Interconnect Complexity-Aware FPGA Placement Using Rent's Rule

G. Parthasarathy M. Marek-Sadowska Arindam Mukherjee Amit Singh
Department of Electrical and Computer Engg.
University of California – Santa Barbara
California–93117, USA
{ gpartha@bigbend, mms@apex, arindam@guitar, asingh@guitar}.ece.ucsb.edu

ABSTRACT

Field Programmable Gate Arrays (FPGAs) have gained in commercial acceptance because they offer instant manufacturing turnaround and low costs. However, FPGAs are constantly hard pressed to keep up with the requirements of the more complex and larger scale circuits which are being targeted for them. Routability of a circuit depends on the FPGA architecture, the placement, and the interconnection complexity of the circuit to be placed and routed. This paper explores the use of Rent's rule as a complexity metric for improving the placement of circuits on a target FPGA architecture, such that routing resource utilization is improved. A new simulated annealing based placement algorithm is presented and experimental results are presented to illustrate the validity of the approach for certain example circuits and the ISCAS benchmarks.

Keywords

Rent's exponent, Placement, Interconnect

1. INTRODUCTION

This paper addresses the problem of how to derive a *good* placement such that the routing resources of a target FPGA are best utilized. We propose the use of a well known metric for interconnection complexity, i.e. Rent's exponent, for placement. We show how a well known placement algorithm i.e. simulated annealing as implemented in **VPR** [2], can be modified to use these metrics during the optimization process. This approach has been implemented in a package called **MVPR**, which is essentially **VPR** with a modified cost function.

The organization of this paper is as follows: Section 2 outlines the prior work in the problem. Section 3 describes the problem statement and our CAD flow, analysis of the problem, the theoretical background for the metrics used in this paper, and the methods used to calculate them. We discuss interconnect requirements in Section 4. Section 5 describes the new placement algorithm used in **MVPR**. Section 6 describes the experiments and the results that we have obtained. In section 7, we present the conclusions derived from our work.

SLIP'01, March 31-April 1, 2001, Sonoma, California, USA.

2. PRIOR WORK

Various authors have tried to capture interconnection requirements for routability of circuits. El Gamal [8] used a stochastic model to estimate routability for channeled gate arrays. This was extended by Chan et.al.[4] to FPGAs. Alexander [1] used approximate solutions to a sequence of Steiner Tree-on-a-graph problems for routing. Wu et al. [15], [16] showed that simplified variants of FPGA routing are reducible to the graph coloring problem and used this result to show the NP-hard nature of FPGA routing.

Wood et al.[14] estimated FPGA routability using boolean satisfiability with BDDs, extending work that Devadas et al. [6] had done for ASIC routing. However, the model is impractical for large circuits due to the BDD size problem. Exact methods for routability estimation are very hard to implement. Hence we look at approximate or empirical methods for routability estimation.

The best known empirical routability estimate is Rent's rule. First observed by Rent at IBM, and also derived by several others; e.g. Donath[7], Brown[3], the relation was first studied extensively in large circuits by Landman and Russo [11]. Following this, various methods have been proposed to estimate Rent's exponent based on partitioning algorithms [9] and interconnect behavior e.g. [12]. We calculate Rent's exponent using both empirical analysis and analytical methods. In sections 3 and 5, we show how Rent's exponent is used for achieving a good placement.

2.1 Rent's Rule Overview

E.F. Rent of IBM published two internal memoranda in 1960 that contained the log plots of number of pins versus number of circuits in a logic design, which tends to form a straight line in a log-log plot, and yield the relationship:

$$N_{io} = K.{N_g}^p \quad (1)$$

Here, N_{io} is the number of pins or the number of external signal connections to a logic block, N_g is the number of logic gates in the block, p is the *Rent's exponent*, and K is a proportionality constant, which is the average number of interconnections per block.

We can further define the Rent's exponent for a given architecture, p_a, as the exponent of a characteristic function that

captures the interconnection resource growth of an FPGA *a' la* [5]. While this is, strictly speaking, not the same as the Rent's function for a design, it serves to illustrate the idea. We empirically determine p_a by using the Rent's exponent of a design with uniform interconnect growth, which best utilizes the routing resources of the architecture.

Van Marck et al. [12] proposed a simple technique for estimating the *local* Rent's exponent for a design. The *local* Rent's exponent is defined as the slope of the log-log plot of *number of interconnections from a block* versus *length of interconnection*. This technique fits in very neatly with the design philosophy of **VPR**'s [2] placement algorithm, which uses a linear function of wire length as a cost function for a *simulated annealing schedule*. We show how the cost function can be modified to reflect the placement requirements due to local variations in interconnection complexity in Section 5.

3. FPGA PLACEMENT PROBLEM

In this section, we describe the problem and the approach taken to solve it.

