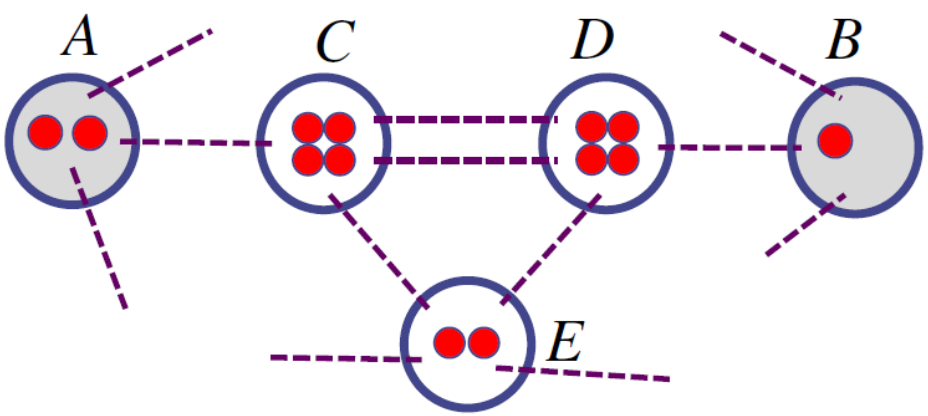
Data Structures
Programming Project #3

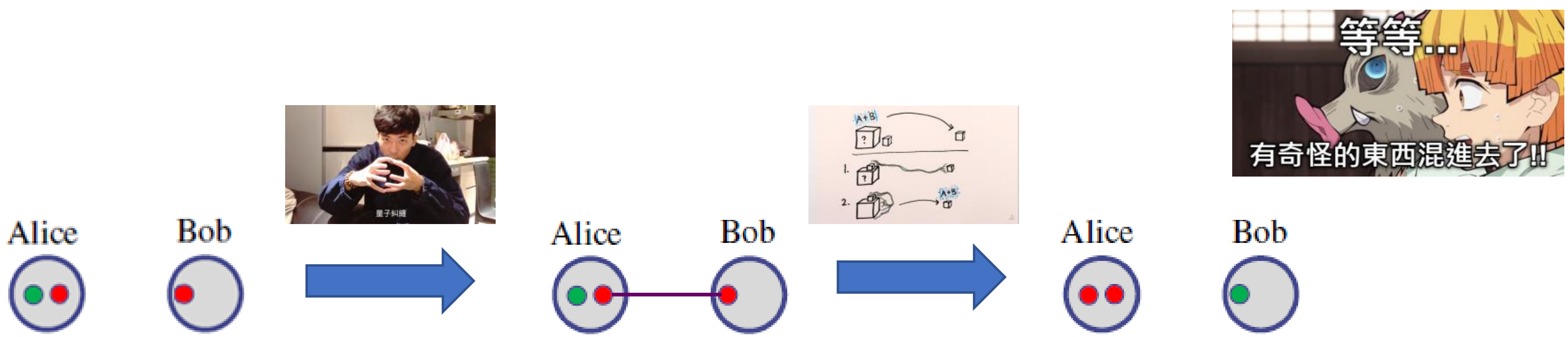
Data Transmission in Quantum Networks

- A quantum network:
- Nodes has a limited units of quantum memory
- Nodes are interconnected with a limited number of quantum channels (e.g., optical fiber)



