GA^2CO: Peak Temperature Estimation of VLSI Circuits

Ya-Hsin Chang, Chun-Yao Wang, and Yen-An Chen
Dept. of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, R.O.C.
g9662520@oz.nthu.edu.tw, wcyao@cs.nthu.edu.tw, g9762639@oz.nthu.edu.tw

***Abstract*—With the continuing increase of chip density and the shrinkage of feature size of transistor in VLSI circuits, high temperature has become a concerned issue. High temperature not only decreases the functionality and reliability of chips, but also causes high package cost in order to cool down the system. For design consideration, one important issue related to temperature is how hot the chip may be. Thus, this paper investigates on the lower bound of peak temperature of a packaged chip and on the patterns that cause such bound. Two algorithms, Genetic Algorithm and Ant Colony Optimization, are applied for finding this lower bound of peak temperature. Experimental results show that the proposed approach obtains an average of 39.03% higher lower bound for ISCAS'85 combinational benchmarks and 6.80% for ISCAS'89 sequential benchmarks as compared to random approach under the TSMC $0.18\mu m$ library.**

I. Introduction

With the relentless push for higher integration and higher performance, the chip density and clock frequency increase rapidly. This results in the extensive rises of power density and temperature in VLSI circuits. High temperature has a significant impact on the functionality and reliability of chips. Small [12] shows that the failure rate for components doubles every $10^{\circ}C$ increase. Thus, to cool down the temperature in a system such that the peak temperature is confined below a threshold, package design has to consider the worst case heating condition [11], and results in the high expense of package cost. On the other hand, the understanding of the peak temperature and the temperature distribution of VLSI circuits can serve as guidelines for design references.

Temperature profile has a close spatial correlation with power profile; therefore, extensive power density areas can lead to localized overheatings, called *hotspots*. Power consumption is mainly depending on input signals and circuit states once process parameters and architecture are fixed. Therefore, peak temperature associates with a specific starting state of circuit and a specific sequence of patterns. An exhaustive search from the space of all input pattern combinations for finding the pattern sequence that leads to peak temperature is impractical for large circuits, since there are $(2^n)^2 = 4^n$ possible pattern pair combinations for $n-$input circuits, where one pattern pair corresponds to a switch between two input patterns.

However, as indicated in [1], the location with the highest power density does not necessarily coincide with the location having peak temperature. It is because temperature distribution is a localized phenomenon governed not only by heat generation but by heat dissipation to ambient air. Thus, thermal simulation should be performed for more accurate results. Inspired by [4][7], our approach intergrates Ant Colony Optimization and Genetic Algorithm to produce the patterns with higher power consumption in the hotspot areas. Then thermal simulation is performed, and the results will be feedback for further pattern generation. Our approach can deal with large combinational and sequential circuit with the considerations of delay models and package cooling effect. The overall chip temperature profile is obtained at the end of algorithm for design reference. The experimental results show that our approach can find a tighter lower bound than that obtained by random patterns under the same amount of patterns.

This work was supported in part by the National Science Council of R.O.C. under Grants NSC 98-2220-E-007-015 and NSC 98-2220-E-007-023.

II. Preliminaries

A. Temperature Accumulation

Peak temperature occurs in localized overheating areas, hotspots. For the recognition of hotspots, we define a *hotspot frame* which is a square of high power density area. The frame size is determined heuristically. For combinational circuits, the power dissipation is governed by a pair of input patterns V_i and V_{i+1}. The average power consumption caused by switching this pattern pair is the same. Thus, we can re-apply a apir of patterns to reach steady-state peak temperature.

For sequential circuits, however, the state values of flip-flops within the circuit also affect the transition activities. Thus, the same input pattern under different states could lead to different next states and power consumption. On the other hand, if a state transition loop exists, same states can be recaptured, and the power consumption can be sustained over many cycles. We try to find the state transition loop and re-apply it to reach peak temperature in sequential circuits. Nevertheless, the state transition loop with higher power consumption in some sequential circuits may be difficult to be found or unreachable from the given initial state. Thus, we produce a sequence of patterns that consumes higher average power in the framed area instead.

