18

SoC Test Applications Using ACO Meta-heuristic

Hong-Sik Kim[1], Jin-Ho An[2] and Sungho Kang[1]
[1]*Department of Electrical and Electronic Engineering, Yonsei University*
[2]*Department of Electronic Engineering, Hoseo University*
Korea

1. Introduction

As the integrity of VLSI (very large scale integration) circuits increases, their test has become more complex and time consuming task so that test cost has been significantly increased. Recent SoC (system on chip) design and test environments have deteriorated this trend more significantly. For example, the traditional scan cell ordering problem for low power testing is a well known NP-complete problem and has become more time consuming job as circuit density increases. In addition, as the SoC design environment takes root in the semiconductor industry, the test sequence for each IP (intellectual property) core comes to influence the total test time and the total power consumption of the system chip under test. Recently hot spot induced by scan testing also should be considered during SoC testing in order to improve circuit reliability and reduce unexpected yield loss. Test scheduling, therefore, became more important in order to meet the design specifications, such as the test time constraint, the power consumption limitation, the thermal constraint, and so on. Finally, SoC design methodology requires more test storage since more IP cores are integrated in a SoC product so that the test compression techniques are widely used to reduce both the test time and the external ATE (automatic test equipment) memory/channel requirements. In many cases, the test compression efficiency depends on the given test cube set and as the size of test cube set increases in SoC test environment, it will take too much time to calculate the optimal seed set for the given test cube set in case of arithmetic built-in self test (ABIST) scheme.

ACO (ant colony optimization) meta-heuristic is an algorithm inspired by the real ant system in order to find the optimal solutions for TSP (travel salesman problem) which is well known to be an NP-complete problem and has been successfully applied to lots of NP-complete problems in various fields. In this chapter, we try to transform several important problems in the field of SoC testing into ACO applicable ones, which are solved by the ACO meta-heuristic. For the ACO-based test applications, three important test problems such as test scheduling, scan cell ordering, and test seed calculation for ABIST, are selected. Each problem has unique characteristics in order to be transformed into ACO applicable one, so that bundles of techniques have been devised and will be described in this chapter. According to the experimental results, the ACO meta-heuristic could considerably improve the quality and cost efficiency of SoC test methodologies such as test scheduling, scan cell ordering, and test seed calculation.

2. SoC test scheduling using ACO heuristic

The number of cores embedded in an SoC is increasing rapidly, and cores are more deeply embedded. Therefore, testing the cores by means of direct access through the SoC's I/O pins is almost impossible. In order to solve this problem, methods like the IEEE 1500 standard and Test Access Mechanism(TAM) have been proposed. An SoC test scheduling is a process to minimize the test application time of all built-in cores in the SoC under given constraints like TAM bandwidth and power budget. It includes the optimization of the test wrapper design, the assignment of TAM width to each core, and the determination of test start and finish time for each core.
In this section, we introduce an ant colony optimization (ACO)-based SoC test scheduling method including power and layout information. The proposed method efficiently combines the rectangle packing method with ACO and improves the scheduling results by dynamically choosing the TAM widths for cores and changing the testing orders. The power dissipation and adjacency of cores are incorporated for actual testing conditions.

2.1 ACO-based rectangle packing [Ahn, J.-H. & Kang, S., 2008]

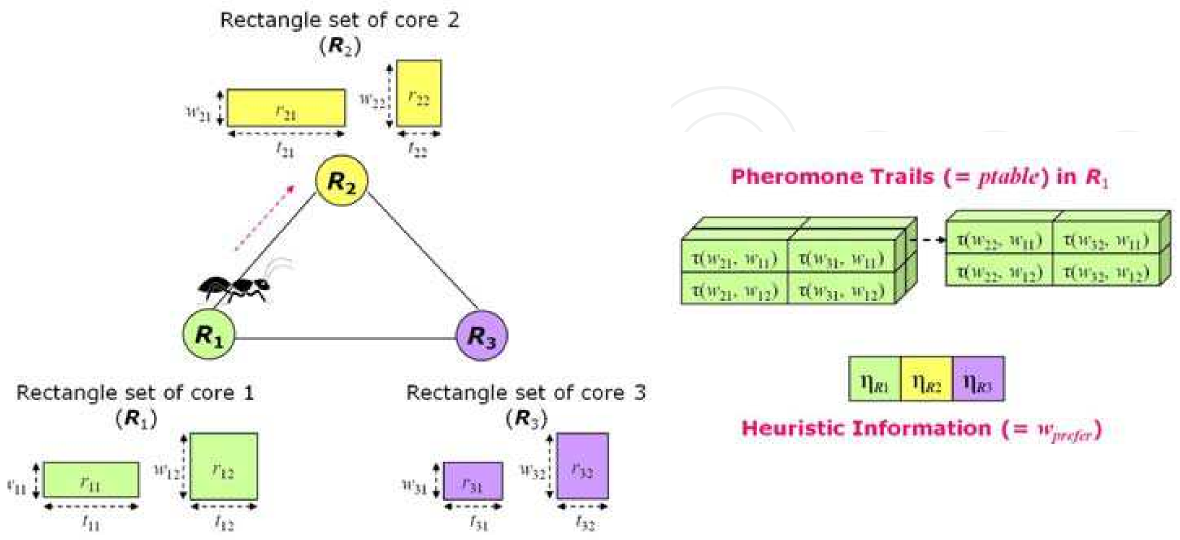
An SoC test scheduling problem can be formulated in terms of a 2-dimensional bin packing problem [Huang, Y. et al, 2001; Iyengar, V. et al, 2002]. The test time varies with TAM width in a staircase pattern, and the testing of a core is represented as a rectangle whose height indicates the TAM width assigned to that core and whose width denotes the test time of the core for the corresponding value of the TAM width. Thus, we can obtain a number of TAM width and test time combinations for the same core. Taken as a whole, a test scheduler chooses just one rectangle from the candidate rectangle set of each core and then packs it into a bin of a fixed height and an unlimited width until the bin is filled with rectangles of all cores embedded in SoC, while minimizing the overall width of the bin without overflowing the bin's height.
Now, we describe how the ACO algorithm can be implemented for a rectangle packing solution. A rectangle packing problem consists of two sub-parts: rectangle selection and rectangle packing. The ACO algorithm can be applied to part rectangle selection and/or rectangle packing. Through some experiments, we determined that ACO should be used only to select rectangles due to the computation time. Therefore, the rectangle packing method is based on the procedure used in [Ahn, J.-H. & Kang, S., 2006]. Before describing the implementation of ACO for the rectangle packing solution, some features related to ACO need to be clarified.
a. Definition of Pheromone Trail $\tau_i(k)$
We define the pheromone trail, $\tau_i(k)$, as the favorability of choosing k as the TAM width assigned to core i and calculate it as

$$\tau_i(k) = \sum_{j \in S} \tau(k_i, g_j), \quad 1 \le i,\ j \le m, \quad 1 \le 2^k, 2^g \le W, \tag{1}$$

where S denotes the cores already visited by the current ant, g_j is the TAM width selected for core j, m is the number of embedded cores, and W is the channel width of TAM. Consequently, (k_i, g_j) is the favorability of choosing k as the TAM width for core i when the TAM width assigned to core j is g.
b. Heuristic Variable η_i
The ACO algorithm combines pheromone information with heuristic values to find solutions, since heuristic parameters can help ACO be applicable to various conditions.

