

Discussion of Parallel Model of Multi-Objective Genetic Algorithms on Heterogeneous Computational Resources

Kengo Yoshii, Tomoyuki Hiroyasu, Mitsunori Miki,

Abstract—In this paper, a parallel model of multi-objective genetic algorithm supposing a grid environment is discussed. In this proposed parallel model, we extended master-slave model which has high degree of parallelism, and 2 individuals as a crossover pair are transmitted to each slave process. Then the number of offspring generated by crossover is changed dynamically adapting to the performance of the each calculation resource. This mechanism is effective for heterogeneous computational resources. In addition, total communication cost can be reduced by increasing processing load of the slave processes, and reduction of the overhead time is expected. Moreover, we incorporated the neighborhood crossover, in which the crossover is performed between individuals that are close to each other in the objective space. Therefore, 2 individuals which are close to each other are sent to each slave process. This neighborhood crossover improves the search ability. Computational experiments on heterogeneous computational resources indicated that the proposed model was able to utilize the maximum performance of all calculation resources and reduce the overhead time.

I. INTRODUCTION

THE computational time needed to solve real-world optimization problems is usually large. Therefore, it is very important to reduce the computational time with parallel processing. There have been a number of studies regarding parallel distributed implementation. For example, Deb *et al* [1] proposed and discussed an approach that uses a distributed GA. Although there have been many studies of parallel EMO [1], [2], [3], [4], [5], [6], the degree of parallelization is very small. On the other hand, the recent development of large clusters and grid computing, which have unified the calculation resources online, have made huge resources easily available for such computational tasks. To make use of these huge resources, it is necessary to consider the parallel model where many processes can be performed in parallel.

In this paper, we discuss the parallel model of Multi-objective Genetic Algorithms. In the proposed algorithm, many processes can be performed in parallel and crossover operation and evaluation are performed on each process. Therefore, this algorithm can be applied on huge clusters and the Grid. At the same time, the number of offspring generated by crossover can be changed dynamically in this model. This

mechanism is suitable for hetero calculation environments, such as the Grid computational environment.

In this paper, the performance of the proposed parallel model is compared with original master-slave model and the feasibility of the proposed model is discussed.

II. PARALLEL MULTI-OBJECTIVE GENETIC ALGORITHM

Similarly to single-objective optimization studies, huge computational time is needed to solve real-world problems. Especially, the dimension of the true Pareto-optimal front increases when the number of objectives increases. Therefore, a large population size is required to reach a well-distributed Pareto-optimal front. This results in a huge computational time. One of the solutions to reduce the computational time is to perform EMO in parallel.

Many studies of parallel EMO have been performed, most of which have made use of the master-slave model or island model [1], [2], [3], [4], [5], [6]. In the master-slave model, one master processor runs the GA operations and slave processors are used for evaluation purposes only. In this model, when the number of processors P is used, the ideal acceleration is P . Any algorithm can be applied to this model and the obtained solutions would be equivalent to the solutions obtained with the original algorithms using a single processor. In the island model, a population is divided into a number of subpopulations and a processor is assigned per subpopulation. In this model, different EMOs are run on different processors and some solutions are migrated between processors after every few generations.

A good parallel EMO implementation was presented by Deb [1]. However, in this model, only a small degree of parallelization is assumed because this study uses the island model. The master-slave model is effective to increase the degree of parallelization by at least the population size. On the other hand, many resources are becoming available due to the development of large clusters and grid computing, which have unified online calculation resources. At the same time, the resources on the Grid are not homogeneous but are heterogeneous. Therefore, it is also necessary to consider the parallel model that can be applied to heterogeneous environments, such as grid computing.

