PERGAMON

Computers and Structures 80 (2002) 23–30

Computers & Structures

www.elsevier.com/locate/compstruc

Coefficients identification in electronic system cooling simulation through genetic algorithm

G.R. Liu *, J.-J. Zhou, J.G. Wang

Department of Mechanical Engineering, Centre for Advanced Computations in Engineering Science, National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260, Singapore

Received 9 March 2001; accepted 14 September 2001

Abstract

An inverse procedure is presented based on the improved genetic algorithm and is applied in electronic system cooling simulation to identify heat transfer coefficients through temperature distributions. This procedure includes forward and inverse procedures. Commercial software, *I-deas* (ESC and TMG), is adopted for forward computation and uniform-micro-genetic algorithm (μGA) is for inverse procedure. μGA is global optimization but its slow convergence affects its computational efficiency. A search domain narrowing operation is proposed to speed up the convergence so that the whole computational procedure can be carried out within an acceptable time. As an example, an electronic system cooling simulation, which includes fan cooling, water duct flow cooling and several printed circuit boards inside an electronic enclosure, is carried out and the efficiency increases approximate 33%. © 2002 Elsevier Science Ltd. All rights reserved.

Keywords: Inverse procedure; Genetic algorithm; Domain narrowing; Fluid flow; Heat transfer; Electronic system cooling

1. Introduction

Micro-electronic industry has made great process in recent years in increasing processing speeds and decreasing chip size. The by-product from this process is to accelerate a tendency to use the parts of more and more high heat density and the equipment or devices cannot be cooled down by natural air convection. Cooling technology such as forced air-cooling and sometimes water-cooling is necessary and important to guarantee a suitable working condition for these equipments.

In situ evaluation of a cooling system is important even to electronic system. Experimental approach has been widely used in practice to understand how these cooling technologies work. One disadvantage of such an approach is its costly and time consuming. It is also dangerous when an experimental test is carried out within extreme environments such as high temperature, high pressure and high velocity. Moreover, in situ test often lacks flexibility when a sensitivity study is required in design stage. For example, it is quite difficult to determine experimentally the most suitable location of fan and vent when an electronic system package is designed. More important is that some design parameters, for example the heat transfer coefficient between two contacted surfaces as shown in Fig. 1, are difficult to obtain directly through experiments. As a supplementary approach, numerical simulation for cooling system can reduce experimental work [1] and fully understand the physical process if all parameters and configuration are known. It can even be used to identify some design parameters through easily acquired data like temperature distribution. This parameter identification is usually done by means of inverse procedure [2–5,10].

Most inverse procedures can be classified into optimization processes. In recent years, genetic algorithms (GAs) have been widely applied in solving complex

* Corresponding author. Address: SMA Fellow, Singapore-MIT Alliance. Tel.: +65-874-6481; fax: +65-874-4795.

E-mail address: mpeliugr@nus.edu.sg (G.R. Liu).

0045-7949/02/$ - see front matter © 2002 Elsevier Science Ltd. All rights reserved.
PII: S0045-7949(01)00163-8

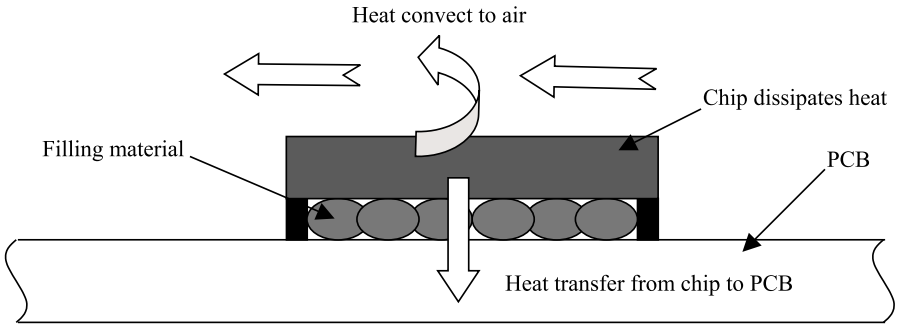

Fig. 1. Heat transfer at interfaces.

