Journal of Circuits, Systems, and Computers
Vol. 21, No. 5 (2012) 1250041 (13 pages)
© World Scientific Publishing Company
DOI: 10.1142/S0218126612500417

INTEGRATED CIRCUIT CHANNEL ROUTING USING A PARETO-OPTIMAL GENETIC ALGORITHM*

THEODORE W. MANIKAS
Department of Computer Science and Engineering,
Southern Methodist University,
POB 750122 Dallas, Texas 75275-0122, USA
manikas@lyle.smu.edu

Received 20 July 2011
Accepted 4 January 2012
Published 30 August 2012

An important part of the integrated circuit design process is the channel routing stage, which determines how to interconnect components that are arranged in sets of rows. The channel routing problem has been shown to be NP-complete, thus this problem is often solved using genetic algorithms. The traditional objective for most channel routers is to minimize total area required to complete routing. However, another important objective is to minimize signal propagation delays in the circuit. This paper describes the development of a genetic channel routing algorithm that uses a Pareto-optimal approach to accommodate both objectives. When compared to the traditional channel routing approach, the new channel router produced layouts with decreased signal delay, while still minimizing routing area.

Keywords: Detailed routing; circuit layout; evolutionary algorithms; optimization.

1. Introduction

The integrated circuit design process transforms a system specification into a manufactured chip. An important part of the design process is the routing stage, which determines how to interconnect components on the chip, subject to certain constraints. Typically, the circuit components (*cells*) are arranged in sets of rows, separated by channels. In this case, the cells are interconnected with wires in the channels using a *channel router*. The signal inputs and outputs on each cell are called *terminals*, while each set of common terminals that are to be connected together is called a *net*. A *netlist* defines all the nets and their terminal connections for a given circuit.

Routing channels are assumed to be horizontal, with fixed connections (cell terminals) on the top and bottom of the channel, and open right and left ends. A channel can be formally described as two sequences of size C (*channel length*) that

*This paper was recommended by Regional Editor Eby Friedman.

describe the top (T) and bottom (B) connections in the channel: $T = \{t_1, t_2, \ldots, t_C\}$ and $B = \{b_1, b_2, \ldots, b_C\}$. For each sequence, t_i is the net connected to the top of column i, while b_i is the net connected to the bottom of column i. The netlist for a channel can be formally described as a sequence of size M (number of nets) that describes all the nets (N) to be routed in the channel: $N = \{n_1, n_2, \ldots, n_M\}$.

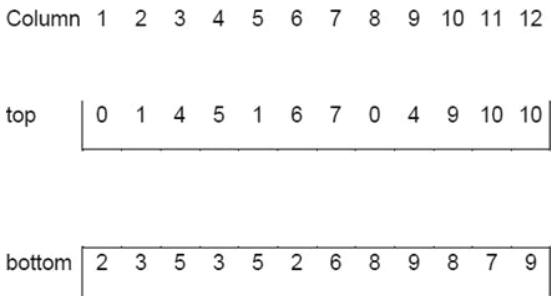
A routing channel example from Ref. 1 is shown in Fig. 1. This channel has 12 columns ($C = 12$) and 10 nets ($M = 10$). The netlist $N = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$. The top channel connections $T = \{0, 1, 4, 5, 1, 6, 7, 0, 4, 9, 10, 10\}$ and the bottom channel connections $B = \{2, 3, 5, 3, 5, 2, 6, 8, 9, 8, 7, 9\}$. A net name of "0" indicates that no net is connected to the specified channel terminal. The given sequence indicates that column 1 has a terminal connection to net 2 at the bottom of the channel, and no terminal connection at the top. Column 2 has a terminal connection to net 1 at the top of the channel, and terminal connection to net 3 at the bottom.

Traditional channel routing assumes two routing layers, with all horizontal wire segments routed on one layer and all vertical wire segments routed on the second layer, to avoid unintended electrical connections (*shorts*) between nets. Layers on the same net are connected using *vias.* The traditional goal of channel routers is to minimize the channel area, which means minimizing the channel height. This goal translates into minimizing the total number of horizontal tracks: each track contains one or more horizontal segments.

1.1. *Channel routing constraints*

There are two primary constraints that affect routing: *horizontal* and *vertical* constraints. A *horizontal constraint* exists between two nets j and k ($j \neq k$) if their horizontal ranges overlap. For our working example (Fig. 1), net 1 has the horizontal range (2, 5) since its leftmost terminal connection is in column 2 and its rightmost terminal connection is in column 5. Similarly, net 4 has the horizontal range (3, 9).

