A TWO-LAYER BUS ROUTING ALGORITHM FOR HIGH-SPEED BOARDS

Muhammet Mustafa Ozdal

Dept. of Computer Science
Univ. of Illinois at Urbana-Champaign
Urbana, IL 61801
ozdal@uiuc.edu

Martin D. F. Wong

Dept. of Electrical and Computer Engineering
Univ. of Illinois at Urbana-Champaign
Urbana, IL 61801
mdfwong@uiuc.edu

ABSTRACT

The increasing clock frequencies in high-end industrial circuits bring new routing challenges that can not be handled by traditional algorithms. An important design automation problem for high-speed boards today is routing nets within tight minimum and maximum length bounds. In this paper, we propose an algorithm for routing bus structures between components on two layers such that all length constraints are satisfied. This algorithm handles length extension simultaneously during the actual routing process so that maximum resource utilization is achieved during length extension. Our approach here is to process one track at a time, and choose the best subset of nets to be routed on each track. The algorithm we propose for single-track routing is guaranteed to find the optimal subset of nets together with the optimal solution with length extension on one track. The experimental comparison with a recently proposed technique shows the effectiveness of this algorithm both in terms of solution quality and run-time.

1. INTRODUCTION

During the past several years, we have seen dramatic increases in the clock frequencies of industrial circuits. As the circuits become smaller and faster, the routing problems become more and more difficult, and the designers face new challenges, most of which can not be handled by the traditional routing algorithms. Many high-end board designs in the industry today need to be routed manually, since the existing routing tools are usually ineffective for high-end circuits.

Routing nets within minimum and maximum length bounds is an important requirement for high-speed VLSI layouts. There have been several algorithms proposed for the objective of minimizing path lengths, or satisfying prespecified maximum length constraints, especially in the context of timing-driven routing [14, 8, 3, 18, 6, 16]. However, the problem of routing nets with lower bound constraints has not been studied extensively in the literature. The main reason is that these bounds were loose most of the time, and non-sophisticated ad-hoc strategies (such as greedy length extension in post-processing) were sufficient for most applications. However as circuits start to use clock frequencies in the order of gigahertz in the current technology, the timing constraints become extremely tight, and more aggressive methods for achieving length bounds are needed in the industrial applications.

We have been working on a project to develop a routing system for high-end IBM circuit boards, where a typical board contains several bus structures between MCM, memory and I/O modules, in addition to individual nets. Data is clocked into registers or other circuits in a typical bus structure; so all signals traveling over different wires of the bus must arrive at their destinations *approximately* at the same time. To achieve this, all the wires constituting this bus need to have *approximately* the same length. The precision with which matching must be done is directly related to the clock frequency [19]. As the clock frequency increases, the skew requirements on the propagation delays become more strict, and so a higher degree of length matching is required. Note here that this problem is completely different from the *minimum-skew clock tree routing* problem, which has been studied extensively in the literature [12, 13, 5, 22]. The difference here is that a bus consists of a large number of different two-terminal nets, as opposed to a single net with multiple terminals. The objective here is to find independent routing solutions for each net while satisfying all length constraints. Furthermore, there may be more than one bus structures interleaved with each other, or there may be individual nets not belonging to any bus. So, we will consider the general case where each net has individual min-max length constraints assigned to it.

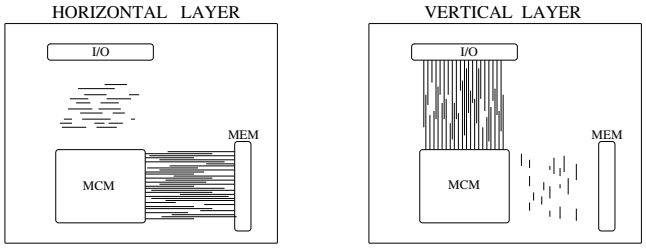

Figure 1: A typical two-layer routing solution. There are two separate bus structures here: (1) between MCM and I/O (2) between MCM and memory. No length extension (to satisfy min-length constraints) has been performed yet.

In this paper, we focus on a two-layer bus routing problem, where each layer has a primary routing direction of either horizontal or vertical. A sample routing solution is illustrated in Figure 1, where there are two bus structures: (1) a vertical bus between MCM and I/O module, (2) a horizontal bus between MCM and memory module. Observe in the horizontal layer that the congestion in the area corresponding to the vertical problem (i.e. the area between MCM and I/O) is considerably lower than the congestion in the area corresponding to the horizontal problem (i.e. the area

This work was partially supported by the National Science Foundation under grants CCR-0244236 and CCR-0306244.