engineering problems [5,6] where other conventional optimization techniques fail or are not effective. The advantage of the GAs is their capability in searching for the global optimum instead of local optimum. This is because GAs are based on the principle of genetics and natural selection to construct search and optimization procedures. This search is stochastic and global. The search starts from a population of points rather than a single point, which enables the search to avoid being trapped at any local optimal point, thus realizing global optimization and no initial guess necessary. However, genetic algorithm is poor in convergence speed. A large number of forward analyses are usually required. When searching has zoomed into a localized region near the global optimum, their convergence speed is much slower in comparison with conventional local search techniques. Thus, the key to reduce the high computation cost of genetic algorithm is to speed up the convergence.

To improve the convergence performance, GAs are often recommended to combine with the local search techniques [7], where GAs perform global search to escape from falling into local optima whilst the local search carries out refine-tuning at the localized region. These combination approaches can be classified into two typical groups: (a) to run GAs until satisfactory results and then apply a hill-climbing heuristic to achieve the final solution, and (b) to integrate a hill-climbing method in each generation [8]. These combination approaches are usually high computational cost because the local hill-climbing approach still requires a large number of extra evaluations for the objective function. This evaluation takes considerable computation time.

In inverse procedure of coefficients identification for electronic system cooling problem, each forward simulation takes us tens of minutes which cover most of the computation time. The reduction of calling forward simulation in GAs is critical to save computation time. For this, a search domain narrowing operation is proposed to improve μGA performance. Being different from above combination approaches, this operation does not require any additional evaluation of objective function and forward calculation, thus saving computational cost. As an example, two parameters are identified through the operation. One parameter is the heat transfer coefficient between mainboard and CPU chip, which dissipates high-density heat of CPU chip. The other is a multiplier which forces convection heat transfer between bottom printed circuit board (PCB) and duct water flow. Temperature distribution of 20 points at chip and mainboard is pre-assumed. A commercial software code, ESC and TMG in *I-deas* Master Series package [9], is used to perform forward computation. Because the forward computation needs much longer time compared to what GAs required, improvement of the micro-genetic algorithm (μGA) is necessary to find the global optimal point with smaller number of generation. Our effort includes (a) interface programming inside the *I-deas* using its second developing language. This interface makes the GAs program call *I-deas* modules within GAs, (b) improvement of GAs to speed up the convergence through a search domain narrowing operation.

2. Identification coefficients in electronic system cooling simulation

In electronic system cooling analysis, computational fluid dynamics and heat transfer should be analyzed. There are two solvers named ESC and TMG for fluid flow and thermal transfer in the software *I-deas*, respectively. The flow solver considers nonlinear and coupled partial differential equations with conservation of mass, energy and momentum in general 3D geometry. It uses an element-based finite volume method and a coupled algebraic multigrid method to discretize governing equations. The physical models include laminar or turbulent incompressible flow, natural convection

and general boundary conditions for fluid flow and heat transfer in ducts and enclosures in the electronic cooling system.

Fluid flow is governed by the time-averaged Navier–Stokes equations for an incompressible Newtonian fluid. They are expressed as mass, momentum and energy equations, respectively:

$$\frac{\partial \rho_j}{\partial x_j} = 0 \tag{1}$$

$$\frac{\partial(\rho U_i)}{\partial t} + \frac{\partial(\rho U_j U_i)}{\partial x_j} = -\frac{\partial P}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\mu_{\text{eff}}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)\right] + S_{U_i} \tag{2}$$

$$\frac{\partial(\rho H)}{\partial t} + \frac{\partial(\rho U_j H)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\Gamma_{\text{eff}}\left(\frac{\partial H}{\partial x_j}\right)\right] + S_H \tag{3}$$

where the Einstein summation convention is applied ($i, j = 1$–3). ρ is the material density. U_i is the Cartesian components of mean velocity, P pressure and H static enthalpy. S_U and S_H are source terms for momentum and energy. μ_{eff} and Γ_{eff} are effective viscosity and effective coefficient of energy diffusion.

