Application of Estimation of Distribution Algorithm in HW/SW Partition

Juan Yu, Yuyao He
School of Marine Science and Technolgy,
Northwestern Polytechnical University,
Xi'an, Shaanxi, P.R.China
yujuan@snnu.edu.cn, heyyao@nwpu.edu.cn

Xiaoqiang Li
School of Automation and Informaton Engineering,
Xi'an University of Technology,
Xi'an, Shaanxi, P.R.China
mrglhome@xaut.edu.cn

Abstract—**Hardware/software (HW/SW) partitioning problem is NP hard problem. An improved algorithm based on estimation of distribution algorithms is proposed to solve HW/SW partitioning problem. Estimation of distribution algorithm is good in globe search but poor in local search and may suffer from"premature convergence" beacause of diversity loss.The improved algorithm strengthens the local searching ability by cloning and searching the elite solutions and improves the diversity loss by correcting the probability model. Numerical simulation is carried out and compared with existing algorithm, the results show the effectiveness of the improved estimation of distribution algorithm in solving HW/SW partitioning problem.**

Keywords—***hardware/software partitioning , estimation of distribution algorithm, elite clone, probability model correction***

I. INTRODUCTION

Typical embedded system includes hardware（FPGAs or ASICs）and programmable part, namely processor (DSPs or ASIPs) [1]. Many function blocks of the system can be implemented by either hardware or software. Generally software implementation is relatively flexible and cheap, hardware implementation could improve the speed of system, but the cost is relatively high. When design a hybrid system with hardware and software, we will face hardware/software (HW/SW) partitioning problem . The task of HW/SW partitioning problem is mapping the function blocks to target architecture to optimize the specified system overhead and meets the constraint conditions. The results of HW/SW partitioning problem have very important influence on the performance of the final product[2].

There are two types of HW/SW partitioning algorithm: exact algorithm and heuristic algorithm. Because HW/SW partitioning problem is NP hard, the exact algorithm can not solve large-scale problem,but heuristic algorithm do, such as artificial bees[2], genetic algorithm[3], particle swarm optimization algorithm[4], tabu search[5], constructed heuristic algorithm[6].These algorithms can solve the large-scale problem, but cannot ensure optimal solution and each has its own disadvantages. Tabu search is dependent on the initial solution and is serial search process; genetic algorithm is poor in local search and converges slowly; particle swarm optimization algorithm is liable to premature convergence; for bee algorithm ,the search speed is slow when approximating to the global optimal solution, and suffers from the population diversity reduction.

As a new algorithm, estimation of distribution algorithm (EDA) has good global search ability, can solve the high dimensional problems and difficult optimization problems, so it is widely used in recent years[7-12], however, no literature shows that EDA was used in HW/SW partitioning problem. This paper improved the shortcoming of estimation of distribution algorithms and applied it in HW/SW partitioning problem. The results were compared with original EDA and the HA algorithm in [6], experiment results show that the proposed algorithm is effective and can solve the HW/SW partitioning problem successfully.

The rest of this paper is organized as follows. Section II describes briefly target architecture and partitioning model. Section III analyzes basic estimation of distribution algorithm. Section IV describes the proposed algorithm named IEDA for HW/SW partitioning problem. Experimental results are presented in section V. Finally, we conclude this paper in section VI.

II. TARGET ARCHITECTURE AND PARTITIONING MODEL

The target architecture adopted is architecture of main processor with coprocessor. The main processor represents software subsystem(SW) which implements a function by software;The coprocessor represents hardware subsystem(HW) which implements a function by special hardware circuit.

Most of the existing HW/SW partitioning algorithms only consider the execution time and cost, but power consumption is also a crucial factor, high power consumption may lead to overheating and damage of circuit; On the other hand, the fact that effective working time of mobile computing devices relied on battery power has great influence on the consumer's purchase decision. So it should limit power consumption in a certain range to ensure the effective working time. This paper will minimize hardware cost under the constraints of power consumption and execution time, the same as [6].

For comparison, the same partitioning model of [6] is used, as shown in Fig.1.

