M.S. THESIS

An Efficient Method for Partitioning Initial Chromosomes in the Island Model Genetic Algorithm

섬모델 유전 알고리즘에서의 효과적인 초기 유전자 분할 방식

FEBRUARY 2014

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

최현돈

An Efficient Method for Partitioning Initial Chromosomes in the Island Model Genetic Algorithm

섬모델 유전 알고리즘에서의 효과적인 초기 유전자 분할 방식

지도교수 문 병 로

이 논문을 공학석사 학위논문으로 제출함

NOVEMBER 2013

서울대학교 대학원

컴퓨터 공학부

최 현 돈

최현돈의 공학석사 학위논문을 인준함

DECEMBER 2013

위 원 장	신 현 식	(인)
부위원장	문 병 로	(인)
위 원	박 근 수	(인)

Abstract

In this thesis I propose a proper method of partitioning initial population in Island Model GAs. Island Models are a popular and efficient way to implement a genetic algorithm on a parallel machine. In an Island Model each machine maintains its own subpopulation using a genetic algorithm for search. In the basic algorithm of parallel GAs, all populations are shuffled randomly across the different "islands". That is, the initial population is partitioned in random, although we can control this unknown factor on purpose. I designed two methods of partitioning initial population which considers the distances among population rather than random partitioning. Experimental results show that one of the methods brings a considerable improvement in finding a global optima.

Keywords: Genetic algorithm, Island Model, graph partition problem
Student Number: 2012-20877

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Island Models are a popular and efficient way to implement a genetic algorithm on a parallel machine. In an Island Model each machine maintains its own subpopulation using a genetic algorithm for search[8]. The Island Model involves running several single population genetic algorithms in parallel. Each "island" is an Simple Genetic Algorithms(SGA) with its own subpopulation[1]. The machines work in consort by periodically exchanging a portion of their populations in a process called migration. Parallel Island Models have often been reported to display better search performance than serial single population models, both in terms of the quality of the solution found and effort as measured in the total number of evaluations of points sampled in the search space[6, 9]. One reason for the improved search quality is that the various "islands" maintain some degree of independence and thus explore different regions of the search space while at the same time sharing information by means of migration[10].

In the basic algorithm of parallel GAs, all populations are shuffled randomly across the different "islands", and repeat this shuffle at regular intervals[11].

That is, the initial populations are also shuffled and partitioned in random, although we can control this unknown factor on purpose. If we control the process of assigning initial populations, would it be meaningful for the solutions of GAs? If it is not meaningless, then what is the effective way of controlling the initial populations? This paper plans experiments to answer these questions. Knapsack problem is introduced and solved to compare the result, using the Island Model. The ways of assigning initial populations are also introduced and evaluated.

The remainder of this thesis is organized as follows. I explain the Island Models and introduce the problem to proceed the experiment in chapter 2. In Chapter 3, basic methods of partitioning chromosomes and their detail implementations are described. Chapter 4 provides experimental results and discussion, and finally the conclusion is given in Chapter 5.

Chapter 2

Preliminaries

In this chapter, I explain the Island Models and introduce the problem to proceed the expriment. The Island Model is based on the single Simple Genetic Algorithm,

2.1 Island Model GAs

In SGA, the whole population take part in the process, sequentially. There is no parallel part in this algorithm. When several SGAs process their job with their own populations, and sometimes exchange some part of their population to each other, the algorithm becomes parallelizable. This algorithm is called "Island Model GAs". A representative Island Model GA structure is given in Algorithm 1.