Data Transmission in Quantum Networks

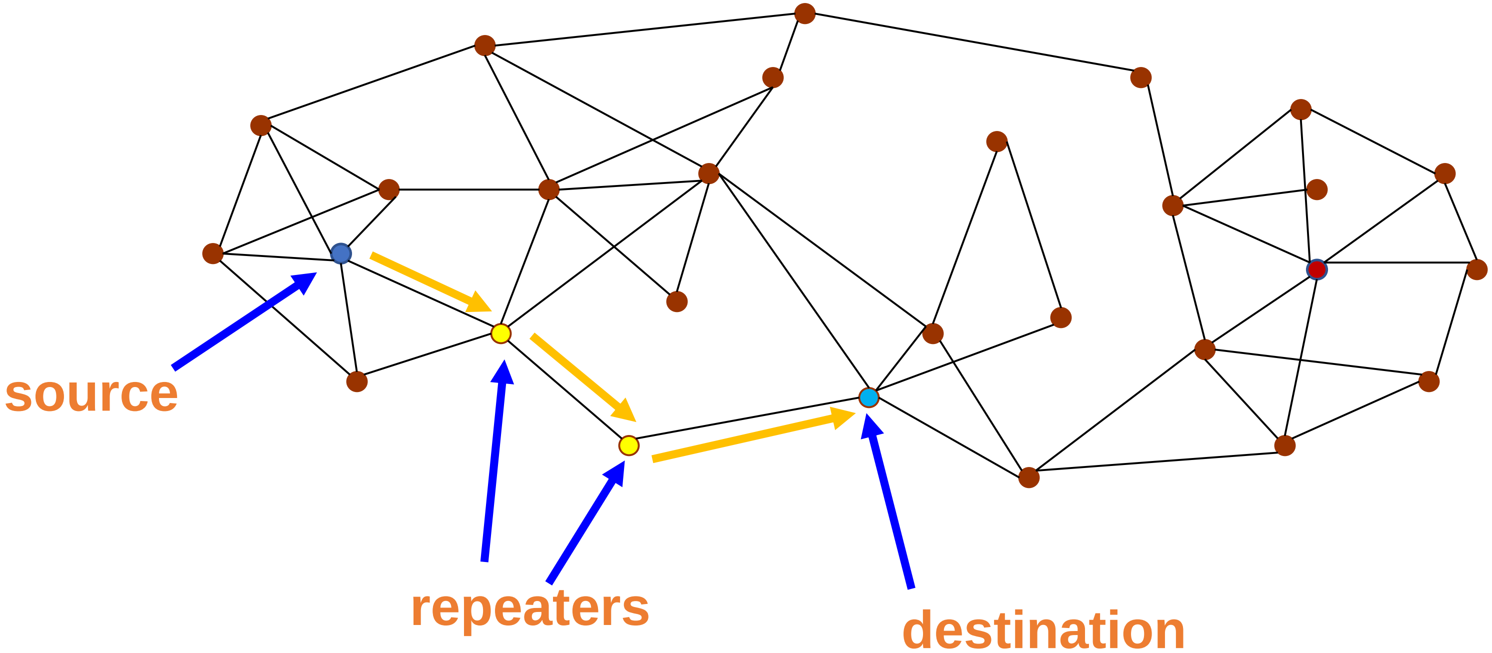
- Entangling (building an entangled link): Create an entangled pair between two nodes
- Precondition: Two nodes each with a quantum memory are interconnected with a quantum channel



