ICIC Express Letters
Volume **6**, Number **3**, March **2012**

ICIC International ©2012 ISSN 1881-803X
pp. 771–776

A TWO-STEP BINARY PARTICLE SWARM OPTIMIZATION APPROACH FOR ROUTING IN VLSI

Zulkifli Md Yusof[1], Amar Faiz Zainal Abidin[1,2], Asrul Adam[1,2]
Kamal Khalil[1], Jameel Abdulla Ahmed Mukred[1], Mohd Saberi Mohamad[3]
M. Khalil Hani[1] and Zuwairie Ibrahim[1,2]

[1]Faculty of Electrical Engineering
[2]Centre for Artificial Intelligence and Robotic
[3]Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia
81310 UTM Skudai, Johor, Malaysia
{ zmdyusof; amarfaiz; kamal; jameel; khalil; zuwairie }@fke.utm.my
asruldm@gmail.com; saberi@utm.my

Received June 2011; accepted September 2011

Abstract. *Manipulation of wire sizing, buffer sizing, and buffer insertion are a few techniques that can be used to improve time delay in very large scale integration (VLSI) circuit routing. This paper enhances an existing approach, which is based on Particle Swarm Optimization (PSO) for solving routing problem in VLSI circuits. A two-step Binary Particle Swarm Optimization (BPSO) approach, which is based on BPSO, is chosen in this study to improve time delay through finding the best path of wire placement with buffer insertion from source to sink. The best path of wire placement is found in the first step by the first BPSO and then the second BPSO finds the best location of buffer insertion along the wire. A case study is taken to measure the performance of the proposed model and the result is obtained compared with the previous PSO approach for VLSI routing.*
Keywords: Very large scale integrated, Routing problem, Particle swarm optimization

1. **Introduction.** One of the main important criteria of the performance of VLSI routing is time delay. Factors that influenced time delay are wire sizing, buffer sizing, and buffer insertion. Previously, Ayob et al. [1] had proposed the use of PSO to find the shortest path between source and sink while the buffers are placed randomly in a fix segment along the path. Based on the wire and buffer parameters, time delay is calculated. Since interconnect is divided into grids, wire without buffer or wire with buffer model was used to calculate delay at a vertex to the next vertex. In their case study, they only consider single wire and buffer type. There are few disadvantages of this approach in real practice; types of wire and buffer use are more than one. Another disadvantage is that the buffer placement is not optimized. Due to the disadvantages, this paper proposes a two-step BPSO approach for routing in VLSI where the shortest path of wire placement is found in the first step using first BPSO and then next BPSO will find the best wire size, buffer type and location of buffer insertion along the wire between two vertices for producing minimum delay.

2. **Fitness Function Evaluations.** The mathematical representation of VLSI Routing is done in [1]. In this paper, two fitness functions were considered: (1) total path and (2) time delay in picoseconds (ps). A path represents a number of segments from source to sink while time delay is an accumulative delay calculation from source to sink. For total

segments, p can be formulated as follows:

$$p = \left(\sum_{n=1}^{m-3} |e_n - e_{n+2}| \right) + |x_2 - e_{m-1}| + |x_1 - e_1| + |y_2 - e_{m-2}| + |y_1 - e_2| \quad (1)$$

where m is a maximum number of dog-leg and e_n represents the location of a node in grid-graph. x_1 and x_2 are x-axis coordinates of source and sink, respectively. y_1 and y_2 are y-axis coordinates of source and sink, respectively.

The Elmore Delay formulation [1] is chosen in order to calculate the time delay from source to sink. Based on Elmore Delay formulation, each node inside the wire is corresponded with a pair of resistance-delay (r, t), where r is donated as accumulated resistance of wire and t is accumulated time delay to that node. The calculation of subsequent segment (r', t') can be grouped into two models: (1) wire model and (2) wire with buffer model. For wire model, r' and t' are defined in the following equations:

$$r' = r_w + r \quad (2)$$

$$t' = (r + {}^{r_w}/_2)\, c_w + t \quad (3)$$

where r_w and c_w are resistance and capacitance of the wire segment, respectively. For subsequent segment that consists of wire with buffer, the equation is denoted as follows:

$$r' = r_b \quad (4)$$

$$t' = r\,(c_w + c_b) + r_w\,({}^{c_w}/_2 + c_b) + d_b + t \quad (5)$$

c_b and d_b are the model parameters, where c_b is input capacitance of buffer and d_b is intrinsic delay of buffer.

