Mega Process Genetic Algorithm Using Grid MP

Yoshiko Hanada[1], Tomoyuki Hiroyasu[2], and Mitsunori Miki[2]

[1] Graduate School, Department of Knowledge Engineering and Computer Sciences, Doshisha University, 610-0321 Kyoto, Japan
hanada@mikilab.doshisha.ac.jp
[2] Department of Knowledge Engineering and Computer Sciences, Doshisha University, 610-0321 Kyoto, Japan
{tomo@is, mmiki@mail}.doshisha.ac.jp

Abstract. In this study a new Genetic Algorithm (GA) using Tabu · Local Search mechanism for large-scale computer systems is proposed. We call the GA that uses huge computing resources a Mega Process GA. The GA described in this paper is considered a Mega Process GA. Our proposed method has a GA-specific database that possesses information of space that has been already searched. At the same time, the proposed GA performs a local search for the space that is not searched. Such mechanisms enable us to comprehend the quantitative rate of a searched region during the search. Using this information, the searched space can be expanded linearly as the number of computing resources increase and the exhaustive search is guaranteed under infinite computations.
This study is implemented as a project named Open Bioinformatics (OBI) [3] Grid Crowd Project, which is aimed at developing algorithms and applications on optimization problems under large-scale computer systems. We examine the performance of the proposed method on a distributed computing environment composed of machines belonging to RIKEN Genomic Sciences Center (GSC) [4], which is built up by the commercially available middleware produced by United Devices Inc., named Grid MP[5].

The Concept of Our Proposed Method

Genetic Algorithm (GA) is one of the most effective approximation algorithms for optimization problems. Various types of mechanisms are discussed for improving GAs. Applying a GA to problems has a considerable drawback: GAs require lots of computing costs. One of the solutions to this problem is performing GAs in parallel. Recently, because of the emergence of super PC clusters and Grid computation environments, the number of computational calculation resources

[3] OBI Grid : http://www.obigrid.org/OBIGrid/
[4] RIKEN GSC : http://big.gsc.riken.jp/
[5] United Devices : http://www.ud.com

is getting larger. GA is familiar with parallel processing, consequently the application in large scaled computing has been done. The easiest way for GAs to use many resources is to increase the population number. However, when the population becomes large, the diversity of the solutions also becomes larger. Therefore, the convergence speed becomes slow. Subsequently when GAs use a lot of computer resources, the optimum solution is not derived quickly. Many kinds of GA methods are not optimized to use enormous computing environments effectively since they were developed within limited computing environments. At the same time, for conventional GAs, there is no guarantee of an exhaustive search on all search space although infinite computations are performed. Owing to this, though one applies simple parallelization to those methods, there is no assurance of improvement of their performance in accordance with an increased number of available computing resources.

In this study, a new GA using Tabu · Local Search mechanism for large-scale computer systems is proposed. We call such a GA using huge computing resources a Mega Process GA. The proposed method has a database that possesses information of space that is already searched. At the same time, the proposed GA performs the local search for the space that is not searched to expand the searched space. To obtain optima earlier, our method of searches used mainly schemes of GA; any methods of operators such as a crossover and a mutation or a generation alternation model can be applied. To use idle computing resources of enormous computing environments effectively, a local search is applied. Our proposed method is outlined as follows:

Step 1. Generate N_{pop} individuals randomly, where N_{pop} is the population size. In addition, no individuals are stored in the database.

Step 2. Apply operators such as crossovers, mutations and selections to individuals in a GA population.

Step 3. Store the individual y_{best}, which is the best individual of a GA population, in database; however, y_{best} is not stored when it is included in searched regions, which are indicated by individuals that have been already stored in the database.

Step 4. / Local search/ Expand a searched region indicated by a certain individual stored in the database. When a better individual is found, replace the worst individual of the population of GA with it.

Step 5. Go back to step 2 until some termination conditions, e.g. computing cost reaches a limited amount or the exhaustive search is done, are satisfied.

At the step 3, when there are N_{DB} individuals in the database, replace the individual x_{worst} that has the worst fitness in the database with y_{best} if the fitness of y_{best} is bigger than that of x_{worst}, otherwise y_{best} is not stored, where N_{DB} is the parameter of capacity of the database.

At the phase of local search, the searched space increases linearly as the number of computing resources increase and an exhaustive search is guaranteed under infinite computations.

Computational Experiments

We examine the performance of the proposed method on a distributed computing environment composed of machines belonging to RIKEN Genomic Sciences Center (GSC) , which is built up by the commercially available middleware produced by United Devices Inc., named Grid MP. Grid MP is one of the toolkits of Grid computing and currently used to power several large-scale distributed computing projects. In the Grid MP platform, the under-utilized resources of many computers are aggregated and used as a virtual computer system. Distributed computing environment that is constructed by Grid MP consists of MP Server and Devices. MP Server is the server which carries out users or Devices authentication, monitoring and scheduling jobs, dispatching jobs to Devices, and so on. Devices execute jobs which are assigned by the MP Server. In the computational experiments, MP Server is provided by Intelligent Systems Design Laboratory of Doshisha University. Devices are provided by RIKEN GSC and Intelligent Systems Design Laboratory.

To implement the proposed method on the distributed computing environment, operations of a GA is executed on the Device of Intelligent Systems Design Laboratory, in contract, at the phase of local search, expanding searched region is applied parallelization and executed on Devices provided by RIKEN. The computing environment for experiments is shown in Fig.1.

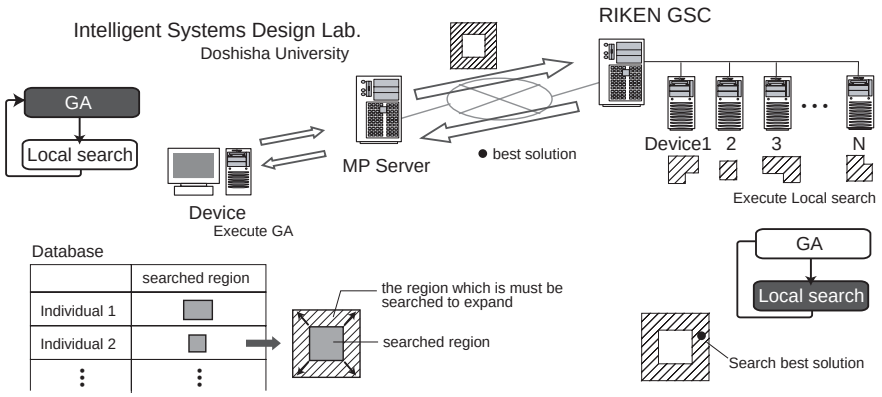

Fig. 1. Computing Environment for Computational Experiments

We show the searching abilities of our proposed method applying it to the test functions of continuous optimization problems. Additionally we examine the performance of the proposed method on the distributed computing environment.