The Algorithm starts with the generation of a population and splitting the population into n subpopulations. This subpopulation is called "island". After partitioning the population, the part of SGAs are run in each island. The part

Algorithm 1 Island Model GA

create an initial population of fixed size, and record in P;

▷ P : the set of all chromosomes

split P into $P_1, P_2, ...P_n$ randomly;

▷ n : the number of islands(the number of separated subpopulation set)

▷ $P_1, P_2, ...P_n$: the set of chromosomes of each island

repeat

  for all islands **do**

    repeat

      select $parent_1$ and $parent_2$ from population;

      $offspring \leftarrow$ crossover($parent_1$,$parent_2$);

      mutation($offspring$);

      replace an appropriate chromosome with $offspring$;

    until the set number of generation is proceeded

  end for ▷ This loop is parallelizable.

  let some chromosomes to migrate between islands;

until stopping condition

return the best chromosome;

of SGAs consists of *selection*, *crossover*, *mutation*, and *replacement*. In a selection part, two chromosomes in a subpopulation are selected based on the probability distribution to make a new chromosome. These two chromosomes are combined and produce a new chromosome(offspring) in the crossover part. The offspring takes the characteristics of its parents. This offspring can be randomly changed partially according to a user-definable probability, in the mutation part. This part helps maintain the diversity of a subpopulation. And Finally, the offspring is replaced to the subpopulation in the replacement part. The algorithm described in Algorithm 1 is the $steady - state$ GA, that is, the only one offspring is created in one generation. These four stages are the basic part of SGAs, and each island of Island Model can run these steps in parallel. While the four steps of SGAs are proceeding, some chromosomes can be moved from one island to another island. This is called *migration*. Like a mutation part, migration also help maintain the diversity of a subpopulation. In this thesis, the best individual in each island is copied and moved to other islands, and the worst individual is deleted to maintain the size of the island. This migration method comes from [8].

In the Island Model GAs, there are many factors that would improve the best fitness, such as the number of splited semi-isolated subpopulations, size of each subpopulations, method of migration, and so on. Among these factors, this paper focuses on the method of partitioning initial population. That is, this paper introduces the effects of controlling the initial populations of each subpopulations.

2.2 Island Models with Local Optimization

When problem specific information exists it is advantageous to consider a Hybrid GA. They combine local search heuristics with crossover operators. Genetic algorithms may be crossed with various problem-specific search techniques to form a hybrid algorithm that exploits the global perspective of the GA and the convergence of the problem-specific technique.[4] Like usual hybrid GAs, Island Models apply a local optimization algorithm after crossover and mutation. The local optimization moves chromosome toward a local optimal solution. Although there are countless local optima, the solutions applied with a local optimization usually form crowds with each other. Thus, the geometric distances among the chromosomes applied with a local optimization become more distinct than those of chromosomes with no local optimization. In this reason, I also applied a local optimization to the initial population so that the result of partitioning chromosomes are more distinct than before.

2.3 Knapsack problem

The knapsack problem is a problem in combinatorial optimization: Given a set of items, each with a mass and a value, determine the subset of items to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible. More formally, the kanpsack problem can be stated as follows.

$$maximize\, z = \sum_{j=1}^{n} p_j x_j$$

$$subject\, to \sum_{j=1}^{n} w_j x_j \leq c,$$

$$x_j \in \{0, 1\},\ j \in \{1, ..., n\}.$$

where p_i is a value of item i, x_i is 0 if item i includes in the subset solution, and 1 if not. w_i is a weight of item i, c is a given limit.

The solution of this problem can be presented as $x_1 x_2 ... x_n$, that is, a sequence of 0 or 1. Since the presentation of solution is very simple, calculation of the geographical distance between two solutions is also simple;we can just use Hamming Distance[2]. Geographical distances among solutions are very important factors in the partitioning initial population, thus I chose the knapsack problem to experiment Island Models.

2.4 Knapsack Heuristic

To take advantage of a Hybrid GA, a local optimization algorithm for knapsack problem is needed. Thus, I implemented a heuristic algorithm which is similar to 2-OPT in the traveling salesman problem, and the algorithm is described in Algorithm 2.

This heuristic is based on the greedy algorithm. The heuristic selects only one item to add or remove, and adjust another items to remove or add, considering the limit and value.