B. Power model

The power consumption of CMOS circuits can be roughly divided into two groups: static power consumption, which is caused by static current drawn from power supply; and dynamic power consumption, which associates with signal switching activities. Once the process parameters and circuit architecture are determined, the power dissipation is dominated by dynamic power which corresponds to the input patterns and gate delay. Consider a digital circuit with m gates in the framed area, the dynamic power dissipation in the framed area over a cycle time T can be computed as Equation (1)

$$P = \sum_{i=1}^{m} \frac{V_{dd}^2}{2T} C_i D_i \quad (1)$$

where V_{dd} is the supply voltage, C_i is the load capacitance of gate i; D_i is the number of transitions of gate i from 0 to 1 or vice versa in the time interval T.

We aim to increase the power dissipation of the gates within the framed area. Instead of finding the C_i value, we use the *actual power dissipation value* AP_i which is calculated by HSPICE with the given cell library [8] to represent the effect of C_i. Then we define the *framed gate power* (FP_g) as the average power consumption of the gates in a frame during the time interval T. FP_g is an important parameter in our approach and is detailed in Equation (2). Note that different framed area may have different numbers of gates, therefore m is not a constant.

$$FP_g = \frac{\sum_{i=1}^{m} AP_i D_i}{m} \quad (2)$$

For example, in Fig. 1, assume that there are 4 gates in the framed area. The NOT gate A and AND gate C are with 2-unit delays, and the OR gates B and D are with 3-unit delays shown as a subscript on the top of the gate. Thus, the transition time of the gates can be calculated and are specified on the top right of the gate. We can find three transitions from 1 to 0 at gates A, B, and D, at t_2, t_5, and t_8, respectively. If assume the power consumption for NOT gate is 1.2 unit, and is 1.8 unit for the other types of gates. Then, the FP_g will be 1.2 as calculated in Fig. 1.

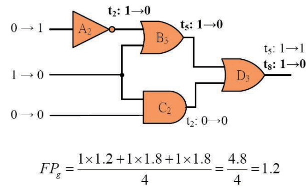


Fig. 1. An example of framed gate power (FP_g) calculation.

C. Thermal model

The temperature distribution on a chip is governed by the heat diffusion equation as Equation (3) shows [10]:

$$\rho c_p \frac{\partial T(r,t)}{\partial t} = \nabla \left[k(r,t) \nabla T(r,t) \right] + g(r,t) \quad (3)$$

where T is the temperature $^\circ C$, ρ is the density of the material (kg/m^3), c_p is the specific heat capacity ($J/kg^\circ C$), k is the thermal conductivity ($W/m^\circ C$), g is the power density (J/kg), and t is time (s). The heat conducting differential equation has a similar form to the electrical differential equation. A well known duality between them states that the heat flow passing through a thermal resistor acts as the electrical current passing through a resistor, and the temperature gradient corresponds to voltage gradient. Equation (3) can be solved for temperature distribution by finite difference methods [5] [13].

In this work, a chip is divided into *thermal blocks* when performing thermal simulation. The center of each block is regarded as the heat source in which the power generated within the block comes, and thermal resistances exist between each pair of adjacent blocks.

III. GA^2CO FOR COMBINATIONAL CIRCUITS

A. Hotspot evaluation

In our thermal model, a chip is divided into thermal blocks, and a *hotspot frame* is delimited and sequentially moved for hotspot recognition. To construct the initial hotspots, 10,000 random patterns are applied. After the simulation of a pattern pair, the hotspot frame is moved to search the area with the highest power density, and push this suspicious hotspot into a *hotspot queue* for succeeding simulations. To avoid a rapid increase of hotspot number, a *clarification* mechanism is performed by combining *adjacent* hotspots into the one which has the highest power density among them, where adjacency means two radiuses that centered by two hotspots are overlapped. Take Fig. 2 as an example, the radiuses of frame A and frame B are intersected, the frame B will be neglected if the frame A has a higher power density than that of frame B. This clarification mechanism can also avoid ambiguous input bit extraction in the next stage.