If nets 1 and 4 were routed in the same horizontal track, the horizontal wire segments for nets 1 and 4 would overlap and short. Therefore, a horizontal constraint exists between nets 1 and 4. For a given netlist and channel, horizontal constraints



Column 1 2 3 4 5 6 7 8 9 10 11 12
top 0 1 4 5 1 6 7 0 4 9 10 10
bottom 2 3 5 3 5 2 6 8 9 8 7 9

Fig. 1. Routing channel example.

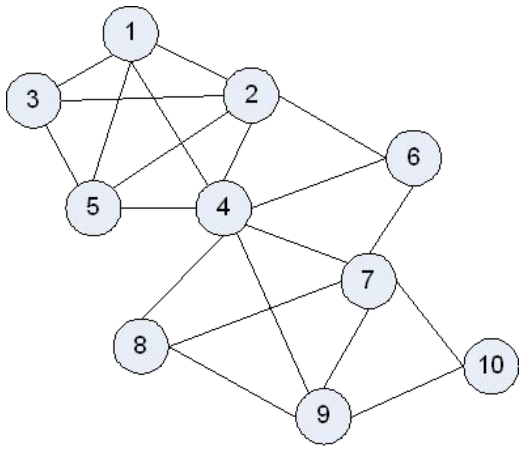


Fig. 2. Horizontal constraint graph for Fig. 1.

can be represented by a *horizontal constraint graph* $H(V, E)$, an undirected graph where $V =$ [set of all nets in the netlist] and $E =$ [horizontal constraints between nets]. The horizontal constraint graph for Fig. 1 is shown in Fig. 2.

Figure 3 illustrates the horizontal constraints for nets 8 and 9. Note that net 9 has a horizontal constraint with nets 8 and 10, but net 8 does not have a horizontal constraint with net 10. Therefore, the horizontal segments for nets 8 and 10 could theoretically be assigned to the same horizontal track. However, we also need to check if any vertical constraints exist between these nets.

A *vertical constraint* exists between two nets j and k ($j \neq k$) if their vertical ranges overlap. For our working example (Fig. 1), column 3 has top net connection 4 and bottom net connection 5. This means that the horizontal track containing net 4 *must* be routed *above* the horizontal track containing net 5; otherwise the vertical wire segments will overlap and short.

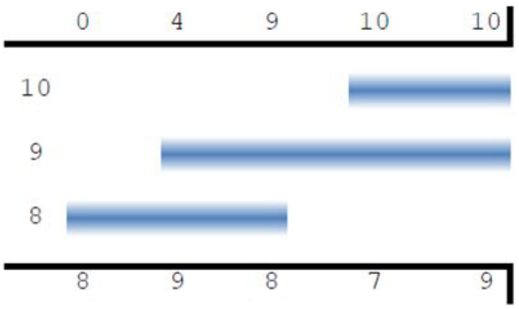


Fig. 3. Horizontal constraints for nets 8, 9 and 10.

Note that column 4 has top net connection 5 and bottom net connection 3, so net 5 must be routed above net 3. Vertical constraints are *transitive*: since net 4 must be routed above net 5, it must also be routed above net 3.

A *vertical constraint graph* is a directed graph that indicates routing precedence from top to bottom. An ancestor vertex must be routed above its descendent vertices. The vertical constraint graph for Fig. 1 is shown in Fig. 4.

Figure 5 illustrates the vertical constraints for nets 8, 9 and 10. Note that net 10 must be routed above net 9, while net 9 must be routed above net 8. Since vertical constraints are transitive, nets 8 and 10 *cannot* be assigned to the same horizontal track.

2. Multiple-Objective Genetic Algorithm for Channel Routing

Early attempts to solve the channel routing have used deterministic methods (see Refs. 1 and 2 for more details). However, the channel routing problem has been shown to be NP-complete.[3] Therefore, recent channel routing methods have used

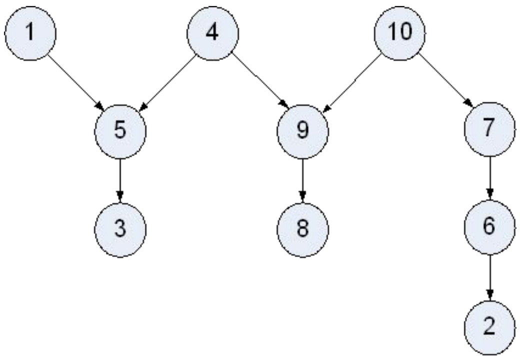


Fig. 4. Vertical constraint graph for Fig. 1.

