**COSC 6368 (Dr. Eick) – Fall 2017**

**Project 1: Solving Travelling Salesman Problems**

Individual Project

Third Draft



|  |  |
| --- | --- |
| Due Date: | Friday, October 6, 11p (5% bonus); Monday, October 9, 11p |
| LastUpdated: | September27, 2017, noon |

Design two search strategies that solve N-city Traveling Salesman Problems (TSP) with cities being numbered 0,…,N-1; one of these search strategies (SIM) can be quite simple; however, the second strategy (SOPH) should be more sophisticated. Design a software system that implements SIM and SOPH and provides the capability to run the SIM and SOPH for 3 given cost functions (specified in the Benchmark section at the end of this doucment). Your software should have an interactive[[1]](#footnote-2) interface[[2]](#footnote-3) that allows you to select

* the search strategy (either SIM or SOPH)
* the cost function (either c1 or c2 or c3)
* the number of cities N for which the TSP should be solved
* a maximal effort bound MEB (maximum number of states searched)
* an integer random number (optional parameter; is only needed if your search strategy is randomized in which case the supplied random number is used as the seed for the random number generator you use in your implementation).

The software you develop then runs the selected search strategy with the selected parameters exploring at most MEB[[3]](#footnote-4) states and reports in an output file the best solution found, its cost, the final MEB value, and maybe other summaries about the search conducted.

Write a report (approx. 8 to 11 single spaced pages) that gives a clear description of the explored and employed search strategies, describes the evolution of the project, and summarizes the results of running the system for the TSP project benchmark. Moreover, the report should interpret the experimental results you obtained for the TSP-benchmark and summarize the important findings and accomplishments of your project. in a short conclusion.

**TSP Project Benchmark**

Assumption: Cities are numbered 0,…,N-1 (N is the number of cities for the problem); moreover, you can assume that all cities are connected with the cost of traveling from one city to another city being defined by a cost function. In this project we will use cost functions named c1, c2, and c3 that are defined as follows:

c1(x,y)= if x=y then 0

else if x<3 and y<3 then 1

else if x<3 then 200

else if y<3 then 200

else if mod(x,7)= mod(y,7) then 2

else |x-y|+3

c2(x,y)= if x=y then 0

else if x+y<10 then |x-y|+4

else if mod(x+y,11)=0 then 3

else (|x-y|\*\*2)+10

c3(x,y)= if x=y then 0

else (x+y)\*\*2

Example Cost Computations (N=5):

Cost(0-1-2-3-4,c1)= 1+1+200+4+200=406

Cost(0-1-2-3-4,c3) = 1 + 9 + 25 +49 +16=100

Cost(0-2-4-1-3,c3) = 4+ 36 +25 +16 +9=90

***Experimental Evaluation***

Run your system for SIM and SOPH for the cost functions c1, c2, and c3 and the following values for N: 10, 30 and 60. Additionally, if feasible, run your program for c1 and c2 for N=120. The best solutions found and their cost and wall clock time needed for those 11 test cases should be included in your report; if your were not able to obtain any results due to time limitations, space limitations, program error, report TL, SL, PE for the particular test case. The running of the program should be limited to 20 minutes for each test case; if your program needs more than 20 minutes to complete a test-case report ‘TL’. However, for N=120 you can run your program for up to 1 hour.

If one or both of your search strategies are randomized, run each test case 3 times using a different random number seed; and report each result of the three runs.

**Peer Review:**

It is the most successful and established method for evaluating academic performance. We hope practicing it for the course projects provide you an opportunity to learn from your classmates (both well done and mistakes) as well.

**How it works:**

Every student has to submit two zip folder. One including her information, another anonymous. After the due, I will assign each student a random project to review. She has to grade it and provide me her one-page concise review, including:

1. Strengths: This can be some comments about the implemented algorithms, coding style, documentation and reporting.
2. Weaknesses: Same as the previous one.
3. Suggestions to improve (report&/code): Any suggestion you can provide for improvement. Try to be as clear and specific as you can.
4. Overall evaluation (assign a number grade between 0 and 100 to the project; assume 83 is the median number grade for project1).

Moreover,

* You will receive some credit for your Peer review (these points will be added to your project 1 score)
* Initial project1 scores will be potentially adjusted based on Peer reviews.

**Submission instructions:**

Create a folder and name it as LastName\_StudentId\_Project1. Then create a sub folder in it named project1. Project1 folder should include:

* The program source code
* The executable
* A readme.txt file on how to execute your program
* A file called *2runs.txt* that contains the output that your program created for the testcase N=30, CostFunction=c2 when running SIM and SOPH; just past the output of those 2 program runs into the file *2runs.txt*..
* The project report

Next make a copy of your project1folder and rename it to anonymous. Then edit files inside the anonymous folder and remove any information that a reviewer can identify you. At the end, you should submit your LastName\_StudentId\_Project1 folder in a zipped folder through your blackboard. if you are going to submit via email (in case of problems with Black Board), please use this keyword for the beginning of the subject: [Cosc6368-Fall2017-Project1]

**Academic Honesty:**

Reporting false results or solutions that were not obtained by running your programs is a serious academic honesty violation. Moreover, if you used software not developed by yourself, describe this software in detail, and its role in your project. Not reporting external software you used in this project is an academic honesty violation. Moreover, if you use a software that solves “most” of the actual problem, you will receive a low grade, even if you acknowledge that you used the particular software.

1. Having a command-line interface is also fine! [↑](#footnote-ref-2)
2. It might be also worthwhile to implement a second interface that reads input parameters from a file and is capable to run the system for multiple test cases, as this will facilitate creating experimental results more easily. [↑](#footnote-ref-3)
3. If MEB states have been searched, your program should abandon the search and terminate, possibly reporting the best solution you have found so far. You will get instructions what values for MEB to use when running the benchmark later in the semester! [↑](#footnote-ref-4)