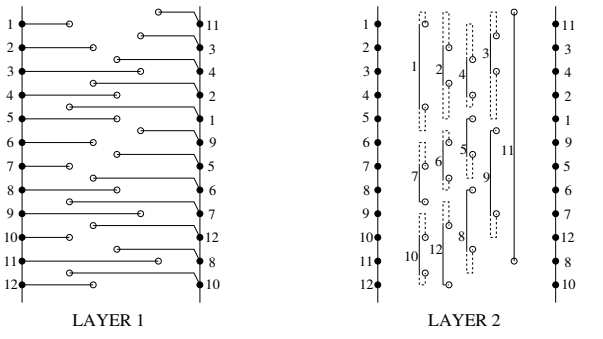

Figure 2: A sample routing solution on two layers, where each net has individual min-max length constraints. The terminals for 12 nets are aligned on the left and right side of the channel. Two vias (represented as empty circles) are used to route each net. The dashed lines on layer 2 indicate the length extension performed to satisfy min-length constraints.

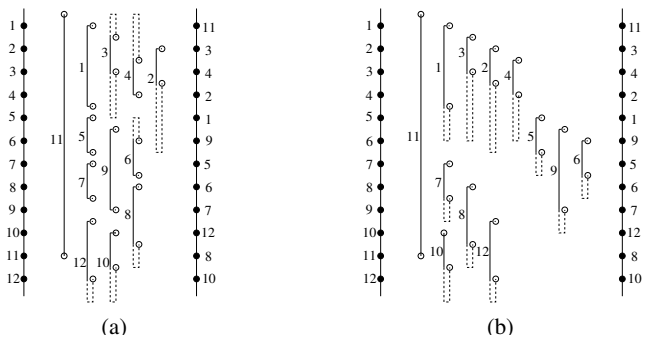

Figure 3: Alternative routing solutions corresponding to Figure 2, if (a) length extension is performed in post-processing, and (b) length extension is performed in preprocessing. In part (a), min-length constraints of nets 1, 5, 7 and 9 are violated. In part (b), the number of vertical tracks necessary increases to 8 (from 5). For clarity, only the results on the secondary layer are illustrated.

between MCM and MEM), and vice versa. The main reason is that the horizontal distance between terminals of a net in a horizontal problem corresponds to the distance between two different components, and it is typically much larger than the respective vertical distance.

Routability in a highly congested area is expected to be limited; so it will be more effective to perform length extension (to satisfy min-length constraints) within the less congested areas. For example, the vertical layer of a horizontal problem is expected to be significantly less congested; so it makes more sense to perform length extension on this layer. In this paper, we propose an algorithm that incorporates the objective of length extension into the actual routing algorithm. For a horizontal problem, our algorithm simultaneously extends the lengths of the nets and assigns them to vertical tracks.

Recently, a general Lagrangian relaxation framework has been proposed for satisfying min-max length constraints of printed circuit boards [17]. Although the target class of problems of that framework is more general, our algorithm has distinct advantages for the bus routing problem described above. First of all, we route multiple nets simultaneously on one track in an optimal way, instead of using a net-by-net approach, which has no theoretical guarantees. Furthermore, the routing solutions are more uniform in our algorithm, i.e. all nets use two vias, and the number of bends (due to length extension) is at most 4 for each net. Also, we consider a certain type of length extension methodology here, which is especially effective for this problem.

The rest of the paper is organized as follows. In Section 2, we describe the target problem in more detail, and discuss why simple ad-hoc methodologies are not sufficient for this problem. Then, we propose an algorithm in Section 3 based on some assumptions about input circuits. In Section 4, we relax these assumptions, and discuss how to generalize this algorithm. Finally, we perform experiments in Section 5 to show the effectiveness of our algorithm compared to the Lagrangian relaxation framework [17].

2. PROBLEM FORMULATION AND MOTIVATION

For a given set of nets $\mathcal{N}$, and min-max length constraints T_i^{min}, T_i^{max} for each net i, our purpose is to find a two-layer routing solution such that all length constraints are satisfied, and the routing resources are utilized most effectively. We assume that routing within dense components (escape routing) has already been accomplished by the earlier stages of the routing system; hence all terminals are now aligned on the opposite sides of the channel. At first glance, this problem may seem similar to the traditional *channel routing* problem [11, 23, 20, 4], which has been studied extensively in the literature. However, the existence of min-max length constraints due to the high-speed design rules makes this problem significantly different from the traditional problem.

For simplicity of presentation, we will first focus on a restricted problem instance, where (1) each pair of adjacent terminals is separated by at least one grid cell on each side, and (2) there are no obstacles within the routing area. The algorithms we propose in Section 3 will be based on these assumptions; however we will extend our algorithms in Section 4 for the general case.

All algorithms in this paper will be presented for a horizontal problem (i.e. terminals are aligned on the left and right sides of the channel); however it is trivial to modify them for a vertical problem. Let us denote the horizontal routing layer as the *primary layer*, and the vertical routing layer as the *secondary layer*. As mentioned before, since routing resources are very scarce on the primary layer, length extension will be performed on the secondary layer to satisfy all min-length constraints.

