Parallel Hyperheuristic

A Self-Adaptive Island-Based Model for Multi-Objective Optimization

Coromoto León
Dpto. Estadistica, I. O. y Computación
Universidad de La Laguna
38271, Tenerife, Spain
cleon@ull.es

Gara Miranda
Dpto. Estadistica, I. O. y Computación
Universidad de La Laguna
38271, Tenerife, Spain
gmiranda@ull.es

Carlos Segura
Dpto. Estadistica, I. O. y Computación
Universidad de La Laguna
38271, Tenerife, Spain
csegura@ull.es

ABSTRACT

This work presents a new parallel model for the solution of multi-objective optimization problems. The model combines a parallel island-based scheme with a hyperheuristic approach in order to raise the level of generality at which most current evolutionary algorithms operate. This way, a wider range of problems can be tackled since the strengths of one algorithm can compensate for the weaknesses of another. Computational results demonstrate that the model grants more computational resources to those algorithms that show a more promising behaviour.

Categories and Subject Descriptors

I.2.8 [**Computing Methodologies**]: Artificial Inteligence—*Problem Solving, Control Methods and Search* Heuristic Methods; D.1.3 [**Software Engineering**]: Programming Techniques—*Concurrent Programming* Parallel Programming

General Terms

Algorithms

Keywords

Multi-objective Optimization, Evolutionary Algorithms, Island-Based Models, Hyperheuristics, Hybrid Algorithms

1. INTRODUCTION

Multi-objective optimization evolutionary algorithms (MOEAs) have shown great promise for calculating solutions to large and difficult *multi-objective optimization problems* (MOPs) [1]. Several studies have been performed in order to reduce the resource expenditure when using MOEAs. These studies naturally lead to considering the MOEAs parallelization [3]. Among the existing parallel MOEAs (pMOEAs), standard island-based models appear to be the most efficient method. If there exists a MOEA that clearly outperforms the other ones in solving one type of problem, the homogeneous island-based model using such MOEA allows to obtain good quality solutions. However, it is difficult to know a priori which MOEA is the most appropriate to solve a problem. By contrast, heterogeneous models allows to execute different MOEAs and/or parameters on each processor at the same time, but if some of the MOEAs are not suitable to optimize the problem, a waste of resources is done.

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.
ACM 978-1-60558-130-9/08/07.

2. SELF-ADAPTIVE PARALLEL MODEL

The proposed pMOEA model breaks from the island-based model adding a self-adaptive property behaviour to it. The self-adaptive property allows, by applying a hyperheuristic, to change in an automatic and dynamic way the MOEAs and/or parameters that are used in the islands along the pMOEA run. The architecture of the new hybrid model is similar to the island-based model, i.e. it is constituted by a set of *slave islands* that evolve in isolation applying a certain evolutionary algorithm to a given population (see Algorithm 1). The number of islands and the different MOEAs to execute over the local populations are defined by the user. Also, as in the island-based model, a tunable migration scheme allows the exchange of solutions between neighbour islands. However, a new special island is introduced into the scheme. That island is called the *master island* (see Algorithm 2) and it is in charge of maintaining the global solution achieved by the pMOEA and selecting the MOEA configurations that are executed on the slave islands. The *global solution* is obtained by joining the local solutions achieved by each one of the slave islands.

In the proposed model, besides the global stop criterion, local stop criterions are defined for the execution of the MOEAs on the islands. When a local stop criterion is reached, the island execution is stopped and the local results are sent to the master. The master scores, by means of the contribution metric, the configurations defined by the user according to their obtained results. Based on such quality indicator, the hyperheuristic is applied and the master selects the configuration that will continue executing on the idle island.

3. EXPERIMENTAL EVALUATION

The two-objective ZDT [4] test problems have been chosen to test the behaviour of the designed model. The MOEAs that have been implemented to test the model are: SPEA [6], SPEA2 [5] and NSGA-II [2].

Algorithm 1 Slave Islands Pseudocode

```
1: configureMigration()
2: lastMOEA ← NULL
3: while (not globalStopCriterion()) do
4:    newMOEA ← receiveMOEAConfiguration()
5:    if (newMOEA != lastMOEA) then
6:       lastMOEA ← initMOEA(newMOEA)
7:       receiveInitialPopulation()
8:    end if
9:    while (not localStopCriterion()) do
10:      runGeneration()
11:      migrate()
12:   end while
13:   sendLocalSolution()
14: end while
```

