.* ***Objective:*** *Find a path connecting s and t such that the total cost of the path is minimized subject to the delay not exceeding d_{spec}.*

The Delay Reduction to Cost Ratio Maximization(**DRCR**) problem is stated as follows.

Formulation 2 (*DRCR*) ***Given:*** *A directed graph $G = (V, E)$, where V represent buffers, each $e \in E$ is annotated with a cost g_e and delay d_e, a source terminal s with driving resistance R_s, a sink terminal t and a reference delay D_{ref}.* ***Objective:*** *Find a buffered path $p : s \rightsquigarrow t$ in G such that the ratio $\frac{D_{ref} - \sum_{e \in p} d_e}{\sum_{e \in p} g_e}$ is maximized.*

Note that the selection of the reference delay D_{ref} value will influence the optimal path; this issue is addressed in section 6.

The following subsection outlines a pseudo-polynomial labeling algorithm for solving the constrained optimization problem as in Formulation 1.

Labeling Algorithm

Since the labeling algorithm for Formulation 1 is not the focus of this work, we will not present the entire algorithm. We point the reader to [4] and [1]. We note however that the main idea is based on maintaining for each vertex u in the graph sets of non-dominated paths $P(u)$. A path is characterized by its cost and delay (g, d) and $(g, d) \in P(u)$ indicates that there exists a path from node u to the sink t (in a bottom-up approach) with cost g and delay d which is not dominated by any other u to t path. These sets are updated in what can be viewed as an extension of Dijkstra's algorithm by examining the solutions at neighboring vertices. At termination, the set $P(s)$ encodes all of the non-dominated paths from s to t. The algorithm is pseudo-polynomial because the the sets $P(u)$ are not bounded in size by a polynomial function of the graph size; rather their size depends on the values of the problem instance (delays and costs).

5 Delay Reduction to Cost Ratio Maximization

The **DRCR** problem is the focus of this work. Fortunately, the DRCR problem appears to be computationally easier than Formulation 1 while still capturing key cost vs. delay characteristics. We now present a strongly polynomial time algorithm solving DRCR.

The algorithm employs binary search on the optimal ratio and is similar in spirit to algorithms for the minimum time-to-profit cycle problem (see, e.g. [5]).

Recall that for a given value of the reference delay D_{ref}, our objective is to find a buffered path $p : s \rightsquigarrow t$ in G, such that the ratio $\frac{D_{ref} - \sum_{e \in p} d_e}{\sum_{e \in p} g_e}$ is maximized.

Let R_{max} represent this maximum ratio. Then

$$R_{max} = \frac{D_{ref} - \sum_{e \in p} d_e}{\sum_{e \in p} g_e} \tag{1}$$

for some path $p : s \rightsquigarrow t$ in G. We can rearrange the above equation as

$$R_{max} \sum_{e \in p} g_e + \sum_{e \in p} d_e = D_{ref} \tag{2}$$

The left hand side of (2) can be interpreted as the total length of the path $p : s \rightsquigarrow t$ in G, where all the edges $e \in G$ are relabeled as $w_e = R_{max}\, g_e + d_e$. The idea behind our algorithm is to start with a conjecture I for the value of R_{max}, and iteratively correct the value of I until we find the actual value of R_{max} which satisfies (2) corresponding to the given D_{ref}, and also the associated path p,which has this maximum ratio.

Starting with the initial conjecture I, for each edge $e \in G$, we assign edge weights $w_e = I\, g_e + d_e$, where g_e and d_e are the original cost and delay values associated with edge e. With this relabeled graph, we find the *shortest path* p from s to t. Clearly, the length of the path p is given by $\sum w_e = I \sum g_e + \sum d_e$, where $e \in p$.,i.e., the sum of the cost and delay values associated with all the edges in the path. One of the following three situations is possible.

- If the length of the path p is **equal** to D_{ref}, then (2) is satisfied and we have the current $I = R_{max}$ and also the corresponding path p, and we are done.

