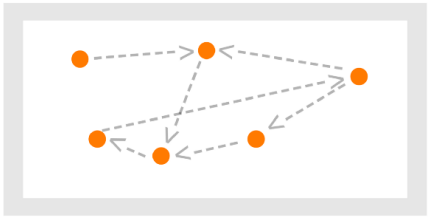
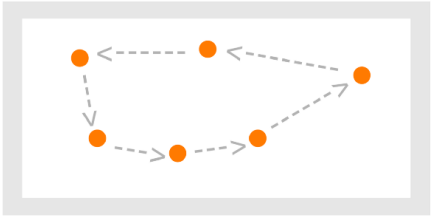


FROM THIS

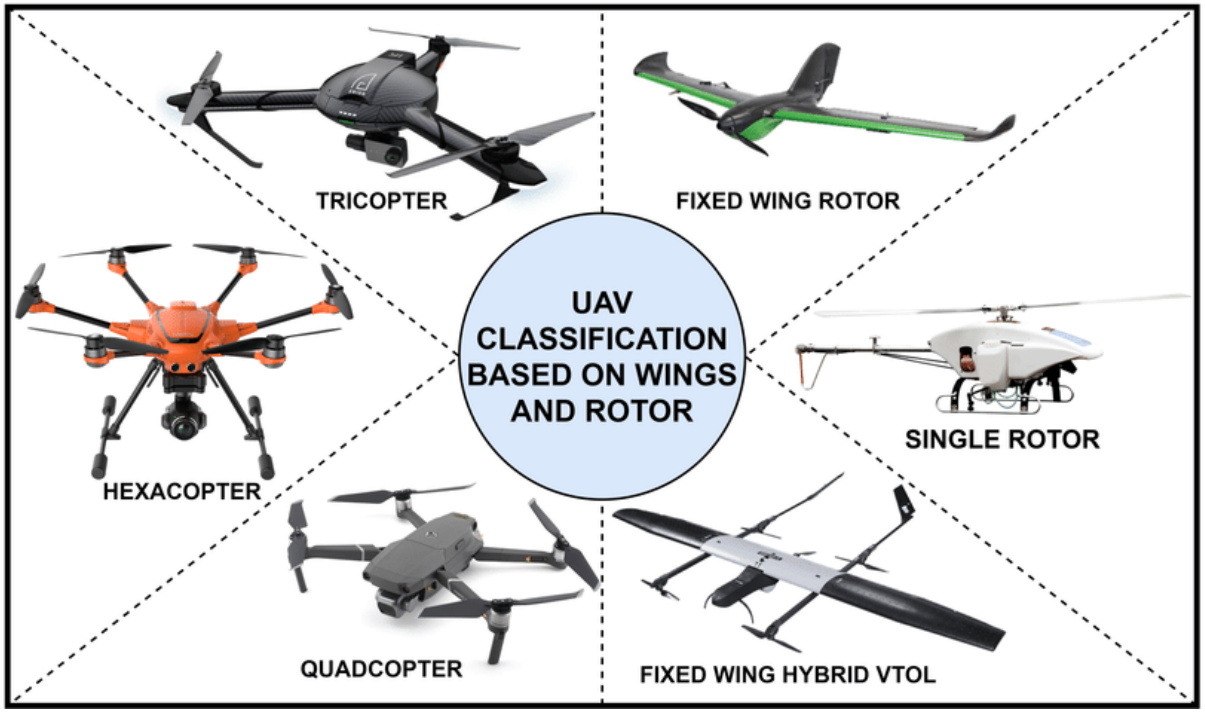


TO THIS

Data Structures
Programming Project #3

Unmanned Aerial Vehicles (UAVs)

- Widely used in many applications:
- Good delivery, target tracking, emergency aid, charging wireless sensor networks, and so on



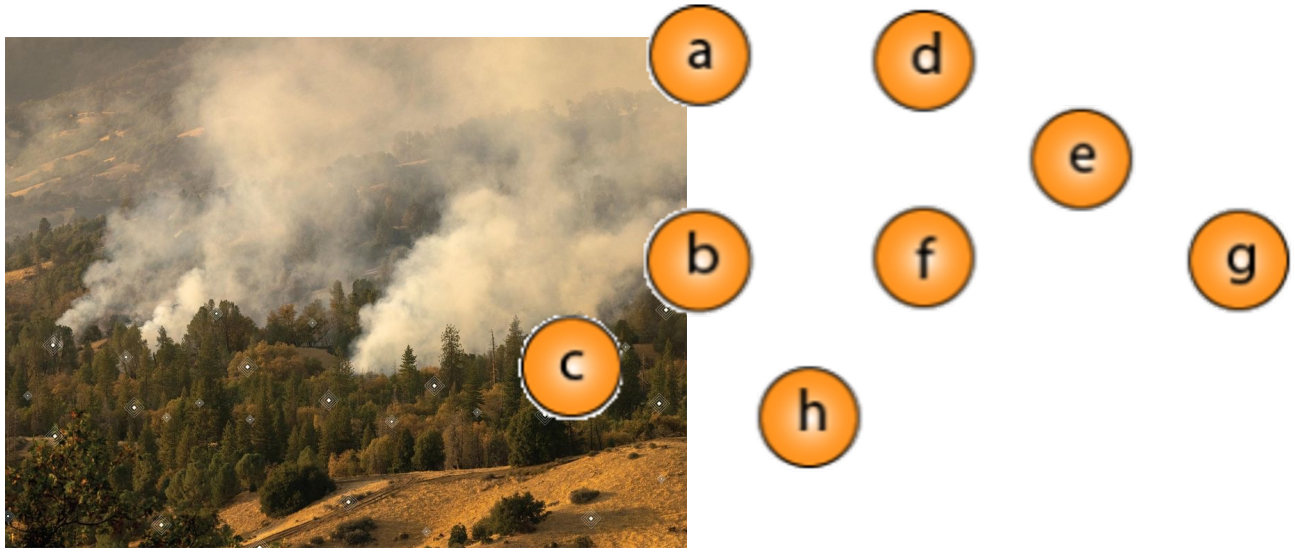
Data Collection of large-scale IoT devices

- Wildfires cause animal deaths, health/psychological problems, environment problems (e.g., air pollution), and economic impact, etc

- The 2019-2020 bushfires in Australia led to the death of at least 33 people and over 3 billion animals

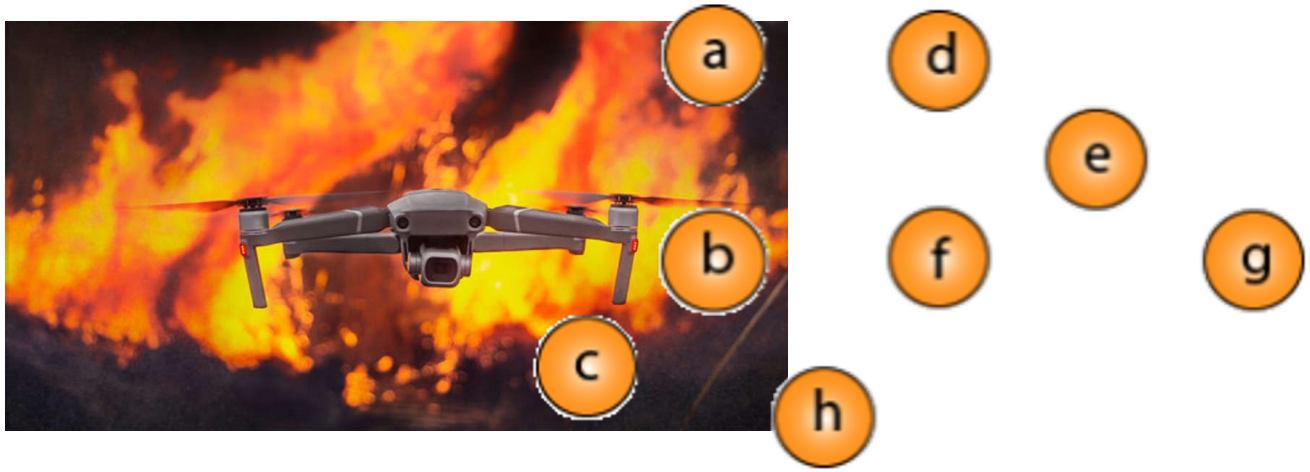
Data Collection of large-scale IoT devices