3. **Binary Particle Swarm Optimization.** BPSO has been introduced by J. Kennedy and R. Eberhert to solve discrete optimization problem [2,3]. Applications of BPSO can be seen in many engineering problems, such as PCB routing [4], computational biology [5,7,8], job scheduling [6] and agriculture [9]. The flow chart of the general BPSO is shown in Figure 1(a).

The first step in implementing any BPSO is to model the particle position, $\boldsymbol{s}$ to suit with the problem of interest. Usually, $\boldsymbol{s}$ represents a possible solution of the problem as shown in Figure 1(b). The information of the problem is shared with the algorithm. Then, BPSO parameters, as listed in Table 2, are initialized based on user's desired values. Particle's velocity, $\boldsymbol{v}$ and particle's position, $\boldsymbol{s}$, are randomly assigned. Next, the proposed solution by $\boldsymbol{s}$ is then checked to ensure it passes all the problem's constraints. $\boldsymbol{s}$ that does not fit the constraint will be ignored while others will proceed with the $\boldsymbol{s}$'s fitness, $f(\boldsymbol{s})$. After that, the particle's personal best, $\boldsymbol{p}$, and the swarm's global best, $\boldsymbol{g}$, are updated if required. Then, $\boldsymbol{v}$ is updated for next iteration, k for each bit, d, using Equation (6).

$$\boldsymbol{v}_{id}^{k+1} = \omega \boldsymbol{v}_{id}^{k} + r_1 c_1 (\boldsymbol{s}_{id}^{k} - \boldsymbol{p}_{id}^{k}) + r_2 c_2 (\boldsymbol{s}_{id}^{k} - \boldsymbol{p}_{id}^{k}) \quad (6)$$

where i is the particle's number. r_1, r_2 and r_3 are random number of [0,1]. c_1 and c_2 are cognitive parameter and social parameter, respectively. Next, $\boldsymbol{s}$ is updated using Equation (7) based on the probability of the normal distribution. Step 1 to Step 6 are repeated until maximum iteration is reached.

$$\boldsymbol{s}_{id}^{k+1} = \begin{cases} 1, & r_3 < \dfrac{1}{1 + e^{-v_{id}^{k+1}}} \\ 0, & r \geq \dfrac{1}{1 + e^{-v_{id}^{k+1}}} \end{cases} \quad (7)$$

The proposed model proposes the use of BPSO algorithm twice in designing the VLSI routing, where the first BPSO algorithm function is to find the shortest path between the source and the sink. Based on this shortest path, the second BPSO algorithm finds the