Here, we use a preferred TAM width for core i, $w_{prefer}(i)$, as the heuristic favorability, η_i. The calculation flow to seek $w_{prefer}(i)$ is shown in [Ahn, J.-H. & Kang, S., 2006].

c. *Stochastic Decision with $\tau_i(k)$ and η_i*



Rectangle set of core 2 (R_2)
Rectangle set of core 1 (R_1)
Rectangle set of core 3 (R_3)
R_1
R_2
R_3
Pheromone Trails (= $ptable$) in R_1
$\tau(w_{21}, w_{11})$
$\tau(w_{31}, w_{11})$
$\tau(w_{21}, w_{12})$
$\tau(w_{31}, w_{12})$
$\tau(w_{22}, w_{11})$
$\tau(w_{32}, w_{11})$
$\tau(w_{22}, w_{12})$
$\tau(w_{32}, w_{12})$
η_{R1} η_{R2} η_{R3}
Heuristic Information (= w_{prefer})

Fig. 1. ACO-based rectangle selection

The probability pi(k) that an ant will choose k for core i is given by

$$p_i(k) = \frac{\tau_i(k) \cdot \beta}{\sum_{x=1}^{W} (\tau_i(x) \cdot \beta)}, \tag{2}$$

where $\beta \geq 1$ if $w_{prefer}(i)=k$ and 1 otherwise.

d. *Policy for Pheromone Updating*

According to test scheduling results, pheromone trails are updated as follows:

$$\tau(k_i, g_j) = \rho \cdot \tau(k_i, g_j) + \Delta\tau(k_i, g_j), \tag{3}$$

where ρ is the evaporation parameter, which determines the speed of decay of pheromone; Δτ is the amount of pheromone deposited by the ant on the basis of the quality of the scheduling result and is defined as

$$\Delta\tau(k_i, g_j) = N_{k_i g_j} \cdot \lambda, \tag{4}$$

where $N_{k_i g_j}$ indicates how many times k of core i and g of core j go together in the best scheduling result S_{best}, and λ is the constant value to weigh.

The method of rectangle selection using ACO is explained in the following example. As shown in Fig. 1, let an SoC include three cores. The rectangle sets of cores, R_i ($1 \leq i \leq 3$), can be represented as nodes linked with one another in the graph. At first, an ant randomly selects a starting node and a rectangle in that node. In Fig. 1, we assume that the ant chooses R_1 as a starting node and rectangle r_{11}. Then, the ant can select R_2 or R_3 as a next node. After arbitrarily selecting a next node, according to pheromone trails (ptable in the figure) stored in R_1 and heuristic information, w_{prefer}, the ant chooses a rectangle in that node. For example, if R_2 becomes the next node, two pheromone trails, $\tau(w_{21}, w_{11})$ and $\tau(w_{22}, w_{11})$, and one

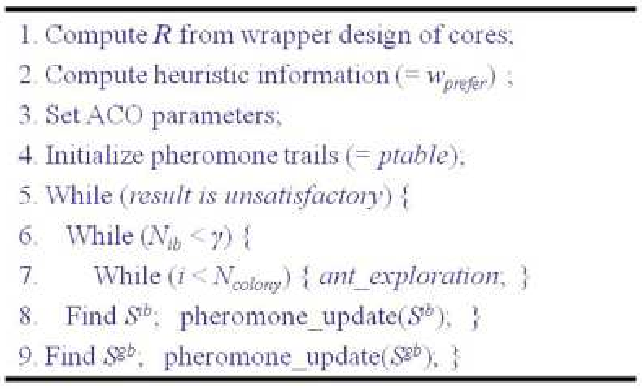
heuristic value, ηR_2, are used to choose a rectangle in R_2 by (5). As previously mentioned, $\tau(w_{21}$(or w_{22}), $w_{11})$ is the favorability of choosing w_{21}(or w_{22}) as the TAM width for core 2 when the TAM width assigned to core 1 is w_{11}, and ηR_2 is the preferred TAM width of core 2. In this example, we assumed that r_{21} in R_2 is selected by (2). The ant continues this process until all nodes are visited just once. After choosing all rectangles in R_i, the ant packs them using the method used in [Ahn, J.-H. & Kang, S., 2006] and gets a scheduling result. Figure 1 illustrates a scheduling result with three rectangles, r_{11}, r_{21}, and r_{32}. Finally, the result is based on updating pheromone values. If the result is better than the current best result, pheromone values, such as $\tau(w_{21}, w_{11})$ in R_1 and $\tau(w_{32}, w_{21})$ in R_2, can be reinforced.
In addition to basic ACO methods, we adopt several heuristic tips, such as iteration-best ant (S_{ib}), global-best ant (S_{gb}), and lower limit of the pheromone values (τ_{min}) [Glover, F. et al, 2003; Stutzle, T. et al, 2000]. These methods are efficient means to balance exploitation of the best solution found and exploration of the solution space. Furthermore, we assume that just one ant per colony can form a pheromone trail.

2.2 Thermal distribution consideration

During testing, the peak power cannot be over the power limit of the system [Cota, E. et al, 2003]. In addition, thermal management is also considered to reduce hot spots for the purpose of minimizing the local heating. In order to make thermal-aware scheduling, we added a thermal constraint by the analysis of geometrical adjacency of cores. It is for this reason that cooling effect diminishes when cores tested concurrently are geometrically close to each other. In [Liu, C. et al, 2005], the corresponding distance matrix is used to measure the adjacency of cores. However, as the matrix only represents the relative position within chip layout, the amount of adjacent area cannot be identified. Thus, we calculate the adjacency value as follows.

