JOURNAL OF CIRCUITS, SYSTEMS AND COMPUTERS, 1996

GASBOR: A Genetic Algorithm for Switchbox Routing in Integrated Circuits

Jens Lienig and K. Thulasiraman

Department of Electrical and Computer Engineering
Concordia University
1455 de Maisonneuve Blvd. West
Montreal, Quebec H3G 1M8, Canada
E-mail: jensl@ece.concordia.ca

Abstract. A new genetic algorithm for switchbox routing in the physical design process of integrated circuits is presented. Our algorithm, called GASBOR, is based on a three-dimensional representation of the switchbox and problem-specific genetic operators. The performance of the algorithm is tested on different benchmarks and it is shown that the results obtained using the proposed algorithm are either qualitatively similar to or better than the best published results.

1 Introduction

Routing is one of the major tasks in the physical design process of integrated circuits. Pins that belong to the same signal net are connected with each other subject to a set of routing constraints during the routing procedure.

A switchbox is a rectangular routing region of fixed size on an integrated circuit. A switchbox contains pins located on all four boundaries. An example of a simple switchbox with a possible routing solution is shown in Figure 1.

The problem of switchbox routing is twofold: (1) to determine if a valid routing exists within the boundaries of the switchbox, and if so, (2) to optimize the routing structure according to certain quality factors. Both problems are considered as NP-complete. Although different algorithms have been proposed over the years (e.g., [1]-[4], [7], [10], [11]), the problem of finding a globally optimized routing solution of a switchbox is still open.

Genetic algorithms (GAs) are a new class of heuristic search methods based on the biological evolution model. During the last few years, GAs have been applied more and more successfully to find good heuristic solutions to NP-complete optimization problems [6].

To our knowledge, only two papers have been published in which strategies derived from the concept of GAs are applied to switchbox routing [3],[10]. The router in [3] combines the so-called steepest descent method with features of GAs. The crossover operator, however, is restricted to the exchanging of entire nets and the mutation procedure performs only the creation of new initial individuals. In [10], a rip-up-and-rerouter that is based on a probabilistic rerouting of nets of one routing structure is presented. However, the routing is carried out

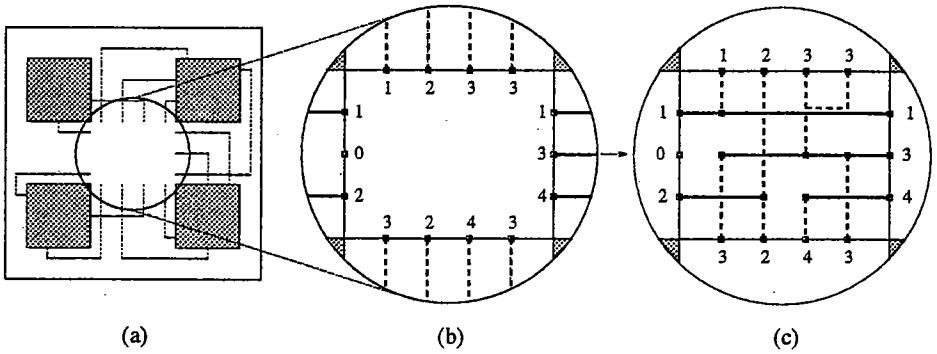

Fig. 1. An example of an integrated circuit (a) with a switchbox routing problem (b), and a possible routing solution of the switchbox (c). Dashed lines represent interconnections on the first layer, and solid lines represent interconnections on the second layer.

by a deterministic Lee algorithm [8]. Furthermore, main components of GAs, such as the crossover of different individuals, are not applied.

In this paper, we present a new GA for switchbox routing, called GASBOR, that is fundamentally different from the above mentioned approaches. First, our algorithm assumes that the switchbox is extendable in both directions. Subsequently, these extensions are reduced with the goal to reach the fixed size of the switchbox. Second, our algorithm is based on a problem-specific lattice-like representation of the switchbox instead of the commonly used binary string representation scheme in traditional GAs. Third, all routing structures are created randomly with a special random, rather than deterministic, routing strategy. And fourth, in contrast to [3], where the crossover operator exchanges only entire nets, our genetic operators work on the lowest level of a routing structure, namely, its grid points.