The goal of this paper is to discuss the parallel EMO where the master-slave model is extended supposing a heterogeneous grid environment. While proposing the parallel EMO in grid computing, the following two aspects should be kept in mind:

Kengo Yoshii is with the Doshisha University, Graduate School, Department of Knowledge Engineering and Computer Sciences (email: kyoshii@mikilab.doshisha.ac.jp)

Tomoyuki Hiroyasu is with the Doshisha University, Department of Knowledge Engineering and Computer Sciences (email: tomo@is.doshisha.ac.jp)

Mitsunori Miki is with the Doshisha University, Department of Knowledge Engineering and Computer Sciences (email: mmiki@mail.doshisha.ac.jp)

- The resources in the grid environment differ in their performance. Therefore, it is necessary to take into consideration the parallel model corresponding to the heterogeneous grid environment.
- Overheads, such as communication time, must be sufficiently small as compared with evaluation time. As communication time in the grid environment becomes large as compared with the case where parallel processing is performed on a PC cluster, it is necessary to take into consideration the parallel model that can hide the overhead.

As the calculation resources in the grid environment have differences in performance, when all calculation resources have the same number of individuals to be evaluated, the calculation resources with inferior performance would require more time for evaluation, and would act as a bottleneck to progression to the next generation. Therefore, it is necessary to distribute the tasks adapted for the calculation resources. Here, we propose a parallel technique that improves on the master-slave model. In our other paper, we proposed a new crossover, *i.e.*, neighborhood crossover, in which the crossover operation is performed between individuals that are close to each other in objective space [7], [8]. Our neighborhood crossover helps maintain the diversity of the Pareto-optimal solutions. Here, we have combined the neighborhood crossover and conventional multi-objective genetic algorithm. Moreover, the number of offspring generated by the crossover operation is changed dynamically adapting to the performance of the calculation resources. In next section, we describe our proposed model in detail.

III. PARALLEL MULTI-OBJECTIVE GENETIC ALGORITHM IN HETEROGENEOUS COMPUTATIONAL RESOURCES

A. Basic Model

This section presents an explanation of the proposed parallel model of multi-objective GA. The proposal model extends master-slave model, *i.e.*, master process sends two individuals in the current population to each slave process. After each slave process receives two individuals, crossover is performed several times and the number of generated offspring changes adapting the performance of the calculation resource.

The basic model of the proposed parallel GA is illustrated in Fig.1. As the proposed method is based on NSGA-II, the algorithm has the same flow, and other powerful GAs, such as SPEA 2 [9], can also be utilized. The proposed method differs from Multi-Objective GAs such as NSGA-II in two respects: the neighborhood crossover and the number of generated offspring after the crossover. In the neighborhood crossover, two individuals that are close each other are selected as parents. These two individuals are sent to the calculation resource. Then offsprings are generated in each calculation resources. In this step, the number of generated offspring changes with the performance of the calculation resources. Therefore, many offspring are generated on the high performance calculation resource and a smaller number of offspring are generated on the low performance calculation resource. Then, the two best offspring are chosen and returned to the archive. This

mechanism is suitable for hetero calculation environments, and a high degree of parallelization can be achieved.

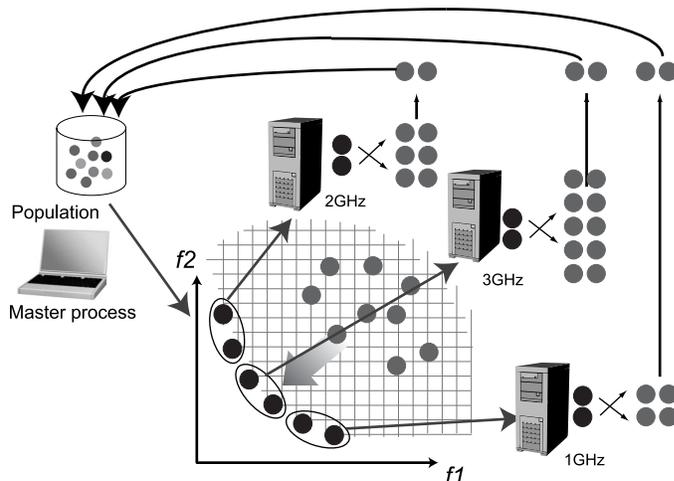


Fig. 1. Basic model of the proposed parallel EMO

B. Neighborhood Crossover

In typical multi-objective genetic algorithms, effective crossover often cannot be performed, as the search directions of each parent individual are different from each other. Therefore, we proposed neighborhood crossover, which generates offspring from two parent individuals neighboring each other in the objective space [7]. By crossing over individuals that are close to each other, offspring can be generated near the parent individuals. Therefore, the search progresses maintaining diversity.

