FFR 105 - Report Home Problem 2

Name : Pierre-Eliot Jourdan
Civic registration number : 20001219-T136

Problem 2.1, 3p, The traveling salesman problem (TSP) (mandatory)

In this problem we want to build an Ant System algorithm for solving the traveling salesman problem (TSP).

(*See Matlab code – file HP2 Problem1*).

To run the algorithm, we used the following parameter set: $numberOfAnts = 50$, $\alpha = 1$, $\beta = 3.0$, $\rho = 0.3$ *and* $\tau 0 = 0.1$.

After a few runs, we found the shortest possible path between the cities. The script BestResultFound.m gives us the following vector with the city indices corresponding to the shortest path (50 *elements*):

$bestPath$ = [35 41 25 32 50 29 47 30 34 42 28 46 48 33 38 45 27 39 44 40 37 49 36 8 16 21 7 19 20 11 3 6 15 12 2 14 22 17 24 23 4 10 9 13 1 5 18 26 31 43].

Note: *I used the function the Matlab function fprintf() to store the best path in the script BestResultFound.m.*

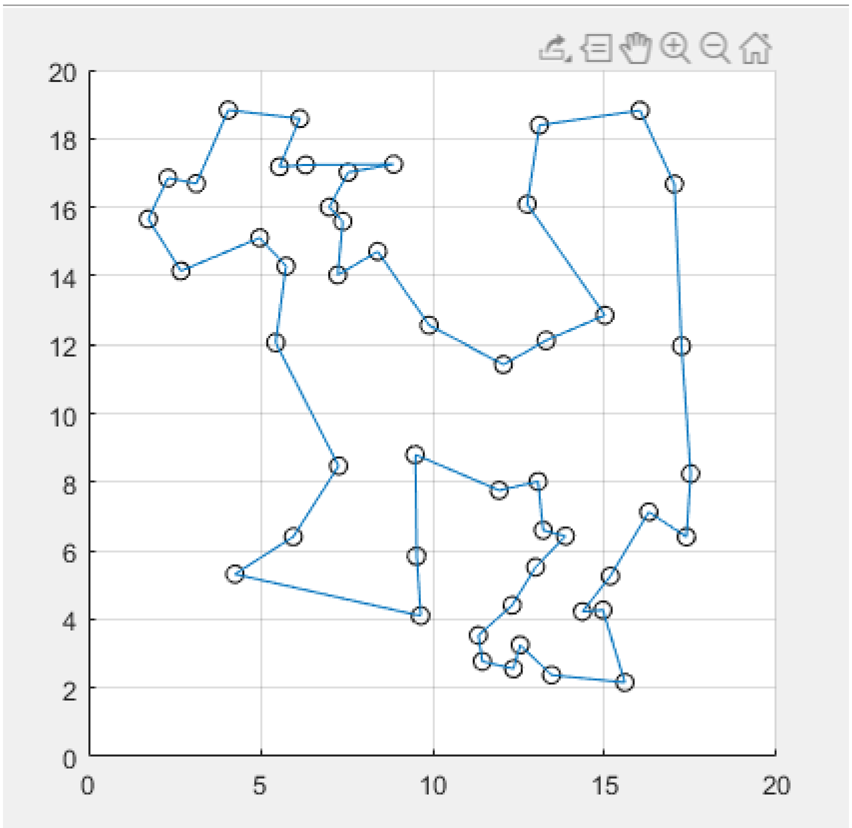
Then, the following Matlab output (in the command window) shows us the length of all paths found (the last one corresponding to the shortest one) :

```
Iteration 1, ant 2: path length = 175.18341
Iteration 1, ant 5: path length = 173.07120
Iteration 1, ant 6: path length = 157.61905
Iteration 1, ant 41: path length = 156.33861
Iteration 1, ant 44: path length = 148.72721
Iteration 2, ant 2: path length = 141.21794
Iteration 2, ant 16: path length = 133.62209
Iteration 3, ant 17: path length = 131.51853
Iteration 3, ant 37: path length = 131.44330
Iteration 3, ant 47: path length = 127.84032
Iteration 4, ant 3: path length = 118.75796
Iteration 5, ant 33: path length = 111.32256
Iteration 7, ant 14: path length = 110.95173
Iteration 9, ant 27: path length = 108.94589
Iteration 13, ant 24: path length = 106.62610
Iteration 14, ant 21: path length = 106.37631
Iteration 16, ant 45: path length = 106.20910
Iteration 17, ant 26: path length = 104.59571
Iteration 28, ant 46: path length = 104.01258
Iteration 32, ant 41: path length = 97.39834
>>
```

We notice that the length of the shortest path is around 97.4.

Finally, here is a plot of this best path:

Plot of the best path between the 50 *cities*



We also notice that, as required in the instructions of Problem 2.1, *each city is visited once and the tour returns to the starting city.*

Thus, in this problem we implemented an Ant System algorithm in order to solve a TSP. The length of the shortest path is 97.4 *and the plot of the corresponding path proves, indeed, that this is the shortest path.*

Problem 2.2, 3p, Particle swarm optimization (mandatory)

In this problem we want to implement and use a particle swarm optimization (PSO) *algorithm to find the location of all the local minima of the function* :

