Wotan: A Tool for Rapid Evaluation of FPGA Architecture Routability Without Benchmarks

Oleg Petelin and Vaughn Betz
Department of Electrical and Computer Engineering
University of Toronto, Toronto, Canada
{opetelin, vaughn}@eecg.toronto.edu

Abstract—**FPGA routing architectures consist of routing wires and programmable switches which together account for a significant portion of the fabric delay and area. Routing architectures have traditionally been evaluated using a full CAD flow with a suite of benchmark circuits. While the results of such a flow can be accurate, CAD tools are often tuned to a specific architecture type and can take a long time to run which prohibits quick exploration of different architectures early in the design process. In this paper we present an alternative approach that quickly estimates routability for a wide range of architectures without the use of benchmark circuits. Our new routability predictor first assigns congestion probabilities to the architecture's routing resources based on demand estimates found via efficient path enumeration through the routing graph. Next, we compute the probabilities of successfully routing different source/sink connections and finally we combine them to assign an overall routability score. We describe our predictor and present routability estimates for a range of 6-LUT and 4-LUT architectures, showing reasonable agreement with routability results from the full VPR CAD flow in much less CPU time.**

I. Introduction

The routing architecture of a Field-Programmable Gate Array (FPGA) consists of a set of wires and programmable switches that define the interconnect between reconfigurable function blocks. Accounting for a large fraction of total chip area and critical path delay [1], a good routing architecture is a crucial component of FPGA design. In this paper we focus on the routability of an FPGA architecture; a more routable architecture is able to implement circuits with more complex connectivity requirements.

Traditionally routing architectures have been evaluated using academic tools such as VPR [2] or commercial tools such as Altera's FMT [1]. These tools are part of a full CAD flow that synthesizes, packs, places and routes a set of benchmark circuits and evaluates the area and delay of each circuit to asses the overall architecture quality. While quite accurate, the traditional CAD flow has three main challenges for exploring routing architectures early in a design process:

- *Slow speed.* Multiple benchmark circuits must be used to obtain statistically valid results; running the CAD flow over each benchmark circuit can take a long time.
- *Limited insight.* A full CAD flow can accurately estimate the area, delay and routability of an architecture, but these results give limited insight into *why* one routing architecture performs better than another.
- *Tuned to an architecture.* Traditional CAD tool flows have been targeted at a specific architecture type. Evaluating a different type of architecture with the same CAD tool may not accurately represent an architecture's potential. As an example, VPR assumes that placing netlist blocks closer together according to Manhattan distance is better for the router, but if hard obstacles in the FPGA required that they be routed around, then a placement according to Manhattan distance could increase the difficulty of the routing problem.

We propose a routability predictor that we call Wotan which seeks to quickly evaluate a wide range of routing architectures without the use of benchmark circuits. Such a predictor would be useful early in the architecture design process where quick design iteration is necessary in an architecture space to which traditional CAD tools may not be well-tuned. As described in later sections Wotan evaluates routing architectures in three steps:

1) Probabilities of congestion (demands) are assigned to logic block pins and routing wires based on path enumeration between different sources and sinks.
2) Probabilities of successfully routing different sources to different sinks are computed based on the routing resource probabilities assigned in step 1.
3) An overall routability metric is assigned based on the connection probabilities computed in step 2.

In addition to the routability predictor our contribution includes a method to efficiently traverse and enumerate paths in a cyclic graph; this is used to set node demands and analyze routing probabilities (exponential in complexity if done naively).

The rest of this paper is organized as follows. Section II reviews background and related work. Section III gives an overview of our predictor method. Section IV presents predictor results and compares them to routability results from the full VPR CAD flow. Section V concludes.

II. Background and Related Work

This section briefly reviews literature related to early-stage architecture evaluation and network reliability. For a comprehensive overview of the different routing architecture elements please refer to [3].

A. Early-Stage Routing Architecture Evaluation

Early-stage architecture evaluation refers to techniques for quickly exploring architecture choices or for exploring entirely new architecture types to which current CAD tools are not well-suited.

In [4] Rose and Brown introduced the concept of 'flexibility' for the connection block ($F_{c,in}$ & $F_{c,out}$) and switch block (F_s) which specifies the number of programmable switches to be used without considering the precise switch patterns. Later, the differences between the planar/subset, universal [5] and Wilton [6] switch blocks would highlight that the switch patterns of routing elements play a significant role in routability. Analytical models like [7] [8] quantify the effect of coarse parameters like F_c and F_s on architecture routability but do not account for different wire segment lengths or the precise switch patterns of the routing elements.