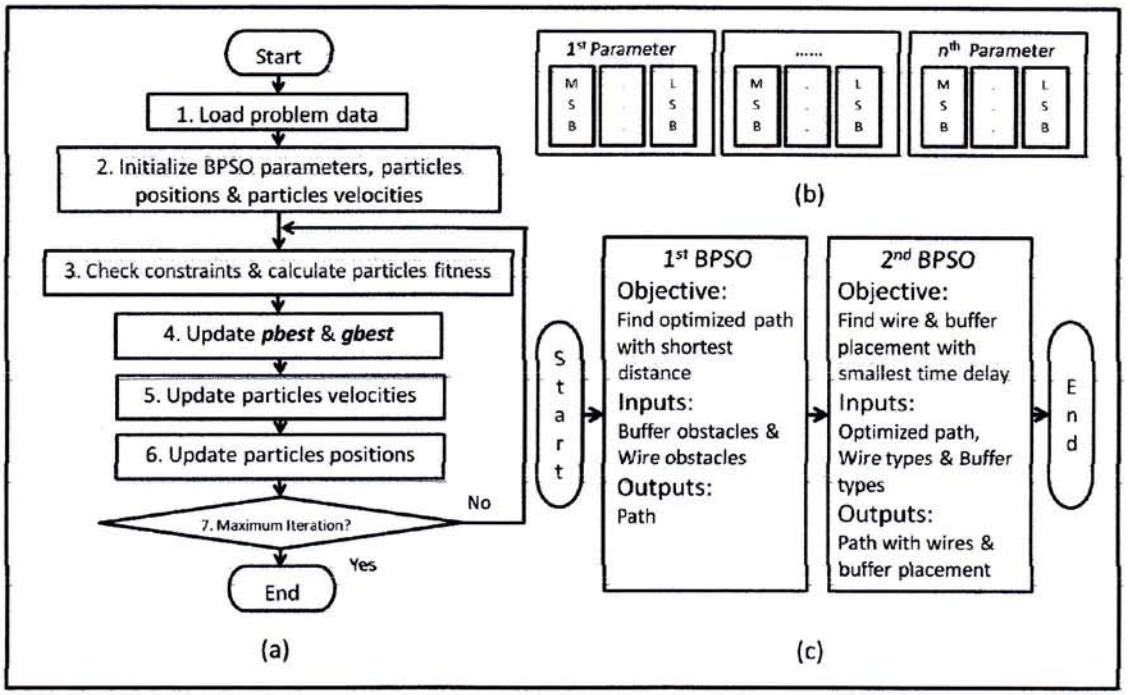


FIGURE 1. (a) BPSO algorithm, (b) BPSO modelling, (c) the proposed approach

best combination of wire and buffer selection and placement in order to obtain minimal time delay, as illustrated in Figure 1(c).

4. **Implementation of BPSO Algorithm in Finding the Shortest Path.** As shown in Figure 1(c), the first BPSO is used to find the shortest distance between source and sink. Here, $\boldsymbol{s} = [e_1\, e_2\, \ldots\, e_n]^T$, where e_n represent the location of a node in grid graph; e_{even} is measured from origin along x-axis and e_{odd} is measured from origin along y-axis. Combination of these consecutive nodes, if connected together, will form a complete path from source to sink. Note that each e is represented in binary value, as shown in Equation (8).

$$\boldsymbol{s} = \begin{bmatrix} e_1^1 & e_1^2 & \ldots & e_1^d \\ e_2^1 & e_2^2 & \ldots & e_2^d \\ & \ldots & & \\ e_n^1 & e_n^2 & \ldots & e_n^d \end{bmatrix} \tag{8}$$

The BPSO parameters are listed in Table 2. Fitness value of the particle can be calculated using Equation (1). During updating $\boldsymbol{s}$, there is a possibility of out of boundary condition where $\boldsymbol{s}$ value is larger than maximum grid graph size. If that happened, that particular particle is ignored for that iteration. The shortest path obtained from the simulation is then passed to the second BPSO for wire and buffer placement consideration.

The second BPSO is used to find the best combination of wire and buffer placement that minimizes time delay of the VLSI routing. The proposed model suggests $\boldsymbol{s} = [w_1\, w_2\, \ldots\, w_q]^T$, where w_q represents the type of wire with/without buffer for the respective to grid segment, q. For each w, 7 bits are used for the case study, where the first 3 bits represent the type of wire used, the 4^{th} bit is to indicate the use of buffer and the last 3 bits represent the type of buffer used. Figure 2 further illustrates the proposed model. Note that if buffer flag is 0, any value given by the bits represented buffer type is ignored. Out of boundary condition might happen if the type of buffer or wire used is less than 8. In this algorithm, the default type of wire, usually Type 0, will be inserted for

that grid segment if that condition occurred. The fitness of the particle can be calculated using Equations (2)-(5).

