CROSS TALK AWARE MULTI-OBJECTIVE OPTIMAL ROUTING FOR ISLAND-STYLE FPGAs

Rajesh Tiwari[1] **V. Sahula**[2]

Abstract

Field programmable gate arrays (FPGAs) have become imperative for implementing large digital circuits in customized VLSI circuits. A key component in the design of an FPGA is its routing architecture, which comprises of the wiring segments and routing switches that interconnect the FPGA's logic blocks. In order to obtain optimal routing solution in a reasonable amount of time, evolutionary approaches are employed, which provide better quality solutions in shorter run times compared to other approximate algorithms. Evolutionary approaches perform even better when optimization of multiple objectives is considered by virtue of their population-based exploration strategy. This paper illustrates the optimization of the delay and cross talk both for FPGA routing using non-dominated-sorting-genetic-algorithm (NSGA) methodology as proposed by N. Srinivas and K. Deb (1994), formulated as multiple objective optimization problem. The proposed global router accepts placed-netlist as input; divides connections into two-point connections and then routes each of them separately. The routing is optimal for both the delay and cross-talk for a connection.

1. Introduction

Nowadays, Field-Programmable Gate Arrays (FPGAs) are being widely accepted as means of implementing large digital circuits in a customized VLSI chip. *Array-based* architecture is one of the most significant types among a large number of commercially available styles of FPGAs, It consists of rows and columns of logic blocks with horizontal routing channels between the rows and vertical channels separating the columns.

The traditional routing strategy is a two-stage approach, in which global routing is followed by detailed routing. There are three main concerns of the present work- (i) Multiple Objectives, (ii) Evolutionary algorithm approach (i.e. Genetic algorithm), and (iii) Routing for FPGAs. One of the most important aspects of our work is multi-objective optimization, where we consider two objectives crosstalk and delay both of which have to be minimized for a connection. Genetic algorithm (GA) works on a collection of several alternative solutions to a given problem. Each solution in the population is called a string or chromosome. GA, which is a probabilistic approach, searches for the population of points instead of single point. Our approach borrows the genetic operators' formulation from C. W. Ahn and R. S. Ramakrishna (2002). The crossover operation exchanges partial chrom+----------osomes (partial routes) at positionally independent crossing sites and the mutation operation maintains the

[1] Design Engineer, Texas Instruments, Bangalore. rktiwari@ti.com
[2] Reader (Assoc. Prof.), Department of ECE, NIT Jaipur. sahula@ieee.org

genetic diversity of the population. The algorithm can cure all the infeasible solution with a simple repair function.

Although FPGAs nowadays have high logic capacities, yet speed performance has become a critical issue during digital design implementation for systems. The generalized interconnect structure of the FPGA forms the largest performance bottleneck. In particular, routing architecture research needs good routing software to fairly evaluate the capabilities of each FPGA design. Presently available routers like VPR as proposed in V. Betz and J. Rose (1997) and time driven router as proposed in V. Betz, J. Rose, and A. Marquardt (1999) etc. are single objective optimization based tools. This paper discuses a global routing solution and describes a framework that is adapted to island style FPGA routing architectures. We present a brief overview of multi-objective optimization and discuss the algorithm parameters in Section 2. Alongside brief introduction to genetic algorithm and cross talk model, we provide an algorithm statement in Section 3. We present the evaluation results of optimization framework in Section 4 and conclude in Section 5.

2. Multi-objective optimization: An overview

A general multi-objective optimization problem consists of a number of objectives and is associated with a number of inequalities and constraints. An example statement is as follows.

Minimize/Maximize $f_i(x)\ \forall i=1,2,\ldots N;$ Subjected to $g_j(x)\ \forall j=1,2,\ldots J$ and $h_k(x)\ \forall k=1,2,\ldots K.$

2.1. Terminology

We discuss few terms and concepts related to optimization space and genetic algorithms before going through the overview of solution approaches, for details, see K. Deb (2001).

