Journal of Theoretical and Applied Information Technology
10th April 2013. Vol. 50 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SOFTWARE AND HARDWARE PARTITIONING BASED ON GENETIC ALGORITHM

[1] YUE-MIN WANG
[1]Assoc. Prof., Computer Science Department, Suqian college, Suqian Jiangsu 223800, P. R. China
E-mail: [1] yuemin_wang01@163.com

ABSTRACT

Data stream-based multimedia application is studied and a software/hardware partitioning method is proposed using both flexible granularity mechanism and extended mapping mechanism in this paper. This partitioning method focuses on genetic algorithm (GA), considering flexible granularity mechanism and extended mapping mechanism, designing special genetic coding method--double-link coding structure and corresponding genetic operations, meanwhile, the method makes GA and the solving software/hardware partitioning issue seamlessly connected, and the issue can be perfectly settled under the operation framework of GA. This method can effectively improve the hardware/ software/hardware partitioning quality, which has been proved by experiment.

Keywords: *Multimedia Application, Software/hardware partitioning, Genetic Algorithm (GA)*

1. INTRODUCTION

Software/hardware partitioning Method is based on Genetic Algorithm to solve software/hardware binary mapping, software/hardware extended mapping, flexible particle and a series of related problems. Because of the complexity of the Software/hardware partition problems, basic genetic algorithm can't meet the requirements, the basic genetic algorithm can not meet the requirements, so the basic algorithm needs to be improved to adapt to the need to resolve software/hardware partitioning problem and seamlessly be integrated into the entire partition process, which is the focus of this research [1]-[2].

2. BASIC GENETIC ALGORITHM

The basic genetic algorithm process is shown in Figure 1.Genetic algorithm starts from a population representing probable potential solution set. When generating the initial population, the first step is to realize from the phenotype to genotype mapping namely coding work, usually with binary coding. Initial population is produced in accordance with the principle of survival of the fittest, and the evolution of each generation to produce better and better approximate solution. Each generation of individuals are selected according to the size of individual fitness in problem domain, and by means of genetic operators to select a combination of crossover and mutation a new population produce [3].With the whole process, new population would be more responsive than the previous generation, and the last population the best individual can be used as approximate optimal solution of the problem when decoded. Genetic process contains a chromosome coding, individual fitness evaluation, selection operation, crossover operation and mutation operation and several key operations [4].

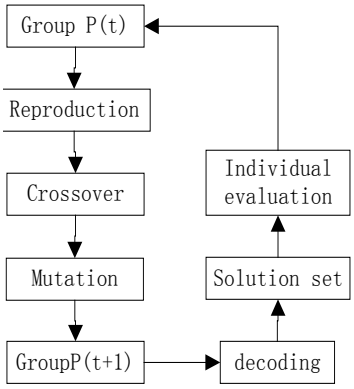


Figure 1 Basic Genetic Algorithm

3. CODING SCHEME DESIGN

When using genetic algorithm to solve practical problems, genetic coding should be considered at first, which is the key to design genetic algorithm. Genetic coding method not only determines the individual chromosome arrangement form also determines decoding methods from genetic type of individual searching space of transformation to representation type of the solution space, at the

Journal of Theoretical and Applied Information Technology
10th April 2013. Vol. 50 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

same time ,which also directly affect the genetic operation such as crossover, mutation operation methods and operating efficiency For the software/hardware partitioning problem, because of the high complexity of the problem, many factors considered, the genetic code design has so many challenges. It is also the difficulty lying in this study.

With the difference from the usually Software/hardware partition problem, Software/hardware partition problem has the following features:

Flexible granularity rather than fixed granularity. Based on this feature, hierarchical data flow diagrams is introduced to meet the needs of the design as shown in Figure 2, Where C, D.are complex nodes, B,E are simple node, and source, sink are start and end nodes separately, the last two nodes are virtual nodes, which do not affect the design.

Consider the software/hardware extensions mapping problem rather than just binary mapping of software/hardware .These characteristics need to be taken into account in the design of genetic encoding.

