Purdue University Purdue e-Pubs

ECE Technical Reports

Electrical and Computer Engineering

1-26-1995

An Efficient Lower Bound Algorithm For Channel Routing

Heng Yi Chao Purdue University School of Electrical Engineering

Mary P. Harper Purdue University School of Electrical Engineering

Follow this and additional works at: http://docs.lib.purdue.edu/ecetr

Chao, Heng Yi and Harper, Mary P., "An Efficient Lower Bound Algorithm For Channel Routing" (1995). *ECE Technical Reports*. Paper 110. http://docs.lib.purdue.edu/ecetr/110

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.

AN EFFICIENT LOWER BOUND Algorithm for Channel Routing

Heng-Yi Chao Mary P. Harper

TR-EE 95-3 January 1995



School of Electrical Engineering Purdue University West Lafayette, Indiana 47907-1285

An Efficient Lower Bound Algorithm For Channel Routing

Heng-Yi Chao and Mary P. Harper School of Electrical Engineering 1285 Electrical Engineering Building Purdue University West Lafayette, IN 47907-1285 Phone: (317) 494-4652, Fax: (317) 494-6440 Email: hengyi@ecn.purdue.edu, harper@ecn.purdue.edu

January 26, 1995

Abstract

Channel routing plays a central role in the physical design of VLSI chips. For two-layer dogleg-free channel routing, d_{max} and v_{max} are the two traditional lower bounds. In this paper, we present two efficient algorithms for computing a tighter lower bound for the channel routing problem. Our algorithms successfully compute a lower bound of 26 for Deutsch's Difficult Example (DDE). The experiment on some large-scale randomly generated channel routing problems shows that our lower bound algorithms are much tighter than the traditional lower bounds, and are much more efficient than Pals' algorithm [20] while obtaining similar (sometimes better) results.

Keywords: CAD on VLSI, physical design, channel routing, lower bound, DAG, transitive closure.

List of Figures

1	A CRP example. its vertical constraint graph (VCG) and horizontal constraint graph									
	(HCG). The small boxes represent the vias									
2	A CRP example (RKPC3) for demonstrating LB3									
3	An optimal routing of HYC1									

List of Tables

1	Conflicting results for Deutsch's difficult example.	6
2	Definitions of height h_i , density d_i , lower-bound lb_i , and their co-label counterparts.	8
3	Comparison of lower bounds for some benchmark CRPs	15
4	Comparison of computed lower bounds for our benchmark CRPs	17
5	Comparison of running times (seconds) for our benchmark CRPs	17
6	Random CRP examples - set I.	20
7	Random CRP examples - set II.	21

1 INTRODUCTION

Channel routing plays a central role in the physical design of VLSI chips. To meet the increasing demands of functionality, the number of transistors on a chip today has increased considerably. For example, a new MPEG2 decoder chip consists of 700,000 transistors on an area of 87.23 mm². Most layout systems begin by placing modules on a chip, and then wiring together terminals that should be electrically connected in different modules. An efficient approach for solving the wiring problem is to heuristically partition the chip into a set of rectangular *channels*, and then route each channel separately. This effectively divides a difficult problem into smaller subproblems that can be conquered more easily.

In this paper, we consider the two-layer *restricted* Manhattan model [17, 19, 26]. Although a three-layer process is available, the two-layer model is still attractive for the following reasons:

- The yield is higher for the two-layer process.
- The two-layer process is much less expensive than the three-layer process.
- If a product is time critical on the market, the two-layer model provides a faster way of bringing the product to the market.

A two-layer *channel* is a gridded rectangular area on a chip consisting of a metal layer running *horizontally* and a polysilicon layer running *vertically* (or vice versa). A wire in the horizontal layer is called a *track* and a wire in the vertical layer is called a *column*. There are fixed terminals on the top and bottom sides, and floating terminals on the left and right sides of the channel. Each set of terminals that need to be electrically connected is called a *net*. A net can connect terminals from the top and bottom of the channel and can exit the channel at the left and right sides. Connections of wires on different layers are made through *vias*. A channel routing instance is shown in Figure 1.

A two-layer Channel Routing Problem (CRP) is the problem of assigning a set V of nets, |V| = n, to a minimum number of tracks such that no nets overlap on any layer. We consider routings without *doglegs*, that is, the horizontal segment of a net cannot be *split*. This wiring style has the advantage that the number of vias is minimal [18, 19, 26].

For a channel routing instance, let S^* denote the minimum number of tracks required. If $lb \leq S^* \leq ub$, then lb is called a lower bound and ub is called an upper bound on S^* . Clearly, lb



Figure 1: A CRP example, its vertical constraint graph (VCG) and horizontal constraint graph (HCG). The small boxes represent the vias.

(ub) should be as large (small) as possible, with the goal of having $lb = S^* =$ ub. Since the channel routing problem is NP-complete [19, 25, 26], most previous work has focused on finding a heuristic solution (an upper bound). In this paper, on the other hand, our objective is to find a tighter (larger) lower bound on S*. This lower bound approach [2, 4] is significant because:

- A solution that equals the lower bound is optimal.
- A tight lower bound provides a good measurement of a heuristic's quality.
- A tight lower bound can be a powerful heuristic for node selection and pruning in branchand-bound methods [18, 26].

1.1 Horizontal Constraints and Vertical Constraints

A channel routing problem can be characterized by two types of constraints, the horizontal constraints and vertical constraints.

The constraint that two nets cannot overlap on the horizontal layer is called the horizontal constraint. Let l_i be the leftmost and r; be the rightmost column of net i. A net i is said to span the c-th column if $l_i \leq c \leq r$;. The set of columns $[l_i, r_i]$ is called the span of net i.

