The Shadow Network

Homework 4

CS2351 Data Structures (Spring 2023)

Background

- In Wakunda, a secret spy devised a plan to send top-secret information to their headquarters by using unsuspecting civilians and fellow spies to pass messages.
- Text messages were used as a means to transfer information, with the **distance calculation based on the number of messages exchanged** between contacts.
- These spies always **uses the shortest path for message delivery** and **avoid direct contact with other spies** (excluding the headquarter)

Challenges

- Wakunda intelligence team hired you to **build a network simulation process to analyze and find the shortest-path** in the spy communication network, aiding in counteracting spy activity.
- Your mission has the following tasks:
    1. Build the network using nodes and edges
    2. Determine the shortest-path using **Dijkstra algorithm**
    3. Supplying the sequence of node traversal, the number of messages, and the distance of the shortest path

Network Structure

Nodes / Actor

- The people involved in the network
- 4 types of nodes:
    1. SOURCE: The spy who **initiates the message**.
    2. SPY: Other spies in the network.
    3. CIV: Civilians unwittingly involved.
    4. HQ: Headquarter, the **message's destination**.

Actor ID	Actor Name	Type
1	Anya	SOURCE
2	Naruto	CIV

Network Structure

1 — Total message = 70 — 2

Distance = 14.29

Edge / Distance

- Distance between actors, rounded to nearest 2 decimal points (use **#include<cmath>**)

$$Distance = \frac{1000}{Total\ Message}$$

$$Distance = \frac{1000}{70} = 14.29$$

Actor ID	Actor Name	Type
1	Anya	SOURCE
2	Naruto	CIV

Actor ID 1	Actor ID 2	Total Message
2	1	70

Rule: Connections between Types

Allowed Connections

- **SOURCE ----- CIV** : Less-suspicious
- **SPY ----- CIV** : Less-suspicious
- **CIV ----- CIV** : Less-suspicious
- **SOURCE ----- HQ** : Source (spy) have access to HQ
- **SPY ----- HQ** : Spy have access to HQ

Unallowed Connections

- **SOURCE --X-- SPY** : Suspicious
- **SPY --X-- SPY** : Suspicious
- **CIV --X-- HQ** : Civilian have no access to HQ

Rule: Connections between Types

Allowed Connections

- SOURCE ----- CIV
- SPY ----- CIV
- CIV ----- CIV
- SOURCE ----- HQ
- SPY ----- HQ

Unallowed Connections

- SOURCE --X-- SPY
- SPY --X-- SPY
- CIV --X-- HQ

SOURCE ----- SPY ----- SPY -----CIV ----- CIV ----- HQ

SOURCE ----- CIV ----- CIV ----- SPY ----- HQ

Rule: Connections between Types

Allowed Connections

- SOURCE ----- CIV
- SPY ----- CIV
- CIV ----- CIV
- SOURCE ----- HQ
- SPY ----- HQ

Unallowed Connections

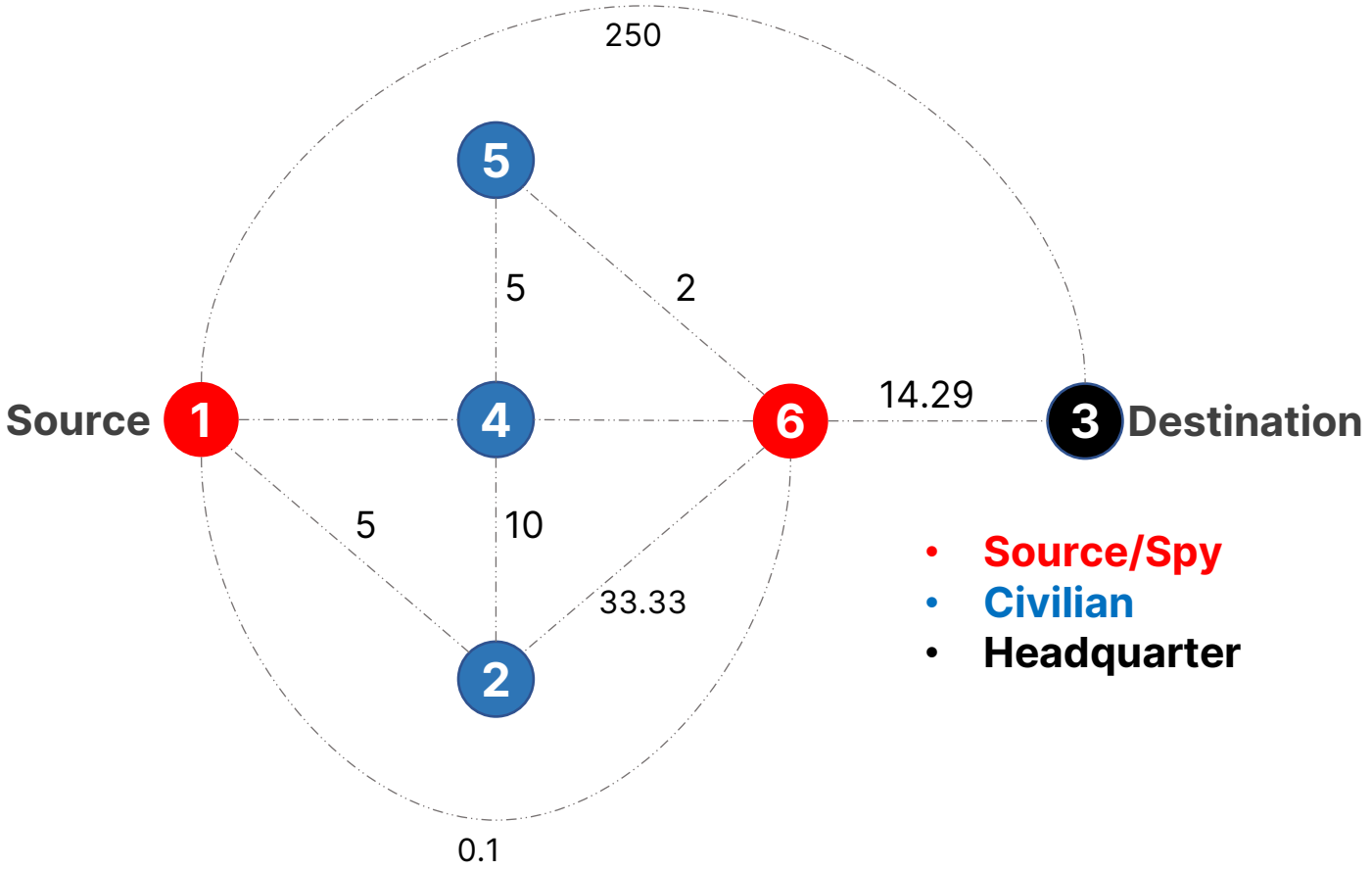
- SOURCE --X-- SPY
- SPY --X-- SPY
- CIV --X-- HQ