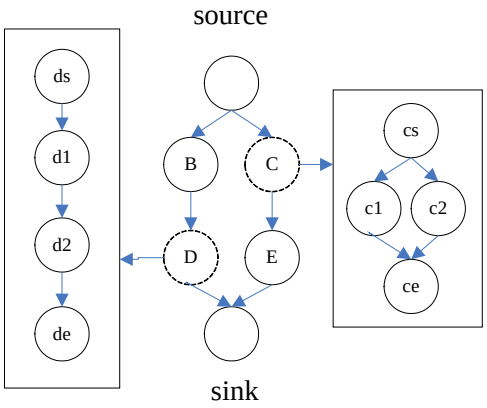


Figure 2 Hierarchical Data Flow Diagram

Considering all kinds of factors, coding scheme is posed as Figure 3.

In nature, the forms of a simple biological structure are usually haploid chromosome, but there are also some of the diploid or polyploidy chromosome structures, within which diploid structure contain two homologous genome, wherein each chromosome contain the gene information of the same functions, usually with dominant and recessive quality[5]. These two kinds of genes showing the individual entry type is determined by the following rules: In each locus, when one of two homologous chromosomes is dominant, the gene corresponding traits dominant; when two homologous chromosomes in the corresponding genes are recessive, the gene corresponding to the trait is expressed as hidden [6].

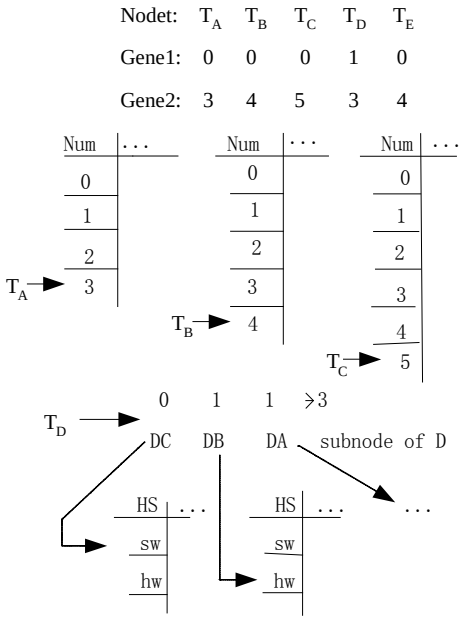


Figure 3 Coding Design

Proposed coding scheme in this paper borrow the diploid structure and concept with improvement: similar to the double-stranded diploid chromosome coding structure, in which one chain is a fixed-length binary encoding, indicating whether the node sub-graph is replaced, which is used to solve the flexibility and granularity of exploring issues; another chain uses fixed-length real number coding node to indicate parameter selection for software/hardware binary mapping and extended mapping problem. Showed in Figure3, the value of node TD as chromosome chain gene is 1, indicating that the node is a complex node and will be replaced by the corresponding sub-graph when exploring the flexible granularity.

However, only the first chromosome chain can not completely solve the a flexible granularity exploration and node mapping including binary mapping and the extension mapping problems, which can be solved with the second chromosome chain. The second chromosome chain is coded using real-coded. For the same locus, according to the value of the chromosome chain, the second chromosome chain decoding function Decode (g_i) will be different : Assuming K_i is the number of implement parameter table entries of node Ti ∈ Vt. g_i is the chromosomal locus, $r_i^{(1)}$, $r_i^{(2)}$, respectively,

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

are the locus values at g_i of the first chromosome chain and the second chromosomal chain. When sg_i is the complex node, it correspondents to the number of nodes of the corresponding sub-graph (not including the sink / source node). Then the decode functions Decode (g_i) of the second

$$Decode(gi)=\begin{cases} r_i^{(2)}\%k_i, & \text{if } r_i^{(1)}==0 \\ Binary(r_i^{(2)})[\,sg_i\text{-}1{:}0], & \text{if } r_i^{(1)}==1 \end{cases}$$

chromosome is:

Form the above expression, when r_i (1) = 0, that is, when the node is simple node or complicated node but not replaced with sub-graph in the process of software/hardware partition, the decode function 1 is used: the value some entry of a parameter table is got when taking the real value of second chromosome chain modulo the number of entries of the implement parameter table. The process is as the nodes T_A, T_B, T_C showed in Figure 3.