Pareto optimality: A point $X \in f$ is pareto optimal if for every $X \in f$ either $\prod_{i \in I}\left(f_i\left(X\right) = f_i\left(X^*\right)\right)$ or, there is at least one $i \in I$ such that $\left(f_i\left(X\right) > f_i\left(X^*\right)\right)$. Thus when a set P is the entire search space, or P≡S, the resulting non dominated set P_P is called the pareto optimal set.

Dominance : A solution $x^{(1)}$ is said to dominate the other solution $x^{(2)}$, only if both of the following conditions are true- (i) the solution $x^{(1)}$ is no worse then $x^{(2)}$ in all objectives, and (ii) the solution $x^{(1)}$ is strictly better than $x^{(2)}$ in at least one objective. For a given finite set of solutions, we can perform all possible pair wise comparisons and find which solution dominates which and which solutions are non-dominated with respect to each other. At the end we get a set of solution that does not dominate each other. This set has another property that any solution outside of this set, will be dominated by a solution in the set. This mean that the solutions of this set are better compared to rest of the solutions. This set has a special name called the *non-dominated* set for the given set of solution.

Model for sharing function

Instead of replacing a solution by a similar solution, D. E. Goldberg and J. Richardson (1989) suggested a concept, where the focus was on degrading the fitness of the similar solution. It is required to find q optimal solutions with the available resource of finite population with N

slots. As $q<<N$, each optimum can work with an adequate subpopulation (niche) of the solution. Since the population will need representative solution of all q optima, somehow the available resources of N population slot must have to be shared among all representative solutions. Goldberg and Richardson (1989) suggested an adaptive strategy where a sharing function is used to obtain an estimate of number of solution belonging to each optimum.

$$Sh(d) = \begin{cases} \left(1 - \left(\frac{d}{\sigma_{share}}\right)\right)^{\alpha} & if \quad d \le \sigma_{share} \\ 0 & otherwise \end{cases}$$

If d=0 then $Sh(d)=1$, then a solution has full sharing effect on itself. If $d > \sigma_{share}$, the two solutions are at least a distance of σ_{share} away from each other. If $Sh(d)$=0 then the two solutions, which are at a distance of σ_{share} away from each other, do not have any sharing effect on each other. A niche count nc_i, which is the estimate of the extent of the crowding near a solution, is given by $nc_i = \sum_{j=1}^{N} Sh(d_{ij})$ .The parameter d_{ij} is the distance between the i^{th} and j^{th} solution. Finally, the shared fitness value is computed as $f_i' = f_i / nc_i$.

2.2. Overview of Evolutionary approaches

Two major problems must be addressed when an evolutionary algorithm is applied to Multi-objective optimization- (i) How to accomplish fitness assignment and selection, respectively, in order to guide the search towards the Pareto-optimal set; and (ii) How to maintain a diverse population in order to prevent premature convergence and achieve a well distributed trade-off front. Different approaches are classified with regard to the first issue, where one can distinguish between criterion selection, aggregation selection, and Pareto selection. Methods performing criterion selection switch between the objectives during the selection phase. Each time an individual is chosen for reproduction, potentially a different objective will decide which member of the population will be copied into the mating pool. Aggregation selection is based on the traditional approaches to multi-objective optimization where the multiple objectives are combined into a parameterized single objective function. The parameters of the resulting function are systematically varied during the same run in order to find a set of Pareto-optimal solutions. Finally, Pareto selection makes direct use of the dominance relation. D. E. Goldberg and J. Richardson (1989) were the first to suggest a Pareto-based fitness assignment strategy. The other approaches reported in literature are C. W. Ahn and R. S. Ramakrishna (2002), S. Brown, J. Ross, and Z. G. Vranesic (1992), J. Greene et al (1990); Niched Pareto Genetic Algorithm (NPGA) by J. Horn and N. Nafpliotis (); Nondominated Sorting Genetic Algorithm (NSGA) by N. Srinivas and K. Deb (1994); Vector Evaluated Genetic Algorithm (VEGA) as discussed in J. D. Schaffer (1984) and Steven J. E. Wilton (2001).

