The Efficient Hybrid Approach to Channel Routing Problem

Chuleui Hong and Yeongjoon Kim

Department of Computer Science, Sangmyung University, Seoul, Korea
{hongch, yjkim}@smu.ac.kr

Abstract

Known as an NP-Complete problem, the channel routing problem is very important in the automatic layout design of VLSI circuit and printed circuit boards. A distributed hybrid algorithm for this channel routing problem is presented in MPI environments. This system is implemented on a network of personal computers running Linux operating system connected via 100Mbps Ethernet. Each slave processor generates its own sub-population using genetic operations and communicates with the master processor in an asynchronous manner to form the global population. The proposed hybrid algorithm of Mean Field Annealing and Simulated annealing-like Genetic Algorithm combines the benefit of rapid convergence property of MFA and the effective genetic operations of SGA. The experimental results show that the proposed algorithm maintains the convergence properties of sequential genetic algorithm while it achieves linear speedup as the nets of the channel routing and the number of computing processors increase.

Keywords: *Channel Routing, Genetic Algorithm, Mean Field Annealing*

1. Introduction

The channel routing problem is very important in the automatic layout design of VLSI circuits. "Fig. 1" illustrates an example of channel routing. A channel consists of two parallel horizontal rows of points called terminals. These terminals are placed at a regular interval and identified by the columns of the channel. A net consists of the same terminals that must be interconnected without being overlapped with others. A layer is an interconnected routing area of the nets to connect the terminals. The channel routing problem is routing the given nets between terminals on the minimum multilayer channel areas.

The proposed Mean Field Genetic Algorithm (MGA) is a hybrid algorithm based on mean field annealing (MFA) [1] and genetic algorithm (GA) [2, 3, 4]. MFA has the characteristics of rapid convergence to the equilibrium state comparing with the simulated annealing (SA) [5, 6]. This distributed hybrid algorithm is verified practical in the solution quality and computation time.

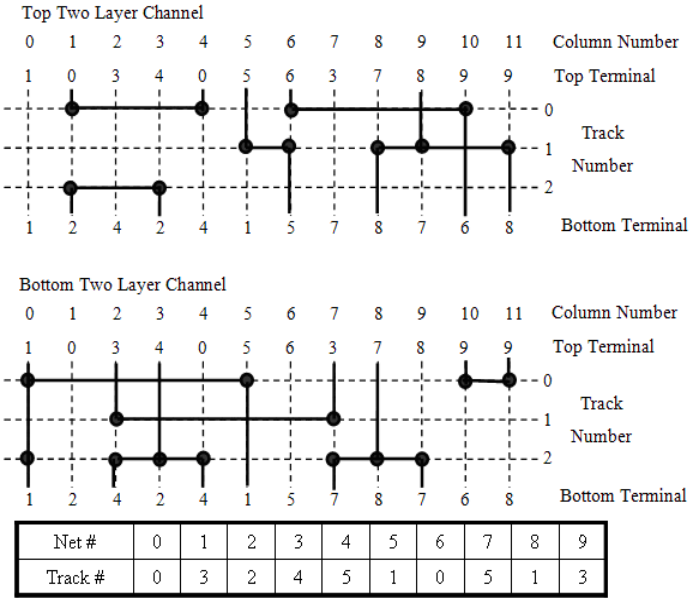
2. Genetic Operations for Channel Routing

Genetic algorithm provides an effective means for global optimization in a complex search space such as NP-complete optimization problems including VLSI routing.

2.1 Representation of Chromosome

Funanbiki et. al., [7] used $n \times m \times l$ processing elements each of which represents the output of the solution in n nets with m tracks and l layers. However, we use a scalar value representation for unique track number, a_i ranging 0 to $m \times (l\text{-}1)$. When the i^{th} net is assigned to the j^{th} track of layer k, $a_i = j + (k\text{-}1) \times m$. So in Figure 1 case, a_i ranges 0 to 5.

Funabiki's algorithm needs more processing elements as *m* and *l* increases, while the proposed algorithm does not depend on the size of a track and layer but only on the size of nets which will be the chromosome in GA. For example as in Figure 1, which is a 10-Net, 3-Track and 2-Layer problem, Funabiki's algorithm needs 10×3×2 = 60 processing elements, while 400 processing elements are needed for a 10-Net, 10-Track and 4-Layer problem. However, our algorithm needs 10 integers for both cases in MGA representations. This advantage is from the GA nature.