When r_i $^{(1)}$ = 1, i.e. when the node is a complex node and is substituted with sub-graph in the process of software/hardware partition, decoding function 2 is used: This value is converted to binary representation, according to the number of nodes in sub-graph (not including the sink / source), following the sequence of a pre-determined sub-graph node, the value is matched with the binary representation of the values from low to high. As node T_D Figure 3 showed, its sub-graph nodes is 3, fixed sequence is DC, DB, DA form low to high. The real value of its second chromosome at its locus is 3→011. Matching from low to high, DA→ 1, DB→1, DC→0 is obtained. According to the top of the first assumption, without further hardware implementation explore for the sub-graph node, implement parameter table of sub-graph nodes has only two entries, representing software implementation and a hardware implementation respectively. According to the result of matching results, binary mapping values of software/hardware are obtained directly. By this treatment method, the process of a flexible granularity of the software/hardware partition exploration can be supported.

3.1. Genetic Operation Based on Double-link Coding

In the basic genetic algorithm, the survival of the fittest mechanism is implemented by selecting operation [7]. Higher fitness individuals will also have the opportunity to be inherited to the next generation .While using the roulette wheel selection method and optimal retention strategies, the best individual in the previous generation are compared with the worst contemporary individual. If the former is better, the previous generation of the best individual will replace the worst individual in the present generation, which guarantees the best individual will not be destroyed by genetic operations[8]. In addition, the software/hardware partitioning method can treat the target system performance or hardware consumption as a constraint, which can be selected depending on the design requirements. F_i^A is the fitness function when optimizing system performance (O (T)) as hardware consuming for constraints (C (A)).

Assuming T_i is the system execution time value of the individual i, A is the consumption value of hardware resources of individual i. The T_{max} ,A_{max} ,a, b are sufficiently large constant value,

$$Fi=\begin{cases} a\text{-}T_i/T_{max}\text{-}(A_i\text{-}C(A))/A_{max} & \text{if } A_i> C(A) \\ b\text{-}\ T_i/T_{max} & \text{if } A_i<= C(A) \end{cases}$$

and b> a + 1> 2.

According to chromosome coding scheme in Figure 3, genetic crossover operation of the program is designed, as Figure 4 showed.

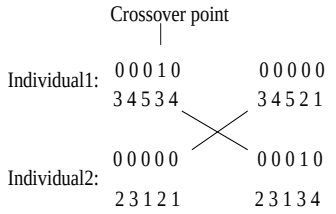


Figure 4 Genetic Crossover Operation Programs

Using the method similar to the commonly used in the single-point crossover in the binary-coded, one locus on chromosome is randomly selected, crossing over each other between two matching individuals. Despite the use of double-linked chromosome encoding method, which uses the hierarchical structure, the length of double-linked chromosome are equal, so simple cross can be proceeded simultaneously for the double-link.

The local search capabilities can be improved and the diversity of individuals is maintained by genetic diversity operation [9]. For the double-linked chromosome schema as Figure 2 showed, genetic variation program is designed based on the uniform mutation: pre-setting a minimum constant value mp as the mutation probability of the genes in the chromosome, for every locus on chromosome, a random value rd is generated by the random

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

number generator. Meanwhile, the gene of the locus will be mutated when rd<mp. Special encoding method using the double-stranded chromosome also reflected in mutation operations, i.e. if the node corresponding to the current loci is a simple node, then the locus at the first chromosome chain value does not participate in variation, the second chromosome chain value will mutate: An random generated integer value rk, taking this value modulo the value k_i of the parameters array item to the corresponding node of related loci, i.e. rk%k_i. By the means of this way, simple node is mutated. When the loci corresponding node is a complex node and need variation to the loci according to the gene mutation probability, make the mutation of the first chromosome chain and the second chromosome to the loci with 50% of the probability, generating a random number in [0, 1] using random generator. if this value is less than 0.5, then value to the loci at the first chromosome chain mutates(complementation), otherwise, the loci in the second chromosome chain at the loci would mutate with the mutation method as the simple node. Variation process is as figure 5 showed.