$$ADJ_{ij} = \sum \sqrt{(c_i(x) - c_j(x))^2 + (c_i(y) - c_j(y))^2}, \quad 1 \le i,\ j \le m, \tag{5}$$

where c_i denotes the unit cell of core i and $c_i(x)$ is the x-axis position of core i. c_j, $c_i(y)$, $c_j(x)$, and $c_j(y)$ can be defined in similar way. Position in X/Y-axis is normalized, and ranges from 0 to 99. In order to get the effective adjacency value, the value can increase iff the distance between two cells is less than 10% of the longest distance. Since we normalize the chip size as (100, 100), the longest distance is 140 or so.
a. Overall Test Scheduling Procedure



```
1. Compute $R$ from wrapper design of cores;
2. Compute heuristic information (= $w_{prefer}$) ;
3. Set ACO parameters;
4. Initialize pheromone trails (= $ptable$);
5. While ($result\ is\ unsatisfactory$) {
6.    While ($N_{ib} < \gamma$) {
7.        While ($i < N_{colony}$) { $ant\_exploration$; }
8.    Find $S^{ib}$;  pheromone_update($S^{ib}$);  }
9. Find $S^{gb}$;  pheromone_update($S^{gb}$); }
```

Fig. 2. ACO-based test scheduling procedure

```
Select a TAM width, $w_{selected}$, using ACO-based rectangle selection procedure
Initialize data structure of all cores;
While (there exist untested cores) {
    Select TAM position, $p$, that has the lowest current time;
    // Start of rectangle packing heuristics
    If (there remains TAM available) {
      If core $i$ satisfies the next conditions:
          - power budget and adjacency limit
          - largest test time among remained cores
          - smaller TAM width than currently available,
        Rectangle packing process with $w_{assigned}(i) = w_{selected}(i)$;
     Else
        If core $i$ satisfies the next conditions:
            - power budget and adjacency limit
            - largest test time among remained cores which are not over
                                the smallest time among cores tested currently,
          Find $w(i)$, where $w(i)$ is the highest Pareto-optimal width,
                such that requires smaller TAM width than currently available;
          Rectangle packing process with rectangle insertion in idle time;
          Set $w_{assigned}(i) = w(i)$;
        Else
            If core $i$ satisfies the next conditions:
                - scheduled at current time
                - largest test time with extended TAM width,
              Find $w(i)$, where $w(i)$ is the highest Pareto-optimal width,
                    such that extended TAM is smaller than TAM currently available;
              Rectangle packing process with increasing TAM widths to fill idle time;
              Set $w_{assigned}(i) = w(i)$;
    }
    Else  { Move to next time and update information; }
} // End of rectangle packing heuristics
Update the best test time;
```

Fig. 3. Ant exploration process

The ACO-based test scheduling procedure is given in Fig. 2. In Fig. 2, Ncolony denotes the number of ants in one colony, N_{ib} is the iteration-best ant number, γ is the waiting number until S_{gb} is used again, and ant_exploration means an ant's action. The ACO parameters include the variables previously mentioned, such as N_{colony}, γ, λ, β, ρ, τ_{min}, and so on. R in Fig. 2 (line 1) indicates the test wrapper set for embedded cores and wprefer is the preferred TAM width.

In the ant exploration procedure as shown in Fig. 3, an ant will search the solution space continuously with new r_i combinations. After choosing r_i for all cores, Each ant packs them while minimizing the idle space. As in [Ahn, J.-H. & Kang, S., 2006], we use the packing method, which is based on TAM_optimizer [Iyengar, V. et al. 2002] for its simplicity and feasibility. Finally, if the packing result of the ant is better than the current best result, the scheduler updates the best test time. $w_{selected}$ is the TAM width from the ACO-based rectangle selection process and $w_{assigned}$ is the TAM width assigned at the end. If a core doesn't satisfy the power budget, the acceptable maximum power consumption in a whole chip level, and the adjacent limit, the acceptable adjacency value of cores tested concurrently, the core will not be selected by the ant. $w_{assigned}$ will be reinforced at the end of the ant exploration process, if the packing result is excellent.

3. Experimental results

We simulated three ITC'02 benchmark circuits to evaluate the proposed scheduling algorithm. The final results reported in this section are based on the following ACO parameter values:

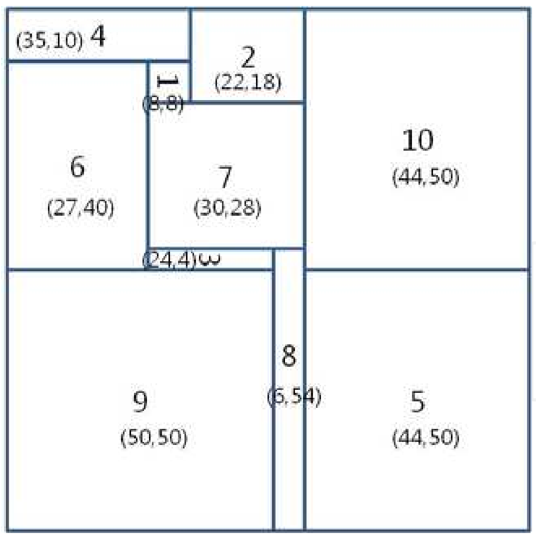
N_{colony}: 10, γ: 10, ρ: 0.9~0.96, τ_{min}: 2.0, $\tau(0)$: 20.0, β : 10~20, λ: 0.2,

where $\tau(0)$ is an initial value of a pheromone trail. To obtain parameter values that achieve good performance, tests using circuits with different sizes and structures are required. The parameter values used here are chosen for balancing the test application times with the calculation times through some experiments.

SoC Name	Power Constrants	TAM Width					
		W=32			W=16		
		[Iyengar, V. et al. 2002]	[Huang, Y. et al 2002]	Proposed	[Iyengar, V. et al. 2002]	[Huang, Y. et al 2002]	Proposed
p22810	No Power Constraint	**246150**	**223462**	**238109**	**452639**	**446684**	**441408**
P9379	No Power Constraint	**975016**	**900798**	**896835**	**1851135**	**1791860**	**1783397**
d695	No Power Constraint	**23021**	**21389**	**21429**	**43723**	**42716**	**42315**
	Pmax=2000	NA	**24171**	**21437**	NA	**43221**	**42351**
	Pmax=1500	NA	**27573**	**23097**	NA	**45560**	**42587**

Table 1. Scheduling Results with Various Power Constraints

Table 1 displays the results of an experiment in which various power constraints were used for the core test. The given TAM width is either 32 or 16 bits. First, in the experimental results without power budgets, we compare the test times of the proposed method with those of the method presented in [Iyengar, V. et al. 2002] and [Huang, Y. et al 2002] using the bin packing algorithm for test scheduling. As a result, compared with [Iyengar, V. et al. 2002], the test time reduction ratio increases by up to 8% at W=32 and 3% at W=16 on average. However, our results are almost similar to [Huang, Y. et al 2002]. Next, we show the experimental results incorporating power constraints on the power consumption model in [Huang, Y. et al 2002]. As the results demonstrate, though the reduction ratio is somewhat variable on a case-by-case, the proposed algorithm shows good performance in the main. When P_{max} is set to 2000, the test time reduction ratio goes up to 16% at W=32 and 6% at W=16.