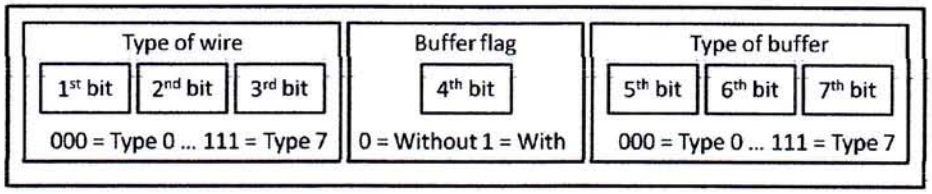
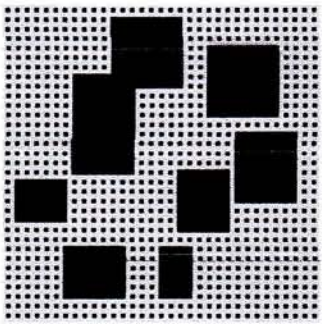


FIGURE 2. Binary representation for buffer placement

5. **Results and Discussions.** The results obtained based on a case study are as shown in Figure 3. Figure 3(a) consists of a grid-graph (blue colour), area of wire obstacle (black colour), area of buffer obstacle (green colour), and location of source and sink (red colour). The parameters setting of the case study shown in Figure 3(b) include a size of grid, maximum of dog-leg, number of buffer, number of obstacle, resistance for source, load capacitance for source, and coordinate for source and sink. This case study also considers six types of wire and buffer, as shown in Table 1. The BPSO parameters used in this study are shown in Table 2.



(a)

Parameters	Setting
Grid-Size	32 x 32
Maximum dog-leg	8
Number of buffer	4
Number of obstacle	4
Resistance source, Ω	140
Load capacitance, pF	0.002
Location of source	(3, 3)
Location of sink	(29, 26)

(b)

FIGURE 3. (a) Illustration of a case study, (b) parameters setting

TABLE 1. Wire parameters and buffer parameters

Category	**Wire ($z = w$)**						**Buffer ($z = b$)**					
Type	W0	W1	W2	W3	W4	W5	B0	B1	B2	B3	B4	B5
r_z (Ω)	140	180	162	145.8	131.2	118.1	140	180	162	145.8	131.2	118.1
c_z (pF)	2	22	24.2	26.6	29.3	32.3	2	22	24.2	26.6	29.3	32.3
d_b (ps)	Not Applicable						40	15	22.5	33.8	50.6	75.9

The experiment is executed 50 times with the average, standard deviation, mode, median, best, and worst result recorded. The average of time delay was 537.723 ps, the standard deviation was 20.104 ps, the mode was 563.461 ps, the median was 538.456 ps, the best result was 481.186 ps, and the worst result is 575.464 ps.

The best simulation result is shown in Figure 4, where Figure 4(a) is graphically illustrated the best wire placement (red colour) with the best buffer insertion (yellow colour). The wire is successfully placed to avoid the wire and buffer obstacles in the best condition. In detail, the segment info (the type of wire and buffer) of each subsequent segment is presented in Figure 4(b). Note that W2 means wire type 2 and W2B1 means wire type 2 with buffer type 1.