There is a horizontal constraint between net i and net j if and only if their spans overlap. The horizontal constraints are often represented by an undirected graph, the **horizontal constraint** graph (HCG) (see Figure 1), where vertices represent the nets and edges represent the horizontal constraints. In this paper, the horizontal constraints are also represented by a bit matrix hc such that hc(i, j) = 1 if and only if there is a horizontal constraint between i and j.

Let Z_i be the set of nets that span the i-th column, $d_{max} \equiv \max\{|Z_i|: i \text{ is a column}\}$ is called the density of the CRP. Clearly, d_{max} is a lower bound on S* because nets spanning the same column cannot be assigned to the same track.

The constraint that two nets cannot overlap on the vertical layer is called the vertical constraint. Note that if net i connects to the c-th column in the top row and net j connects to the c-th column in the bottom row, i # j, then net i must be assigned to a track higher than net j. In this case, we say that net i must precede net j and there is a vertical constraint from i to j. The vertical constraints define a partial ordering between nets. The vertical constraints are often represented by a directed graph, the **vertical constraint graph (VCG)** (see Figure 1), where vertices represent the nets and arcs represent the vertical constraints. In this paper, the vertical constraints are also represented by a bit matrix vc such that vc(i, j) = 1 if and only if there is a vertical constraint from *i* to *j*.

Note that vertical constraints are transitive, i.e., if $i \prec j$ and $j \prec k$, then $i \prec k$. Hence, if there is a path from *i* to j in the VCG, then *i* must be assigned to a track higher than j. In a directed graph G, we say that an arc (i, j) is a transitive arc if there exists k # i, j such that there is a path from *i* to k and there is a path from k to j in G; a direct arc if not.

Note that if there is a cycle in the VCG, a dogleg routing [8, 16] is necessary. Because we assume a dogleg-free routing, VCG must be a directed acyclic graph (DAG). The length of a path P is the number of vertices on the path. Nets on any path cannot be assigned to the same track. Let v_{max} be the length of the longest path in the VCG. Clearly, v_{max} is a lower bound on S*.

From previous discussion, we can see that the traditional lower bound $\max\{d_{max}, v_{max}\}$ is an obvious lower bound on S* for CRPs. Note that an initial vertical constraint implies a horizontal constraint, that is, if there is a vertical constraint from i to j, then there is a horizontal constraint between i and j (because they share at least one common column). Hence, usually $d_{max} \ge v_{max}$. The traditional lower bound may not constitute a tight lower bound on S*because of the interaction of constraints [2].

1.2 Our Approach and Contributions

Our approach is to integrate both the horizontal constraints and vertical constraints into a directed acyclic graph G by assigning labels to each vertex, and then traversing the graph to compute a tighter lower bound using the labels. Our work is distinct from previous work in several respects:

- Our algorithms successfully compute a lower bound of 26 for Deutsch's difficult example [8, 19, 28].
- The experimental results (see Section 5) show that our lower bounds are much tighter than the traditional lower bounds, which indicates that it is very important to consider the interaction of constraints for multiple-constraint problems.
- Our lower bound algorithm effectively combines the effects of the horizontal constraints and vertical constraints into a directed acyclic graph. This technique is useful for other problems with capacity and precedence constraints [2] (we have used a similar approach to compute a tight lower bound for the superscalar pipeline scheduling problem [4]).

• The time complexity of our lower bound algorithm is $O(n^3)$ as opposed to $O(n^6)$ in [20], while obtaining similar results.

The rest of the paper is organized as follows. In Section 2, we review related work, and motivate our approach with a simple example. A new lower bound algorithm LB2 is presented in Section 3, and a tighter lower bound algorithm LB3 is presented in Section 4. Experimental results are detailed in Section 5, and conclusions are drawn in Section 6.

2 RELATED WORK

The channel routing problem has been studied extensively [8, 12, 14, 15, 16, 17, 18, 20, 21, 22, 24, 26, 27, 28]. However, most previous work has focussed on finding a routing, and little work has been done on finding a tighter lower bound for CRPs. In recent years, several branch-and-bound type algorithms have been proposed to find optimal solutions [18, 26]. These algorithms require maintaining a search tree and finding all cliques for each node in the search tree. For the simplest case where the VCG has no arcs, an optimal routing can be obtained easily by using the left-edge algorithm [13, 16, 28]. In this case, however, an enormous number of nodes is expanded in the search tree.

For channel routing problems, researchers have used the examples in [8] as benchmarks to test their algorithms, especially the so-called Deutsch's difficult example [8, example B]. However, there are conflicting results in the literature for DDE routed without doglegs (see Table 1). For all examples but DDE in [8], $d_{max} \ge v_{max}$, which is not surprising since an initial vertical constraint implies a horizontal constraint.

K. K. Lee and H. W. Leong [17] improved the traditional lower bound by considering the impact of the horizontal constraints on the vertical constraints. Let S be a set of nets. If net $i \notin S$ has a horizontal constraint with every net in S, then *i* is said to *intersect* with S. For each path P in the VCG, let S_p be the set of nets that are not in P but intersect with P. A lower bound on S^* is $t_p := |P| + \max |clique(S_p)|$. For some of their randomly generated examples, they obtained an improvement of one or two over the traditional lower bounds. However, no improvement was obtained for DDE. Furthermore, the lower bound is computed by using a branch-and-bound method, which may not be efficient (for DDE, their algorithm A took 15 hours and 30 minutes and algorithm B took 104.88 seconds to terminate [17]).

authors	ref.	results
Deutsch	[8]	The best routing obtained by using the branch-and-bound pro-
		gram (an exhaustive search) of [16] used 26 tracks.
Wang and Lee	[26]	$S^* = 27$ using their OCR algorithm (an exhaustive search).
Bernstein	[19]	There was a typographic error in $[8]$. The longest path length
		in the vertical constraint graph is 28 (this again may be a typo-
		graphic error since $v_{max} = 23$ for DDE).