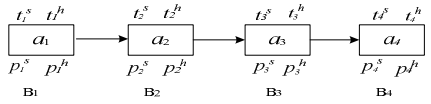


Fig.1. Partitioning model

For the function block set $B=\{B_1,B_2\ldots B_n\}$, HW/SW partitioning problem will map all the function blocks to HW and SW sets, and $HW \subset B$, $SW \subset B$, $HW \cup SW = B$, $HW \cap SW = \Phi$.

According to the requirements of the partitioning algorithm, parameterize B_i as follow: $B_i=< a_i, t_i^s, t_i^h , p_i^s , p_i^h >$. Where a_i is the hardware cost when B_i implemented with hardware and it is proportional to the circuit area occupied; t_i^s and p_i^s are execution time and power consumption respectively when B_i is implemented with software; t_i^h and p_i^h are execution time and power consumption respectively when B_i is implemented with hardware. In general,for the same B_i, the execution time and power consumption implemented with hardware is less than that implemented with software[6],so $t_i^s>t_i^h$, $p_i^s>p_i^h$.

We can establish the mathematical model of HW/SW partitioning problem as Eq.(1):

$$\begin{cases} \min \cos t(X) = \sum_{i=1}^{n} a_i x_i \\ s.t.\sum_{i=1}^{n} \left[t_i^s (1-x_i) + t_i^h x_i\right] \le T \\ \sum_{i=1}^{n} \left[p_i^s (1-x_i) + p_i^h x_i\right] \le P \\ x_i \in \{0,1\} \end{cases} \quad (1)$$

Among them: n is the number of functional blocks; x_i=1 means B_i implemented with hardware, x_i=0 means B_i implemented with software; T is the time constraint, P is the power consumption constraint.

Let $T'=\sum_{i=1}^{n} t_i^s - T$, $P'=\sum_{i=1}^{n} p_i^s - P$, $t_i = t_i^s - t_i^h$, $p_i = p_i^s - p_i^h$, we obtained Eq.(2) which is eqcuivalent to Eq.(1):

$$\begin{cases} \min \cos t(X) = \sum_{i=1}^{n} a_i x_i \\ s.t.\sum_{i=1}^{n} t_i x_i \ge T' \\ \sum_{i=1}^{n} p_i x_i \ge P' \\ x_i \in \{0,1\} \end{cases} \quad (2)$$

The improved estimation of distribution algorithm is applied to solving the HW/SW partitioning problem represented by Eq.(2) in this paper.

III. ESTIMATION OF DISTRIBUTION ALGORITHM

Estimation of distribution algorithm is a new branch of evolutionary computation, which is the combination of statistical learning and evolutionary algorithms. It is proposed to solve the building block damage problem of genetic algorithm. EDA builds a probability model from promising solutions set, then attempts to estimate the probability distribution of the promising solutions. Once the model is built, new solutions are generated by sampling the distribution encoded by this model. The operation is repeated and the evolution of population is realized. EDA procedure is outlined as follows:

Step 1 g=0, generate initial population pop(g) .
Step 2 Select S(g) from pop(g).
Step 3 Build probability model ρ (g) from S(g) .
Step 4 Randomly sample from ρ (g) .
Step 5 Generate pop(g+1).
Step 6 g=g+1.
Step 7 Judges whether the termination criterion is met, if met then output the optimization results; otherwise, turn to Step2.

Sampling probability model avoids damaging the promising solutions, which is contrary to the operation of crossover and mutation in genetic algorithm. It is also the main characteristics different from genetic algorithm. EDA controls the evolutionary direction of population from macroscopic, solves high dimensional problem and difficult optimization problem effectively and reduces the time complexity.

But EDA also has some shortcomings: ①.the local search ability is poor relatively; ②. when studying the probability model in the process of evolution, it tends to overfit the distribution of solution space , no longer generate diversity solutions after several generations and results in premature convergence.

IV. IMPROVED EDA SOLVING HW/SW PARTITIONING PROBLEM

There is an underlying assumption in most heuristic algorithms: good solutions have similar structure.This assumption is reasonable for most real-world problems,e.g., the percentage of common edges in any two locally optimal solutions of a traveling salesman problem obtained by the Lin–Kernighan method is about 85% on average[8].Based on this theory, the proposed method strengthens the local search ability of original algorithm by cloning and searching good solutions, and alleviates diversity loss by correcting probability model. The improved algorithm based on EDA is named as IEDA, which is applied to solving HW/SW partitioning problem.