https://www.popularmechanics.com/science/a25699/how-quantum-teleportation-works/

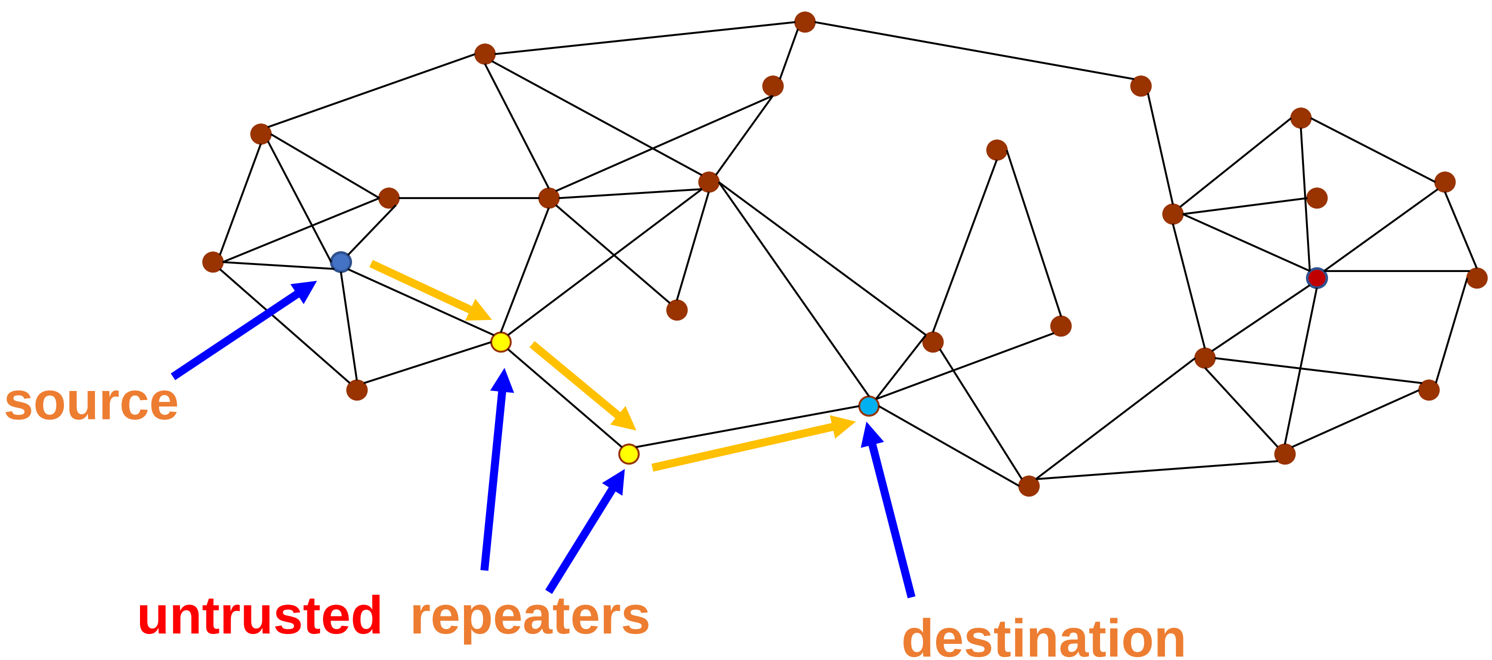
Data Transmission in Quantum Networks

- The two nodes may be distant from each other
- Classical networks:
  Repeaters use store and forward to transmit packets from a source to a destination