(a)

SoC Name	W=32				
	100%	80%	60%	50%	30%
d695	**21437**	**22350**	**22403**	**22571**	**23121**
	W=16				
	100%	80%	60%	50%	30%
	42351	**42481**	**42481**	**42786**	**43456**

(b)

Table 2. Scheduling Results with Thermal Management

Finally, we introduce the scheduling results using thermal management in Table 2. As layout information on benchmark circuits is not available, we make it freely in terms of the numbers of scan flip-flops, inputs, outputs, and bidirectional ports of cores. The example floorplan of d695 is shown in Table 2 (a). Each rectangle denotes a core and center number in a rectangle is the core ID. Numbers in parentheses mean the scaling size of a core, (X-axis size, Y-axis size). The results are shown in Table 2 (b). To evaluate the scheduling results considering the core adjacency, we set the adjacency limit, ADJmax, to 30~100% of the maximum adjacency value among cores.

Experimental results show that the adjacency constraint partly influences the scheduling results as we expected. When the adjacency limit goes down to 30 %, the scheduling time extends by up to 7%. The proposed algorithm can be utilized to estimate the overall test application time just with the number and size of flip-flops, memory and rough floorplan information.

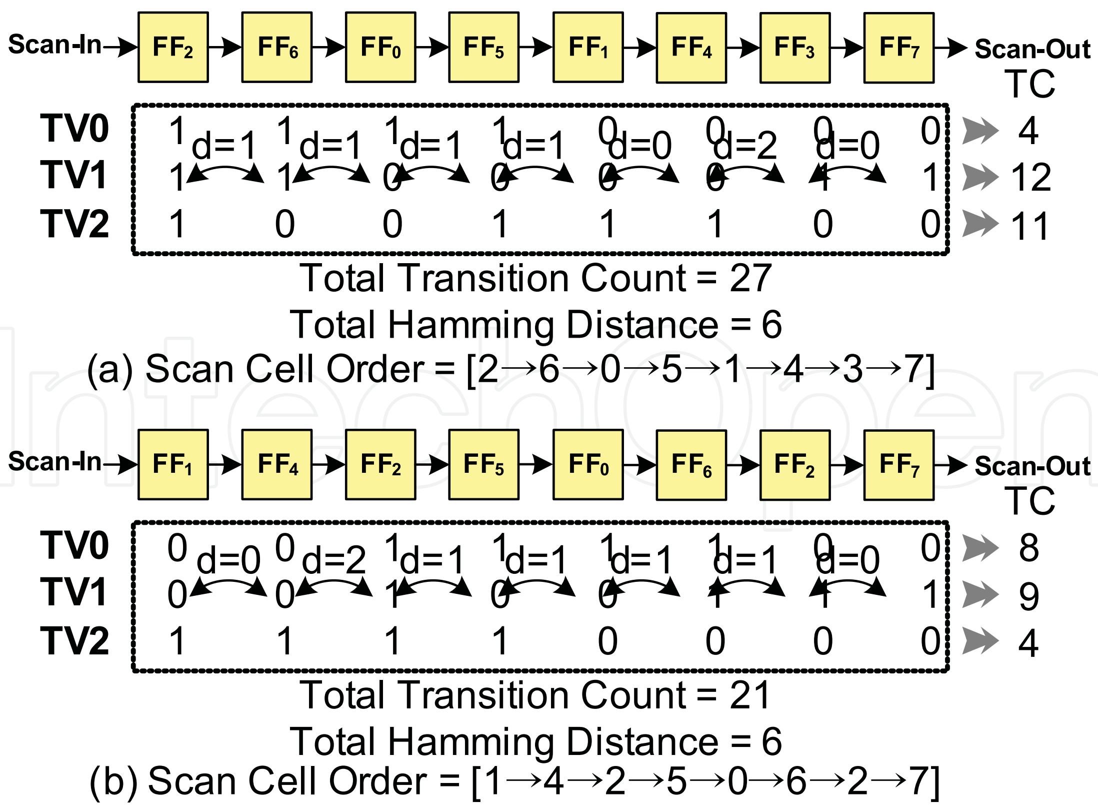
3. Scan cell ordering using ACO heuristic

The scan-based test structure is widely used for its high controllability and observability, which is obtained by direct access to the memory elements in the circuit. Power consumption during a scan test is much greater than that of normal operation, because all scan flip-flops are clocked during shift operations, and a much larger percentage of the flip-flops will change values in each clock cycle. Excessive power consumption during a test can cause several problems: circuit damage, yield loss, decreased system reliability, and increased product costs. Since the switching activities in scan flip-flops are the dominant source of test power consumption, the number of transitions during a scan test should be minimized to prevent these problems. Scan cell ordering methods have been proposed to reduce the switching activity in a scan-chain during scan test. Genetic algorithm (GA) is used to determine an optimized scan cell order [Jelodar, M. S. et al, 2006; Giri, C. et al, 2005]. However, the power reduction rate of these methods decreases as the number of scan-cells increases.

We propose an efficient scan-cell ordering method using an ACO meta-heuristic [Dorigo, M. et al, 1999] to reduce the transition count during scan testing. The ACO decision-making-based experiential probability can provide a scan-cell order optimized with respect to both the sum of the Hamming distances between successive test vector columns and the total transition count during a scan test. According to the experimental results based on ISCAS 89 benchmark circuits, the proposed scan cell ordering methodology could reduce considerable power consumption compared to the previous works.

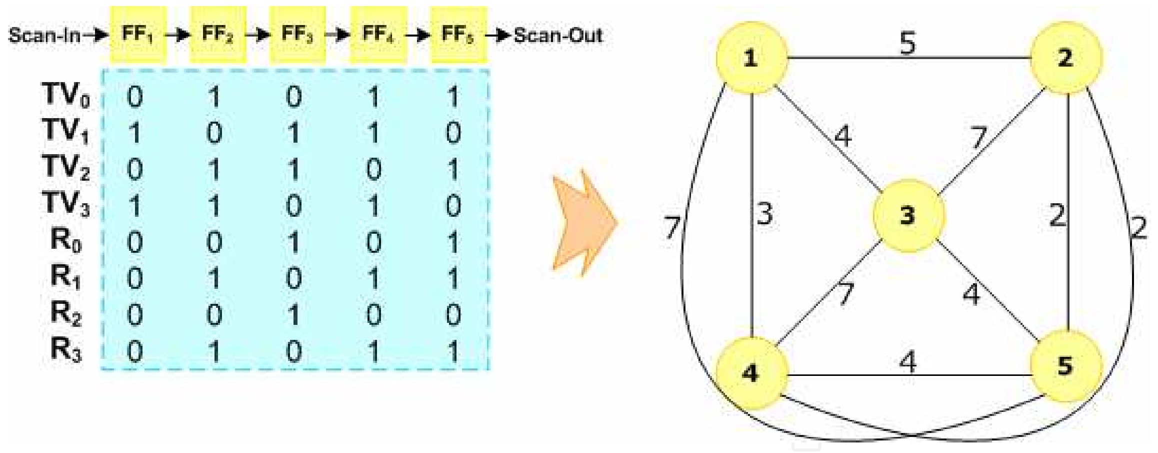
3.1 Proposed scan cell ordering methodology

