

Wire Type Assignment for FPGA Routing

Seokjin Lee
Department of Electrical and
Computer Engineering
The University of Texas at
Austin
Austin, TX 78712
seokjin@cs.utexas.edu

Hua Xiang, D. F. Wong
Department of Electrical and
Computer Engineering
University of Illinois at
Urbana-Champaign
Urbana, IL 61801
huaxiang,
mdfwong@uiuc.edu

Richard Y. Sun
Xilinx Inc.
2100 Logic Drive
San Jose, CA 95124
ysun@xilinx.com

ABSTRACT

The routing channels of an FPGA consist of wire segments of various types providing the tradeoff between performance and routability. In the routing architectures of recently developed FPGAs (e.g., Virtex-II), there are more versatile wire types and richer connections between them than those of the older generations of FPGAs (e.g. XC4000). To fully exploit the potential of the new routing architectures, it is beneficial to perform wire type assignment for all channels as an intermediate stage between global routing and detailed routing. In this paper, we present a wire-type assignment algorithm that is based on iteratively applying min-cost max-flow technique to simultaneously route many nets. At each stage of the network flow computation, we have guaranteed optimal result in terms of routability and delay cost. We use the routing architecture of the Virtex-II FPGAs from Xilinx as a target architecture in our experiments. Experimental results show that our algorithm outperforms the traditional sequential net-by-net approach.

Categories and Subject Descriptors

B.7.2 [Integrated Circuits]: Design Aids—*Placement and routing*

General Terms

Algorithms, Experimentation

Keywords

FPGA routing, wire type assignment, min-cost flow algorithm

1. INTRODUCTION

As the gate density of FPGAs grows larger, the routing architectures of FPGAs have changed significantly. The num-

ber of routing segments per channel has greatly increased, and the structure of the switch blocks have become very complicated to accommodate more connections between the wire segments. To take advantage of the tradeoff between interconnection delay and routability, most of the recent FPGAs have several types of routing wires in which each type of wire exhibits different length and connectivity. Longer wire segments are intended for high-fanout, time-critical signal nets. On the contrary, shorter wire segments are intended for short connections to avoid wasting routing resources. These wires with versatile set of length are placed in hierarchically designed channels or placed in the same channels using different kinds of switching components [3, 14]. To utilize routing resources even more efficiently, some of the recently developed FPGA architectures adopt several different wire segment types with different connectivity even though they have the same length [14]. Because routing is the dominating factor in determining the performance of the overall FPGA system and because the routing resources take up a significant portion of the chip area, it is very important for the routing algorithms to fully exploit the potential of the new routing architectures.

Existing FPGA routing algorithms can be classified into two categories. One of them is a two-stage approach in which global routing and detailed routing are performed sequentially [4, 6, 11, 15], and the other is a one-stage approach in which only a detailed routing algorithm is used [9, 10, 12, 13]. Detailed routers use detailed graphs to model the target architectures. Because they are applied to the detailed graphs that model each component of the routing resources directly, detailed routing algorithms can give very accurate routing results. However, these algorithms may not be able to handle big graphs, which model the routing architectures of the recent FPGAs. For example, the largest Virtex-II FPGA from Xilinx contains close to 60 million edges and 6 million nodes in the routing graph [8], and the graph size will grow for future FPGAs. In a two-stage approach, the global router abstracts the details of the routing architecture, and the routing is performed on a coarser graph to assign a series of channels to each path used by a signal, and the detailed router routes each signal on the detailed graph along the channels determined by the global router. Although this approach may be able to handle large graphs, it is possible that the coarse routing result determined by global routing may not be accurately refined into the underlying detailed routing. This is especially due to

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

FPGA'03, February 23–25, 2003, Monterey, California, USA.
Copyright 2003 ACM 1-58113-651-X/03/0002 ...\$5.00.