The contributions of this paper are:

- To formulate a GA that is capable of handling the routing problem of switchboxes.
- To compare the performance of our algorithm with previous approaches and describe the advantages and limitations of our strategy.
- To provide suggestions for other applications of GAs in the physical layout design of integrated circuits.

2 Problem Description

The switchbox routing problem is defined as follows. Consider a rectangular routing region, called *switchbox*, with a number of *pins* located on all four boundaries.

The pins that belong to the same net have to be connected subject to certain constraints and quality factors. The connection has to be made inside the boundaries of the switchbox on a symbolic routing area consisting of horizontal *rows* and vertical *columns*.

The interconnections are associated with the following constraints:

- Two layers are available for routing (see Figure 1).
- A net may change from one layer to the other one using a contact window called *via*.
- Different nets cannot cross each other on the same layer and must respect a minimum distance.
- The perimeter of the switchbox is not used for routing.

Three quality factors are used in this work to assess the quality of the routing result:

- Routing area
  The routing area, expressed as the number of rows and columns, is minimized during the evolutionary process until it reaches the fixed size of the switchbox. A final size larger than the fixed size means an unroutable solution.
- Net length
  The shorter the length of the interconnection nets the smaller the propagation delay.
- Number of vias
  The fewer the number of vias the better the routing quality.

3 Description of the Algorithm

3.1 Survey

GAs, in general, carry out optimization by simulating biological evolutionary processes. A population of individuals representing different problem solutions is subjected to genetic operators, such as, selection, crossover and mutation that are derived from the model of evolution. Using these operators the individuals are steadily improved over many generations and eventually the best individual resulting from this process is presented as the best solution to the problem.

An overview of the GA presented in this paper is shown in Figure 2. The number of individuals $|\mathcal{P}_c|$ is kept constant throughout all generations. Our mutation operator is applied after the reduction procedure, i.e., the modifications caused by the mutation operator remain "unpunished" in the population during the next mate selection and crossover procedure. This separation of the crossover and mutation procedures improves the ability of our approach to overcome local optima. Since the mutation operator has access to all individuals, the best individual is saved in each generation before the mutation operator is applied. At the end of the algorithm, the best individual p_{best} that has ever existed constitutes our final routing solution.

```
create initial population (P_c)
fitness_calculation (P_c)
p_best = best_individual (P_c)
for generation = 1 until max_generation
  P_n = ∅
  for offspring = 1 until max_descendant
    p_α = selection (P_c)
    p_β = selection (P_c)
    P_n = P_n ∪ crossover (p_α, p_β)
  endfor
  fitness_calculation (P_n)
  P_c = reduction (P_c ∪ P_n)
  p_best = best_individual (p_best ∪ P_c)
  mutation (P_c)
  fitness_calculation (P_c)
endfor
routing solution = best_individual (p_best ∪ P_c)
```

Fig. 2. Outline of the algorithm.

3.2 Creation of an Initial Population

The initial population is constructed from randomly created individuals.

First, each of these individuals is assigned a random initial row number y_{ind} and a random initial column number x_{ind}.

Let $\mathcal{S} = \{s_1, ...s_i, ...s_k\}$ be the set of all pins of the switchbox which are not connected yet and let $\mathcal{T} = \{t_1, ...t_j, ...t_l\}$ be the set of all pins having at least one connection to another pin. Initially $\mathcal{T} = \emptyset$. A pin $s_i \in \mathcal{S}$ is chosen randomly among all elements in $\mathcal{S}$. If $\mathcal{T}$ contains pins $\{t_u, ...t_j, ...t_v\}$ (with $1 \leq u < v \leq l$) of the same net, a pin t_j is randomly selected among them. Otherwise a second pin of the same net is randomly chosen from $\mathcal{S}$ and transferred into $\mathcal{T}$. Both pins (s_i, t_j) are connected with a so-called "random routing". Then s_i is transferred into $\mathcal{T}$. The process continues with the next random selection of $s_i \in \mathcal{S}$ until $\mathcal{S} = \emptyset$.

