Parallelizing Multi-objective Evolutionary Genetic Algorithms

G. N. Shinde *Member IAENG*, Sudhir B. Jagtap and Subhendu Kumar Pani

***Abstract*: In this paper a hybrid parallel multi-objective genetic algorithm is proposed for solving 0/1 knapsack problem. Multi-objective problems with non-convex and discrete Pareto front can take enormous computation time to converge to the true Pareto front. Hence, the classical multi-objective genetic algorithms (MOGAs) (i.e., non- Parallel MOGAs) may fail to solve such intractable problem in a reasonable amount of time. The proposed hybrid model will combine the best attribute of island and Jakobovic master slave models. We conduct an extensive experimental study in a multi-core system by varying the different size of processors and the result is compared with basic parallel model i.e., master-slave model which is used to parallelize NSGA-II. The experimental results confirm that the hybrid model is showing a clear edge over master-slave model in terms of processing time and approximation to the true Pareto front.**

Index Terms: Multi-Objective Genetic Algorithm, Parallel Processing Techniques, NSGA-II, 0/1 Knapsack Problem, Trigger Model, Cone Separation Model, Island Model

I. Introduction

Many of the real-world engineering optimization problems are multi-objective in nature, since they normally have several non-commensurable objectives that must be satisfied at the same time. These problems are known as multi-objective optimization problems (MOP) [1] in contrast with single objective optimization problems (SOP). The notion of optimum has to be redefined in this context and instead of aiming to find a single optimal solution; a procedure for solving MOP should determine a set of good compromises or trade-off solutions, generally known as Pareto optimal solutions, where the decision maker will get enough flexibility to choose a particular solution. These solutions are optimal in the wider sense that no other solution in the search space is superior when all objectives are considered. Pareto optimal solutions form the Pareto front in a *k*-dimensional objective space, where *k* is the number of the objectives in the optimization problem. Evolutionary algorithms (EAs) have the potential for finding multiple Pareto optimal solutions in a single run and have been widely used in this area.

Manuscript received February 9, 2011; revised April 18, 2011.
Dr. Ganeshchandra N. Shinde with the Indira Ghandhi (SR) College, Nanded-431603, Maharashtra (India). He is working as a Principal & Professor in the department of Electronics. (Corresponding author Phone No. : 91-2462-262620; fax: 91-2462-227426; E-mail: shindegn@yahoo.co.in).
Sudhir B. Jagtap is working as a Assistant Professor in the department of Computer Science, Swami Vivekanand Mahavidyalaya, Udgir, India (E-mail:sudhir.jagtap7@gmail.com)
Dr. Subhendu Kumar Pani is working as Associate Professor in the department of Computer Science RCMA, BPUT Bhubaneswar-751004, Orissa, India (E-mail:Subhendu_pani@rediffmail.com)

One of the major drawbacks of multi-objective genetic algorithm (MOGA) is that a relatively large number of solutions have to be evaluated before generating good results which is true for multi-objective optimization problems in higher domain. As a result, a large population size is required for this. The above mentioned drawback can be compensated by parallelizing the MOGA [2].

With regards to parallelization, the main difference between single and multi-objective evolutionary algorithms seems to be that in the multi-objective case, a set of solutions is sought rather than a single optimum. This opens the possibility of having the different processors search for different solutions, rather than to follow an identical goal. The hope is that such a divide-and-conquer principle is more efficient if all processors work on the whole problem [3-5].

There are different models like Jakobovic master slave, trigger, island and cone-separation models proposed by different scientist [2, 3, and 6] to implement parallel MOGA (PMOGA), e.g., NSGA II. In this paper, we have proposed a hybrid model to implemented the MOGA for solving the real world multi-objective 0/1- knapsack problem and studied their characteristics on the basis of convergence quality and time factor as parameter.

The 0/1 knapsack problem is a widely studied problem due its NP-hard nature and practical importance. Generally, a 0/1 knapsack problem consists of a set of items, weight and profit associated with each item, and an upper bound for the capacity of the knapsack. The task is to find a subset of items which maximizes the total of profits in the subset, yet all the selected items fit into the knapsack, i.e. the total weight does not exceed the given capacity [7]. This single objective problem can be extended directly to a multi-objective case by allowing an arbitrary number of the knapsacks. Formally, the multi-objective 0/1 knapsack problem is defined through (1) and (2).

