project: Iran Tour shortest Path

Course title: Data driven Decision Making

Student Name: Mohadeseh(Mona) Najafi

Under supervision of: Dr. Paolo Dell'Olmo

February 2020

Sapienza University

Contents

Introduction

In this project, we want to calculate optimum path (shortest) for a IRAN Tour among different alternative possible paths defined as follows:

This Tour begins from starting point Urmia city (north west of Iran – green point on map) and will finish in Chabahar city (South east of Iran – green point on map) The map and different paths are connecting cities (vertices) as a sequence of edges. Graph, in which every city is a vertex, and distance (km with car) of each two cities are shown as edges. Thus: The shortest path will be selected based on distance

Distance: between two cities that is measured with km Reference: google maps

Thus, every edge has 4 features ; < index, fromnode , tonode, distance>

- Index: number of edge counter
- Fromnode: beginning city of that Edge
- Tonode: destination city of edge

There are 17 cities(Vertex) as follows :

1. Urmia (West Azerbaijan - yellow point)

2. Tabriz (East Azerbaijan)

3. Ardabil

4. Kermanshah

5. Gilan

6. Lorestan

7. Hamedan

8. Khuzestan

9. Isfahan

10.Kashan

11.Tehran

12.Razawi Khorasan

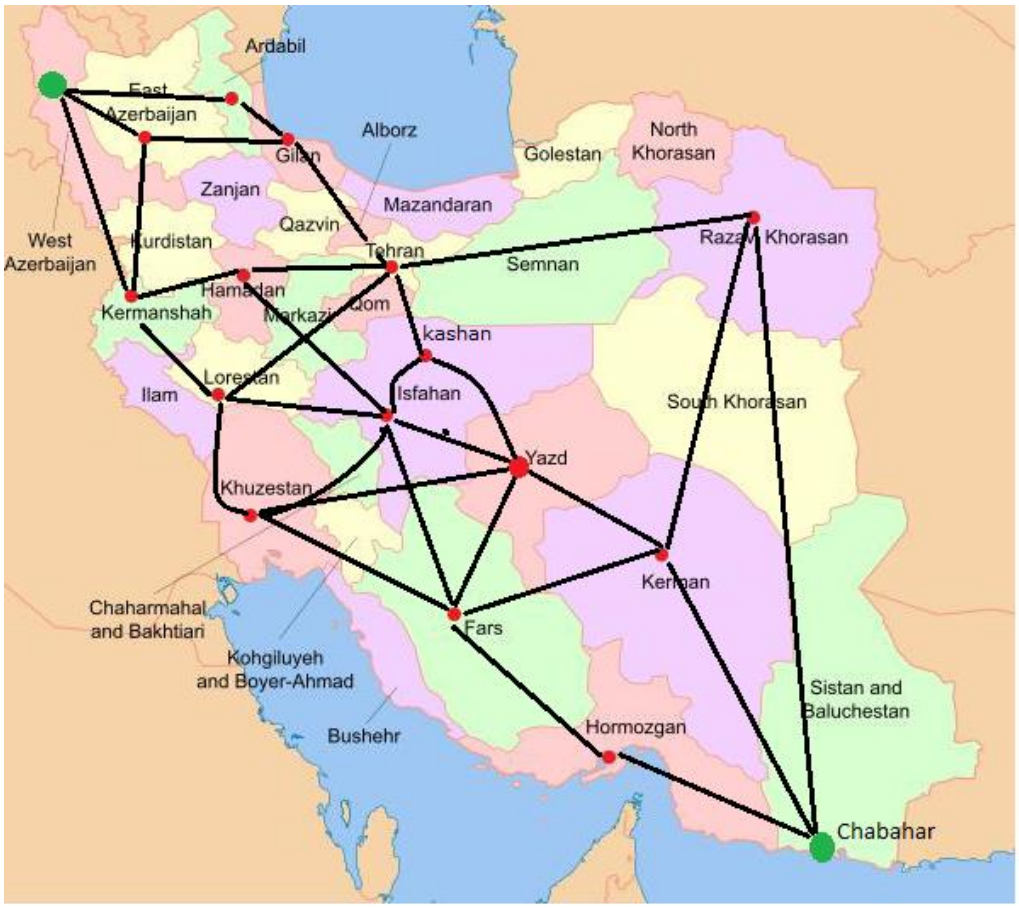
13.Yazd

14.Fars

15.Kerman

16.Hormozgan

17.Chabahar (green Point)



The User of this program will find the shortest path for a Tour with the criteria of distance.