Thermal transfer is governed by the heat equations as follows:

$$\int_A Q_{\text{b}}\,\mathrm{d}A + \int_V Q_{\text{o}}\,\mathrm{d}V = \int_V \rho c\left(\frac{\partial T}{\partial t}\right)\mathrm{d}V \tag{4}$$

$$Q_{\text{b}} + Q_{\text{o}} = \left(\frac{\mathrm{d}U}{\mathrm{d}t}\right) \tag{5}$$

C is the specific heat, T the temperature and t time. Q_{b} is the heat flow across the control volume boundaries, Q_{o} the heat generated within the control volume and U the energy stored within the region.

Heat convection through fluid is described as:

$$q = G_{ij}(T_{\text{w}} - T_{\text{f}}) \tag{6}$$

where G_{ij} is the convective conductance from the element i to the fluid face j, T_{w} is wall temperature and T_{f} fluid temperature. The conductive conductance is calculated as:

$$G_{ij} = h_{\text{c}} A \tag{7}$$

where h_{c} is the convection heat transfer coefficient and A the overlapping area between convecting element and fluid face. The overlapping surface area is calculated during solution and h_{c} is calculated from the following equations:

$$\begin{aligned} Nu &= \frac{h_{\text{c}} l}{k} \\ Re &= \frac{\rho u_m l}{\mu} \\ Nu &= f(Re, Pr) \end{aligned} \tag{8}$$

where Nu is Nusselt number, l a characteristic length, k the thermal conductivity of fluid, Re Reynolds number and Pr the Prandlt number.

Thermal simulation uses a finite control volume (finite difference) method. Thermal code is tightly coupled with flow solver. Energy is transferred at the interface of fluid and solid between the two solvers. Control volume is established on convection faces. Convective rate is calculated from the thermal model to the faces of the flow model.

As shown in Fig. 1, the heat transfer through the contact area can be expressed as:

$$q = \frac{T_1 - T_2}{R_{\text{j}}} \tag{9}$$

$$R_{\text{j}} = \frac{1}{hA} \tag{10}$$

where A is the apparent contact surface, h the joint conductivity, R_{j} thermal joint resistance. T_1 and T_2 are the temperatures at the contacted surfaces, respectively. q is the transferred heat. The parameter h is difficult to be evaluated because the surfaces contact is complicated. In electronic system cooling, most heat dissipated by the chips will transfer to the PCB where the chips mounted. Of course, some heat is directly convected from chips surfaces to air. Although the temperatures of the chips and PCB can be measured, h is still unknown because the transferred heat is unknown. Therefore, h is inversely determined here.

3. Inverse procedure

3.1. Procedure for uniform-micro-genetic algorithm

Fig. 2 gives the flowchart for the identification of the two parameters in an electronic system cooling analysis. In the procedure, *I-deas* ESC module is used for forward calculation. A small model shown in Fig. 3 is used as an example to study the feasibility of the proposed procedure. There are 330 shell elements (to simulate structures like PCBs) and 30 water material beam elements (to simulate water flow) in the model. There are two small surfaces dissipating heat into the mainboard. The heat coefficients for the two small interfaces are expressed by h_1, h_2 which is to be identified. The heat is conducted to the two edges of the surface and taken away by the water inside water pipe.

