Introduction
à l'optimisation
Fall 2014 - 2015

Enseignants: M. Bierlaire
Assistant: Sh. Sharif Azadeh
Semester Project

Traveling Salesman Problem (Mathematical Model/Heuristic Approach)

1 Instruction

The project focuses on implementing mathematical model and applying heuristics for the Travelling Salesman Problem (TSP).

1) Each group is provided with a sample of data (coordinates of different cities). In the data file, every line repesents the coordinate (x_i, y_i) of each city (i) on the map.
2) Each group has to solve the problem with their assigned data set (the data file is different for each group).
3) All questions should be answered inside the formatted file named "Answer.doc".
4) You are provided with an informative description about the TSP model and heuristic methods at the end of this document.
5) All groups are provided with the framework (skeleton) of MATLAB files (m files) that need to be completed.

2 Project Description

This problem needs to be solved with two resolution approaches:

2.1 Mathematical model

There are many mathematical representations for Travelling Salesman Problem. Here, you are given an assignment based formulation for TS problem (next page).

In this case, the **decision variables** are expressed as binary variables ($x_{ij} \in 0, 1$), because each city has to be met only once by the salesman. The value of x_{ij} equals to one if city i is connected to city j and it equals to zero otherwise. In the first part, you are asked to solve the problem ignoring integrality condition of x_{ij}. That is, you need to solve the relaxation problem (1)-(4). Then, you are asked to plot your solution.

If you detect any sub-tours, you could add constraint (5) to eliminate the sub-tours as needed. Then, again, you need to plot your solution. You add these constraints to eliminate subtours that have been detected in the previous step. You might end up finding other subtours of different sizes but you do not need to go further and you can stop at this point.

2.2 Heuristic Approach

Motivation to use heuristic approach: As you will see, the number of added constraints to eliminate sub-tours increases exponentially. It takes a lot of time to solve the problem using the mathematical model if we want to avoid all possible sub-tours. In this case, usually, heuristic algorithms are used to find a "good" solution in a reasonable time. In this project you are asked to implement two heuristics.

H1 **Greedy heuristic:** This heuristic will be used to construct an initial solution. **An initial solution means "order of visited cities in a tour"**

H2 **2-Opt heuristic:** This algorithm is used to improve the solution obtained from the greedy method. 2-Opt as seen in the course and in the lab, is a searching strategy that swaps the order of visited cities pairwise. We continue the search as long as the solution is improved or a certain number of iterations is met.

3 Deliverables

A printed copy of "Answer.doc" should be submitted at TRANSP-OR (GC B3 444) during the working hours before **8th of December at 17h00.**

Every member of each group has to sign the document before submission, otherwise her/his grade will be zero.

The following files are required:

1. A hard copy of "Answer.doc".
2. MATLAB programming files **(in a single .zip format)** have to be sent to :
   shadi.sharifazadeh@epfl.ch
   In the e-mail subject, you should mention your group number in the following format "Project Gr: # " and inside the email please write the names of all group members.

4 Guide on TSP model and heuristic algorithms

4.1 Mathematical Model

The TSP can be formulated as an assignment optimization problem with integrality and sub-tour eliminaition constraints. Suppose n represents the number of cities. As mentioned before, binary decision variables x_{ij} equal to 1 if the salesman travels from city i to j. Cost or c_{ij} defines the traveling distance between city i and city j. The mathematical model is presented as follows:

$$\min \quad \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \tag{1}$$

$$\sum_{j=1, i\neq j}^{n} x_{ij} = 1 \quad i = 1, .., n \tag{2}$$

$$\sum_{i=1, i\neq j}^{n} x_{ij} = 1 \quad j = 1, .., n \tag{3}$$

$$0 \leq x_{ij} \leq 1 \quad i, j = 1, .., n \tag{4}$$

(2) and(3) present assignment constrains for Travelling Salesman Problem (these constraints show that each city is met only once by the salesman). After solving the above model, in case sub-tours are detected, sub-tour elimination constraints are required to be added. Here, we introduce a way to write sub-tour elimination constraints:

Suppose that S represents all possible subtours of different sizes.

$$\sum_{i,j\in S} x_{ij} \leq |S| - 1 \quad S \subseteq 2, .., n \qquad 2 \leq |S| \leq n - 1 \tag{5}$$

That is, if we have for example, 8 cities and we want to eliminate the subtours of size three, we need to add $\binom{8}{3} = 56$ constraints to the original TSP formulation (1)-(4).

Example:

Suppose there is a sub-tour for three cities 1,4,5. Therefore, $S = 1, 4, 5$ is a representative of subtours of size three; then,

$$x_{14} + x_{41} + x_{15} + x_{51} + x_{45} + x_{54} \leq 3 - 1$$

is one of sub-tour elimination constraints. That is, if we take a sub-tour $4 \to 1 \to 5 \to 4$ that generates a tour, the above constraint will be violated.

Note: As mentioned before, we notice that by increasing the number of cities, the number of subtours and consequently the number of added constraints become larger and larger.

4.2 Heuristics

4.2.1 Greedy Heuristic (For finding an initial solution)

The Greedy heuristic follows an iterative approach to insert the nearest city to the tour (a tour is an order of visited cities).

Algorithm 1 Greedy Heuristic

$C = \{$set of cities that needs to be visited$\}$
Start from an initial city (Consider row 1 in your data files as an initial city (node))
$k = 1$; $C = C - \{1\}$; $Tour \leftarrow k$
while $|C| \neq \phi$
  $j =$ a city in set C with shortest distance to k
  $k = j$; $C = C - \{k\}$; $Tour \leftarrow k$
Return $Tour$

4.2.2 2-Opt

The 2-opt algorithm starts with an initial solution and aims to improve the current solution by swapping cities.

Algorithm 2 2-Opt Heuristic

$Tour \leftarrow$ Find an initial tour (use the tour that has been obtained by greedy algorithm)
improvement=true;
while improvement=true
  improvement=false;
  for i:1 to $|Tour|$-1
    for j:i+1 to $|Tour|$
      Swap the location of cities i and j in the $Tour$
      evaluate total travelling disctance
      if the solution is improved **then** improvement = true
    end
  end
end