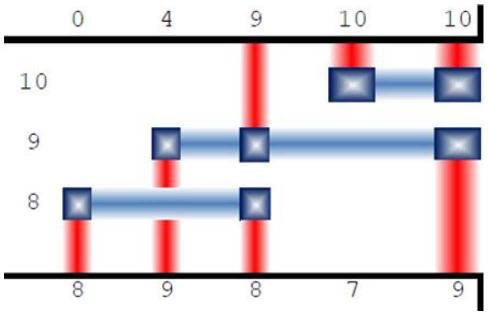


Fig. 5. Vertical constraints for nets 8, 9 and 10.

genetic algorithms (see Refs. 4–10 for more details). The traditional objective for most channel routers is to minimize channel height, which minimizes the total area required to route all interconnection wires. Minimizing channel height is accomplished by minimizing the total number of horizontal tracks required to complete the routing.

However, another important objective is to minimize signal propagation delays in the circuit. Signal propagation delay is directly proportional to the square of the interconnection wire length.[11] Delay also depends on wire width; however, wire widths are fixed in the channel routing problem, so wire length is the predominate variable in delay. Therefore the genetic algorithm for the channel router must optimize for two objectives: minimal number of tracks and minimal total wire length. For many multiple-objective problems, there is no single optimal solution, but rather a set of alternative solutions. These solutions, called *Pareto-optimal* solutions, are optimal in the sense that they *dominate* other solutions in the search space. That is, the Pareto-optimal solutions are superior to the other solutions when all objectives are considered (see Refs. 12–14 for more details). Therefore, we developed a multiple-objective genetic algorithm for channel routing that uses the Pareto-optimal approach.

Given the sequences T, B, and the netlist N for a specific channel, the horizontal and vertical constraints are identified using the methods described in the previous section. Next, the genetic algorithm is applied to determine the track assignment of the nets:

```
Begin
  Generate initial population P of size 200
  While (not termination condition) do
    Evaluate fitness of P
    For 100 iterations do
      Select parents $R_1$,$R_2$ from P
      Create offspring X by crossover on parents
      With probability 0.15, mutate offspring
      Replace weaker parent with offspring
      Evaluate fitness of offspring
    End for
  End while
  Final solution = individual with "best" fitness
End
```

2.1. *Population generation*

The population size is 200 individuals, and the initial population is randomly generated. The chromosome structure, based on Refs. 8 and 10, is an integer string $\{x_1 x_2 \cdots x_M\}$ for M nets. Each integer $x_i = j$ means that net i is assigned to track j, where $j \in [1, T]$ for T tracks. The chromosome locus is the net number, while the chromosome allele is the track number. For example, the chromosome $\{2\ 3\ 1\}$ means that net 1 is assigned to track 2, net 2 is assigned to track 3, and net 3 is assigned to track 1.

2.2. *Fitness evaluation*

For a given chromosome, the represented solution may not be feasible: there may be horizontal and/or vertical constraints if the nets are assigned to the tracks specified in the chromosome. Therefore, a chromosome's feasibility must be determined during fitness evaluation.

2.2.1. *Feasibility check*

First, we need to identify all tracks that have more than one net assigned; if a track has only one net, then there are no constraint issues. For each multiple-net track, check for horizontal constraints, then vertical constraints, between the nets assigned to the track. If there are no constraints for *any* of the net assignments, then the solution is feasible.

2.2.2. *Fitness function*

The objectives of the channel router are to minimize the number of horizontal tracks and minimize the total interconnection wire length of the final solution. For the channel routing problem, the lengths of the horizontal segments will be fixed, as they depend on the fixed terminal connections of the cells. However, the lengths of the *vertical* segments will change, depending on the particular track assignment of the corresponding horizontal segment of the net. Therefore, the objective of minimizing total interconnection wire length becomes the objective of minimizing the total *vertical segment* wire length.