A. Representation of Solution

The solution of HW/SW partitioning problem is represented as a binary string with length of n, i.e., $\boldsymbol{X}=(x_1\ldots x_i\ldots x_n)$.

B. Elite Clone Selection Operation

The elite clone selection operation is applied before Step 2 of EDA algorithm.

1）Size of Clone

Elite population $s^1(g)$ which consists of Ms the best fitnesses individuals in pop(g) is selected and cloned, Ms=[α *M],M is population size, 0<α <1. nc_i is the clone size of the ith elite solution $\boldsymbol{X}_i$, when nc_i is determined, two factors should be considered:objective function value cost($\boldsymbol{X}_i$) and the similarity of $\boldsymbol{X}_i$ with the rest solutions of the population. In the problem represented by Eq.(2),the smaller cost($\boldsymbol{X}_i$) and the smaller the similarity of $\boldsymbol{X}_i$, $\boldsymbol{X}_i$ is more promising and should strengthen the search for it. nc_i is calculated by Eq.(3) :

$$nc_i = \mathrm{int}\left(Nc * \frac{1/\cos t(X_i)}{\sum_{j=1}^{M_s} 1/\cos t(X_j)} * \frac{q_i}{\sum_{j=1}^{M_s} q_j} \right) \quad (3)$$

Where int(x) is the minimum integer greater than x. Nc is a constant for clone, it is an integer greater than M. $q_i = \min\{d_{ij}\}$, $i = 1,2,3...,\mathrm{Ms}$, $j = i+1...,\mathrm{M}$, d_{ij} is the hamming distance between $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$ .The greater the q_i , the smaller the similarity of $\boldsymbol{X}_i$.

2) T transformation

Supposing X_i^j is the jth cloned individual of $\boldsymbol{X}_i$, transformation from X_i^j to X'^j_i is defined as T($X_i^j \rightarrow X'^j_i$): randomly select k genes which value equal to 0 and turn them to 1; randomly select k+t genes which value equal to 1 and turn them to 0.The meaning of T transformation in HW/SW partitioning problem lies in switching the mapping domain of k function blocks in software domain and k+t function blocks in hardware domain.

3) Dominance Replacement

Let cost(X_i^k)=min(cost(X'^j_i)), $\boldsymbol{X}_i \in$ $s^1(g)$, j=1,2,…,.nc_i,if cost(X_i^k) $\leq$ cost($\boldsymbol{X}_i$), $\boldsymbol{X}_i$ is replaced by X_i^k . For $\boldsymbol{X}_i \in$ $s^1(g)$, the process of elite clone selection operation offers nc_i different search directions in the neighborhood of $\boldsymbol{X}_i$ and makes local search in the neighborhood, the dominance replacement ensures that the solutions will not be worse , and accelerates the convergence speed. After dominance replacement , a new population pop '(g) is formed.

C. Probability Model

Probability vector $\rho(g) = (\rho_1, \ldots \rho_k \ldots, \rho_n)$ is probability model to describe the distribution of solution space. ρ_k is the probability of B_k mapped to the hardware domain i.e., the probability of $x_k = 1$ .Offspring solutions are generated by sampling from probability model, i.e., a number r can be generated randomly , r$\in$ [0 ,1] ,if r< ρ_i , x_i=1; otherwise x_i=0.

This paper adopts the UMDA [7] probability model , it builds the probability model on $s^2(g)$ subset which is composed of [β *M] individuals with best fitness in pop'(g) , 0< β <1. ρ_i is updated by (4) , x_i^j is the value of the ith gene of the jth individual:

$$\rho_i = \frac{\sum_{j \in s^2(g)} x_i^j}{[\beta * \mathrm{M}]} \quad (4)$$

The probability of generating solution X is calculated by Eq.(5) :

$$\rho(X) = \prod_{i=1}^{n} [\rho_i * x_i + (1-\rho_i) * (1-x_i)] \quad (5)$$

Initial population pop (0) is generated randomly with uniform distribution of the solution space , namely $\rho(0) = (0.5...0.5...0.5)$.