The basic idea of the proposed method is to arrange scan-cells to minimize the sum of Hamming distances between successive test vector columns. The total transition count can be decreased as the sum of Hamming distances between test vector columns decreases, because Hamming distances between test vector columns represent the number of transitions caused by shift operations during a scan test. As shown in Fig. 4, however, minimizing the sum of Hamming distances does not always result in minimum total transition count. Therefore, the order of scan-cells must be optimized with respect to both the sum of the Hamming distances and the total transition count. We use an ACO meta-heuristic to find an optimal scan-cell order by formulating the ordering problem as a travelling-salesman problem (TSP). ACO is an algorithm inspired by the behaviour of real ants using pheromone trails to communicate. The pheromone trails are distributive and contain numerical information which the ants use to find solutions probabilistically.



Scan-In
FF2
FF6
FF0
FF5
FF1
FF4
FF3
FF7
Scan-Out
TC
TV0 1 1 1 1 0 0 0 0 4
TV1 1 1 0 0 0 0 1 1 12
TV2 1 0 0 1 1 1 0 0 11
d=1 d=1 d=1 d=1 d=0 d=2 d=0
Total Transition Count = 27
Total Hamming Distance = 6
(a) Scan Cell Order = [2→6→0→5→1→4→3→7]
Scan-In
FF1
FF4
FF2
FF5
FF0
FF6
FF2
FF7
Scan-Out
TC
TV0 0 0 1 1 1 1 0 0 8
TV1 0 0 1 0 0 1 1 1 9
TV2 1 1 1 1 0 0 0 0 4
d=0 d=2 d=1 d=1 d=1 d=1 d=0
Total Transition Count = 21
Total Hamming Distance = 6
(b) Scan Cell Order = [1→4→2→5→0→6→2→7]

Fig. 4. Total transition count and hamming distance during scan shift



	FF1	FF2	FF3	FF4	FF5
TV_0	0	1	0	1	1
TV_1	1	0	1	1	0
TV_2	0	1	1	0	1
TV_3	1	1	0	1	0
R_0	0	0	1	0	1
R_1	0	1	0	1	1
R_2	0	0	1	0	0
R_3	0	1	0	1	1

Fig. 5. An example of constructing a graph based on hamming distance

The probability that a solution candidate is chosen to construct a solution is analogous to the amount of pheromone in trails formerly laid by the ants. The objective of scan-cell ordering using ACO is to find the minimum transition Hamiltonian cycle on the graph G=(C, L) where C={c_1, c_2, ... , c_{Nc}, N_c=the size of scan chain} is a set of scan-cell bit positions in a target scan chain and L={l_{ij} | (c_i, c_j)∈C', C'=subset of a Cartesian product C×C}, [L]≤N_c^2 represents the Hamming distance between test vector column i and j corresponding to scan-cell bit positions i and j, respectively. Fig. 5 shows an example of constructing a graph using a set of test vectors and responses.

Scan-cell ordering problems can be characterized as follows using ACO:

- J_{cicj} is a Hamming distance between scan-cell bit positions i and j.
- A={a_{ij} | (c_i, c_j) C'}, [A]≤N_c^2 is a local information table between each scan-cell bit position. a_{ij} can be calculated as 1/J_{cicj}. So this value becomes larger as the Hamming distance decreases.
- J_k is a total transition count estimated using the order constructed by an artificial ant k.
- τ_k is a pheromone trail stored by an ant k. It can be calculated as 1/J_k. When the solution is constructed more efficiently, larger pheromone trails can be stored.
- τ={τ_{ij} | (c_i, c_j)∈C'}, [τ]≤N_c^2 represents pheromone trails stored on the arc between scan-cell bit positions i and j. It can be calculated as follows: $\tau_{ij}=(1-\rho)\cdot\tau_{ij}+\rho\cdot\tau_k$. The variable $\rho\in(0, 1]$ is the pheromone trail evaporation coefficient. The evaporation of pheromone trails takes place to avoid overly-rapid convergence of the algorithm toward a sub-optimal solution.
- p_{ijk} = $[\tau_{ij}]\alpha \cdot [a_{ij}]\beta$ is the probability with which an ant k positioned in bit position i chooses the bit position j. The parameters α and β determine the relative weight of the total transition count and the sum of Hamming distances, respectively.

According to these definitions, the proposed scan-cell ordering proceeds as shown in Fig. 6. Once the test vector columns are ordered by the proposed scan-cell ordering method, the rows of the test vectors are ordered again. A transition occurs when the MSB (most significant bit) of a test response to be scanned-out is different from the LSB (least significant bit) of the next test vector to be scanned-in. Since this transition is propagated throughout the whole scan chain, the number of transitions caused by this difference is equal to the length of a scan chain. To reduce these transitions, the consecutive test vectors are, therefore, arranged by the simple condition of whether the LSB of one vector is equal to the MSB of its predecessor's response.

```
Procedure ScanCellReordering()
   ants_generation();
   while (current iteration != target_iteration)
      initialize_ants();
      while (available_ants)
        while (current_state != target_state)
          P = compute_transition_probabilities();
          next _state = apply_ant_decision_policy(P);
          move _to_next_state(next_state);
        end while
      end while
      pheromone _update_on_the_visited_arc();
   end while
End procedure
```

Fig. 6. Proposed scan cell ordering heuristic

3.2 Experimental results

Experiments were performed to compute scan test power reduction rates on the ISCAS'89 benchmark circuits. Test vectors were generated from the full-scan versions of the circuits using the Synopsys Design Analyzer with two vector filling methods (random-fill (R-fill) and minimum-transition fill (MT-fill)) and two compaction types (high compaction and no compaction). The heuristic parameters of ACO were set to a=1, β=5 and ρ=0.5. All the experiments were carried out for 100 ant-cycles and averaged over 10 trials.