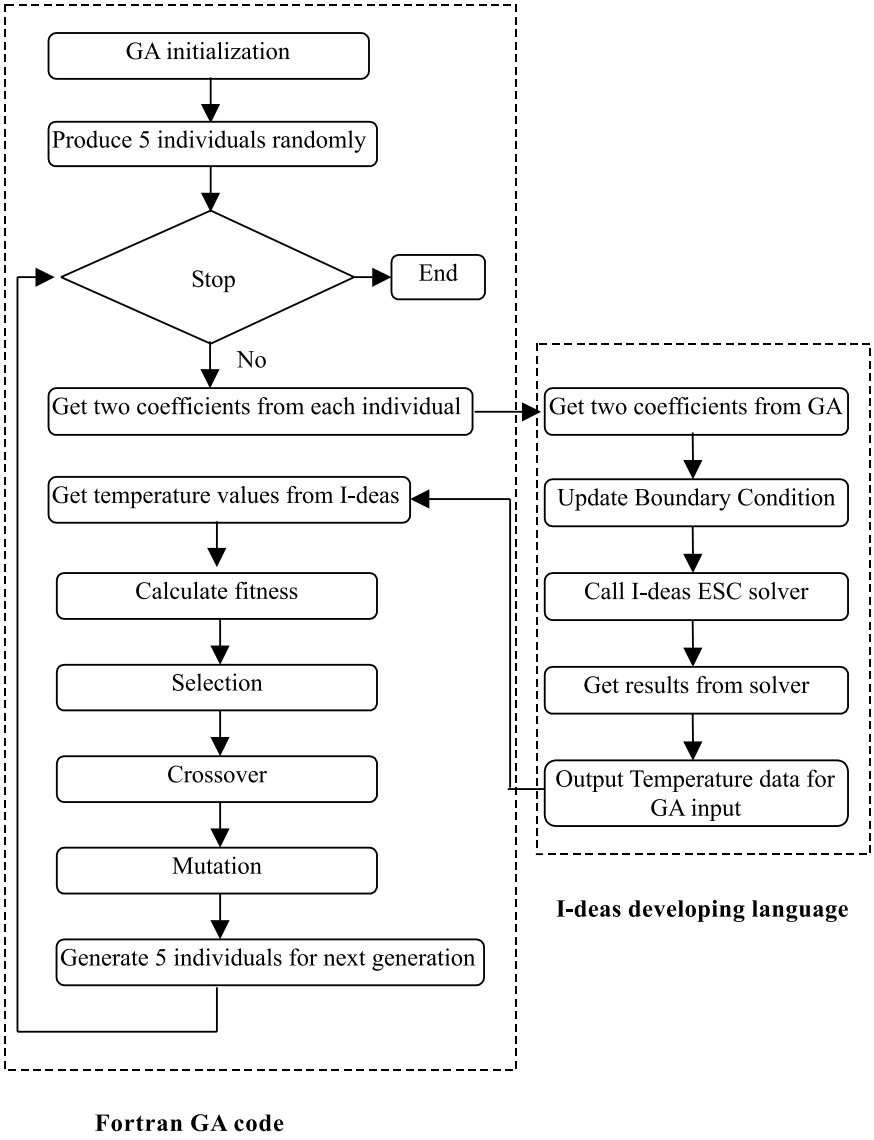

Fig. 2. Flowchart for the inverse procedure by GA.

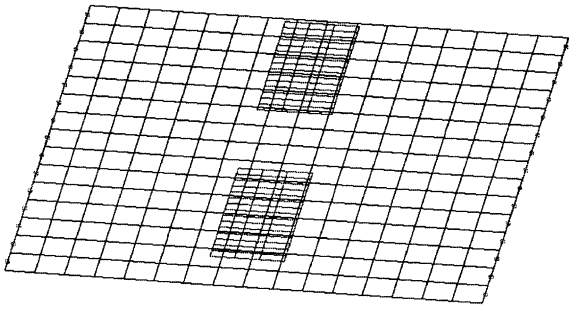

Fig. 3. Simple model for a two-interface problem.

In a GA run, each individual chromosome represents a candidate combination of these two coefficients, denoted by C. For each candidate combination, forward calculation is carried out to obtain a set of theoretical distribution of temperature T^{c}. The theoretical temperature data are compared with the test temperature to form a fitness value as follows:

$$\mathrm{ERR}(C) = \sum_{i=1}^{N} \left(T_i^{a} - T_i^{c}\right)^2 \tag{11}$$

where N is the number of points at which the temperatures have measured, T_i^{c} the temperature at point i from *I-deas* calculation, and T_i^{a} the known temperature at

point i. In real application of electronic products development, the known temperature T_i^a is from prototype temperature measuring. In this paper, T_i^a is obtained from a simulation instead of measurement.