3.1 Problem Statement

Given an FPGA with a 2-D mesh architecture, with Rent's exponent, p_a, and a design mapped to k-input LUTs, having Rent's exponent, p_d, fit the design in the architecture such that the area of the design is minimized subject to the constraint of a given bounding box aspect-ratio requirement.

Some of the architecture LUTs will be left unused, so that there may be enough routing resources to interconnect the design's computing LUTs. The goal is to decide which of the architecture LUTs should be left unused. As will be shown later (See Section 5), for a good placement algorithm, this is mainly true for cases where $p_d > p_a$.

3.2 Analysis

The problem that we have to solve is to integrate knowledge of the interconnect complexity of a given design into the FPGA placement process. For a given architecture, the possible variables that we have during the placement process are as follows:

1. Position of LUT in FPGA fabric: This corresponds to the mapping of the design LUT to a specific LUT in the FPGA fabric. The placement process should be such that the interconnect requirements of the design LUTs are taken into account when doing the mapping.

2. Size of the FPGA fabric: This parameter warrants further examination. Let us call a *Minimum Size fabric*, a geometric area, with aspect ratio corresponding to the required aspect ratio, such that the number of the LUTs in that area are approximately equal to the number of LUTs in the design. For example, a design with 100 LUTs and an aspect ratio of 1, would require a minimum FPGA fabric of 10x10 LUTs.

We should increase the fabric to an appropriate size based on the difference in Rent's exponents to intelligently place the design on the new sized fabric. In a sense, we change the routing resource growth of the architecture to match that of the design. We do this by choosing the right LUTs to be mapped to. The fact that we leave certain LUTs untouched, is, equivalent to making sure that the routing resources of these LUTs are advantageously used for other LUTs in the placed design.

The methodology we used for our CAD flow is shown in the Figure 1.

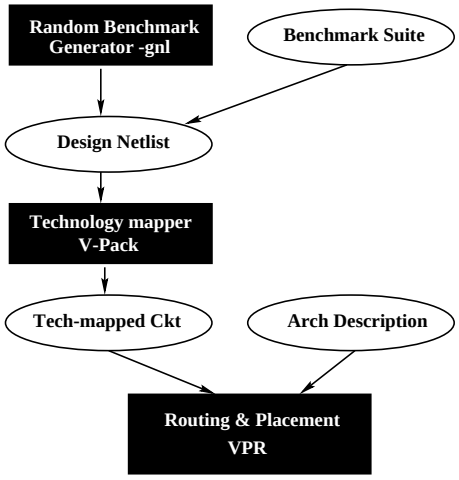


Figure 1: Basic FPGA CAD flow

3.3 Rent's exponent for the Design

Finding the Rent's exponent for a given design is relatively easy using the method proposed by Van Marck et al. [12]. Their method proposes using the net length distribution as a means to calculate the relation between interconnection complexity and block count. They describe a *Local Rent's exponent* for fine grain estimation of circuit complexity and methods to calculate it. We use this metric in our methodology.

For a block in a design, v_i, and for a given interconnect length, l_j, we count the number of interconnections N_j, of length l_j from that block v_i (i.e. both input and output nets), and find the best-fitting line on the $\log(N_j)$ versus $\log(l_j)$. The *local rent's parameter* is defined as the slope of this derived function. This local rent's parameter, p_d^l, of the design is a good measure of the local interconnection complexity of the circuit. The *global Rent's exponent* p_d^g, which is the exponent defined by Rent's rule, is computed by taking the sum of all the interconnection length distributions over all the blocks or LUTs, and will equal the one defined by Landman [11], only if the design has a uniform interconnection complexity.

4. FPGA INTERCONNECT ANALYSIS

Programmable interconnect is the dominant contributor to die area and cycle time in FPGAs. To support large, active functional density, the LUTs must be richly interconnected. In this section, we analyze interconnect growth with circuit size and establish a relationship between FPGA size and richness of FPGA interconnect. Figure 2 shows the features and a canonical LUT tile of conventional mesh-type FPGA architectures.

4.1 Minimum FPGA Fabric Calculation

We now look at how interconnect requirements and FPGA fabric size are related. As we have discussed before, the

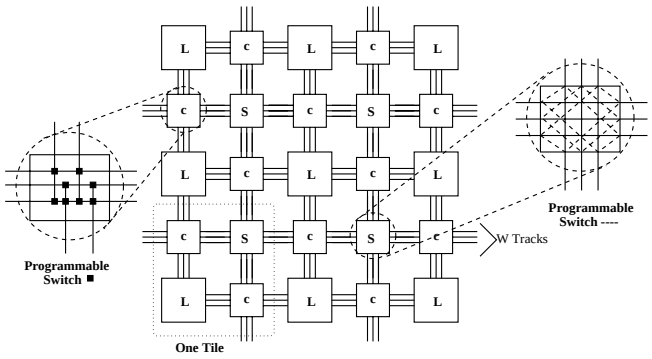


Figure 2: Conventional FPGA Architectural Definitions.

best characterization to-date which empirically estimates interconnect requirements is Rent's Rule (Equation (1)).The amount of interconnect we need to provide depends upon the connectivity of the FPGA. If $p_d > p_a$, we cannot simply map the design net-list on top of the device LUTs. We have to estimate how much larger the FPGA must be than the number of LUTs in the design in order to accommodate the highly connected design.

