CSE 551 Programming Assignment

Sahil Patil

Programming Language: Java

IDE: IntelliJ IDEA

Fundamental underlying algorithm: Ford-Fulkerson Algorithm for Maximum Flow

Dataset:

Data source: Google flights, Expedia, AA, United, Delta

All flight information in CSV format having columns as following:

1. Source
2. Destination
3. Departure time
4. Arrival time
5. Airlines
6. Aircraft type

Input:

Processed information out of above dataset in CSV format having columns as follows:

1. Source
2. Destination
3. Departure time (rounded up to the hour – ignoring the succeeding minutes)
   Example – if time is 11:45 it is rounded up to 11 (irrespective of the minute value)
4. Arrival time (rounded up similarly as above)
5. Capacity (using the aircraft type)

After reading the above input CSV file, we are creating an adjacency matrix to represent the flights in a graph.

Adjacency matrix design:

- We are creating a node in the graph for every hour (0-23) for every airport (excluding LAX and JFK)
- For LAX and JFK, we are creating a node each in the graph
- So total no. of nodes in the graph will be = (no. of airports excluding LAX and JFK) *24 + 2
- We have total 10 airports. Excluding LAX and JFK we have 8.
- Therefore total no. of nodes in the adjacency graph = (8) * 24 + 2 = 194
- Now we have assigned the values of adjacency graph as follows:
    - Example: SFO – ORD 5-10 with capacity 180
    - Then the adjacency matrix will be updated as adjMatrix[SFO_5][ORD_10] = 180
    - We are connecting an hour node of an airport to all other hour nodes of the same airport which have **higher** hour value with an infinite capacity (here Integer.MAX_VALUE)
    - Example: adjMatrix[SFO_1][SFO_2] = Integer.MAX_VALUE

Pseudo code:

```
A.    Create adjacency matrix (initial residual graph)
  1.  Initialize the adjacency matrix of size 194X194 with each value as zero.
```

2. Iterate over each flight in the data, i.e. row in dataset, to update the adjacency matrix with the capacity of the flight. The capacities are added based on (departure time and airport) and (arrival time and airport).
3. For all the direct flights (which start at LAX and end at JFK), add the capacity directly to the result.

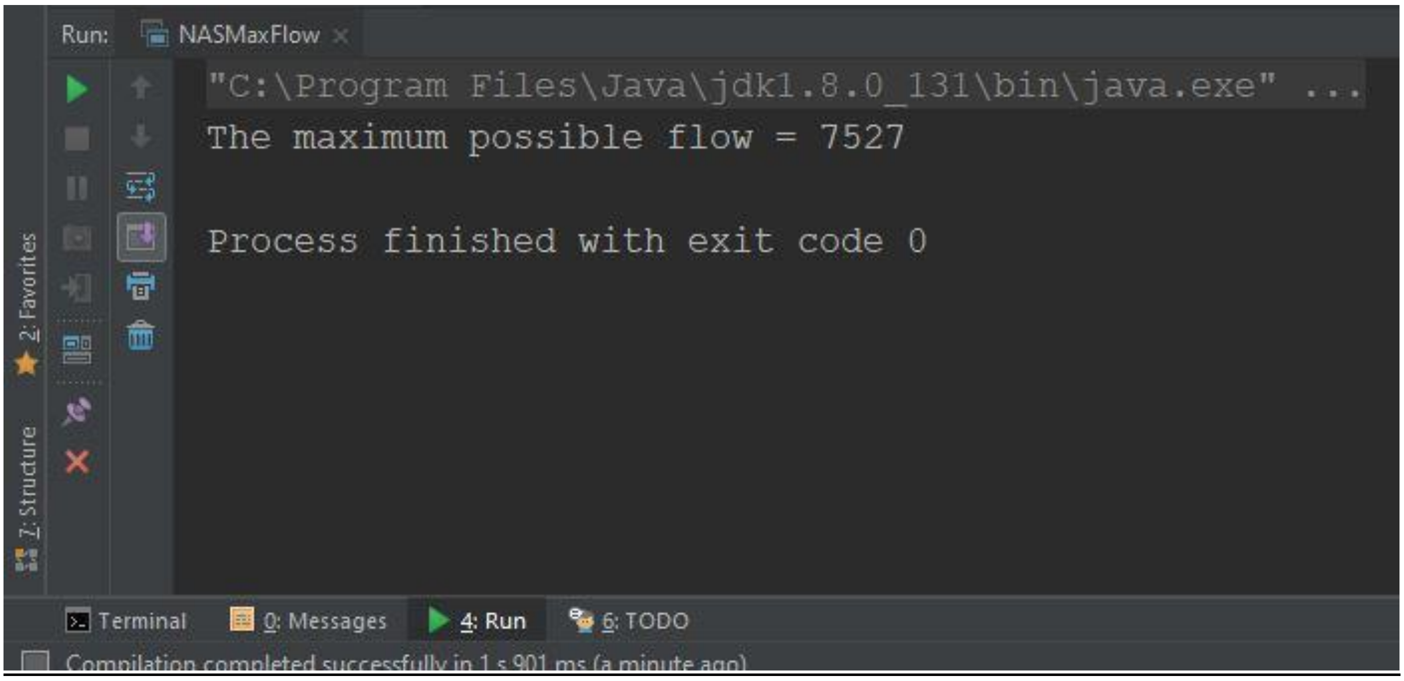
B. Calculate maximum capacity of network (Ford Fulkerson)

```
1.  Initialize MAX_FLOW = 0
2.  While an augmented path from source to sink exists in the residual graph
    a.  Calculate PATH_FLOW = min flow (bottleneck) along the path
    b.  Add PATH_FLOW to MAX_FLOW
    c.  Subtract PATH_FLOW from all edges along the path
3.  Return MAX_FLOW as no augmenting paths exist
```

Time complexity:

- Let the number of airports be N.
- The number of vertices in the adjacency matrix will be [(N*24)+2]. Let this be 'V'.
- So creation of adjacency matrix will be done in **$O(V^2)$** where V=[(N*24)+2]
- Let the number of flights be X.
- To update the details of all the X flights into the matrix we will need O(X) time.
- This matrix is used as an input to the Ford Fulkerson algorithm which will take $O(EV^3)$
- Where V =no. of vertices and E = no. of edges
- Total time complexity is $O(V^2 + X + EV^3) = O(EV^3)$
- **Total time complexity is $O(EV^3)$**

Output:



```
"C:\Program Files\Java\jdk1.8.0_131\bin\java.exe" ...
The maximum possible flow = 7527

Process finished with exit code 0
```

Compilation completed successfully in 1 s 901 ms (a minute ago)