A Pareto-optimal approach[15] is used to compare possible solutions. Let $\mathbf{v} = \{v_1, v_2\}$ be a *feasible solution vector* for a given individual of the genetic algorithm, where v_1 = total number of horizontal tracks used by the solution and v_2 = total vertical segment wire length for the solution. Then, for two solution vectors $\mathbf{x}$ and $\mathbf{y}$, $\mathbf{x}$ *dominates* $\mathbf{y}$ if $\forall i : x_i \le y_i$ *and* $\exists j : x_j < y_j$.

For example, if $\mathbf{x} = \{5, 80\}$ and $\mathbf{y} = \{5, 100\}$, then solution $\mathbf{x}$ dominates solution $\mathbf{y}$. Both $\mathbf{x}$ and $\mathbf{y}$ use 5 tracks; however, $\mathbf{x}$ has shorter total wire length than $\mathbf{y}$. Therefore, solution $\mathbf{x}$ is "fitter" than solution $\mathbf{y}$.

2.3. *Parent selection*

After fitness evaluation, the population is ranked: the individuals are sorted based on their fitness values. The individual with the highest fitness value is ranked 1, and the lowest is ranked $|P|$(population size).

Two individuals are randomly selected from the population to be parents, using binary tournament selection.[16] A random number r $(0 < r < 1)$ is generated. If $r < k$ (a selected bias parameter), then the fitter parent is selected. Otherwise, the less fit parent is selected. For our genetic algorithm, $k = 0.75$, so the bias is toward the fitter parent.

2.4. *Crossover*

The parents produce an offspring during crossover. The crossover rate is 0.5, which means that during each generation (population size * crossover rate), offsprings are created. Therefore, 100 sets of parents are selected, and each set of parents produces an offspring.

Crossover is performed using a "pointwise" approach.[8] First, a crossover point (index p) is randomly selected, where $p \in [1, M]$. The offspring chromosome contains alleles [1, p] of the first parent, and alleles $[p + 1, M]$ of the second parent.

For example, given the following parents:

Parent 1: {8 6 3 4 9 5 7 3 1 4}

Parent 2: {5 4 9 8 6 6 1 9 1 8}

If the crossover point $p = 4$, then the resulting offspring is: {**8 6 3 4** *6 6 1 9 1 8*}.

2.5. *Mutation*

The offspring may also undergo mutation, with a probability of 15% (*mutation rate* = 0.15). The approach of Ref. 8 was used, which is to randomly select various loci on the offspring and change their alleles to valid track numbers. For the previous offspring example: {8 6 3 4 6 6 1 9 1 8}.

Assume that loci 3, 5, 9 are randomly selected to be changed. We replace them with a random value in the range [1, #nets]:
{8 6 $\underline{\mathbf{7}}$ 4 $\underline{\mathbf{10}}$ 6 1 9 $\underline{\mathbf{5}}$ 8}.

2.6. *Termination*

The genetic algorithm terminates if there has been no change in the "best solution" after 1000 generations.

3. Results

The channel routing genetic algorithm was developed in C++ and run in UNIX on channel data sets provided in Refs. 1 and 5. First, the channel router was run only considering the objective of minimizing number of tracks (the traditional approach). Next, the channel router was run considering both objectives (minimizing number of tracks and minimizing total wire length). The results are shown in Table 1. For multiple-objective optimization methods, there may be trade-offs between objectives

Table 1. Comparison of single objective versus multiple objectives for channel routing.

	Channel height		Channel height and wire length	
Netlist	#Tracks	Total wire length	#Tracks	Total wire length
chan1 (Ref. 1)	5	84	5	41
ch1 (Ref. 5)	6	50	6	35
ch2 (Ref. 5)	6	66	6	44
ch3 (Ref. 5)	6	66	6	46
ch4 (Ref. 5)	7	80	6	40

to obtain a "near-optimal" solution. However, as seen in Table 1, optimizing for both objectives reduces total wire length, without increasing the channel height. The wire length units depend on the technology scaling factor λ. For example, given a typical contemporary value of $\lambda = 90\,\text{nm}$, then a wire length of 50 units $= (50)(90\,\text{nm}) = 4500\,\text{nm} = 4.5\,\mu\text{m}$.