Let us call this scaling factor C. In order for the FPGA to accommodate the design, it must have enough i/o bandwidth into each lower-level partition. If we denote the interconnect requirements at the top level for the design as N_{io_d} and for the architecture as N_{io_a}, then the above requirement is summarized in Equation (2):

$$N_{io_d} \leq N_{io_a} \tag{2}$$

The only way to accommodate the requirement in Equation (2) with a fixed p_a is to scale up the size of the available FPGA area. Since the average number of interconnections for a design is fixed before and after it has been placed-and-routed,$K_d = K_a$. Now, applying Equation (1), the above means:

$$K_d.N_g^{p_d} \leq K_d(C.N_g)^{p_a} \tag{3}$$

Solving this relation for equality, at the tight bound :

$$N_g^{p_d} = (C.N_g)^{p_a} \tag{4}$$

$$N_g^{\frac{p_d}{p_a}} = C.N_g \tag{5}$$

$$C = N_g^{\frac{p_d}{p_a}-1} \tag{6}$$

We assume here, that once we can accommodate interconnect requirements at the top level of hierarchy of the design, all other levels are also accommodated as well.

With the above discussion, we conclude that for a given FPGA with fixed channel width, and a given aspect ratio A, the FPGA fabric size should be scaled by C, which is calculated based on the relationship between p_d and p_a. The X- and Y- dimensions of the FPGA fabric are calculated based on the required aspect ratio A, as done in **VPR** [2].

5. MVPR'S PLACEMENT ALGORITHM

MVPR uses a simulated annealing algorithm [10] for placement [2]. **VPR** uses an approximation of the total net length for a particular placement as the cost function. Using the *local Rent's exponent* measure that is described in Section 3, we derive a modified cost function, that attempts to take into account the difference in the interconnect complexity for a design LUT and the available routing resources for a particular placement.

We begin by calculating the *local Rent exponent*, p_d^l for every LUT in the circuit. Since the FPGA that we are targetting is a 2-D mesh, we can make the assumption that Rent's exponent is uniform throughout the architecture and hence conclude that the *local Rent's exponent*, p_a^l, is the same as the *global Rent's exponent*, p_a^g or p_a.

For every placement, we calculate the absolute difference between the local Rent's exponents $|p_d^l - p_a^l|$ and use this to scale the cost function using the bounding-box lengths. In other words, for every iteration of the simulated annealing, the cost associated with the change in the net lengths is modified by the cost associated with changing the local Rent's exponent.

The cost of this calculation is not significant since, we need to do a calculation of the local Rent's exponents for all the blocks only once. Following the initial calculation, the local Rent's exponents are calculated only for those blocks that can participate in a swap. This initial global calculation takes $O(nlog(n))$ time, since we have to do a sort of the net-lengths to calculate the distribution. Following this, we need to update it only for the blocks and associated nets that are swapped in an iteration of the anneal.

The cost function is now:

$$\sum_{n=1}^{N_{nets}} (1 + |p_{d_n}^l - p_a|)\,(wire\_length_n) \tag{7}$$

where $p_{d_n}^l$ corresponds to the local Rent's exponent for the n^{th} block, and $wire\_length_n$ corresponds to the wire length of the n-th wire. In our experiments, the wire length is estimated using the estimation method implemented in **VPR** [2].

Here we provide a qualitative analysis of the modifying factor $|p_{d_n}^l - p_a|$. Again, the effects are different depending on the value of $p_{d_n}^l - p_a$. We see two cases:

- $p_d^l \leq p_a$: In this case, the interconnect requirements of the design is less than the routing resources of the architecture. So, at first glance, this is uninteresting. However, as the Figure 3 shows, the design may have regions with different local interconnect requirements. As shown in the figure, R_2 has complexity less than R_1, while R is a global metric that does not capture this. In such a case, during the iterative placement process, a swap of LUTs that tries to ensure that this variation in interconnect requirement is minimized will have less cost than a swap that assumes that all interconnect requirements are uniform. Hence, a more fine grain approach to interconnect complexity comparison is done.

- $p_d^l > p_a$: In this case, the algorithm has the freedom to move into the direction of changing the interconnect complexity of the architecture. This is facilitated by the increased area fabric that we allow for placement. The algorithm has the freedom to choose certain LUTs in this fabric so that p_a is changed by the "elimination" of certain LUTs from the fabric, such that it is closer to p_d.