D. Probability Model Correction

Along with the iteration, EDA will lose diversity, population variance is growing smaller, eventually reduces to zero, probability model evolves to only produce same solutions[12].

EDA diversity loss occurred in two stages :

- Sampling from the probability model to generate population . This diversity loss is due to the fact that variance of population sampled from probability model will be less than the variance of parent population. Elite clone selection operation introduced by part B not only strengthens the local search ability but also improves the population diversity loss problem since X'^k_i is obtained by T transformating the sampled solution.
- Selecting M solutions to build the probability model of next generation .Usually high fitnesses solutions are selected to build probability model, thus distribution characteristics of good solutions are achieved, but will cause loss of diversity. It can be relieved by slowing down the update speed of probability model, making update speed relatively slower than the search speed of algorithm. Once ρ_i is close enough to 0 or 1, the value of x_i tends to fix and lead to the combination reduction of n genes,i.e., the diversity of population is made to decrease. So the boundary of probability model is corrected by Eq.(6):

$$\rho_i' = \begin{cases} \gamma, \rho_i < \gamma \\ p_i, \lambda < \rho_i < 1-\gamma \\ 1-\gamma, \rho_i > 1-\gamma \end{cases} \quad (6)$$

ρ_i is probability before correction, ρ_i' is probability after correction, $\gamma = 1/n$.

The correction prevents ρ_i reaching 1 and 0 , lowers the update speed of probability model, also improves the diversity loss of population and avoids falling into local optimum .

E. Repair Infeasible Solution

Sampled solutions may not satisfy the time or power consumption constraints, so it is necessary to repair the infeasible solutions. The repair process is as follows:

Step1 Compute *tuse,puse*: *tuse*= $\sum_{i=1}^{n} t_i x_i$ *puse*= $\sum_{i=1}^{n} p_i x_i$.

Step2 Compute b_i :

if *tuse*< *T* '&& *puse*>= *P* ', solution not meeting the time constraint , $b_i = a_i / t_i$;

if *puse*< *P* ' && *tuse*>= *T* ', solution not meeting the power consumption constraint, $b_i = a_i / p_i$;

if *puse*<*P* ' && *tuse*<*T* ', solution meets neither time constraint nor power consumption constraint, $b_i = a_i / \sqrt{t_i^2 + p_i^2}$;

Step3 Sort b_i in ascending order, let the sequence after sort is $b_1 < b_2 < \ldots < b_n$.

Step 4 for i=1:n

if x_i=0， let x_i=1, *tuse*=*tuse*+t_i, *puse*=*puse*+p_i ;

if *tuse*>=*T* ' && *puse*>= *P* ',break.

F. IEDA Algorithm

The algorithm flow of IEDA is as follows:

Input: a_i, t_i^s, p_i^s, t_i^h, p_i^h, *T*, *P*

Output: $\boldsymbol{X}_{bestsofar}$ =(x_1,x_2,…,x_n)

Step1 Parameter settings: set value of G, M , Nc, α , β , k and t.

Step2 Let g=0,generate initial population pop(0)={$\boldsymbol{X}_1$,…$\boldsymbol{X}_M$} and repair infeasible solutions, then calculate the objective function value *cost*($\boldsymbol{X}_i$) and sort in ascending order,let $cost_{bestsofar}$=*cost*($\boldsymbol{X}_1$), $\boldsymbol{X}_{bestsofar}$=$\boldsymbol{X}_1$.

Step3 Perform elite clone selection operation for pop(g), and form a new population pop '(g).

Step4 Select s^2 (g) from pop '(g) and build probability model from s^2 (g).

Step5 Correct probability model.

Step6 Generate new population pop (g + 1) by sampling , and repairing infeasible solutions .

Step7 Calculate *cost*($\boldsymbol{X}_i$) and sort in ascending, $X_i \in pop(g+1)$.

Step8 Update $cost_{bestsofar}$ and $\boldsymbol{X}_{bestsofar}$.

if *cost*($\boldsymbol{X}_1$)<$cost_{bestsofar}$ then $cost_{bestsofar}$=*cost*($\boldsymbol{X}_1$) , $\boldsymbol{X}_{bestsofar}$= $\boldsymbol{X}_1$.