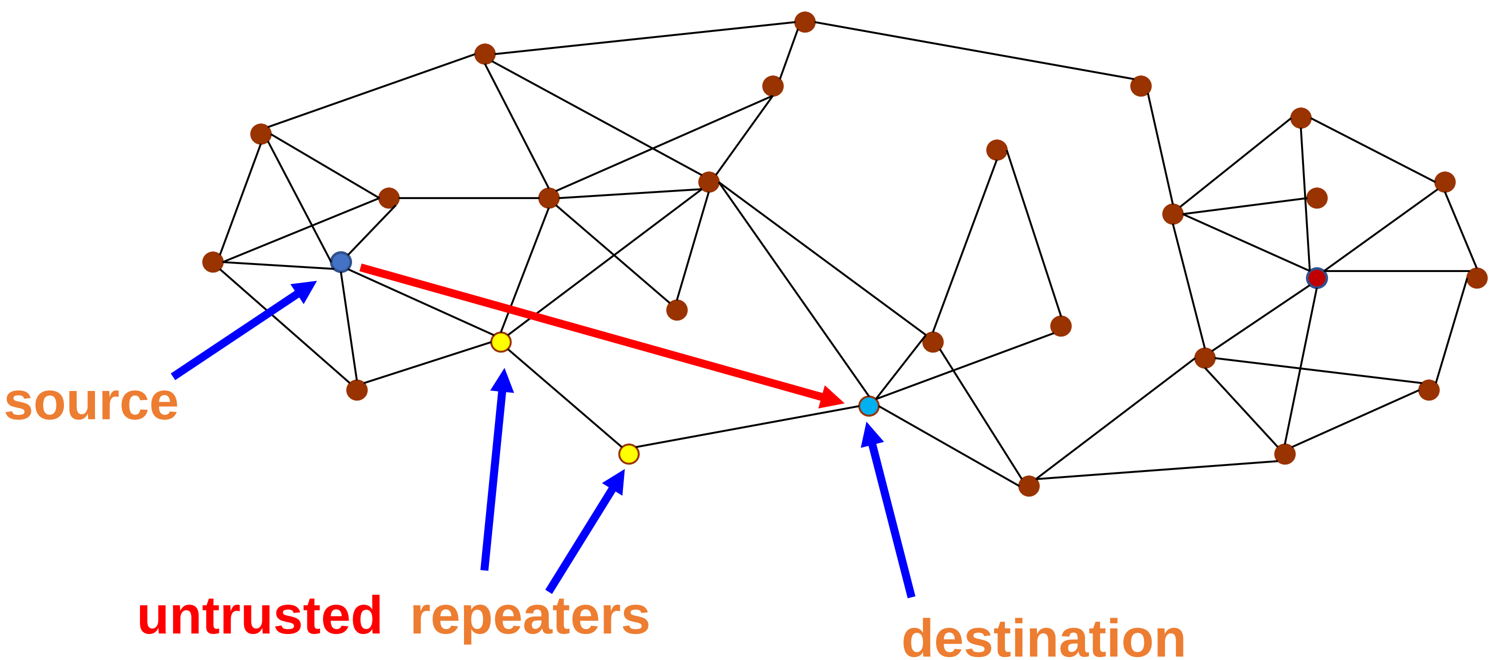
Data Transmission in Quantum Networks

- However, the data qubit may visit untrusted repeaters
- It could be destroyed, peeked at, or faked



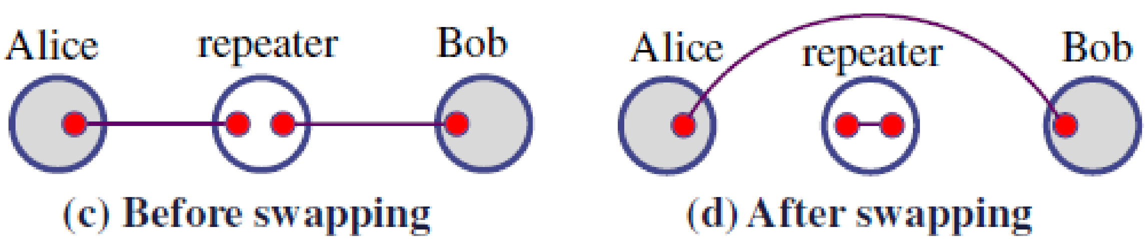
Data Transmission in Quantum Networks

- Can we send the data qubits without letting repeaters know?
- Yes, via Entanglement Swapping



Entanglement Swapping

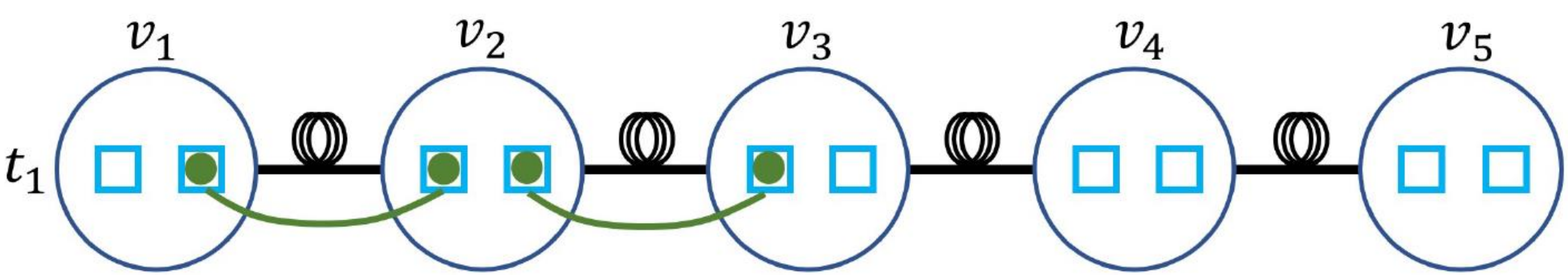
- Alice has a data qubit for Bob
- Entangling: build the links between Alice and the repeater and between Bob and the repeater
- Swapping: build a long-distance entanglement



Scheduling of Entangling and Swapping

- To construct an entanglement between v_1 and v_5

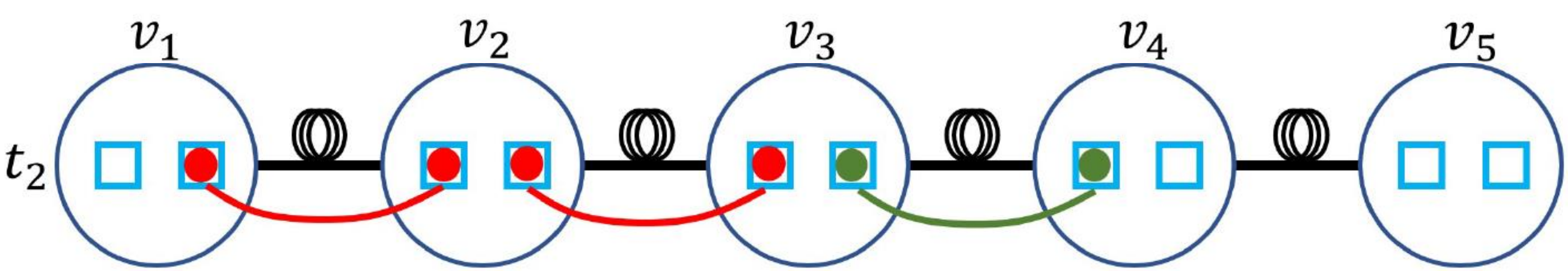
- One possible scheduling is linear:
- Conduct swapping one hop by one hop from the source to the destination