- If the length of the path is **less** than D_{ref}, then we increase the value of I, relabel the graph with the new I value and repeat the algorithm until we reach a value of I for which equation (2) is satisfied.
- If the length of the path is **greater** than D_{ref}, then we decrease the value of I, relabel the graph with the new I value and repeat the algorithm until we reach a value of I for which equation (2) is satisfied.

We use a binary search technique to probe the values of I. The idea is to find two values of I, namely I_{low} and I_{high} such that

$$I_{low} \sum_{e \in p_{low}} g_e + \sum_{e \in p_{low}} d_e \leq D_{ref} \text{ and} \quad (3)$$

$$I_{high} \sum_{e \in p_{high}} g_e + \sum_{e \in p_{high}} d_e \geq D_{ref} \quad (4)$$

where p_{low} and p_{high} are the shortest s to t paths in the graphs relabeled with I_{low} and I_{high} respectively.

Identifying I_{low} and I_{high} can be done as follows. For the initial value of the conjecture I, if the length of the shortest path in the relabeled graph is less than D_{ref}, then we repeatedly keep doubling I, till we find the two successive values I_{low} and I_{high} such that equations (3) and (4) are satisfied. On the other hand, if starting from the initial I, the length of the shortest path is greater than D_{ref}, we keep halving I until we find I_{low} and I_{high}.

Once we find I_{low} and I_{high}, we can do a binary search in the range $[I_{low}..I_{high}]$ to identify the final value of I which maximizes the delay reduction to cost ratio. The **DRCR** algorithm which is based on such a binary search technique is shown in Figure 6.

```
Algorithm DRCR
    Subroutine AssignWeights (G, I)
        For each e ∈ G,
            w_e = I g_e + d_e
    Main Routine
1   Find I_low and I_high
2   I ← (I_low + I_high)/2
3   AssignWeights(G, I)
4   P ← shortest s ⇝ t path in G.
5   while (D_ref ≠ length(P))
6       if (length(P) > D_ref)
7           I_high ← I
8       else
9           I_low ← I
10      endif
11      I ← (I_low + I_high)/2
12      AssignWeights(G, I)
13      P ← shortest s ⇝ t path in G.
14  endwhile
15  return P
```

Figure 6: *DRCR algorithm to solve the ratio maximization problem.*

The correctness of the algorithm follows from this discussion as stated in the following lemma (a formal proof appears in [7]).

Lemma 1 *Given D_{ref}, the iterative search technique finds the path for which*

$$I = \frac{D_{ref} - d}{g}$$

is a maximum, *where g is the cost and d is the delay of the path.*

Complexity

Observe that if D is the delay of the path with minimum cost, then the number of invocations of Dijkstra's algorithm is bounded by $O(\log D)$.

The running time of the **DRCR** algorithm is given by (number of invocations of the shortest path algorithm) $\times$ (time to find the shortest path) *i.e.* $O(\log D \times E \log |V|)$.

6 Properties

In this section, we explain some interesting properties of the solutions generated by the DRCR algorithm.

Lemma 2 *For any value of D_{ref}, the optimal ratio solution is not dominated by any other solution.*

Proof: Let (g, d) be the path found by the DRCR algorithm. Suppose there exists a path (g', d') which dominates (g, d) – i.e., $g' < g$ and $d' < d$. Since (g, d) is found by the DRCR algorithm and has the maximum delay reduction to cost ratio,

$$\frac{D_{ref} - d}{g} > \frac{D_{ref} - d'}{g'}.$$

Therefore,

$$\frac{D_{ref}}{g} - \frac{d}{g} > \frac{D_{ref}}{g'} - \frac{d'}{g'}$$

$$\frac{g'}{g}(D_{ref} - d) > D_{ref} - d'$$

Since we assumed that $d' < d$,

$$D_{ref} - d' > D_{ref} - d$$

Hence,

$$\frac{g'}{g}(D_{ref} - d) > D_{ref} - d$$

This is a *contradiction* as $\frac{g'}{g} < 1$ since $g' < g$ $\square$.