In [9] and [10] Sharma et al proposed the 'Independence' placement and routing algorithms which avoid making assumptions about the underlying architecture but at the cost of very high runtime.

In [3] Lemieux and Lewis proposed a method of evaluating the goodness of sparse crossbars by using network flow (over many random test vectors) to measure a crossbar's worst-case capacity. Since a cascade of FPGA routing elements form a compound crossbar it is tempting to extend this work to evaluate an FPGA routing architecture as a whole. However, network flow cannot account for routability differences between short and long connections, and has no view of typical congestion patterns in the routing architecture.

The shortfalls of the routability evaluation methods described above are that they don't account for wire segment length or switch pattern topology [4] [7] [8], take a very long time to run [9] [10], or are limited in scope to a single component of the routing architecture [3]. In contrast Wotan seeks to quickly evaluate the routing fabric as a whole over a wide range of architectures.

B. Network Reliability

Network reliability investigates methods of evaluating connections between nodes in probabilistic graphs $G(V, E)$ where nodes V and/or edges E have independent probabilities of failure (e.g. due to congestion). In [11] Valiant showed that all measures of network reliability belong to the class of counting problems *#P-complete* and are more difficult than problems in *NP-complete*. For example calculating the probability of connecting a source node s to $T(s)$ other nodes in an arbitrary graph requires knowledge of all shortest paths which connect s to $T(s)$; but such shortest paths are Steiner trees and finding any Steiner tree is *NP-complete*.

Despite the computational complexity, network reliability offers interesting insights [12]. Consider Figure 1 – we want to evaluate the probability of successfully connecting the top and bottom nodes of graphs $G1$ and $G2$ where intermediate nodes have a probability p of being operational. Plotting the probability of successfully connecting the nodes versus the value of p shows that $G2$ is more reliable in low-stress conditions (when node operational probability p is high) while $G1$ is more reliable at high-stress conditions (when p is low) [1]. Comparisons of routing networks must therefore be qualified by their level of stress; one network is not necessarily better than another at all loading levels. The routability metric of our predictor is based on the level of stress required for a network to achieve a certain value of reliability.

[1]Intuitively at high-stress the probability of long paths being operational is extremely low so the effect of *short* paths dominates; at low-stress points both short and long paths have a high probability of operation so the overall number of paths connecting the nodes (short or long) is more important.

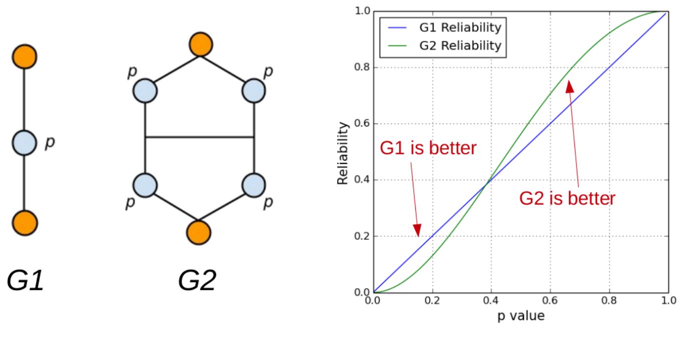


Fig. 1: Plot of probability of connecting the top and bottom nodes in graphs $G1$ and $G2$. Intermediate nodes have probability p of being operational.

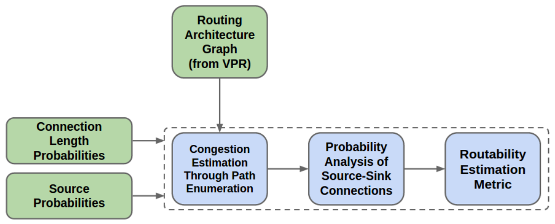

Routing Architecture Graph (from VPR)
Connection Length Probabilities
Source Probabilities
Congestion Estimation Through Path Enumeration
Probability Analysis of Source-Sink Connections
Routability Estimation Metric

Fig. 2: Wotan flow. Routing graph read-in from VPR.

III. Routability Predictor – Overview

This section describes the high-level features of the Wotan tool. Due to space constraints we cannot describe the predictor implementation fully, but provide implementation details and examples at *www.eecg.utoronto.ca/~opetelin/wotan*.