The data set *chan1* of Ref. 1 is our working example of Fig. 1. Figure 6 shows the final routing layout when the genetic algorithm only considers channel height, while Fig. 7 shows the layout when optimizing for both channel height and total wire

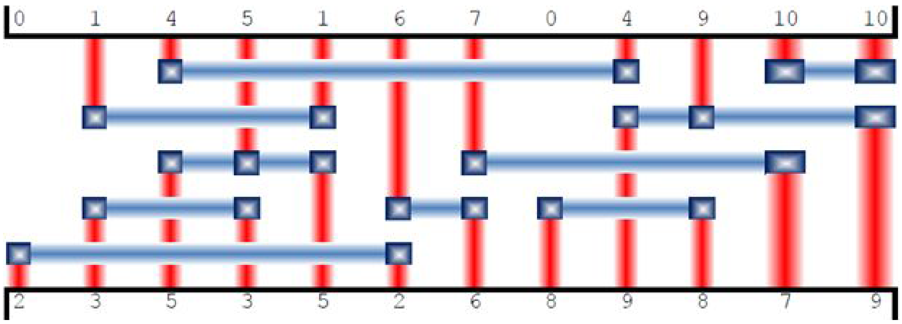


Fig. 6. Channel routing solution for *chan1*, optimizing for number of tracks only.

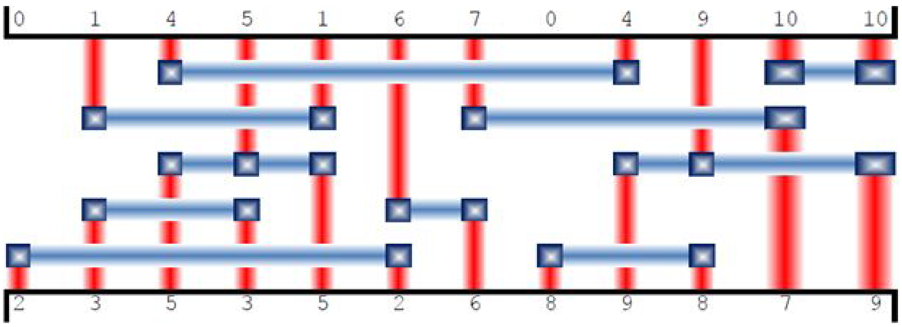


Fig. 7. Channel routing solution for *chan1*, optimizing for number of tracks and total wire length.

length. Both solutions required five horizontal tracks to complete all interconnections. For Fig. 7, note that the horizontal tracks for nets 8 and 9 are assigned to lower tracks than for Fig. 6. This reduces the overall interconnection wire length.

Similar results occur for the data set *ch1* of Ref. 5. Figure 8 shows the final routing layout when the genetic algorithm only considers channel height, while Fig. 9 shows the layout when optimizing for both channel height and total wire length. Both solutions required six horizontal tracks to complete all interconnections. However, the layout of Fig. 9 assigns net 2 to the uppermost horizontal track, as net 2 only has connections to the top of the channel. As with *chan1*, the overall interconnection wire length is reduced when it is considered as an objective in the algorithm. For data sets *ch2*, *ch3* and *ch4*, the final routing layouts are shown in Figs. 10–15.

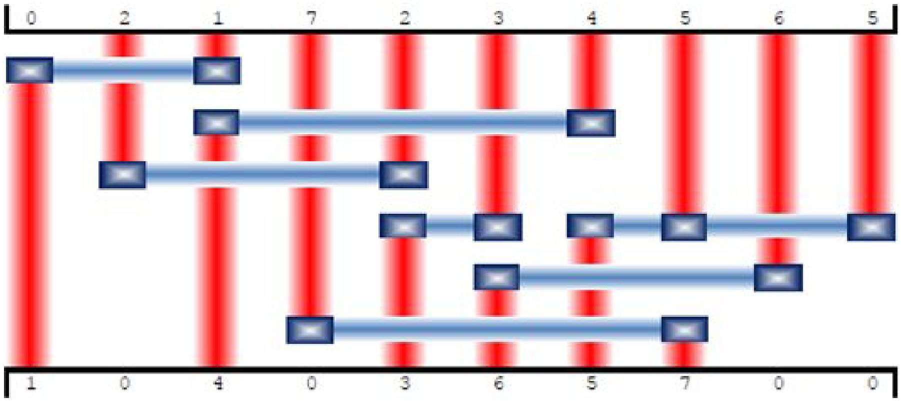


Fig. 8. Channel routing solution for *ch1*, optimizing for number of tracks only.