SOURCE --X-- SPY --X-- SPY -----CIV ----- CIV --X-- HQ

SOURCE ----- CIV ----- CIV ----- SPY ----- HQ

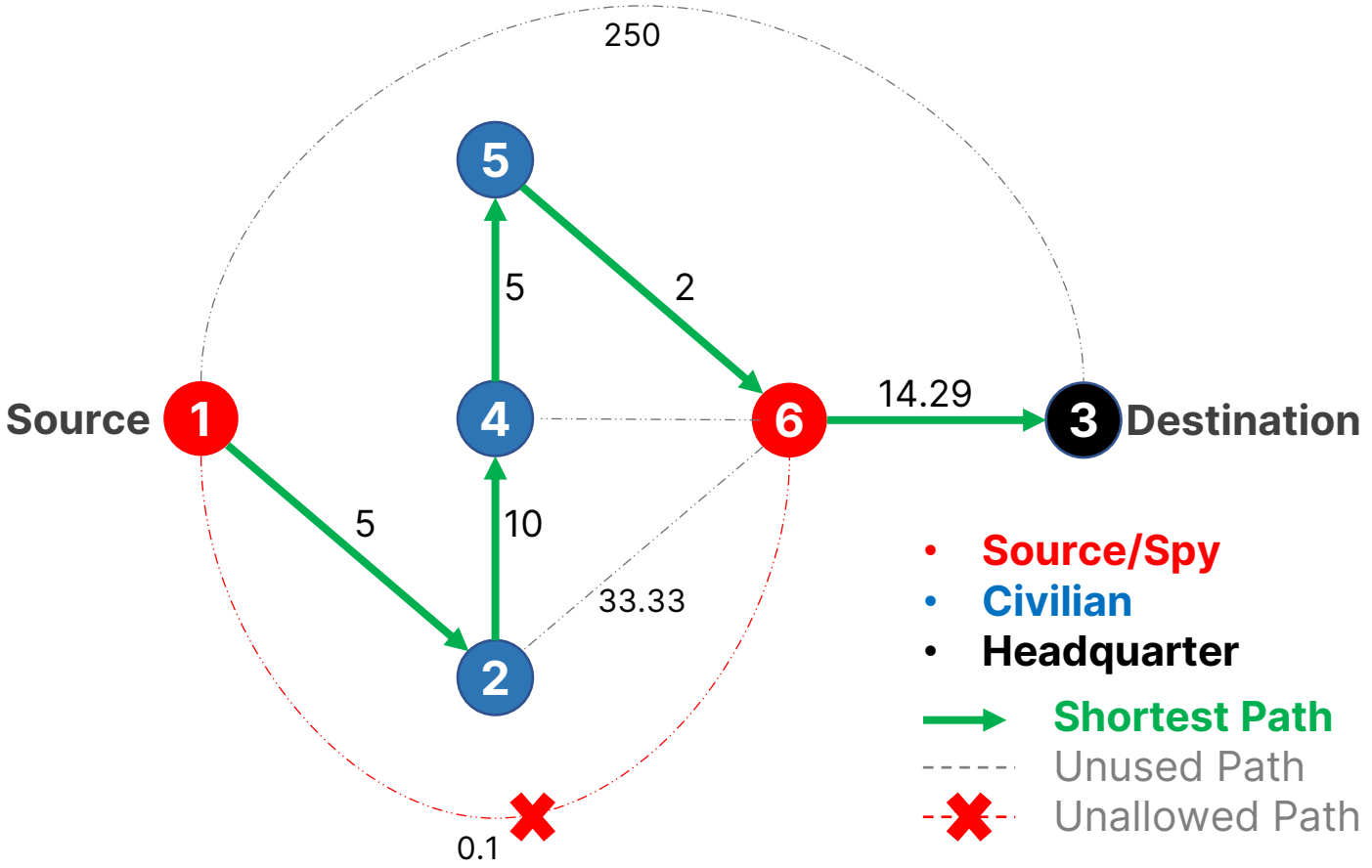
Network Analysis

- Determine the shortest-path using **Dijkstra algorithm**



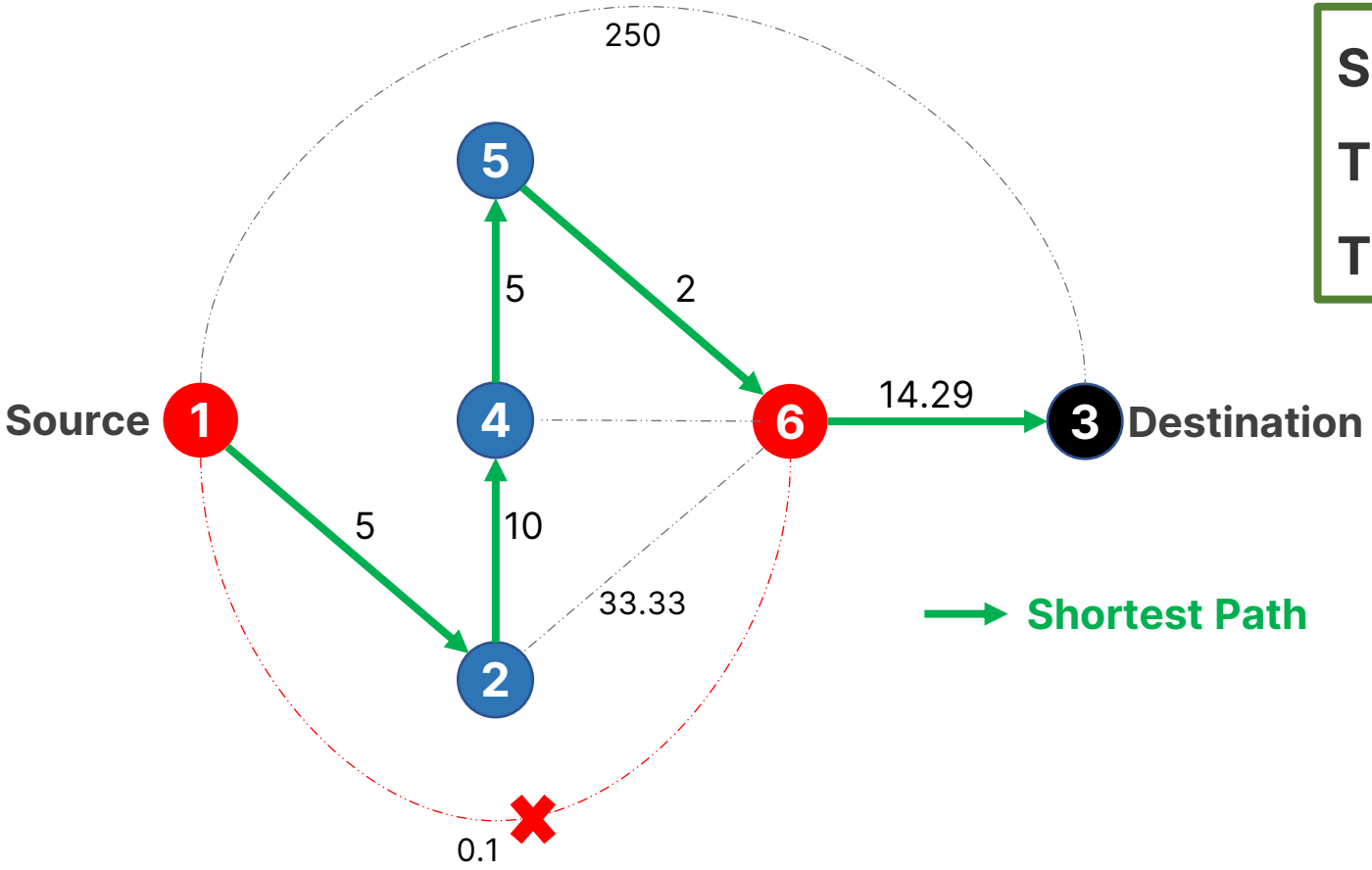
Network Analysis

- Determine the shortest-path using **Dijkstra algorithm**



Network Analysis

- Supplying the **sequence of node traversal, the number of messages, and the distance of the shortest path**



Sequence: 1 → 2 → 4 → 5 → 6 → 3

Total Message: 1070

Total Distance: 36.29

Inputs & Outputs

Inputs:

1. **Insert Node**
2. **Insert Edge**
3. **Analyze Network**

Outputs:

Shortest-path information:

1. **Sequence of Traversal**
2. **Total Number of Messages**
3. **Total Distance**

Input 1 : Insert Node

```
INSERT <nodetype> <name>
```

- **<nodetype>** values: SOURCE, SPY, CIV, HQ
- **<name>** : name of the actor always consist of 1 word
- Assign each a unique ID number based on insertion order, irrespective of node type (SOURCE, SPY, CIV, or HQ), starting from 1.

Input 1 : Insert Node

Assign each a unique ID number based on insertion order, irrespective of node type (SOURCE, SPY, CIV, or HQ), starting from 1.