Neighborhood crossover is performed as follows:

- 1) From the best individual for one of the function values, the population is sorted in close order in the objective space.
- 2) Neighborhood shuffle, which changes individuals randomly in some range of population size, is performed for the sorted population to prevent crossing over repeatedly between the same pair of individuals.
- 3) Crossover is performed between two individuals which are side by side.

C. Increasing the Number of Generated Offspring

In the proposed model, the number of generated offspring is changed with the performance of the calculation resources. In conventional GA, two offspring are usually generated after crossover. We increase this number and select the best two offspring according to the following procedure:

- 1) Slave processes receive two individuals from the master process, and creates empty offspring population C .
- 2) Crossover is performed several times depending on the performance of each calculation resources, and the offspring population C is formed.
- 3) The mutation operation is performed against C , and all offspring are then evaluated.

- 4) The Non-Dominated Sort [10] is performed against C to rank all offspring.
- 5) The two best offspring that are rank1 and most excellent about objective function values are returned to the parent population. If the number of rank 1 offspring is 1, the best offspring in rank 2 is then returned to the parent population.
- 6) The master process receives two offspring from all slave processes, and the archive is then updated.

In this algorithm, when the number of offspring generated by crossover increases, the number of evaluations per generation also increases.

IV. COMPUTATIONAL EXPERIMENTS ON HETEROGENEOUS COMPUTATIONAL RESOURCES

In this section, we clarify the effects of the proposed parallel model through computational experiments on heterogeneous computational resources.

A. Test Problem

In this study, the proposed algorithm was applied to a following test function: KUR [11].

KUR in Kursawe can be written as follows:

KUR

$$\begin{cases} \min & f_1 = \sum_{i=1}^n (-10 \exp(-0.2 \sqrt{x_i^2 + x_{i+1}^2})) \\ \min & f_2 = \sum_{i=1}^n (|x_i|^{0.8} + 5 \sin(x_i)^3) \\ \text{s.t.} & x_i \in [-5, 5], i = 1, \dots, n, n = 100 \end{cases} \quad (1)$$

KUR is a problem with an interaction between two continuous variables in $f_1(x)$ and a multi-convex in $f_2(x)$. In this experiment, there were 100 design variables and it was very difficult to find the solutions.

B. Performance Measures

To evaluate the derived Pareto-optimal solutions, mainly two factors should be measured: accuracy and diversion. The derived Pareto-optimal solutions should be close to the real Pareto solutions. At the same time, the derived solutions should not concentrate on a certain point. The derived solutions should also be scattered over a wide area. For this purpose, various performance measures have been proposed to evaluate non-dominated solution sets. In this paper, we use the following performance measures to compare a number of solution sets simultaneously:

- 1) cover rate: I_{cover}
- 2) Spread [12]
- 3) Hypervolume [13]
- 4) Ratio of Non-dominated Individuals:RNI [14]

The cover rate is a method of evaluating whether the solution set is distributed uniformly in the objective space and to calculate the rate of number k_i of the small domain when the domain of Pareto-optimal solutions is divided into K parts. The cover rate calculated for the solution set in N objective functions is as follows. The closer to 1.0, it is estimated that

the solution can be found to all domains. In this experiment, we set the number of divisions K to the population size.

$$I_{cover} = \frac{1}{N} \sum_{i=1}^N \frac{k_i}{K}$$

The Spread measure is a method of evaluating whether the solution set is obtained widely and can be calculated for the solution set as follows:

$$\text{Spread} = \sum_{i=1}^N [\max f_i(x) - \min f_i(x)]$$

The Hypervolume calculates the size of the dominated space by the obtained solutions in the objective space.