3. Cross talk aware Genetic Algorithm

It has been shown by S. Trimberger, ed. (1994) that the feasibility of FPGA design is constrained by routing resources more than by logic resources. Routing channels in symmetrical-array architecture FPGA contain predefined wiring segments of various lengths. These may be connected to the pins of the gates or

joined end-to-end to form longer segments by programmable switches. In this section, algorithm formulation is discussed with twin objectives of minimizing delay as well crosstalk.

3.1. Genetic operators

We adopt genetic operators formulation from C. W. Ahn and R. S. Ramakrishna (2002).

3.1.1. Chromosome encoding

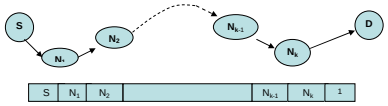


Figure 1 Encoding Scheme and Routing Path

A chromosome of the proposed GA consists of sequences of positive integers that represent the IDs of nodes through which a routing path passes. The length of the chromosome is variable, but never exceeds the maximum length, equal to the total number of nodes in the network. A chromosome (routing path) encodes the problem by listing up node IDs from its source node to its destination node based on depth first search database of the circuit.

3.1.2. Crossover

In the proposed scheme, two chromosomes chosen for crossover should have at least one common gene (node) except for source and destination nodes, but there is no requirement that they be located at the same locus. The procedure is illustrated in Figure 3. It is possible that loops are formed during crossover. To solve this we uses repair function.

Repairing for Routing Loop avoidance

It should be remembered that none of the chromosomes of the initial population or after the mutation is infeasible because when once a node is chosen, it is excluded from the candidate nodes forming the rest of the path. The repair function finds and eliminates loops in a routing path without unduly increasing computational costs.

```
// C: input chromosomes, C*: output chromosomes, ℓ: length of chromosome
For (all i, j AND i < ( ℓ - j) {
        if( C[i] ==C[ℓ -j])                    //Find a loop
        C*=C[1:i]||C[ℓ -i+1:l];    }           // Eliminate the loop
```

Figure 2 Procedure for the Repair Function

The proposed repair function is described in Figure 2 and illustrated in Figure 3. One of the offspring produced after crossover becomes infeasible because the new route contains the loop. The repair function detects the loop by a simple search. After that, the lethal genes (forming the loop) that violate the constraint condition are deleted; those nodes are (but one of two) in this example.

3.1.3. Initial population generation

We use the depth first traversal method for initial population generation. After the initial population has been generated, we evaluate each individual and select the half of the best solutions and add them to next generation. Then we apply genetic operators to create offsprings and if they are valid we keep adding them to the next generation till they all form a population of preconceived size.

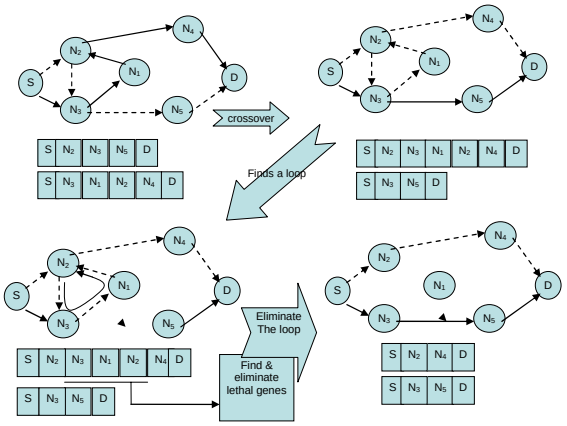


Figure 3 Loop prevention- (a) Two Parents (b) Two Children after crossover (c) Child having loop (d) Child after repair

3.2. Delay and cross talk modeling

3.2.1. Delay model

Figure 4 shows the RC tree corresponding to a path for a net. For each of the paths, it is assumed that the logic block driving the net is characterized by an output resistance and capacitance called R_{source} and C_{source}.

