# Parallelization of Multi-objective Evolutionary Algorithms Using Clustering Algorithms

Felix Streichert, Holger Ulmer, and Andreas Zell

Center for Bioinformatics Tübingen (ZBIT), University of Tübingen,
Sand 1, 72076 Tübingen, Germany
streiche@informatik.uni-tuebingen.de

**Abstract.** While single-objective Evolutionary Algorithms (EAs) parallelization schemes are both well established and easy to implement, this is not the case for Multi-Objective Evolutionary Algorithms (MOEAs). Nevertheless, the need for parallelizing MOEAs arises in many real-world applications, where fitness evaluations and the optimization process can be very time consuming. In this paper, we test the 'divide and conquer' approach to parallelize MOEAs, aimed at improving the speed of convergence beyond a parallel island MOEA with migration. We also suggest a clustering based parallelization scheme for MOEAs and compare it to several alternative MOEA parallelization schemes on multiple standard multi-objective test functions.

## 1 Introduction

For some time, Evolutionary Algorithms (EAs) have developed into a feasible optimization approach for many industrial applications. Especially, Multi-Objective EAs (MOEAs) receive a lot of attention, because many practical optimization problems are often multi-objective. Unfortunately, EA approaches often require huge amounts of fitness evaluations to solve a given optimization problem. This holds true especially in case of multi-objective optimization problems, where the search space may be of same size, but a significantly larger portion of it needs to be explored to obtain the whole Pareto front.

There are two kinds of limitations on fitness evaluations that make EAs and MOEAs infeasible for many real world applications. First, a fitness evaluation requires real-world experiments, which can be both costly and time consuming. Or second, the fitness evaluation is computationally too expensive to allow optimization through EAs and MOEAs in reasonable time.

The first kind of limitation can only be solved by making EAs and MOEAs more efficient by means of improved evolutionary operators, supporting the EA with problem specific knowledge or local search heuristics. Alternatively, a surrogate model of the fitness function can be used instead of true fitness function evaluations. This approach can be applied both to EAs [10] and MOEAs [8].

The second problem is all about speed and can be resolved by using parallelization schemes for EAs and MOEAs on multiple processors. Due to the

C. A. Coello Coello et al. (Eds.): EMO 2005, LNCS 3410, pp. 92–107, 2005.

population based approach of EAs, single-objective EAs are very easy to parallelize. There is a large amount of literature on parallel EAs [3] and we will outline three major parallelization schemes for Single-Objective EAs in Sec. 2.1. But parallelization schemes for MOEAs are not as frequent, a good overview is given in [18], although the need for efficient parallelization schemes is even greater.

The fact that MOEAs search for a whole set of solutions (the Pareto front) instead of a single solution, suggests dividing the optimization problem into multiple subproblems, which are hopefully more efficient to solve. This approach is called the 'divide and conquer' approach and offers the prospect of parallelization schemes, which are even more efficient than non-parallelized MOEAs. The problem is, how to find a suitable partitioning of a given optimization problem without additional *a priori* knowledge about the topology of the search space? And second, is the 'divide and conquer' approach actually feasible? We suggest to use clustering algorithms to analyze the current population to find a suitable search space partitioning to aid the 'divide and conquer' approach and test whether or not an advantage of the 'divide and conquer' idea can be observed on standard benchmark functions.

The remaining publication is structured as follows: First, we outline the current state of the art regarding parallelization schemes for single and multi-objective EAs in Sec. 2 and we give details on the new clustering based parallelization scheme for MOEAs in Sec. 3. Then, the new algorithm is compared to other MOEA parallelization schemes on multiple test functions in Sec. 4. Finally, we discuss whether or not the 'divide and conquer' approach is actually feasible and how we intend to proceed with our future research on parallelizing MOEAs given in Sec. 5.

## 2 Related Work

As outlined before, EAs are well suited for parallelization, due to the population based search strategy. First, because individual alternative solutions of a population can be evaluated in parallel, same holds true for mutation and crossover. And second, although selection typically occurs on the whole population in a standard EA, selection could also act on a local subset of a population for several generations instead, without suffering major losses in performance. These two properties are usually used to parallelize EAs.

### 2.1 Parallel Evolutionary Algorithms

The most common parallel EAs can be grouped into one of three categories [3]:
**The Island Model.** utilizes the second property by running independent subpopulations on multiple processors. To prevent premature convergence and to increase convergence speed, every $m_{rate}$ generations the $w$ best individuals are migrated from one subpopulation to another. This approach is well suited for computer clusters, due to the limited amount of communication.