Step9 g=g+1.

Step10 If the termination criterion is met , then $\boldsymbol{X}_{bestsofar}$ is output;otherwise , turn to Step3.

V. SIMULATION

Reference [6] solves the HW/SW partitioning problem by a heuristic algorithm(HA) based on greedy algorithm . In this paper, basic EDA, IEDA and HA algorithms under 36 different constraint conditions are simulated by C language and the simulation results are compared.

A． Parameters Settings

1) Function Block Parameters

In order to compare with HA algorithm , parameter setting is the same as [6]. t_i^s is generated randomly in (0, 30), p_i^s is generated randomly in (0, 20); t_i^h and p_i^h are generated randomly in (0, $\lambda * t_i^s$) and (0, $\lambda * p_i^s$) respectively.

2) IEDA Parameters

According to the experience, let M =40 , Nc=25*M， $\beta = 1/4$; α decides the number of elite solutions needed to clone, larger α makes more comprehensive search, but also brings longer computing time. For comprehensive consideration , let $\alpha = 1/8$.

Under five kinds of typical constraint conditions (i.e.,(θ_T , θ_P) takes (1/6,1/6), (1/6,6/6), (3/6,3/6), (6/6,1/6), (6/6,6/6) respectively), a total of six kinds of combination of k and t are simulated , in which k takes 1,2 and 3, t takes 1, 2 respectively, and eventually k = 1, t = 1 are determined.

Termination criterion is achieving maximum number of iterations G, G = 2000.

B． Simulation Results

For further describe the simulation results，we employ the following notations.

- Suppose $\cos t(A)$ is hardware cost obtained by algorithm A, and $\cos t\_hw$ is hardware cost when all the blocks are mapped to hardware domain. Let

$$\cos t\_used(A) = \frac{\cos t(A)}{\cos t\_hw} \times 100\% \qquad (7)$$

Because the optimization objective is minimizing cost, the smaller $\cos t\_used(A)$, the better the optimization results of the algorithm. Let

$$\delta_{\text{A-B}} = \cos t\_used(A) - \cos t\_used(B) \qquad (8)$$

If $\delta_{A-B} > 0$,the results of algorithm B is better than that of algorithm A , and the greater the value,the better the algorithm B relative to algorithm A . If $\delta_{A-B} < 0$, the results of algorithm A is better than that of algorithm B .

- The HW/SW partitioning problem is solved under different constraints, so the time constraint and power consumption constraint are set by Eq.(9):

$$\begin{cases} T(\theta_T) = T\_hw + \theta_T * (T\_sw - T\_hw) \\ P(\theta_P) = P\_hw + \theta_P * (P\_sw - P\_hw) \end{cases} \quad (9)$$

T_hw and P_hw are time and power consumption when all the blocks are mapped to hardware domain; T_sw and P_sw are time and power consumption when all the blocks mapped to software domain; $0 \le \theta_T \le 1$, $0 \le \theta_P \le 1$, different θ_T, θ_P values represent different time constraint and power consumption constraint. As can be seen, the value of time constraint is limited within the range of $[T\_hw, T\_sw]$, the value of power consumption constraint is limited within the range of $[P\_hw, P\_sw]$.

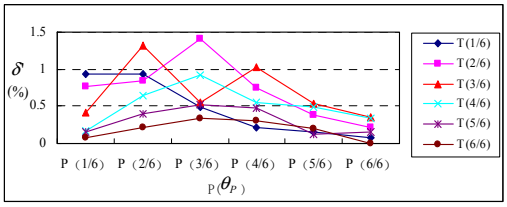
The same as [6], n=100, $\lambda = 1/2$, θ_T and θ_P take 1/6, 2/6, 3/6, 4/6, 5/6, 6/6 respectively, the algorithm is simulated under 36 kinds of constraint conditions, the simulation results are shown in TABLE Ⅰ.The results are average value of 20 random instances . For comparison , δ' is also listed, δ' is the error between the solution of HA algorithm and the optimal solution.The smaller the value of δ', results of HA are closer to the optimal solution, the data of δ' come from [6].