- Internet-of-Things (IoT) devices are sparsely deployed to monitor Points of Interest (PoIs)
- They may be used to monitor wildfires in a forest



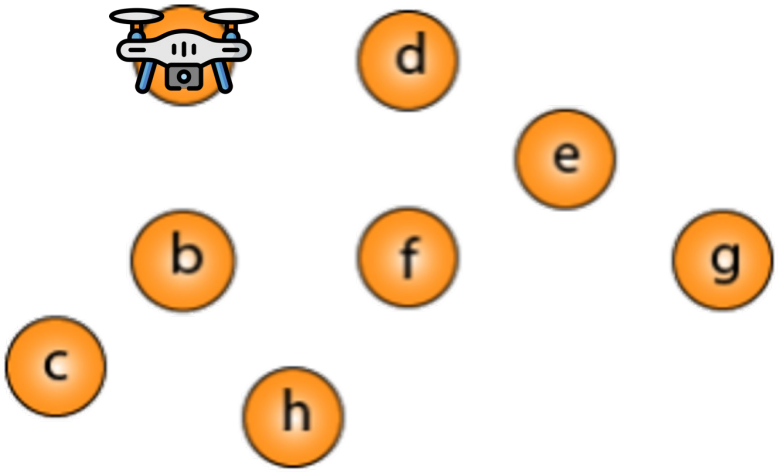
Data Collection of large-scale IoT devices

- IoT devices may not directly transmit or relay their sensing data to a base station
- Deploy multiple UAVs to collect data
- Save the energy consumption of IoT devices



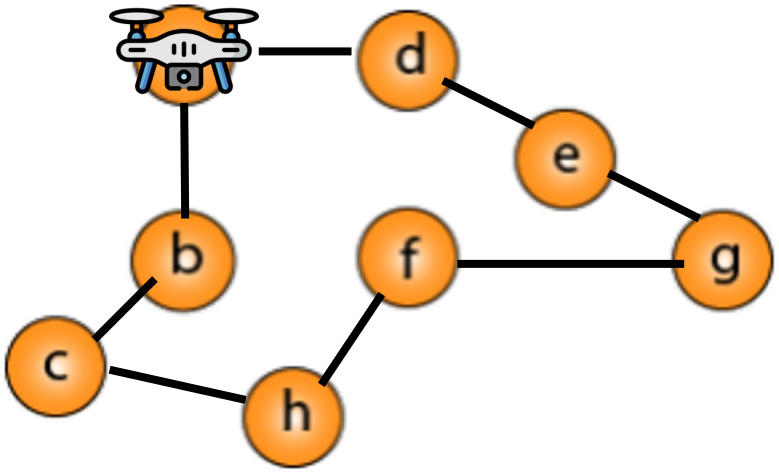
Data Collection of large-scale IoT devices

- A UAV can fly to a location nearby an IoT device to collect its data



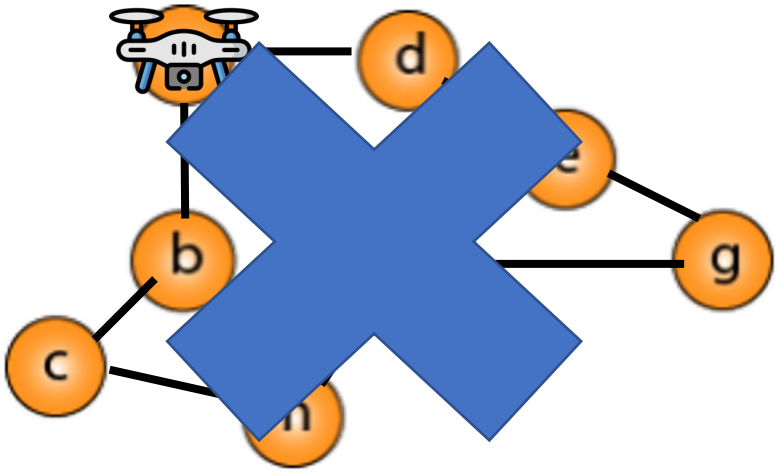
Data Collection of large-scale IoT devices

- A UAV can fly to a location nearby an IoT device to collect its data
- If the flying time of a UAV is unlimited…



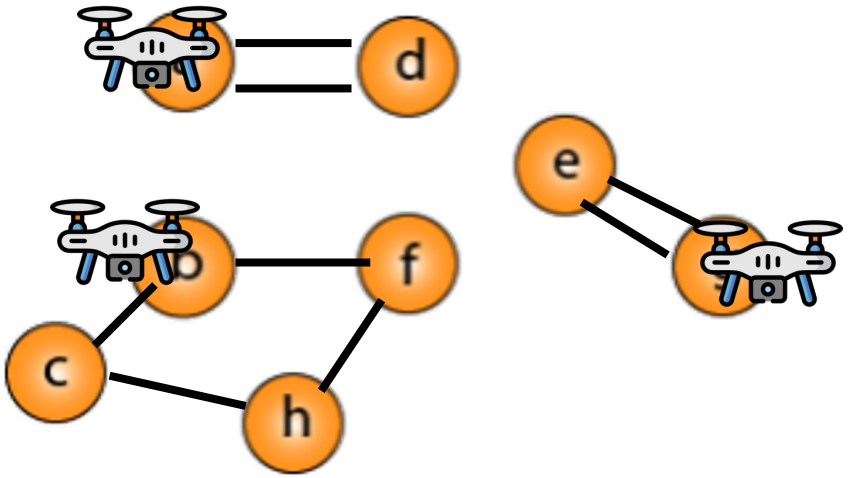
Data Collection of large-scale IoT devices

- The total time spent by any UAV is no greater than a given time threshold (e.g., power limit)



Data Collection of large-scale IoT devices

- The total time spent by any UAV is no greater than a given time threshold (e.g., power limit)
- Therefore, our goal is to minimize the number of UAVs and find their tours



Programming Project #3: Minimum UAV Deployment Problem

- Input:
  - Each IoT device's coordinate
  - A time threshold B to limit a flying tour
- Procedure:
  - Deploy UAVs' tours to span all IoT devices
- Output:
  - The number of UAVs to-be-deployed and their tours
- The grade is inversely proportional to the number of UAVs to-be-deployed

Bad News

- The problem is NP-hard
- We may not always find the optimal solution in polynomial time
- Alternatively, we aim at a near-optimal solution

The Competition

- The grade is inversely proportional to # UAVs to-be-deployed
- Basic: 75 (deadline)
  - A baseline solution (see the following pages)
- Performance ranking (decided after the deadline)
  - [0%, 50%) (bottom): +0
  - [50%, 75%): + 5
  - [75%, 90%): + 9
  - [90%, 95%): + 12
  - [95%, 100%] (top): + 15
- Homework assistant (superb deadline)
  - +10