```
INSERT SOURCE Anya
INSERT SPY Ben
INSERT HQ Charlie
INSERT CIV Dave
```

Actor ID	Actor Name	Type
1	Anya	SOURCE
2	Ben	SPY
3	Charlie	HQ
4	Dave	CIV

Input 2 : Insert Edge

```
INSERT_EDGE <id_number1> <id_number2> <total_message>
```

It is guaranteed that:

- 1 <= **<total_message>** <= 10^5
- 1 <= **<id_number1>** <= total nodes
- 1 <= **<id_number2>** <= total nodes
- **<id_number1>** != **<id_number2>**
- Edges consisting of the same pair will not be inserted more than once

Input 2 : Insert Edge

Actor ID	Actor Name	Type
1	Anya	SOURCE
2	Naruto	CIV

```
INSERT_EDGE 1 2 70
```

is equivalent with

```
INSERT_EDGE 2 1 70
```

1

Total message = 70

Distance = 14.29

2

Input 3 : Analyze

ANALYZE

- Run Dijkstra algorithm to get the shortest path
- Print the output (the sequence of node traversal, the number of messages, and the distance of the shortest path)
- End/terminate the program

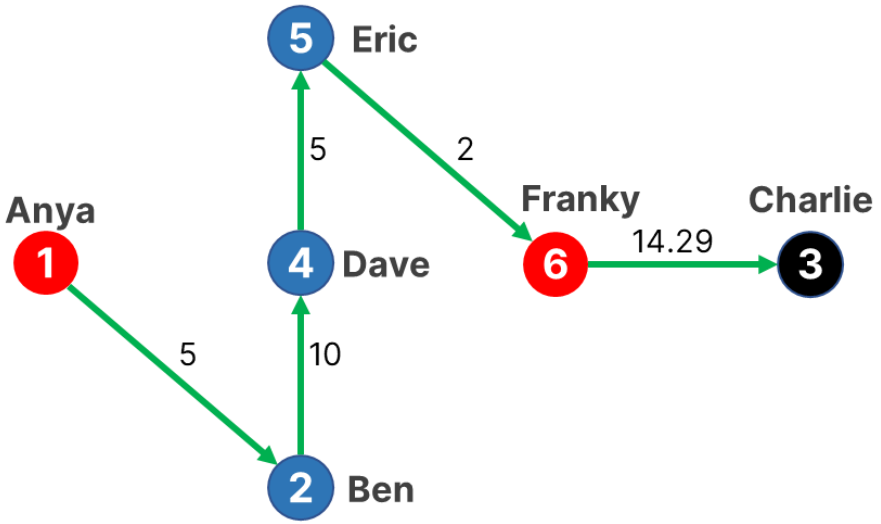
Outputs

```
<order of contact’s name in shortest path>
<total message in shortest path>
<total distance in shortest path>
<new line>
```