Net #	0	1	2	3	4	5	6	7	8	9
Track #	0	3	2	4	5	1	0	5	1	3

Figure 1. The Example of Channel Routing for 10-net, 3-track, 2-layer Problem

2.2 Objective Function

When a net is assigned to the track, it may overlap to other nets in the vertical and horizontal directions. Since the objective is to find the placement of nets free of overlaps, the fitness is the reciprocal of the total overlaps of a chromosome. The fitness function, C(s), is α / (β+TOTAL_C) where TOTAL_C is the total number of overlaps which is the sum of horizontal and vertical overlapped numbers. α is the square of the total number of nets and β is the small positive real number forbidding division by zero.

2.3 Genetic Operators

For the selection of new population from the old population, a fitness proportionate selection method called roulette scheme is used. The proposed novel crossover operators follow two steps as shown in Figure 2.

- Simple crossover: First step is copying the configuration of the parents, *P1* and *P2*, to the children, *C1* and *C2*, respectively. But unlike the standard simply crossover, only alleles of the larger part over the crossover position are copied.
- Overlap-free crossover: Second step is filling the remaining alleles. For the undetermined alleles of *C1*, the candidate configuration is smaller part of *P2*. In this step we check whether the candidate allele assignment generates vertical or horizontal overlaps with previous assignments of *C1*. If overlaps are found, we search for a new

non-overlapped allele assignment. For a child configuration, *C2*, we follow the same steps.

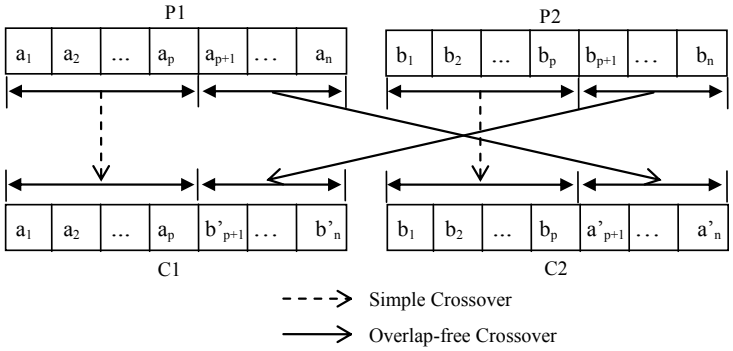


Figure 2. Crossover Operation

3. The Proposed Distributed Algorithm

3.1 Distributed Mean Field Annealing (MFA)

MFA is derived from SA based on mean field approximation method in physics [1]. The N×T spin matrix is partitioned column-wise such that each net is assigned an individual. The randomly selected net-i is responsible for updating its spin value, s_{ip}. The objective function, C(s), of MFA is same as that of GA. When the all nets are placed free of horizontal and vertical overlaps, the objective value will be 0.

$$C(s) = \sum_{i=0}^{N-1} \sum_{j \neq i} \sum_{p=0}^{T-1} s_{ip} s_{jp} o_{ij}$$

N : The number of Nets

T : The number of Tracks

s_{ip} : The probability of net i mapping to track p

s_{jp} : The probability of net j mapping to track p

o_{ij} : The overlapped tracks and columns for net i and j

The pseudo code for the distributed mean field annealing algorithm of each node is as follows.

```
<Distributed Mean Field Annealing>
while (cost change is less than ε for continuous N
       annealing process)
   Select a same net-i at random by using same seed;
   Compute the local mean field;
```

$$\phi_{ip} = \sum_{j \neq i}^{N} s_{jp} o_{ij} \quad \text{for } 0 \le p \le T-1$$

```
   Compute the new spin values at the i^th
      row by using global sum operation;
```

$$s_{ip}^{new} = \frac{e^{\phi_{ip}/T}}{\sum_{p=1}^{K} e^{\phi_{ip}/T}} \quad \text{where } T \text{ is temperature}$$

```
Compute the cost change due to spin
      updates by using global sum operation;
```

$$\Delta C = \sum_{p=1}^{K} \phi_{ip}(s_{ip}^{new} - s_{ip})$$

```
Update the spin values at the i^th row
```

$$s_{ip} = s_{ip}^{new} \quad \text{for} \quad 0 \le p \le T-1$$

```
Perform global collect
  for a spin value, s_ip, at the i^th row;
```

In implementing MFA, the cooling schedule has a great effect on the solution quality. Therefore the cooling schedule must be chosen carefully according to the characteristics of problem and cost function. Length of the Markov chain at a certain temperature is the number of state transition to reach the equilibrium state. It is set to the number of state transitions where the cost change is less than ε=0.5 for continuous N annealing process.