Wotan operates on a directed routing resource graph [13] $G(V, E)$ with a set of vertices V and edges E. Each node $v \in V$ receives a cost $c(v)$ which we take to represent the wirelength of the corresponding routing resource (but could also represent node delay or something else). Sources and sinks are represented by the set of nodes $S \subseteq V$ and $T \subseteq V$ respectively; in the VPR routing graph sources and sinks represent inputs and outputs of primitives such as flip-flops or LUTs. A directed path through the graph can be written as an ordered set of nodes $(v_0, v_1, ...)$ provided that there are edges which connect each pair of consecutive nodes. Since nodes have cost, the cost of a path, $c(v_0, v_1, ...)$ is defined as the sum of the constituent node costs. The distance betweeen two nodes, $d(v_i, v_j)$, is defined as the cost of the minimum-cost path that connects v_i to v_j. Lastly, demands (congestion probabilities) are placed on nodes; a demand $De(v)$ indicates the probability for node v to be congested so that it is unavailable for routing.

Wotan analyzes routability based on pairwise connections between source nodes $s \in S$ and a set of sink nodes associated with each source, $T(s) \subseteq T$. The predictor steps are summarized below:

1) *Estimate congestion:* Node demands $De(v)$ are estimated by efficiently enumerating paths from all sources s to each of the corresponding sinks in

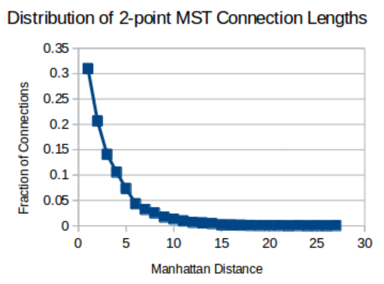


Fig. 3: Probability distribution of 2-point MST connection lengths for placed CLMA benchmark.

$T(s)$. The demand due to a path between (s,t) is added to all nodes which the path traverses. For each source/sink pair (s,t) the maximum allowed path cost is a linear function of the distance between s and t, $c_b(d(s,t))$.

2) *Analyze source/sink pair routability:* For each source/sink pair, (s,t) $t \in T(s)$, efficiently *estimate* the probability of routing s to t. As in the previous step, only paths of cost $\leq c_b(d(s,t))$ are considered.
3) *Compute routability metric:* The routability metric is a weighted sum of the individual routing probabilities from step 2.

Wotan also accepts two user parameters which account for some of the behaviour of real netlists and CAD tools by applying appropriate weights in the steps above.

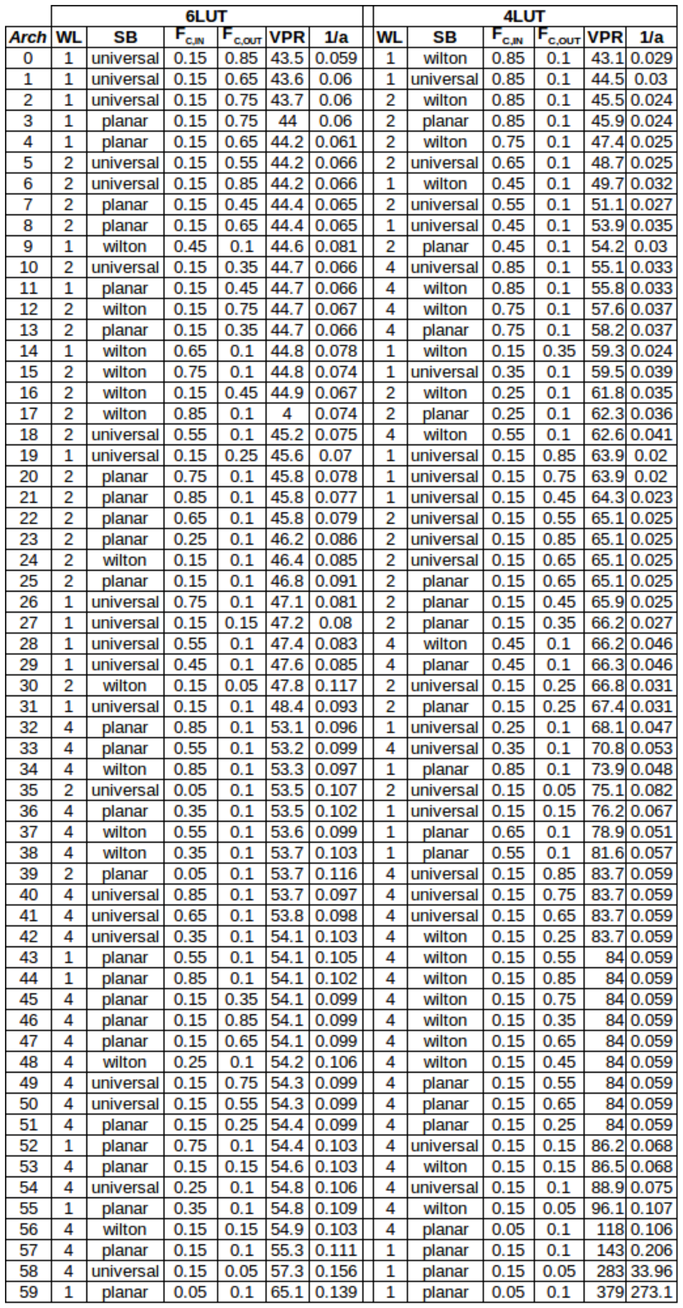
- The connection probability distribution $P(l)$ is intended to capture the tendency of a placer to make certain connection lengths more likely. For an island-style architecture connection length is defined as the Manhattan distance between the source and the sink.
- The probability of using different sources, $P(s)$. This passed-in parameter is meant to reflect behaviour where certain source nodes are more likely to be used than others.