Algorithm 2 A Heuristic for Knapsack Problem

Input :$X = x_1x_2...x_n$

repeat

for i = 1 to n **do**

$X0 := x_1x_2...x_n$;

$gain := 0$;

if $x_i = 0$ **then**

gain = p_i;

cost := cost($X0$) + w_i;

while cost > limit **do**

find j, such that p_j is minimal if$x_j = 1$;

set $x_j = 0$ in $X0$, and recalculate the gain and cost;

end while

else

gain = $-p_i$;

cost := cost($X0$) - w_i;

while cost < limit **do**

find j, such that p_j is maximal if$x_j = 0$;

set $x_j = 1$ in $X0$, and recalculate the gain and cost;

end while

roll back the last modify in the last loop;;

end if

if $gain > 0$ **then**

replace X with X0;

end if

end for

until no improvement

Chapter 3

Methods of Partitioning

I present 2 methods for partitioning populations. First, we can gather some similar chromosomes into the same subpopulation. Second, we can separate chromosomes from the similar one. Of course we can assign chromosomes to the subpopulations randomly, and this method will play a role of control group. In this chapter, I decribe the above 2 methods in detail, and the way of how to implement them.

3.1 Detailed Methods

Consider a fully-connected, undirected, weighed graph $G(V, E)$. Each chromosome corresponds to a vertex, and the weight of the edge $E_{v1,v2}$ corresponds to the hamming distance of $v1$ and $v2$. If we want to partition this graph into k subgraphs, we can do it by cutting some edges in the graph G. To make the similar chromosomes far apart, we should cut edges with smaller weight(smaller weight means the closer Hamming distance, that is, the more similar chromo-

some), not the edges with bigger weight. If we consider the uniform partitioning so that the size of each subpopulation are uniform, then the problem of partitioning G into k islands so that the similar chromosomes are separated is identical with the k-way uniform graph partition problem.

Uniform graph partition problem is defined in the form of a graph $G = (V, E)$, with V vertices and E edges, such that it is possible to partition G into the same size of smaller components with least edges running between separated components. This problem is based on the non-weighted edge graph, but weighted edge graph can easily be converted to the non-weighted edge graph. For example, if you convert the weighted edge $E1_{v1,v2}$, you can draw additional edges between $v1$ and $v2$, so that the number of edge between $v1$ and $v2$ becomes $W(E1)$, the weight of the edge $E1$. So, the graph partition problem in the weighted edge graph can be defined as partitioning G into the smaller components with least sum of weight of edges running between separated components.

Let s be the sum of the weight of all edges in G, w be the sum of weight of edges running between separated components, and N be the number of all edges between the vertices in the same component. Then, the sum of weight of all edges between the vertices in the same component becomes $w - s$, and the average weight of it becomes $(w - s)/n$. It is trivial that the value of w is constant, and if the number of vertices of each component are uniform, the value of e is also constant. Thus, if the value of s gets smaller, then the average of weight of edges between the vertices in the same component gets bigger. In the same way, if we want to gather some similar chromosomes into the same subpopulation, we can solve the graph partition problem so that the value of s becomes the most.

3.2 Graph Partitioning

Since graph partition problem is NP-complete[3], if we partition the whole pupulation into small components considering the average of weight of uncut edges, we have to use a practical solutions based on local heuristics. One of the local heuristic procedure for partitioning graphs was introduced by Kernighan and Lin[5], and I modified this algorithm for the k-way graph partitioning. Let I_a be the *internalcost* of a, that is, the sum of the costs of edges between a and other nodes in subset A, where $a \in A$. Let E_aK be the *externalcost* of $atoK$, that is, the sum of the costs of edges between a and nodes in subset K. Furthermore, let $D_aK = E_aK - I_a$ be the difference between the external and internal cost of a and K. If a and b are interchanged, then the reduction in cost is

$$T_{old} - T_{new} = D_{aB} + D_{bA} - 2c_{a,b}$$

where $c_{a,b}$ is the cost of the possible edge between a and b, $a \in A$, and $b \in B$.

The modified algorithm is given in Algorithm 3. The pseudocode is also similar to the original KL[7].