Table 1: Conflicting results for Deutsch's difficult example.

R. K. Pal et al. [20] improved the traditional lower bound by considering the impact of the vertical constraints on the horizontal constraints. Note that the HCG is an interval graph, and hence is chordal (triangulated) [10, 20] (an undirected graph is called chordal if every cycle of length strictly greater than three possesses a chord, that is, an edge joining two nonconsecutive vertices of the cycle [10]). The chromatic number and maximal cliques in a chordal graph G = (V, E) can be computed in O(|V| + |E|) time [10, page 98].

For each shortest path from i to j in the VCG, an edge (i, j) is added to the HCG if the resulting graph is chordal (chordality of an undirected graph G = (V, E) can be tested in O(|V| + |E|) time [10, chapter 4]). They try to add as many edges as possible to the HCG while maintaining the chordality of the resulting graph. Finally, the size of the maximal clique in the modified HCG becomes a lower bound on S*. We will refer to the lower bound (the size of the maximum clique in the final modified HCG) computed by Pals' algorithm as LB1 in the rest of the paper.

Pals' approach [20] yielded quite significant improvements for several small examples (see Table 3). Pal et al. also reported that they computed a lower bound of 25 for DDE. However, the algorithm may not be efficient to implement in practice as the time complexity is $O(n^6)$ because it requires $O(n^2)$ time for each chordality test and in the worst case this test has to be executed $O(n^4)$ times (since there are $O(n^2)$ shortest paths). Furthermore, it is not clear in which order the paths should be scanned so that the modified HCG has the largest possible chromatic number (in [20], paths are scanned in increasing order by length and lexicographic label). Hence, a more efficient algorithm is needed to compute a tighter lower bound.

2.1 Motivation of Our Lower Bound Approach

The traditional lower bound, $\max\{d_{max}, v_{max}\}$, provides a good estimate of S* for many CRPs. Usually, d_{max} is the dominant component (for all examples but DDE in [28], the optimal solution equals d_{max}). However, if the combined effects of the horizontal and vertical constraints are not considered, the error can be as large as 100% as shown in the following example.

Consider the CRP in Figure 1, d_{max} is 4 and v_{max} is 3, but S* is 7. The lower bound cannot be improved by using the algorithm in [17]. For example, (5,4,3) is a path of length 3, but no net intersects with the path. The lower bound of 7 can be obtained by using LB1; however, it can be computed more efficiently. Consider the VCG, nets 5,6,7 must precede net 4, while net 4 must precede nets 1,2,3. Three tracks are required for nets $\{1,2,3\}$ and $\{5,6,7\}$ because of the horizontal constraints. Therefore, at least three tracks are needed above net 4, and three tracks are needed below it. Hence, at least 3+1+3=7 tracks are required for the CRP.

In the next two sections, we present two efficient lower bound algorithms, LB2 and LB3. Algorithm LB2 is based on labeling a directed acyclic graph G, which is essentially the VCG of the CRP. Algorithm LB3 improves on LB2 by separately handling the nets that cannot be placed in a track with other nets. We will show the performance of these lower bounds in Section 5.

3 A NEW LOWER BOUND, LB2

Our first lower bound algorithm LB2 is based on labeling a directed acyclic graph G, which is essentially the VCG of the CRP. In this section, we first introduce basic DAG terminology, then define some useful labels, and finally use these labels to compute a tighter lower bound for CRPs.

Let G be a directed acyclic graph. If there is an arc from i to j in G, then i is called a *parent* of j, and j is called a *child* of i. If there is a path from i to j in G, then i is called an *ancestor* of j, and j is called a *descendant* of i. The set of ancestors of i is denoted A;; the set of descendants of i is denoted D_i . A vertex i is called a *head* vertex if A; is empty, a *tail* vertex if D; is empty. If the *cost* c_i of each vertex i is one, then the cost of a path P $(\sum_{i \in P} c_i)$ is the number of vertices on the path. An induced subgraph of G with vertex set V is denoted as G[V]. For convenience, we will add two pseudo vertices 0 and X (with zero cost) to G, adding an arc from 0 to i if i is a head vertex and an arc from i to X if i is a tail vertex. Thus, G becomes a *single-entry, single-exit* DAG.

3.1 Labeling a DAG

In this section, we introduce various labels and co-labels (see Table 2) to compute a tighter lower bound for the channel routing problem. A label (height, density, lower-bound) is computed over the descendant set; a co-label (co-height, co-density, co-lower-bound) is computed over the ancestor set. Height h_i is the critical path length in the subgraph G[i + D;]. Density d_i is the density of $G[D_i]$. Lower-bound lb_i is computed by combining height and density. A counterpart of h_i , d_i , and lb_i can be computed similarly over the ancestor set.

label	notation	definition
height	h_i	$\max\{h_j: \mathbf{j} \in child(i)\} + \mathbf{c};$
co-height	h_i'	$\max\{h'_j: j \in parent(i)\} + c;$
density	d_i	$d_{max}(G[D_i])$
co-density	d'_i	$d_{max}(G[A_i])$
lower-bound	lbi	$\max \left\{egin{array}{ll} h_i,\ d_i+c_i,\ \max\{lb_j:j\in child(i)\}+c_i\end{array} ight.$
co-lower-bound	lb'i	$\max \begin{cases} h'_i, \\ d'_i + c_i, \\ \max\{lb'_j : j \in parent(i)\} + c_i \end{cases}$

Table 2: Definitions of height h_i , density d_i , lower-bound lb_i , and their co-label counterparts.