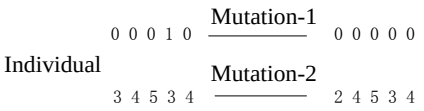


Figure 5 Genetic Mutation Operation Program

3.2. Performance Estimation Method Based on the Critical Path Scheduling

The genetic algorithm in running needs to decode the individual to determine the individual software/hardware binary mapping and extension mapping problem. According to the design of genetic coding, decoding method is obvious. However, relying solely on individual decoding does not completely determine the size of the individual fitness, so it need obtain hardware area of each individual resource consumption and system performance[10]-[12], that is, to determine Ti， Ai value in the fitness function. Since the determination of these values is inside of the algorithm loop, requiring repeated calls, so it's unrealistic to obtain these parameters based on the actual circumstance. The precise values for these parameters approximation obtained through the fast and efficient estimation method is an effective method.

Estimates of the values of A_i(A_{total}) is relatively simple, and by obtaining the type and number of the system consumption of hardware resources, the simple accumulation is able to obtain the hardware resources consumed by the system area, as (3) showed:

$$A_{total}=\sum rs_i * ns_i * au_i$$

Assuming M for the number of the individual consumed hardware resources, rs_i for the i resource type of system consumption, ns_i for the number of consumed resources, au_i for the hardware area value of such resources (from the library).

However, the determination of system performance can not use this simple accumulation to obtain, it needs to introduce the task scheduling mechanism, obtained on the limited resources of scheduling tasks (including communication tasks).

Through the individual decoding process, software/hardware binary mapping of the node, the implementation of the node selection information and edge information abstracted from the graph model can be obtained, which has been completely satisfy the task scheduling condition[13]. Different from the common in the field of scheduling problem, through the decoding process, it has been clearly determined which node will be mapped to which execution unit and the implement parameters of the node, so the scheduling mechanism mainly confirms the two questions:

Node priority in the scheduling : F_{cycle}.

Node temporal position in the whole scheduling graph.

Scheduling mechanism based on the critical path used, the most critical nodes in the path are assigned the highest priority to attain priority scheduling of these key nodes, through this mechanism to optimize the overall length of scheduling and to optimize system performance. Further, in order to reduce the difficulty of scheduling, accelerating the whole scheduling process, the design uses a non-preemptive scheduling mechanism. That is to say, during execution of a task, some task can not be interrupted by other tasks (in addition to the interrupt service routine (ISR)) until the task is completed. A complete period of task execution (F_{cycle}) includes obtaining the desired data (R_{cycle}) from the external, internal execution (I_{cycle}) and the output (W_{cycle}). as shown in Figure 6, when the communication unit to perform the same internal communication, it can be ignored.

In this study, the fixed target architecture uses a single CPU + the single ASIC + single bus structure as the target architecture of the system. Such a

Journal of Theoretical and Applied Information Technology
10th April 2013. Vol. 50 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

target architecture determines scheduling resources available in the whole scheduling process with the CPU, ASIC (for addition to communication node scheduling) and the bus (for scheduling) of the communications node, wherein the CPU and the bus are a serial execution apparatus, the hardware ASIC is a unit that can be executed in parallel. Scheduling graph is as Figure 7 shown.

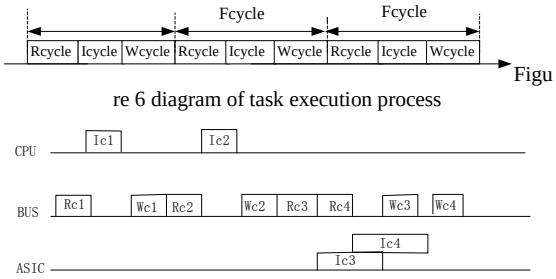


Figure 6 diagram of task execution process

Figure 7 Task Scheduling Diagram

Scheduling mechanism based on a critical path is to calculate earliest start execution time (ASAP) of the node (excluding the communication nodes) involved in the scheduling and the latest start time (ALAP) value. ALAP-ASAP value as the node priority standard of judgment, the node is a key node when ALAP = ASAP. All key nodes in the path form the critical path. Since the introduction of a flexible granularity in software/hardware partitioning process, the whole scheduling process is divided into two steps: firstly, schedule all the complex nodes substituted with sub-graph among the genetic individuals and obtain the scheduling length. Secondly, schedule at the top-level, i.e., anti-sign complex nodes information replaced by sub-graph from step one. The complex nodes exist in the form of a single node in the scheduling process. The genetic individual system performance and system scheduling length value can be obtained through the top layer scheduling.