Scheduling of Entangling and Swapping

- To construct an entanglement between v_1 and v_5

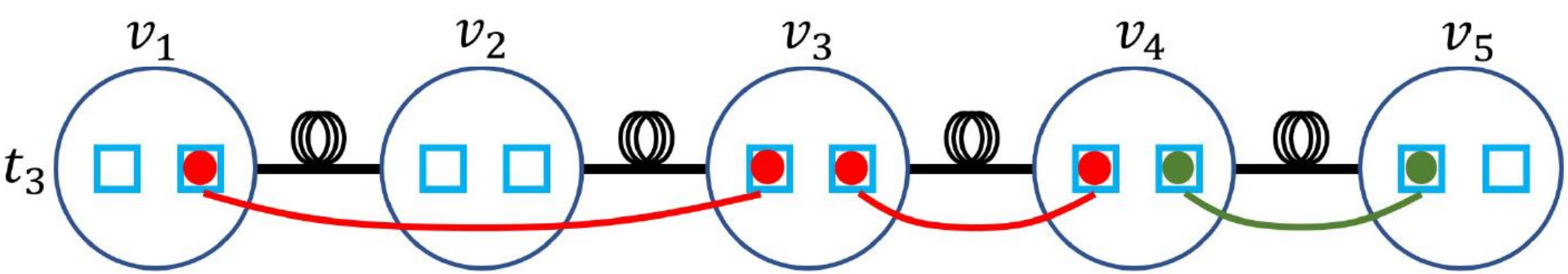
- One possible scheduling is linear:
- Conduct swapping one hop by one hop from the source to the destination



Scheduling of Entangling and Swapping

- To construct an entanglement between v_1 and v_5

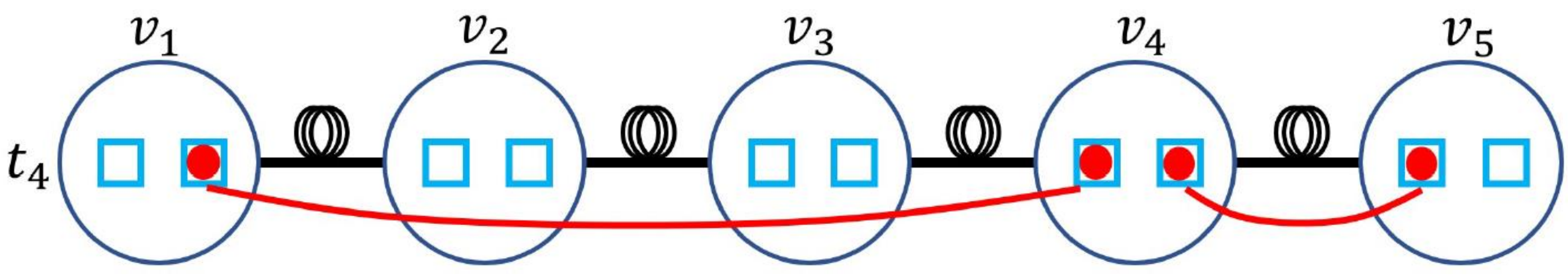
- One possible scheduling is linear:
- Conduct swapping one hop by one hop from the source to the destination