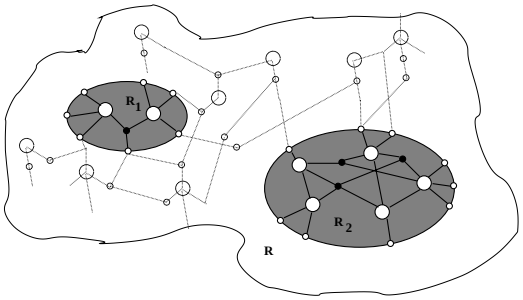


Figure 3: Interconnect complexity variations

6. EXPERIMENTS

In this section, we detail the various experiments that we conducted on our tool and the data that we collected from them. We used three architectures, all of which use the subset switch and 4-input/1-output LUTs. The track segmentation is the only parameter changed (1, 0.5:0.5 distribution of 1 and 2 segments, and 2). For the ISCAS benchmarks, only the architecture with segmentation of 1 was used.

6.1 Rent's exponent for the FPGA

Finding the Rent's exponent analytically for a given FPGA architecture is a difficult problem. Prior research in this area (see Section[2]), indicates that this is beyond the scope of this paper. However, it is still possible to find the Rent's exponent for a given architecture, p_a, empirically.

We define *network utilization* as, the ratio of used interconnect to the available interconnect in the FPGA after place-and-route. We use the idea that, given an architecture description and a good place-and-route tool, the interconnection complexity of the circuits that best utilize the routing resources on the FPGA, is a close measure of the available interconnection complexity on the FPGA.

In other words, we place-and-route various designs with various known rent's exponents on the target FPGA architecture and find the function between the rent's exponent of the placed circuits and the *network utilization* of the FPGA. The maximum value of this function is taken as the Rent's exponent of the architecture p_a.

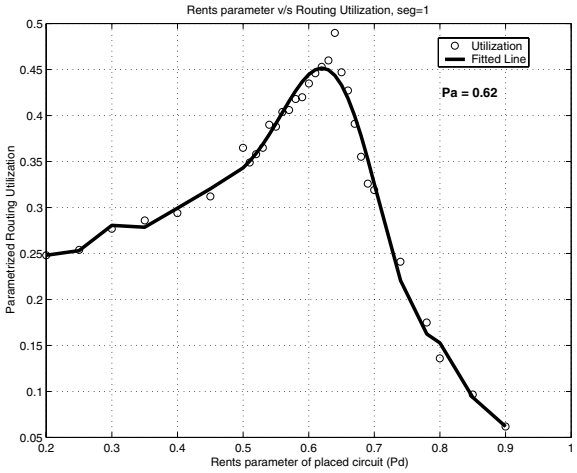
We generate random benchmark designs of uniform interconnection complexity, for a constant gate count and a range of Rent's exponents ($0.1 < p_d < 0.9$), using **gnl** [13]. The size of the circuits (about a 1000 LUTs) and the number of routing tracks was chosen (7) in order to minimize the effect of small circuit size and granularity of the tracks. The resulting data from the above empirical methodology is plotted and the graphs are shown in the Figures 4(a),4(b), and 4(c).

6.2 Random Benchmark Circuits

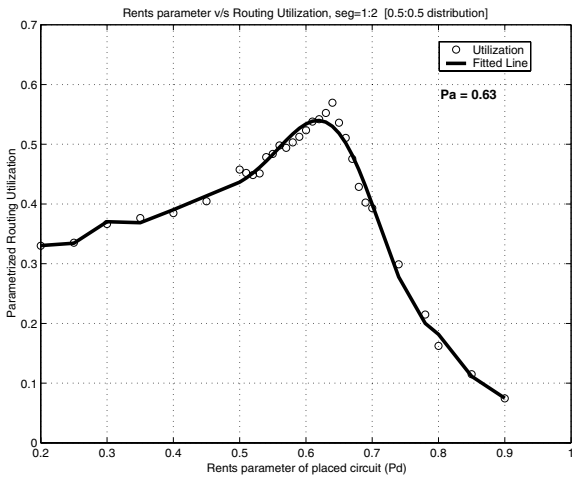
34 benchmark circuits with a uniform interconnect distribution throughout the circuit was generated using **gnl** and was used as the benchmark set for the following experiments. All the circuits have 1000 LUTs and have varying Rent's exponents from 0.2 to 0.9.