Figure 2 illustrates a sample routing solution for 12 nets, where the dashed lines indicate length extension performed on each net. Observe that layer 1 and 2 are primarily for routing horizontal and vertical segments, respectively. However small deviations from the primary directions are allowed on each layer. For instance, there are small diagonal segments (for alignment) on layer 1, and small horizontal segments (for length extension) on layer 2. Furthermore, the second layer is defined to be consisted of *vertical tracks*, where each track has width equal to the sum of via diameter and wire width (plus clearance between them). For example, 5 vertical tracks are used on layer 2 of this figure. Note that via diameters are typically much larger than wire widths; so the increase in track widths due to length extension is normally negligible.

It is important here to note that length extension needs to be

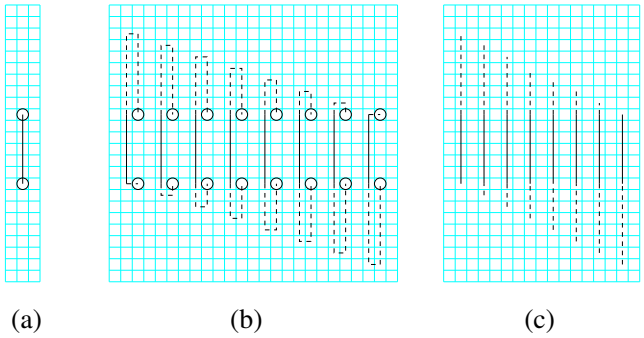

Figure 4: (a) A vertical segment which needs to be extended by 16 units. (b) Eight possible configurations corresponding to the extended segment. (c) Each configuration is represented as a single line, where dashed lines represent the length extension.

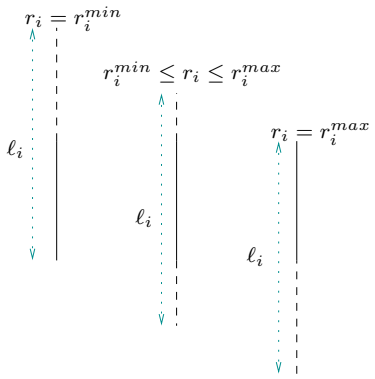

Figure 5: Three different cases for the vertical segment of net i are illustrated. Here, length ℓ_i is fixed, since it is determined by the min-length constraint. However, the top row r_i can vary between r_i^{min} and r_i^{max}. The solid and dashed lines here represent the original and extended segments, respectively.

performed simultaneously while determining the positions of vertical segments on the secondary layer. In the example of Figure 2, the number of vertical tracks used is kept minimum (i.e. 5 vertical tracks used on the second layer), and the routing resources are utilized most effectively. However, this utilization will be significantly reduced if length extension is performed as a separate step in the routing process. For instance, one can use a traditional channel routing algorithm (such as left-edge algorithm [11]) first to assign routing segments to the vertical tracks, and then extend the lengths of vertical segments in post-processing. Figure 3(a) shows the corresponding solution of the left-edge algorithm. Observe that, segments have been assigned to vertical tracks without considering the min-length constraints. So, it is not guaranteed that there is enough space around each net such that its length will be successfully extended in post-processing. For instance, consider net 5 in this figure, which is assigned to the second vertical track between the segments of nets 1 and 7. Obviously, its length can not be extended in post-processing (due to lack of space), and its min-length constraint will be violated. Specifically, there are 4 nets in this example of which length constraints will not be satisfied even after post-processing: nets 1, 5, 7, and 9. This example clearly demonstrates that min-length constraints need to be considered during the actual routing process, not just as a post-processing step.

Another approach here can be to extend the lengths of vertical segments using a predefined pattern in preprocessing, and then to apply a traditional channel routing algorithm to assign them to vertical channels. Figure 3(b) shows such an example, where segments have been extended (from bottom) first; then left edge algorithm has been applied on the extended segments. The disadvantage of this approach is that the routing algorithm has no control over the length extension process; so the resulting solution can not utilize the routing resources most efficiently. In this example, 8 vertical tracks are used to obtain a feasible solution, while Figure 2 shows that 5 tracks would be sufficient to satisfy all length constraints. This example shows that performing length extension as a preprocessing step is also not an effective strategy.

The algorithms we propose in this paper handle length extension and track assignment simultaneously, so that a feasible routing solution is obtained while using minimum number of vertical tracks. For instance, observe in Figure 2 that net 1 is extended both from top and from bottom, and such an extension allows 3 nets to fit on one track. The next section describes our models and algorithms in more detail.

3. ALGORITHM DESCRIPTION

3.1. Routing Model

Routing on the horizontal layer is straightforward, because of the assumptions that there are no obstacles in the routing area, and each adjacent pair is separated by at least one grid cell (see Section 4 for the general case without these assumptions). As illustrated in the example of Figure 2, a horizontal connection[1] is possible from each terminal on one side of the channel to the other side, without any conflicts with others. So, the main problem here is to determine the positions of vertical segments on the secondary layer. Once the positions of these vertical segments are fixed, the horizontal segments on the first layer can be connected to them using vias, as illustrated in this example.