Fitness value is used to determine the probability of the candidate being chosen as a future parent. The true combination obviously leads to minimum of error. GAs will generate the best combination through Eq. (11) of fitness value. The fitness value is more near to zero, the candidate coefficients combination more approaches to its true value.

The ranges of the two coefficients are supposed to be within 400–900. GA program will generate a serial of combinations for *I-deas*. If population size of each generation is limited to 5 and maximum generation is limited to 100, the best result is achieved at the 63rd generation. At this time, $h_1 = 660.51$ and $h_2 = 587.19$. The error function is $\mathrm{ERR}(C) = 1.76$.

Although GA convergence for this simple model is fast, the forward computation time is still huge. It spends totally about 18 h in SGI Indigo2 (IMPACT 10000) workstation. Detail analysis reveals that convergence speed is much slower when smaller ERR is achieved and the identified coefficients near to true values. Fig. 4 gives the convergence speed for iteration. An improvement is necessary for GAs to reduce the computation time.

3.2. Improvement of genetic algorithms

GA is strong in global optimum search and poor in local search. To improve the local search performance, GAs are often recommended to combine with local search techniques [8]. However, the local search techniques usually require running forward calculation, too. On this meaning, the local search techniques do not save forward calculations and thus the computational time does not reduce. A feasible scheme should be taken to achieve: (1) less forward calculation; (2) less generations in GAs procedure.

The first improvement is to inherit the best individual of current generation to the next generation. That is, μGA brings the best individual into next generation, including coefficients combination and fitness. If the candidate coefficients combination of next generation is the same coefficients, it is simply assigned the fitness value without forward calculation. For the population number of 5, it reduces one-fifth forward calculations, and saves nearly one-fifth calculation time.

The second improvement is to narrow search domain once after completion of specified generations. In μGAs running, the best ten individuals are kept, both coefficients and fitness, and re-ordered by their fitness values. Once after G generations, the maximum and minimum values, PMAX_j, PMIN_j, of each coefficient are found out from the I ($I \leqslant 10$) best individuals. Where j refers to the parameter number to be identified. A new search domain is formed as follows:

$$\mathrm{PMAX}_j^{\mathrm{new}} = \mathrm{PMAX}_j + f^*(\mathrm{PMAX}_j^{\mathrm{old}} - \mathrm{PMIN}_j^{\mathrm{old}}) \quad (12)$$

$$\mathrm{PMIN}_j^{\mathrm{new}} = \mathrm{PMIN}_j - f^*(\mathrm{PMAX}_j^{\mathrm{old}} - \mathrm{PMIN}_j^{\mathrm{old}}) \quad (13)$$

where f is an factor, $\mathrm{PMAX}_j^{\mathrm{old}}$, $\mathrm{PMIN}_j^{\mathrm{old}}$ are the maximum and minimum values of each coefficient in the previous search domain. This procedure is depicted in Fig. 5. According to the mechanism of GAs, certain number (G) of generations must be carried out to ensure the recorded I best individuals include the global optimization features of the problem. Our experience shows that $G = 21$ is sufficient. The parameter I is used to avoid trapping at any local optimal point when the objective function is not unimodal or not continuous. The parameter f is used to ensure the local best individual is not kicked out from the GAs search process.

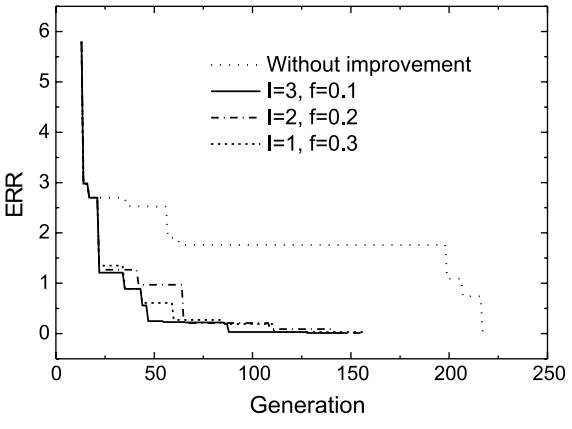

Fig. 4. Convergence study with and without improvement.