The following two lemmas are fundamental to the development of our new lower bound. Lemma 1 describes the way we partition a problem into subproblems to determine a tighter lower bound. Lemma 2 states that for any subproblem G' of G, its d_{max} and v_{max} are lower bounds of $S^*(G')$.

Lemma 1 [Partition] If A; is the set of ancestors of *i* and D_i is the set of descendants of *i*, then $S^*(G[A_i + i + D_i]) = S^*(G[A_i]) + c_i + S^*(G[D_i]).$

Proof: It follows from the fact that i must be assigned to a track below all ancestors of i and above all descendants of i.

Lemma 2 For any subgraph G' of G, $S^*(G') \ge d_{max}(G')$ and $S^*(G') \ge v_{max}(G')$, where $d_{max}(G')$ computes the density in G', $v_{max}(G')$ computes the longest path length in G'.

Lemma 3 Let $h_X = 0$, D; be the set of descendants of i. Define the *height* h_i of vertex i as $h_i := \max\{h_j : j \in child(i)\} + c_i$. Define the *density* of vertex i as $d_i := d_{max}(G[D_i])$. Then $S^*(G[D_i]) \ge d$; and $S^*(G[i + D_i]) \ge h_i$.

Proof: Note that h_i computes the length of a longest path from i to X, hence, $h_i = v_{max}(G[i + D_i])$. $G[D_i]$ and G[i + D;] are subgraphs of G. The results follow directly by Lemma '2. Theorem I defines the lower-bound lb_i of vertex i and proves that lb_i is a lower bound for $G[i + D_i]$.

Theorem 1 Let $lb_X = 0$ and define the lower-bound lb_i of vertex *i* as:

$$lb_i = \max \begin{cases} h_i, \\ d_i + c_i, \\ \max\{lb_j : j \in child(i)\} + c_i \end{cases}$$

Then $S^*(G[i + D_i]) \ge lb_i$

Proof: It can be proven by induction on depth.

- (i) <u>basis</u>: $S^*(X) \ge lb_X$.
- (ii) <u>hypothesis</u>: suppose $S^*(G[j + D_j]) \ge lb_j$.

(iii) <u>induction</u>: Let i be a parent of j. Three inequalities must be maintained:

- $S^*(G[i + D;]) \ge h_i$ by Lemma 3.
- $S^*(G[D_i]) \ge d_i$ by Lemma 3. By Lemma 1, $S^*(G[i + D_i]) = c_i + S^*(G[D_i])$. It follows that $S^*(G[i + D_i]) \ge c_i + d_i$.
- $S^*(G[D_i]) \ge S^*(G[j + D_j]) \ge lb_j$ because $G[j + D_j]$ is a subgraph of $G[D_i]$. By Lemma 1, $S^*(G[i + D_i]) = c_j + S^*(G[D_i])$. It follows that $S^*(G[i + D_j]) \ge c_i + lb_j$.

The conclusion follows directly.

The duals of Lemma 3 and Theorem 1 for co-labels are parallel to the previous proofs. Algorithm LB2 computes a tighter lower bound than traditional lower bounds for a CRP. To compute the density and co-density requires finding the transitive closure of G. The transitive closure of a

Algorithm LB2(vc, hc)

- 1. compute the transitive closure vc^{S} and transitive reduction vc^{-} of vc
- 2. construct VCG from vc^-
- 3. compute the density d; and co-density d'_i for each vertex
- 4. compute the height h; and co-height h'_i for each vertex
- 5. compute the lower-bound lb_i and co-lower-bound lb'_i for each vertex
- 6. return $\max\{lb_i c; + lb'_i : 0 \le i \le X\}$

DAG can be computed in $O(n^3)$ time [1, 3, 6]. It can also be obtained by logn matrix multiplications. Each matrix multiplication can be computed in $O(n^{2.81})$ time by using Strassen's algorithm [1, 6]. The other labels $(h_i, h'_i, lb_i$ and $lb'_i)$ can be computed in a depth-first fashion in O(n + |G|)time [6], where n is the number of vertices and |G| is the number of arcs in G. Hence, the overall time complexity is $O(n^3)$ (or $O(n^{2.81} \log n)$), which is dominated by the time for computing the transitive closure of G. Theorem 2 demonstrates that LB2 computes a lower bound for a CRP.

Theorem 2 $S^* \ge LB2 = \max\{lb_i - c_i + lb'_i : 0 \le i \le X\}$

Proof: For each vertex $i, S^*(G[D_i]) \ge lb_i - c_i$ by Theorem 1. Similarly, $\mathbf{S}^*(G[A_i]) \ge lb'_i - c_i$. Hence, by Lemma 1, $S^*(G[A_i + \mathbf{i} + \mathbf{D};]) = S^*(G[A_i]) + c_i + S^*(G[D_i]) \ge (lb'_i - c_i) + c_i + (lb_i - c_i) = lb_i + lb'_i - c_i$. It follows that $\mathbf{S}^* \ge \max\{S^*(G[A_i + \mathbf{i} + D_i]) : 0 \le \mathbf{i} \le X\} \ge \text{LB2}$.

It is worth noting that our lower bound LB2 subsumes the traditional lower bounds, d_{max} and v_{max} since $d_0 = d_{max}$ and $h_0 = v_{max}$ by definition.

Lemma 4 LB2 $\geq d_{max}$, LB2 $\geq v_{max}$.