To avoid presentation error:

- No whitespace in the end of each line
- Provide a new line after output the **<total distance in shortest path>**

Outputs



Actor ID	Actor Name	Actor Type
1	Anya	SOURCE
2	Ben	CIV
3	Charlie	HQ
4	Dave	CIV
5	Eric	CIV
6	Franky	SPY

Order of Contacts	**Anya -> Ben -> Dave -> Eric -> Franky -> Charlie**
Total Messages	**1070**
Total Distance	**36.29**

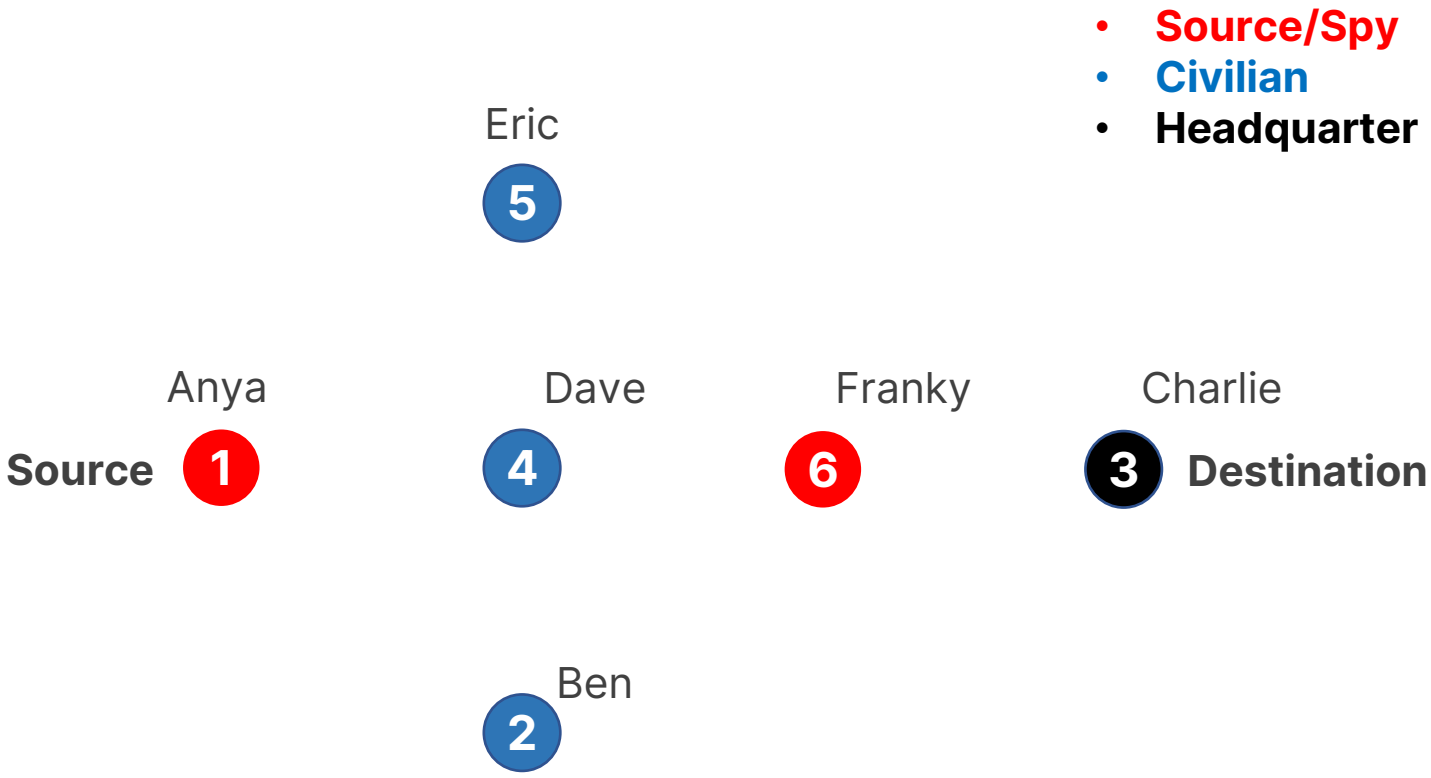
Note: There is a space before and after the "->" symbol

Demo : 1 / 5

```
INSERT SOURCE Anya
INSERT CIV Ben
INSERT HQ Charlie
INSERT CIV Dave
INSERT CIV Eric
INSERT SPY Franky
INSERT_EDGE 1 2 200
INSERT_EDGE 1 3 4
INSERT_EDGE 6 1 10000
INSERT_EDGE 2 4 100
INSERT_EDGE 2 6 30
INSERT_EDGE 4 5 200
INSERT_EDGE 6 4 80
INSERT_EDGE 5 6 500
INSERT_EDGE 6 3 70
ANALYZE
```