The random routing of two points (s_i, t_j) is based on our random routing strategy proposed in [9]. Its main characteristic is a line-search algorithm with random positions of alternate horizontal and vertical extension lines. A random routing of (s_i, t_j) is illustrated in Figure 3.

The extension is stopped when

- the extension lines of both points meet each other on the same layer, or
- the extension lines of s_i touch a net point which is already connected with t_j as shown in Figure 3 (e) (or vice versa).

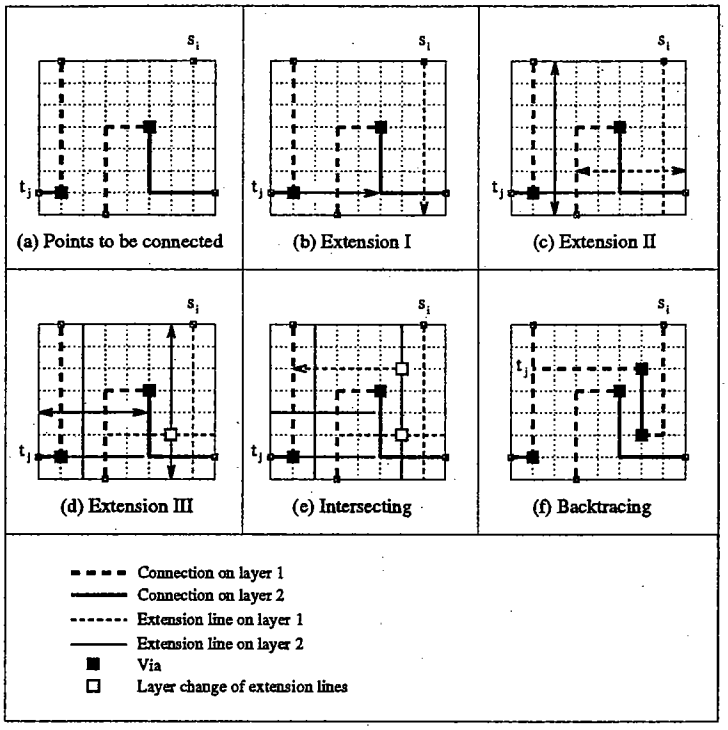

Fig. 3. Random routing of (s_i, t_j).

In the latter case, t_j (or s_i) is replaced with this meeting point (see Figure 3 (f)).

If the creation of extension lines does not succeed in one of these conditions within a certain number of iterations, all extension lines are erased and the switchbox is extended either with an additional row on a random position y_{add} with $1 \leq y_{add} \leq y_{ind}$ or an additional column on a random position x_{add} with $1 \leq x_{add} \leq x_{ind}$. After adjusting all previous routed nets to this new row and column, respectively, s_i and t_j undergo a new attempt to connect them with randomly created extension lines.

Tracing the shortest path on the extension lines between s_i and t_j (see Figure 3 (e,f)) avoids unnecessary loops in the connection without sacrificing the randomness of the resulting routing path.

The creation of the initial population is finished when the number of completely routed switchboxes is equal to the population size $|\mathcal{P}_c|$. As a consequence of our strategy, these initial individuals are quite different from each other and scattered all over the search space.

3.3 Calculation of Fitness

The fitness $F(p)$ of each individual $p \in \mathcal{P}$ is calculated to assess the quality of its routing structure relative to the rest of the population $\mathcal{P}$. The selection of the mates for crossover and the selection of individuals which are transferred into the next generation are based on these fitness values. The following fitness calculation is an extension of our approach in [9] that was developed for channel routing of integrated circuits.