6.2.1 Routing Efficiency

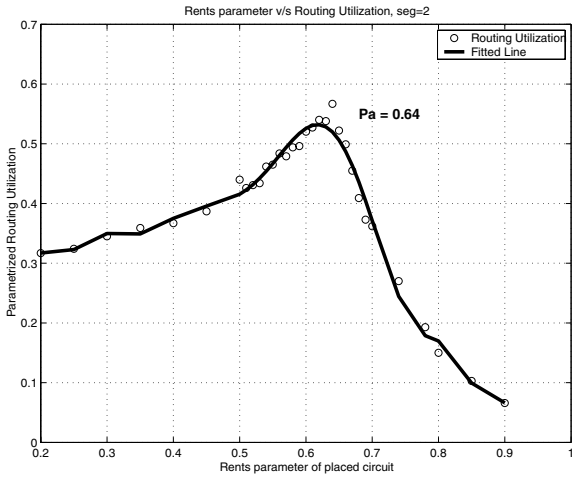
In this experiment, we tried to determine whether the track utilization required to place-and-route a design changes with the new placement methodology. The same benchmark cir-



(a) Routing Utilization v/s P_d, Seg=1, P_a=0.62.

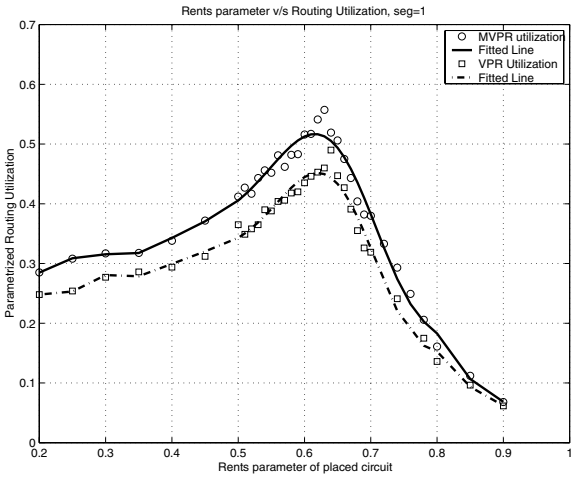


(b) Routing Utilization v/s P_d, Seg=1:2, P_a=0.63

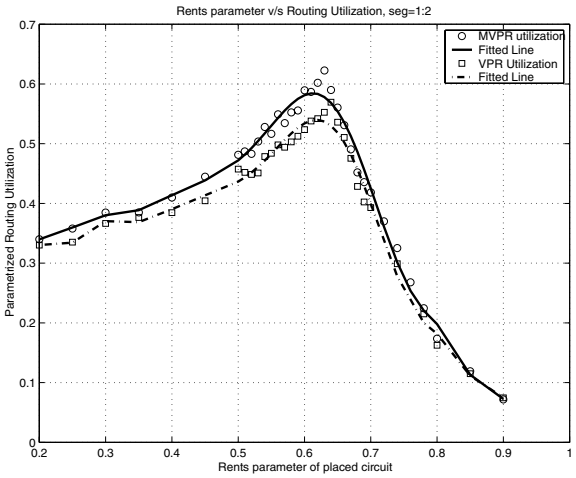


(c) Routing Utilization v/s P_d, Seg=2, P_a=0.64

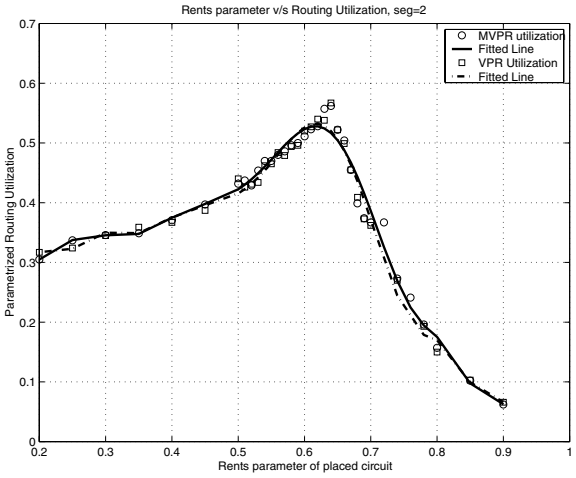
Figure 4: Routing Utilization v/s P_d



(a) Routing Utilization v/s p_d, Seg=1, p_a^{est}=0.62.