Circuits	No. of scan cells	Total Transition Reduction Rate					
		R-fill & Compaction high			R-fill & No-Compaction		
		[Jelodar, M. S. et al, 2006]	[Giri, C. et al 2005]	Proposed	[Jelodar, M. S. et al, 2006]	[Giri, C. et al 2005]	Proposed
s298	14	11.51%	11.81%	16.30%	7.65%	13.07%	15.02%
s1423	74	17.01%	18.92%	24.19%	9.74%	11.39%	15.46%
s5378	179	12.41%	13.53%	25.66%	6.73%	13.53%	18.34%
s9234	211	10.41%	11.34%	29.19%	7.66%	8.19%	20.40%
s13207	638	4.02%	4.25%	27.34%	3.12%	3.34%	20.02%
s15850	534	5.15%	5.83%	26.84%	3.58%	4.03%	19.06%
s35932	1728	2.87%	3.14%	42.95%	2.33%	2.69%	27.13%
s38417	1636	1.76%	2.41%	18.44%	1.04%	1.03%	10.79%
s38584	1426	2.58%	2.98%	24.14%	1.46%	1.55%	14.68%

Table 3. Comparison using random-filled test vectors

Circuits	No. of scan cells	Total Transition Reduction Rate					
		MT-fill & Compaction high			MT-fill & No-Compaction		
		[Jelodar, M. S. et al, 2006]	[Giri, C. et al 2005]	Proposed	[Jelodar, M. S. et al, 2006]	[Giri, C. et al 2005]	Proposed
s298	14	17.80%	16.98%	26.78%	9.42%	10.57%	19.22%
s1423	74	16.18%	19.16%	19.22%	27.28%	28.58%	41.59%
s5378	179	11.35%	19.61%	25.20%	20.75%	18.11%	41.10%
s9234	211	4.63%	9.43%	21.01%	10.15%	13.81%	31.48%
s13207	638	2.60%	3.78%	19.79%	6.38%	7.86%	48.67%
s15850	534	2.27%	4.81%	17.52%	8.53%	10.65%	42.41%
s35932	1728	0.22%	6.00%	26.72%	3.47%	29.43%	82.50%
s38417	1636	8.79%	15.67%	38.86%	21.78%	29.82%	84.19%
s38584	1426	4.40%	4.30%	29.07%	2.36%	6.23%	51.55%

Table 4. Comparison using minimum transition-filled test vectors

Tables 3 and 4 include the comparison of the proposed method with previous works [Jelodar, M. S. et al, 2006; Giri, C. et al 2005] using R-filled and MT-filled test vectors, respectively, in terms of the transition reduction rates during scan testing. As can be seen in Table 3 and 4, the proposed method gave 14% to 42% power reduction for R-filled test vectors and 19% to 84% power reduction for MT-filled test vectors. The maintenance of the high reduction rates for the circuits that have a large number of scan-cells demonstrates that the proposed method can efficiently find an optimized scan-cell order regardless of circuit size. It shows, too, that power reduction rates obtained by the proposed method are superior to those of the previous works in all cases.

4. ABIST triplet calculation using ACO heuristic

In arithmetic built-in self test scheme [Gupta, S. et al, 1999; Chiusano, S. et al, 2000; Manich, S. et al 2007], the accumulator with an n-bit adder is used to generate a sequence of binary patterns by continuously accumulating a constant as shown in Figure 4-1. First, the initial vector is loaded into the accumulator register, and new test patterns are generated as a result of the iterative addition of the incremental value to the initial vector. Consecutive test patterns are described by the following equation: $T_0 = S_j$, $T_i = T_{i-1} + I_j$, ($i = 1, 2, \dots, L_j$), where S_j is the j-th initial vector, I_j is the increment value, T_i is the test pattern by j-th initial vector at the i-th cycle, and L_j is the total number of cycles. The combination of values (S_j, I_j, L_j), called as a triplet, will generate $L_j + 1$ test patterns. If n triplets are used to achieve target fault coverage, then random patterns will be applied to the CUT (circuit under test). Without a loss of generality, the number of the triplets, n, should be decreased without any loss of target fault coverage to reduce the test application time.
The proposed methodology finds an optimal triplet by selecting the best solution in terms of fault coverage among the solutions of the ant agents generated by the ACO heuristic and by a local search algorithm. Once ants have completed their solution construction, the solutions are taken to their local optimum by the application of a local search routine. Then

pheromones are updated on the arcs of the locally optimized solutions. The local search procedure to send local solutions into the regions of the global ants is performed. The local search starts by generating a new triplet by cross-mutating two solutions that have been randomly selected from the global ants. Then the fault coverage of the random test patterns generated by the new triplet is calculated by fault simulation. If the fault coverage of the new triplet is higher than the minimum fault coverage of the global ant set, then the new triplet is sent to the region of the global ants, and the ant of which the fault coverage was the minimum is dropped from the global ant set. This process is repeated until a predefined number of successive iterations with no improvement in fitness values have been reached. Elitist ant system was used for the best ant selection.

ACO based heuristic and local search method were implemented by C language, and Hope fault simulator [Lee, H. K. & Ha, D. S., 1991] was used for the fault simulation of the test patterns generated by each ant agent. Experiments were performed on both largest ISCAS 85 and ISCAS 89 benchmark circuits in terms of the fault coverage, the total test length, and the number of triplets. In most of the benchmark circuits, the proposed scheme showed the highest fault coverage, fewest triplets, and shortest test lengths. Since the memory and test channel bandwidth requirements of ATE directly depend on the number of triplets, even though the proposed scheme requires slightly longer test time in some benchmark circuits, the proposed scheme could reduce the test cost significantly. On average, the number of triplets of the proposed scheme was smaller than the previous schemes by about 0.8~63.0% for all the benchmark circuits. Reduced number of triplets can decrease the memory requirements of external ATE so that the proposed methodology can guarantee a considerable reduction of test costs. In addition, the proposed methodology reduced the average test length considerably compared to two previous schemes, respectively. The proposed scheme, therefore, can shorten the test application times significantly.

4.1 Proposed triplet calculation methodology for ABIST

The proposed methodology finds an optimal triplet by selecting the best solution in terms of fault coverage among the solutions of the ant agents generated by the ACO heuristic and by a local search algorithm. Once ants have completed their solution construction, the solutions are taken to their local optimum by the application of a local search routine. Then pheromones are updated on the arcs of the locally optimized solutions. In case of triplet calculation application, fault coverage is not available for heuristic information since new fault simulation is required for each generated ant so that the number of fault simulation increases impractically. Therefore, in the proposed methodology, instead of using heuristic information, a new local search method is applied to prevent local minimum problem.

Procedural steps of the proposed scheme are summarized Fig. 7. First, an initial solution is created, and the values of pheromones and other variables are initialized. The initial solution consists of an initial vector, increment data, and test length, and is used as a seed to generate ant agents in the next processes. First, an initial vector, $S_j = (s_1, s_2, s_3, \ldots, s_n)$, where n is the total number of primary inputs of a CUT, is randomly generated. If the initial value is odd, then S_j is used for the increment data, I_j. If the initial value is even, $S_j + 1$ is used for I_j to ensure the generation of the maximum number of random test patterns. Random test patterns are generated by iteratively accumulating the initial vector, S_j, with the increment data, I_j, and are fault simulated until a predefined number of successive test patterns with no improvement in fault coverage have been reached. Then the final number of random test patterns will be the value for L_j.