CPLEX Model program;

```
tuple arc {
  key int ID;
  key int fromnode;
  key int tonode;
  float length;
}

// Get the set of arcs
{arc} Arcs = ...;

// The network flow model has decision variables indexed on the arcs.
dvar boolean Flow[a in Arcs];

// minimize the total length of the path from o to d (point4)
dexpr float TotalLength = sum (a in Arcs) a.length * Flow[a];
minimize TotalLength;

// minimize the total length of the path from o to d (point7)
//dexpr float TotalLength = sum (a in Arcs) a.length * Flow[a];
//dexpr float Discount   = sum (a in Arcs: a.ID == 11) 0.15*Flow[a]*(sum (a in Arcs) a.length*Flow[a]);
//minimize TotalLength - Discount;

subject to {
```

```
  // Preserve flows at each node
  forall (i in Nodes) ctNodeFlow:
    sum (<l,i,j,c> in Arcs) Flow[<l,i,j,c>] - sum (<l,j,i,c> in Arcs) Flow[<l,j,i,c>] == SupDem[i];

 //constrains an arc to be part of the solution (point5) _ optional : if we plan the path necessarily cross
//one specific city
 //forall (a in Arcs: a.ID == 16)
 //Flow[a] == 1;

 //constrains an arc to not be part of the solution (point5) _ optional : if we plan the path necessarily not
//cross a specific city
 //forall (a in Arcs: a.ID == 17)
 //Flow[a] == 0;
}

execute {
  var outputfile = new IloOplOutputFile("outputsinglepath.txt");
  //var outputfile = new IloOplOutputFile("outputsinglepathC.txt");
  //var outputfile = new IloOplOutputFile("outputsinglepathD.txt");
  outputfile.writeln("minimum path solution:")
  outputfile.writeln("Total length ", TotalLength)
  //outputfile.writeln("Total lenth  ", TotalLength - Discount)
  outputfile.writeln("Arc Seclection")
  outputfile.writeln("<from node,to node>");
  for(var a in Arcs)
    if(Flow[a] > 0)
```

```
        outputfile.writeln("<",a.fromnode,",",a.tonode,">");
}
```

Input Data file;

```
// --------------------------------------------------------------------
// Licensed Materials - Property of IBM
//
// 5725-A06 5725-A29 5724-Y48 5724-Y49 5724-Y54 5724-Y55
// Copyright IBM Corporation 1998, 2013. All Rights Reserved.
//
// Note to U.S. Government Users Restricted Rights:
// Use, duplication or disclosure restricted by GSA ADP Schedule
// Contract with IBM Corp.
// --------------------------------------------------------------------
NumNodes = 17;
SupDem = [1, 0, 0 ,0 ,0 ,0 ,0 ,0 ,0 ,0 ,0, 0, 0, 0, 0, 0, -1];
Arcs = {
 //< ID,from, to, length>
  <1,    1,   3,   358 >,
  <2,    1,   2,  146 >,
  <3,    1,   4,  544  >,
  <4,    2,   4,  701 >,
  <5,    2,   5,  497  >,
  <6,    3,   5,  262  >,
  <7,    4,   7,  184  >,
  <8,    5,   11,  328  >,
  <9,    4,   6,  192  >,
```

<10, 7, 11, 318 >,

<11, 6, 11, 484 >,

<12, 11, 12, 891 >,

<13, 10, 12, 1044 >,

<14, 11, 10, 243 >,

<15, 7, 9, 529 >,

<16, 6, 9, 390 >,

<17, 6, 8, 331 >,

<18, 8, 9, 515 >,

<19, 8, 13, 810 >,

<20, 10, 9, 202 >

<21, 10, 13, 391 >,

<22, 9, 13, 432 >,

<23, 8, 14, 627 >,

<24, 13, 14, 499 >,

<25, 9, 14, 481 >,

<26, 13, 15, 369 >,

<27, 14, 15, 593 >,

<28, 12, 15, 1068 >,

<29, 14, 16, 577 >,

<30, 15, 17, 836 >

<31, 16, 17, 667 >,

<32, 12, 17, 1590 >

};

Mission and Vision

For the next step of this optimization scheme, I am planning to work on bi-criteria shortest path problem , based on two criteria of distance and ranking of tour cities.

THE END