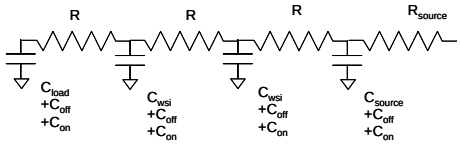


Figure 4 RC tree network

Also, a load capacitance called C_{load} is added at the sink end of the path. In Figure 4 path shown involves three resistors, labeled R, that correspond to routing switches. Also, the values of the capacitors account for the number of routing switches in the path that are turned ON, the number of OFF switches that "hang on" the wire segments in the path, and the capacitance of the wire segments (C_{wsi} represents the capacitance of a wire segment that spans i logic blocks). The values of R and C are taken from M. Khellah, S. Brown, and Z.Vranesic (1993). Using the analytic model, the delay of a net is defined as the largest delay from the net's source to any of its sinks. We then define the speed-performance of an entire circuit implemented in an FPGA as the *average of the net delays in the circuit*.

3.2.2. Crosstalk Model

Crosstalk occurs when a change in voltage on one trace causes a change in voltage on a nearby trace. We assume a simple model, in which crosstalk between two traces is modeled by a change in the effective capacitance seen by both traces. The capacitance of a trace i can be written as $C_i = C_{fixed} + \sum_{j \in T \wedge j \neq i} \left(\lambda_{ij} C_c l_{ij}^{\eta} / d_{ij}^{\gamma} \right)$. Assuming all tracks are within a channel are separated by the same distance, by combining $d_{i,j}$ with the technology dependent terms, and approximating η as 1, we can write $C_i = C_{fixed} + \sum_{j \in T} \lambda_{ij} C_c l_{ij}$. Here,

C_{fixed} is the capacitance of the metal trace itself, as well as any switches or buffers attached to the trace, T is the set of all traces, l_{ij} is the distance that traces i and j are adjacent, $d_{i,j}$ is the distance between trace i and j, λ_{ij} is a parameter described ahead, and γ, η, and C_c are process-dependent constants (typical values of γ and η are 1.8 and 0.92 respectively). If the two tracks are switching in the same direction, $\lambda_{i,j}$ can be approximated as 0, since there is no induced crosstalk for signals switching together. If the two signals are switching in the opposite direction, $\lambda_{i,j}$ can be approximated as 2. If only one of the signals is changing, λ_{ij} can be approximated as 1.

```
Step 1 Convert the two-point placed netlist into intermediate format.
Step 2 Calculate the overall segments required to route a net. Sort the nets in decreasing
            order of number of    segments required to route each of the net.
            Sort in decreasing order of criticality, among sub-groups.
Step 3 While (there exist a net to be routed) {
            Input Source and Destination Nodes;
            GENERATE initial population;
            Calculate delay and crosstalk for each Individual;
            For(ng=0;ng < no_of_gen;ng++) {
              EVALUATE fitness (Using NSGA);
              SORT (Decreasing Order of Dummy Fitness);
              SELECT (population_size/2);
              While(no_of_sec_pop < population_size) {
                   Perform CROSSOVER to create new offspring;
                   MUTATE(Offspring with Mutation Rate);
                        REPAIR solutions with Loops
                        ADD offspring to New Population; }          }
            EVALUATE Population (Using NSGA);
            SORT( Decreasing Order of Dummy Fitness);
            Select Best Individual;
            Check for constraint of Maxnet to be routed through Channel;
            Select that satisfies constraint;
            Show the delay and net's overlap with previous five routed nets;    }
Step 4 For(Netno=1;Netno ≤ maxnet;Netno++)
            Calculate the Exact Overlap of Net;
```