4 A TIGHTER LOWER BOUND, LB3

Although LB2 performs very well for many CRPs (as we will see in Section 5), improvements can still be made if we consider the criticalnets in a CRP. Two nets i and j are said to be incompatible if $vc^+(i, j) = 1$ or $vc^{S}(j, i) = 1$ or hc(i, j) = 1, that is, there is either a horizontal constraint or a (transitive) vertical constraint between them. Obviously, incompatible nets cannot be assigned to the same track. We can construct an undirected graph, the *InCompatibility* Graph (ICG), where

vertices represent the nets and edges represent the incompatibility relation between nets. Thus, the cardinality of the maximal clique in the ICG is a lower bound on S^* . We say that a net *i* is critical if it is incompatible with all other nets. For example, in Figure 1, all nets are critical (e.g., net 4 has vertical constraints with all other nets, net 5 has vertical constraints with nets 1,2,3,4 and horizontal constraints with nets 6,7).

A critical net must occupy an individual track (no other nets can share the track with it). Note that the set of critical nets is contained in all cliques of the ICG. Hence, the critical nets can be factored out as described in the following lemmas.

Lemma 5 If S is a set of critical nets in V, then $S^*(V) = |S| + S^*(V \setminus S)$

Proof: In an optimal routing of V, each net in S occupies an individual track (|S| tracks are required by these critical nets), and the nets in V\S (the set difference of V and S) share the other $S^*(V) - |S|$ tracks.

Algorithm LB3(vc, hc)

- 1. compute the transitive closure vc^+ of vc
- 2. find the set S of critical nets
- **3.** if n |S| < 1 **then** return n
- 4. construct vc' and hc' from vc^+ and hc respectively by eliminating nets in S
- 5. return |S| + LB2(vc', hc')

Algorithm LB3 computes an improved lower bound for a CRP. It first computes the transitive closure vc^+ of vc which represents the vertical constraints. This step is essential as we want to eliminate the critical nets (S) without changing the precedence constraints (reachability) for the remaining nets ($V \setminus S$). For example, in Figure 1, if we just remove net 4 in the VCG, the path from 5 to 1 would be lost. To recover the vertical constraint from 5 to 1, a transitive arc from 5 to 1 should be added to the VCG. Lemma 6 justifies the correctness of adding transitive arcs.

Lemma 6 [Equivalence] Let the VCGs of two CRPs be G_1 and G_2 respectively. If G_1 and G_2 are transitively equivalent (i.e., have the same transitive closure) and both CRPs have the same HCGs, then both CRPs are equivalent, that is, a feasible routing for one CRP is feasible for the other. In other words, the solution of a CRP is not changed by adding the transitive arcs in the VCG.

Proof: As G_1 and G_2 are transitively equivalent, there is a path from i to j in G_1 if and only if there is a path from i to j in G_2 . Hence, a feasible routing of one CRP is feasible for the other since both CRPs have equivalent constraints.

The time complexity of algorithm LB3 is $O(n^3)$ (or $O(n^{2.81} \log n)$), which is dominated by the time to compute the transitive closure of vc. However, the problem size is reduced from n to n - |S| when LB2 is called to compute a lower bound for nets in $V \setminus S$. In Section 5, we observe a significant improvement in the lower bound by using LB3 for many large scale CRPs.

An optimal routing of a CRP example (RKPC3 in [20]) is shown in Figure 2. The (initial) VCG and HCG of the CRP are shown in (a) and (b). It is easy to see that v_{max} is 3 in (a) and d_{max} is 4 in (b). Nets 1, 2 and 3 are critical nets, hence, $\mathbf{S} = \{1, 2, 3\}$. The VCG and HCG after factoring out nets 1,2,3 are shown in (c) and (d). Note that for each pair of nets in V\S ($\{4, 5, 6\}$), the vertical constraints and horizontal constraints are the same before and after the critical nets are removed. That is, for each pair of nets i, j in $V \setminus S$, there is a path from i to j in (a) if and only if there is a path from i to j in (c); there is an edge between i and j in (b) if and only if there is an edge between i and j in (d). It is easy to see that v_{max} is 2 in (c) and d_{max} is 2 in (d). Hence, a lower bound of the original CRP is $3 + \max\{2, 2\} = 5$.

Theorem 3 LB3 computes a lower bound for a CRP

Proof: As the solution is not changed by adding the transitive arcs in the VCG by Lemma 6, we can simply consider vc^+ . Note that vc'(hc') is obtained from $vc^+(hc)$ by eliminating the critical nets (S). Hence, both the vertical and horizontal constraints between each pair of nets in V\S are the same before and after the critical nets are removed. By Lemma 5, the total number of tracks required equals the number of nets in S plus the tracks required by the nets in $V \setminus S$.

- Given an optimal routing for the nets in V, an optimal routing for the nets in $V \setminus S$ can be constructed simply by removing the tracks occupied by the nets in S.
- Given an optimal routing for the nets in $V \setminus S$, an optimal routing for the nets in V can be constructed as follows:



Figure 2: A CRP example (RKPC3) for demonstrating LB3

for each $i \in S$

 $t_1 :=$ the lowest track in which an ancestor of *i* is assigned $t_2 :=$ the highest track in which a descendant of *i* is assigned create a new track and assign *i* to the track insert the track between tracks t_1 and t_2

end

LB2 computes a lower bound given vc' and hc' (which represents the problem of routing $V \setminus S$). Hence, to route V requires at least |S| + LB2(vc', hc') tracks.

5 EXPERIMENTAL RESULTS

To test the effectiveness of our lower bound algorithm, we have implemented the algorithms in C language on a Sun SPARC/5 workstation running SunOS 5.3. For channel routing problems, Deutsch's examples [8] are used extensively as benchmarks for evaluating the performance of new algorithms, especially the so-called Deutsch's difficult example. For all these examples but DDE, the density is a tight lower bound (a routing using d_{max} tracks can be obtained by using Yoshimura's algorithm [28]); hence, they are not considered in this paper. The lower bounds for DDE and the examples in [20] are compared in Table 3. Note that both LB2 and LB3 compute a lower bound of 26 for DDE.