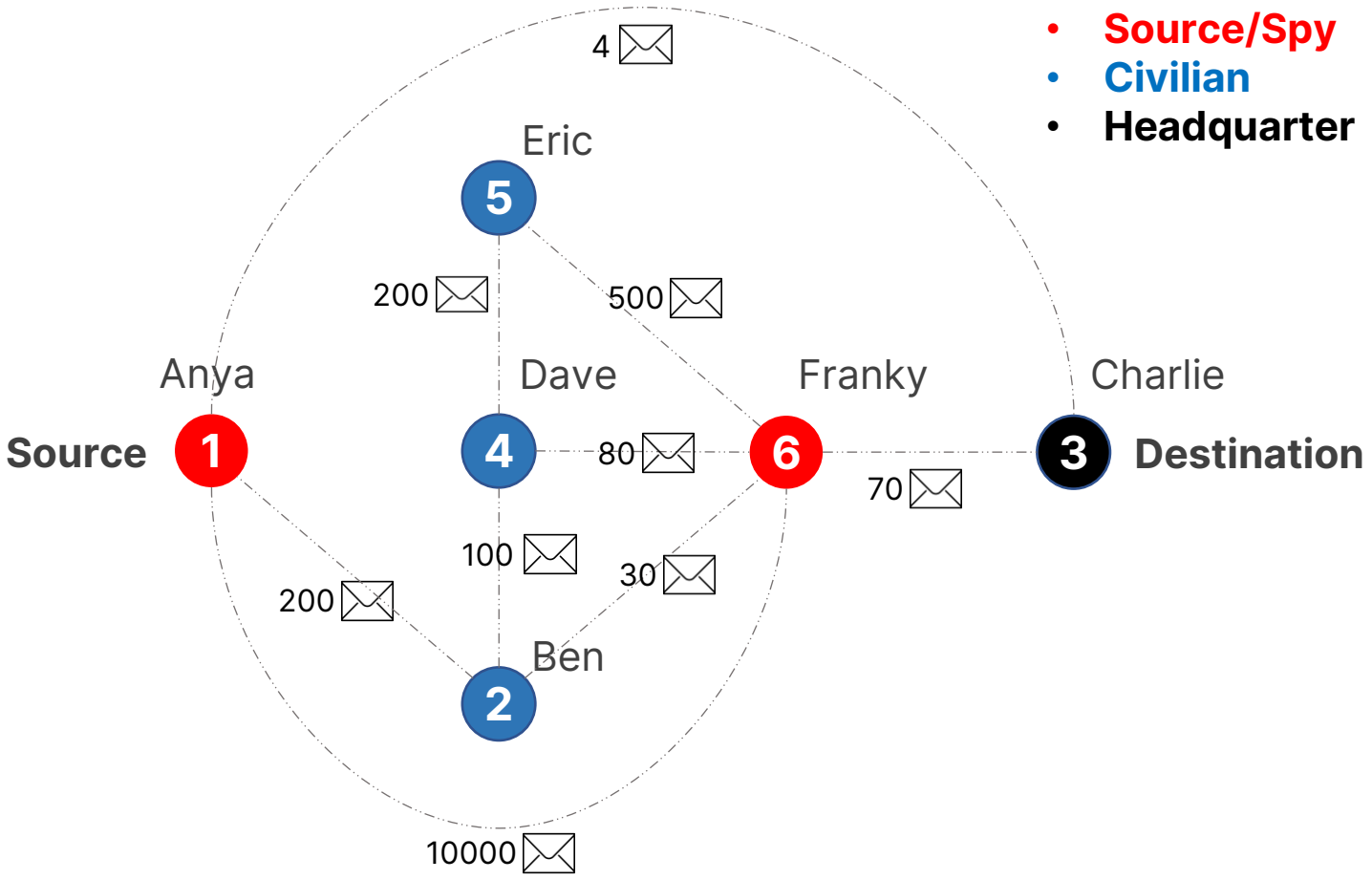
Demo : 2 / 5

```
INSERT SOURCE Anya
INSERT CIV Ben
INSERT HQ Charlie
INSERT CIV Dave
INSERT CIV Eric
INSERT SPY Franky
INSERT_EDGE 1 2 200
INSERT_EDGE 1 3 4
INSERT_EDGE 6 1 10000
INSERT_EDGE 2 4 100
INSERT_EDGE 2 6 30
INSERT_EDGE 4 5 200
INSERT_EDGE 6 4 80
INSERT_EDGE 5 6 500
INSERT_EDGE 6 3 70
ANALYZE
```



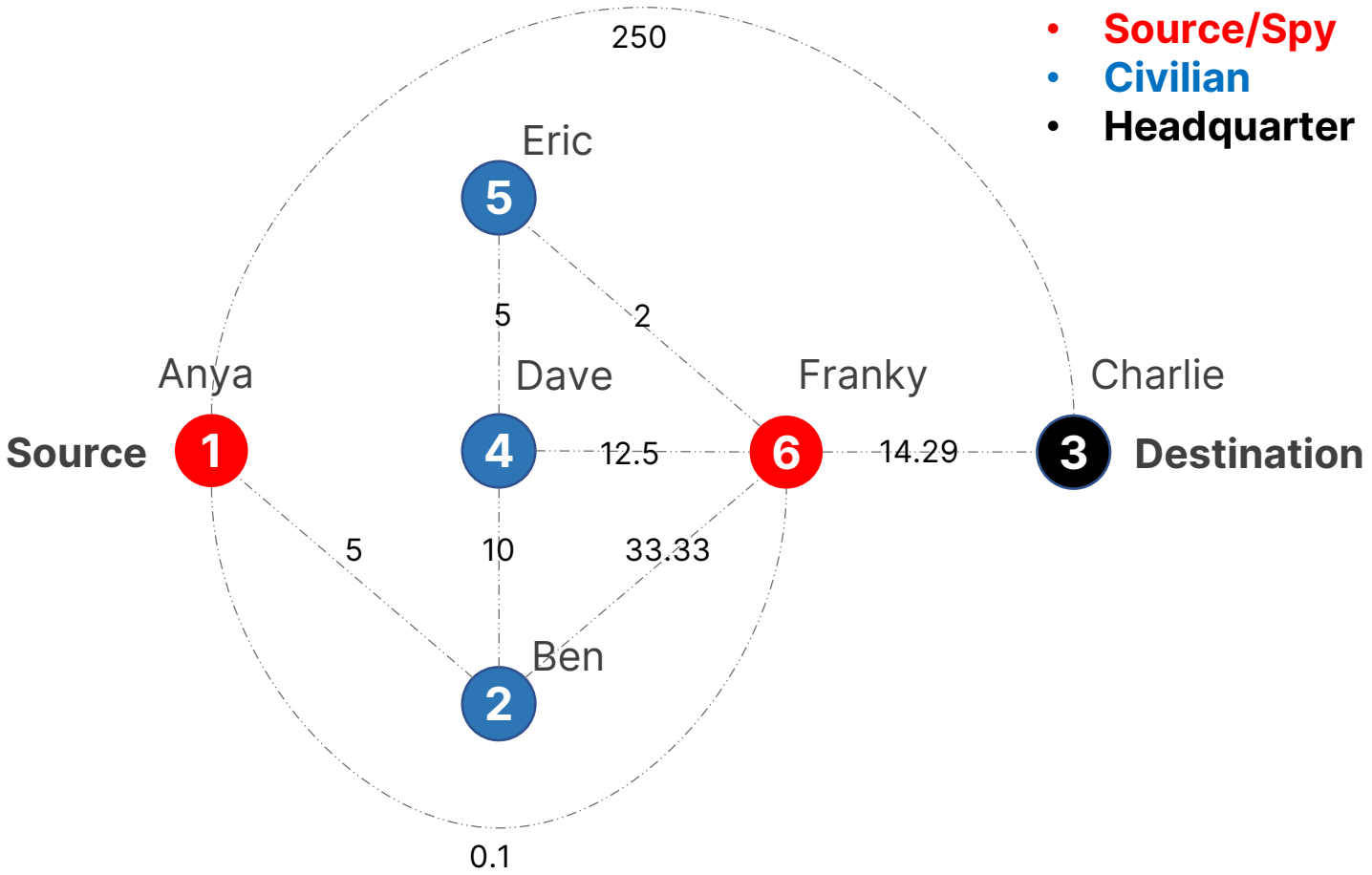
Demo : 3 / 5

```
INSERT SOURCE Anya
INSERT CIV Ben
INSERT HQ Charlie
INSERT CIV Dave
INSERT CIV Eric
INSERT SPY Franky
INSERT_EDGE 1 2 200
INSERT_EDGE 1 3 4
INSERT_EDGE 6 1 10000
INSERT_EDGE 2 4 100
INSERT_EDGE 2 6 30
INSERT_EDGE 4 5 200
INSERT_EDGE 6 4 80
INSERT_EDGE 5 6 500
INSERT_EDGE 6 3 70
ANALYZE
```



