Local Optimization in Global Multi-Objective Optimization Algorithms

Algirdas Lančinskas, Julius Žilinskas
Vilnius University

Andreas Grothey
University of Edinburgh

Introduction

Many real world optimization problems deal with more than one objective functions. The difficulty arises when there are conflicts among the different objectives – improving values of one objective leads to deterioration of another. Often it is impossible to find a single solution which is the best by all objectives. Therefore the goal of solving Multi-objective Optimization Problem (MOP) is to find a set of compromising solutions, called Pareto Set.

One of the most popular algorithms for solving MOPs is Non-dominated Sorting Genetic Algorithm (NSGA-II). NSGA-II is genetic algorithm based on population elements sorting according to the Pareto dominance relation.

A new Multi-objective Optimization Single Agent Stochastic Search (MOSASS) algorithm has been developed by modifying the single-objective local optimization algorithm Single Agent Stochastic Search (SASS) and hybridizing it with the NSGA-II.

Strategies to Parallelize

The main performance bottleneck in parallel NSGA-II algorithm is frequent communication between processors. Therefore the communication strategy is very important for the performance of the parallel algorithm.

Four strategies have been investigated during our experimental investigation. All strategies distribute and gather data from all processors using hierarchic fashion. First of them – NSGAPar – divides population among all processors two times per generation – for generation and evaluation of new population, and for Pareto ranking. The second strategy – NSGAParHS – performs one distribution of population and makes Pareto ranking while new is being gathered. The third and fourth strategies – NSGAParRed and NSGAParHSRed – are analogous to the previous ones respectively, but new population is reduced by removing probably bad solutions before gathering and Pareto ranking.

Results of the Experimental Investigation

The MOP consisting of two objectives with 10 variables (MOP_{10}), and the same MOP with 50 variables (MOP_{50}) have been used during investigation. Depending on the population size, different numbers of processors have been used.

The speed-up of the algorithm values using different strategies for solving MOP_{50} are given in the Table 1. Green and red colors indicate the best and the worst values respectively.
The speed-up using strategy NSGAParRed for solving MOP\textsubscript{10} are given in the Table 2. Denotations “Th=4” and “Th=16” means that mixed MPI-OpenMP algorithm was used with 4 and 16 shared memory processors respectively.

The same experiment has been performed using population size 1024 for solving MOP\textsubscript{50} (Table 3).

The strategy NSGAParRed has been applied for parallel version of hybrid algorithm NSGA+MOSASS. Obtained algorithm NSGA+MOSASSParRed has been experimentally investigated by solving MOP\textsubscript{10} using population size 1024 (Table 4). Local search has been performed for 512 selected points. Parameter LI indicates number of MOSASS iterations. Local search was parallelized by dividing selected points among processors. Therefore no more than 512 processors can be used for the local search.

Since 512 points are selected for local optimization, performing computations on 1024 processors, half of them have to be idle during local optimization. In order to avoid this, asynchronous algorithm NSGA+MOSASSParRedAsync, which enables the idle processors to perform a number of NSGA generations, has been developed (Table 4).
<table>
<thead>
<tr>
<th># of processors</th>
<th>128</th>
<th>256</th>
<th>512</th>
<th>1024</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSGA+MOSASSParRed (LI=10)</td>
<td>118.1</td>
<td>228.2</td>
<td>431.9</td>
<td>637.2</td>
</tr>
<tr>
<td>NSGA+MOSASSParRed (LI=20)</td>
<td>122.7</td>
<td>240.9</td>
<td>455.9</td>
<td>657.5</td>
</tr>
<tr>
<td>NSGA+MOSASSParRedAsync (LI=20)</td>
<td>122.7</td>
<td>240.9</td>
<td>455.9</td>
<td>715.7</td>
</tr>
<tr>
<td>NSGA+MOSASSParRedAsync (LI=20, Th=16)</td>
<td>126.4</td>
<td>255.9</td>
<td>498.8</td>
<td>834.8</td>
</tr>
</tbody>
</table>

**Acknowledgments**

The work was carried out under the HPC-EUROPA2 project (project number: 228398) with the support of the European Commission - Capacities Area - Research Infrastructures.

This work made use of the facilities of HECToR, the UK's national high-performance computing service, which is provided by UoE HPCx Ltd at the University of Edinburgh, Cray Inc and NAG Ltd, and funded by the Office of Science and Technology through EPSRC's High End Computing Programme.