First, two functions $F_1(p)$ and $F_2(p)$ are calculated for each individual $p \in \mathcal{P}$ according to equations (1) and (2).

$$F_1(p) = \frac{1}{n_{row} + n_{column}} \tag{1}$$

where n_{row} = number of rows of p and
n_{column} = number of columns of p.

$$F_2(p) = \frac{1}{\sum_{i=1}^{n_{ind}} (l_{acc}(i) + a * l_{opp}(i)) + b * v_{ind}} \tag{2}$$

where $l_{acc}(i)$ = net length of net i of net segments according to the preferred direction of the layer,
$l_{opp}(i)$ = net length of net i of net segments opposite to the preferred direction of the layer,
a = cost factor for the preferred direction,
n_{ind} = number of nets of individual p,
v_{ind} = number of vias of individual p and
b = cost factor for vias.

In order to assure that the area minimization, i.e. the sum of rows and columns of p, predominates the net length and the number of vias, the fitness $F(p)$ is derived from $F_1(p)$ and $F_2(p)$ as follows:

Assume that $(p_i, ...p_x, ...p_j)$ are individuals with the same area, i.e. the same value $F_1(p)$. These individuals are arranged in an ascending order according to $F_2(p)$. Then p_i is the individual with the lowest value $F_2(p)$ in this group ("worst individual with this area"). Its fitness value $F(p_i)$ is defined by

$$F(p_i) = F_1(p_i). \tag{3}$$

The individual p_j has the highest value $F_2(p)$ in this group ("best individual

with this area"). Let $F_1(p_{j+1})$ be the F_1-value of the next ("better") group with one row or column less. The fitness $F(p_j)$ is calculated as follows:

$$F(p_j) = F_1(p_{j+1}) - \frac{\Delta F_1}{j - i + 1} \tag{4}$$

where $\Delta F_1 = F_1(p_{j+1}) - F_1(p_j)$.

Now $F(p_x)$ of the remaining individuals of this group can be calculated relative to their F_2-values between the lower bound $F(p_i)$ and the upper bound $F(p_j)$:

$$F(p_x) = F(p_j) - \frac{\Delta F * (F_2(p_j) - F_2(p_x))}{\Delta F_2} \tag{5}$$

where $\Delta F = F(p_j) - F(p_i)$ and
$\Delta F_2 = F_2(p_j) - F_2(p_i)$.

After the evaluation of $F(p)$ for all individuals of the population $\mathcal{P}$ these values are scaled linearly as described in [5], in order to control the variance of fitness in the population.

3.4 Selection Strategy

The selection strategy is responsible for choosing the mates among the individuals of the population $\mathcal{P}_c$.

According to the terminology of [5], our selection strategy is stochastic sampling with replacement. That means any individual $p_i \in \mathcal{P}_c$ is selected with a probability

$$\frac{F(p_i)}{\sum_{p \in \mathcal{P}_c} F(p)}$$

The two mates needed for one crossover are chosen independently of each other. An individual may be selected any number of times in the same generation.

3.5 Crossover Operator

During the crossover, two individuals are combined to create a descendant. We developed a crossover operator that gives compact, high-quality routing structures of these two individuals an increased probability to be transferred intact to their descendant. Let p_α and p_β be copies of the mates (Figure 4 (a,b)) and p_γ be their descendant.

First, the direction of a cut line (vertical cut column x_c or horizontal cut row y_c) is randomly chosen [1].

From the pin occupation list of the vertical boundaries of the switchbox a pin combination (located on the same horizontal row) is randomly chosen to determine the position of the cut row in each parent p_α and p_β.

The individual p_α transfers its routing structure to p_γ which is

- located on (x_α, y_α, z) with $1 \leq x_\alpha \leq x_{ind\alpha}$ ($x_{ind\alpha}$ = number of columns of p_α), $1 \leq y_\alpha \leq y_{c\alpha}$ ($y_{c\alpha}$ = position of the cut row in p_α), $1 \leq z \leq 2$ and
- not cut by the cut row $y_{c\alpha}$.

Accordingly, p_β transfers to p_γ the uncut connections located on (x_β, y_β, z) with $1 \leq x_\beta \leq x_{ind\beta}$, $y_{c\beta} < y_\beta \leq y_{ind\beta}$ and $1 \leq z \leq 2$ (see Figure 4 (c,d)).