The Competition

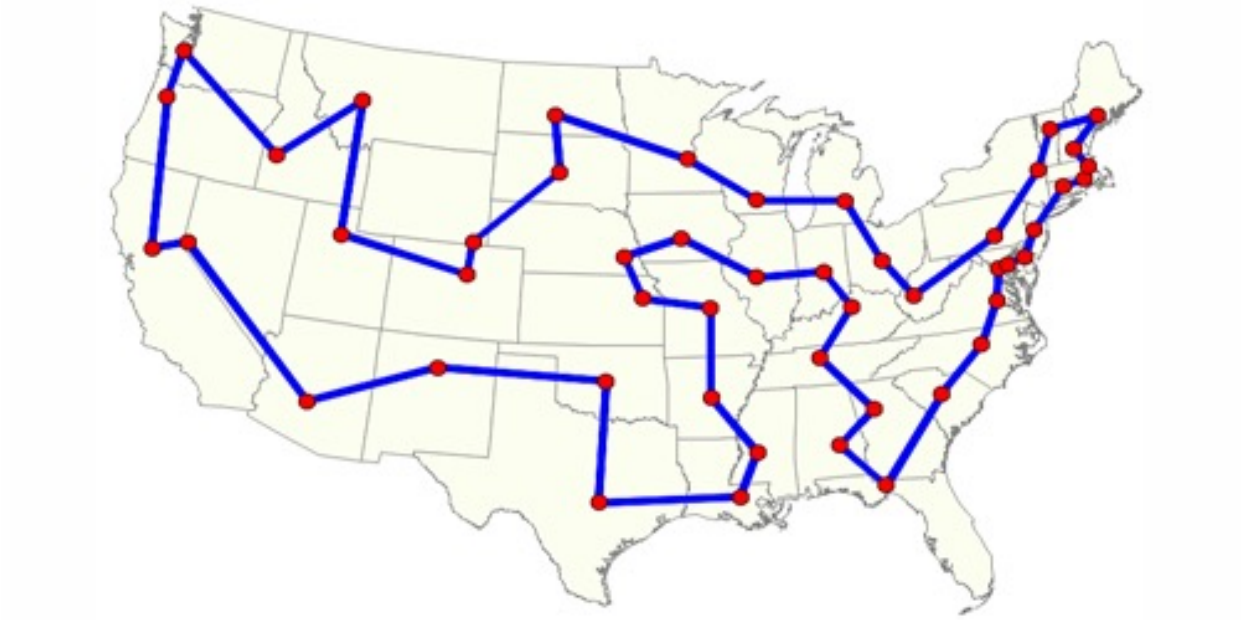
- The grade … # UAVs to-be-deployed
- Basic: 75
  - A baseli… s)
- Performa… deadline)
  - [0%, 50…
  - [50%, 7…
  - [75%, 9…
  - [90%, 9…
  - [95%, 1…
- Homework assistant (superb deadline)
  - +10

The Competition

- The grade … UAVs to-be-deployed
- Basic: 75
  - A baseli…
- Performa…
  - [0%, 50…
  - [50%, 7…
  - [75%, 9…
  - [90%, 9…
  - [95%, 1…
- Homework assistant (superb deadline)
  - +10

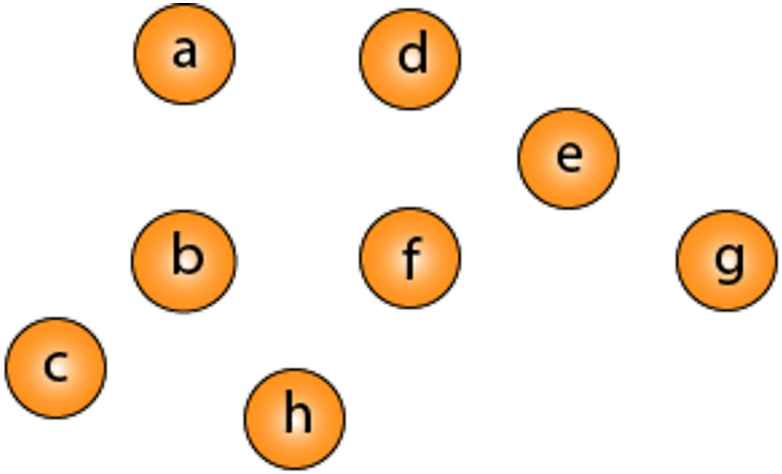
Idea of Baseline Algorithm

- When B is unlimited, the problem is exactly TSP
- Construct a TSP tour (with a 2-approximation)
- Then, divide it into several subtours, each of which has length of at most B

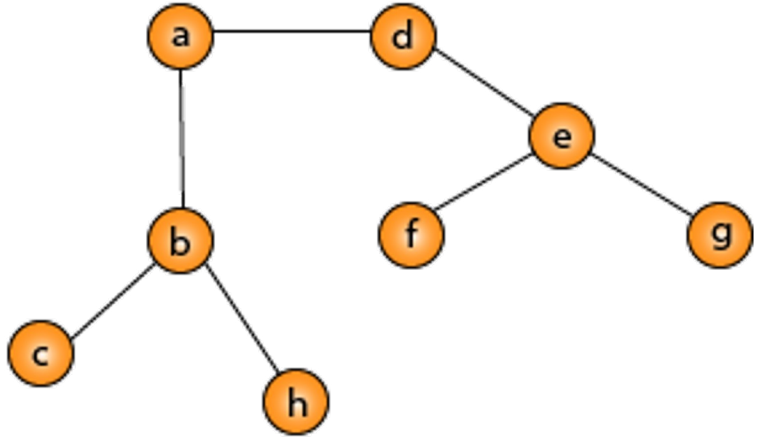


The Baseline Algorithm

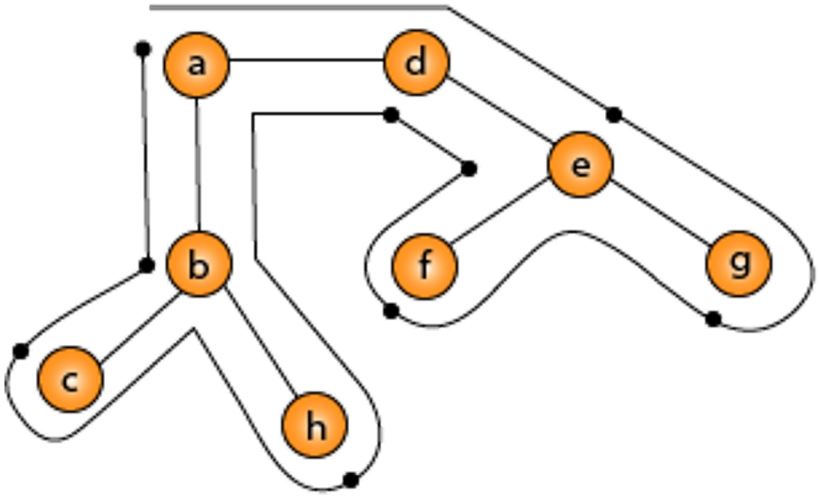
- Construct a Minimum Cost Spanning Tree
  - You can use Kruskal's or Prim's algorithm
  - You don't need to consider tie-breaking (i.e., unique cost)
- Traverse the tree via Depth First Search from the first node (i.e., node 0 or node a) to generate a sequence
  - If there are multiple choices, give the priority to the node with a smaller ID
- Remove the repeated nodes in the sequence except for the end node
- Divide the tour when the path length is greater than $B/2$ (start the tour from node 0 or node a)