In addition to the benchmarks reported in the literature, we have also tested some randomly generated examples using the channel routing generator in [5]. Our benchmark examples are listed in Tables 6 and 7. There is a net number in each column. A 0 in a column indicates that there is no net connected to the column in that row. For instance, an optimal routing of HYC1 is shown in Figure 3.

The performance of various lower bounds for our benchmarks are compared in Tables 4 and 5, where LB1 corresponds to the maximal clique size in the final HCG by using the algorithm in [20]. The computed lower bounds are compared in Table 4; the running times are compared in Table 5, where #(test) is the number of chordality tests LB1 has performed. :Some observations can be drawn from the experimental results:

Our lower bounds are much tighter than the traditional lower bounds $(d_{max} \text{ and } v_{max})$, which

problem	problem d_{max}		[20]	LB2	LB3	optimal
	19	23	25	26	26	28
RKPC1	3	3	4	3	3	4
RKPC2	3	4	6	6	6	6
RKPC3	4	3	5	4	5	5
RKPC4	4	4	5	5	5	5
RKPC5	4	4	6	5	5	6
RKPC6	4	5	7	6	6	7
RKPC7	5	3	7	7	7	7
RKPC8	5	5	7 7		7	7
RKPC9	6	6	10	9	9	10

Table 3: Comparison of lower bounds for some benchmark CRPs.





indicates that it is very important to consider the interaction of constraints for multipleconstraint problems.

- LB3 has achieved a significantly tighter lower bound than LB2.
- Our algorithms LB2 and LB3 are much more efficient than LBI. For instance, it took about 25 minutes for LBI to compute a lower bound of 62 for HYC8; while LB3 computed the same lower bound in less than 0.03 seconds.
- For DDE, HYC4, and HYC7, LB3 is a tighter lower bound than LBI. For other cases, LB3 is the same as or very close to LBI (off by one).

The timings were obtained by using gettimeofday in the standard C library. The initialization overheads were ignored. All algorithms assume that the horizontal constraints (represented by hc) and the vertical constraints (represented by vc) are given. In our implementation, HCG and VCG are in linked list form, hc and vc are in bit matrix form. For LBI, the HCG and hc need to be updated whenever an edge is added. As the time required for computing the labels h_i , h'_i , lb_i and lb'_i is proportional to the number of arcs in the VCG, we compute the transitive closure and transitive reduction [3, 11] before the VCG is constructed.

6 CONCLUSION

In this paper, we have presented two efficient algorithms, LB2 and LB3, for computing a tighter lower bound for the channel routing problem. Algorithm LB2 is based on partitioning and labeling a directed acyclic graph. Algorithm LB3 improves LB2 by factoring out the critical nets. Our algorithms have efficiently obtained a tighter lower bound than previous work for Deutsch's difficult example and some other large scale CRPs.

The capacity constraints and precedence constraints are two important constraints for many optimization problems. For the channel routing problem, the vertical constraint is a precedence constraint, while the horizontal constraint can be considered a capacity constraint. To determine a tight lower bound for problems with these two constraints requires careful consideration of the interaction of constraints. The presented technique provides an efficient way of integrating the precedence constraints and capacity constraints, and is applicable to other problems impacted by these two constraints (for example, the data dependency graphs in superscalar pipeline scheduling problems [4, 23] and microcode compaction problems [7, 9]).

problem	d_{max}	v_{max}	LB1	LB2	LB3	optimal			
HYC1	7	7	8	8	8	8			
HYC2	8	7	9	9	9	9			
НҮС3	11	8	15	15	15	15			
HYC4	20	22	31	31	33	?			
HYC5	35	32	45	39	44	45			
HYC®	50	42	66	56	65	?			
HYC8	39	49	6 Ø	40	6 Z	62			

Table 4: Comparison of computed lower bounds for our benchmark CRPs.

Table 5: Comparison of running times (seconds) for our benchmark CRPs.

problem	LB1	#(test)	LB2	LB3
HYC1	0.008658	11	0.001155	0.000814
HYC2	0.008105	10	0.000952	0.000698
НҮС3	0.037368	41	0.002006	0.000874
HYC4	318.003003	77745	0.021058	0.103750
HYC5	14.590356	2819	0.026399	0.014310
HYC6	268.003396	26109	0.053892	0.019800
HYC7	756.686119	91872	0.046057	0.413557
HYC8	1499.201131	182601	0.054984	0.027951