The following lemma can be shown by algebraic manipulation (see [7]).

Lemma 3 *As the value of D_{ref} decreases, the cost of the corresponding maximum ratio path increases.*

By Lemma 2, we know that the solutions generated by the DRCR algorithm are non-dominated and hence lie on a cost vs. delay tradeoff curve. Consequent to Lemma 3, we see that decreasing D_{ref} moves us right on the cost vs. delay curve; increasing D_{ref} moves us left.

Thus, a natural goal is to characterize the solutions to the DRCR problem and the set of *all* non-dominated paths (i.e., those generated by the labeling algorithm). This relationship is studied in the next subsection.

Lower Convex Hull

Definition : Let $S = \{(g_0, d_0), (g_1, d_1), ..., (g_k, d_k)\}$ be the set of non-dominated s to t paths. We define the *lower convex hull* (LCH) of S as follows.

- The minimum delay solution is on the LCH.
- The minimum cost solution is on the LCH.
- Any point (g_i, d_i) is on the LCH iff $\forall\, (g_l, d_l) \in S$ such that $g_l < g_i$ and $\forall\ (g_h, d_h) \in S$ such that $g_i < g_h$, (g_i, d_i) lie **below** the line segment joining (g_l, d_l) and (g_h, d_h).

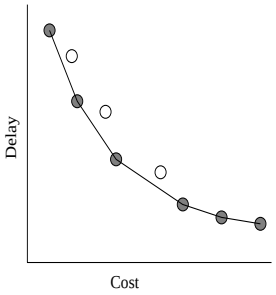

Figure 7: *Illustration of the Lower Convex Hull property. All circles represent the set of all non-dominated paths; the shaded circles represent points on the LCH.*

An example of a set of non-dominated points appears in Figure 7.

Since Formulation 1 is NP-hard, it is clear that our polynomial time algorithm must sacrifice something. What we sacrifice is summarized in the following theorem.

Theorem 1 *There exists a D_{ref} for which a (g,d) path is optimal* **iff** *(g,d) lies on the lower convex hull of the trade-off curve.*

Proof : (i) $\exists D_{ref} \Rightarrow (g,d)$ is on LCH.
Let D be a value of D_{ref} for which this is true and let (g,d) be the corresponding path. This yields

$$\frac{D-d}{g} > \frac{D-d'}{g'} \tag{5}$$

for all other non-dominated paths (g',d'). If (g,d) is the min-cost or the min-delay solution, then the proof is trivial (these paths are always on the lower convex hull).
Hence we only need to prove that for any pair of solutions (g_1,d_1) and (g_2,d_2) such that $g_1 < g < g_2$ (as in Figure 8),

$$\frac{d-d_1}{g-g_1} < \frac{d_2-d_1}{g_2-g_1} \tag{6}$$

i.e., the slope of the line joining (g,d) and (g_1,d_1) should be less that the slope of the line joining (g_1,d_1) and (g_2,d_2).
Rearranging inequality (6),

$$d < d_1 + (g-g_1)\frac{d_2-d_1}{g_2-g_1} \quad \text{and hence} \tag{7}$$

$$d < \frac{g_2d_1 + gd_2 - gd_1 - g_1d_2}{g_2-g_1} \tag{8}$$

Hence our goal is to prove inequality (8) holds.
Since D is the value of D_{ref} for which the path (g,d) has a maximum ratio,