Original Graph

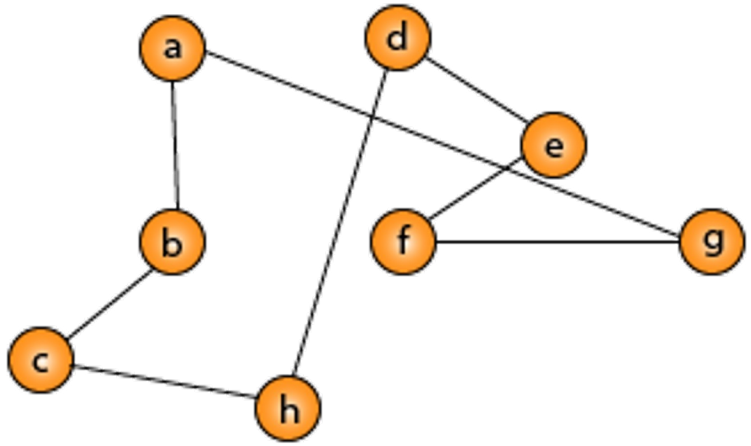


Minimum Cost Spanning Tree



Traverse via DFS from 1st node

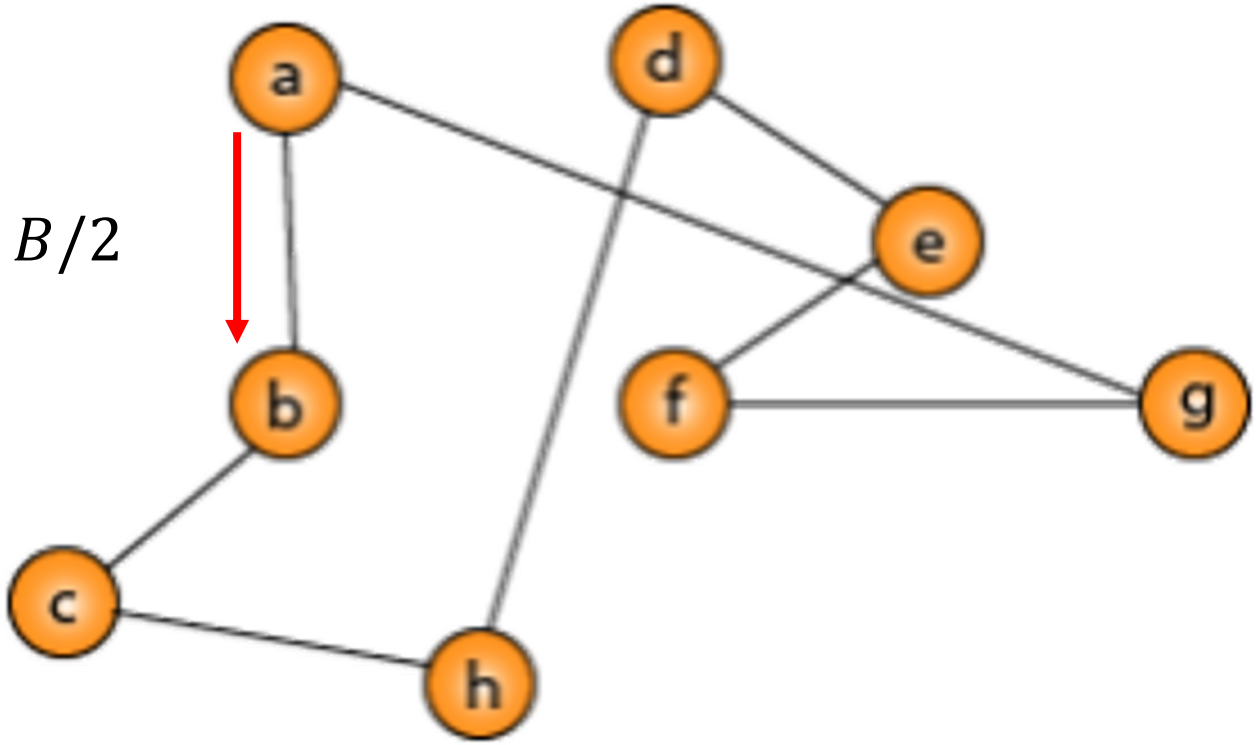
a → b → c → b → h → b→ a → d → e → f → e→ g → e → d → a