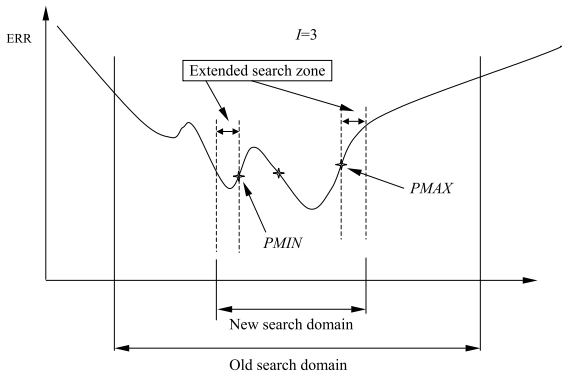

Fig. 5. Search domain reduction and best individual genetic scheme.

The combination of I and f makes certain that the GAs can find the best individual from the domain no matter how complicated the objective function is. Smaller parameters I and f lead to faster reducing of search domain but higher untouchability of the global optimum. However, for engineering problems like the examples of electronic system cooling in this paper, the objective function Eq. (11) is nearly unimodal and continuous. So the parameter I and f can be set to smaller values. Several combinations of the two parameters shown in Fig. 4. They all converge much faster than the normal μGas, and the combination $I = 3$ and $f = 0.1$ is best. The advantage over hill-climbing methods is no additional forward computation is required, thus saving computational cost.

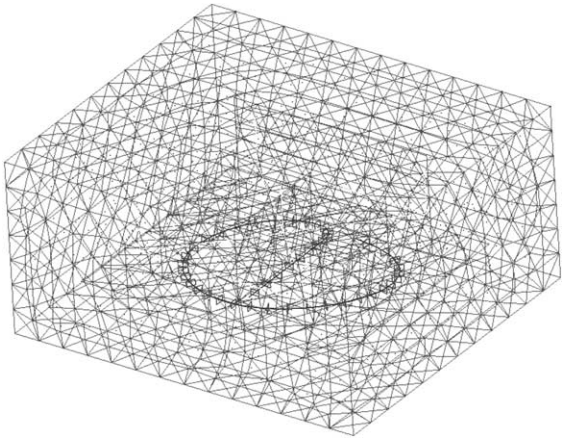

Fig. 7. Finite element model for the PCB problem.

4. Assessment through PCB example

4.1. Problem statement

A more complicated electronic system model is studied here to check the feasibility of the proposed method. This system includes one fan, one vent and five PCBs. A CPU is installed on the PCB 1 (Fig. 6). Besides fan cooling, water duct flow cooling is introduced to prevent the extreme temperature condition. The water duct is finned so that more convection heat transfers from bottom PCB to water. Two coefficients to be identified are the interface coefficient between the CPU and PCB 1 and a multiplier in the forced convection heat transfer between PCB and duct water flow. In *I-deas* TMG module, this multiplier is used to simulate the convection through fins. Temperatures at 20 points locating on CPU and PCB 1 are given before.

Fig. 7 shows the finite element mesh for forward computation. It includes 280 shell elements, 45 beam elements and 8006 air fluid elements. The fan is defined by air velocity and the fan cover was meshed by shell elements. All PCBs have different heat load, especially the PCB in the middle of the assembly has a concentrated heat load due to a CPU. Forced air convection is defined by two-dimensional thin shell elements. The small components attached in the PCBs are simulated by surface roughness. Roughness can be specified to the convection surfaces thereby creating drag on the surrounding fluid flow. Two kinds of heat convections are in the model: PCBs to airflow and PCBs to water duct flow. Beam element is used to express the forced water convection. The section of beam element and the velocity of water flow define pump characteristics. Boundary conditions are specified as follows: air velocity through fan is 8 m/s; heat loads of the five PCBs are 60, 10, 5, 6 and 5 W, respectively. The heat load of the CPU is 6 W. The roughness of the flow surfaces is defined as 1 mm. The water velocity through the pump is 0.2 m/s.