Algorithm 3 Modified Kernighan-Lin for k-way Graph Partition

procedure KL-MODIFIED($G(V, E)$)

determine a balanced initial partition of the nodes into sets $N_1, N_2, ..., N_k$;

for i = 1 to k **do**

$N1_i := N_i$;

end for

repeat

compute D values for all v in $N_1, N_2, ..., N_k$;

for $i = 1$ to $|V|/2$ **do**

find (a, b), such that $g_i = D_{aNa} + D_{bNb} - 2c_{a,b}$ is maximal, where $a \in N_a$ and $b \in N_b$;

exchange a and b and save (a, b) in Queue;

remove a and b from further consideration in this pass;

update D values for the elements of all $N1_i$;

end for

find k which maximizes g_{max}, the sum of $g_1, ..., g_k$;

if $g_{max} > 0$ **then**

roll back the exchanges from kth-($|V|/2$)th;

end if

until $g_{max} <= 0$ **return** G(V,E)

end procedure

Chapter 4

Experiments

4.1 Experimental Setup

In this chapter, I provide experimental results. I generated 400, 800, 1600, 3200 items at random. The length of chromosome is the same as the number of item, so I also generated populations of each of the case. The size of population is 120 in all of the case. Each chromosome set is partitioned to 8 components after local optimization, with the method of 3 types: gathering some similar chromosomes into the same subpopulation(gather), separating chromosomes from the sililar one(split), assign chromosomes randomly(random). These partitioned chromosome sets work as initial inputs of Island Models. There is a migration part in the original Island Model, but the Island Models without migration part is run in a part of experiment to minimize variables. Final experiment runs Island Model with migration to evaluate the validity of the gathering and spliting population. The Island Model in this thesis is a $steady - stateGA$, to make convergence faster. All experiment is carried out 10 times, and the result

is the average of 10 experiments.

MapReduce in Hadoop is used to parallelize the job in the Island Model. The code was programmed in Java language, and compiled using javac version 1.6.0_27. The program was executed on a Xeon CPU 2.4GHz computer.

4.2 Distribution of the Initial Subpopulations

Before running a program, I present the analysis about partitioned initial subpopulations. According to Hamming Distance, the distance is 0 if the two chromosomes are exactly the same, and is the same as length of the chromosomes if the two chromosomes are completely different from each other. Naturally, a value of Hamming Distance is likely to increase as the number of items is increased. Therefore, when comparing the distance among different length of chromosomes, it is reasonable to compare the ratio of Hamming Distance to the length of chromosome. That is,

$$d(a, b) = H(a, b)/l$$

where $H(a, b)$ is a Hamming Distance between chromosome a and b, and l is a length of chromosome. In this case, $d(a, b) = 0$ if the chromosome a and b are exactly the same, and $d(a, b) = 1$ if the chromosome a and b are completely different from each other.

As I did in section 3.1, Consider a undirected, weighed graph $G(V, E)$. Each chromosome corresponds to a vertex, and the weight of the edge $E_{v1,v2}$ corresponds to the ratio of hamming distance of $v1$ and $v2$ to the length of chromosome. If $v1$ and $v2$ are in the different partition, there would be no edge between two. In that case, edge $E_{v1,v2}$ exists if only $v1$ and $v2$ are in the same partition. Then the average distances of subpopulations is

$$average\,distance = \sum_{e \in E} W(e)/|E|$$

where $W(e)$ is a weight of edge e.

Table 4.1 Average Distance of Initial Subpopulations

# of item	gather	random	split
400	0.082	0.093	0.096
800	0.121	0.134	0.138
1600	0.105	0.112	0.115
3200	0.118	0.123	0.125

Table 4.2 Average Distance of the Best Solutions of Initial Subpopulations

# of item	gather	random	split
400	0.077	0.073	0.066
800	0.129	0.093	0.097
1600	0.102	0.098	0.096
3200	0.119	0.108	0.106

Table 4.1 presents average distances of initial subpopulations. It is clear that the "gather" method has the smallest average distance, and the "split" method has the biggest one. This means that the implement of these methods are practically correct. The result of the "random" method is similar to the split method, but the result of the split method is bigger than that of random method anyway. It seems that there is no correlation between the size of a problem and the average distance. But in fact, the actual Hamming Distance becomes bigger as the size of problem grows. So the real distance of the chromosomes are much farther as the size of problem grows.