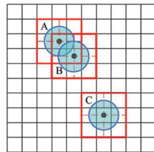


Fig. 2. Initial suspicious hotspot construction.

B. Critical input extraction

With the suspicious hotspots, the next stage of our approach is to increase the transitions in the framed areas where suspicious hotspots located. We apply the Ant Colony Optimization (ACO) algorithm [2] to find the input transitions that significantly cause the power dissipation in the framed area.

The ACO is an algorithm which imitates ants' behavior of seeking a "good" path between the nest and food. A chemical named pheromone is deposited by ants and forms a pheromone trail when ants pass by. This pheromone also evaporates with time such that the pheromone concentrations of trails are different with respect to the frequency of re-visiting. Ants tend to choose a path with a stronger pheromone concentration.

For each primary input i of a circuit, there are four transition conditions: 0→0, 0→1, 1→0, 1→1. Four parameters $\tau^i_{0\to0}$, $\tau^i_{0\to1}$, $\tau^i_{1\to0}$, $\tau^i_{1\to1}$ are used to represent the corresponding pheromones for these four transition conditions of primary input i. During the logic simulation, the FP_g value of an input pattern pair is added to these four parameters for pheromone accumulation. The amount of pheromone is updated heuristically as expressed in Equation (4)

$$\tau^i_{0\to0}(t+1) = \triangle\tau^i_{0\to0}(t) + (1-k)\,\tau^i_{0\to0}(t) \quad (4)$$

where $\tau^i_{0\to0}(t+1)$ is the updated pheromone for transition $0 \to 0$ of primary input i, $\Delta\tau^i_{0\to0}$ is the amount of pheromone newly deposited, i.e., FP_g value, and $k \in [0,1]$ is the pheromone evaporation rate. The other three parameters, $\tau^i_{0\to1}$, $\tau^i_{1\to0}$, and $\tau^i_{1\to1}$, are also expressed as Equation (4) except the subscripts of transition conditions.

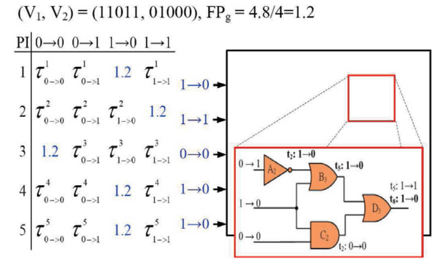


Fig. 3. An example for demonstrating pheromone trail accumulation.

Fig. 3 is an example for pheromone trail accumulation. Assume the circuit has five primary inputs and one of the framed area is the same as Fig. 1 shows. Four parameters $\tau_{0\to0}$, $\tau_{0\to1}$, $\tau_{1\to0}$, and $\tau_{1\to1}$ represent the pheromones of four transition conditions on the primary inputs. These τ values are all initialized to zero. After the input pattern pair (V_1, V_2) = (11011, 01000) is applied, the input signals are switched as $010 \to 100$ in the framed area. The FP_g value of this pattern pair (V_1, V_2) is derived as shown in Fig. 1. Then, this value 1.2 is added to each τ^i with respect to the transition condition. For instance, the transition condition of the primary input 1 between (V_1, V_2) pair is $1 \to 0$, then the parameter $\tau^1_{1\to0}$ is increased by the FP_g value 1.2.

The higher τ values are regarded as the higher probabilities to produce more transitions in the framed area. Each suspicious hotspot in the hotspot queue will keep its pheromone parameters for the generation of high power pattern pairs in the third stage.

C. Pattern generation

Genetic Algorithm [3] is a technique inspired by evolutionary biology. Parent genes generate offspring genes with higher fitness value through inheritance, mutation, and crossover. In this work, input pattern pairs with higher transitions are encoded into gene fragments. Then these pattern pairs perform inheritance, mutation, and crossover operations to generate new pattern pairs which may have higher power consumption in the framed area, and hence, heat up the hotspots.

