A critical step in VLSI design (perhaps the most critical) is the definition of a detailed routing philosophy and procedure. The design task, simply put, consists of solving the problem of how to connect a well defined set of pins in a multilayer rectilinear area subject to well defined design rules and minimizing well defined quality factor. The problem has been shown to be NP-complete [SZY85]; that is to say, there is no known deterministic algorithm capable of solving it in a polynomial time. Therefore despite the number and variety of perspective approaches suggested, the search for a routing solution closest to the optimal solution is still on and the problem is still open.

The present work implements a new channel router algorithm (Monreale) based on a modified Genetic Algorithm (GA). The distinctive feature of the algorithm is a well balanced combination of 1) the fast performance characterizing the steepest descent method and of 2) the high level of parallelism and of local minimum trapping avoidance characterizing the GA approach.

The Monreale approach, pretty much like the traditional GA approach, starts out with a simulation of population evolution of individuals (each being 100% channel routed implementation) subject to a well defined set of rules in the presence of rules violations. The Monreale algorithm however in a substantial departure to the traditional GA approach (where new individuals are generated by combining genetic information of two individuals) permits in a mutation mode, new individuals to be generated by genetic contributions of all individuals.

Monreale has automatically and completely routed several difficult benchmarks problems present in literature (including the Burstein's difficult channel with only three rows and one empty column); performances on benchmarks are better than or comparable to the most popular channel routers.

Since channel routing is an NP-complete problem, no deterministic algorithm solution in a polynomial time is known. A non deterministic algorithm that approaches the routing problem as a general combinatorial optimization problem may get close to a global optimum solution in a reasonable time.

Monreale is a non deterministic algorithm that derives from GA and maintains its best features such as the intrinsic high level of parallelism and the avoidance of local minima trapping. We have modified in an original manner the GA's procedures, as described in the next section, using a technique of "steepest descent".

The main advantages of Monreale's algorithm are:

A) Effective and efficient search procedure in providing a self-modifying subset of the hyperspace of all possible nets. This quality enhances the probability of finding a solution close to a global optimum.

B) Simple implementation with respect to traditional methods.

C) Absence of ad hoc assumptions about the search space such as continuity, existence of derivatives etc.

D) Steepest descent methodology leading promptly towards high quality factor regions.
E) Finite probability of moving from a solution to a worse one and consideration of several points of search space at the same time therefore providing ease of escape from a local optima solutions.

F) Intrinsic high grade of parallelism equating the number of individuals used.

4. A CHANNEL ROUTER BASED ON GENETIC ALGORITHMS.

4.1 SOME REMARKS ON GENETIC ALGORITHMS

GA simulate the evolution of a population of individuals each of which represents a candidate solution to the problem. Starting from a randomly generated set of individuals, a new set is found by collecting information from the old ones.

The main procedures in GA are: initialization, evaluation, selection, cross-over and mutation.

4.1.1 Procedure Initialization

In GA the search starts by creating an initial random population containing a wide variety of individuals. An individual is constituted by an ordered sequence of L "genes". Each gene is coded as a finite length string over some finite alphabet.

4.1.2 Procedure Evaluation

After a population has been created, the quality of each individual is evaluated, according to some criteria, in order to know how it fits the environment, that is in order to know the value of quality factor in the particular point of search space represented by the individual.

4.1.3 Procedure Selection

The selection procedure works, by choosing in analogy with natural selection, two individuals to carry out evolution. Different selection criteria may be implemented; generally higher quality individuals have higher probabilities of being selected.

4.1.4 Procedure Cross-over

Cross-over procedure creates new individuals by combining genes of the two selected individuals. A simple cross-over may proceed in this way: an integer number K is randomly selected between 1 and L-1, a new individual is created with the genes between positions 1 and K of the first old individual and with the genes between positions K+1 and L of the second old individual.

The selection and cross-over procedure provides more likely for new individuals to inherit higher quality genetic codes from the previous generation. Thus the action of selection and cross-over, has a high probability of leading into zones of search space with higher quality factor.

4.1.5 Procedure Mutation

Mutation procedure randomly changes the genes of some individuals to explore new points of search space.

Mutation is needed because selection and cross-over may exclude from new generations some potentially useful genes belonging to individuals with low quality factor. The mutation procedure protects against such premature loss of important information.