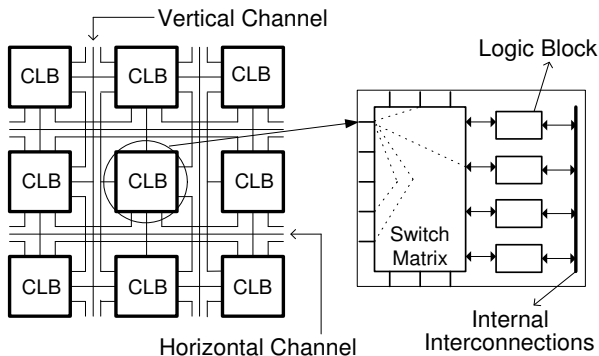


Figure 1: Target FPGA architecture

the special routing structures of FPGAs, such as versatile length of routing wires and limited connectivity between the wire segments.

The problem of assigning proper types of wire segments to nets was incorporated in the global routing stage in [6, 15], and it was performed in the detailed routing stage in [4]. But in both cases, wire type assignment is applied to conventional XC4000-style routing architectures where different types of wire segments are sparsely linked, and they used net-by-net approaches, which may suffer from net ordering problems.

In this paper, we consider the wire type assignment as a procedure in a separate stage between global routing and detailed routing. We modeled our Virtex-II-style architecture with a *wire type connection graph* in which each node abstracts all the wire segments from each CLB (configurable logic block) and each edge represents all the connection switches between the wire types. The wire type connection graph is coarser than the graph used by typical detailed routers, and it is finer than the grid routing graph used by most of the global routers. As the nature of the graph suggests, the wire type assignment stage can play a role of reducing inconsistency between global routing and detailed routing.

With global routing result given for each net, our algorithm solves the problem of wire type assignment for all the net segments in a channel. First, we find the wire type assigned routes for all the net segments from one CLB to all the other CLBs in a channel using a polynomial-time exact algorithm. Our algorithm is based on min-cost flow computations, and it is guaranteed to find a congestion-free wire type assignment solution if one exists. Furthermore, it can find a solution with minimum total delay at the same time. Applied on each CLB in a channel iteratively, it provides a randomized polynomial-time algorithm to find the wire type assigned routes for all the net segments in a channel. The optimality of our algorithm also can be very helpful for incremental improvement of the routing result through the interaction with the global router or the detailed router. Although we targeted the Virtex-II-style routing architecture, we believe our algorithm can be used in most of the recent architectures that feature various wire types.

The rest of the paper is organized as follows. Some preliminary notation and the architectural model used in this paper are introduced in Section 2. The wire type assignment

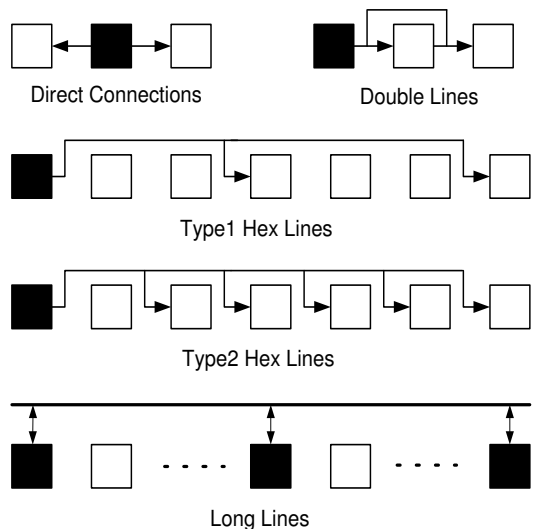


Figure 2: Virtex-II wire types

problem is defined in Section 3. In Section 4, we present the flow network graph construction scheme and our network flow based algorithm for assigning wire types to nets. Experimental results are shown in Section 5, and our conclusions are in Section 6.

2. PRELIMINARIES

The model for FPGAs assumed in this paper is an array-based FPGA which is similar to the Xilinx Virtex-II architecture. It is quite different from the XC4000-style architecture used in most of the FPGA routing algorithms [4, 6, 9, 11, 13]. As shown in Figure 1, it consists of configurable logic blocks (CLBs) and interconnection wires. A CLB consists of a switch matrix, logic blocks, and internal interconnections. The switch matrix in a CLB performs the role of a switch module and connection modules of XC4000-style architectures. The connections among global wire segments and the connections between the global wire segments and logic module pins are made within the switch matrix. Each of the vertical and horizontal channels has several types of wire segments, and all channels have the same structure.

