Exercise 4: Part 1

Optimisation with Simulated Annealing

Python Programming Bootcamp by Dr Rohitash Chandra
UNSW, 2021

Introduction

Your are required to use ***object oriented programming*** techniques in your program.

Description

Even though we have seen how dynamic programming algorithms often result in a more efficient algorithm than other approaches, this need not necessarily be the case. The Traveling Salesman problem (TSP) belongs to a class of problems for which it is believed that no efficient algorithms exist, i.e. essentially any algorithm for this problem will have exponential time complexity. Here, you will implement such a dynamic programming algorithm which has exponential time complexity. Although exact solutions for TSP are very difficult to obtain for sufficiently large problems, probabilistic methods may find very good solutions quite quickly. Simulated annealing is a generic optimization method which finds such good solutions for very difficult problems relatively fast. You will implement and compare the solution found with the exact method with that found by simulated annealing.

The Traveling Salesman Problem (TSP) is to determine a shortest route through n cities that starts at the salesman's home city, visits each of the cities once, and ends up at the same city. In this project, you will implement various methods for finding such a tour including probabilistic algorithms.

Implementation:

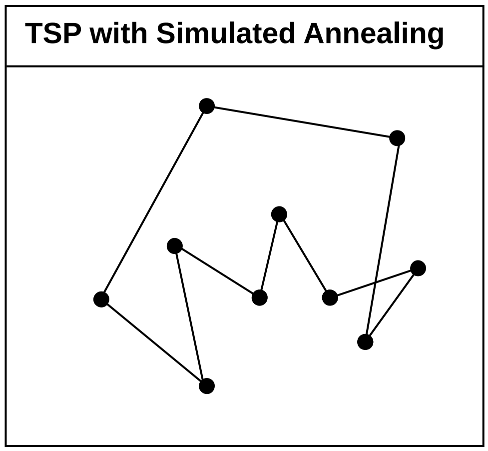
1. Use naive method to solve the TSP for tours for N= 8, 16, 32 cities. What is the time complexity of your program? (20%)
2. Use simulated annealing to solve the TSP for tours for N= 8, 16, 32 cities. Can your implementation handle a tour of 1024 cities? How good are the solutions found by simulated annealing? (40%)
3. Give report of your experiments and discuss your results by comparing them. You need to run 10 experiments for each case and compare the mean.
4. Randomly generate cities and display them in a graphics window. Show graphical display where both algorithms solve TSP for a single run of 16 cities. (30%)

Extra credit problem:

Show the evolution of solution for the simulated annealing after each certain iterations and the best solution found after each tour is computed. (30%)

Output:

Show the solutions found by the two methods in separate graphics windows.



Simulated Annealing:

The algorithm for simulated annealing works as follows for TSP (go and look-up work by Kirkpatrick on the Internet!):

BEGIN
Select an initial temperature T;
Randomly generate a tour S_1 with length $L_{1.}$

WHILE (TRUE) DO
BEGIN
FOR i:=1 TO c DO /* define the number of changes at a constant temperature */
BEGIN
Create a new tour S_2 from tour S_1 by randomly swapping two neighboring cities. Let L_2 be the length of tour $S_{2.}$
IF $L_2 < L_1$ THEN
$S_1 := S_2$; /* accept S_2 as the new tour */
$L_1 := L_2$;
ELSE
Accept S_2 as the new tour S_1 with probability
$P = e^{-x}$ where $x = (L_2 - L_1)/kT$ /* k is Boltzmann constant */
with length $L_{1.}$
END
T := T/(1+T); /* cool down the temperature */
END
END

Note that the probability of temporarily accepting a longer tour declines with decreasing annealing temperature. A detailed description of simulated annealing for TSP is given below from Wikipedia: http://en.wikipedia.org/wiki/Simulated_annealing

Resources:

1. https://www.geeksforgeeks.org/traveling-salesman-problem-tsp-implementation/

2. https://stackoverflow.com/questions/46506375/creating-graphics-for-euclidean-instances-of-tsp
3. Potential Solution: https://github.com/jedrazb/python-tsp-simulated-annealing