IV. Experimental Results and Discussion

We used Wotan to estimate routability for 60 6-input-LUT architectures and 60 4-input-LUT architectures as shown in Figure 4 and compared these estimates to routability results from the full VPR CAD flow. Both 6LUT and 4LUT architectures vary the length of unidirectional [14] wire segments (1, 2 or 4), connection block flexibilities $F_{c,in}$ and $F_{c,out}$, as well as switch block topology (planar, universal or Wilton).

The logic blocks of the 6LUT architectures are based on Altera's Stratix V architecture [1] with 10 6LUTs per logic block, 40 equivalent input pins (connected through a full crossbar) and 20 output pins. The 4LUT architectures had logic blocks based on Lattice Semiconductor's iCE40 architecture [15] with 8 4LUTs per logic block, 32 input pins and 8 output pins; although the logic blocks in the 4LUT architecture did not have a full crossbar on the input pins, the four pins of each 4LUT are still logically equivalent amongst themselves.

As in Figure 2, VPR was used to generate a routing architecture to be read-in by the predictor. This routing architecture represented a 10-by-10 grid of logic blocks with the precise routing fabric defined by wire segment length, switch block topology, $F_{c,in}$ and $F_{c,out}$. While Wotan evaluated routability



	6LUT						4LUT					
Arch	WL	SB	$F_{C,IN}$	$F_{C,OUT}$	VPR	1/a	WL	SB	$F_{C,IN}$	$F_{C,OUT}$	VPR	1/a
0	1	universal	0.15	0.85	43.5	0.059	1	wilton	0.85	0.1	43.1	0.029
1	1	universal	0.15	0.65	43.6	0.06	1	universal	0.85	0.1	44.5	0.03
2	1	universal	0.15	0.75	43.7	0.06	2	wilton	0.85	0.1	45.5	0.024
3	1	planar	0.15	0.75	44	0.06	2	planar	0.85	0.1	45.9	0.024
4	1	planar	0.15	0.65	44.2	0.061	2	wilton	0.75	0.1	47.4	0.025
5	2	universal	0.15	0.55	44.2	0.066	2	universal	0.65	0.1	48.7	0.025
6	2	universal	0.15	0.85	44.2	0.066	1	wilton	0.45	0.1	49.7	0.032
7	2	planar	0.15	0.45	44.4	0.065	2	universal	0.55	0.1	51.1	0.027
8	2	planar	0.15	0.65	44.4	0.065	1	universal	0.45	0.1	53.9	0.035
9	1	wilton	0.45	0.1	44.6	0.081	2	planar	0.45	0.1	54.2	0.03
10	2	universal	0.15	0.35	44.7	0.066	4	universal	0.85	0.1	55.1	0.033
11	1	planar	0.15	0.45	44.7	0.066	4	wilton	0.85	0.1	55.8	0.033
12	2	wilton	0.15	0.75	44.7	0.067	4	wilton	0.75	0.1	57.6	0.037
13	2	planar	0.15	0.35	44.7	0.066	4	planar	0.75	0.1	58.2	0.037
14	1	wilton	0.65	0.1	44.8	0.078	1	wilton	0.15	0.35	59.3	0.024
15	2	wilton	0.75	0.1	44.8	0.074	1	universal	0.35	0.1	59.5	0.039
16	2	wilton	0.15	0.45	44.9	0.067	2	wilton	0.25	0.1	61.8	0.035
17	2	wilton	0.85	0.1	4	0.074	2	planar	0.25	0.1	62.3	0.036
18	2	universal	0.55	0.1	45.2	0.075	4	wilton	0.55	0.1	62.6	0.041