3.2 Distributed Simulated Annealing-like Genetic Algorithm (SGA)

We modified GA such that the new evolved state is accepted with a Metropolis criterion like simulated annealing in order to keep the convergence property of MFA. The modified GA is called SGA. ΔC is the cost change of new state from old state. It is made by subtracting the cost of new state from that of old one. T is the current temperature.

Pr[ΔC is accepted]= min[1, exp($\Delta C/T$)]

A string in the order of nets whose value is allocated track number represents the individual of Genetic Algorithm. For example, a string, "1,0,2,4,5", means that nets are allocated to tracks such that net 0 to track 1, net 1 to track 0, net 2 to track 2, net 3 to track 4, net 4 to track 5.

The individuals are generated randomly with the probability as same as that of spin matrix in MFA. For example, if spin values of an arbitrary i^{th} net, which is the elements of i^{th} row, is 0.2, 0.4, 0.1, 0.1, 0.2, an individual is made such that the i^{th} character in a string can be 0 with a probability of 0.2, 1 with that of 0.4, 2 with that of 0.1, 3 with that of 0.1 and so on.

The pseudo code for the distributed genetic algorithm of each node is as follows.

```
<Distributed SGA>
Initialize subpopulation(P_sub) from MFA spin matrix;
for (iteration is less than max_epoch)
  Calculate fitness for P_sub ;
  for (generations = 1 until epoch_length)
   Select individuals from subpopulation;
   Reproduce next population;
   for (select 2 individuals by turns)
    Perform crossover with probability of crossover;
    Calculate the cost change (ΔC);
    if (exp(-ΔC/T)> random[0,1]) then
     Accept new individuals;
   for (all individuals)
    Perform mutation with probability of mutation;
    Calculate the cost change (ΔC);
    if (exp(-ΔC/T)> random[0,1]) then
     Accept new individuals;
  broadcast P_sub to all other nodes;
```

```
select new P_sub randomly;
Keep the best individual;
```

In the experiment, the subpopulation size in each processor is set to the number of nets, N. Therefore the size of global population is the multiplication of the number of nets and the number of processors, $N \times P$.

In our synchronous distributed genetic algorithm, each processor generates subpopulation randomly from the MFA's spin matrix. And then the subpopulation and its fitness value are broadcast to all other nodes and they form the global population.

Next, the individuals are selected as much as the size of subpopulation from the global population randomly. Each processor executes the sequential genetic algorithm in parallel. Independent genetic operation are implemented and evaluated to its subpopulation. The duration of isolated evolution is called one epoch and the *epoch length* is the number of predefined generations for a processor before synchronizing communication among the processors. The epoch length is set to the N/P, where N is the number of nets and P is the number of processors. *max_epoch* is the number of synchronous communications. It is set to P.

3.3 MGA Hybrid Algorithm

A new hybrid algorithm called MGA combines the merits of mean field annealing (MFA) and simulated annealing-like genetic algorithm (SGA)[8,9]. MFA can reach the thermal equilibrium faster than simulated annealing and GA has powerful and various genetic operations such as selection, crossover and mutation. First, MFA is applied on a spin matrix to reach the thermal equilibrium fast. After the thermal equilibrium is reached, the population for GA is made according to the distribution of net to track allocation in the spin matrix. Next, GA operations are applied on the population while keeping the thermal equilibrium by transiting the new state with Metropolis criteria. MFA and SGA are applied by turns until the system freeze.