References

- [1] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Deszgn and Analysis of Computer Algorithms*. Addison-Wesley Publishing Company, San Francisco, CA, 1976.
- [2] H. Y. Chao. Finding a Tighter Lower Bound for Optimization Problems with Capacity and Precedence Constraints. PhD thesis, Purdue University, 1994.
- [3] H. Y. Chao and M. P. Harper. Minimizing redundant dependencies and interprocessor synchronizations. In Proceedings of the Sixth IASTED/ISMM International Conference on Parallel and Distributed Computing and Systems, pages 294–298, October 1994.
- [4] H. Y. Chao and M. P. Harper. Scheduling a superscalar pipelined processor without hardware interlocks. Technical Report TR-EE 94-29, Purdue University, September 1994.
- [5] H. Y. Chao and M. P. Harper. A difficult channel routing generator. Technical Report TR-EE 95-1, Purdue University, January 1995.
- [6] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. Introduction to Algorithms. McGraw-Hill Book Company, New York, NY, 1990.
- [7] S. Davidson, D. Landskov, B. D. Shriver, and P. W. Mallett. Some experiments in local microcode compaction for horizontal machines. *IEEE Transactions on Computers*, C-30(7):478-477, July 1981.
- [8] D. N. Deutsch. A dogleg channel router. In Proceedings of the 13th Design Automation Conference, pages 425–433, 1976.
- [9] J. A. Fisher. Trace scheduling: A technique for global microcode compaction. *IEEE Transactions on Computers*, C-30(7):478-490, July 1981.
- [10] M. C. Golumbic. Algorithmic Graph Theory and Perfect Graphs. Academic Press, New York, 1980.
- [11] D. Gries, A. J. Martin, Jan L. A van de Snepscheut, and J. T. Udding. An algorithm for transitive reduction of an acyclic graph. *Science of Computer Programming*, 12:151-155, 1989.
- [12] S. Haruyama, D. F. Wong, and D. S. Fussell. Topological channel routing. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 11(10):1177-1197, October 1992.
- [13] A. Hashimoto and S. Stevens. Wire routing by optimizing channel assignment within large apertures. In Proc. 8th Annual Deszgn Automation Workshop, pages 155–169, 1971.
- [14] T. T. Ho. A density-based greedy router. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 12(7):973–981, 1993.

- [15] T. T. Ho, S. S. Iyengar, and S. Q. Zheng. A general greedy channel routing algorithm. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 10(2):204–211, February 1991.
- [16] B. W. Kernighan, D. G. Schweikert, and G. Persky. An optimum channel-routing algorithm for polycell layouts of integrated circuits. In *Proceedings of the 10th Design Automation Workshop*, pages 50–59, 1973.
- [17] K. K. Lee and H. W. Leong. An improved lower bound for channel routing problems. *IEEE International Symposium on Circuits and Systems*, 4:11–14, June 1991.
- [18] Z. M. Lin. An efficient optimum channel routing algorithm. In IEEE Proceedings of the SOUTHEAST-CON '91, volume 2, pages 1179–1183, 1991.
- [19] T. Ohtsuki. Layout Design and Verification, volume 4. Elsevier Science Publishers B.V., Amsterdam, 1986.
- [20] R. K. Pal, S. P. Pal., and A. Pal. A new lower bound for channel routing. In Proceedings of the IEEE Region 10 Conference on Computer, Communication, Control and Power Engineering, pages 507–510, October 1993.
- [21] J. Reed, A. Sangiovanni-Vincentelli, and M. Santomauro. A new symbolic channel router: YACR2. IEEE Transactions on Computer-Aided Design, CAD-4(3):208-219, July 1985.
- [22] R. L. Rivest and C. M. Fiduccia. A greedy channel router. In 19th Design Automation Conference, 1981.
- [23] Y. H. Shiau and C. P. Chung. Adoptability and effectiveness of microcode compaction algorithms in superscalar processing. *Parallel Computing*, 18(5):497–510, 1992.
- [24] S. Soh, S. Rai, and R. Mehrotra. An efficient approach for channel routing in VLSI. Computers Electrical Engineering, 17(2):105–112, 1991.
- [25] T. G. Szymanski. Dogleg channel routing is NP-complete. IEEE Transactions on Computer-Aided Design, CAD-4(1):31-41, 1985.
- [26] J. S. Wang and R. C. T. Lee. An efficient channel routing algorithm to yield an optimal solution. IEEE Transactions on Computers, 39(7):957-962, July 1990.
- [27] U. Yoeli. A robust channel router. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 10(2):212–219, February 1991.
- [28] T. Yoshimura and E. S. Kuh. Efficient algorithmsfor channel routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, CAD-1:25–35, 1982.

problem			column													
HYC1	top	8	1	0	4	0	2	1	4	3	9	5	0	0	0	0
		7	0	6	5											
	bottom	9	2	10	6	0	4	3	5	4	10	8	0	0	0	0
		9	0	7	7											
HYC2	top	6	8	0	3	0	3	9	0	3	0	1	2	7	4	0
		5	0	2	0	1										
	bottom	9	9	10	5	0	6	10	0	7	0	4	3	9	8	0
		8	0	4	0	3										
HYC3	top	10	7	6	1	1	2	2	3	7	4	1	3	6	8	5
		5	9	11	12	8	13	13	0	0						
	bottom	11	9	7	3	5	4	6	8	11	7	2	6	10	10	8
L		6	12	12	13	9	14	15	14	15						
HYC4	top	1	4	4	2	3	2	4	1	6	3	20	4	12	15	14
		15	9	10	9	9	9	14	11	15	12	10	23	13	17	13
		16	18	18	21	24	8	5	12	20	19	19	22	16	21	11
		7	23	18	23	8	24	25	26	26	27	28	29	30	31	31
		32	33	33	34	35	35	36	37	37	37	38	38	38	39	39
	1 44	39	39	40	40	41	41	41	42	43	44	44	46	47	10	20
	bottom	3	9 22	14	4	4	5 95	18	16	8 19	10	22	17	1.5	19	32
		20	32 22	14	29 22	21	20	20	10	10	11	14 91	32 20	10	19	14
		10	22	21	22	20 12	30	0 28	20 31	24	२२ २१	21	29	10 31	24 33	10
		30	35	20 40	35	36	38	20 37	39	42	43	41	44	48	44	45
		46	48	41	46	43	47	49	46	50	47	49	49	50	45	10
HVC5	ton	20	28	30	36	<u>6</u>	0	45		0	0		0			0
11105	ιοp	12	20	0	8	45	1	10	4	10	18	9	43	34	40	16
		10	0	48	45	49	20	Ő	0	43	0	0	38	0	22	34
		0	Õ	0	0	2	14	0	0	0	0	0	23	0	0	5
		0	37	44	0	0	9	0	0	0	33	0	0	0	0	0
		15	0	0	0	0	0	0	0	0	0	0	0	32	7	0
		0	11	25	19	15	0	41	24	17	0	0	3	4:0	30	7
		0	0	10	41	0	0	1	0	4	39	35	0	0	26	0
		0	0	41	0	21	0	0	0	29	46	46	0	31	0	14
ĺ		27	0	0	13	4 0	0	42	0	24	0	0	0	0	39	
	bottom	0	32	31	38	7	0	48	0	0	0	0	0	0	0	0
		14	8	0	9	47	2	0	8	12	19	12	44	35	42	19
		U	0	50	49	50	21	0	0	49	U	0	39 ე∤	U	23	37
		U	ປ ງຄ	U 19	U	6 0	17	U	U A	U	U 24	U	24	U	U	0 A
		0 16	აშ ი	40 0	U O	U O	11	U O	0	U O	ა <u>4</u> ი	0	0	บ ว.ว.	u Q	0
		01	U 12	20	0 20	0 19	U N	43	0 28	10	0	0 A	5	ن. 11	32	10
		0	10	29 11	20 45	10	n	40 4	20 0	13 7	41	36	0		$\frac{52}{27}$	10
		0	0	42	10	22	Ő	0	0	30	47	48	õ	33	0	15
	1	29	Ō		14	45	Õ	46	0	25	0	0	0	0	4 0	-