As can be seen from the data shown in TABLE Ⅰ:

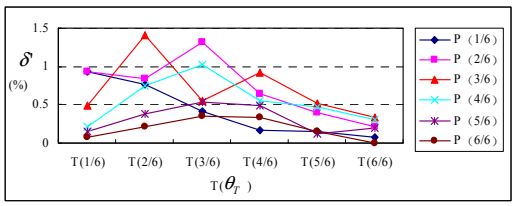
- Compare $\delta_{\text{HA-EDA}}$ and $\delta_{\text{HA-IEDA}}$, all the remaining 35 kinds of constraint conditions are $\delta_{\text{HA-IEDA}} > \delta_{\text{HA-EDA}}$,except under the constraint condition of (6/6, 6/6) that EDA and IEDA both obtained the optimal value of 0, lead to $\delta_{\text{HA-EDA}}$ equal to $\delta_{\text{HA-IEDA}}$. The results indicate that IEDA can obtain better optimization results compared to EDA for HW/SW partitioning problem, so the improvement of EDA is effective.
- The relation between δ' and P(θ_P) is shown in Fig.2 (a), the relation between δ' and T(θ_T) is shown in Fig.2 (b). For the same time constraint T(θ_T), δ' is larger under P(2/6), P(3/6), P(4/6) than it is under P (5/6) and P (6/6), There is the same situation for the same power consumption $P(\theta_p)$. It shows that when ①.at least one constraint condition is loose or ②. difference between θ_T and θ_P (degree of relaxation of two constraint conditions) is larger, δ' is smaller, and HA could get better results, that is to say, results of HA solving the HW/SW partitioning problem are dependent on the constraints.

TABLE Ⅰ .SIMULATION RESULTS OF HA,EDA AND IEDA

		δ (%)					
		P (1/6)	P (2/6)	P (3/6)	P (4/6)	P (5/6)	P (6/6)
T(1/6)	$\delta_{\text{HA-EDA}}$	0.1	0.48	0.34	0.08	-0.13	-0.04
	$\delta_{\text{HA-IEDA}}$	0.97	1.09	0.66	0.26	0.17	0.10
	δ'	0.93	0.93	0.49	0.22	0.15	0.08
T(2/6)	$\delta_{\text{HA-EDA}}$	-0.76	-0.43	0.73	0.40	0.04	-0.24
	$\delta_{\text{HA-IEDA}}$	0.79	0.77	1.41	0.79	0.40	0.31
	δ'	0.77	0.84	1.41	0.75	0.39	0.22
T(3/6)	$\delta_{\text{HA-EDA}}$	0.35	0.22	-0.22	0.22	0.41	0.01
	$\delta_{\text{HA-IEDA}}$	0.49	1.4	0.78	1.23	0.68	0.37
	δ'	0.42	1.31	0.55	1.03	0.54	0.35
T(4/6)	$\delta_{\text{HA-EDA}}$	0.20	0.25	0.71	0.04	0.38	0.43
	$\delta_{\text{HA-IEDA}}$	0.27	0.77	0.94	0.4	0.57	0.44
	δ'	0.17	0.64	0.92	0.55	0.49	0.34
T(5/6)	$\delta_{\text{HA-EDA}}$	0.11	0.02	0.30	0.25	-0.01	0.10
	$\delta_{\text{HA-IEDA}}$	0.17	0.42	0.60	0.69	0.13	0.15
	δ'	0.16	0.40	0.52	0.47	0.12	0.15
T(6/6)	$\delta_{\text{HA-EDA}}$	-0.06	-0.23	-0.15	0.10	0.05	0.01
	$\delta_{\text{HA-IEDA}}$	0.13	0.25	0.49	0.37	0.18	0.01
	δ'	0.08	0.22	0.33	0.31	0.2	0



(a) Relation between δ' and P(θ_P)



(b) Relation between δ' and T(θ_T)

Fig. 2. Relation between δ' and P(θ_P)、T(θ_T)

- Can be seen from TABLE I that $\delta_{\text{HA-IEDA}} > 0$, and the change trend of $\delta_{\text{HA-IEDA}}$ with P(θ_P) under different T (θ_T) is almost the same as change trend of δ'. $\delta_{\text{HA-IEDA}}>0$ shows that the results obtained from IEDA are better than results from HA; $\delta_{\text{HA-IEDA}}$ and δ' having same change trend shows that the error between the results of IEDA and the optimal solutions is substantially constant. That is to say the constraint does not affect the performance of the IEDA algorithm . IEDA algorithms could obtain good results both under the loose constraints and strict constraints . Because the results are average value of 20 instances generated randomly, there is a small error between $\delta_{\text{HA-IEDA}}$ and δ' .The small error implies that the results obtained from IEDA are very close to or equal to the optimum value. So the IEDA algorithm can more effectively solve the HW/SW partitioning problem than HA .