Figure 2 shows the wire types used in the Virtex-II. The internal CLB local interconnections from logic block outputs to logic block inputs are omitted because our attention is focused on the global interconnections. Different from those of conventional XC4000-style architectures, the routing resources of the Virtex-II architecture include both unidirectional and bidirectional wires. Among the global wire segments, the long lines are bidirectional wires and they span the full height and width of the device. All other wires are directional wires. Organized in a staggered pattern, directional wires can be driven only from one end. As in the case of hex lines, lines with the same length can have different connectivity to the CLBs. Unlike the switch modules of the XC4000-style architecture, the connection topology in a switch matrix of our model is quite irregular. Each type of wire segments has different connection topology. For example, a type2 hex line in a vertical channel has connection to vertical double lines, vertical type1 hex lines, horizontal

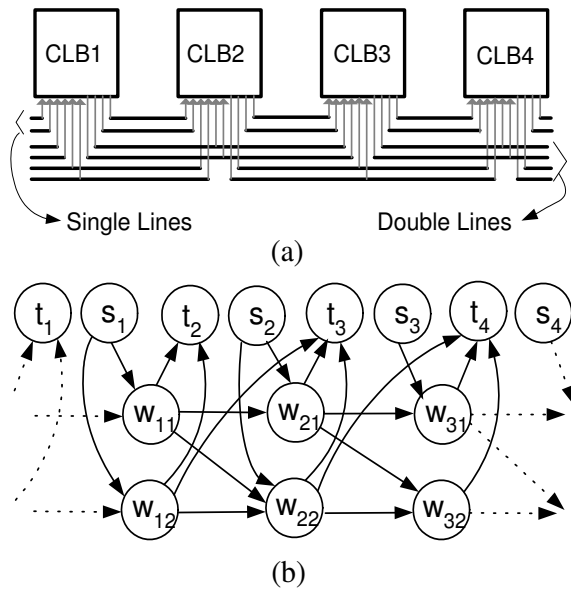


Figure 3: The graph representation of the routing resources: (a) Partial view of a horizontal channel (only single lines and double lines of one direction are shown) (b) Corresponding wire type connection graph

type1 hex lines, and horizontal double lines. But a vertical double line has connection to horizontal double lines and vertical double lines in the same switch matrix. All the switch matrices have the same connection topology, and all the connection switches between the wires are buffered switches.

Because our algorithm is applied to a general graph, we do not assume any particular length, direction, or connectivity of the wire types. To elaborate the results from global routing, the graph we are using for our wire type assignment algorithm is a more detailed graph than the grid routing graph, which is commonly used in most of the global routing algorithms [5, 6, 15]. At the same time, it is much coarser than the detailed routing resource graph used in detailed routing algorithms. We modeled the routing structure in a channel of the Virtex-II style architecture with the wire type connection graph $G_w(V_w, E_w)$.

The set of nodes V_w represents wire types or CLBs. Each node, which corresponds to a wire type, encapsulates all wire segments of the same type driven from a CLB. If two wire segments have the same length, direction, and connectivity, we consider their wire type to be the same. To model the starting point and the end point of signals in a channel, each CLB is modeled with two nodes, a CLB source node and a CLB sink node. The set of edges E_w represents architecture-specific connections among the wire types or connections between CLBs and the wire types. Although the actual connections between the wires are made at a switch matrix in a CLB, the connections between the wire types in a channel are represented by edges between the nodes. An edge between a wire type node and a CLB sink node represents the connections from the wire segments of that type to the input pins of logic blocks or to the wire segments of other

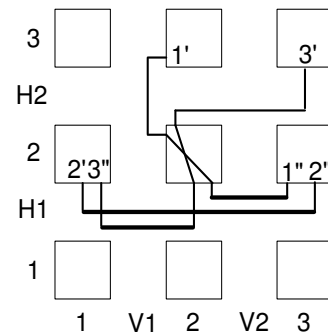


Figure 4: An example of globally routed nets.

channels. Similarly, an edge between a CLB source node and a wire type node represents the connections to the wire segments of that type from the output pins of logic blocks or from wire segments of other channels. Figure 3 (a) shows a partial view of a routing structure in a channel, and its graph representation is shown in Figure 3 (b). For simplicity, only single lines and double lines of one direction are shown in this figure. Node t_i 's are the CLB sink nodes, and s_i 's are the CLB source nodes. The single lines and double lines driven by i th CLB are represented by node w_{i1} 's and w_{i2} 's, respectively. Edges between w_{ij} 's and t_k 's model the connections from type j wire segments which are driven by i th CLB and connected to k th CLB. From this wire type connection graph, we construct the flow network [7] which will be used in our algorithm.