Note that the connections of p_α and p_β cut by $y_{c\alpha}$ and $y_{c\beta}$ are traced until their next Steiner point or pin is reached and not transferred into p_γ.

Assume that the parts of p_α and p_β which have to be transferred into p_γ contain columns not occupied by any vertical segments anymore. Then the number of columns of p_α and p_β are decreased by deleting these empty columns and the sizes of p_α and p_β are shrunk accordingly.

The initial number of columns $x_{ind\gamma}$ of p_γ is obtained by extending p_α and p_β with additional columns to achieve a vertical accordance of pins with the pin occupation list of the switchbox [2].

The routing of the open connections in p_γ is done as follows: Let $\mathcal{N}_\alpha$ be the set of all Steiner points or pins which are end points of a cut segment in p_α. Accordingly, let $\mathcal{N}_\beta$ be the set of these points in p_β. If $\mathcal{N}_\alpha$ contains more than one point of the same net, these points are connected with each other in a random order by our random routing strategy (see Section 3.2). Except for one randomly chosen point, all points of this net in $\mathcal{N}_\alpha$ are now deleted. The same "inner routing" in $\mathcal{N}_\beta$ is performed. As a result, $\mathcal{N}_\alpha$ and $\mathcal{N}_\beta$ do not contain more than one point per net. These points in $\mathcal{N}_\alpha$ are now selected randomly and compared with all points in $\mathcal{N}_\beta$. If a point of the same net is found in $\mathcal{N}_\beta$, both points are connected by means of our random routing (see Figure 4 (e,f)).

If the random routing of two points does not lead to a connection within a certain number of extension lines, the extension lines are deleted and the switchbox is extended at a random position x_{add} with $1 \leq x_{add} \leq x_{ind\gamma}$. If the repeated extension of the switchbox also does not enable a connection, p_γ is deleted entirely and the crossover process starts again with a new random cut row y_c (or cut column x_c) applied to p_α and p_β.

---

[1] In the following explanation of the crossover operator, we assume the direction of the cut line to be horizontal. The crossover procedure resulting from a vertical cut line can be obtained by exchanging the variable x with y and vice versa and the term "row" with "column" and vice versa.

[2] Pins of the horizontal switchbox boundaries which are placed opposite to each other on the same column in the pin occupation list have also to be located in the same manner in p_γ.

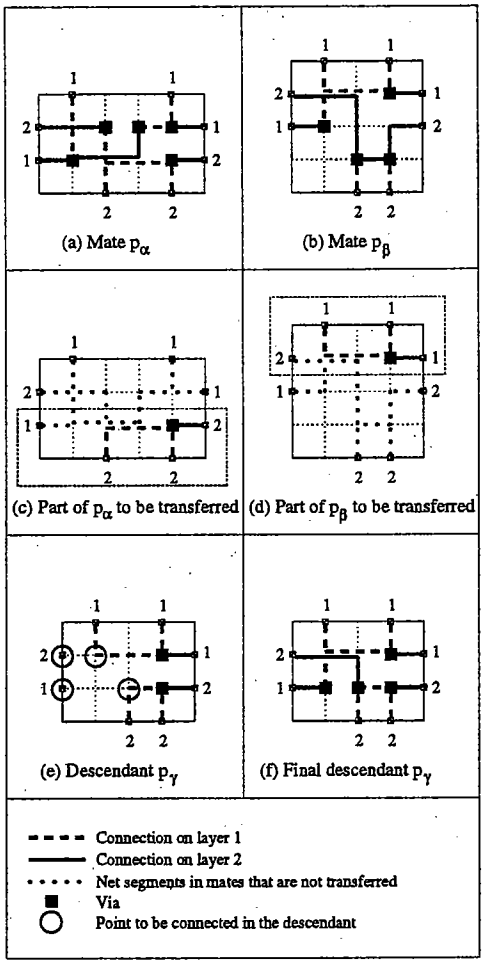

Fig. 4. Crossover of (p_α, p_β) to p_γ.