$$\frac{D-d}{g} > \frac{D-d_1}{g_1} \quad \text{and} \tag{9}$$

$$\frac{D-d}{g} > \frac{D-d_2}{g_2}. \tag{10}$$

Rearranging inequality (9), we have

$$d < D - \frac{g}{g_1}D + \frac{g}{g_1}d_1. \tag{11}$$

Multiplying (11) by g_1,

$$g_1d - gd_1 < D(g_1-g).$$

Since the term (g_1-g) is negative, we have

$$\frac{g_1d - gd_1}{g_1-g} > D. \tag{12}$$

Rearranging inequality (10) and multiplying g_2 yields

$$\frac{g_2d - gd_2}{g_2-g} < D. \tag{13}$$

Combining (12) and (13) , we have

$$\frac{g_1d - gd_1}{g_1-g} > \frac{g_2d - gd_2}{g_2-g}.$$

Multiplying both sides by $(g_1-g)(g_2-g)$, which is negative,

$$(g_2-g)(g_1d - gd_1) < (g_1-g)(g_2d - gd_2).$$

By algebraic manipulation, we see that this reduces to

$$(g_2-g_1)d < g_2d_1 + gd_2 - gd_1 - g_1d_2$$

which is the same as inequality (8).

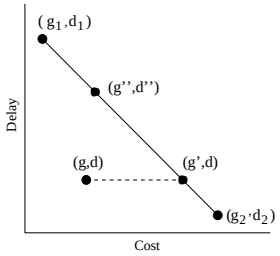

Figure 8: *Figure illustrating the existence of a D_{ref} for every (g,d) on the LCH.*

(ii) To prove the other way –*i.e.*, (g,d) is on the LCH $\Rightarrow$ $\exists D_{ref}$ which has the maximum ratio at (g,d), set

$$D_{ref} = \frac{g_2d_1 - g_1d_2}{g_2-g_1}$$

which makes the ratio

$$r = \frac{D_{ref}-d_1}{g_1} = \frac{D_{ref}-d_2}{g_2}.$$

As in Figure 8, for any point (g'',d'') on the line joining (g_1,d_1) and (g_2,d_2),

$$\frac{D_{ref}-d''}{g''} = r.$$

Consider any point (g',d) which is on the line and has the same d value with (g,d). This yields

$$\frac{D_{ref}-d}{g'} = r \quad \text{and thus}$$

$$\frac{D_{ref}-d}{g} > r \quad \text{since } g < g'$$

which completes the proof. □.

This theorem can be seen as a generalization of Lemma 2. We argue that sacrificing the solutions not on the LCH

is typically not a serious drawback in practice since the solutions which make the most cost-effective use of the available resources are precisely those which lie on the LCH.[2] Further, in practice optimization problems such as buffer insertion tend to have largely convex tradeoffs, so the algorithm fits such applications nicely.

Finally, we note that a variant of the algorithm allows us to explicitly explore the LCH via a similar binary search scheme. The main idea is to note that D_{ref} is really artificial and that for any I, the resulting shortest path is optimal for *some* D_{ref}. In this way we can explore the tradeoff curve by modifying I (increasing it to move left, decreasing to move right) as inferred from Lemma 3. This eliminates the need for another level of binary search and the algorithm retains the same complexity.

7 Experiments

We implemented the pseudo-polynomial algorithm and our DRCR algorithm in C and tested on different test cases on a $200MHz$ Sun Ultra-Sparc 1. The main objective of our experiments was to evaluate the computational feasibility of the DRCR algorithm and compare it with the pseudo-polynomial algorithm.

Test graphs were generated by randomly placing some macro blocks inside a rectangular routing region. Candidate buffer locations are then chosen randomly from the area that is not covered by the blocks and two buffers are considered as candidate neighbors (i.e., there is an edge between the corresponding nodes) only if they lie within a threshold distance (e.g., $2000\mu m$). The routing regions vary from $1cm \times 1cm$ to $2cm \times 2cm$. We use the following values for the technology parameters : unit length capacitance $c_0 = 0.15fF/\mu m$; unit length resistance $r_0 = 0.12\Omega/\mu m$; driver resistance $R_s = 270\Omega$; loading capacitance $C_t = 50fF$. We use a single buffer with the following parameters and build the buffer library by scaling these parameters: $r_b = 814\Omega$; $c_b = 28fF$; $d_b = 125ps$. We use the Elmore [6] model to compute the delay of the interconnect in our test cases.