3. PROBLEM DEFINITION

In two-stage routing algorithms for array-based FPGAs, global routing is performed on a coarser grid routing graph where the edges represent channels and the nodes represent connections between the neighboring channels. After global routing, each net can have a global route from its source pin to sink pins in terms of the sequence of channels and switch blocks it encounters. The detailed router decides wire segments and connection switches along the global route for each path. But the selection of a proper wire type for each wire segment along the route is very important for both effective use of resources and timing performance of the final routing results.

Because all the switching blocks as well as logic blocks are included in CLBs, each globally routed portion of a net within a channel (we call it a *net segment*) can be expressed with an interval between the two CLBs it spans in the form of (index of source CLB node, index of target CLB node), which is denoted by a *spanning interval* of a net segment.

Figure 4 shows an example of a global routing result for 3 nets. Each CLB is indexed from left to right horizontally, and bottom to top vertically. Net1 is routed through the (3, 2) portion of the vertical channel V1 and the (2, 3) portion of the horizontal channel H1. Similarly, net2 is routed through the (1, 3) portion of H1 channel, and net3 is routed through the (3, 2) portion of H2 channel and the (2, 1) portion of H1 channel. In H1 channel, there are 3 net segments, and their spanning intervals are (2, 3), (1, 3), and (2, 1). By finding a path from a CLB source node to a CLB sink node for a net segment in the wire connection graph, we can assign proper

types of wires to route the net segment in a channel. We can alleviate the net ordering problem, which can be found in some net-by-net approaches, by finding paths for all the net segments from the same CLB source node simultaneously.

Because no routing resource can be shared by different nets in an FPGA, wire types should be assigned to net segments such that the number of net segments assigned to a wire type should not exceed the capacity of that wire type. To make the route feasible, connections between wire segments along the route should be available. Hence we can associate capacity to each node and edge of the wire type connection graph $G_w(V_w, E_w)$. Because all the wire segments (or connections) of a type have the same delay, we can associate delay cost to each node and edge of G_w with the delay value of corresponding routing resource. Note that we assume that all the connection switches are buffered switches, which holds true for actual Virtex-II FPGAs.

Before we define the problems of this paper, we first define the following notations for a given channel with m CLBs.

- $S_s = \{s_1, s_2, \dots, s_m\}$, where s_i is a CLB source node of the i th CLB.
- $S_t = \{t_1, t_2, \dots, t_m\}$, where t_i is a CLB sink node of the i th CLB.
- N_i denotes a set of all net segments from the i th CLB.
- N_{ij} denotes a set of all net segments whose spanning interval is (i, j) .
- I_i denotes a set of spanning intervals of all the net segments in N_i .
- $r_w(e)$ denotes the routing resource capacity of edge e . It corresponds to the number of available switches connecting two wire types.
- $r_w(v)$ denotes the routing resource capacity of node v . It corresponds to the number of wire segments of the type modeled by v .
- $c_w(e)$ denotes a nonnegative integer value delay cost of edge e , which is obtained by scaling actual delay at the switch modeled by e to an integer.
- $c_w(v)$ denotes a nonnegative integer value delay cost of node v , which is obtained by scaling actual delay at the wire segment modeled by v to an integer.

We define the problems of this paper as follows:

The Wire Type Assignment for One Source CLB (WTAO) Problem: Given a wire type connection graph $G_w(V_w, E_w)$ with capacity costs and delay costs associated with the nodes and edges, a source CLB node s_{i_s} , a set of net segments N_{i_s} , and a set of spanning interval I_{i_s} , find a path from s_{i_s} to the end point of a spanning interval of every net segment in N_{i_s} such that each edge and node is used no more than its capacity while the total delay cost of all net segments is minimized.