Given a set of m items and a set of n knapsacks with $p_{i,j}$ = profit of item j according to knapsack i, $w_{i,j}$ = weight of item j according to knapsack i, c_i = capacity of knapsack i.

Find a vector $x = (x_1, x_2, \ldots, x_m) \in \{0, 1\}^m$ such that, $\forall\, i \in \{1, 2, \ldots, n\}$: $\sum_{j=1}^{m} w_{ij} . x_j \le c_i$ (1)

and for which $f(x) = (f_1(x), f_2(x), \ldots, f_n(x))$ is maximum, where

$f_i(x) = p_{ij} x_j$ (2)

and $x_j = 1$ iff item j is selected.

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

In order to obtain reliable and sound result, three different test problems are investigated where both the number of knapsacks (i.e., number of objectives) and the number of items are varied. Two knapsacks (i.e., two objectives) are taken under consideration in combination with 250, items. Following suggestions in [7], profits and weights are chosen, where $p_{i,j}$ and $w_{i,j}$ are random integers in the interval [10, 100]. Also as reported in [7], about half of the items are expected to be in the optimal solutions when this type of knapsack capacity is used. Thus, the knapsack capacities are normally set to half the total weight according to the corresponding knapsack as indicated in equation (3)

$$c_i = 0.5 \sum_{j=1}^{m} w_{ij} \quad (3)$$

The paper is structured as follows: Section 2 deals with review on PMOGA and its models. Section 3 presents the proposed hybrid model. The problem is evaluated and studied empirically by using the above model in Section 4. Finally conclusion is drawn in Section 5.

II. Review of Parallel MOGA Models

Evolutionary algorithms are very suitable for parallelization, as crossover, mutation, and in particular the time-consuming fitness evaluation can be performed independently on different individual.

The approaches to parallelize multi-objective GAs can be grouped into three categories:

1) Jakobovic Master-Slave: Jakobovic model is one type of master slave models [1]. In this model the master only triggers the slaves. Each slave and the master (also perform as a slave instead of being idle), then creates random initial population, evaluates created individuals, perform whole evaluation process and then return the final results to the master. This eliminates the time required to generate each population at the server for slaves, the communication overhead and allows for an exhaustive search of the solution space by the slaves through random explorations [1].
2) lsland Model: In this model, every processor runs an independent EA, using a separate sub-population. The processors cooperate by regularly exchanging *migrants* (good individuals). The island model is particularly suitable for computer clusters, as communication is limited [3]. This model scales very well. Low communication overhead as less population migration between the islands. This model is robust to failure as it is willing to lose small populations. This model has higher diversity as it has the working principle of isolated population with migration.
3) Cone Separation model: It was suggested by Branke et al. in 2004 [2]. The basic idea of this model is to divide the search space in several regions and assign it to the different processor. The partitioning of the search space is adopted at regular intervals by normalizing the fitness values in such a way that the whole non-dominated front is within the unit square (hypercube, for more than 2 dimensions). After the normalization, the fitness space is partitioned into equal cones. Each processor is then assigned one part. Therefore, whenever the constraints are adapted, individuals violating the constraints are migrated into the population where they do not violate the constraints. Thereby, individuals are just added to the receiving population, without explicitly deleting others. Overall, the approach is integrated into NSGA-II [4, 6].

III. Hybrid Model

Master-slave and multiple-deme parallel GAs can quickly reach to good solutions when they are designed properly, but even better quality result can be found by combining any two basic form of parallel GA, in to a hierarchical algorithm. The basic idea is to design a multiple-deme algorithm where the demes themselves are some form of parallel MOGAs. In our proposed model we have integrated two basic models Island i.e., multiple deme model at the higher level with Jakobovic model at the lower level. In this, population of each deme is taken care by Jakobovic model. Each deme is connected through a ring topology, and the migration takes place between each deme at a regular interval. The model is pictorially represented in Figure1.The working principle of the model can be explained in following steps:

I. Each Island generats its population.
II. The Jakobovic model distributes the population equally among the slaves including the master.
III. All master and slave processor starts running independent MOGA in parallel.
IV. Master collects the best population from the slaves.
V. Best individuals of each island are migrated to the neighboring island.
VI. Neighboring island receives the individuals and distributes equally to all the processor inside it.
VII. 3rd step onward is repeated till termination condition is achieved.