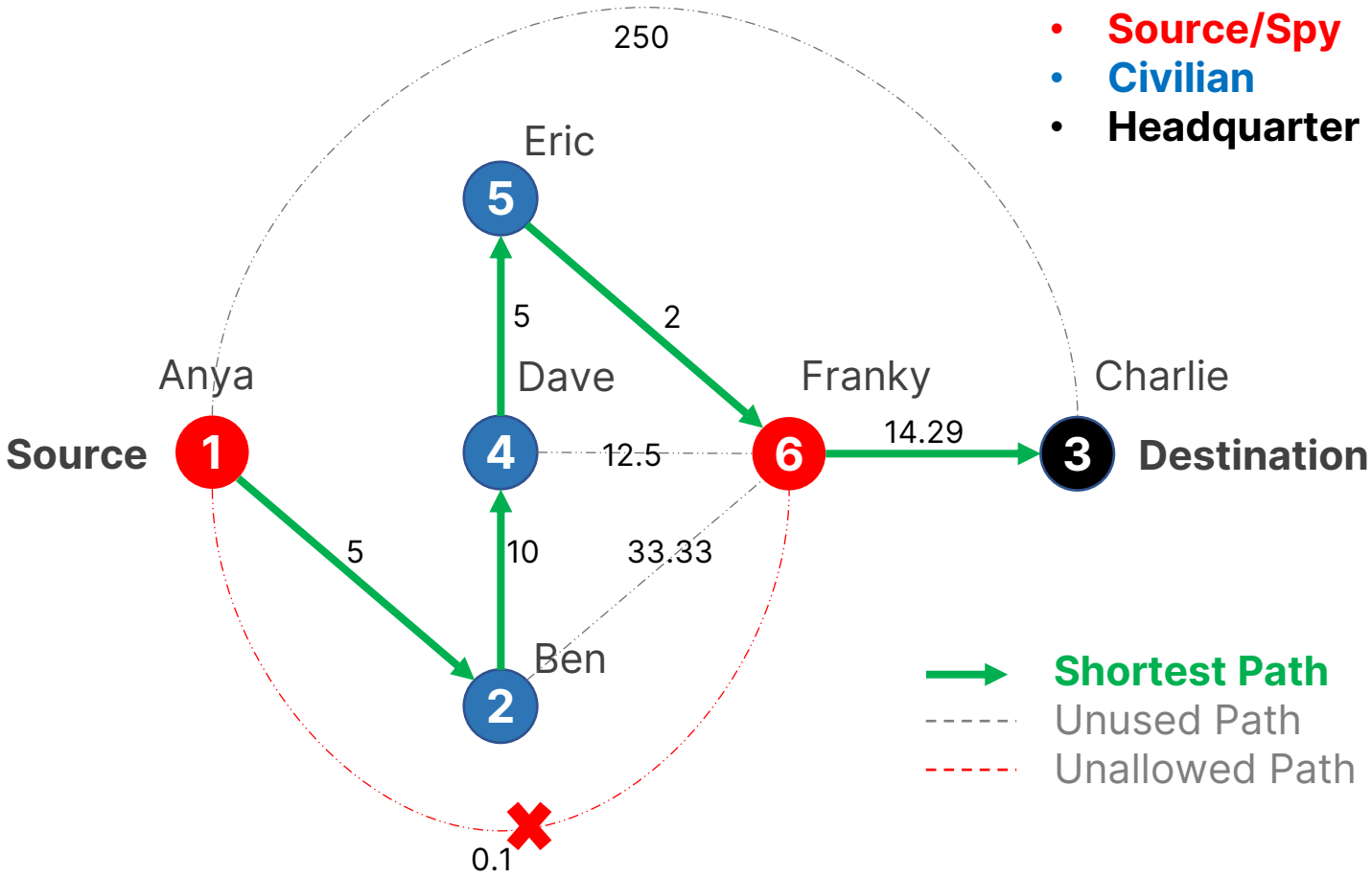
Demo : 3 / 5

```
INSERT SOURCE Anya
INSERT CIV Ben
INSERT HQ Charlie
INSERT CIV Dave
INSERT CIV Eric
INSERT SPY Franky
INSERT_EDGE 1 2 200
INSERT_EDGE 1 3 4
INSERT_EDGE 6 1 10000
INSERT_EDGE 2 4 100
INSERT_EDGE 2 6 30
INSERT_EDGE 4 5 200
INSERT_EDGE 6 4 80
INSERT_EDGE 5 6 500
INSERT_EDGE 6 3 70
ANALYZE
```