By solving the WTAO problem for all the CLBs in a channel, we can solve the following problem:

The Wire Type Assignment (WTA) Problem: Given a wire type connection graph $G_w(V_w, E_w)$ for a channel, a set of all net segments in a channel, and a set of spanning intervals of all net segments, find a set of paths connecting

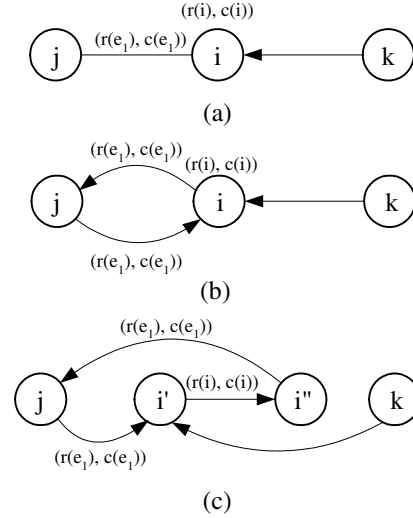


Figure 5: Network transformations: (a) **Original network.** Note that the edge between node i and node j (e_1) is an undirected edge. (b) **Transformation to directed edges.** (c) **Node splitting.** Node i is split to i' and i'' . A capacity and a cost are assigned to the edge (i', i'') . All the incoming edges to node i are connected to i' , and an outgoing edge is replaced by an edge going out of node i'' . (Node splitting for node j and node k are omitted.)

two end points CLBs of spanning interval of every net segment in the channel such that each edge and node is used no more than its capacity while the total delay cost of all net segments is minimized.

4. ALGORITHM DESCRIPTION

In this section, we describe the algorithms to solve the problems introduced in the previous section. By performing the min-cost max flow computations on the flow network which is constructed from G_w , our algorithm for the WTAO problem, WTAO_NF, can solve the WTAO problem exactly in polynomial time. We also solve the WTA problem by applying WTAO_NF iteratively on each CLB in a channel. To alleviate the influence of the order of the source CLB selection, we adopt a randomized iteration scheme.

4.1 The Wire type assignment for one source CLB (WTAO)

Given WTAO problem, we construct the flow network $G(V, E)$ as follows:

1. $V = V_w \cup \{s, t\}$, where s is a source node, and t is a sink node of $G(V, E)$.
2. $E = E_w \cup E_s \cup E_t$, where $E_s = \{(s, s_i) | s_i \in S_s\}$, $E_t = \{(t_i, t) | t_i \in S_t\}$
3. Edge Capacity:
for edges $e(s, s_i)$,

$$r(e) = \begin{cases} |N_i|, & \text{if } i = i_s \text{ (index of a source CLB)} \\ 0, & \text{otherwise} \end{cases}$$

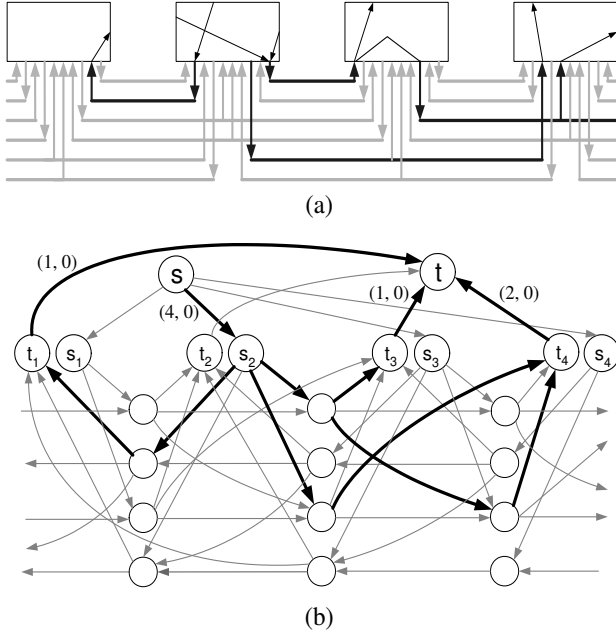


Figure 6: A flow corresponds to wire typed routes for 4 net segments from CLB2 whose spanning intervals are (2, 1), (2, 3), (2, 4), and (2, 4), respectively. (a) Wire type assigned routes in a channel. (b) A constructed flow network and a flow corresponding to the routes in (a).

for edges $e(t_j, t)$, $r(e) = |N_{i_s j}|$
for other edges e , $r(e) = r_w(e)$

- Node Capacity:
 $r(v) = r_w(v) \forall v \in V_w$, node s and node t are incapacitated.