3.3. Algorithm Process

The whole algorithm flowchart is as shown in the following Figure 8. This algorithm flow path integrated with the proposed coding scheme and corresponding genetic method of operations, such as selection, crossover and mutation. With the different from the basic genetic algorithm, in addition to special coding method and corresponding genetic operation, repairing process to the invalid solution is added.

In fact, because software/hardware partition is constrained by C ( ), according to genetic coding scheme, it can't guarantee all the produced individuals can satisfy the system constraint C ( ) according to this code rule and it may produce some invalid individuals, including invalid individuals of the randomly initial group at the beginning stage and all the produced new individuals throughout all of genetic operations. Invalid proportion of individuals in the population will increase with the tight constraints of system. Although some invalid individual can provide valuable gene fragment, they increase the space to explore, affecting the operating efficiency of the algorithm. Repairing operating to invalid individuals changes the invalid into effective individuals and it can greatly enhance the operating efficiency of the algorithm and accelerate the convergence of the algorithm.

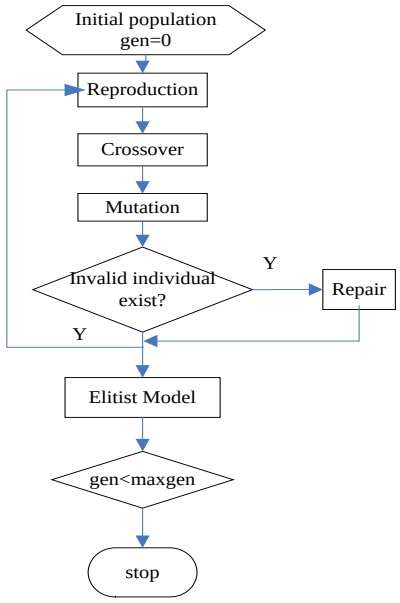


Figure 8 Algorithm Running Processes

The system hardware overhead of C (A) as a constraint, for example, when some individuals in the population consume hardware resources is greater than the C (A), the individual is considered to be invalid individuals. In the process of patching operations, it need reduce the usage of the individual hardware, so that the hardware resource overhead is less than or equal to C (A), which will turn invalid individual into effective individual.

In this kind of situation, the invalid individual repair plan is put forward: simple nodes and complex nodes mapping to hardware implementation among the individuals sort according to the priority as the following definition of $ratio_i$. The smaller the value is, the higher priority becomes.

$$ratio_i=(T_i^{sw}-T_i^{hw})/A_i^{hw}$$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

As shown in above, when loci i corresponds to simple node or complex node and not replaced by sub-graph, T_i^{sw}, T_i^{hw}, A_i^{hw} represent respectively software implementation parameters, selected hardware implementation performance and area parameters when hardware extended mapping ;When the node is a complex node and replaced by sub-graph node, T_i^{sw} represents the software implementation parameters of the complex node not replaced. T_i^{hw}, A_i^{hw} represent the hardware performance parameters (sub-graph scheduling) and hardware area value (sub-graph of mapping to hardware sub-graph node consumption sum of the hardware area) of the individual nodes been replaced by the sub-graph.

In the repairing operations, according to node priority from high to low, change the node implementation form hardware execution unit into implementation on CPU, especially, when the node is complex node and be replaced by sub-graph, recover implementation of the node to the state without substituted, i.e. turn the loci value of the first chromosome chain from 1 into 0, then the entire complex nodes as a whole moves to implementation on CPU. Through this method, invalid individuals can be effectively repaired into valid individuals.

In addition, a variable flaton and a parameter value ITER_GE are added in algorithm. Flaton and ITER_GE are used to control whether using enable flexible granularity explore mechanism in the Software/hardware partitioning process. In software/hardware partitioning initialization process, enable flexible granularity mechanisms is not used, but to do the software/hardware binary mapping and extended mapping. If you still do not get individual fitness improvement after ITER_GE generation, then enable flexible granularity mechanism, finer partition granularity explore is conducted to further improve the quality of software/hardware partitioning, as shown in Figure 9.

In above methods, flexible granularity enabled processes can play the flexible of granularity mechanisms and the extended mapping mechanism, thereby enhancing the quality of the software/hardware partition final partition of quality.

Partitioning results can be optimized after iterations of the algorithm, including node binary mapping as well as expansion mapping scheme.