The crossover process of creating p_γ is finished with deleting all columns and rows in p_γ that are not used for any routing segment.

3.6 Reduction Strategy

Because the population size of a genetic algorithm should be constant, a reduction strategy is necessary to decide which individuals among the current population $\mathcal{P}_c$ and the set of descendants $\mathcal{P}_n$ should survive for the next generation.

Our reduction strategy simply chooses the $|\mathcal{P}_c|$ fittest individuals of $(\mathcal{P}_c \cup \mathcal{P}_n)$ to survive as $\mathcal{P}_c$ into the next generation.

3.7 Mutation Operator

Mutation operators perform random modifications on an individual. The purpose is to overcome local optima and to exploit new regions of the search space.

Our mutation operator works as follows. Define a surrounding rectangle with random sizes (x_r, y_r) around a random centre position (x, y, z). All routing structures inside this rectangle are deleted. The remaining net points on the edges of this rectangle are now connected again in a random order with our random routing strategy.

4 Implementation and Experimental Results

The algorithm, called GASBOR, has been implemented in FORTRAN on a SPARC workstation. The approximate size of the source code is 10,000 lines.

4.1 Measurement Conditions

The performance of the algorithm has been tested on different benchmarks. The routing results of the benchmarks, presented later, are the best results obtained in 10 consecutive executions of the algorithm for each benchmark. All executions are based on an arbitrary initialization of the random number generator. We terminated our program when there was no improvement to the best individual in 100 consecutive generations.

The values of the other parameters are as follows:

$\|\mathcal{P}_c\|$	= 50
max_descendant	= 30
a	= 1.01 (Equation (2))
b	= 2.00 (Equation (2))
Mutation probability	= 0.002

The same parameter setting is used for all benchmarks.

4.2 Switchbox Routing Results

The routing results of GASBOR for different benchmarks are presented in Table 1. It can be seen that our results are qualitatively similar to or better than the best known results from popular switchbox routers published for these benchmarks.

The detailed results of the 10 executions of GASBOR for each benchmark are presented in Table 2. It must be noted, that the diversity of the routing results of each of the benchmarks is caused by our decision to terminate the program when there is no improvement to the best individual in 100 consecutive generations. As we shall see later, the best results of Table 1 can always be achieved if such a strict abortion criterion is avoided.

In [7, Fig. 6-17], the knowledge-based expert system router "WEAVER" was able to route the so-called switchbox Joo6_17 which is provably unroutable by

Benchmark	System	Rows	Col.	Netlength	Vias
Simple switchbox	Lee [7]	7	7	60	11
	WEAVER [7]	7	7	60	4
	GASBOR	7	7	60	4
Joo6_17	WEAVER [7]	9	11	166	19
	SILK [10]	9	11	166	18
	GASBOR	9	11	164	18
Pedagogical switchbox	BEAVER [2]	16	15	396	38
	PACKER [4]	16	15	406	45
	SAR [1]	16	15	393	31
	GASBOR	16	15	395	31
Dense switchbox	WEAVER [7]	17	16	517	31
	Mighty [11]	18[a]	16	530	32
	SILK [10]	17	16	516	29
	Monreale [3]	18[a]	16	529	32
	SAR [1]	17	16	519	31
	GASBOR	17	16	519	29
Judy	Monreale [3]	17	17	506	32
	GASBOR	17	17	498	32

[a] Additional row at the bottom edge of the switchbox.

Table 1: Benchmark results.

Bench-mark	Generations [a]			Routing result[b]			Run-time[c]
	Min	Max	Avg	Worst	Best	Avg	
Simple switchbox	51	115	69	7/7/ 60/4	7/7/ 60/4	7/7/ 60/4	3
Joo6_17	96	681	398	9/11/ 168/19	9/11/ 164/18	9/11/ 165/18	41
Peda-gogical switchbox	181	1002	517	16/15/ 404/38	16/15/ 395/31	16/15/ 397/32	101
Dense switchbox	198	1192	327	18/16/ 529/32	17/16/ 519/29	17/16/ 520/31	97
Judy	161	998	316	17/17/ 506/32	17/17/ 498/32	17/17/ 500/32	78

[a] Number of generations when the best individual appears in the population.
[b] Routing results given as rows/columns/netlengh/vias.
[c] Average CPU runtime in minutes.