- Edge Cost:

$$c(e) = \begin{cases} c_w(e), & \text{if } e \in E_w \\ 0, & \text{if } e \in E_s \cup E_t \end{cases}$$

- Node Cost:

$$c(v) = \begin{cases} c_w(v), & \text{if } v \in V_w \\ 0, & \text{if } v \in s, t \end{cases}$$

The constructed flow network for the WTAO problem is illustrated in Figure 6. To make our problem conform to the classical network flow framework, we transformed $G(V, E)$ to a directed graph in which only edges have capacities and costs. Note that any undirected edges, which can be formed due to bidirectional wire segments, in $G_w(V_w, E_w)$ can be transformed to a pair of directed edges with the cost and the capacity of the original undirected edge [2]. By node splitting transformation, any node i with nonzero cost and capacity is transformed into the two nodes i' and i'' . This transformation replaces each of the original edges (j, i) and (i, k) into (j, i') and (i'', k) , respectively. It also adds an edge (i', i'') with the cost and the capacity of node i . Figure 5 shows an example of the network transformations.

It can be shown that any flow in G is a wire type assignment solution for a subset of the given net segments.

Algorithm WTAO_NF
Input: $G_w(V_w, E_w)$, r , c , i_s , N_{i_s}
Output: A wire type assigned route for all net segments in N_{i_s}
begin
1. Construct the flow network $G(V, E)$
2. Assign costs and capacities
3. Run min-cost max-flow algorithm on $G(V, E)$
4. Derive the corresponding wire type assigned routes from the computed flow
end

Figure 7: Algorithm WTAO_NF

Each flow from s to t through s_{i_s} and t_j corresponds to a wire type assigned route for a net segment in $N_{i_s j}$. The occupied capacity of a node in V_w is the number of used wire segments of that type, and the flow amount along an edge in E_w is the number of used connection switches. If a flow f exists and $|f| = |N_{i_s}|$, then we can find a feasible solution for all the net segments in N_{i_s} , and the cost of the flow is the cost of a solution to the WTAO problem. Since we assigned $|N_{i_s j}|$ to each edge $(t_j, t) \in E_t$, the total capacities of edges connected to the sink node t are $\sum_{j=1}^m r(t_j, t) = \sum_{j=1}^m |N_{i_s j}| = |N_{i_s}|$. Therefore $|f^*| \leq |N_{i_s}|$, where f^* is the maximum flow in G . If $f^* < |N_{i_s}|$, there is no feasible solution to the WTAO problem, and the min-cost maximum flow assigns wire types to the routes for as many net segments in N_{i_s} as possible with minimum total delay costs. The following theorem shows that the wire type assignment for one source CLB problem can be exactly solved by a network flow computation on G .

THEOREM 1. A min-cost maximum flow f^* in G corresponds to a solution to WTAO problem with minimum total delay cost. If the size of f^* is $|N_{i_s}|$, then the WTAO problem is feasible so that all the net segments in N_{i_s} can be routed.

From a flow in G , a solution to the WTAO problem can be derived by a depth-first search from each CLB sink node to the source node in G . Figure 6 shows a flow f corresponding to a WTAO solution for 4 net segments in N_2 . Figure 7 summarizes our WTAO_NF algorithm.

There are several polynomial-time optimal algorithms available for finding a min-cost maximum flow in a network[2]. Deriving a solution to the WTAO problem from a flow can be done in $O(E)$ time. Thus, the WTAO problem can be solved efficiently as stated in the following theorem, if we adopt the double scaling algorithm[1].

THEOREM 2. The WTAO_NF algorithm exactly solves the WTAO problem in $O(VE \log R_{max} \log(VC_{max}))$ time, where R_{max} is the maximum value of the capacities and C_{max} is the maximum value of the costs.