19	1	universal	0.15	0.25	45.6	0.07	1	universal	0.15	0.85	63.9	0.02
20	2	planar	0.75	0.1	45.8	0.078	1	universal	0.15	0.75	63.9	0.02
21	2	planar	0.85	0.1	45.8	0.077	1	universal	0.15	0.45	64.3	0.023
22	2	planar	0.65	0.1	45.8	0.079	2	universal	0.15	0.55	65.1	0.025
23	2	planar	0.25	0.1	46.2	0.086	2	universal	0.15	0.85	65.1	0.025
24	2	wilton	0.15	0.1	46.4	0.085	2	universal	0.15	0.65	65.1	0.025
25	2	planar	0.15	0.1	46.8	0.091	2	planar	0.15	0.65	65.1	0.025
26	1	universal	0.75	0.1	47.1	0.081	2	planar	0.15	0.45	65.9	0.025
27	1	universal	0.15	0.15	47.2	0.08	2	planar	0.15	0.35	66.2	0.027
28	1	universal	0.55	0.1	47.4	0.083	4	wilton	0.45	0.1	66.2	0.046
29	1	universal	0.45	0.1	47.6	0.085	4	planar	0.45	0.1	66.3	0.046
30	2	wilton	0.15	0.05	47.8	0.117	2	universal	0.15	0.25	66.8	0.031
31	1	universal	0.15	0.1	48.4	0.093	2	planar	0.15	0.25	67.4	0.031
32	4	planar	0.85	0.1	53.1	0.096	1	universal	0.25	0.1	68.1	0.047
33	4	planar	0.55	0.1	53.2	0.099	4	universal	0.35	0.1	70.8	0.053
34	4	wilton	0.85	0.1	53.3	0.097	1	planar	0.85	0.1	73.9	0.048
35	2	universal	0.05	0.1	53.5	0.107	2	universal	0.15	0.05	75.1	0.082
36	4	planar	0.35	0.1	53.5	0.102	1	universal	0.15	0.15	76.2	0.067
37	4	wilton	0.55	0.1	53.6	0.099	1	planar	0.65	0.1	78.9	0.051
38	4	wilton	0.35	0.1	53.7	0.103	1	planar	0.55	0.1	81.6	0.057
39	2	planar	0.05	0.1	53.7	0.116	4	universal	0.15	0.85	83.7	0.059
40	4	universal	0.85	0.1	53.7	0.097	4	universal	0.15	0.75	83.7	0.059
41	4	universal	0.65	0.1	53.8	0.098	4	universal	0.15	0.65	83.7	0.059
42	4	universal	0.35	0.1	54.1	0.103	4	wilton	0.15	0.25	83.7	0.059
43	1	planar	0.55	0.1	54.1	0.105	4	wilton	0.15	0.55	84	0.059
44	1	planar	0.85	0.1	54.1	0.102	4	wilton	0.15	0.85	84	0.059
45	4	planar	0.15	0.35	54.1	0.099	4	wilton	0.15	0.75	84	0.059
46	4	planar	0.15	0.85	54.1	0.099	4	wilton	0.15	0.35	84	0.059
47	4	planar	0.15	0.65	54.1	0.099	4	wilton	0.15	0.65	84	0.059
48	4	wilton	0.25	0.1	54.2	0.106	4	wilton	0.15	0.45	84	0.059
49	4	universal	0.15	0.75	54.3	0.099	4	planar	0.15	0.55	84	0.059
50	4	universal	0.15	0.55	54.3	0.099	4	planar	0.15	0.65	84	0.059
51	4	planar	0.15	0.25	54.4	0.099	4	planar	0.15	0.25	84	0.059
52	1	planar	0.75	0.1	54.4	0.103	4	universal	0.15	0.15	86.2	0.068
53	4	planar	0.15	0.15	54.6	0.103	4	wilton	0.15	0.15	86.5	0.068
54	4	universal	0.25	0.1	54.8	0.106	4	universal	0.15	0.1	88.9	0.075
55	1	planar	0.35	0.1	54.8	0.109	4	wilton	0.15	0.05	96.1	0.107
56	4	wilton	0.15	0.15	54.9	0.103	4	planar	0.05	0.1	118	0.106
57	4	planar	0.15	0.1	55.3	0.111	1	planar	0.15	0.1	143	0.206
58	4	universal	0.15	0.05	57.3	0.156	1	planar	0.15	0.05	283	33.96
59	1	planar	0.05	0.1	65.1	0.139	1	planar	0.05	0.1	379	273.1