Scheduling of Entangling and Swapping

- To construct an entanglement between v_1 and v_5

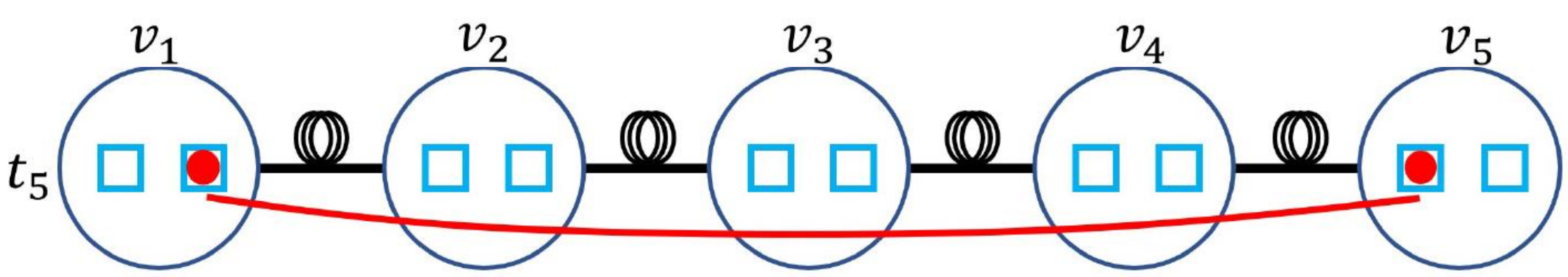
- One possible scheduling is linear:
- Conduct swapping one hop by one hop from the source to the destination



Scheduling of Entangling and Swapping

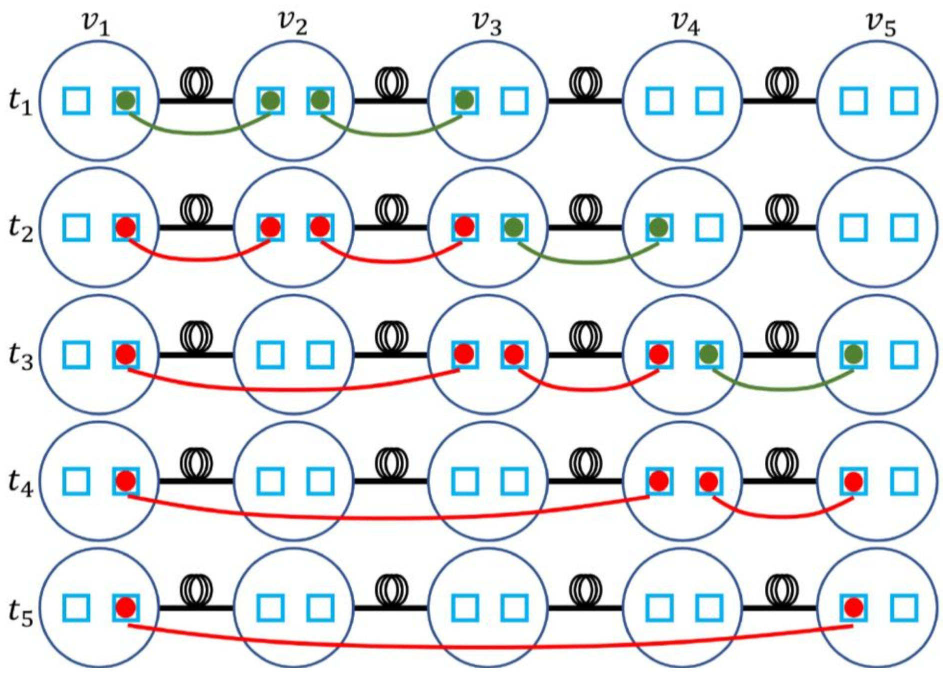
- To construct an entanglement between v_1 and v_5