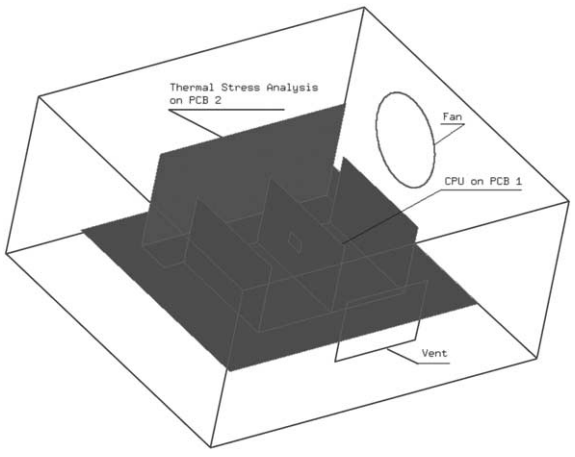

Fig. 6. Heat problem for PCB box.

4.2. Inverse procedure

Both normal μGAs and its improvement are applied in this problem. The maximum generation is limited to 150 for both methods and the ranges of the two coefficients are given as 0.8–2.5 and 1000–3000, respectively. It needs totally 76 h in SGI Indigo2 (IMPACT 10000) workstation for the normal μGAs. The best result is obtained at the 137th generation. The multiplier is 1.80 and the heat transfer coefficient is 2000.98. The error function reaches $\mathrm{ERR} = 0.0$. Fig. 8 compares the convergence speed before and after improvement. The convergence speed increases around 33%. This is the same conclusion as the simple model.

The temperature distribution is given for the identified coefficients. Fig. 9 gives the distribution of air velocity. The structural temperature is given in Fig. 10.

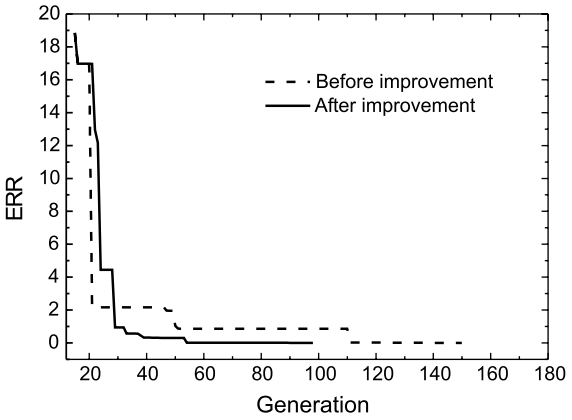

Fig. 8. Convergence study before and after improvement.

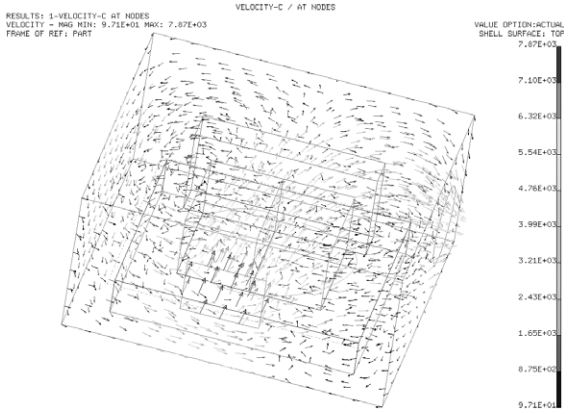

Fig. 9. Velocity distribution of airflow in the PCBs.