Algorithm 2 Master Island Pseudocode

```
1: globalSolution ← ∅
2: initAdaptiveModel()
3: assignInitMOEAConfigsToIslands()
4: while (not globalStopCriterion()) do
5:    island ← checkForIdleIsland()
6:    if (island != NULL) then
7:       islandLocalSol ← receiveIslandSolution(island)
8:       globalSolution ← globalSolution ∪ islandLocalSol
9:       scores ← scoreMOEAConfigs()
10:      nextMOEAConf ← selectMOEAConfig(scores)
11:      if (nextMOEAConf != currentMOEAConf(island)) then
12:         assignMOEAConfig(nextMOEAConf, island)
13:         assignInitialPop(subset(globalSolution), island)
14:      else
15:         resumeExecution(island)
16:      end if
17:   end if
18: end while
```

The experiment compares the proposed model with some standard pMOEAs. For each implemented MOEA a homogeneous scheme is considered: "*homo*-SPEA", "*homo*-SPEA2" and "*homo*-NSGA-II". A heterogeneous scheme constituted by the three implemented MOEAs is also considered: "*heterogeneous*". Two self-adaptive executions are tested: "*3-adaptive*" and "*6-adaptive*". The first one uses the same properties as previous considered models: it is constituted by three slave islands, subpopulations on the islands are fixed to 45 individuals, and the archive size for the SPEA and SPEA2 configurations is fixed to 45. Instead, the 6-adaptive version uses 6 different MOEAs configurations: the same three as the 3-adaptive besides three new ones with subpopulations and archive sizes of 30 elements. For the mutation, the uniform mutation operator has been selected. The uniform crossover operator has been used to perform the crossover. The mutation and crossover rates has been fixed to 0.01 and 1, respectively. In all cases, the same migration scheme is specified. It consists in an unrestricted topology where the migration is performed from a slave to a randomly selected partner. The migration probability has been fixed to 0.05 and the number of migrated individuals limited to 4 each time. The *global stop criterion* for every execution is 90000 evaluations. The *local stop criterion* in 3-adaptive is fixed to 900 evaluations. In all cases, the final solution is limited to 100 elements.

Table 1 presents, for each model and test problem, the average number of evaluations needed to reach the 75%, 95% and 99% of the best hypervolume [7] achieved. The results achieved by the 6-adaptive configuration outperforms the ones of the 3-adaptive configuration. Results obtained by the new scheme improve the ones reached by the heterogeneous model, getting close to the best homogeneous model.

HV	*3-adap*	*6-adap*	*homo* SPEA	*homo* SPEA2	*homo* NSGA-II	*hete*
ZDT1						
75%	8173	7840	10617	8833	7910	8339
95%	27549	27142	29685	27423	25282	27555
99%	58451	57110	61327	58869	55779	59436
ZDT2						
75%	21928	20391	28653	20853	20108	22467
95%	52623	49557	59722	50335	50295	53301
99%	75928	72668	79110	74077	79110	76714
ZDT3						
75%	5734	5419	8121	6422	6153	6033
95%	22049	21360	25223	22203	21304	22414
99%	52800	51403	56039	50914	52279	53864
ZDT4						
75%	40392	37496	43580	38628	39371	40468
95%	62763	63344	67778	61064	64810	67470
99%	67758	66710	69893	66522	69558	72835

Table 1: Hypervolume for ZDT Problems

Thanks to the adaptive behaviour of the model, more computational resources are given to the most promising configurations, and thus, high-quality solutions are obtained without forcing the user to have a prior knowledge about each MOEA behaviour when applied to the considered problem.

4. ACKNOWLEDGMENTS

This work has been supported by the EC (FEDER) and the Spanish Ministry of Education and Science inside the 'Plan Nacional de I+D+i' with contract number (TIN2005-08818-C04-04). The work of Gara Miranda has been developed under grant FPU-AP2004-2290.

5. REFERENCES

[1] C. A. Coello, G. B. Lamont, and D. A. V. Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Genetic and Evolutionary Computation. 2007.

[2] K. Deb, S. Agrawal, A. Pratab, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In *VI Conference on Parallel Problem Solving from Nature*, volume 1917 of *LNCS*, pages 849–858. Springer, 2000.

[3] D. A. V. Veldhuizen, J. B. Zydallis, and G. B. Lamont. Considerations in engineering parallel multiobjective evolutionary algorithms. *IEEE Trans. Evolutionary Computation*, 7(2):144–173, 2003.

[4] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8(2):173–195, 2000.

[5] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. *Evolutionary Methods for Design, Optimization and Control*, pages 19–26, 2002.

[6] E. Zitzler and L. Thiele. An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach. Technical Report 43, Computer Engineering and Networks Laboratory, Zurich, Switzerland, 1998.

[7] E. Zitzler and L. Thiele. Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study. In *Parallel Problem Solving from Nature (PPSN-V)*, volume 1498, pages 292–301. Springer, 1998.