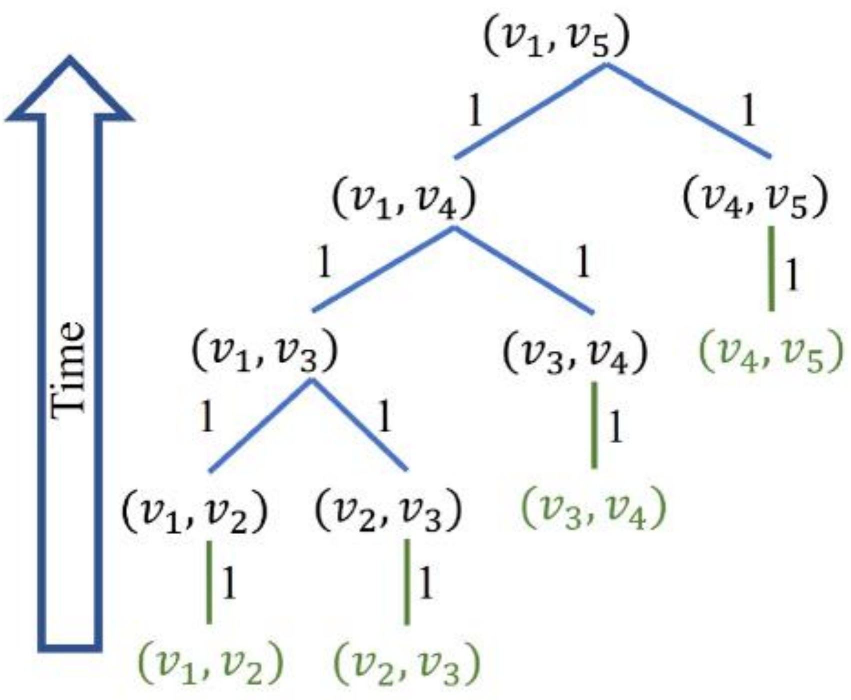
- One possible scheduling is linear:
- Conduct swapping one hop by one hop from the source to the destination



Scheduling vs Strategy Tree (Binary Tree)

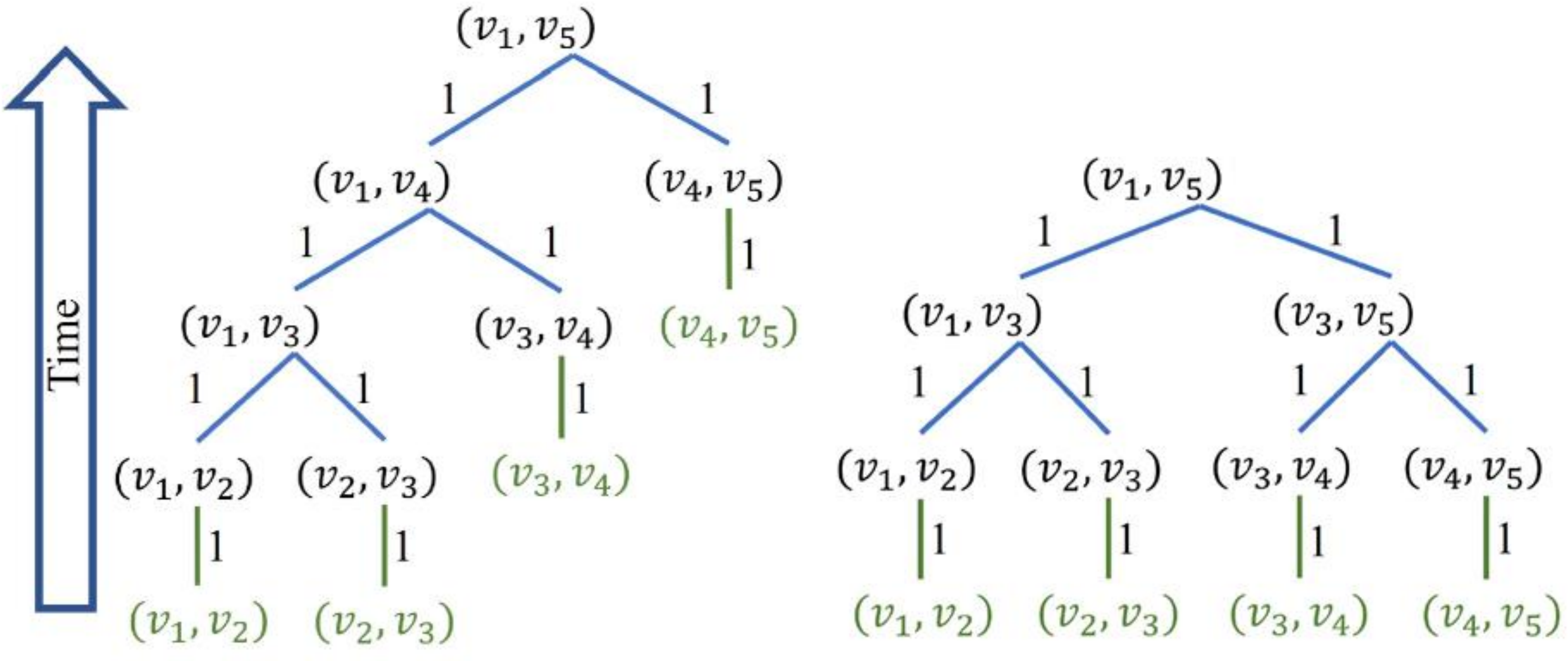
- Any scheduling can be represented by a binary tree