After the simulations for hotspot i in the queue, three kinds of parent pattern pairs can be obtained. The first one is the pattern pair that obtained from pheromone accumulation of hotspot i in the critical input extraction stage. The pheromone parameter of each primary input with the highest value is selected. The second one is the pattern pair that causes the peak temperature among all simulations for the hotspot i. The last one, which is extracted from all hotspots, is a group of top 100 pattern pairs that cause the highest temperature among all hotspots.

Fig. 4 gives an example of our pattern generation. The row above the left top block contains the pheromone parameters that have the highest pheromone values for the hotspot i. The pheromone parameters

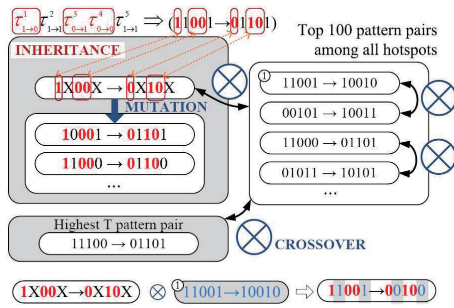


Fig. 4. The Genetic Algorithm-based pattern generation in our approach.

whose pheromone concentration are exceed 1.5 standard deviation, e.g., $\tau^1_{1\to 0}$, $\tau^3_{0\to 1}$, and $\tau^4_{0\to 0}$ are picked up and are inherited since these input transitions have significant impacts on the transition times in the framed area. Thus, we can obtain a gene fragment (**1**X**00**X→**0**X**10**X) accordingly, where X means don't care. The mutation operation replaces the don't care bits of this gene fragment with random values, e.g., (**1**0**00**1→**0**1**10**1). The crossover operation between the gene fragment and the top 100 pattern pairs is shown on the bottom of Fig. 4, where the non-don't care bits are replaced with top 100 pattern pairs in the corresponding position. Thus, an offspring pattern pair (**1**1**00**1→**0**0**10**0) is generated. The crossover operation also occurs between the highest temperature pattern pair and the top 100 pattern pairs, and occurs among the top 100 pattern pairs themselves. For these crossover operations, the offspring pattern pair is composed of parent genes which are randomly selected.

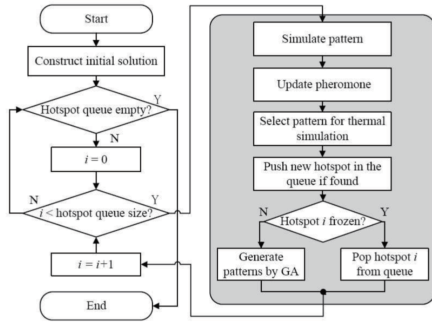


Fig. 5. The flow chart of our approach for combinational circuits.

D. Overall algorithm

The overall algorithm of our approach for combinational circuits is shown in Fig. 5. First, we construct the initial solution for suspicious hotspots by randomizing 10,000 patterns. Second, for each hotspot in the hotspot queue, ACO is performed to find the critical input transitions. And finally, the pheromone parameters of primary inputs and higher power pattern pairs are passed to the GA-based pattern generation stage to generate the patterns of the next generation. The population size generated from GA is 4,000 pattern pairs, and another 1,000 random pattern pairs are added for each generation to avoid being trapped in the local optimum.

For the sake of time saving, only the pattern pairs with extreme localized overheating or excessive overall power consumption will be passed to thermal simulation. After thermal simulation, newly discovered hotspots would be pushed into the queue. On the contrary, if a hotpot cannot reach a higher temperature for a period of time, then it is frozen. The frozen hotspots will be popped out from the queue. The algorithm is terminated when the hotspot queue is empty.

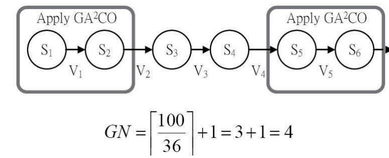
IV. GA^2CO FOR SEQUENTIAL PEAK TEMPERATURE ESTIMATION

As explained in Section II.A, when a high power pattern pair is found, it is re-applied to reach a steady state temperature for combinational circuits. For sequential circuits, however, we should also take the state issue into consideration. Therefore, we try to reach peak temperature by producing a sequence of patterns or a state transition loop that has high transition activities in the hotspot area. The same GA and ACO mechanisms as illustrated in the previous section are applied for producing these patterns.