Figure 4 shows an example illustrating the way length extension is performed on vertical segments. Here, assume that we need to extend the length of the segment in part (a) by 16 units (in terms of grid cells) to satisfy its min-length constraint. Figure 4(b) shows eight possible configurations, each of which is the extended version of the original segment. As mentioned before, one via and one wire is defined to fit on a *vertical track* together; so each extended segment in this figure is defined to be on a single track. Figure 4(c) gives a simpler representation, where a solid line represents the original segment, and a dashed line represents the extended length.

Min length constraint for a net directly determines the minimum length requirement for its vertical segment. Let x_i denote the amount of length extension required to satisfy min length constraint of net i. The value of x_i is simply equal to the Manhattan distance between net i's terminal points subtracted from its min length constraint. Here, the vertical segment of net i must be extended by at least $x_i/2 - 1$ from top or from bottom, as in Figure 4(c). In the example of Figure 4, x_i is given as 16, and the

[1] A small diagonal segment might be necessary to align the horizontal segments on opposite sides, as shown in Figure 2.

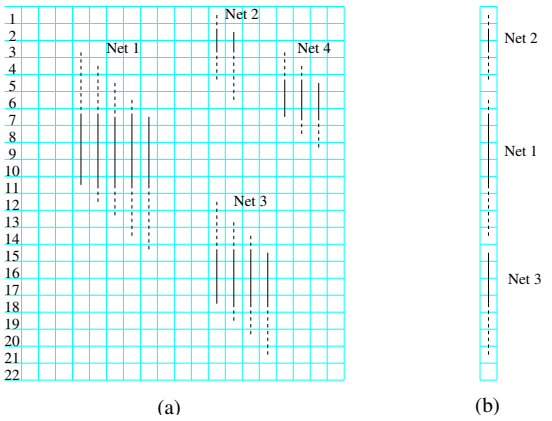

Figure 6: (a) A sample single-track assignment problem with 4 nets. Multiple routing configurations are illustrated for each net. (b) The optimal solution, which utilizes 21 out of 22 grid cells of one track. The dashed lines indicate the length extension performed.

vertical segment of net i needs to be extended by at least 7 units[2]. These concepts are formalized by the following definitions:

Definition 3.1. *The routing solution for net i is defined based on the position of its vertical segment, and it is denoted as $R_i = (t_i, r_i, \ell_i)$, where t_i is the track number, r_i is the top row, and ℓ_i is the length of the vertical segment of net i. Here, ℓ_i is determined directly from the min length constraint of net i, as discussed before. Furthermore, r_i must be chosen such that $r_i^{min} \leq r_i \leq r_i^{max}$, where $r_i = r_i^{min}$, and $r_i = r_i^{max}$ correspond to the extreme cases where no length extension is performed from the bottom, and from the top, respectively. The main idea is illustrated in Figure 5.*

Definition 3.2. *The routing problem for a given set of nets is defined as finding a solution $R_i = (t_i, r_i, \ell_i)$ for each net i such that: (1) no two vertical segments on the same track overlap with each other, and (2) the number of vertical tracks used is minimized.*

Note that the min length constraints are captured by the target length ℓ_i defined for each net i, while the max length constraints do not need to be considered explicitly. The reason is that each net is routed using the minimum possible length in this algorithm.

3.2. Algorithm Proposed

The problem defined by Definitions 3.1 and 3.2 is actually a special case of *task scheduling problem with release times and deadlines on a multi-computer*. Here, each vertical track can be considered as a computer; each vertical routing segment can be considered as a task with length ℓ_i, release time r_i^{min}, and deadline r_i^{max}. Note that this problem is known to be an NP-complete problem in the strong sense, even in the case where there is a single computer [2]. However, the special property of our problem will allow us to give a polynomial-time exact algorithm for the single-track case.

Here, our approach will be to process one track at a time, and *pack* as many routing segments as possible on each track. Note that the best routing configuration for each net should also be determined simultaneously during this process. The following definition gives a formal description of this objective.

[2]As shown in Figure 4(b), the extended part actually consists of two adjacent vertical wire segments, and one unit of horizontal wire segment. However for simplicity, we represent it as a single wire as in part (c).