The followings are the pseudo code for the distributed MGA algorithm of each node.

```
<Distributed MGA Hybrid Algorithm>
Forms the spin matrix,s=[s_11, …, s_ip, …, s_NK];
Get the initial temperature T_0, and set T= T_0 ;
while (T ≥ T_f)
  Executes MFA;
  Forms GA population from a spin matrix of MFA;
  Executes SGA;
  Forms the spin matrix of MFA from GA population;
  T= α×T;   /*decrease the temperature*/
```

Initial temperature, T_0, is set such that the probability where the cost change is less than ε(=0.5) is more than 95% for the number of nets(=N) annealing process. Final temperature (T_f) is set to the temperature where the value of the cost change is in ε/1,000 for continuous N temperature changes. A fixed decrement ratio, α, is set to 0.9 experimentally. This strategy decreases the temperature proportional to the logarithm of the temperature.

4. Simulated and Evaluation

The simulation is implemented in distributed environments which are made up of 1.2Ghz personal computers running Linux connected via 100Mbps Ethernet. The benchmark

problems were from Funabiki's[7] examples. We ran 100 times for each example. In Table 1, the minimum channel widths of MGA are same as those of Funabiki. This means that MGA reach the optimal solutions. The minimum channel width is the minimum number of tracks needed to contain all nets without overlapped.

Table 1. Minimum Channel Width(Tmax) Comparison of MGA and Funabiki's Results

# of nets	# of columns	Tmax	
		MGA	Funabiki
21	43	6	6
45	90	8	8
47	84	9	9
54	103	9	9
55	119	9	9
61	128	10	10
72	175	11	11

The experiment shows that the crossover probability is optimal at 0.6 and the mutation probability has a great effect on the solution quality. When the mutation probability is large, it becomes quite a random search. While the mutation probability is small, it is difficult to escape from the local optimum configurations. We simulated the proposed algorithm using various mutation probabilities. Table 2 shows the parameters in different number of nets.

Table 2. Parameters of Distributed Genetic Algorithm

# of nets	Population size	Prob. of crossover	Prob. of mutation	Run Time (in sec)
21	128	0.6	0.05	129.3
45	128	0.6	0.01	169.2
47	128	0.6	0.01	76.2
54	128	0.6	0.01	1083.7
55	128	0.6	0.01	1301.2
61	256	0.6	0.005	30.6
72	256	0.6	0.005	7027.8

The run times of sequential algorithms are relatively long as in Table 2, so we turn over the parallelizing. The inherent nature of a mean field annealing and genetic algorithm makes it possible to be paralleled in multiprocessing environment. Thus this distributed version keeps the solution quality as much as the sequential one with less computation time.

The parallel speedup generally increases proportional to the number of nets and the population size as in Figure 3. However, for the net-45 and net-72 problem, they are much complicated and the computing times are extremely long, so their epoch which is the duration of isolated evolution as defined in section 3.2, are longer and the speedups are larger than other problems. This shows that proposed distributed MGA algorithm is useful as the problem size increases.

This distributed MGA algorithm was successful to find the solutions on several benchmark problems as shown in Figure 4.