However, if we apply the GA^2CO greedily to find the every next cycle pattern with high power consumption, it is easy to get trap in state transition loops and obtain a local optimum solution. Our heuristic is to determine the frequency GA^2CO would be applied by defining a *grasping number* (GN). The GA^2CO will be performed every GN cycles for finding the next pattern. At the other cycles, random approach is applied instead for producing more "different" patterns to escape from the loops. For consecutive n-cycles in a circuit, if the number of reached states is fewer, it is easier to get trapped in loops. Hence, the period of applying GA^2CO should be larger. That is, the GN is inversely proportional to the number of reached states within the initial cycles. The GN is defined as Equation (5)

$$GN = \lceil \frac{Initial\ cycles}{Number\ of\ reached\ states} \rceil + 1 \quad (5)$$

where initial cycles is the number of random pattern consecutively applied from the initial state. We set this value as 100. We add the number "1" at the end of this equation for performing GA^2CO at most every two cycles. Take Fig. 6 as an example, after applying 100 cycles of random patterns, the number of reached states is 36. Thus, the GN is 4 as calculated in Fig. 6. Therefore, the GA^2CO is applied to find the next high power pattern every four cycles.



$$GN = \left\lceil \frac{100}{36} \right\rceil + 1 = 3 + 1 = 4$$

Fig. 6. An example of determining the GN for sequential circuits.

The overall algorithm for sequential circuits is illustrated as Fig. 7. First, we set the cycle limit which represents the length of pattern sequence allowed for peak temperature estimation, and set the initial state as a legal state. The hotspot frame is initially placed on the area with the highest average power density through initial sequence, and this location will be adjusted according to the average power profile every 100 cycles. For every GN cycles, the ACO and GA are performed to find the critical input transitions and to generate the high transition patterns respectively. Otherwise, the next pattern is randomly generated. If a state transition loop is found, and its average power consumption is higher than that of the simulated sequence, we record this loop. When the pattern sequence length reaches the cycle limit, thermal simulation is performed for reporting the steady-state temperature of these sequences and loops.

V. EXPERIMENTAL RESULTS

We have implemented our approach in C++ for peak temperature estimation over a set of ISCAS'85 combinational benchmarks and ISCAS'89 sequential benchmarks. The layout and parasitic information are provided by [8], under the TSMC 0.18um library. The layout information and delay values are extracted from the DEF and SDF files. The power profile is obtained from HSPICE with the given cell libraries in [8] under 100MHz simulation frequency. The glitch behavior is modeled by the method [6]. Initial temperature is set to $25°C$, package information and thermal constants such as thermal conductivity are as that in Hotspot4.0 [5]. The experiments were conducted on Linux CentOS workstation with 32GByte memory. To show the effectiveness of the temperature elevation, the risen temperature ($\Delta T = T - 25°C$) is shown in the experimental results rather than actual temperatures.

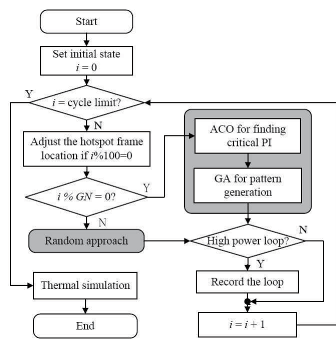


Fig. 7. The flow chart of our approach for sequential circuits.