The time complexity of our algorithm is mainly dependent on the number of nodes and edges in $G(V, E)$. Because we abstracted all the wire segments of the same type as one node and all the connections between wire segments between a pair of types as an edge, the number of actual wire segments only affects R_{max} term in the time complexity. Therefore the runtime of our algorithm does not grow seriously with the increased number of wire segments.

circuit	CLBs /channel	# of nets	runtime (s)		routed nets		total delay (ns)		
			net-net	wta_nf	net-net	wta_nf	net-net (per net)	wta_nf (per net)	improve
C1	16	774	3.95	3.00	727.6	774	30.72 (0.042)	25.54 (0.033)	21.4%
C2	32	1549	12.05	9.36	1478.4	1549	80.3 (0.054)	55.8 (0.036)	33.3%
C3	48	2349	19.42	19.27	2236	2349	150.7 (0.067)	106.4 (0.045)	32.8%
C4	64	2900	39.12	31.58	2826	2900	226.8 (0.080)	123.6 (0.043)	47.5%
C5	96	4269	62.82	65.57	4131	4269	442.5 (0.11)	292.5 (0.069)	37.3%
C6	128	5134	130.81	113.37	5070.2	5134	707.4 (0.14)	437.6 (0.085)	39.3%
C7	256	8070	416.58	417.76	8053.5	8070	2262.3 (0.28)	1779.4 (0.22)	21.4%

Table 1: Comparisons between a net-by-net approach and WTA_NF. These are average results from 5 runs.

Algorithm WTA_NF
Input: G_w, R, C, N, M (number of iterations)
Output: A wire type assigned route for all net segments in N
begin
1. Construct the flow network $G(V, E)$
2. **for** $i = 1$ to M
3. Generate a random order D on source CLB
4. **for each** d_i in D
5. Assign costs and capacities
6. Run min-cost max-flow algorithm on $G(V, E)$
7. Derive the corresponding wire type assigned routes from the computed flow
8. Adjust capacities
end

Figure 8: Algorithm WTA_NF

4.2 The wire type assignment for a channel (WTA)

In this section we solve the wire type assignment problem for a channel (WTA). We apply the WTAO_NF algorithm iteratively on all the CLBs in a channel to solve this problem.

Given a wire type connection graph $G_w(V_w, E_w)$, a source CLB is selected and WTAO_NF is applied to solve the WTAO problem. The flow network G is generated only once during the whole procedure. After getting solution for the WTAO problem, the capacity of each node and edge along the obtained routes is adjusted by subtracting the size of the flow from the original capacity. Then, a new CLB is selected as a source CLB and the WTAO_NF algorithm is applied after updating the capacities of edges in E_s and E_t according to the interval information of the net segments from the new source CLB. This procedure is repeated until all the CLBs in a channel are selected as source CLBs. Note that there is no change in the nodeset and edgeset of G for each step, and only the capacities of some edges are updated.

Because the availability of the routing resources are updated over iterations, the ordering of the source CLB selection can influence the final result. To reduce the effect of this ordering, we randomly select the source CLB in every step. To enhance the result, we iterate the whole procedure several times. Due to the optimality of WTAO_NF, the result obtained by the current iteration is guaranteed to be no worse than the result from previous iterations.

The optimality of WTAO_NF can be very helpful for incremental improvement of the routing result, especially when

WTAO_NF is used interactively with the global router or the detailed router. Suppose there are nets that violate some timing or congestion constraints. After ripping up those nets and rerouting them using global router, WTA_NF can be applied to resolve the violations. Figure 8 summarizes our algorithm for the wire type assignment problem for a channel, WTA_NF.

5. EXPERIMENTAL RESULTS

We have implemented our algorithms in the C++ programming language on a SUN Sparc Ultra 5 (360MHz) with 128M memory. The experiments were performed on 7 randomly generated global routing results. They were routed on 7 different sizes of channels. The size of a channel in the smaller 6 examples are the same as that of some of FPGAs in the Virtex-II family. We added an example with 256 CLBs in a channel, and it is twice as big as the largest Virtex-II FPGA. We assumed that the number of wire segments per CLB is the same for all the FPGAs, which holds true for actual Virtex-II FPGAs. We used the same number and types of wire segments and connection switches as those in Virtex-II FPGAs.