Table 4.2 present average distance of the best solutions of initial subpopulations. If the number of islands is 8, there are 8 best solutions for each islands. This table shows the average distance of these 8 solutions. It is difficult to find the pattern of random method, but it is clear that the result of gather method is bigger than that of split method. It can be assumed that the searching space of each island is far apart from each other in the gather method, thus the best solutions of each islands are also far apart from each other. On the other hand, It can be assumed that the searching space of each island is overlapped a little in the split method, thus the best solutions of each islands are a little closer.

4.3 Qualities of best solutions

I present the result of experiment in this section. The problem is a 3200-items knapsack problems, run in Island Models with 8 islands. I observed the property of subpopulations in terms of 200 generation, that is, 200 offsprings are made and replace in this term;this is a $steady - stateGA$. Figure 4.1 and 4.2 presents the results of the experiment without migration. Vertical axis means the relative fitness of the best solution in all islands, and horizontal axis means the elapsed generations. Relative fitness means $RF_n = F_n - F_0$, where F_n is the fitness value in nth generation. Number 1 on the horizontal axis means 200 generation, and I observed it 40 times. Figure 4.1 plots the data including initial state, and Figure 4.2 plots the data excluding initial state.

It is clear that the split method provides the better solution than gather method and random method. Gather method is the worst among the three methods, on the other hand. The fitness of the best solution seems to be similar until the 20th observation(4000 generation). But the best solution in gather method is frozen after the 20th observation, and the best solution in

split method goes on improving until 32th observation. Thus, can be assumed that the premature convergence is occured in the gather method. And the reason of premature convergence is the narrow searching space.

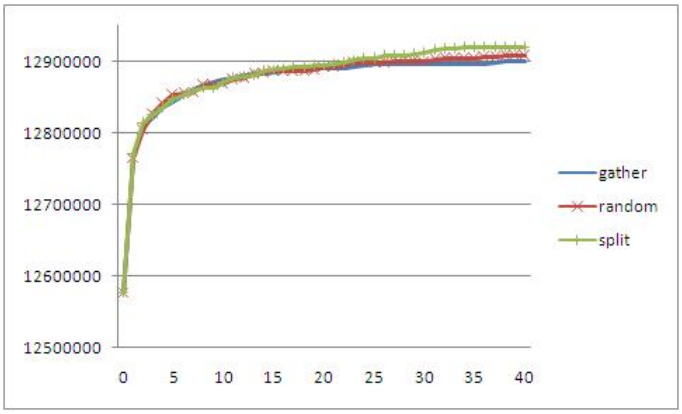

Figure 4.1 Change of the Value of Best Solution without Migration

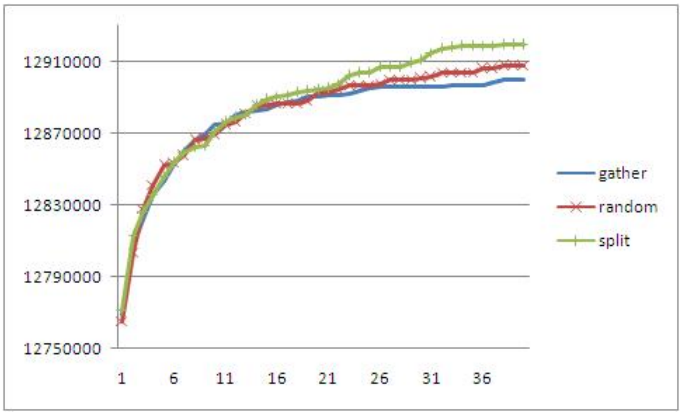

Figure 4.2 Change of the Value of Best Solution without Migration

Figure 4.3 and 4.4 presents the changing average distances along the generation. Vertical axis means the average Hamming Distance among the population like section 4.2, and horizontal axis means the elapsed generations. Figure 4.3 plots the data including initial state, and Figure 4.4 plots the data excluding

initial state. The average distance draws a downward curve in all method. As you can see Figure 4.4, the differences among the gather, random, and split methods does not appear in this observation. They are very similar to each other.

In other words, the speed of convergence is similar to each other, but the premature convergence occured earlier in the gather method. It is clear that the difference of the searching space of each islands made this result.