TABLE I
PEAK TEMPERATURE ESTIMATION FOR ISCAS'85 COMBINATIONAL CIRCUITS

Circuit	PI	NG	ΔT ($°C$)			Time(m)			$\|Pattern\|$
			RAN	Ours	Impro.(%)	RAN	Ours	$\frac{Ours}{RAN}$(%)	
c432	36	160	6.40	8.87	38.63	3.08	0.38	12.34	119000
c499	41	202	3.60	3.62	0.49	4.41	0.14	3.09	69000
c880	60	383	8.93	13.97	56.44	20.32	1.56	7.69	159000
c1355	41	546	6.91	10.11	46.32	23.65	2.56	10.81	152000
c1908	33	880	11.45	15.76	37.65	20.50	2.73	13.33	129000
c2670	234	1269	8.86	14.50	63.67	40.99	4.77	11.64	255000
c3540	50	1669	13.40	17.02	27.06	24.40	3.44	14.10	149000
c5315	**178**	**2307**	**10.60**	**16.72**	**57.80**	**47.23**	**5.72**	**12.10**	**278000**
c6288	32	2416	62.81	75.27	19.83	27.39	6.02	21.98	119000
c7552	207	3513	13.12	18.70	42.45	63.20	12.03	19.03	360000
Average	-	-	-	-	39.03	-	-	12.61	-

Table I shows the experimental results of our approach and random approach for ISCAS'85 combinational benchmarks. For example, c5315 has 178 PIs and 2307 gates. ΔT is $10.60°C$ for the random approach and is $16.72°C$ for ours. The improvement of our approach against the random approach is 57.80%. The CPU time for random approach is 47.23 minutes and is 5.75 minutes for ours. The CPU time ratio of ours and random approach is 12.10%, and the number of patterns simulated for both approaches are 278000.

According to Table I, on average, our approach gives 39.03% tighter lower bound of peak temperature than that of random approach under the same amount of pattern pairs, and consumes only 12.61% CPU time. The reason that our approach consumes less CPU time against the random one, is because a selection condition is set for thermal simulation in our approach while not in the random one. This is because each random pattern is treated equally. Thus, thermal simulation was performed for every pattern pair among all patterns for finding out the pattern pair that leads to peak temperature.

Another comparison is taken by the risen temperature between the normal operating condition and the worst case condition found by GA^2CO. In the normal temperature estimation, the power profile was estimated by 10,000 random patterns. Note that in the worst case temperature estimation, we only find one pattern pair and reapply it to achieve a steady-state temperature. Table II shows the experimental results. Row 1 lists the benchmarks. Rows 2 and 3 list the risen temperature in the normal case and ours (worst case), respectively. Row 4 shows the ratio of risen temperature under these two conditions (worst case/normal). According to Table II, the average risen temperature in the worst case is larger than that of the normal case one with a ratio of 2.78.

Table III shows the experimental results of our approach and random approach for ISCAS'89 sequential benchmarks. Since [8] only provides the layout of some ISCAS'89 benchmarks, the experiments were conducted on these benchmarks which are list in Column 1. Columns 5 through 8 list the results of random approach. Column 5 lists the steady-state temperature difference after applying 10,000 pattern sequence ($S\Delta T$). Column 6 lists temperature difference when state transition loops exist ($L\Delta T$). Column 7 lists the number of states reached by applying 10,000 cycles. (Maximum = next states (10,000) + initial state

TABLE II
THE COMPARISON BETWEEN NORMAL AND WORST CASE RISEN TEMPERATURE.

Circuit	c432	c499	c880	c1355	c1908	c2670	c3540	c5315	c6288	c7552
Normal $\triangle T$	2.38	2.08	3.34	3.95	5.68	4.38	5.97	6.81	40.18	7.98
Worstcase $\triangle T$	8.87	3.62	13.97	10.11	15.76	14.50	17.02	16.72	75.27	18.70
$\frac{Worstcase\ \triangle T}{Normal\ \triangle T}$	3.73	1.74	4.18	2.56	2.77	3.31	2.85	2.46	1.87	2.34
										$Average\ ratio = 2.78$

TABLE III
PEAK TEMPERATURE ESTIMATION FOR ISCAS'89 SEQUENTIAL CIRCUITS.