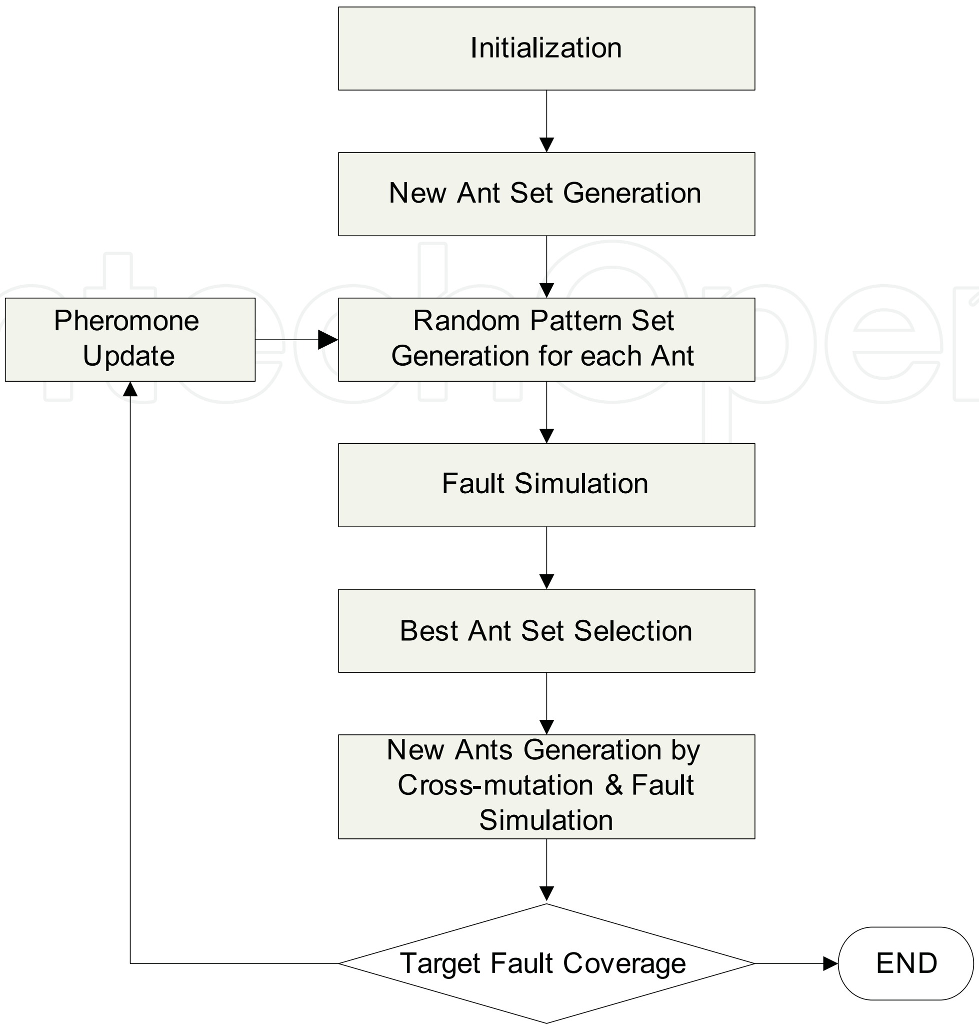


Fig. 7. Proposed Triplet Calculation Flow

New ants are created according to the fitness function, and then fault simulation is performed for random test patterns generated by each ant to achieve its fault coverage. The purpose of Step 2 is to create N new global ants using a Roulette wheel selection scheme by the following fitness function,

$$p_i(k) = \tau_i(k) \,/\, \sum_{k=1}^{N} \tau_i(k), \tag{1}$$

where $1 \le i \le n$, $\tau_i(k)$ is the pheromone value in the i-th primary input of ant k. $p_i(k)$ denotes the probability that the k-th ant lets the i-th primary input of CUT have a logic value of 1.

After best ant set is selected, new ant agents are generated by cross-mutating randomly selected ants from the best ant set. Then the fault coverage of the random test patterns generated by the new ant is calculated by fault simulation. If the fault coverage of the new ant is higher than the minimum fault coverage of the best ant set, then the new ant is sent to the region of the best ants, and the ant of which the fault coverage was the minimum is dropped from the best ant set. If the fault coverage of the ant set meets the target fault coverage, the process stops. Otherwise, the trails of the best solution found in the present cycle are reinforced and the process repeats. At the end of every local search cycle, the pheromone trails laid on the solutions found in this cycle are reinforced to facilitate the search around the specified point for the best solutions. The pheromone value in the *i*-th primary input of an ant *k* is calculated by the following equation,

$$\tau_i(k) = \tau_i(k) + \sum_{k}^{N} \Delta\tau_i(k), \tag{2}$$

where $\Delta\tau_j(k) = 1/FC_k$. FC_k means the fault coverage of an ant *k*. For this process, Elitist ant system has been used. In our previous work [Kang, S. et al, 2008], the normal ant system was used for the pheromone update. According to the experimental results, with the new ant system, the proposed system could considerably improve the memory requirement and test application time reduction, which will be discussed in the following sub-section.

4.2 Experimental results

The ACO based heuristic and GA-based local search method were implemented by C language, and Hope fault simulator was used for the fault simulation of the test patterns generated by each ant agent. Experiments were performed on both ISCAS 85 and ISCAS 89 benchmark circuits. In case of ISCAS 89 benchmark circuits, scanned versions have been used. Table 5 shows the comparison between the proposed scheme and other arithmetic BIST schemes [Chiusano, S. et al, 2000; Manich, S. et al, 2007; Kang, S. et al, 2008] in terms of fault coverage (FC), the number of triplets (M), and total test length (L). In case of [Kang, S. et al, 2008] and the proposed method, the same test patterns were used so that the fault coverage results of the two schemes were the same. In most of the benchmark, the proposed scheme showed the highest fault coverage, fewest triplets, and shortest test lengths. On