4.2. THE MONREALE APPROACH

Monreale is a channel router implemented with an original and efficient algorithm that combines the speed of gradient descent method with the best features of GA.

In Monreale's approach genes are nets and an individual is a 100% routing where violations to the design rule a) (see par. 2.) may be present.

Initialization and Evaluation procedures are similar to the previously presented while Selection, Cross-over and Mutation are quite different.

The router works as follows: an initial random population is created by respecting some heuristics. Individuals of the new population are generated by a new procedure called Cross-gene and by a new way of using mutation procedure. Most individuals are generated by Cross-gene that substitutes classical Selection and Cross-over procedures; the remaining individuals of the population are randomly generated by mutation, here used to construct a whole individual instead of modifying a single gene. Evolution is performed until a routing without violations of design rule a) (see par. 2.) and with the desired quality factor is generated.

A description of the procedures used follows.

4.2.1 Procedure Initialization

The initialization procedure creates the initial population of 100% routings. Each net is randomly created independently from the others, therefore violations to design rule a) (see par. 2.) will be present.

In order to reduce the search space some problem dependent heuristics are introduced; only some kind of nets are considered, the most useful to optimize the wire length, the number of vias and the chip area consumption. Some characteristics of used nets are reported below.

A net can be traced only inside the rectangle delimited by its own terminals.

Different layers do not have reserved directions.

Nets with many horizontal segments, referred to as 'dogleg', are allowed.

Vias in adjacent grid positions are not allowed.

The shape of a net may be influenced by other nets' terminal position.
4.2.2 Procedure Evaluation
The evaluation procedure computes the quality factors of routings generated. The object function used is:

\[ Q = \alpha \cdot (\# \text{ violations}) + \beta \cdot (\# \text{ vias}) + \gamma \cdot (\text{wire length}) \]

where \( \alpha, \beta \) and \( \gamma \) are constants.

4.2.3 Procedure Cross-gene
To generate a new routing first we choose randomly a permutation of the integers between 1 and \( L \), where \( L \) is the number of nets constituting a routing. This permutation represents the placing order of nets in the new routing.

Each net is chosen by searching in a randomly ordered set of old individuals the first net without rule violation, with respect to the already placed ones. If such a net does not exist the better fitting one is chosen. This step is iterated until the major fraction of the new population is generated.

4.2.4 Procedure Mutation
Mutation procedure randomly generates some routings using the same routines as the initialization procedure.

4.2.5 Monreale's topics
The efficiency of the proposed algorithm is largely due to the following two differences with respect to traditional GA:

I) Traditional GA find higher quality factor zones by randomly splitting and combining the genetic code of two of the best individuals. Monreale's search is more effectively guided by choosing, net by net, the best fitting in the current routing, therefore implementing a steepest descent technique.

II) Monreale's Mutation procedure provides a collection of new generated nets that may be used by Crossgene procedure to construct new routings. Therefore a new generated net is put in the routing, where is best fit, and not in a routing randomly selected such as it would be implemented by traditional GA.

The pseudo code description of Monreale's algorithm is shown in Fig.1.

5. EXPERIMENTS
We have implemented the algorithm in about 2000 lines of Pascal programming language on a VAX-6000/420 under VMS operating system.

We have tested Monreale with several standard bench-marks found in literature.

In this section we present and compare Monreale's results versus the most popular channel routers. The following set of constants was used:

| number of individuals in a population | 60 |
| percentage of random individuals in a population | 30% |
| value of \( \alpha \) in the object function | 50 |
| value of \( \beta \) in the object function | 1 |
| value of \( \gamma \) in the object function | 1 |

5.1 CHANNEL RESULTS
Fig.2 shows tables of comparison with respect to several bench-marks. As may be seen, in all cases Monreale's results are qualitatively better than or comparable to the most popular channel routers. Table I shows the data relative to the solution of the Burstein's difficult channel. In 1986 this channel was routed with five tracks and two empty columns in the middle of the channel [SPY89]. Subsequent Mighty [SHI87] was able to solve it with four tracks and only one empty column and Packer [GER89] with four tracks and no empty columns. Monreale's approach is able to solve Burstein's problem either with four tracks and no empty columns (see Fig.3) as Packer does, or with three tracks and one empty column; this solution, to our knowledge, represent the minimum area usage and wire length obtained to date (see Fig.4).
TABLE I
Routing of Burstein’s difficult channel