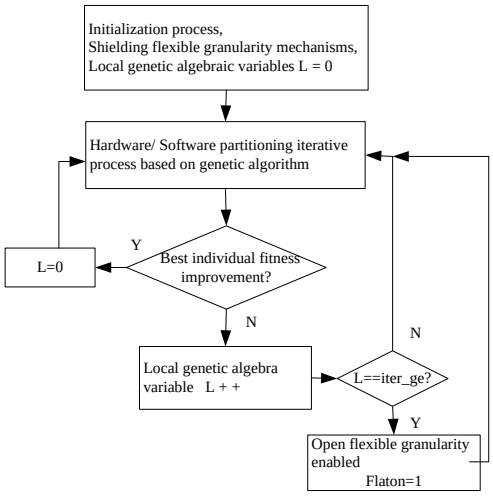


Figure 9 Flexible Granularity Mechanism Enabled Process

4. THE ALGORITHM VALIDITY TESTING BASED ON RANDOM GRAPH

With software/hardware partitioning algorithm in C / C + + language, a large number of random graph test have been conducted in order to verify the effectiveness of the algorithm. During the test, the use of random graph TGFF (Princeton University developed custom output random graph generator) developed by Princeton produce a large number of customized random graphs as the basis on test.

Several random reference point given as below, as shown in Table 1: From the table, it can be seen the random reference number of nodes and edges of the number and complexity of the number of nodes increased gradually, which can test quality of the partition method of the proposed software/hardware partition method in the case of different reference points.

Table 1 Random Benchmark Of Basic Situation

Name	Node number	Edge number	Complex node number	Complex node ratio
Rand11	11	19	0	0%
Rand42	42	77	4	9.5%
Rand88	88	160	10	11.36%
Rand124	124	230	14	11.3%

For stochastic input table reference point, the following comparative test likes as follows:

Compared with partitioning method having no flexible partition granularity mechanism

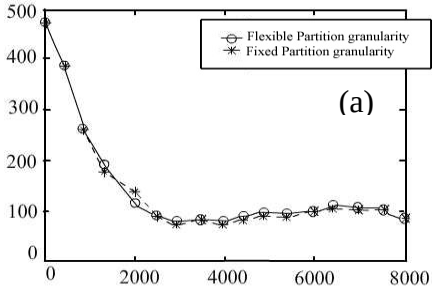
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Compared with the situation without extended binary mapping, i.e. not considering the hardware design space exploration.

4.1. The Longitudinal Comparative Experiment Flexible Partition Granularity Mechanism

In order to analyze the impact of the quality of the software/hardware partition under the flexible granularity mechanism, longitudinal comparative experiments are proposed. Based on genetic algorithm to implement software/hardware partition, through the enable control to the flexible granularity mechanism, the impact level to software partition quality is got under the mechanism.

Based on the test result to the above provided random datum point, the experimental results are showed in figure 10. Analysis represents that due to the random test chart Randll, complex node number is 0, explaining whether to adopt a flexible partition size mechanism, this mechanism is invalid. From Figure 10, it can be seen that as constraint (area) enhanced continuously, the system performance declined gradually, which conforms to the real design. Meanwhile, since the flexible granular mechanism ineffective and the genetic algorithm have a random uncertainty, therefore flexible partition granularity can make the two curves coincide roughly (but not completely coincide). For random test chart Rand42 / ralld88 / rand124: The number of complex nodes in random test chart Rand42 is four, accounting for 9.5% of the total number of the top node. Complex node number in Rand88 is 10, accounting for 11.36% of the total, complex node number in Randl24 is 14, accounting for 11.3% of the total. Under these test input, using the flexible granularity mechanism can improve the final quality partition. It can be seen that, from the figure, the introduction of the flexible granularity mechanism improve the quality of the whole partition compared with a fixed granularity, especially in the case of very strict constraints. Flexible granularity can meet a strong constraint.