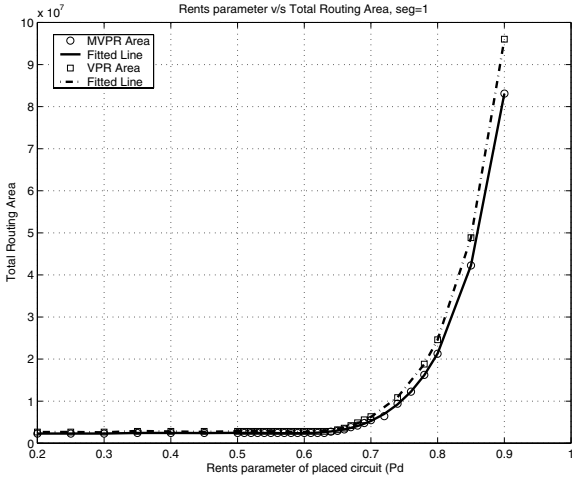


(b) Routing Utilization v/s p_d,Seg=1:2,p_a^{est}=0.63

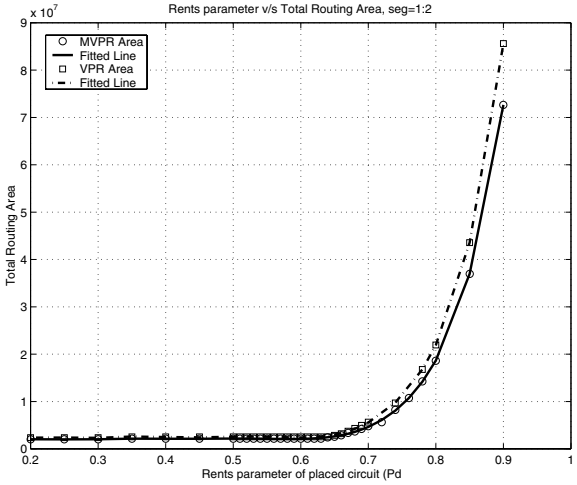


(c) Routing Utilization v/s p_d, Seg=2, p_a^{est}=0.64

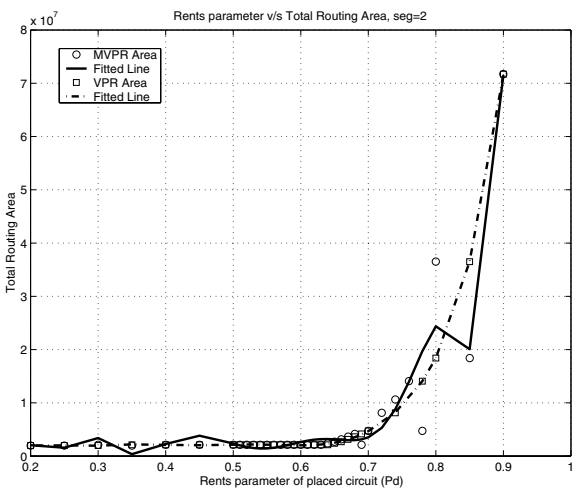
Figure 5: Routing Utilization v/s p_d



(a) Total Area in Trans. Eqvts v/s p_d, Seg=1

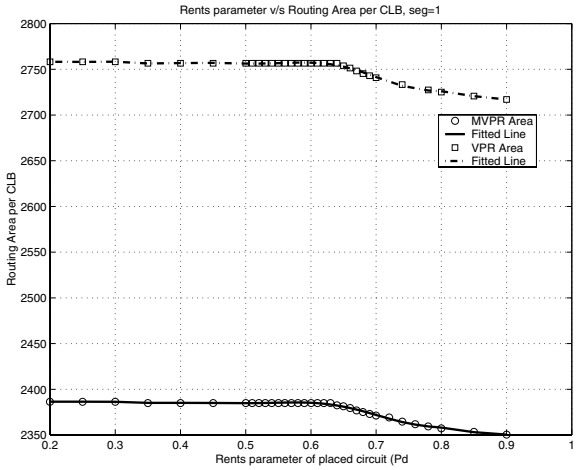


(b) Total Area in Trans. Eqvts v/s p_d, Seg=1:2

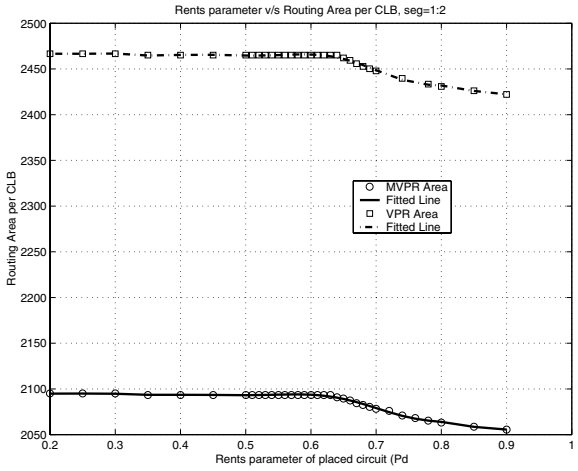


(c) Total Area in Trans. Eqvts v/s p_d, Seg=2

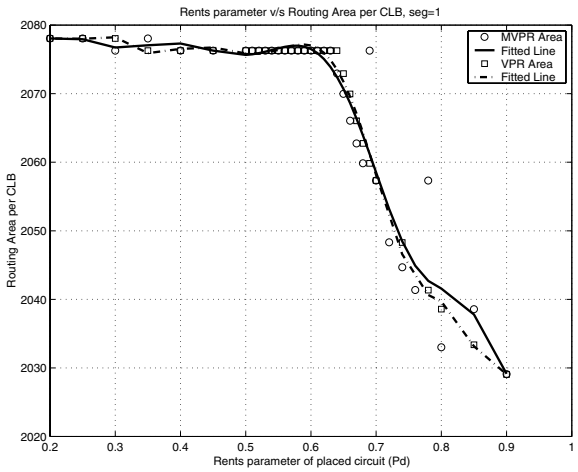
Figure 6: Total Area in Transistor Eqvts v/s p_d