IV. Experimenrtal Study

In this section we present the experimental setup and discuss the empirical results of each experiment in detail.

A. Experimental setup

The algorithm is implemented using C language on a multi-core system core i7 with 8 cores, each of 1.6 GHz, 4GB RAM, under Linux OS. The communication between the processors has been supported by the free available MPICH (Message Passing Interface) library. The parameters of the parallel MOGA are shown in Table 1, in which the first row explains the population size per deme, second, third, fourth, and fifth row represents the crossover probability, mutation probability, migration rate and termination condition respectively, assumed for the experiment.

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

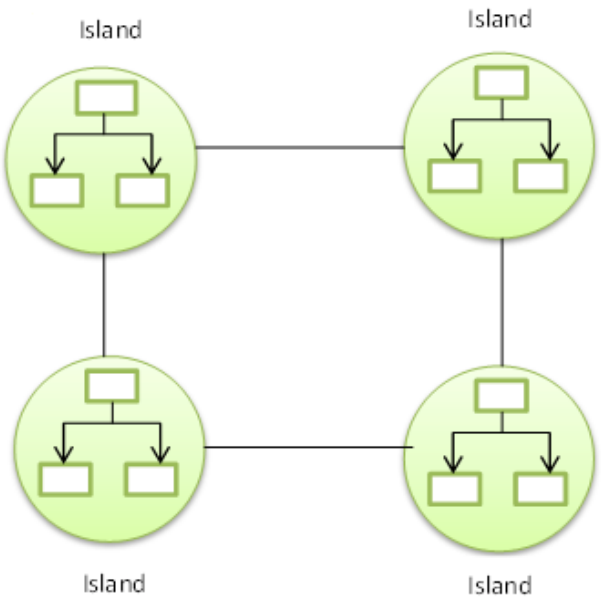

Figure1: A hybrid model where at the upper level an Island model and lower level a Jakobovic model.

B. Experimental results

The experiment is conducted with the very well-known multi-objective 0/1- knapsack problem. The problem is solved by proposed hybrid model and the result is compared with the master-slave model. Figure 2 explains the convergence result with different models with two and four processors respectively. From Figure 2, it can be seen that in a master slave model, the convergence quality detoriates as the number of processor increases. The convergence of two processors is better than four processors. In hybrid model it is clearly observed that the convergence quality increases as we go on increasing the demes. Over all it is observed that the quality of result we get is better in the proposed hybrid model than the master-slave model.

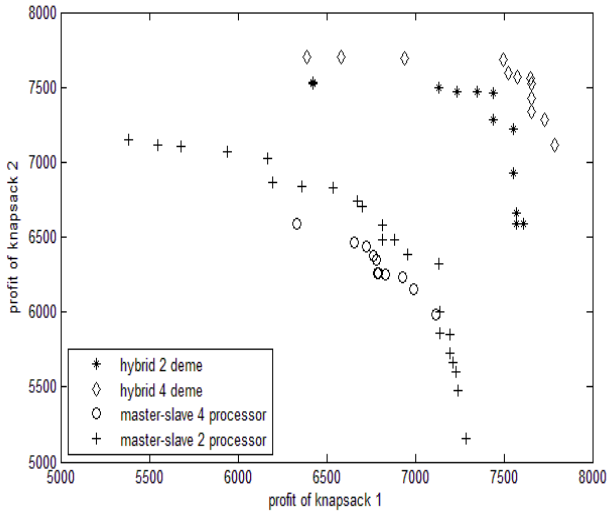

Figure 2: Convergence result of the hybrid model with two and four demes and Master-slave model using two and four processors.

Table 1: Parameter setting

Population size / Deme	200
Crossover probability	0.8
Mutation probability	0.016/bit
Migration rate	After every 10 generation
Termination condition	Average knapsack profit greater than 10000 or the movement of non-dominated solution remains stagnant for 20 generation

To observe the behavior of the proposed model we varied the number of processors in each deme, keeping the number of deme constant. It is observed that, if we increase the number of processors inside the deme, the quality of the result detoriates, but however the time taken for each generation is less. The speedup graph is shown in Figure 3 which is drawn by varying the number of processor in each deme, keeping number of deme as two.