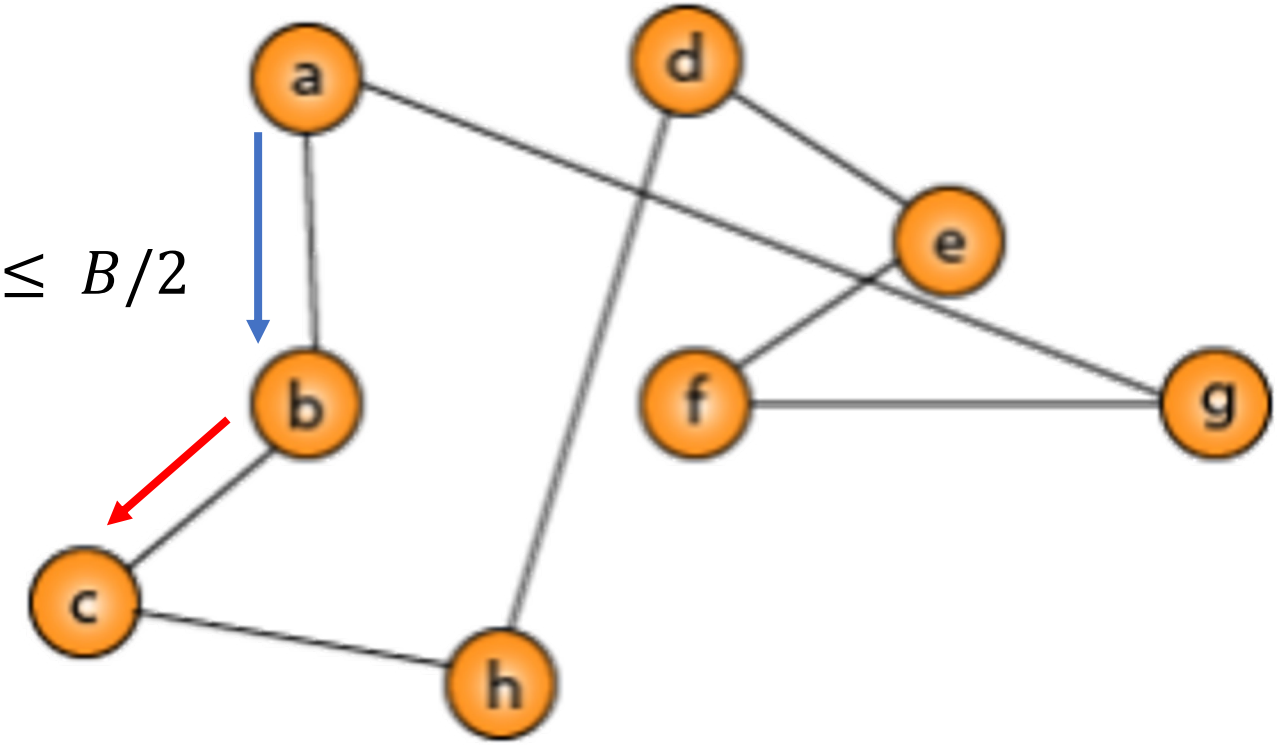
Remove the repeated node

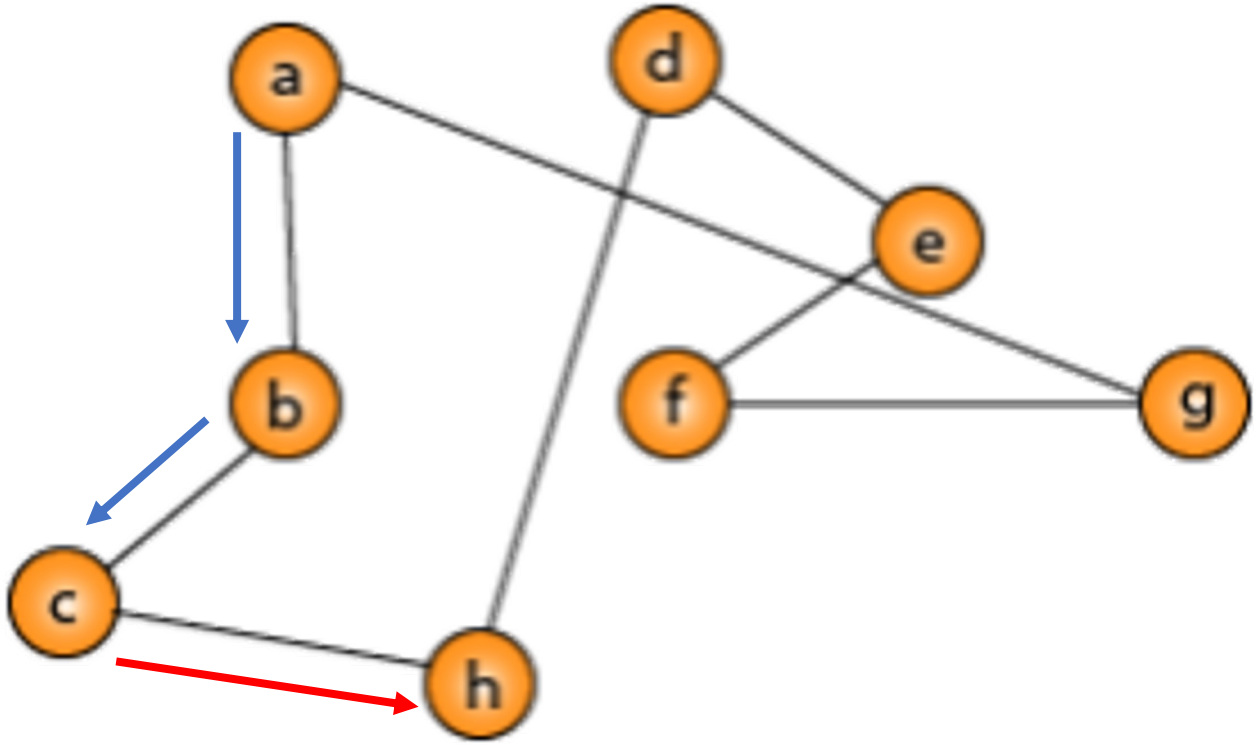
a → b → c → h → d → e → f → g → a

$ab \leq B/2$



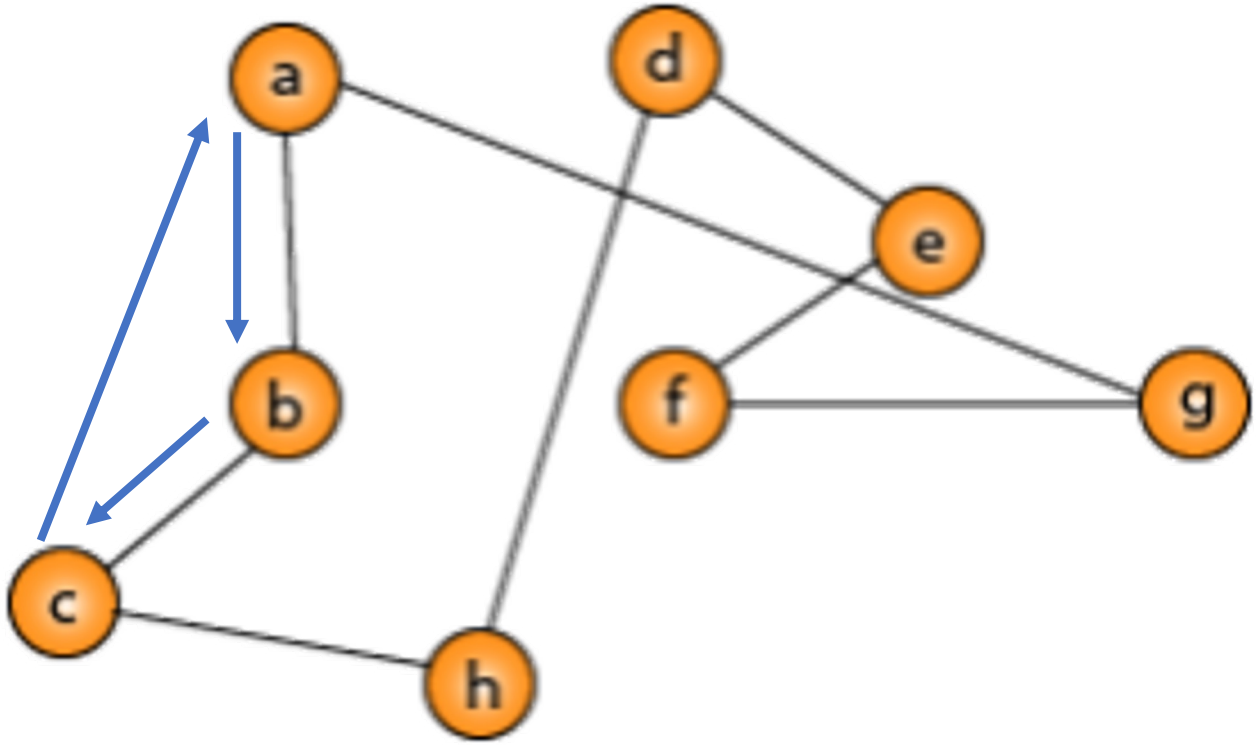
$ab + bc \leq B/2$

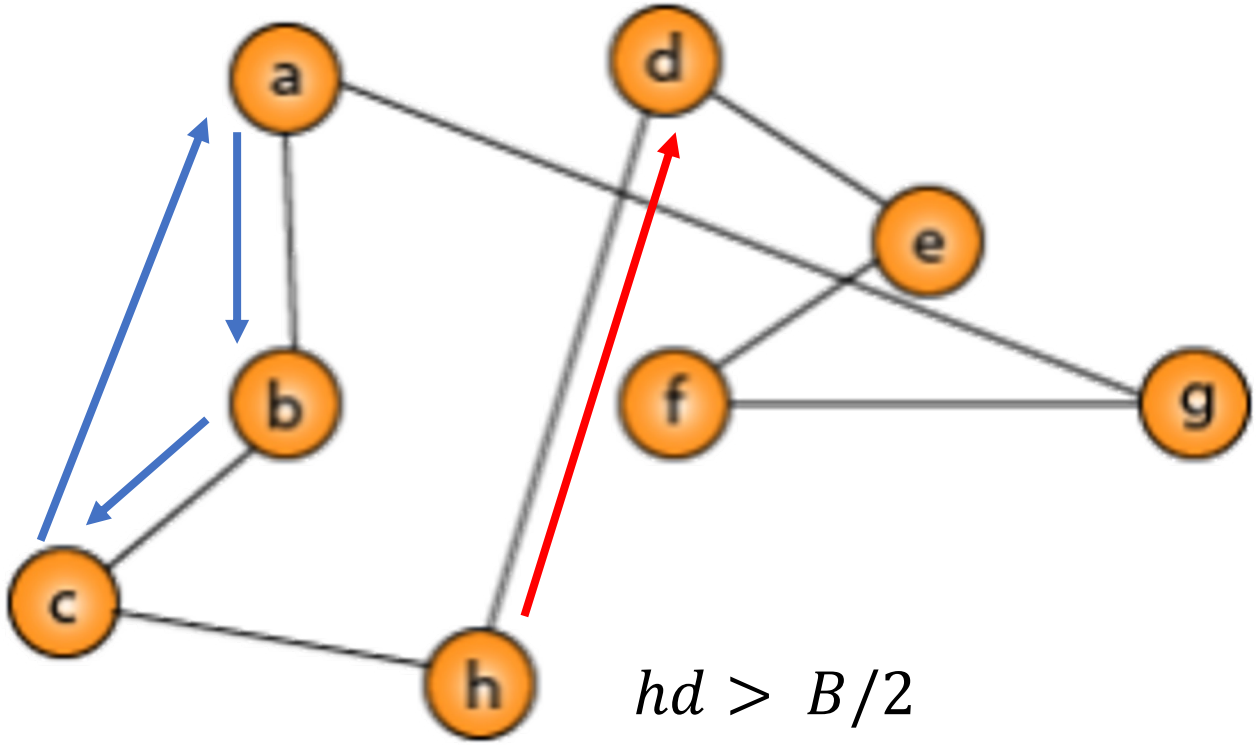




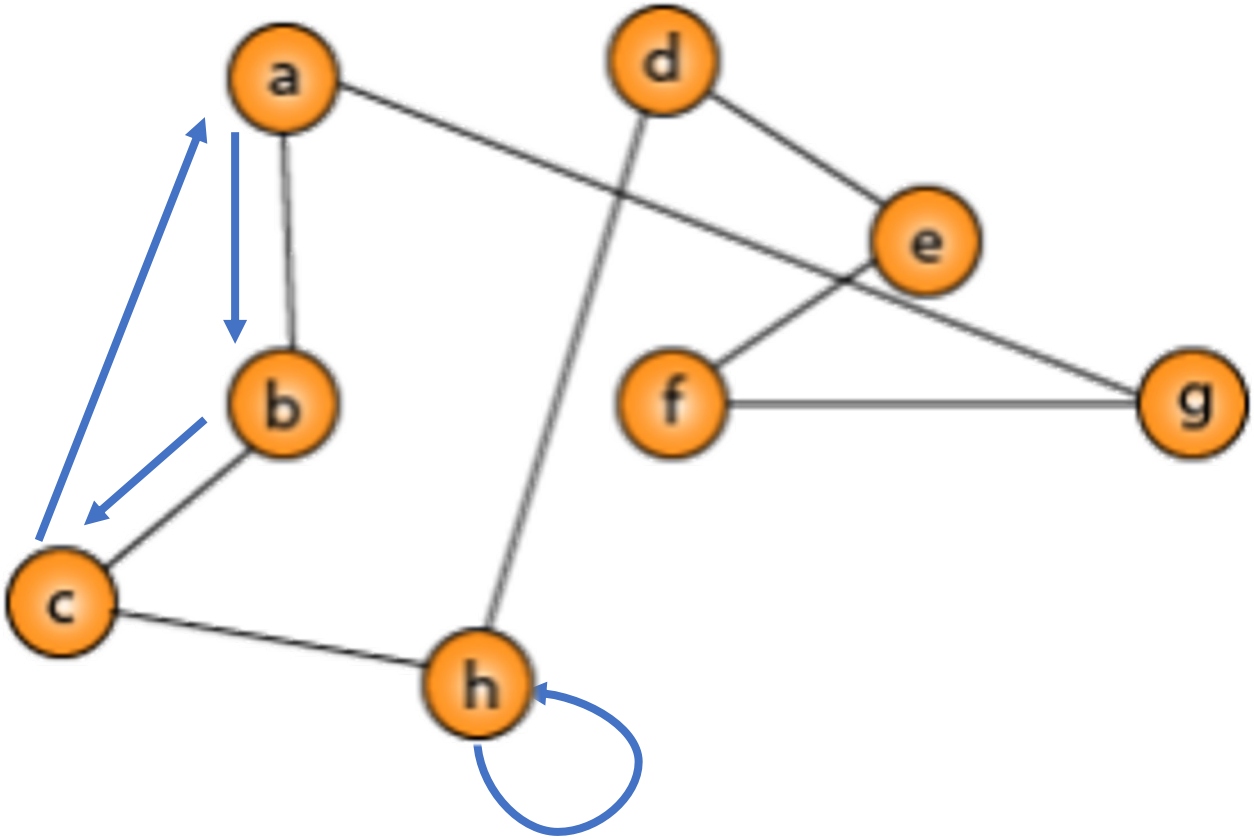
$ab + bc + ch > B/2$

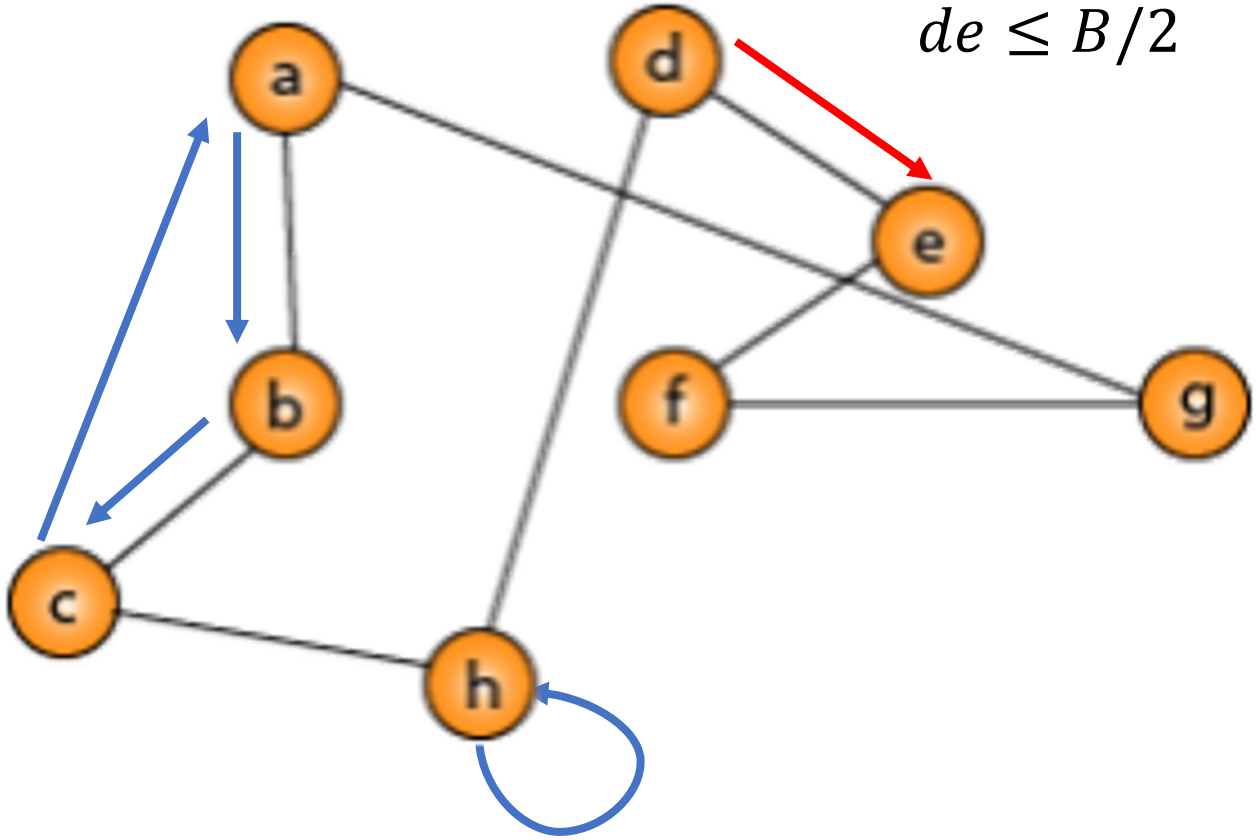
Get 1st tour with a time smaller than B