SINGLE-TRACK-ASSIGNMENT ($\mathcal{N}$: set of nets, t: current track)
create a graph $\mathcal{G}$ as follows:
  for each row j of track t
    create a vertex $v[j]$
    add a unit-weight edge from $v[j-1]$ to $v[j]$
  for each net i in $\mathcal{N}$
    for $r_i = r_i^{min}$ to r_i^{max} do
      create a zero-weight edge from $v[r_i]$ to $v[r_i+\ell_i+1]$

Compute the shortest path P from v[1] to v[last] in $\mathcal{G}$
for each edge $e \in P$
  if e is a zero-weight edge from $v[k]$ to $v[m]$
    select the vertical segment that spans rows from k to m-1

Figure 7: Algorithm for selecting the maximal subset of non-overlapping vertical segments

Definition 3.3. *The problem of single-track assignment is defined as follows: Given a set of nets $\mathcal{N}$, and a set of vertical segments for each net i in $\mathcal{N}$, the objective is to select a subset of these vertical segments, such that (1) at most one vertical segment is selected corresponding to each net i, (2) the selected segments do not overlap with each other, and (3) maximum resource utilization is achieved on one track (i.e. the number of grid cells unused is kept minimum).*

As an example, consider Figure 6(a), where there are 4 different nets, and each net has multiple routing configurations. The corresponding optimal solution for single-track assignment is shown in Figure 6(b). Observe that 21 out of 22 grid cells have been utilized on this track.

As mentioned above, this problem is a special case of the task scheduling problem on a single computer, which is an NP-complete problem in the strong sense. However, we propose an algorithm in Figure 7, which is guaranteed to find the optimal solution in polynomial time (due to the special property given in Lemma 3.2). Here, the main idea is to represent each row of the track as a vertex, and to model each vertical segment as a zero-weight edge between the respective rows. Furthermore, there is a unit-weight edge from each $v[j]$ to $v[j+1]$, which corresponds to the case where the grid cell on row j is unused. Then, the shortest path from the first row to the last row is computed to find the optimal assignment with the maximum resource utilization. Intuitively, the weight of an edge from $v[k]$ to $v[m]$ indicates the number of grid cells that will be wasted between rows k and $m-1$ if this edge is selected. So, the shortest path from the top row to the bottom row will give us the assignment with the minimum waste. Figure 8 illustrates the graph model corresponding to the problem given in Figure 6(a). The highlighted path in this graph is the shortest path, and it corresponds to the optimal solution in Figure 6(b). For example, the edge from $v[1]$ to $v[6]$ on the shortest path corresponds to the vertical segment of net 2 from row 1 to row 5. The formal analysis of this algorithm is given as follows:

Lemma 3.1. *Consider any pair of edges $e_i, e_j \in \mathcal{G}$. If there exists a path P such that $e_i, e_j \in P$, then the vertical segments*

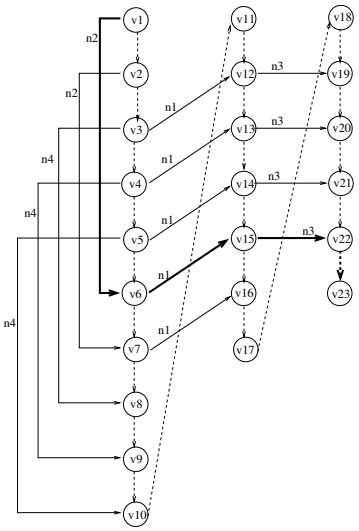

Figure 8: The graph model corresponding to the problem of Figure 6(a). The edges corresponding to net segments (solid arrows) have zero weights, while the others (dashed arrows) have unit weights. The shortest path (with total weight 1) is highlighted, and it corresponds to the optimal solution in Figure 6(b).

corresponding to e_i and e_j are guaranteed not to overlap with each other.

PROOF. The direction of edges in $\mathcal{G}$ are always towards larger vertex indices. Furthermore, for a vertical segment that spans rows k to m, the corresponding edge will be from $v[k]$ to $v[m+1]$. Hence, any edge selected after this edge will correspond to a segment starting from row $m+1$. So, any pair of edges in a path can not correspond to overlapping net segments. □

Lemma 3.2. *Consider the set of edges E_n corresponding to net n. There exists no path P in $\mathcal{G}$ such that $e_i, e_j \in P$ and $e_i, e_j \in E_n$. In other words, a path can not contain two edges corresponding to different vertical segments of the same net.*

PROOF. There are different vertical segments corresponding to net n, because there are different ways of extending the length of n. However, as can be seen in Figure 5, the original segment (represented with solid lines) is always fixed; hence all vertical segments corresponding to the same net will overlap with each other. From Lemma 3.1, a path can not contain edges corresponding to overlapping net segments. □

Theorem 3.3. *The shortest path in $\mathcal{G}$ between the first and last vertices corresponds to the optimal solution of the single-track assignment problem defined in Definition 3.3.*

PROOF. From Lemma 3.1 and 3.2, the set of edges on any path corresponds to a valid assignment on one track. Furthermore, there is a path in $\mathcal{G}$ corresponding to any valid assignment on one track. Since the unit weighted edges in $\mathcal{G}$ correspond to the unused rows of the track, the total weight of path P will be equal to the number of rows wasted. So, the shortest path from the top row to the bottom row will correspond to the optimal assignment with maximum utilization. □