Figure 5 Algorithm statement- Multi-objective GA

During routing, it is difficult to determine the relative switching activities of each signal. Thus, our timing model makes the following pessimistic assumptions when computing the effective capacitance of tracks *i*- (i) For all neighbors *j* which are used to carry a signal, we assume the worst-case crosstalk ensues (i.e. $\lambda_{i,j}$ is 2). (ii) For all neighbors *j* which are unused, we assume that *j* is tied to a constant voltage, so $\lambda_{i,j}$ is 1. Thus, for any given routing, the effective capacitance of each track can be computed, and this can be used in the Elmore delay to estimate the delay of the net. But this method can be effectively applied during the detailed routing. While performing the global routing it is not meaningful to compute the exact crosstalk as during global routing, although a net may get assigned to a channel, the exact track occupied by it not known. We have chosen a simplified model in which *Cross-talk* $\propto$ *overlapping segment-length*. While assigning the channel for a connection of a net we consider five previously-routed-critical-nets overlapping as a cross-talk value. In this manner, we ensure to follow the philosophy imbibed in aforesaid analysis for computation of C_i.

3.3. Multiple objective cost function and algorithm

Our objective here is to minimize the objectives, the delay and the crosstalk. We assume two variables x_1 and x_2, where x_1 represents the delay of the chromosome and x_2 represents the crosstalk. The problem formulation can now be given as follows.

$$Minimize\ f_1(x)\ AND\ f_2(x)\ when\ f_1(x) = x_1;\ f_2(x) = (1+x_2)/x_1$$

This problem looks simple but has conflicting scenarios between both the objectives, resulting in a set of pareto optimal solutions. This manipulation of the two variables allows us to find the following relation ship between both the objectives as $f_2 = (1+x_2)/f_1$. The values of f_1 and f_2 are calculated for each of the individual after NSGA has been used to assign the dummy fitness. The individual having the maximum value of the shared fitness is the best solution of the present population. A formal statement of the genetic algorithm is provided in Figure 5.

4. Results

An architecture consisting of array of CLBs of size 4×4 is chosen for evaluation of algorithm and routing is performed for the longest connection by varying the *population-size*, maximum number of *generations* used, and the *sharing parameter* (σ_{share}). The graph model is illustrated in Figure 6. The obtained results show that on increasing the number of generations and the population size, better results in terms of required number of segments for routing can be obtained. As illustrated in Figure 6, one of the longest paths in 4x4 array of CLB architecture would be from 0 to 23, which requires 6 segments for routing. For routing of this connection between 0 and 23, the results are presented in Table 1.

Table 1 Solution quality in term of # of segments

σ_{share}	Population size	Number of Generations				
		5	8	10	15	20
0.10	10	17	17	16	16	14
	15	14	14	14	11	11
	20	11	11	11	11	11
	25	11	11	9	11	11
0.15	10	16	16	13	13	13
	15	12	12	12	12	12
	20	11	11	8	6	6
	25	11	11	11	6	6
0.20	10	17	17	16	16	17
	15	13	13	12	11	11
	20	11	11	11	8	11
	25	8	8	9	8	11

On observing the Table 1, it is obvious that increasing number of generations beyond 15 or the population size beyond 20, the improvements in the results are not significant.

Table 2 Optimum Parameters Values

σ_{share}	First optimum segments are obtained for	
	Population size	No. of Generation
0.10	15	15
0.15	20	15
0.20	20	15

In order to get better solution, the population size should be at least 20, and number of generations should be at least 15. Here, σ_{share} is an important parameter on which our optimum path depends. Results show that present case, σ_{share} =0.15 is a good approximation. Further the exploration of routing results is done for the given the circuit connections as shown in Figure 6 (a). The solution paths are illustrated in Figure 6 (b).