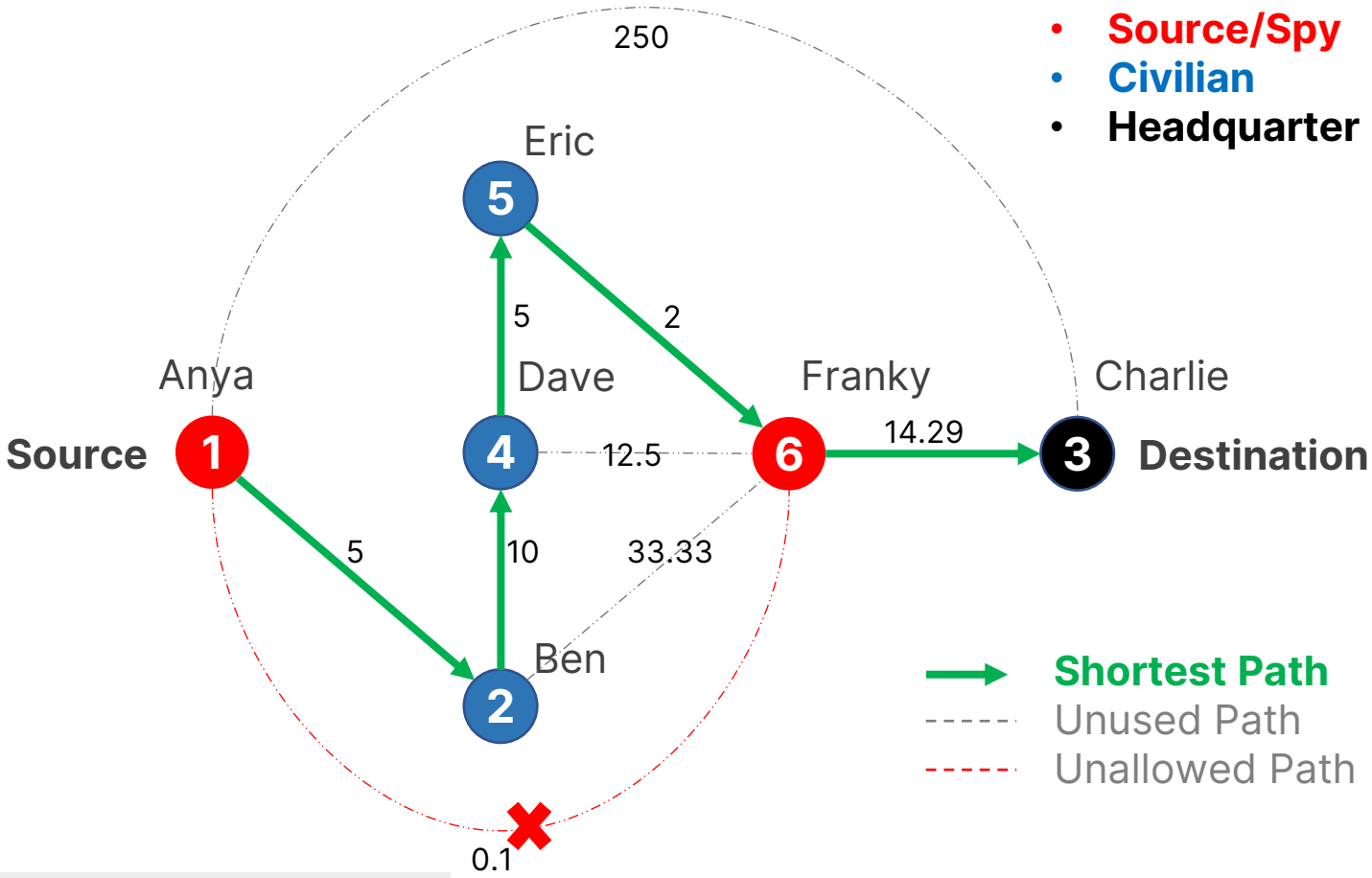
Demo : 4 / 5

```
INSERT SOURCE Anya
INSERT CIV Ben
INSERT HQ Charlie
INSERT CIV Dave
INSERT CIV Eric
INSERT SPY Franky
INSERT_EDGE 1 2 200
INSERT_EDGE 1 3 4
INSERT_EDGE 6 1 10000
INSERT_EDGE 2 4 100
INSERT_EDGE 2 6 30
INSERT_EDGE 4 5 200
INSERT_EDGE 6 4 80
INSERT_EDGE 5 6 500
INSERT_EDGE 6 3 70
ANALYZE
```



Demo : 5 / 5

```
INSERT SOURCE Anya
INSERT CIV Ben
INSERT HQ Charlie
INSERT CIV Dave
INSERT CIV Eric
INSERT SPY Franky
INSERT_EDGE 1 2 200
INSERT_EDGE 1 3 4
INSERT_EDGE 6 1 10000
INSERT_EDGE 2 4 100
INSERT_EDGE 2 6 30
INSERT_EDGE 4 5 200
INSERT_EDGE 6 4 80
INSERT_EDGE 5 6 500
INSERT_EDGE 6 3 70
ANALYZE
```



Output

```
Anya -> Ben -> Dave -> Eric -> Franky -> Charlie
1070
36.29
```

Assumption

- The shadow network is an undirected graph
- The source is always a spy (they can’t communicate to spy, but can communicate to civilian & headquarter)
- The destination is always a headquarter
- There is only one source and one headquarters in each testcases
- At least one spy (including source), one civilian and one headquarter in each testcases
- The insertion of the edge always happened once for each unique pair of nodes
- If there is no message between two nodes, you can assume the distance is infinite
- It’s possible that multiple sequences of nodes (path) will result in the shortest path, read the “Network Analysis” part to find the instructions for selecting the path
- The shortest path between the source and headquarter in every test case never consists of an infinite distance.