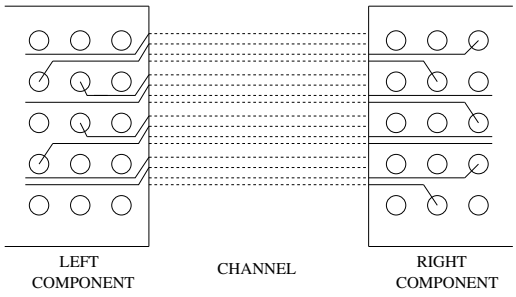

Figure 9: In a typical board routing problem, nets escape from dense components (solid line segments), and the input to the bus routing problem is defined as a set of terminals aligned on the opposite sides of a channel. Since the diameters of the pins within components are much larger than wire widths, these terminals are typically well-separated. So, it is possible to align all horizontal segments (dashed lines) such that there are no overlaps between them.

Theorem 3.4. *Let H denote the number of rows in the channel, and x_i denote the amount of length extension required to satisfy min-length constraint of net i. The time complexity of the algorithm given in Figure 7 is $O(H + \sum_{i \in N} x_i)$.*

PROOF. There are $O(x_i)$ different vertical segments defined for each net. So, the number of edges in $\mathcal{G}$ is $O(H + \sum_{i \in N} x_i)$, while the number of vertices is $O(H)$. Furthermore, $\mathcal{G}$ is a directed acyclic graph, and the shortest path can be computed in linear time [7]. □

4. GENERALIZATION OF THE ALGORITHM

In the previous section, we have assumed that each pair of adjacent terminals is separated by at least one grid cell. However, this is not absolutely necessary, as long as it is possible to extend the horizontal segments from one side of the channel to the other side without any conflicts, as illustrated in Figure 9. If this is the case, then there will be no restrictions on the positions of the vertical segments, and the same algorithm in Section 3 can be used without a change. Note that this assumption is reasonable for a typical industrial circuit, since the pin diameters within chip components are much larger than the wire widths; so there will be enough routing space to align horizontal segments from both sides without any overlaps (as in Figure 9).

Actually, this corresponds to the unrestricted case of the original channel routing problem [11], where there are no vertical constraints, i.e. net segments can be assigned to tracks without any ordering constraints. On the other hand, if overlaps are possible on the horizontal layer, then we need to define *pin constraints* to avoid overlaps. For instance, assume that the horizontal segment of net i originating from a left terminal overlaps with the horizontal segment of net j originating from a right terminal. In that case, the vertical segment of net i must be assigned to a track which is to the left of the vertical segment of net j to avoid an overlap on the horizontal layer. Note that if there were no min-max length constraints, this would correspond to the problem of *channel routing with vertical constraints*, which has been studied in the literature. This problem has been shown to be NP-complete [15, 21]; however there have been several algorithms proposed that give sufficiently good results [20, 4, 23, 10]. If the assumption of well-

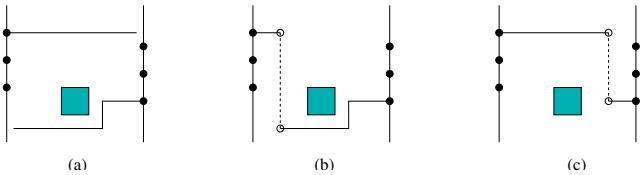

Figure 10: (a) The horizontal segments of a net are not entirely straight due to an obstacle. (b) The vertical segment is assigned on a track to the left of the obstacle. (c) The vertical segment is assigned on a track to the right of the obstacle. The solid and the dashed lines represent the routing segments on the horizontal and vertical layers, respectively. For clarity, only one net is displayed, and length extension on the vertical layer is not shown.

separated terminals (mentioned above) is not valid for a circuit, we can use similar ideas to extend our algorithm to the general case. In particular, we can define pin constraints indicating the ordering of the vertical segments, and then perform track assignment based on this ordering. Since our algorithm processes one track at a time, we can simply consider the set of nets that do not violate the ordering constraints for the track that is being processed.

We can also generalize our algorithm to the case where there are some obstacles in the routing region. If the obstacles are on the horizontal layer, then the horizontal segments will not be entirely straight, as shown in Figure 10(a). Furthermore, the y coordinates of the vertical segments will depend on the track on which it is assigned, as illustrated in parts (b) and (c) of the same figure. Since the algorithm we propose processes one track at a time, the appropriate vertical segments corresponding to each net can be determined for each track, and the algorithm given in Figure 7 can still be used to choose the best subset. On the other hand, we can handle the obstacles on the vertical layer by simply removing the edges corresponding to vertical segments that overlap with an obstacle on the current track.

5. EXPERIMENTAL RESULTS

We have performed experiments to compare our algorithm with a recently proposed Lagrangian relaxation (LR) based approach [17]. All algorithms in this section have been implemented in C++, and experiments were performed on an Intel Pentium 4 2.4GHz system with 1GB memory, and a Linux operating system.