Table 3 Routing Results for circuit shown in Figure 6 (a)

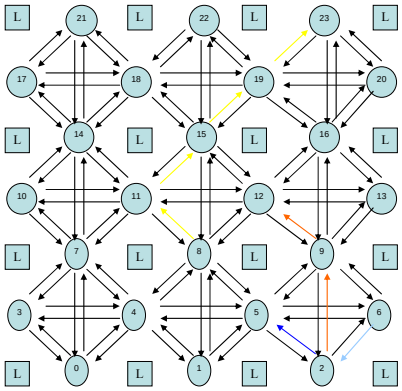
Populaton Size=20, Number of Generation=15, dshare=0.15				
Routing Parameters	***Net 1***	***Net 2***	***Net 3***	***Net 4***
Min No. of Segments Required	1	1	2	4
Segments Required after Routing	1	1	2	4
Delay(ns)	0.2089767	0.2089767	0.341533e	0.4078112
Overlapping Segments	0	0	0	0

5. Conclusions

In this paper, we have presented an optimal FPGA global router, which optimizes delay along with crosstalk using multi-objective genetic approach. Exact crosstalk can be measured only if some detailed routing information is available. We have used the delay model proposed by M. Khellah, S. Brown, and Z.Vranesic (1993) that can be replaced by a better model in future. An efficient method for Crosstalk and Delay estimation will improve the performance of algorithm when replaced in future.

Input Circuit Netlist
1 2 5 1
2 6 2 1
3 2 12 2
4 8 23 4

(a)



(b)

Figure 6 (a) Test circuit netlist;
(b) Path shown graphically for solution in Table 3;
Paths are (2 5) (6 2) (2 9 12) and (8 11 15 19 23)

6. References

[1] C. W. Ahn and R. S. Ramakrishna (2002). *A Genetic Algorithm for Shortest path routing problem and the sizing of population*. IEEE Trans. Evol. Compute, vol. 6, No.6..

[2] M. Beccer and I. Hajj (2000). *An analytical model for delay and crosstalk estimation with application to decoupling.* International Symposium on Quality of Electronic Design.

[3] V. Betz and J. Rose (1997). VPR: *A New Packing, Placement and Routing Tool for FPGA Research*. Int. workshop on Field Programmable Logic and Applications, pp.213-222.

[4] V. Betz, J. Rose, and A. Marquardt (1999). *Architecture and CAD for Deep-Submicron FPGAs*. Kluwer Academic Pub.

[5] S. Brown, J. Ross, and Z. G. Vranesic (1992). *A Detailed Router for Field Programmable Gate Arrays*. IEEE Trans, Computer-Aided Design, vol.11 pp. 620-627.

[6] K. Deb (2001). Multi-Objective Optimization using Evolutionary Algorithms, Reading, John Wiley.

[7] C. M. Fonseca and P. J. Fleming (1995). *An overview of evolutionary algorithms in multiobjective optimization*. Evolutionary Computation, 3(1):1–16.

[8] T. Gao, C.L. Liu (1996). *Minimum Crosstalk Channel Routing*. IEEE Transactions on Computer-Aided Design, Vol. 15, No. 5, pages 465-474.

[9] D. E. Goldberg and J. Richardson (1989). *Genetic algorithm with sharing and multimodel function optimization*. First International conference on Genetic Algorithm and their Applications, pp. 41-49.

[10] J. Greene, V. Roychowdhury, S. Kaptanoglu, and A. El Gamal (1990). *Segmented Channel Routing*. Proc. 27th Design Automation Conference, pp. 567-572.

[11] J. Horn & N. Nafpliotis (). *Multiobjective optimization using the niched pareto genetic algorithm*. Gal Report No. 93005. Illision Genetic algorithm laboratory.

[12] M. Khellah, S. Brown, and Z.Vranesic (1993). *Modeling Routing Delays in SRAM-based FPGAs*. Proc. 1993 CCVLSI, Banff, Canada, pp. 6B.13-6B.18.

[13] J. D. Schaffer (1984). *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. Ph.D. thesis, Vanderbilt University, Nashville, Tennessee.

[14] N. Srinivas and K. Deb (1994). *Multiobjective optimization using nondominated sorting in genetic algorithms*. Evolutionary Computation, 2(3):221–248.

[15] S. Trimberger, ed. (1994). *Field Programmable Gate Array Technology*. Kluwer Academic Publishers.

[16] Steven J. E. Wilton (2001). *Crosstalk aware Time Driven Router for FPGA*. FPGA 2001, Monterey, CA, USA.