The RNI is a method for evaluation by comparing the dominance of two populations obtained by two different algorithms. In RNI, the populations obtained from the two algorithms, S_1 and S_2 are combined to make a union set S_U . Obtain the set of non-dominated individuals S_P from S_U . The number of individuals contained in S_P from each algorithm is used to obtain the ratio, and the value is used as the result of the evaluation. When the value is closer to the maximum of 100%, the algorithm has obtained a better population.

The parameters of GA used in this experiment are shown in TABLE I.

TABLE I
PARAMETERS

Problem	KUR
Population Size	100
Number of Dimensions	100
Chromosome Length	20 × The number of Dimension
Crossover Probability	1.0
Crossover Method	Two Points Crossover
Mutation Probability	1/Chromosome Length

C. Experimental environment and Procedure

In this experiment, the validity of the proposed parallel model was verified using one master process and a total of 50 slave processes using a 4-PC Cluster comprised of PCs that differed in performance. Calculation resources are shown in TABLE II. We used the Grid RPC Ninf-G (version 2.4) [15] for submitting jobs to each PC Cluster, and Open PBS(version 1.2) was used for scheduling jobs in each PC cluster. Ninf-G is a reference implementation of the Grid RPC system using the Globus Toolkit [16].

The flow of execution is shown in Fig.2. First, the master process generates an initial population, and perform Neighborhood Sort which reorders in close order in objective space (and also perform neighborhood shuffle). Then, the master process submits two adjacent individuals to master nodes of each PC Cluster using Ninf-G. At this time, the data transmitted are the chromosome information of two individuals. After the master nodes of each PC Cluster receive these data, they begin scheduling the jobs and distribute them to slave processes. Each slave process repeats the operations of crossover, mutation, and evaluation during a given time, and perform non-dominated sort to choose the best two offspring that should be returned to the master process. The data transmitted from slave processes to the master process are chromosome information and objective function values of the best two offspring.

TABLE II
CALCULATION RESOURCES

	number of CPU	CPU	Memory	OS
Master process	1	Athlon64 3200+	1GB	Fedora Core 4
PC Cluster A	10	Pentium4 2.8GHz	1GB	Debian 3.1
PC Cluster B	15	Xeon 2.4GHz	1GB	Debian 3.1
PC Cluster C	15	PentiumIII 1GHz	512MB	Debian 3.1
PC Cluster D	10	PentiumIII 600MHz	256MB	Debian 3.1

These procedures are carried out on all slave processes with synchronous communications in one generation.

In this experiment, each calculation resource increased the number of offspring adapted for performance of the calculation resources by repeating crossover, mutation, and evaluation during a given time. Thereby, all the calculation resources can terminate processing almost simultaneously after a given time, and the delay by the calculation resource with low performance can be prevented. It is necessary to set up a fixed time based on the calculation load of the object problem on the performance of calculation resources.