A sample output of our algorithm is illustrated in Figure 11, for a routing problem with 128 nets. Here, each net has individual length constraints, and terminals are aligned on the top and bottom sides of the channel. Since this is a vertical problem, length extension is performed on the horizontal (second) layer. Observe that nets have been assigned to tracks and their lengths have been extended so that maximum resource utilization is achieved on each track.

The experiments we have performed on test problems are given in Table 1. Here, "*avg. spacing*" is measured in terms of the number of grid cells between terminal points of adjacent nets, and it indicates how dense the problem is. On the other hand, columns "*length avg*" and "*length stdev*" give statistical information about net target lengths. Each problem in this table contains between 100-300 nets, with individual length constraints for each net. The grid size for the smallest circuit in this table is 100×330, and the largest one is 290×776. The last 3 problems here have been extracted from an IBM design, corresponding to the bus routing problems between MCM, memory and STI modules. Here, layer assignment and routing inside chips have been performed by the previous phases of the routing system; so the input for the bus routing problem is a set of aligned terminals on opposite sides of a channel. While the first 7 circuits in this table have a single layer pair, the IBM circuits have multiple layer pairs.

Table 1: Comparison of our algorithm with the Lagrangian relaxation based approach

				OUR ALGORITHM		LR-BASED	
Input Prob.	Avg. spacing	Length avg	Length stdev	# nets failed	time (m:s)	# nets failed	time (m:s)
B1	3.40	150.1	39.9	0	0:01	5	38:21
B2	3.19	138.5	18.6	0	0:01	6	44:52
B3	3.46	151.6	40.1	0	0:01	21	57:06
B4	2.91	160.8	38.0	0	0:01	4	46:32
B5	3.16	183.9	30.0	0	0:01	19	52:50
B6	2.94	174.7	18.1	0	0:01	48	52:45
B7	2.12	157.4	20.7	0	0:01	45	73:50
IBM1	7.75	417.6	35.8	3	0:01	3	2:29
IBM2	6.41	382.0	46.0	1	0:01	1	1:50
IBM3	9.16	427.6	73.8	0	0:01	0	7:06

The results in this table indicate that our algorithm performs significantly better than the Lagrangian relaxation based approach on most circuits, in terms of both quality and run time. The solution quality for LR-based approach degrades especially when the average spacing between nets decrease, or the target lengths increase (hence more aggressive length extension required). The main reason for this is that LR-based approach uses a variant of Pathfinder [1, 9] algorithm in the low level, where routing conflicts are resolved through *negotiations*. As the problems get denser, these negotiations take more and more time, and they don't always successfully lead to a good result. On the other hand, our algorithm performs length extension in a fast and effective way.

6. CONCLUSIONS

We have proposed a routing algorithm with the objective of satisfying length constraints of high-speed printed circuit boards. The main idea is to perform length extension on the secondary layer (e.g. vertical layer for a horizontal problem), where routing congestion is typically much lower than the primary layer. We have proposed an optimal algorithm to select the best subset of nets to assign to a single track, while satisfying the length constraints. Our experiments show that compared to a general Lagrangian relaxation framework, this algorithm performs considerably better on its target class of problems.

7. REFERENCES

[1] V. Betz and J. Rose. Vpr: A new packing, placement and routing tool for FPGA research. In *Proc. of the 7th International Workshop on Field-Programmable Logic*, pages 213–222, 1997.

[2] J. Blazewicz, K. H. Ecker, and E. Pesch. *Scheduling Computer and Manufacturing Processes*. Springer, 2001.

[3] K. D. Boese, J. Cong, A. B. Kahng, K. S. Leung, and D. Zhou. On highspeed VLSI interconnects: Analysis and design. In *Proc. Asia–Pacific Conf. Circuits Syst.*, 1992.

[4] M. Burstein and R. Pelavin. Hierarchical channel router. In *Proc. of IEEE/ACM 20th Design Automation Conference*, pages 591–597, 1983.