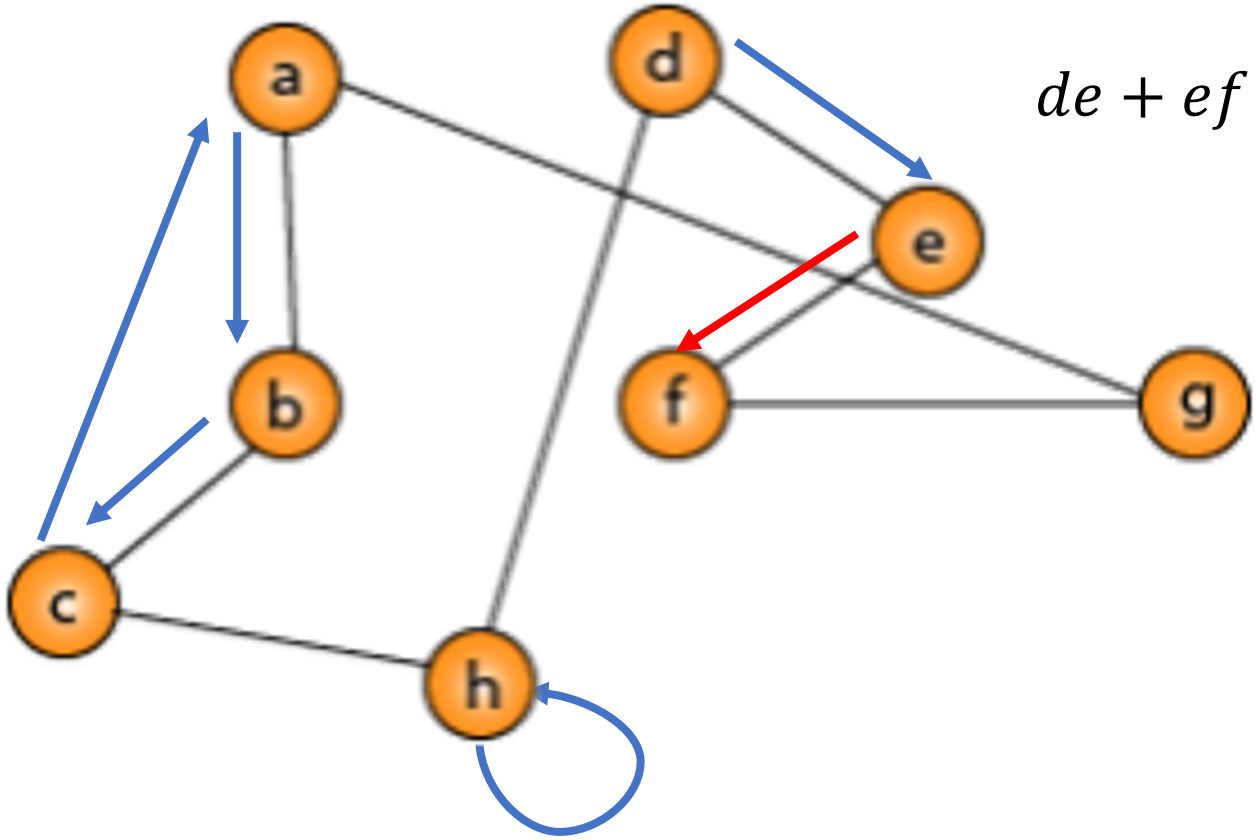



a
d
e
b
f
g
c
h
$hd > B/2$

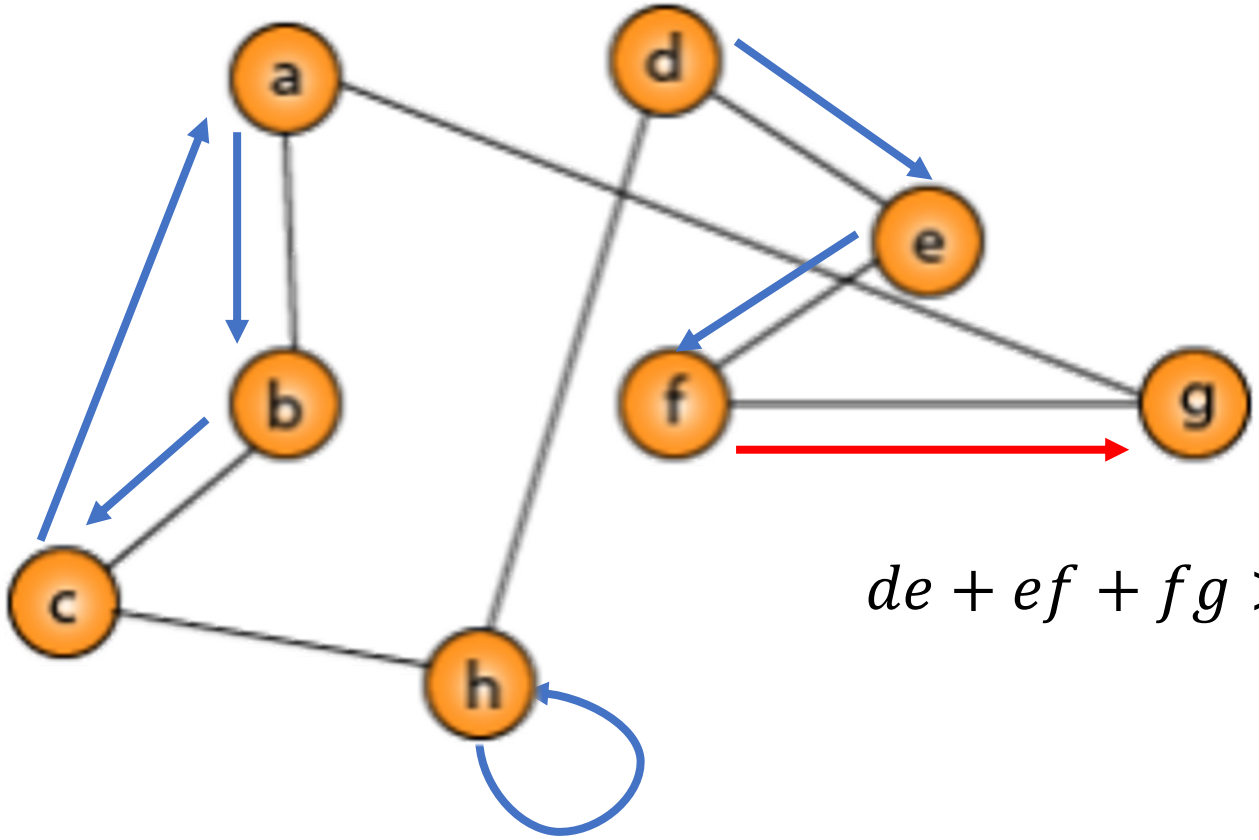
Get 2nd tour with a time smaller than B




$de \leq B/2$
a
d
e
b
f
g
c
h

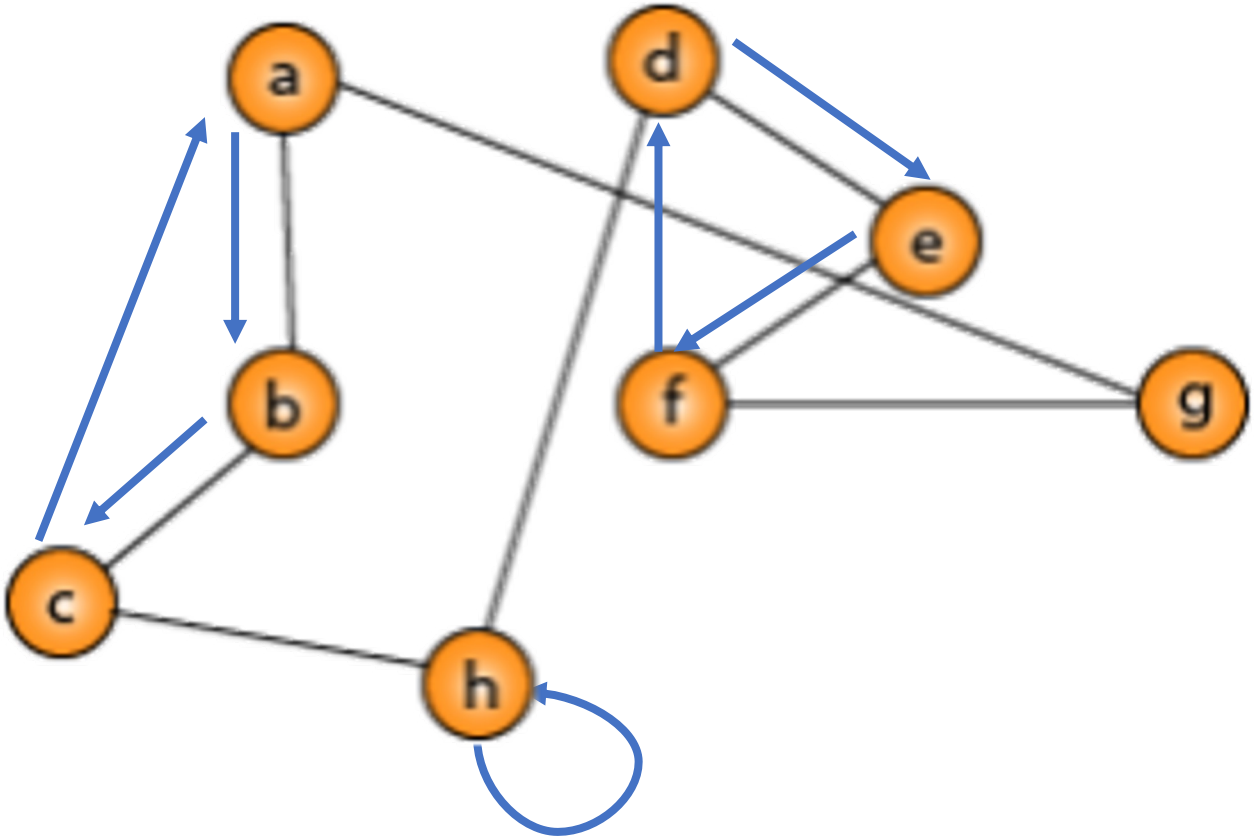


$de + ef \leq B/2$

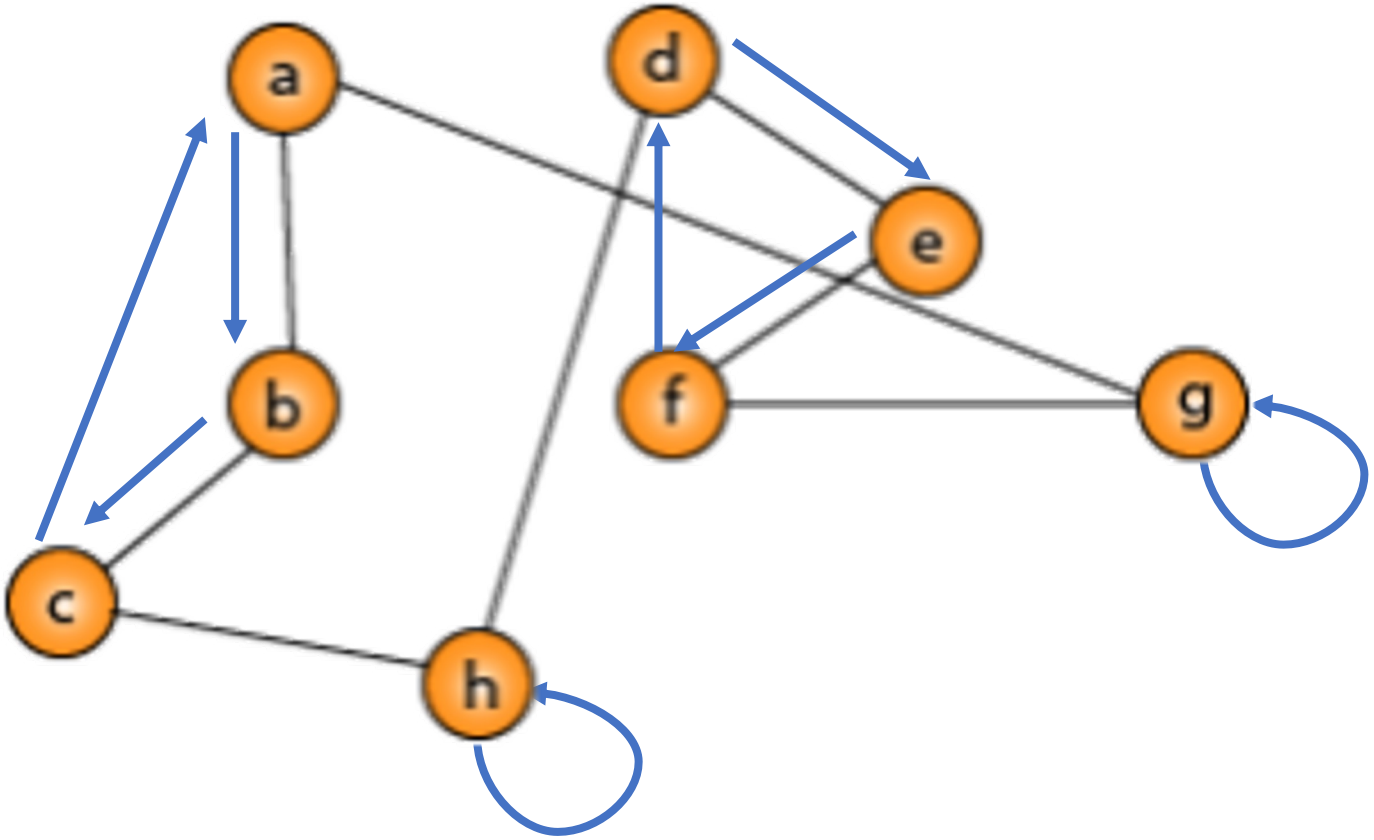


$$de + ef + fg > B/2$$

Get 3rd tour with a time smaller than B



Get 4^{th} tour with a time smaller than B



Idea and Hints of Improvement