Table 6: Random CRP examples - set I.

problem								С	olum	n						
HYC6	top	5	6	5	12	4	19	17	18	25	27	31	31	32	28	36
	-	3	4	2	10	32	38	28	34	34	24	22	13	42	40	21
		24	24	32	46	1	58	41	16	50	59	14	47	45	43	57
		20	56	45	62	63	38	55	46	8	39	64	1	8	30	3
		39	11	6	44	47	30	58	48	29	63	60	28	35	60	15
		19	55	25	48	33	52	33	6	7	43	0	9	42	23	30
		35	0	69	27	37	2	35	26	51	37	34	54	67	49	59
		68	0	50	0	53	61	66	70	0	62					
	bottom	8	10	7	13	6	21	18	19	29	35	38	37	38	33	41
		5	5	4	11	36	41	32	39	38	28	23	14	44	42	22
		26	25	40	50	4	61	42	17	51	60	15	54	47	44	59
		24	58	46	65	65	43	57	49	10	43	66	3	12	33	9
		40	13	12	45	50	32	60	54	30	64	62	31	43	63	16
		20	56	27	50	36	55	34	7	13	48	0	13	48	24	31
		37	0	70	30	40	9	36	30	53	41	37	55	68	54	61
		69	0	52	0	55	69	67	0	0	64					
HYC7	top	26	21	24	7	18	27	22	30	34	34	17	28	9	13	28
		33	20	15	31	32	10	4	26	6	23	13	3	10	31	29
		9	25	18	28	12	15	27	37	7	20	33	10	21	20	38
		11	8	20	16	16	19	23	14	17	34	10	46	34	16	47
		26	24	38	11	37	33	37	17	42	4	7	6	12	29	14
		39	1	41	42	44	45	24	42	4 1	43	36	36	35	43	44
		39	45	12	5	5	5	43	39	2	40	45	46	46	47	4 8
		48	49	4 9	50	50	51	52	52	52	53	54	54	55	56	56
		56	56	57	57	58	58	59	59	60	60	60	60	61	61	61
		63	63	64	64	66	67	0	0	0						
	bottom	30	27	29	14	40	40	28	31	48	42	31	39	12	27	36
		41	22	20	34	35	17	12	39	15	28	16	5	16	53	31
			31	22	32	15	18	39	53	19	30	39	23	23	25	50
		15	9	33	20	18	21	24	23	32	44	15	50	38	29	58
		32	26	49	19	43	38	42	27	51	16	8	26	26	37	15
		50	13	44	45	63	49	25	47	40	44	42	41	37 121	40	41
		53	55	27	12	17	11	45	44 50	10	43	54	48	01 60	50	49
		60	08 67	60	02 61	50 50	67	00 64	00 65	03 64	57	09 66	70	69	09 62	02 67
		64	66	68	60	60	68	1	00	04 2	05	00	10	04	05	01
TIVO		04	00	15	09		10	1			0.0	10			10	1.0
HYC8	top	26	4	15	9	30	10	9 12	.7 C	20	26	12	7	9 10	19	10
		24	15	14	24	30	11	13	0	27	2	10	3	18	25	3
			11	5 20	27	22	29	2	31	32	28 20	30	8 41	40 40	40	0 42
		32	33	29	31	20 46	34	30	31	30	39	40	41	4£2 E O	42	43
		43	44 50	40	40	40 50	40 54	41 57	40 55	40 55	49	56	57	50	57	59
		52	02 60	02 61	23 62	55 62	04 63	54 64	55 65	66 66	55 67	68	60	51	70	00
	hottom	30	5	18	11	35	12	12	8	21	20	14	10	12		17
	Dottom	20	0 16	10 15	11 25	31 21	14	10	0 7	30	23 1	12	7	.⊥⊿ 10	$\frac{44}{27}$	1 I 1
		20	10	10	20 28	23	10 21	14 A	33	37	35	37	à	22	24	ч а
		34	34	32	37	20 20	36	38	38	39	40	41	42	43	44	45
		47	45	46	48	49 49	50	48	50	52	53	51	53	54	55	56
		54	55	$\frac{1}{57}$	- <u>40</u> 55	-10 56	59	60	58	59	60	57	58	59	60	61
		61	61	62	63	66	64	65	68	67	68	69	70	1	0	~ *

Table 7: Random CRP examples - set II.