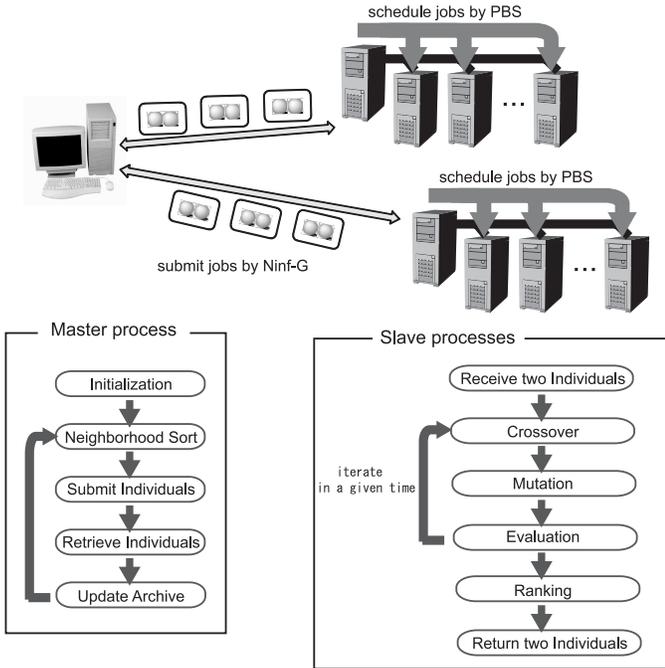


Fig. 2. Execution Flow of Proposed Algorithm on heterogeneous computational resources

The target function is KUR whose calculation load is increased by executing useless calculations for assuming a real problem. The evaluation time of one individual by the calculation resources of each PC cluster for this problem was 5.82 seconds in PC Cluster A, 8.62 seconds in B, 10.17 seconds in C, and 17.06 seconds in D. Then, we set up a fixed time of 1 minute so that the calculation resources of PC Cluster D, which has the poorest performance, could generate at least 2 offspring. Therefore each slave process repeats the operations of crossover, mutation, and evaluation during 1 minute.

We compared our proposed parallel model with the original NSGA-II algorithm, which is parallelized with the original

master-slave model under the same environment and conditions, and set 2 hours as the termination condition.

D. Experimental Results

Average results over three runs for the KUR test problem are summarized in Fig. 3. Figs. 3 a) show the cover rate, b) shows Spread, c) shows Hypervolume, and (d) shows RNI against original NSGA-II with master-slave model. This figure shows that the effectiveness of our proposed algorithm increased sharply, especially in diversity and spread of Pareto-optimal solutions. Therefore, our proposed parallel model is effective on heterogeneous computational resources. Fig. 4 shows the distribution graph of all three runs obtained by our proposed model and the original master-slave model. Fig. 4 shows that the solution set obtained by the proposed algorithm is good in terms of both diversity and spread.

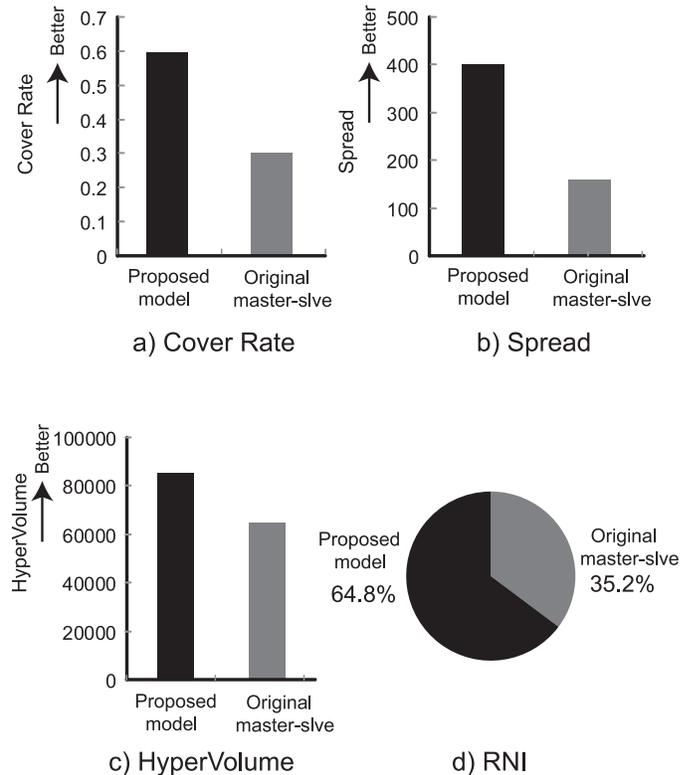
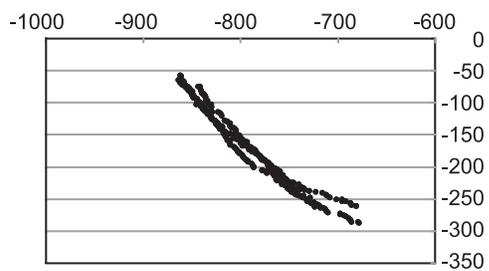