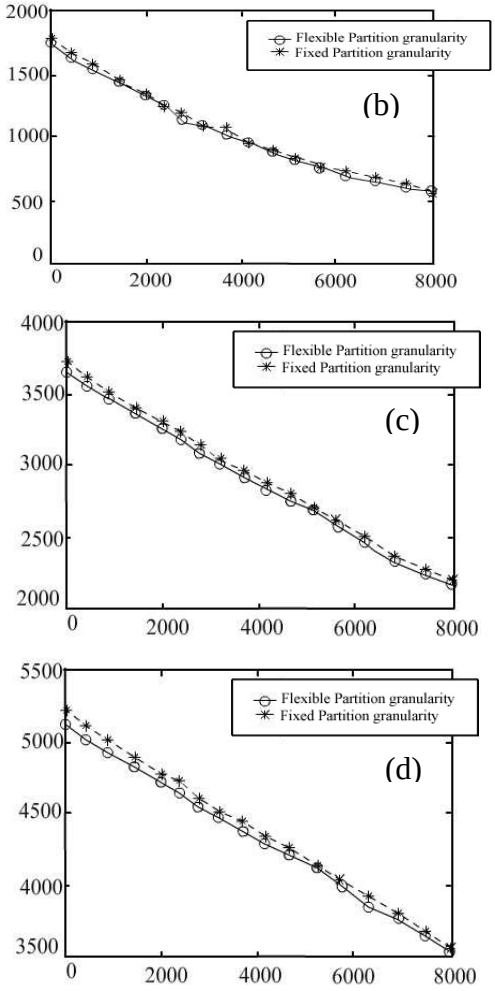


Figure 10 Partition Granularity Mechanisms Impact Analysis

4.2. Longitudinal Contrast Experiment Extended Mapping Mechanism

In order to analyze the impact of the extended mapping mechanism on the quality of software/hardware partitioning, comparative experiments of software/hardware partitioning method based on genetic algorithm-based longitudinal is conducted. According to random reference point in the initialization phase of the hardware/software partitioning process has been randomly obtained from some hardware implementation point in hardware expansion space as the only hardware implantation representation in the whole partition process, only hardware/software binary mapping is conducted , ignoring the extended mapping. Enable control of hardware design space obtains early and later comparison data, analysis of the impact of the extended mapping mechanism on the quality of the

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

software/hardware partition. Random benchmark points provided above are used to conduct experiment, experimental results as shown in Figure 11. From the figure, it can be seen that the introduction of extension mapping mechanism improved the quality of software/hardware partition greatly, with the degree of improvement with respect to the test input.

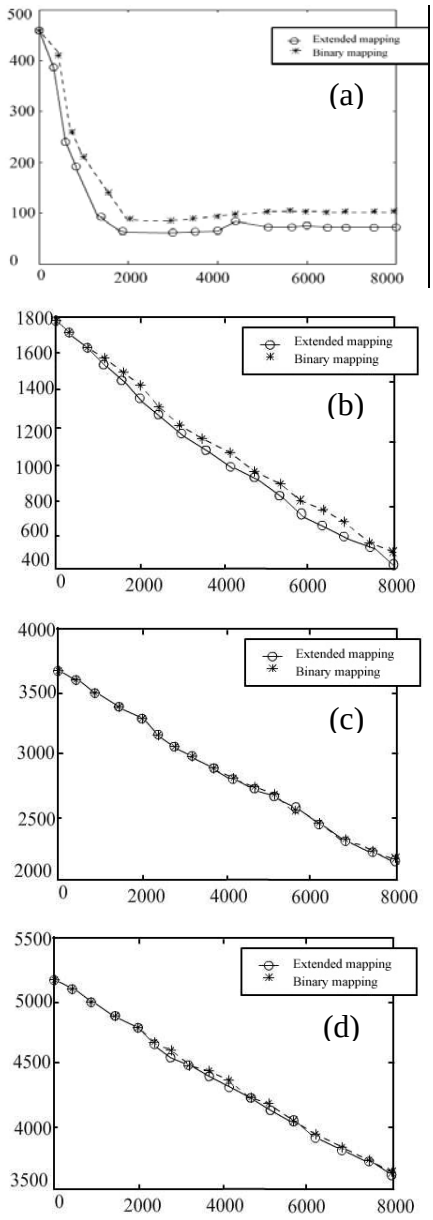


Figure 11 Extended Mapping Mechanism Impact Analysis

5. CONCLUSION

In this study, a detailed analysis of the basic questions need to be addressed in the software/hardware partition, including floating-point to fixed-point conversion, the system diagram representation of the model, the software parameters and hardware parameter extraction and anti-sign issues, through the analysis of these problems, the software/hardware partitioning problem is refined to a mathematical combinatorial optimization problems, making it possible to solve the design problems of system software/hardware partition by the introduction of mathematical optimization.