Circuit	[Chiusano, S. et al 2000]			[Manich, S. et al, 2007]			[Kang, S. et al 2008]			Proposed		
	FC	M	L	FC	M	L	FC	M	L	FC	M	L
c432	99.0	2	243	99.2	2	**111**	**100**	**1**	194	**100**	2	102
c499	98.8	2	369	**100.0**	2	361	**100.0**	**1**	**319**	**100.0**	2	**323**
c880	**100.0**	**2**	2104	**100.0**	**2**	1112	**100.0**	**2**	504	**100.0**	**2**	**472**
c1355	99.4	2	1151	**100.0**	2	1409	**100.0**	**1**	827	**100.0**	**1**	**801**
c1908	99.6	**2**	3773	99.9	**2**	3198	**100.0**	**2**	1819	**100.0**	**2**	**1724**
c2670	95.6	66	10179	97.6	33	2016	**100.0**	28	**1474**	**100.0**	**25**	1483
c3540	96.0	2	3467	97.1	2	2167	**99.8**	**2**	1696	**99.8**	**2**	**1521**
c5315	98.8	2	1324	99.9	2	1453	**100.0**	**1**	925	**100.0**	**1**	**824**
c6288	99.6	2	56	99.7	2	66	**100.0**	**1**	48	**100.0**	**1**	**43**
c7552	96.0	128	4000	**98.9**	**20**	2918	98.7	**20**	2748	98.7	**20**	**2654**
s1196	**100.0**	**8**	10000	**100.0**	**8**	2086	99.0	8	**1731**	99.0	**7**	1821
s1238	94.7	8	7256	98.9	8	4977	**99.0**	**6**	2622	**99.0**	**6**	**2211**
s1423	99.0	**6**	3100	99.5	**6**	**630**	**100.0**	**6**	921	**100.0**	6	879
s5378	**99.0**	14	5000	98.9	14	2078	**99.0**	**3**	3568	**99.0**	5	**2021**
s9234	NA	NA	NA	91.4	73	14803	**93.5**	**5**	**5826**	**93.5**	6	6712
s13207	NA	NA	NA	97.2	42	14476	**98.5**	**9**	8248	**98.5**	**9**	**8123**
s15850	NA	NA	NA	96.7	118	14902	**96.6**	**8**	5239	**96.6**	**8**	**5039**
s38417	NA	NA	NA	99.5	224	10665	**99.5**	87	22021	**99.5**	**86**	**22010**
s38584	NA	NA	NA	95.9	71	8449	**97.4**	43	**4228**	**97.4**	**41**	4321

Table 5. Comparison of the proposed methodology with other arithmetic BIST schemes

average, the number of triplets of the proposed scheme was smaller than the previous schemes by about 0.8% ~ 63.0%. Fewer triplets entail reduced memory requirements and IO channel bandwidth of the external ATE so that the proposed methodology can guarantee a considerable reduction of test costs. In addition, the proposed methodology reduced the average test length by about 2.0% ~ 28.2%. The test length determines the total test time so that the proposed scheme can shorten the test application times significantly in arithmetic BIST based SoC testing.

5. Concluding remarks

In this chapter, three SoC test issues such as the low power test scheduling, the low power scan cell ordering, and the seed calculation for arithmetic BIST, are considered for ACO applications. Unique techniques and problem transformation to apply ACO meta-heuristic to each test issue are described and the experimental results are provided. According to the experimental results, ACO meta-heuristic based test methodologies for SoC test scheduling, scan cell ordering and test seed calculation could improve test efficiency and reduce the test cost significantly compared to the previous methodologies.

6. References

Ahn, J.-H. & Kang, S. (2006). Test Scheduling of NoC-based SoCs Using Multiple Test Clocks, *ETRI Journal*, vol. 28, no. 4, Aug. pp. 475-485.

Ahn, J.-H & Kang, S. (2008). NoC-Based SoC Test Scheduling Using Ant Colony Optimization, *ETRI Journal*, vol. 30, no. 1, pp. 129-140.

Chiusano, S.; Prinetto, P. & Wunderlich, H. J. (2000). Non-intrusive BIST for system-on-a-chip, *Proc. of International Test Conference*, pp. 644-651.

Cota, E.; Carro, L. Wagner, F. & Lubaszewski, M. (2003). Power-Aware NoC Reuse on the Testing of Core-Based Systems, *Proc. of International Test Conference*, vol. 1, pp. 612-621.

Giri, C.; Kumar, B.N. & Chattopadhyay, S. (2005). Scan flip-flop ordering with delay and power minimization during testing, *Proc. of IEEE Indicon 2005 Conference*, pp. 467-471.

Dorigo, M.; Caro, G. (1999). Ant colony optimization: a new meta-heuristic, *Proc. of Evolutionary Computation*, Vol 2, pp. 1470-1477.

Glover, F.; Kochenberger, G. (2003). *Handbook of Metaheuristics*, Kluwer Academic Publishers.

Gupta, S.; Rajski, J. & Tyszer, J. (1996). Arithmetic additive generators of pseudo-exhaustive test patterns, *IEEE Trans. on Computers*, vol. 45, no. 8, pp. 939-949.

Huang, Y.; Cheng, W.-T. Tsai, C.-C. Mukherjee, N. Samman, O. Zaidan, Y. & Reddy, S.M. (2001). Resource Allocation and Test Scheduling for Concurrent Test of Core-Based SOC Design, *Proc. of Asian Test Symposium*, pp. 265-270.

Huang, Y.; Reddy, S.M. Cheng, W.-T. Reuter, P. Mukherjee, N. Tsai, C.-C. Samman, O. & Zaidan, Y. (2002) Optimal Core Wrapper Width Selection and SOC Test Scheduling Based on 3-D Bin Packing Algorithm, *Proc. of International Test Conference*, pp. 74-82.

Iyengar, V; Chakrabarty, K. & Marinissen, E.J. (2002). On Using Rectangle Packing for SOC Wrapper/TAM Co-Optimization, *Proc. of VLSI Test Symposium*, pp. 253-258.

Jelodar, M.S.; Mizanian, K. (2006). Power aware scan-based testing using genetic algorithm, *Proc. of IEEE CCECE/CCGEI*, pp. 1905-1908.

Kang, S.; Kim, H. –S. & Kim, H. (2008). Ant Colony Based Efficient Triplet Calculation Methodology for Arithmetic Built-in Self Test, *ELEX*, Vol. 5, No. 20, pp. 877-881.

Lee, H. K. & Ha, D. S. (1991). An Efficient Forward Fault Simulation Algorithm Based on the Parallel Pattern Single Fault Propagation*, Proc of International Test Conference*, pp. 946-955.

Liu, C.; Veeraraghavan, K. & Iyengar, V. (2005). Thermal-Aware Test Scheduling and Hot Spot Temperature Minimization for Core-Based Systems. *Proc. of the 20th IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems*, pp. 552-560.

Manich, S.; Garcia-Deiros, L. & Figueras, J. (2007). Minimizing test time in arithmetic test-pattern generators with constrained memory resources, *IEEE Trans. On Computer-Aided Design of Integrated Circuits and Systems*, Vol. 26, No. 11, pp. 2046-2058.

Stutzle, T.; Hoos, H. (2000). MAX-MIN Ant System, *Future Generation Computer Systems*, Vol. 16, No. 8, pp. 889-914.