VI. CONCLUSION

Estimation of distribution algorithm is analyzed, the basic distribution estimation algorithm is improved and applied to solving HW/SW partitioning problem in this paper . The elite clone selection operation is introduced to strengthen the local search ability and the probability model is corrected to improve the diversity loss problem. The improved algorithm is used to solve the HW/SW partitioning problem which with time and power consumption as constraint conditions , minimizing the cost as optimal objective. Compared with the basic EDA algorithm and the HA algorithm, the results show that the improved EDA algorithm (IEDA) can obtain better results, and effectively solve the HW/SW partitioning problem.

ACKNOWLEDGMENT

This paper is supported by the National Natural Science Foundation of China (Grant No. 61271143 and No.60871080).

REFERENCES

[1] M.B.Abdelhalim,S.E.-D.Habib. An integrated high-levelhardware/software partitioning methodology. Design Automation for Embedded Systems, vol.15,2011,pp.19-50.

[2] Mouloud Koudil, Karima Benatchba, Amina Tarabet, El Batoul Sahraoui.Using artificial bees to solve partitioning and scheduling problems in codesign.Applied Mathematics and Computation , vol.186 , 2007,pp.1710–1722.

[3] Madhura Purnaprajna, Marek Reformat,Witold Pedrycz .Genetic algorithms for hardware–software partitioning and optimal resource allocation. Journal of Systems Architecture, vol.53, 2007, pp. 339–354.

[4] Yue Wu, Hao Zhang, Hongbin Yang .Research on Parallel HW/SW Partitioning Based on Hybrid PSO Algorithm. Algorithms and Architectures for Parallel Processing, vol.5574,2009, pp.449-459 .

[5] JigangWu , PuWang, Siew-Kei Lam . Thambipillai Srikanthan . Efficient heuristic and tabu search for hardware/software partitioning.J Supercomput DOI: 10.1007/s11227-013-0888-9 ,2013.

[6] Wu Jigang, Thambipillai Srikanthan .Algorithmic aspects of area-efficient hardware/software partitioning. The Journal of Supercomputing ,vol.38,2006,pp. 223-235.

[7] Shih-Hsin Chen, Min-Chih Chen, Pei-Chann Chang, Qingfu Zhang, Yuh-Min Chen.Guidelines for developing effective Estimation of distribution algorithms in solving single machine scheduling problems.Expert Systems with Applications , vol.37,2010,pp.6441-6451.

[8] Christopher Expósito Izquierdo, José Luis González Velarde, Belén Melián-Batista , Moreno-Vega.Hybrid Estimation of distribution algorithm for the quay crane scheduling problem.Applied SoftComputing ,vol.13,2013,pp.4063–4076.

[9] Ling Wang, Sheng-yao Wang, Ye Xu. An effective hybrid EDA-based algorithm for solving multidimensional knapsack problem. Expert Systems with Applications ,vol.39,2012, pp. 5593–5599.

[10] Josu Ceberio, Ekhine Irurozki, Alexander Mendiburu, Jose A. Lozano. A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems. Progress in Artificial Intelligence , vol.1, Issue 1, 2012,pp.103–117.

[11] Roberto Santana, Pedro Larrañaga, José A. Lozano.Combining variable neighborhood search and Estimation of Distribution Algorithms in the protein side chain Placement problem. Journal of Heuristics, vol. 14,2008,pp.519–547.

[12] Jürgen Branke , Clemens Lode , Jonathan L. Shapiro.Addressing sampling errors and diversity loss in UMDA. Proceedings of the 9th annual conference on Genetic and evolutionary computation, London, England, UnitedKingdom,2007,pp.508-515.