Fig. 4: Listing of architecture points evaluated for 6LUT and 4LUT architectures. From left to right the columns represent: wire segment length, switch block topology, $F_{c,in}$, $F_{c,out}$, minimum routable channel width from VPR, inverse of predictor routability metric α.

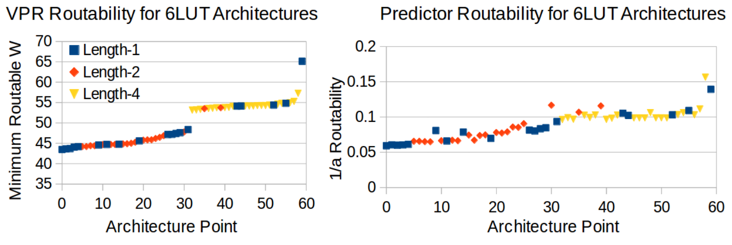


Fig. 5: Routability results for 6-input-LUT architectures.

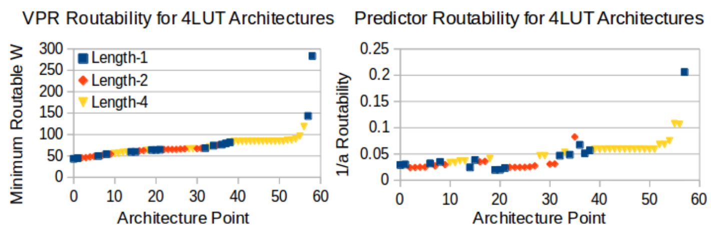


Fig. 6: Routability results for 4-input-LUT architectures

Correlation (6LUT)	0.90
Correlation (4LUT)	0.83
Pairwise Comparisons in Agreement (6LUT)	1582/1770 (0.89)
Pairwise Comparisons in Agreement (4LUT)	1505/1770 (0.85)
VPR Runtime / Architecture Point	10min
Predictor Runtime / Architecture Point	25s

Fig. 7: Level of agreement between VPR and Wotan routability estimates for 6-input LUT and 4-input LUT architectures.

only between logic blocks, VPR routability results also account for signal connections to I/O pads and hard blocks like memory. We therefore set the connection block flexibilities of non-logic-block components to ‘1.0’ to increase the fidelity of comparison.

Wotan evaluated the routability of each architecture over channel widths of 50, 70 and 90; the geometric mean over the channel widths defined the final routability score α. While source probabilities were set to $P(s) = 1$, the connection length probability distribution $P(l)$ was set based on a benchmark circuit profiled in VPR. A placed circuit netlist was decomposed into a minimum spanning tree (MST) to get the length probability distributions as in Figure 3. Wotan then analyzed routability for connections of Manhattan distance ≤ 4 from the source with demands due to different paths weighted according to this distribution. VPR evaluated the routability of each architecture based on 20 MCNC benchmark circuits, evaluating the average minimum routable channel width for each circuit over 4 seeds.