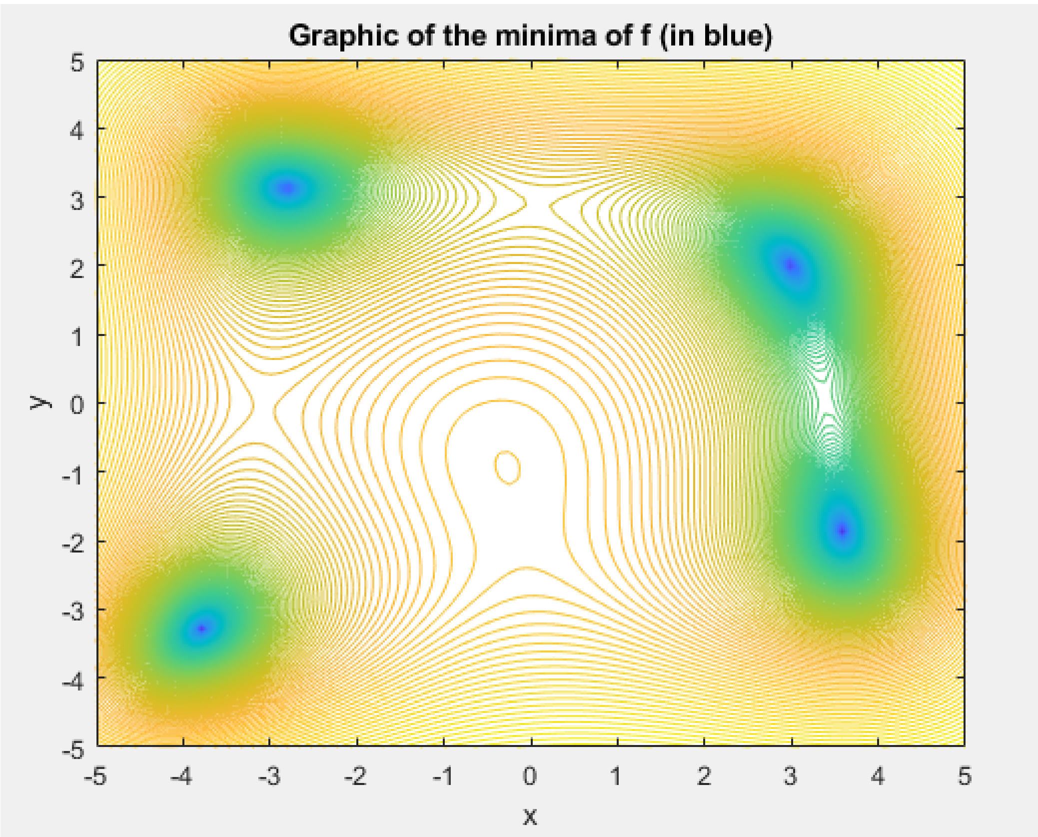
$$f(x, y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2 \quad (1)$$

over the range $(x,\ y) \in [-5,\ 5]$.

(See Matlab code – file HP2 Problem2).

About the code, I initialized the inertia weight to 1.4 *and set the decreasing factor* β *to* 0.9 *(as required in the course book). We also set the parameters* α *and* ΔT *to* 1, *and the constants* $c1$ *and* $c2$ *to* 2, *which are also the default parameters in the course book. Then, as a termination criterium, I chose a certain large number of iterations performed:* $nIterations = 200$.

First, as required in the instructions of the problem, here is the contour plot of the function $ln(a + f(x, y))$ *with* $a = 0.01$. *For more visibility, I chose to plot* 200 *contours of this function* :



Note : It is possible to plot $ln(a + f(x, y))$ *over the range* $(x, y) \in [-5, 5]$ *as* $f(x, y) \geq 0$ *for* $(x, y) \in [-5, 5]$ *(as a sum of two squared terms).*

The dark blue zones on the figure represent the minima of $ln(a + f(x, y))$ *which are also those of* $f(x, y)$ *since* $a = 0.01$ *is constant and ln is growing on* $]0, +\infty[$*. We notice that there are 4 minima.*

Then, let's run the implemented PSO algorithm to find the minima of f. Few runs are needed to determine all of them. At each of these points, the following table give the value of the function f :

x	y	$f(x, y)$
3	2	0
-3.7793	-3.2832	1.8827e-8
-2.8051	3.1313	1.6631e-8
3.5844	-1.8481	4.7176e-8

Conclusion: The PSO algorithm helped us to determine the values of the 4 local minima of $f(x, y)$ *on the contour plot. Indeed the first point in the table correspon to the upper right dark blue zone on the figure, the second to the lower left one, t third to the upper left one and the fourth to the lower right one. In addition, acco to the values of* $f(x, y)$ *in the table, f has a global minimum over the range* $(x, y) \in [-5, 5]$ *reached at the point :*

$$P = (3, 2).$$

Problem 2.3, 5p, Optimization of braking systems (voluntary)

In this problem we want to implement a Genetic Algorithm to optimize the braking system of a truck.

(*See Matlab code − file HP*2 *Problem*3).
I only managed to implement the scripts EncodeNetwork.m and DecodeChromosome.m.
The others are not complete.

Problem 2.4, 4p, Function fitting using LGP (voluntary)

In this problem, we want to implement a LGP program in order to find the best solution fitting with the following function :

$$g(x) = \frac{a_0+a_1x+a_2x^2+...+a_px^p}{b_0+b_1x+b_2x^2+...+b_qx^q}, \quad \text{with } p, q, a_i \text{ and } b_i \text{ unknown constants.}$$

(*See Matlab code — file HP*2 *Problem*3). *I implemented all the scripts of the Genetic Algorithm but my final solution does not fit with the data in LoadFunctionData.m.*

In order to cope with variable — length chromosomes, I decided to use the struct() *concept in Matlab, as required in the problem instructions. After the initialization of the population, I created a structure s with all of its fields named* "*name*" *and each of the corresponding values containing a list representing one chromosome of the population.*

Note: *Like in Problem* 2.4, *I used the function the Matlab function fprintf*() *to store the best chromosome in the script BestResultFound.m. Also, I edited this script the same way as LoadDataFunction.m to facilitate its use in TestLGPChromosome.m.*