CSIT, University of Wollongong

CSCI203/803 Data Structures and Algorithms

Spring 2016

Assignment 4 (Due 11:59 PM Fri 21st October)

Objectives

- Design and implement a program to solve the Travelling Salesman Problem using a greedy algorithm based on nearest neighbours and a branch-and-bound algorithm;
- Use appropriate data structures for different algorithms;
- Analyse the results of different algorithms.

Task

You should write a single program which implements the greedy algorithm based on nearest neighbours and a branch-and-bound algorithm to find approximate and optimal solutions for the Travelling Salesman Problem, respectively.

Program

Your program should read a symmetric matrix from a text file that describes weighted edges of an undirected, complete graph and find the Hamiltonian cycles using the greedy and branch-and-bound algorithms. Your program should also read a list of city names corresponding to the vertices of the graph represented by the adjacency matrix. The program will display the tour and related information.

Since this assignment focuses on the algorithmic solution of the problem, STL or Java Collection or other collection framework of the programming language of your choice can be used. Library functions can also be used.

Your program should be readable and well commented.

Data Structures and Algorithms

- Greedy algorithm
  Strategy: Nearest neighbour

- Branch-and-bound algorithm
  Upper bound: weight by nearest neighbours
  Lower bound: minimal weight without constraints
  Strategies:
  1. Breadth-first;
  2. Depth-first.

Choose suitable data structures for the algorithms and strategies, to be described in your report.

Requirements

<u>CSCI203 students</u> (10 marks or 8, 6 or 4 marks)

1. Program
   You have two choices on how to write your code.
   - Implement the greedy algorithm and Branch-and-bound algorithm with a strategy of your choice; (3 and 5 marks, respectively)

   or
   - Integrate the greedy algorithm and/or Branch-and-bound algorithm with a strategy of your choice. In this case, you may use or adapt the code available in public domains. Properly cite the sources of your code in your report and program. (1 mark for each)

2. Report
   a. Cover page
      Student name
      Subject code
      Student number
      Email ID
      (0 mark; your assignment will not be marked if there is not this section)

   b. Methods and result analysis (no more than 300 words)
      Describe the algorithms, data structures and structure of your code. Analyse and discuss your results from two algorithms. State a conclusion about the algorithms and strategies.
      (2 marks)

<u>CSCI803 students</u> (10 marks)

1. Program
   Implement the greedy algorithm and Branch-and-bound algorithm with two strategies.
   (2, 3 and 3 marks, respectively)

2. Report
   a. Cover page
      Student name
      Subject code
      Student number
      Email ID
      (0 mark; your assignment will not be marked if there is not this section)

   c. Methods and result analysis (no more than 400 words)
      Describe the algorithms, data structures and structure of your code. Analyse and discuss your results from the greedy algorithm and branch-and-bound algorithm and the number of nodes generated with two strategies. State a conclusion about the algorithms and strategies.
      (2 marks)

Other requirements

- All your programs excluding `a4compile` and `a4run` scripts will have the following header:

```
/*
 * CSCI203 or 803 Assignment 4
 *
 * Student name:
 * Subject code:
 * Student number:
 * Email ID:
 *
 * (The following heading goes all places where appropriate if
 *   all or part of code in your program is not written by you.)
 * Author:
 * Sources: (citation of the sources)
 */
```

- Your programs can be compiled and executed with bash scripts called **`a4compile`** and **`a4run`**, respectively, on Ubuntu setup in the lab.
    - The script **`a4run`** should be configured to run for both input files (Australia_roads.txt and Australia_flights.txt) to produce all results.
    - Failure to compile and run your program to produce all results from your scripts will receive 0 mark.
- Report, named **`report.pdf`**, is created in the PDF format.
- Failure to meet any requirements above may result in loss of marks.

Input

Your program will work with selected Australian cities and find tours to visit each and all of them once *starting from and returning to Wollongong* using the greedy and branch-and-bound algorithms

Your program should read **two input file names** from the **command line**. The first file contains the adjacency matrix and the second file the city names in the order of the vertex indices. The adjacency matrix in the input file has a format as follows.

- The integer number of vertices in the graph.
- The positive real number weights of one vertex per line, delimited by a tab. The value 0 or 0.0 is used for no connection.

The city names in the input file has a format as follows.

- Each city name in one line

The number of cities is the same as the number of vertices in the adjacency matrix.

Input files of two adjacency matrices are provided.

- `Australia_roads.txt`: driving distances between cities
- `Austarlia_flights.txt`: flight time between cities

Apparently some flight time in `Australia_flights.txt` is a fake.

And the input file of the city names corresponding to the adjacency matrices is also provided as `Australia_cities.txt`.

Your program should work with any input files of the same format. It may be tested with an adjacency matrix of a different size and different city names.

Output

Your program should display the results to the standard output. An example of the program output format for an adjacency matrix is shown as follows.

```
1. Greedy algorithm:
   Number of cities:  11
   Tour:
     1 2 6 … 9
     Wollongong -> Sydney -> Canberra -> … -> Perth -> Wollongong
   Total cost: 17637.0

2. Branch-and-bound algorithm (Breadth-first):
   Number of cities:  11
   Upper bound:  17637.0
   Lower bound:  9338.0
   Optimal tour:
     1 2 11 … 6
     Wollongong -> Sydney -> Gold Coast -> … -> Canberra -> Wollongong
   Total cost: 14988.0
```

In the result of the tour, the first line displays the vertex numbers (without the returning vertex) and the second line the city names (with returning city).

You should run your program for both adjacency matrices to produce tours for travelling by road and flight.

For CSCI203 students, your program should display one result of the branch-and-bound algorithm with a strategy of your choice.

For CSCI803 students, your program should display one more result of the branch-and-bound algorithm with another strategy, and the total number of nodes generated with each strategy.

Submission instructions

Your completed assignment should contain your **`source code`** including all programs and scripts, and **`report.pdf`** in a zipped file named **`ass4.zip`**. You should not include any graph matrices in the submission. In this assignment, all input files will be generated by your program/scripts.

To submit your assignment, transfer your **`ass4.zip`** file to **`banshee`** and run the command:

```
turnin -c csci203 -a 4 ass4.zip
```

from the directory containing your file.

1. Late submissions will be marked with a 25% deduction for each day.
2. Submissions more than three days late will not be marked, unless an extension has been granted.
3. If you need an extension apply through SOLS, if possible before the assignment deadline.
4. Plagiarism is treated seriously. Use of code not created by you without citation is plagiarism. If we suspect any work is copied, all students involved are likely to receive zero for the entire assignment.