- How to improve the solution of the Traveling salesman problem (TSP)
  - The 2-approximation, double-tree algorithm, is cool, but it is not good enough
- How to avoid a long-distance edge
  - Set a threshold? What is a suitable threshold?
- Balance the time complexity and solution quality

Input Sample: use scanf

Format:

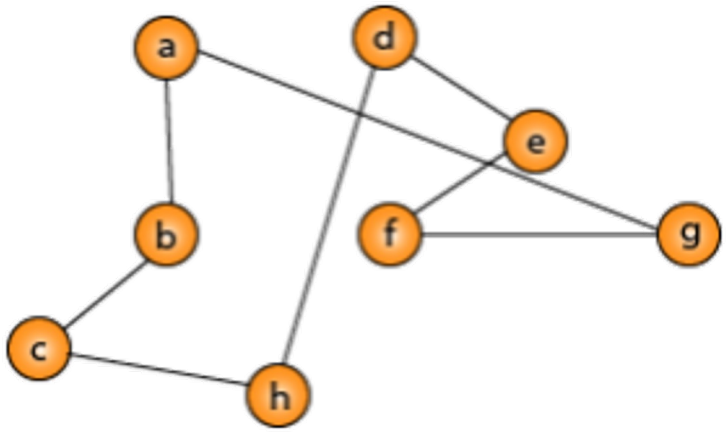
#Nodes B

NodeID CoordinateX CoordinateY

…

Use double

It will be node 4 in the input