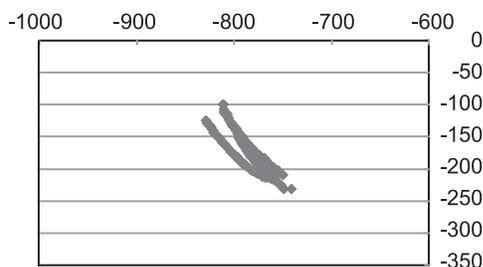
Fig. 3. Results of Icover, Spread, Hypervolume, and RNI on proposed parallel model and original master-slave model in KUR problem

E. Discussions

In the proposed model, all calculation resources can terminate processing almost simultaneously after a given time so



a) Proposed model



b) Original master-slave model

Fig. 4. Pareto-optimal solutions obtained by proposed parallel model and original master slave model in KUR problem

that the idle time of all calculation resources is maintainable to the minimum. It is also possible to reduce overheads time such as communication time or scheduling jobs by increasing the process in remote slave processes. To confirm these, we indicate average CPU usage rate of a process in each PC cluster in TABLE III, and the total overhead time in TABLE IV. TABLE III indicates that the idle time become longer on the high performance calculation resources, especially on PC cluster A, in original master-slave model. However, all processes have loads uniformly in the proposed model. We also confirmed that the influence of overhead time is made small in the proposed parallel model from TABLE IV.

TABLE III

AVERAGE CPU USAGE RATE OF A PROCESS IN EACH PC CLUSTER

	A	B	C	D
Proposed parallel model	88%	91%	95%	91%
Original master-slave model	18%	44%	53%	89%

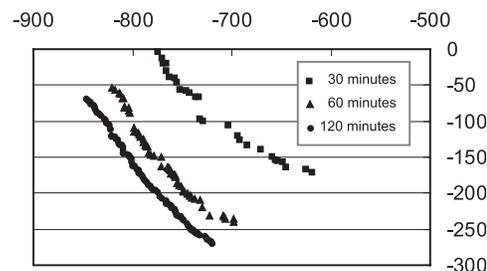
TABLE IV

OVERHEAD TIME AND RATE IN TOTAL EXECUTION TIME

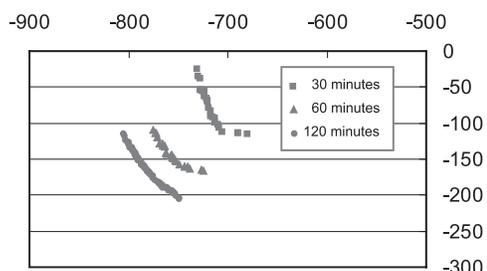
	Time
Proposed parallel model	4m5s
Original master-slave model	4m38s

Finally, Fig. 5 shows the search history of the proposed model and the original master-slave model that plots solution sets at the time of 30 minutes, 60 minutes, and 120-minutes progress. In comparison with the original NSGA-II search process, wide-ranging non-dominated solutions with great diversity are obtained from the early stages of the search.

These observations indicate that it is possible to conduct a search while maintaining the diversity of the population, and to obtain a wide range of Pareto optimum solutions using the proposed model.



a) Proposed model



a) Original master-slave model

Fig. 5. Search history of the proposed model and the original master-slave model

V. CONCLUSION

In this paper, we proposed a new parallel model of EMO supposing a hetero-grid environment and examined the accuracy of the algorithm through computational experiments. We combined our neighborhood crossover with a multi-objective genetic algorithm. We also considered increasing the number of offspring dynamically according to the performance of the available calculation resources. We investigated the validity of our method using a master-slave model on heterogeneous calculation resources. Computational experiments on a numerical test problem indicated that the proposed parallel algorithm was able to utilize the maximum performance of all calculation resources and reduce the overhead time, and also has high search ability of pareto-optimal solutions.