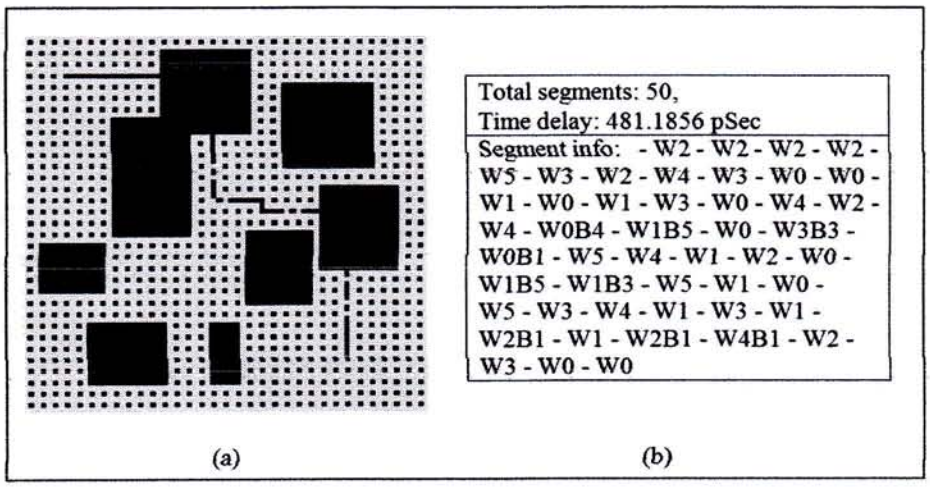


FIGURE 4. (a) Best simulation result, (b) grid info (with type of wire and buffer) for 50 runs

TABLE 2. BPSO parameters used in this study

Parameter	Value
Number of computation	50
Maximum iteration	1000
Number of particle	50
Cognitive parameter, c_1 and Social parameter, c_2	2.00
Inertia weight, ω	$0.9 \rightarrow 0.4$

6. **Conclusion.** This study presented an application of two-step BPSO in VLSI routing. Result obtained from the case study shows that the proposed approach has a potential for further extension. Further work includes optimization of more objectives functions, such as power dissipation and noise.

Acknowledgment. This work is financially supported by the Ministry of Higher Education Malaysia (MOHE) through the UTM GUP Research Funds (VOTE Q.J130000.7123.01J32 and Q.J130000.7107.01H29).

REFERENCES

[1] M. N. Ayob, Z. M. Yusof, A. Adam, A. F. Z. Abidin, I. Ibrahim, Z. Ibrahim, S. Sudin, N. Shaikh-Husin and M. K. Hani, A particle swarm optimization approach for routing in VLSI, *Proc. of the 2nd International Conference on Computational Intelligence, Communication Systems and Networks*, pp.49-53, 2010.

[2] J. Kennedy and R. Eberhart, Particle swarm optimization, *Proc. of the IEEE International Conference on Neural Networks*, pp.1942-1948, 1995.

[3] J. Kennedy and R. Eberhart, A discrete binary version of the particle swarm algorithm, *Proc. of IEEE International Conference on Systems, Man, and Cybernetics*, pp.4104-4108, 1997.

[4] M. H. Othman, A. F. Z. Abidin, A. Adam, Z. M. Yusof, Z. Ibrahim, S. M. Mustaza and L. Y. Yang, A binary particle wwarm optimization approach for routing in PCB holes drilling process, *Proc. of the 1st International Conference on Robotic Automation System*, pp.201-206, 2011.

[5] M. S. Muhammad, K. V. Selvan, S. M. W. Masra, Z. Ibrahim and A. F. Z. Abidin, An improved binary particle swarm optimization algorithm for DNA encoding enhancement, *Proc. of IEEE Symposium on Swarm Intelligence*, pp.60-67, 2011.

[6] H. Izakian, B. T. Ladani, A. Abraham and V. Snasel, A discrete particle swarm optimization approach for grid job scheduling, *International Journal of Innovative Computing, Information and Control*, vol.6, no.9, pp.4219-4233, 2010.

[7] N. K. Khalid, Z. Ibrahim, T. B. Kurniawan, M. Khalid and N. H. Sarmin, Function minimization in DNA sequence design based on continuous particle swarm optimization, *ICIC Express Letters*, vol.3, no.1, pp.27-32, 2009.

[8] Z. Chen, C. Yan, C. Wang, B. Wang and W. Liu, Paticle swarm optimization with probability statistic and mutation for biological multiple sequence alignment, *ICIC Express Letters*, vol.3, no.4(B), pp.1257-1262, 2009.

[9] Q. Hu, J. Tian and D. He, Plant leaf disease recognition based on PSO-SVM algorithm, *ICIC Express Letters*, vol.5, no.9(A), pp.3183-3188, 2011.