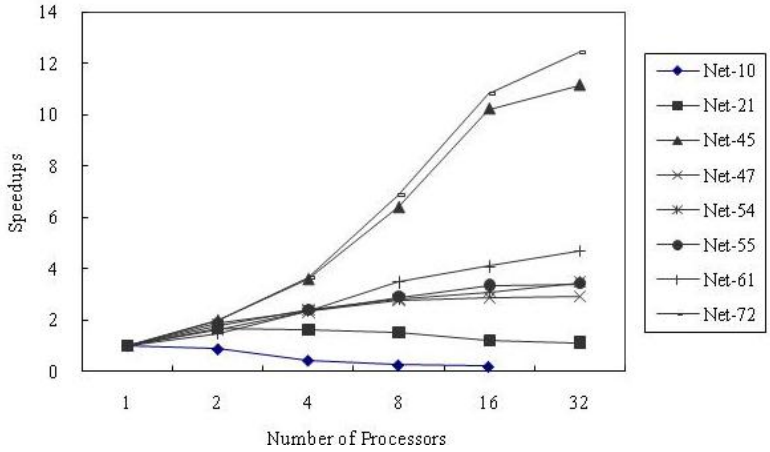


Figure 3. Speedups of Different Nets

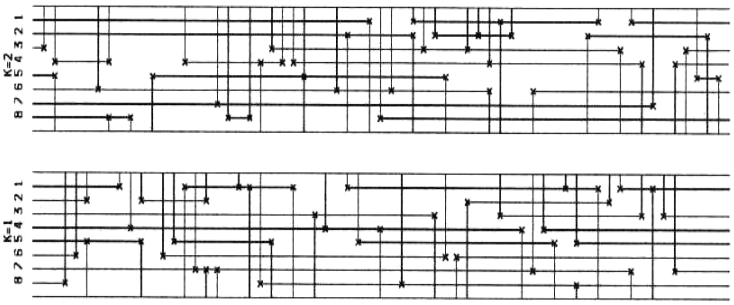


Figure 4. The One Solution of Benchmark Problems

5. Conclusions

In this paper, we proposed a new distributed hybrid algorithm called MGA. This proposed MGA algorithm was verified successful to the several benchmark routing problems. In addition, this algorithm may be easily modified to solve the partitioning and placement problems for the design automation.

The proposed approach combines the merits of MFA and GA in distributed memory multiprocessor systems. The execution time of MGA is shorter than that of MFA and GA alone keeping the same solution quality. MGA was also verified by producing more promising and useful results as the problem size and complexity increases. The proposed algorithm can be easily developed as a distributed algorithm since MFA and GA can be parallelized easily. This algorithm also can be applied efficiently to broad ranges of NP-Complete problems. In the future, we will research on dogleg and area router cases.

References

[1] S. Salleh and A. Y. Zomaya, "Multiprocessor Scheduling Using Mean-Field Annealing", Proceedings of the First Workshop on Biologically Inspired Solutions to Parallel Processing Problems (BioSP3), **(1998)**, pp. 288-296.

[2] A. Brezulianu, M. Fira and L. Fira, "A Genetic Algorithm Approach for a Constrained Employee Scheduling Problem as Applied to Employees at Mall Type Shops", International Journal of Advanced Science and Technology, vol. 14, **(2010)**, pp. 1-14.

[3] V. Golmah and J. Parvizian, "Visualization and the understanding of multidimensional data using Genetic Algorithms: Case study of load patterns of electricity customers", International Journal of Database Theory and Application, vol. 3, no. 4, **(2010)**, pp. 41-56.

[4] T. Rao and S. Madiraju, "Genetic Algorithms and Programming-An Evolutionary Methodology", International Journal of Hybrid Information Technology, vol. 3, no. 4, **(2010)**, pp. 1-18.

[5] C. Hong, "Channel Routing using Asynchronous Distributed Genetic Algorithm", Journal of Computer Software & Media Tech., SMU, vol. 2, **(2003)**.

[6] C. Hong and B. McMillin, "Relaxing synchronization in distributed simulated annealing", IEEE Trans. on Parallel and Distributed Systems, vol. 16, no. 2, **(1995)**, pp. 189-195.

[7] N. Funabiki and Y. Takefuji, "A Parallel Algorithm for Channel Routing Problem", IEEE Trans. of CAD, vol. 11, no. 4, **(1992)**, pp. 464-474.

[8] T. Karthikeyan, S. Peter and S. Chidambaranathan, "Hybrid Algorithm for Noise-free High Density Clusters with Self-Detection of Best Number of Clusters", International Journal of Hybrid Information Technology, vol. 4, no. 2, **(2011)**, pp. 39-54.

[9] M. Balitanas, "Wi Fi Protected Access-Pre-Shared Key Hybrid Algorithm", International Journal of Advanced Science and Technology, vol. 12, **(2009)**, pp. 35-44.

Authors

Chuleui Hong

Chuleui Hong received B.S. degree in Hanyang University, Seoul, Korea in 1985, and his M.S. degree and Ph.D. degree in Computer Science at New Jersey Institute of Technology, USA and University of Missouri-Rolla, USA in 1989 and 1992, respectively. He was a senior researcher in Electronic and Telecommunications Research Institute, Korea, from 1992 to 1997. He joined Sangmyung University as a faculty member in 1997 and is currently professor of Computer Science Department at Sangmyung University, Seoul, Korea. His research interests include parallel and distributed system, optimization algorithm, multimedia application, and intelligent agent.

Yeongjoon Kim

Yeongjoon Kim received the B.E. degree in industrial engineering from Korea University, Seoul, Korea, in 1984, and the Ph.D. degree in computer science from the University of Houston, in 1996. He is currently a professor at the Division of Computer Science, Sangmyung University, Seoul, Korea. His research interests include machine learning, expert systems, and genetic algorithms.