Software/hardware partitioning method is proposed in this paper, genetic algorithm as the basic algorithm framework, combining flexible partition granularity and extended mapping mechanism and the double-linked coding genetic algorithm is developed. According to this particular duplexes encoding mechanism, the corresponding genetic operations comprising selecting operations, crossover operations, and mutation operations were designed. For the arisen invalid solution in the partition process, repairing operations were proposed to improve the efficiency of the algorithm. Compared with the successfully implemented software/hardware partition method in the research field, the proposed method including the flexible granularity mechanisms and the extended mapping mechanism and an enable process with flexible granularity integrated these two mechanisms into the operational framework of the genetic algorithm seamlessly, which is the definite feature of the software/hardware partitioning method.

ACKNOWLEDGMENT

This work is partially supported by science and technology project of Suqian College (No. 0712 and 811314010422)

REFRENCES:

[1] Prakash G. Burade, Dr. J. B. Helonde, "By Using Genetic Algorithm Method For Optimal Location Of Facts Devices In The Deregulated Power System", *Journal of Theoretical and Applied Information Technology*, Vol. 17. No. 1, 2010, *PP*. 64-71.

[2] YE Hua, Wu Jigang, "Computing Models and Algorithms for Complex Co-design Systems", *Journal of University of Electronic Science and Technology of China*, Vol. 03, 2011, *PP*. 333-345.

[3] LI Zheng-min, GUO Jin-jin, LV Ying-ying, "Simulation Research on HW-SW Partitioning of Embedded System Based on Ant Colony

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Algorithm", *Computer Simulation*, Vol. 10, 2011, *PP*. 204-207.

[4] Wang Lei, Kang Qi, XiaoHui, *et al*. "A Modified adaptive particle swarm optimization algorithm", *IEEE*, Shanghai, 2005, *PP*. 209-214.

[5] Ahmed A. A. Esmin, Germ Ano Lambert-Torres, Antonio C. Zambronide Souza, "A hybrid particle swarm optimization applied to loss power minimization", *IEEE Transactions on Power systems*, Vol. 20, No. 2, 2005, *PP*. 859-866.

[6] Daniel W. Beranger, Douglas H. Werner, "Particle Swarm Optimization Versus Genetic algorithms for Phased Array Synthesis ", *IEEE Transactions on Antennas and Propagation*, Vol. 52, No. 3, 2004, *P*P. 771-779.

[7] Sangwook Lee, Sangmoon Soak, Sanghoun Oh, "Modified binary particle swarm optimization", *Natural Science*, 2008: *PP*. 1161-1166.

[8] Chatterjee A, Siarry P. Nonlinear, "inertia weight variation for dynamic adaptation in particle Swarm optimization", *Computers and Operations Research*, Vol. 33, No. 3, 2006, *PP*. 859-871.

[9] LI Lan ying, Zhang Lei lei, Shi Min, "Improved two-dimension enhanced greedy hardware/software partitioning algorithm", *Computer Engineering and Applications*, Vol. 21, 2009, *PP*. 64-67.

[10] SUSMI ROUTRAY, "An Enhanced Genetic Algorithm For Dynamic Routing In ATM Networks", *Journal of Theoretical and Applied Information Technology*, Vol. 16, No. 2, 2010, *PP*. 153-158.

[11] C. Zhang, "Comparisons of selection strategy in genetic algorithm", *Computer Engineering and Design*, Vol. 30, 2009, *PP*.5471-5478.

[12] I. De, B. Chanda, B. Chattopadhyay, "Enhancing effective depth- of- field by image fusion using mathematical morphology", *Image and Vision Computing*, Vol. 24, No. 12, 2006, *PP*. 1278- 1287.

[13] Wu J. G., Srik anthan T., Jiao T., "Algorithmic aspects for functional partitioning and scheduling in hardware / softw arecodesign", *Design Automation for Embedded Systems*, Vol. 12, No. 4, 2008, *PP*. 345-375.