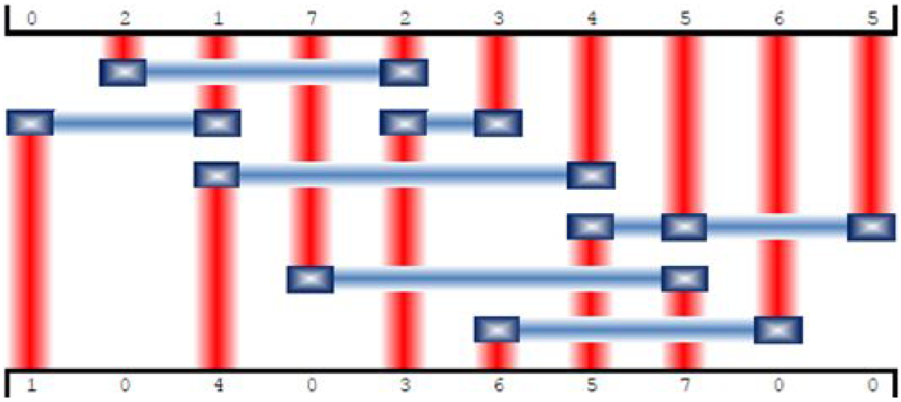


Fig. 9. Channel routing solution for *ch1*, optimizing for number of tracks and total wire length.

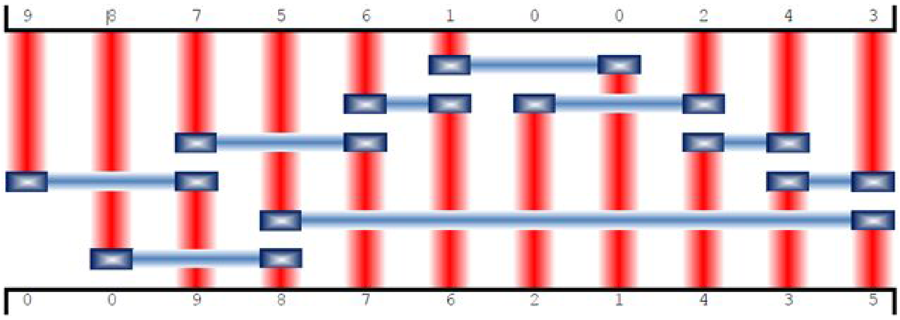


Fig. 10. Channel routing solution for *ch2*, optimizing for number of tracks only.

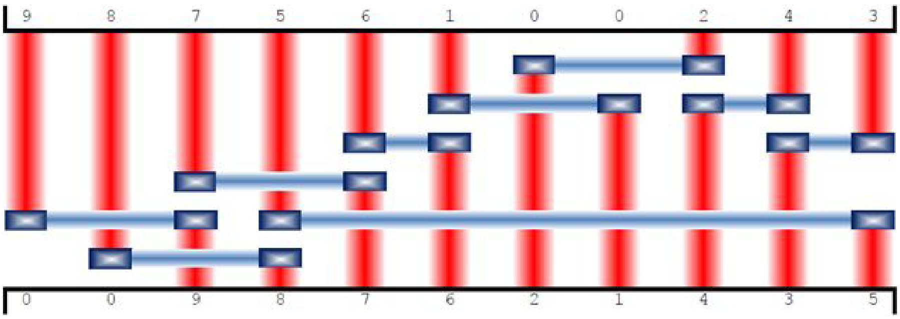


Fig. 11. Channel routing solution for *ch2*, optimizing for number of tracks and total wire length.

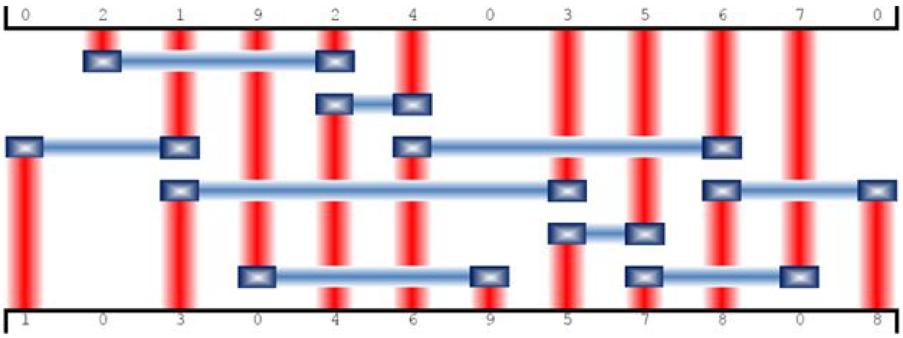


Fig. 12. Channel routing solution for *ch3*, optimizing for number of tracks only.