(a) Area per LUT in Trans. Eqvts. v/s p_d, Seg=1



(b) Area per LUT in Trans. Eqvts. v/s p_d, Seg=1:2



(c) Area per LUT in Trans. Eqvts. v/s p_d, Seg=2

Figure 7: Area per LUT in Transistor Eqvts. v/s p_d

cuits were used for this experiment. The channel width was fixed to the minimum channel width required to place-and-route all the circuits.

The results are shown in the Figure 5. The results are shown for all the 3 architectures evaluated. As we can see, as p_d increases, the track utilization as a fraction of available tracks and utilized tracks follow a bell shaped curve for both **VPR** and **MVPR**. However, in the case of **MVPR**, the routing utilization efficiency is higher than that of **VPR**, by as much as 10-15%, in all cases, except for extreme values of p_d. The results in terms of area per LUT are shown in the Figure 7.

6.2.2 *Area Requirements*

In this experiment, we tried to determine whether the area required to place-and-route a design changes with the new placement methodology. The channel width was fixed to the minimum channel width determined from the previous experiment to place-and-route all the circuits. The results for area utilization are presented in absolute numbers i.e. as total transistor equivalents and per LUT used.

The results in terms of total area are shown in the Figure 6. The results are shown for all the 3 architectures evaluated. As we can see, as p_d increases, the area utilization in absolute terms increases for both **VPR** and **MVPR**. However, in the case of **VPR**, the area requirement is as much as 25% higher than for **MVPR**, especially for cases, where $p_d > p_a$. We see that the area change is not significant for the architecture where segmentation is 2. This could be because due to track segmentation having effects on the routing that we have not considered.

6.3 ISCAS benchmarks

The ISCAS89 benchmarks were evaluated solely for the purpose of determining whether circuits with varying requirements in local interconnection complexity would be placed better with **MVPR** or not. The circuits were placed and routed on an architecture with a segmentation of 1. The results are shown in Table 1. The results for **MVPR** versus **VPR** are almost the same except that we get a decrease in wire length and an increase in routing utilization for some of the benchmarks. We get better routing utilization using **MVPR**, *even when we do not resize the circuit*, as compared to **VPR**. However, this is only about $2\% - 5\%$. The problem seems to be that variation in local interconnect complexity is not being sufficiently weighted in the optimization, since the benchmarks with uniform interconnect complexity show a marked improvement with **MVPR**.

7. CONCLUSION

In this paper, we presented a new simulated annealing approach for taking into account interconnect complexity issues during placement.

This algorithm is able to reduce the area overhead by 10-15% on the selected architectures. It also increases the resource utilization by about 25%, without any significant increase in run-times. We also show analytically, that the FPGA fabric required for placement can be sized to a optimal value depending on the relationship of p_d and p_a. Consequently, we can conclude that using interconnection complexity to guide placement is a worthwhile approach.

However, this approach is highly dependent on accurate estimation of Rent's exponent. We have not explored the