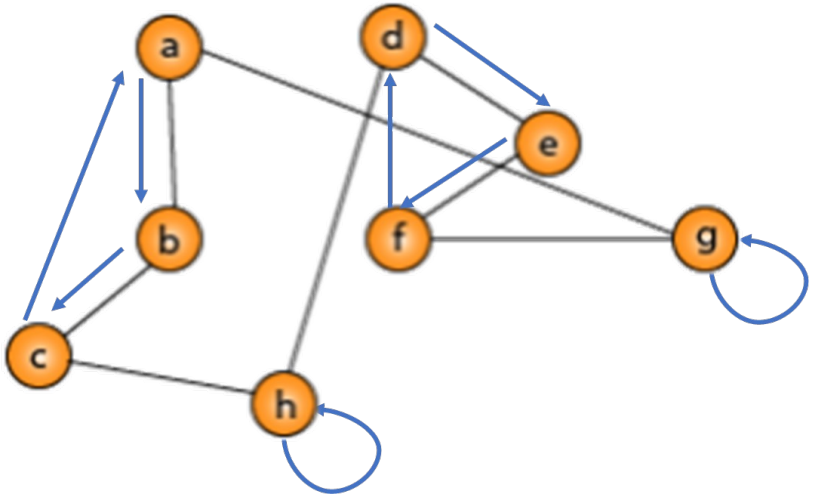
Output Sample: use printf

Format:

#UAVs

UAVID 1st Node 2nd Node 3rd Node … 1st Node

…



Note

- Superb deadline: 12/6 Tue (adjust?)
- Deadline: 12/13 Tue (adjust?)
- Pass the test of our online judge platform
- Submit your code to E-course2
- Demonstrate your code in EA401B or remotely with TA
- **C Source code (i.e., only .c)**
- Show a good programming style

相信你們在做完作業以後