Because there is no standalone wire type assignment algorithm available, we compared our algorithm with a net-by-net approach in which each net is randomly selected and routed along the min-cost path. We compared runtime, number of routed (and wire type assigned) nets, and the sum of the delays of all the net segments of a channel. For each test circuit, we ran the algorithms 5 times. The iteration number used in WTA_NF was 4 for each run. The average results are shown in Table 1. In all the cases, WTA_NF successfully routed and assigned wire types to all the net segments. The total delay of net segments was improved significantly. Because the number of routed nets is different between the two approaches, the delay cost per net is also listed and the improvement is calculated. We could get up to 47.5% (average 33.3%) improvement on the delay cost per net. The runtime of WTA_NF listed in Table 1 is the total runtime for 4 iterations.

6. CONCLUSIONS

In this paper, we have proposed the wire type assignment as a separate stage between global routing and detailed routing, and presented a randomized polynomial-time algorithm for wire type assignment of all net segments in a channel. By routing all the net segments from a CLB at the same time, our algorithm can greatly alleviate the net ordering problem that can be observed in net-by-net approaches. Furthermore, the total delay of the net segments is also mini-

mized, which can contribute to meeting the overall timing constraints of the circuit. Although our algorithm is intended for the net segments in a channel, it can be used for assigning wire types to all the nets in an FPGA by handling all the channels. We compared our algorithm with a net-by-net algorithm and the experimental results shows that our algorithm is very effective.

7. REFERENCES

- [1] R. K. Ahuja, A. V. Goldberg, J. B. Orlin, and R. E. Tarjan, "Finding minimum-cost flows by double scaling," *Mathematical Programming* 53, pp. 243-266, 1992.
- [2] R. K. Ahuja, T. L. Magnanti and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.
- [3] Altera Cooperation, *ApexII Programmable Logic Device Family Data Sheet*, 2002.
- [4] S. Brown, M. Khellah, G. Lemieux, "Segmented Routing for Speed-Performance and Routability in Field-Programmable Gate Arrays," in *Journal of VLSI Design*, 1996.
- [5] Yao-Wen Chang, D. F. Wong, Kai Zhu, and C. K. Wong, "On a New Timing-Driven Tree Problem," in *Proc. Intl. Conf. on Computer-Aided Design*, 1994, pp. 380-385.
- [6] Yao-Wen Chang, D. F. Wong, C. K. Wong, "FPGA Global Routing Based on a New Congestion Metric," in *Proc. Intl. Conf. on Circuit Design*, 1995, pp. 372-378.
- [7] T. H. Cormen, C. E. Leiserson, R. L. Rives, C. Stein, *Introduction to Algorithms, 2nd ed.*, McGraw-Hill, 2001.
- [8] Rajeev Jayaraman, "Physical Design for FPGAs," *Proc. International Symposium on Physical Design*, 2001, pp. 214-221.
- [9] Y.-S. Lee, C.-H. Wu, "A performance and routability driven router for FPGA's considering path delay," in *Proc. Design Automation Conference*, 1995, pp. 557-561.
- [10] Seokjin Lee, D. F. Wong, "Timing-Driven Routing for FPGAs Based on Lagrangian Relaxation," in *Proc. International Symposium on Physical Design*, 2002, pp. 176-181.
- [11] G. G. F. Lemieux, S. D. Brown, D. Vranesic, "On Two-Step Routing For FPGAs," in *Proc. International Symposium on Physical Design*, 1997, pp. 60-66.
- [12] L. McMurchie, C. Eberling, "PathFinder: A Negotiation-Based Performance-Driven Router for FPGAs," in *Proc. ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, 1995, pp. 111-117.
- [13] V. Betz, J. Rose, "VPR: A New Packing, Placement and Routing Tool for FPGA Research," in *Proc. the 7th Annual Workshop on Field Programmable Logic and Applications*, 1999, pp. 213-222.
- [14] Xilinx Inc., *Virtex-II FPGA Advance Product Specification*, 2001
- [15] K. Zhu, Y.-W. Chang, D. F. Wong, "Timing-Driven Routing for Symmetrical-Array-Based FPGAs," in *Proc. Intl. Conf. on Computer Design*, 1998, pp. 628-633.