Project in Computational Science, fall 2014. ABB, Corporate Research

A high performance procedure for feasible initial guesses generation for nonlinear optimization

The problem of finding feasible initial guesses is an important issue in many industrial optimization models. Such examples are hot rolling applications, model predictive control, and many others.

In these applications, the optimization models are extremely computationally demanding, so that the single function evaluation could take several minutes, one iteration up to 2 hours and the complete optimization process takes around 43 hours. The optimization problems are solved using MATLAB optimization solvers such *as multistart, globalsearch, patternsearch*, and *fmicon*.

It is equally important to compute feasible initial guesses and to do this efficiently. Thus, the procedure of seeking for the feasible initial guesses is separate preprocessing step triggered prior to the main optimization loop.

The goal of this project is to develop the MATLAB based preprocessing algorithm based on subrandom number generation procedure. The algorithm should include the feasibility check for the initial guess found. Furthermore, the parallel technique are to be implemented in order to speedup the algorithm.

The prototype developed in this project will be used for cloud/cluster high performance optimization. After the end of this project, we consider a possibility to continue the work in frame of a Master Thesis. This decision will be made prior to the end of the project.

Contents of the project

The task is to develop a MATLAB code generating the set of feasible initial guesses for the given optimization problem. Objective function, constraint set are defined as MATLAB functions, and a single feasible initial guess is given, too. The input parameters are the number of input initial guesses, the output parameter is the set of feasible initial guesses generated by the code. The parallel implementation should be incorporated, too.

1. Prestudy
   - 1.1 Study gradient-based local solver *fmincon* and global solvers *multistart* (parameter setting, the way initial guesses are chosen and parallelization is implemented)
   - 1.2 Study the MATLAB based subrandom points generator *haltonset.m*
   - 1.3 Study the way the parallel for loop operator *parfor* is implemented

2 Algorithm implementation
   - 2.1 Start with *multistart* method.
     - 2.1.1 Run this solver with different settings for initial guess generations. Generate at least 100 points. Analyze the effect of changing the 'StartPointsToRun' from 'off' to 'bounds' and 'bounds-ineqs' options. Analyze the feasibility of the initial guesses.

Global Optimization Toolbox is required!

Remember that only initial guess generation part is of interest, not the solution of the optimization problem.

2.1.2 Generate the set of initial guesses using *haltonset* solver and use them as initial guesses for *multistart.*

2.1.3 Compare the results of 2.1.2 and 2.1.3

2.2 Generate the set of initial guesses using *haltonset* function but no *multistart* function.

2.2.1 Firstly generate at least 100 subrandom points. Check their feasibility with respect to all constraints. Filter only feasible points and use them as an output.

2.2.2 Implement the same procedure in parallel with 500 -1000 points. Make records of the results from the execution one node with 1 – 12 workers. The values to be analyzed are the amount of the feasible points and CPU time.

Parallel Computing Toolbox is required!

2.3 (optional) Run the same tests on cluster with as many initial guesses and workers as possible. *Distributed Computer Server Toolbox is required*

3 Summary and conclusions

3.1 Compare the results obtained in 2.1 and 2.2.

3.2 Make a table with results drawn from experiments in 2.2.2;

3.3 Plot a speedup curve comparing the CPU time for the single and multiple core execution in 2.2.2 and 2.2.3 (if any);

3.4 Analyze and make a summary of the results.

Prerequisites

It is expected that students working on this project have experience in MATLAB programming, high performance computing as well as in numerical optimization.