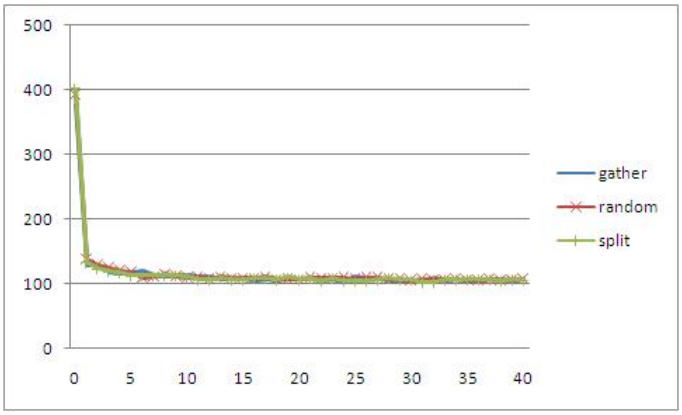

Figure 4.3 Change of the Value of Average Distance

Figure 4.4 Change of the Value of Average Distance

With migration, each island can share some part of searching space. Thus, the effects of the gather and split method in Island Models weaken. Figure 4.5 and 4.6 presetns the results of the experiment without migration. Vertical axis and horizontal axis is the same as 4.1 and 4.2, and Figure 4.6 also plots the data excluding initial state. The difference among the 3 method became smaller. But the split method still provides the best solution among the three methods. Exactly, the relative fitnesses of the gather, random, split method is 361644, 364963, 368035, respectively.

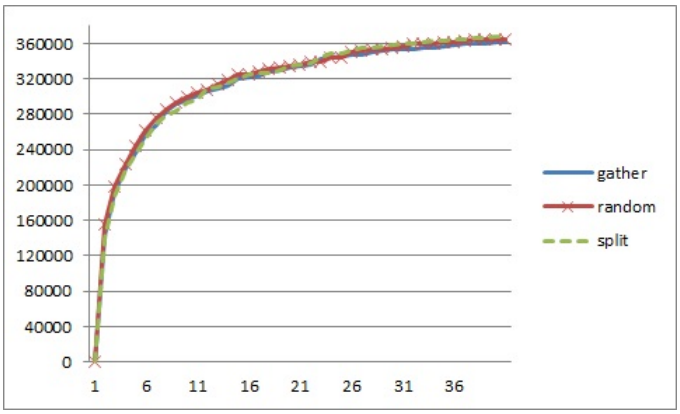

Figure 4.5 Change of the Value of Best Solution with Migration

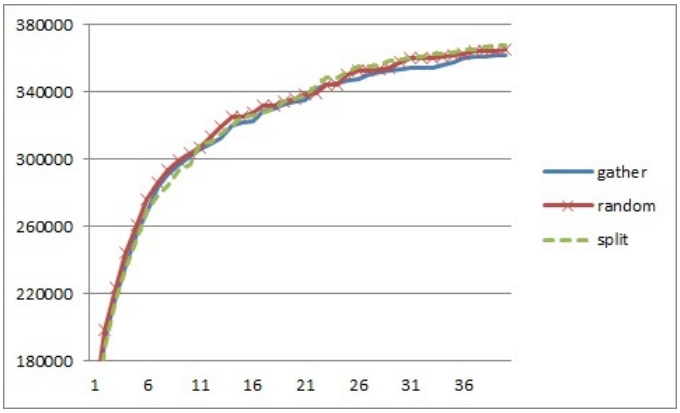

Figure 4.6 Change of the Value of Best Solution with Migration

Chapter 5

Conclusions

In this thesis, I proposed a proper method of partitioning initial population in Island Model GAs. I introduced two methods related to distances among chromosomes in the population. The idea of the methods were simple: gathering some similar chromosomes into the same subpopulation or separating chromosomes from the similar one. Then I introduced a graph partition problem so that this problem is equivalent with the problem of the introduced methods. Thus I could use a KL heuristic to implement these methods. KL heuristic is a local search algorithm, so the probability that its answer is not a global optima is very high. Neverthless, KL heuristic in this thesis is meaningful enough because its solution is good enough and its running time is efficient enough.