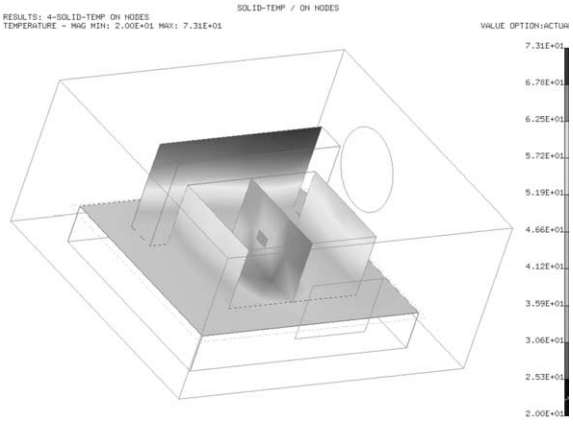

Fig. 10. Temperature distribution of PCBs.

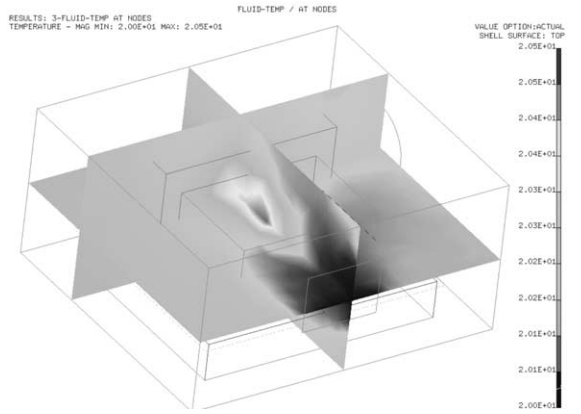

Fig. 11. Temperature distribution of air on cutting planes.

The air temperature at cutting planes is shown in Fig. 11. They are all reasonable under working condition of PCBs. For example, the maximum von-mises stress is only 13.5 MPa.

5. Conclusions

An inverse procedure is developed to determine two heat transfer coefficients in electronic system cooling analysis. This procedure uses a commercial software, *I-deas* (ESC and TMG), as forward solver to compute the temperature and stress distributions when the two coefficients are given. The inverse operation is carried out by an improved genetic algorithm which is based on a search domain narrowing operation. The search domain narrowing operation can reasonably and effectively reduce the search range of the coefficients and hence save the computational time of the whole procedure. Our examples show that the parameter $G = 21$ is adequate. The small I and f (in this paper $I = 3$ and $f = 0.1$) work well for engineering problem. Our calculation shows that this improvement can reduce the computational time 33%.

It is believed that this procedure can be used not only in electronic system cooling simulation, but also in other CAE applications available in *I-deas* software. For example, it can be used in the analyses of stress/strain, natural frequency and force excitation and so on. This approach can be employed in the engineering research and products development widely.

References

[1] Wills M. Thermal analysis of air-cooled PCBs. Electron Prod 1983;11–18.
[2] Balasubramaniam K, Rao NS. Inversion of composite material elastic constants from ultrasonic bulk wave phase velocity data using genetic algorithms. Composite Part B 1998;29B:171–80.

[3] Carroll DL. Genetic algorithms and optimizing chemical oxygen-iodine lasers. Dev Theor Appl Mech 1996;XVIII: 411–24.
[4] Carroll DL. Chemical laser modeling with genetic algorithms. AIAA J 1996;34(2):338–46.
[5] Han X, Liu GR, Lam KY, Ohyoshi T. A quadratic layer element for analyzing stress waves in FGMs and its application in material characterization. J Sound Vib 2000;236(2):307–21.
[6] Liu GR, Chen SC. Flaw detection in sandwich plates based on time-harmonic response using genetic algorithm. Comput Meth Appl Mech Eng 2001;190(42):5505–14.
[7] Gen M, Cheng RW. Genetic algorithms and engineering design. New York: Wiley; 1997.
[8] Xu YG, Liu GR, Wu ZP. An accelerated genetic algorithm using Hooke–Jeeves method for local searching. Proceedings of the 1st International Conference on Structural Stability and Dynamics, Taipei, 2000. p. 781–786.
[9] The I-deas Electronic System Cooling User's Guide and TMG Thermal Analysis User's Guide, MAYA Heat Transfer Technologies, 1995.
[10] Riche RL, Haftka RT. Optimization of laminate stacking sequence for bucking load maximization by genetic algorithm. AIAA J 1993;31(5):951–6.