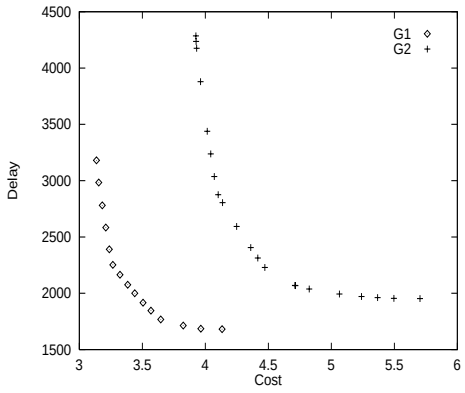

Figure 9: *The figure shows the lower convex hull of the cost vs. delay trade off curve for two different graphs* G_1 *and* G_2.

Figure 9 shows the plot of the solutions which can be found the DRCR algorithm for two different test cases. The graph shows the minimum delay solution, the delay of the minimum cost solution, and the solutions that lie on the LCH of the tradeoff curve. *G1* has 600 candidate buffer locations and *G2* has 2100 candidate buffer positions. The algorithm takes $0.73s$ for *G1* and $1.69s$ for *G2* respectively to identify a solution on the LCH. For reference, the labeling algorithm which uncovers all non-dominated paths took $9.02s$ and $49.6s$ on *G1* and *G2* respectively (naturally, this gap will become even more dramatic as more buffer sizing options are added, etc.)

The fast path algorithm in [1] identifies only the minimum delay path while the labeling algorithm generates all non-dominated solutions. The fast path algorithm finds the minimum delay solution quickly, but this may not be the best solution in practice due to excessive cost. On the other hand, the labeling algorithm uncovers the whole cost vs. delay tradeoff curve, but is computationally expensive. We suggest that our approach gives a better practical solution in terms of running time and the feasibility of solutions generated.

8 Conclusion

The paper studies the problem of buffer insertion in the context of a given floorplan. The **DRCR** problem in the context of buffer insertion was introduced and a fast polynomial algorithm was proposed to solve the problem. Some interesting properties of the formulation were discussed and it was argued that the solutions that lie on the LCH of the tradeoff curve are the best solutions in practice. Thus we believe the proposed algorithm will become a valuable tool in the early stages of design where buffers must be allocated and topological buffer-to-buffer routes determined.

References

[1] Hai Zhou,D.F.Wong,I-Min Liu and Adnan Aziz, *Simultaneous Routing and Buffer Insertion with Restrictions on Buffer Locations*, In *Proc. of Design Automation Conference*, pp.96-99, June 1999.

[2] Chris C.N.Chu and D.F.Wong, *A new approach to buffer insertion and wire sizing*, In *Proc. IEEE Intl. Conf. on Computer-Aided Design*, pp.614-621,1997.

[3] M.R.Garey and D.S.Johnson, *Computers and Intractability*, (Problem ND30) W.H.Freeman and Co.,1979.

[4] H.C. Joksch, *The Shortest Route Problem With Constraints*, J. Math Anal. Appl. 14 191-197.

[5] Eugene Lawler, *Combinatorial Optimization, Network and Matroids*, Holt,Rinehart,Winston, pp.94-97, 1976.

[6] W.C.Elmore, *The transient response of damped linear networks with particular regard to wide-band amplifiers*, *Journal of Applied Physics*, 19(1): 55-63,January 1948.

[7] Ashok Jagannathan, Sung-Woo Hur and John Lillis, *A Fast Algorithm For Context Aware Buffer Insertion*, Technical Report UIC-EECS-00-5, 2000.

[2] As an aside, it can be shown that the path which minimizes the gd product lies on the LCH.