Figures 5 and 6 compare VPR and Wotan routability estimates. The x-axis corresponds to the architecture points in Figure 4 and the architecture points are color-coded by wire segment length. To highlight some interesting predictions, Wotan correctly ranked a 6-LUT length-2 planar architecture to have slightly higher routability than a length-1 Wilton architecture (6LUT architectures 7 & 9) – an unintuitive result which happens because it is not possible to create a truly planar switch block with length-2 unidirectional wires. In the 4LUT case (Figure 6) Wotan correctly ranked the set of length-4 architecture points 39-51 as being equivalent in routability regardless of switch block pattern or connection block flexibility – another unintuitive result which occurs because the restriction of connecting to the drive points of length-4 unidirectional wires actually permutes the switch block connections beyond what a switch block prescribes. These results show the fidelity of our predictor method across wire segment lengths, switch block patterns, and connection block flexibilities. Figure 7 quantifies the level of agreement between Wotan and VPR to show that our predicted routability estimates are in good agreement with the full VPR CAD flow in significantly less CPU time.

On the other hand Wotan does not properly account for fanout effects which we believe to be a major source of misprediction. The predictor considers fanout nets *decomposed* into two-terminal connections (the $P(l)$ parameter) so that all path enumeration is done from logic block outputs to nearby logic block inputs; this does not properly account for high-fanout nets which can consume significant routing resources over a large area of the chip. Fanout effects can potentially be addressed by analyzing paths from virtual sources instantiated at routing wires reachable from logic block outputs – this would emulate fanout behaviour where a signal can travel a long distance before branching out.

V. Conclusion

We have presented an alternative method of evaluating the routability of an FPGA routing fabric. Whereas routability has traditionally been evaluated using a slow but accurate synthesize/pack/place/route CAD flow over multiple benchmarks, our predictor does not require benchmarks and shows reasonably good agreement with routability results from the VPR CAD tool in significantly less CPU time. Furthermore while the traditional CAD flows are often tuned to a specific kind of architecture, our routability evaluation methodology can be easily adapted to a variety of different architecture types. Lastly, it is possible to visualize routing node demands and individual path connection probabilities which can be used to gain insight into routing architectures; future work will explore this fully.

References

[1] D. Lewis *et al.*, “Architectural Enhancements in Stratix V,” in *FPGA*, 2013, pp. 147–156.

[2] J. Luu *et al.*, “VTR 7.0 : Next Generation Architecture and CAD System for FPGAs,” *ACM TRETS*, vol. 7, no. 2, 2014.

[3] G. Lemieux and D. Lewis, *Design of Interconnection Networks for Programmable Logic*. Springer New York, 2004.

[4] J. Rose and S. Brown, “Flexibility of Interconnection Structures for Field-Programmable Gate Arrays,” *JSSC*, pp. 277–282, 1991.

[5] Y.-W. Chang and D. F. Wong, “Universal Switch Modules for FPGA Design,” *ACM TODAES*, vol. 1, no. 1, pp. 80–101, 1996.

[6] S. J. E. Wilton, “Architectures and Algorithms for Field-Programmable Gate Arrays with Embedded Memory,” Ph.D. dissertation, University of Toronto, 1997.

[7] S. Brown, J. Rose, and Z. G. Vranesic, “A Stochastic Model to Predict the Routability of Field-Programmable Gate Arrays,” *IEEE TCAD*, vol. 12, no. 12, pp. 1827–1838, 1993.

[8] J. Das and S. Wilton, “An Analytical Model Relating FPGA Architecture Parameters to Routability,” in *FPGA*, 2011, pp. 181–184.

[9] A. Sharma, C. Ebeling, and S. Hauck, “Architecture Adaptive Routability-Driven Placement for FPGAs,” in *FPL*, 2005, pp. 427–432.

[10] A. Sharma and S. Hauck, “Accelerating FPGA Routing Using Architecture-Adaptive A* Techniques,” in *FPT*, 2005, pp. 225–232.

[11] L. Valiant, “The Complexity of Enumeration and Reliability Problems,” *SIAM J. Comput.*, vol. 8, no. 3, 1979.

[12] E. Moore and C. Shannon, “Reliable Circuits Using Less Reliable Relays,” *Journal of the Franklin Institute*, pp. 281–297, Oct. 1956.

[13] V. Betz, J. Rose, and A. Marquardt, *Architecture and CAD for Deep-Submicron FPGAs*. Kluwer Academic Publishers, 1999.

[14] G. Lemieux, E. Lee, M. Tom, and A. Yu, “Directional and Single-Driver Wires in FPGA Interconnect,” in *FPT*, 2004, pp. 41–48.

[15] Lattice Semiconductor, “iCE40 LP/HX Family Data Sheet.”