Rule

- No plagiarism
- The code should be written in C/C++
- You’re allowed to use the following library & namespace (you might receive a score deduction if you’re using the other libraries):

```
#include<iostream>
#include<vector>
#include<stack>
#include<queue>
#include<list>
#include<string.h>
#include<cmath>
using namespace std;
```

Tips

- Use classes to represent the objects
- Use Dijkstra Algorithm to find the shortest path
- Break down the problem, especially in the Dijkstra Algorithm (divide & conquer)
- If there’s no message between actors, you can assume their distance is infinite
- Testcase 1 is the sample Input & Output
- Please contact the TA Edwin (any channel, Teams chat for fast response) if you have any queries (Please avoid do this during weekend QAQ)

Multiple Options of Shortest Path

Generally, it should be:

- ~~Select the one which involves most message~~
- ~~If it's the same, select the one which involves less node~~
- ~~If it's the same, we select the one based on the order of node index/ID~~

However, let's simplify the rule:

- **If it's the same, we select the one based on the order of node ID**

Multiple Options of Shortest Path

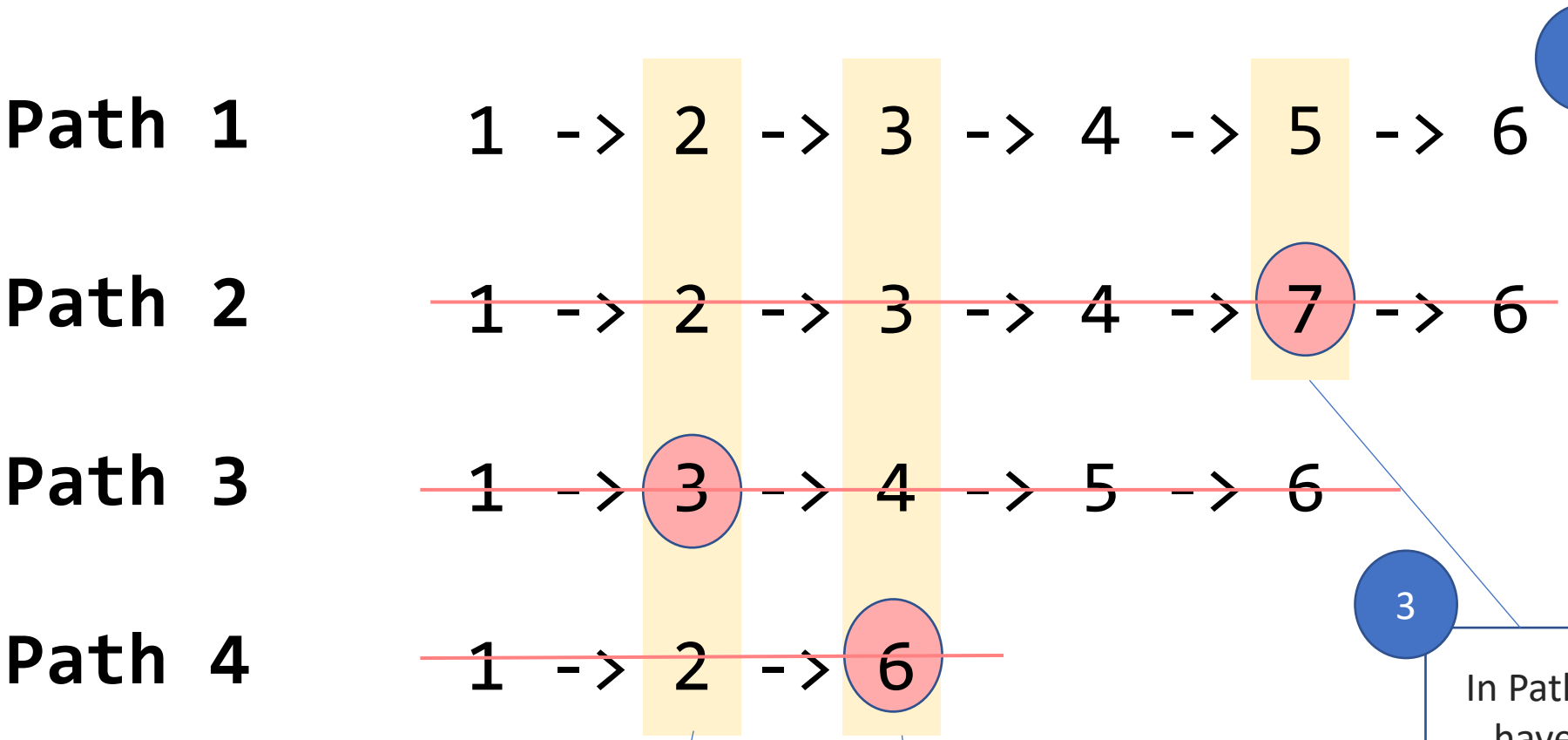
Path 1 1 -> 2 -> 3 -> 4 -> 5 -> 6

Path 2 1 -> 2 -> 3 -> 4 -> 7 -> 6

Path 3 1 -> 3 -> 4 -> 5 -> 6

Path 4 1 -> 2 -> 6

Multiple Options of Shortest Path



4 **We choose Path 1 as the shortest Path** (the remaining candidate)

3 **Remove path 2 from candidate**

In Path 2, the 5th node doesn't have the lowest ID number

1 **Remove path 3 from candidate**

In Path 3, the 2nd node doesn't have the lowest ID number

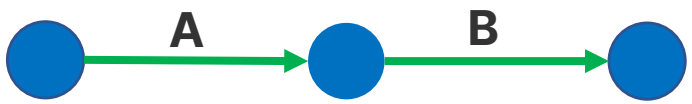
2 **Remove path 4 from candidate**

In Path 4, the 3rd node doesn't have the lowest ID number

Thank you for the time

Do you have any question?

Network Analysis



Sequence: 1 → 2 → 4 → 5 → 6 → 3

Total Message: 1070

Total Distance: 36.29