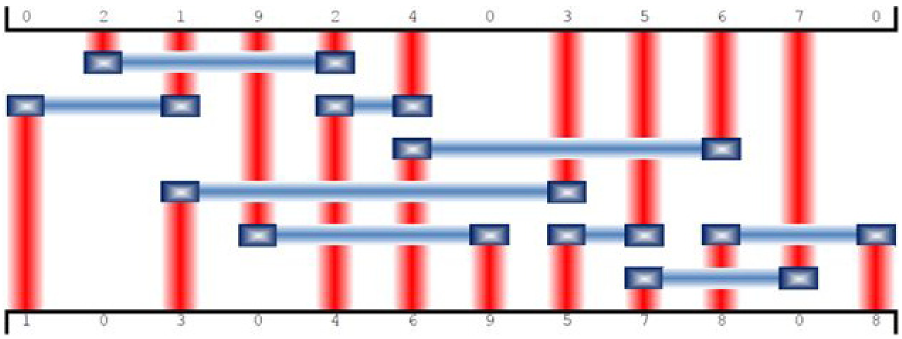


Fig. 13. Channel routing solution for *ch3*, optimizing for number of tracks and total wire length.

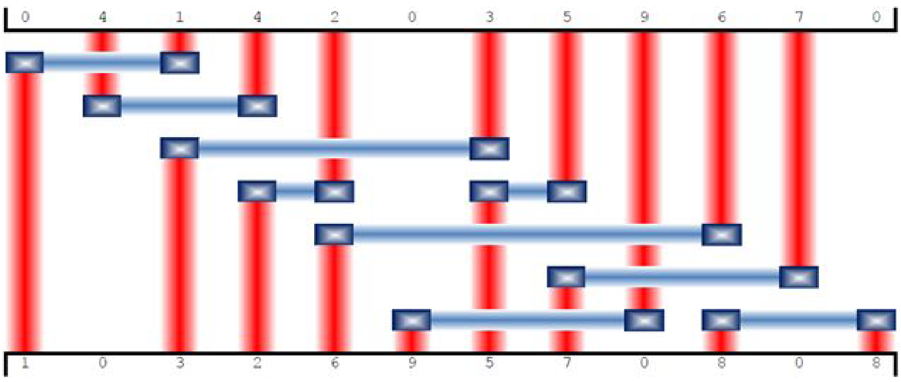


Fig. 14. Channel routing solution for *ch4*, optimizing for number of tracks only.

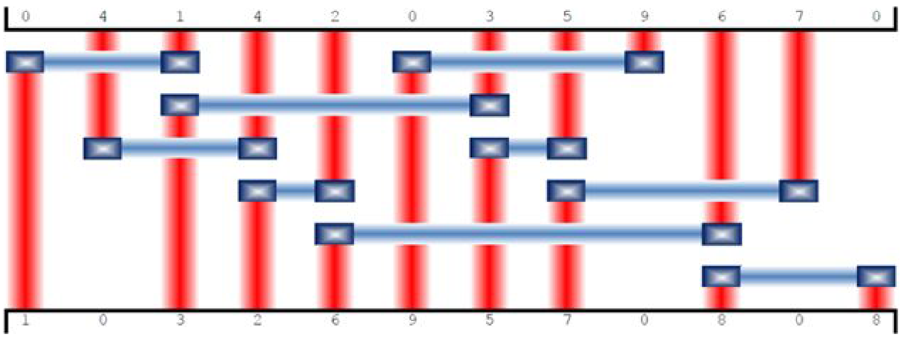


Fig. 15. Channel routing solution for *ch4*, optimizing for number of tracks and total wire length.

Table 2. Comparison of number of iterations for each approach.

Netlist	Channel height	Channel height and wire length
chan1 (Ref. 1)	1614	1561
ch1 (Ref. 5)	1058	1360
ch2 (Ref. 5)	1062	2083
ch3 (Ref. 5)	1526	1904
ch4 (Ref. 5)	1739	1385

For genetic algorithms, a common approach to compare methods is to view the number of iterations required to converge to a solution. This approach is used instead of run-time, since number of iterations is machine independent. Table 2 shows the total iterations required to converge to the solutions for each data set. Recall that the genetic algorithm terminates if there is no change in the "best solution" after 1000 iterations. Therefore, the number of iterations to converge to the final solution = total number of iterations − 1000. For data sets *ch1*, *ch2* and *ch3*, the multiple-objective approach took more iterations than the single-objective approach. However, the opposite was true for data sets *chan1* and *ch4*.