Table 1: Results for ISCAS Benchmarks

Circuit	vpr Trk	mvpr Trks	vpr avg. wire len.	mvpr avg. wire len	vpr area/LUT	mvpr area/LUT	vpr util%	mvpr util%
C2670	4	5	5.82	4.79	1102.63	1368.11	0.19	0.12
C3540	6	7	7.99	8.10	1634.53	1878.25	0.68	0.58
C432	4	4	5.18	4.96	1159.77	1159.77	0.76	0.72
C499	5	5	6.68	6.26	1447.80	1447.80	0.69	0.65
C6288	4	4	5.41	5.29	1121.00	1121.00	0.69	0.67
C7552	5	5	7.30	7.56	1371.93	1371.93	0.48	0.50
C880	5	5	5.83	5.52	1424.67	1424.67	0.69	0.66
alu2	5	5	6.64	6.70	1400.11	1400.11	0.63	0.64
alu4	5	5	7.69	7.57	1387.26	1387.26	0.70	0.69
s1196	4	4	5.28	5.09	1115.68	1115.68	0.63	0.61
s1238	4	4	5.26	5.37	1115.68	1115.68	0.63	0.65
s1423	3	3	3.11	3.18	871.17	871.17	0.49	0.51
s208.1	2	2	2.40	2.43	626.65	626.65	0.57	0.58
s344	2	2	2.50	2.51	616.65	616.65	0.56	0.57
s349	2	2	2.76	2.64	616.65	616.65	0.63	0.60
s382	3	4	3.12	3.04	890.14	1136.63	0.46	0.34
s38417	4	4	4.22	4.05	1090.68	1090.68	0.51	0.49
s400	3	3	3.54	3.25	890.14	890.14	0.56	0.51
s420.1	3	3	2.76	2.83	887.66	887.66	0.45	0.46
s444	3	3	3.17	2.90	887.66	887.66	0.49	0.45
s526	3	3	3.76	3.62	887.66	887.66	0.59	0.57
s526n	3	3	3.78	3.94	887.66	887.66	0.54	0.56
s641	4	3	2.43	2.51	1124.41	873.88	0.28	0.26
s713	3	3	2.81	2.78	880.09	873.88	0.48	0.31
s838.1	3	3	2.84	2.86	876.05	876.05	0.47	0.47
s838	3	3	2.78	2.82	874.92	874.92	0.45	0.45
s953	4	4	4.61	4.60	1116.86	1116.86	0.55	0.55
Average	3.67	3.68	4.27	4.08	1048.38	1066.85	0.53	0.54

issues of how this placement will affect timing or power consumption. Second order effects of the assumptions that we have made have not been considered on the placement results. However, the size of the circuits considered are large enough to bring in some measure of confidence that this is a reasonable approach.

8. ACKNOWLEDGMENTS

The authors thank Dr. K-T Cheng of UCSB, for his support, and M. Iyer for valuable discussions.

9. REFERENCES

[1] M.J. Alexander, J.P Cohoon, J.L. Ganley, and G. Robins. An architecture-independent approach to FPGA routing based on multi-weighted graphs. In *EDAC*, pages 259–264, Sept 1994.

[2] V. Betz and J. Rose. Vpr: A new packing, placement and routing tool for fpga research. In 7^{th} *Int. Workshop on field-programmable logic and applications*, pages 213–222, 1997.

[3] S. Brown, J. Rose, and Z.G. Vranesic. A stochastic model to predict the routability of field-programmable gate arrays. *IEEE Trans. on CAD*, pages 1827–1838, Dec 1993.

[4] P.K. Chan, M.D.F .Schlag, and J.Y. Zien. On routability prediction for field programmable gate arrays. In *DAC*, pages 326–330, June 1993.

[5] Andre DeHon. Balancing interconnect and computation in a reconfigurable computing array. In *Int. Symp. on FPGAs*, pages 125–134, February 1999.

[6] S. Devadas. Optimal Layout via Boolean Satisfiability. In *ICCAD*, pages 294–297, Nov. 1989.

[7] W.E. Donath. Placement and average interconnection requirements of computer logic. *IEEE Trans. on Circuits and Systems*, CAS-26:272–277, 1979.

[8] A.A. El. Gamal. Two-dimensional stochastic model for interconnections in master-slice integrated circuits. *IEEE Transactions on Circuits and Systems*, CAS-28:127–138, Feb 1981.

[9] L. Hagen, A.B. Kahng, F.J. Kurdahi, and C. Ramachandran. On the intrinsic rent's parameter and spectra-based partitoning techniques. In *EDAC*, pages 202–208, Sept 1992.

[10] S. Kirkpatrick, C.D. Gelatt, and M.P Vecchi. Optimization by Simulated Annealing. *Science*, 220:671–680, 1983.

[11] B.S. Landman and R.L. Russo. On a pin versus block relationship for partitions of logic graphs. *IEEE Trans. on Computers*, C-20:1469–1479, 1971.

[12] H. Van Marck, D. Stroobandt, and J. Van Campenhout. Toward an extension of rent's rule for describing local variations in interconnection complexity. In *Proc. of* 4^{th} *Intl. Conference for Young Computer Scientists*, pages 136–141, 1995.

[13] D. Stroobandt, P. Verplaetse, and J. Van Campenhout. Generating Synthetic Benchmark circuits for evaluating CAD tools. *IEEE Trans. on CAD*, 19(9):1011–1022, September 2000.

[14] R.G. Wood and R.A. Rutenbar. FPGA routing and routability estimation using boolean satisfiability. In *Proc. ACM Int. Symposium on FPGAs*, Feb 1997.

[15] Y.L Wu and D.Chang. On the NP-completeness of regular 2-D FPGA routing architectures and a novel solution. In *ICCAD*, pages 362–367, Nov 1994.

[16] Y.L. Wu, S. Tsukiyama, and M. Marek-Sadowska. Graph based analysis of 2-d fpga routing. *IEEE Trans. on CAD*, (1):33–44, Jan 1996.