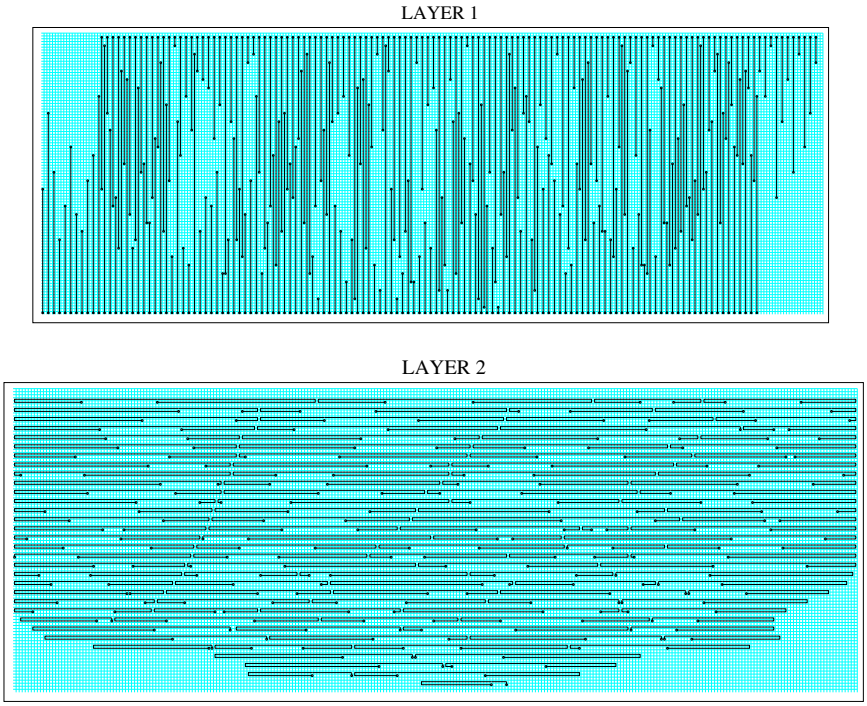

Figure 11: A sample two-layer solution for 128 nets. A vertical problem is illustrated here; hence length extension is performed on the horizontal (second) layer. The length constraints for all nets have been satisfied in this solution.

[5] J. Cong, A. B. Kahng, C.-K. Koh, and C.-W. A. Tsao. Bounded-skew clock and steiner routing. *ACM Trans. on Design Automation of Electronic Systems*, 3(3):341–388, 1998.

[6] J. Cong, A. B. Kahng, G. Robins, M. Sarrafzadeh, and C. K. Wong. Provably good performance-driven global routing. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 11(6):739–752, 1992.

[7] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, 1992.

[8] A. E. Dunlop, V. D. Agrawal, D. N. Deutsch, M. F. Jukl, P. Kozak, and M. Wiesel. Chip layout optimization using critical path weighting. In *Proc. of Design Automation Conference*, pages 133–136, 1984.

[9] C. Ebeling, L. McMurchie, S. A. Hauck, and S. Burns. Placement and routing tools for the triptych FPGA. *IEEE Trans. on VLSI*, pages 473–482, 1995.

[10] S. C. Fang, W. S. Feng, and S. L. Lee. A new efficient approach to multilayer channel routing problem. In *Proc. of the 29th ACM/IEEE Design Automation Conference*, pages 579–584, 1992.

[11] A. Hashimoto and J. Stevens. Wire routing by optimizing channel assignment within large apertures. In *IEEE Proc. of 8th Design Automation Workshop*, pages 214–224, 1971.

[12] D. J.-H. Huang, A. B. Kahng, and C.-W. A. Tsao. On the bounded-skew clock and steiner routing problems. In *Proc. 32nd ACM/IEEE Design Automation Conf.*, pages 508–513, 1995.

[13] A. B. Kahng and G. Robins. *On Optimal Interconnections in VLSI*. Kluwer Academic Publishers, 1995.

[14] E. Kuh, M. A. B. Jackson, and M. Marek-Sadowska. Timing-driven routing for building block layout. In *Proc. IEEE Intl. Symposium on Circuits and Systems*, pages 518–519, 1987.

[15] A. S. LaPaugh. *Algorithms for Integrated Circuit Layout: an Analytic Approach*. PhD thesis, Laboratory for Computer Science, MIT, Cambridge, MA, 1980.

[16] S. Lee and M. D. F. Wong. Timing-driven routing for FPGAs based on lagrangian relaxation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 506–510, 2003.

[17] M. M. Ozdal and M. Wong. Length matching routing for high-speed printed circuit boards. In *Proc. of IEEE Intl. Conf. on Computer-Aided Design*, Nov. 2003.

[18] S. Prastjutrakul and W. J. Kubitz. A timing-driven global router for custom chip design. In *Proc. of IEEE Intl. Conf. on Computer-Aided Design*, pages 48–51, 1990.

[19] L. W. Ritchey. Busses: What are they and how do they work? *Printed Circuit Design Magazine*, 2000.

[20] R. L. Rivest and C. M. Fiduccia. A greedy channel router. In *Proc. of IEEE/ACM 19th Design Automation Conference*, pages 418–424, 1982.

[21] Szymanski85. Dogleg channel routing is NP-complete. *IEEE Transactions on Computer-Aided Design*, CAD-4(1):31–41, 1985.

[22] C.-W. A. Tsao and C.-K. Koh. UST/DME: a clock tree router for general skew constraints. *ACM Transactions on Design Automation of Electronic Systems*, 7:359–379, 2002.

[23] T. Yoshimura and E. S. Kuh. Efficient algorithms for channel routing. *IEEE Transactions on Computer-Aided Design*, CAD-1(1):25–35, 1982.