Table 2: Detailed results of the 10 executions of each benchmark.

traditional algorithms. As is evident from Table 1, our algorithm yields a better result than both WEAVER and SILK for this benchmark. Figure 5 shows our routing solution.

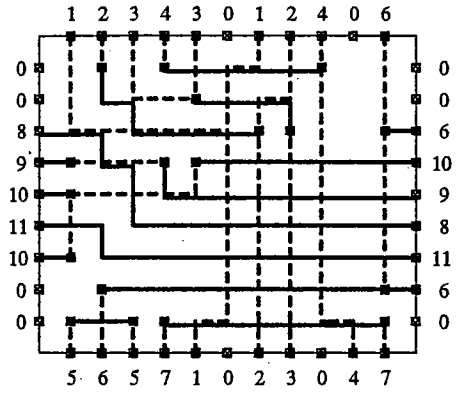

Fig. 5. Our routing solution of Joo6_17.

4.3 Diversity Within the Population

A sufficient population diversity, especially in an advanced stage of the algorithm, is one of the main "secrets" behind a successful GA. Our operators are such that they avoid too much convergence pressure and guarantee sufficient diversity within the population as can be seen in Figure 6.

4.4 Effect of Random Number Generator

Since the methodology of our algorithm is probabilistic, it is important to investigate the effect of the initialization of the random number generator on the routing results.

We conducted an experiment regarding the robustness of our algorithm, i.e., we investigated the impact of the seed of the random number generator on the routing results. The effect of the different initializations of the random number generator on the number of generations needed to achieve our best results was also investigated. We executed our program 1000 times with different initializations of the random number generator to route the switchbox Joo6_17. As soon as the algorithm produced an individual equal to the best one of Table 1, the execution was stopped and the number of generations was noted. The distribution of this number of generations in 1000 program executions is plotted in Figure 7.

Similar results were reached using the other benchmarks of Table 1. At this point we noticed a direct relationship between the complexity of the switchbox

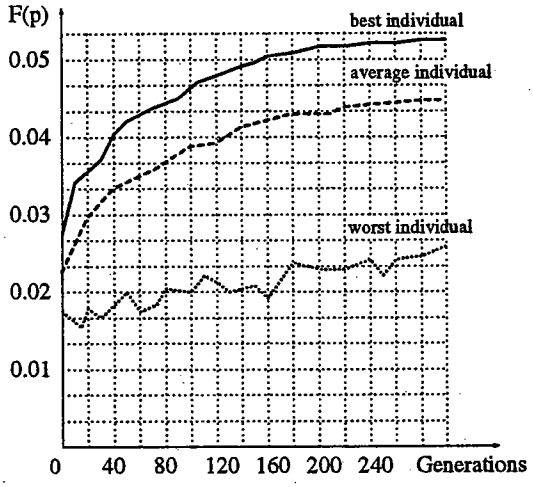

Fig. 6. Average convergence behavior of the individuals for the switchbox Joo6_17 in 10 program executions.

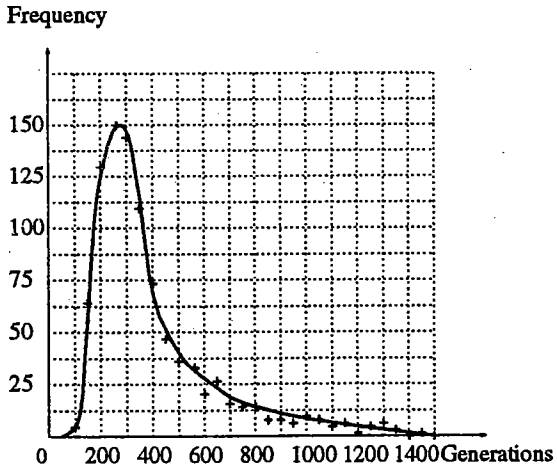

Fig. 7. Distribution of the number of generations needed to achieve the best result of Burstein's difficult channel.

routing problem and the shape of the curve: The more complex the routing structure, the flatter is the curve and the more the curve is shifted towards a higher number of generations.