<table>
<thead>
<tr>
<th>Router</th>
<th>#Rows</th>
<th>#Cols</th>
<th>#Vias</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mighty</td>
<td>4</td>
<td>13</td>
<td>8</td>
</tr>
<tr>
<td>Packer*</td>
<td>4</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>Monreale*</td>
<td>4</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>Monreale</td>
<td>3</td>
<td>13</td>
<td>12</td>
</tr>
</tbody>
</table>

* with no empty column

TABLE II
Routing of Joo_6_16 channel

<table>
<thead>
<tr>
<th>Router</th>
<th>#Rows</th>
<th>#Cols</th>
<th>#Vias</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weaver</td>
<td>8</td>
<td>11</td>
<td>22</td>
</tr>
<tr>
<td>Weaver*</td>
<td>7</td>
<td>11</td>
<td>21</td>
</tr>
<tr>
<td>Monreale</td>
<td>7</td>
<td>11</td>
<td>20</td>
</tr>
</tbody>
</table>

* with the help of user

TABLE III
Routing of Joo_6_13

<table>
<thead>
<tr>
<th>Router</th>
<th>#Rows</th>
<th>#Cols</th>
<th>#Vias</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weaver</td>
<td>7</td>
<td>18</td>
<td>29</td>
</tr>
<tr>
<td>Silk</td>
<td>6</td>
<td>18</td>
<td>28</td>
</tr>
<tr>
<td>Packer</td>
<td>6</td>
<td>18</td>
<td>25</td>
</tr>
<tr>
<td>Monreale</td>
<td>6</td>
<td>18</td>
<td>29</td>
</tr>
</tbody>
</table>

TABLE IV
Routing of Joo_6_12 channel

<table>
<thead>
<tr>
<th>Router</th>
<th>#Rows</th>
<th>#Cols</th>
<th>#Vias</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weaver</td>
<td>4</td>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>Packer</td>
<td>4</td>
<td>12</td>
<td>18</td>
</tr>
<tr>
<td>Monreale</td>
<td>4</td>
<td>12</td>
<td>14</td>
</tr>
</tbody>
</table>

TABLE V
Routing of Yosh. Kuh channel

<table>
<thead>
<tr>
<th>Router</th>
<th>#Rows</th>
<th>#Cols</th>
<th>#Vias</th>
</tr>
</thead>
<tbody>
<tr>
<td>Josh. kuh</td>
<td>5</td>
<td>12</td>
<td>21</td>
</tr>
<tr>
<td>Weaver</td>
<td>4</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>Monreale</td>
<td>4</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>

Fig. 2: Table of comparison with respect to several benchmarks.

Fig. 3: The Burstein’s channel routed by Monreale with four rows

Fig. 4: The Burstein’s channel routed by Monreale with three rows and one empty column

Fig. 5: The JOQ_6_16 channel routed by Monreale with no help of the user

Fig. 6 Qualitative behaviour through the optimum of the object function Q (Burstein’s channel).
- Effective and efficient search.
- Simplicity of implementation.
- No restriction on search space.
- Gradient descent speed.
- Local minima escape.
- Intrinsic high parallelism.

Although results are already quite interesting, a parallelized version of our algorithm is being issued.

Moreover we are extending the foundation of the algorithm to solve the multi-layer and switch-box problem.

The algorithm seem to be suitable to solve a generic combinatorial optimization problem. In particular we are investigating the possibility of a global approach to the problem of placement and routing.

5.2 INFLUENCE OF RANDOM SEED

Since the initial routing is generated by using a random seed, it is important to establish how this may influence the final quality and the computational time.

We have run the proposed bench-marks by using 1000 distinct seeds and with 60 individuals for each population. Fig. 6 shows a typical results obtained running the Burnstein’s difficult channel with three rows and an empty column; the average number of generations needed to route the channel is 30; the corresponding CPU time is about 180 s. It is very important to observe that all the 1000 runs complete the routing for all tests.

6. FUTURE DEVELOPMENTS AND CONCLUSIONS.

We have presented a new approach to solve the channel routing problem using a modified simulated evolution technique and we have tested this approach by well known bench-marks found in the literature. Its main features are summarized below:

REFERENCES


Fig.7: Distribution of the number of iterations needed to route the Burstein’s difficult channel