REFERENCES

- [1] Kalyanmoy Deb, Pawan Zope, and Abhishek Jain. Distributed computing of pareto-optimal solutions with evolutionary algorithms. In *EMO*, pages 534–549, 2003.
- [2] David A. van Veldhuizen, Jesse B. Zydallis, and Gary B. Lamont. Considerations in engineering parallel multiobjective evolutionary algorithms. *IEEE Trans. Evolutionary Computation*, 7(2):144–173, 2003.
- [3] F. Streichert, H. Ulmer, and A. Zell. Parallelization of multi-objective evolutionary algorithms using clustering algorithms. In Carlos A. Coello Coello, Arturo Hernandez Aguirre, and Eckart Zitzler, editors, *Conference on Evolutionary Multi-Criterion Optimization (EMO 2005)*, volume 3410 of *LNCS*, pages 92–107, Guanajuato, Mexico, 9–11 March 2005.

- [4] F. de Toro Negro, J. Ortega, E. Ros, S. Mota, B. Paechter, and J. M. Martín. Psfga: parallel processing and evolutionary computation for multiobjective optimisation. *Parallel Comput.*, 30(5-6):721–739, 2004.
- [5] Gary B. Lamont Carlos A. Coello and David A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, Norwell, MA, USA, 2002.
- [6] Jürgen Branke, Hartmut Schmeck, Kalyanmoy Deb, and M. Reddy. Parallelizing multi-objective evolutionary algorithms: cone separation. In *Congress on Evolutionary Computation*, volume 2, pages 1952–1957. IEEE, JUN 2004.
- [7] T. Hiroyasu S. Watanabe and M. Miki. Neighborhood cultivation genetic algorithm for multi-objective optimization problems. *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution And Learning (SEAL-2002)*, pages 198–202, 2002.
- [8] T. Hiroyasu M. Kim and M. Miki. Spea2+: Improving the performance of the strength pareto evolutionary algorithm2. *Parallel Problem Solving from Nature - PPSN VIII*, pages 742–751, 2004.
- [9] M. Laumanns E. Zitzler and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Gloriastrasse 35, CH-8092 Zurich, Switzerland, 2001.
- [10] A. Pratab K. Deb, S. Agrawal and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In Marc Schoenauer, Kalyanmoy Deb, Günter Rudolph, Xin Yao, Evelyne Lutton, J. J. Merelo, and Hans-Paul Schwefel, editors, *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pages 849–858, Paris, France, 2000. Springer. Lecture Notes in Computer Science No. 1917.
- [11] F. Kursawe. A Variant of Evolution Strategies for Vector Optimization. In H. P. Schwefel and R. Männer, editors, *Parallel Problem Solving from Nature. 1st Workshop, PPSN I*, volume 496 of *Lecture Notes in Computer Science*, pages 193–197, Berlin, Germany, oct 1991. Springer-Verlag.
- [12] Eckart Zitzler and Lothar Thiele. An evolutionary algorithm for multiobjective optimization: The strength pareto approach. Technical Report 43, Gloriastrasse 35, CH-8092 Zurich, Switzerland, 1998.
- [13] K. Deb E. Zitzler and Lothar Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195, 2000.
- [14] T. H. Lee K. C. Tan and E. F. Khor. Incrementing Multi-objective Evolutionary Algorithms: Performance Studies and Comparisons. *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 111–125, 2001.
- [15] "Yoshio Tanaka, Hidemoto Nakada, Satoshi Sekiguchi, Toyotaro Suzumura, and Satoshi Matsuoka". "Ninf-G: A Reference Implementation of RPC-based Programming Middleware for Grid Computing". In *Journal of Grid Computing, Vol. 1, No. 1, pp. 41-51, Kluwer Academic Publishers, June, 2003.*, 2003.
- [16] The Globus Alliance. . <http://www.globus.org/>.