

Problem Set 3

PROBLEM 1. [100 points]

In class, we will soon study the subject of *genetic algorithms*. In the mean time, here a few introductory pointers.

http://en.wikipedia.org/wiki/Genetic_algorithm

<http://www.cs.colostate.edu/~genitor/MiscPubs/tutorial.pdf>

http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/tcw2/report.html

http://www.scholarpedia.org/article/Genetic_algorithms

http://www.egr.msu.edu/~goodman/GECSummitIntroToGA_Tutorial-goodman.pdf

<https://www.lri.fr/~hansen/proceedings/2012/GECCO/companion/p917.pdf>

We will follow the terminology of Darrell Whitley in *A genetic Algorithm Tutorial*, that is:

<http://www.cs.colostate.edu/~genitor/MiscPubs/tutorial.pdf>

The goal of this problem is to realize a Julia implementation of a genetic algorithm solving the *Traveling Salesman Problem*. See for instance:

http://en.wikipedia.org/wiki/Travelling_salesman_problem

<http://www.dbai.tuwien.ac.at/staff/musliu/ProblemSolvingAI/Class9GATutorial.ppt>

Let N be the number of cities to be visited by the Traveling Salesman. We assume that the distances between each pair of cities are known and recorded in a 2D array A . Precise values for N and the array A will be posted later in a revised version of this problem statement.

We denote by \mathcal{P} the current generation and by P be the cardinality of the \mathcal{P} . Each individual in \mathcal{P} is a permutation of the numbers $1, 2, \dots, N$ and represents a possible travel for the Salesman.

The *evaluation function* ℓ maps any individual x to the total length $\ell(x)$ of the trip encoded by x . Let $\hat{\ell}$ be the average of a trip in \mathcal{P} . The *fitness function* f maps any individual x to the ratio $f(x) = \frac{\hat{\ell}}{\ell(x)}$.

Crossover and mutation must be performed as described in

<http://www.dbai.tuwien.ac.at/staff/musliu/ProblemSolvingAI/Class9GATutorial.ppt>

Precise formulas will be posted later in a revised version of this problem statement.

The selection of the individuals participating to the *recombination phase* is performed as follows:

1. $c(x) := \lfloor f(x) \rfloor$ copies of x are placed randomly in the *intermediate generation*, for each $x \in \mathcal{P}$. Let Q be the total number of copies selected in the intermediate generation.
2. Choose $P - Q$ individuals from \mathcal{P} with the constraint that each $x \in \mathcal{P}$ has a probability $f(x)$ if $f(x) < 1$ and probability $x - f(x)$ otherwise. Add those $P - Q$ individuals to the intermediate generation.

The stopping criterion for the algorithm will be as follows

1. If $N \leq N_0$, the algorithm stops when all individuals in \mathcal{P} have the same fitness value.
2. If $N \geq N_0$, the algorithm stops when $\hat{\ell}$ has the same value in two consecutive generations,

where $N_0 = 40$ should work. If this makes your programs too slow, you can set N_0 to 30 (which is known to work in practice).

Question 1. [30 points] Write a serial `Julia` program implementing the above genetic algorithm for solving the Traveling Salesman Problem. The input parameters of the main function must be N , A and the initial value of \mathcal{P} meanwhile its return value must be an individual (from the final generation) with optimum fitness.

Question 2. [50 points] Write a parallel `Julia` program implementing the above algorithm. It is suggested to use two distributed arrays: one for the current generation and one for the intermediate generation. Meanwhile, each processor will have a local copy of the array A for speeding up access time.

Clearly, the main challenge in this parallel program is the concurrent construction of the intermediate generation.

You should explain the design of your implementation and clearly document each of the functions of your code.

Question 3. [20 points] Collect running times and the number of generations for both the serial and the parallel programs for $N = 20, 25, 30, 35, 40, 45, 50$ and $P = N, 10N, 100N$. You should organize the collected data in tables or charts and discuss the observed performance results.

Submission instructions.

Format: Questions 1 and 2 involve programming with `Julia`: they must be submitted as two input files to be called `Q1.jl` and `Q2.jl`, respectively. Each of these two files must be a valid input file for `Julia`. In addition, each user defined function must be documented and the measured running times and answers to Question 3 should be presented in a separate PDF file called `Q3.pdf`.

To summarize, each assignment submission consists of three files: Q1.jl, Q2.jl and Q3.pdf.

Submission: The assignment should be returned to the instructor **and** the TA by email.

Collaboration. You are expected to do this assignment *on your own* without assistance from anyone else in the class. However, you can use literature and if you do so, briefly list your references in the assignment. Be careful! You might find on the web solutions to our problems that are not appropriate. For instance, because the cache memory model is different. So please, avoid those traps and work out the solutions by yourself. You should not hesitate to contact the instructor or the TA if you have any question regarding this assignment. We will be more than happy to help.

Marking. This assignment will be marked out of 100. A 10 % bonus will be given if your answers are clearly organized, precise and concise. Messy assignments (unclear statements, lack of correctness in the reasoning, many typographical or language mistakes) may give rise to a 10 % malus.