I experimented this idea with knapsack problem. First, distribution of the initial partition is analyzed. The result said that the searching area of the island in gathering method is narrower than each other, and the crowd of each island is far apart from each other. And the searching area of the island in spliting

method is wider than each other, and the crowd of each island is overlapped a little. This caused the difference in the time of occurance of convergence. To conclude, the split method returned the better solution more than the others. The difference among the three methods becomes smaller when migration is processed, but the split method still works best with migration.

As another issue for future study, the migration strategy using this idea is required, such as adapting the split method on the every migration. This strategy makes much more overhead for splitting the whole individuals in each migration, but it could improve the quality of the best solution of Island Models.

Bibliography

[1] GORDON, V. Scott; WHITLEY, Darrell. Serial and parallel genetic algorithms as function optimizers. In: ICGA. 1993. p. 177-183.

[2] HAMMING, Richard W. Error detecting and error correcting codes. Bell System technical journal, 1950, 29.2: 147-160.

[3] HENDRICKSON, Bruce; LELAND, Robert W. A Multi-Level Algorithm For Partitioning Graphs. SC, 1995, 95: 28.

[4] JAYALAKSHMI, G. Andal; SATHIAMOORTHY, S.; RAJARAM, R. A hybrid genetic algorithm—a new approach to solve traveling salesman problem. International Journal of Computational Engineering Science, 2001, 2.02: 339-355.

[5] KERNIGHAN, B. W.; LIN, S. An efficient heuristic procedure for partitioning graphs. Bell system technical journal, 1970.

[6] MUHLENBEIN, Heinz. Evolution in time and space-the parallel genetic algorithm. In: Foundations of genetic algorithms. 1991.

[7] Ravikumr, Si. Pi; Ravikumar, C.P (1995). Parallel methods for VLSI layout design. Greenwood Publishing Group. p. 73. ISBN 978-0-89391-828-6. OCLC 2009-06-12.

[8] WHITLEY, Darrell; RANA, Soraya; HECKENDORN, Robert B. Island model genetic algorithms and linearly separable problems. In: Evolutionary computing. Springer Berlin Heidelberg, 1997. p. 109-125.

[9] WHITLEY, Darrell; STARKWEATHER, Timothy. Genitor II: A distributed genetic algorithm. Journal of Experimental & Theoretical Artificial Intelligence, 1990, 2.3: 189-214.

[10] WHITLEY, Darrell; RANA, Soraya; HECKENDORN, Robert B. The island model genetic algorithm: On separability, population size and convergence. Journal of Computing and Information Technology, 1999, 7: 33-48.

[11] VERMA, Abhishek, et al. Scaling genetic algorithms using mapreduce. In: Intelligent Systems Design and Applications, 2009. ISDA'09. Ninth International Conference on. IEEE, 2009. p. 13-18.

요약

본 논문에서는 섬모델 유전 알고리즘에서의 적합한 초기 해집단 분할 방식을 제안한다. 섬모델은 유전 알고리즘을 병렬화하여 구현하기에 적합하다고 널리 알려진 방식이다. 섬모델에서는 각각의 프로세서가 각자의 부분해집단을 보유한 상태로 유전 알고리즘 탐색을 진행한다. 기본적인 병렬 유전 알고리즘에서는 모든 해집단이 각각의 "섬"들에 임의로 할당된다. 즉, 초기의 해집단은 임의로 분할되는데 사실 이러한 해집단의 분할은 사용자가 의도적으로 조정할 수 있는 하나의 변수로도 취급할 수 있다. 본 논문에서는 이를 임의로 분할하기보다는, 각 유전자들 간의 거리를 고려하여 초기 해집단을 분할하는 두 가지 방법을 소개한다. 실험결과 제시한 두 가지 방법 중 한가지 방법은 섬모델 유전 알고리즘 전반에 걸쳐 최상해의 품질을 향상시키는 효과를 보였다.

주요어: 유전 알고리즘, 섬모델, 그래프 분할 문제
학번: 2012-20877