The main challenges associated with parallelism are:

- Minimizing Input/output.
- Minimizing synchronization and communication.
- Effective load balancing.
- Deciding the best search procedure use.
- Maximizing /avoiding duplication of work.

The speed and efficiency of parallel formulations are as below. In a parallel MOGA, the main issues taken into account are:

- Load balancing
- Minimizing communication.
- Overlapping communication and computation.

Speed Up:

A program is basically associated with some serial instructions and some parallel instructions. The serial instructions cannot be parallelized because there may be presence of dependency between the instructions.

Let us assume that, serial fraction of instructions: f_s, parallel fraction of instructions: f_p=1-f_s, then speedup is defined as

$S=T_s/T_p = 1/(f_s+(f_p/p))$ (i.e., p: number of processors).

Efficiency:

The efficiency in terms of f_p and f_s is computed as follows:

$$E = \frac{S}{p} = \frac{1}{(pfs + f_p)}.$$

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

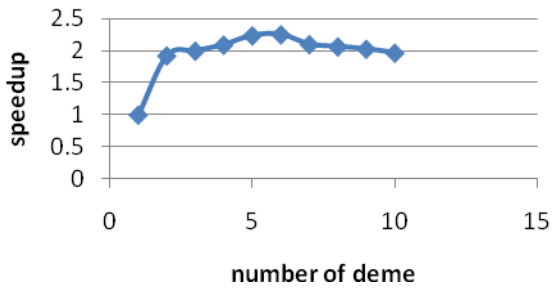

Figure 3: Speedup analysis of hybrid model.

In general, we can say that, the total overhead need in an increasing function of P, at least linearly when $f_s > 0$. In case of parallel processing, we are using more than one numbers of processor, so the total load balancing is required between the processors. There are two types of load balancing static and dynamic load balancing seeks to address these issues by balancing the load and reassigning the loads to the lighter ones. The development of distributing search space is a challenging and critical task. Since it requires knowledge at all the individuals stored at different locations and the ability to combine partial results from individuals search space into a single result.

V. Conclusion

Parallelizing multi-objective evolutionary algorithms is an important issue as it is associated with large computation time with multiple solutions. In this paper, we proposed a new hybrid model to parallelize multi-objective genetic algorithms and implemented it on 0/1 knapsack problem. Further, the result was also compared with the basic master-slave model. Again by discussing the convergence parameter, it is concluded that the hybrid model gives better result in comparison with master-slave model irrespective of any number of processor.

References

[1] T. Al-Somani, & Kalim Qureshi, "*R*eliability Optimization Using Genetics Algorithms", M. Sc. Thesis, Saudi Arabia: King Abdul-Aziz University, 2000.

[2] J. Branke, H. Schmeck, K. Deb, & M. S Reddy, "Parallelizing Multi-objective Evolutionary Algorithms:Cone Separation". *Congress on evolutionary Computation, Vol.2*, pp. 1952– 1957, 2004.

[3] E. Cantu-Paz, "A Survey of Parallel Genetic Algorithms", Calculateurs Paralleles, Vol.10, No.(2), pp. 141-171, 1998.

[4] K. Deb, A. Pratap, S. Agrawal, & T. Meyarivan, "A Fast and Elitist Multi-objective Genetic Algorithm: NSGA-II", IEEE Transactions on Evolutionary Computation, Vol. 6, No.2, pp. 182-197, April 2002.

[5] C. Grosan, "How to compare the multi-objective evolutionary algorithms performances?" Zilele Academice, Clujene, 2003.

[6] F. Streichert, H. Ulmer, & A. Zell, "Parallelization of Multi-objective Evolutionary Algorithms Using Clustering Algorithms", In C. A. Coello Coello, A. Hernández Aguirre, & E. Zitzler, *Evolutionary Multi-Criterion Optimization* ,Vol. 3410, pp. 92-107, Berlin / Heidelberg: Springer, 2005 .

[7] E. Zitzler, K. Deb, & L. Thiele, "Comparison of multi-objective evolutionary algorithms: Empirical results", INSTITUTION Swiss Federal Institute of Technology (ETH), Zurich, 1999.

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)