4. Conclusion

The multiple-objective genetic channel routing algorithm was shown to produce interconnection solutions that had the same or better channel heights, but reduced total wire lengths, when compared to the traditional single-objective approach. Recall that minimizing channel height results in reduced routing area, while minimizing wire lengths results in reduced signal propagation delays, which are two important objectives for integrated circuit design. Future research includes testing for other objectives, such as minimizing signal crosstalk. Crosstalk occurs when there are long parallel lengths of wires close to each other, so an objective would need to be added into the genetic algorithm fitness function to minimize parallel wire lengths.

References

1. T. Yoshimura and E. S. Kuh, Efficient algorithms for channel routing, *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **CAD-1** (1982) 25–35.
2. T. T. Ho, S. S. Iyengar and S. Q. Zheng, A general greedy channel routing algorithm, *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.* **10** (1991) 204–211.
3. T. G. Szymanski, Dogleg channel routing is NP-complete, *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **4** (1985) 31–41.
4. N. Gockel, G. Pudelko, R. Drechsler and B. Becker, A hybrid genetic algorithm for the channel routing problem, *IEEE Int. Symp. Circuits and Systems (ISCAS '96)*, Vol. 4 (1996), pp. 675–678.
5. R. K. Pal, D. Saha and S. S. Sarma, A mimetic algorithm for computing a nontrivial lower bound on number of tracks in two-layer channel routing, *J. Phys. Sci.* **11** (2007) 199–210.

6. B. B. Prahlada-Rao, L. M. Patnaik and R. C. Hansdah, A genetic algorithm for channel routing using inter-cluster mutation, *IEEE World Congress on Computational Intelligence*, Vol. 1 (1994), pp. 97−103.
7. A. Ruiz-Andino and J. J. Ruz, Integration of constraint programming and evolution programs: Application to channel routing, *Proc. 11th Int. Conf. Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE '98)* (1998), pp. 448−459.
8. S. Salcedo-Sanz, Y. Xu and X. Yao, Meta-heuristic algorithms for FPGA segmented channel routing problems with non-standard cost functions, *Genet. Program. Evol. Mach.* **6** (2005) 359−379.
9. S. Singha, T. Bhattacharya and S. R. B. Chaudhuri, An approach for reducing crosstalk in restricted channel routing using graph coloring problem and genetic algorithm, *Proc. 2008 Int. Conf. Computer and Electrical Engineering (ICCEE '08)* (2008), pp. 807−811.
10. L. Wang, L. Zhou and W. Liu, FPGA segmented channel routing using genetic algorithms, *Proc. IEEE Congr. Evol. Comput. (CEC)* (2005), pp. 2161−2165.
11. J. P. Uyemura, *Introduction to VLSI Circuits and Systems* (Wiley, 2001).
12. H. Eskandari and C. Geiger, A fast Pareto genetic algorithm approach for solving expensive multiobjective optimization problems, *J. Heuristics* **14** (2008) 203−241.
13. R. C. Purshouse and P. J. Fleming, On the evolutionary optimization of many conflicting objectives, *IEEE Trans. Evol. Comput.* **11** (2007) 770−784.
14. E. Zitzler and L. Thiele, Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach, *IEEE Trans. Evol. Comput.* **3** (1999) 257−271.
15. R. Dewri, N. Poolsappasit, I. Ray and D. Whitley, Optimal security hardening using multi-objective optimization on attack tree models of networks, *Proc. 14th ACM Conf. Computer and Communications Security*, Alexandria, Virginia, USA (2007), pp. 204−213.
16. D. E. Goldberg and K. Deb, A comparative analysis of selection schemes used in genetic algorithms, *Foundations of Genetic Algorithms*, ed. G. Rawlins (Morgan Kaufmann, 1991).

Copyright of Journal of Circuits, Systems & Computers is the property of World Scientific Publishing Company and its content may not be copied or emailed to multiple sites or posted to a listserv without the copyright holder's express written permission. However, users may print, download, or email articles for individual use.