From the experiment we conclude that the initialization of the random number generator affects only the runtime. Our routing results can be achieved with any initial seed of the random number generator.

5 Summary

We presented a GA that is capable of handling the routing problem of switchboxes. Due to its ability to overcome local optima, our GA always achieves the same results for a given routing problem independently of the initialization of the random number generator. Thus, it can be considered as robust.

As can be seen from Table 2, the runtimes of our algorithm on some benchmarks may be unacceptable. However, due to the inherent parallelism of GAs we are optimistic about reducing the runtimes by implementing a parallel version of our algorithm.

Further investigations are needed to measure the performance of the algorithm for larger switchbox problems. Initial examination suggests that an exponential relationship exists between the CPU runtime and the size of the switchbox routing problem.

In developing this algorithm our suggestions for other applications of GAs in the physical design process of integrated circuits are:

- The representation scheme of a layout problem in a GA should be a three-dimensional, problem-specific representation rather than a one-dimensional binary string (as is common in traditional applications of GAs [5]). Our scheme ensures that high quality parts of the layout structure are preserved as high-fitness building blocks and transferred intact with an increased probability in the next generation.
- The genetic operators of a GA in the physical layout design should be adapted to the specific layout problem.
- A sufficient diversity within the population (including the initial population) is crucial to the robustness of a GA for a design problem.
- The ability to overcome local optima is improved by separation of the crossover and the mutation procedures.

From our results we believe that genetic algorithms are promising tools for solving optimization problems in the physical design process of integrated circuits.

References

1. Acan, A., Ünver, Z.: Switchbox Routing by Simulated Annealing: SAR. IEEE International Symposium on Circuits and Systems 4 (1992) 1985-1988

2. Cohoon, J. P., Heck, P. L.: BEAVER: A Computational-Geometry-Based Tool for Switchbox Routing. IEEE Trans. on Computer-Aided Design **7** No. 6 (1988) 684-697
3. Geraci, M., Orlando, P., Sorbello, F., Vasallo, G.: A Genetic Algorithm for the Routing of VLSI Circuits. Euro Asic '91, Parigi 27-31 Maggio (1991) 218-223
4. Gerez, S. H., Herrmann, O. E.: Switchbox Routing by Stepwise Reshaping. IEEE Trans. on Computer-Aided Design **8** No. 12 (1989) 1350-1361
5. Goldberg, D. E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Reading, MA: Addison-Wesley Publishing Company (1989)
6. Goldberg, D. E.: Genetic and Evolutionary Algorithms Come of Age. Communications of the Association for Computing Machinery (CACM) **37** No. 3 (1994) 113-119
7. Joobbani, R.: An Artificial Intelligence Approach to VLSI Routing. Boston, MA: Kluwer Academic Publishers (1986)
8. Lee, C. Y.: An Algorithm for Path Connections and its Applications. IRE-Transactions on Electronic Computers (1961) 346-365
9. Lienig, J., Thulasiraman, K.: A Genetic Algorithm for Channel Routing in VLSI Circuits. Evolutionary Computation **1** No. 4 (1994) 293-311
10. Lin, Y.-L., Hsu, Y.-C., Tsai, F.-S.: SILK: A Simulated Evolution Router. IEEE Trans. on Computer-Aided Design **8** No. 10 (1989) 1108-1114
11. Shin, H., Sangiovanni-Vincentelli, A.: A Detailed Router Based on Incremental Routing Modifications: Mighty. IEEE Trans. on Computer-Aided Design **6** No. 6 (1987) 942-955

This article was processed using the LaTeX macro package with LLNCS style