Circuit	PI	FFs	NG	Random				Ours				
				$S\Delta T$	$L\Delta T$	States	Time	$S\Delta T$	$L\Delta T$	Impro.(%)	States	Time
s1488	**8**	**7**	**694**	**1.45**	**4.51**	**16**	**0.59**	**1.70**	**4.62**	**2.58**	**26**	**0.87**
s15850	77	534	11067	2.05	-	10001	0.44	2.29	-	11.47	10001	20.24
s35932	35	1728	19876	3.63	9.53	2040	3.33	3.88	9.77	2.53	2390	82.27
s38584	38	1426	22447	3.23	-	10001	2.58	3.32	3.32	2.55	10001	257.25
s38417	28	1636	25585	1.97	-	10001	1.42	2.26	-	14.88	10001	161.77
Average	-	-	-	-	-	-	-	-	-	6.80	-	-

(1) = 10,001). For example, s1488 has 8 PIs, 7 flip-flops, and 694 gates. For the random approach, the ΔT estimated by 10,000 patterns is $1.45°C$, and is $4.51°C$ from a state transition loop. The random patterns reach 16 states and the CPU time is 0.59 minutes. For our approach, the ΔT estimated by 10,000 patterns is $1.70°C$, and is $4.62°C$ from a state transition loop. The higher temperature between $S\Delta T$ and $L\Delta T$ is selected for comparison. Thus, the improvement is 2.58% in comparison of the $L\Delta T$. The number of reached state is 26 and the CPU time is 0.87 minutes. According to Table III, on average, our approach gives 6.80% tighter lower bound of peak temperature than that of random approach under the same cycle limit, 10,000.

VI. CONCLUSIONS

Getting tighter lower bound on estimating the worst case peak temperature requires the assistance of an efficient search algorithm in enormous search space. This paper proposes the GA^2CO algorithm for peak temperature estimation to avoid searching the whole solution space. The delay model, package cooling effect, and the reachability issue in sequential circuits are also considered. The experimental results show that the GA^2CO can find a tighter lower bound for larger combinational and sequential circuits than random approach does.

REFERENCES

[1] M. Cho et al, "PEAKASO: Peak-Temperature Aware Scan-Vector Optimization," in *Proc. VLSI Test Symp.*, pp. 52-57, 2006.
[2] M. Dorigo et al, "Ant Colony Optimization," Cambridge, MA:MIT Press, 2004.
[3] D. E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning," MA: Addison-Wesley, 1989.
[4] M. S. Hsiao et al, "Peak power estimation of VLSI circuits: new peak power measures," *IEEE. Trans. Very Large Scale Integration Systems*, pp. 435-439, Aug. 2000.
[5] W. Huang et al, "Compact Thermal Modeling for Temperature-Aware Design," in *Proc. Design Automation Conf.*, pp. 878-883, 2004.
[6] P. Israsena et al, "Novel pattern-based power estimation tool with accurate glitch modeling," in *Proc. Int. Symp. Circuits and Systems*, pp. 721-724, 2000.
[7] Y. L. Liu et al, "A Novel ACO-based Pattern Generation for Peak Power Estimation in VLSI Circuits," in *Proc. Int. Symp. on Quality Electronic Design*, pp. 317-323, 2009.
[8] X. Lu et al, "Layout and Parasitic Information for ISCAS Circuits," 2003. [Online]. Available: http://dropzone.tamu.edu/ xiang/iscas.html
[9] S. Manne et al, "Computing the Maximum Power Cycles of a Sequential Circuit," in *Proc. Design Automation Conf.*, pp. 23-28, 1995.
[10] M. Ozisik, "Boundary Value Problems of Heat Conduction," Oxford, U.K.: Oxford Univ. Press, 1968.
[11] M. Pedram et al, "Thermal modeling, analysis, and management in VLSI circuits: principles and methods," in *Proc. IEEE*, pp. 1487-1501, 2006.
[12] C. Small, "Shrinking Devices Put the Squeeze on System Packaging," in *EDN.*, pp. 41-46, Feb. 1994.
[13] T. Wang et al, "3D Thermal-ADI: A linear-time chip level transient thermal simulator," *IEEE Trans. on Computer-Aided Design*, pp. 1434-1445, Dec. 2002.