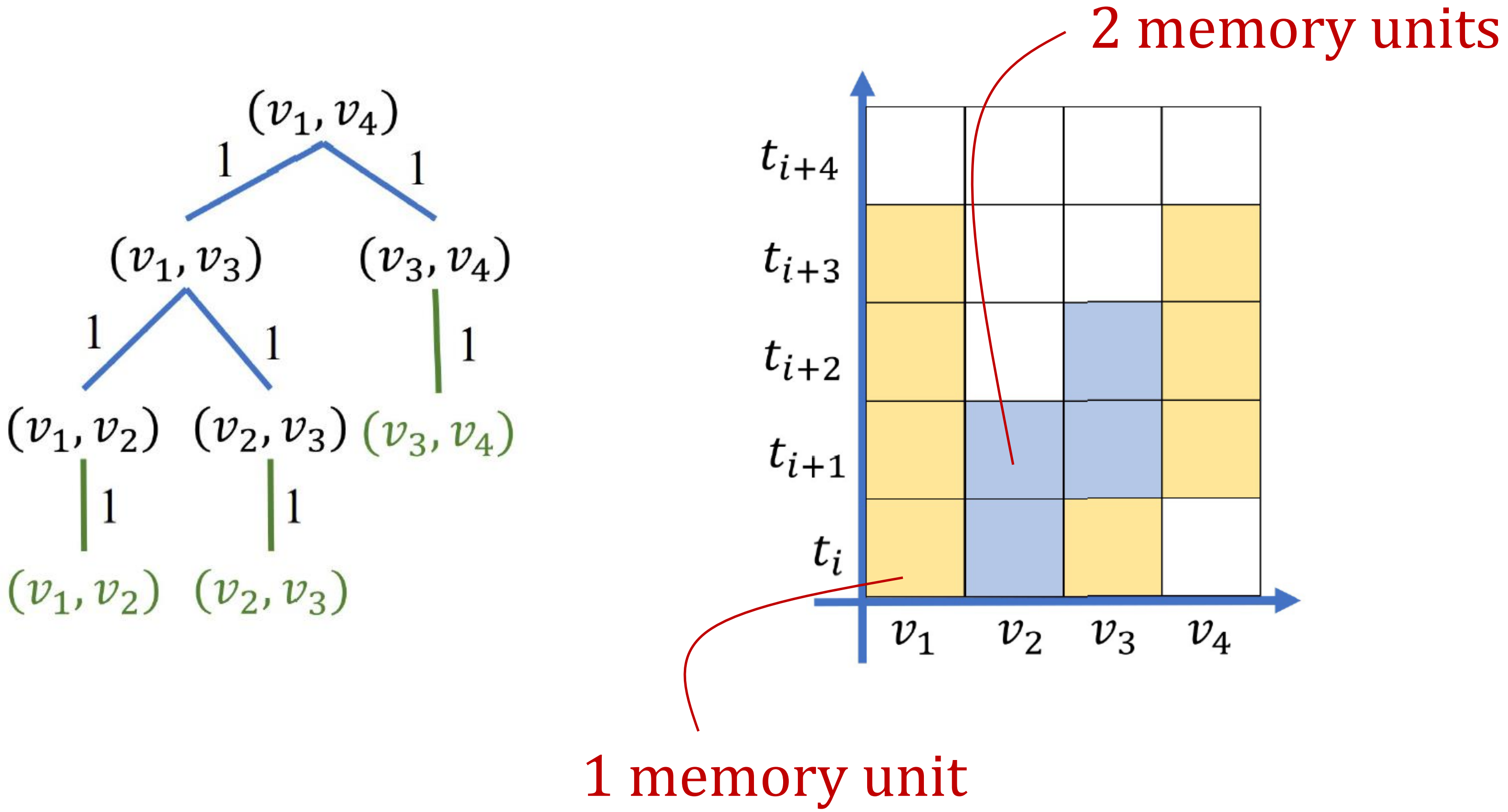
Scheduling vs Strategy Tree (Binary Tree)

- Different schedulings lead to different binary trees



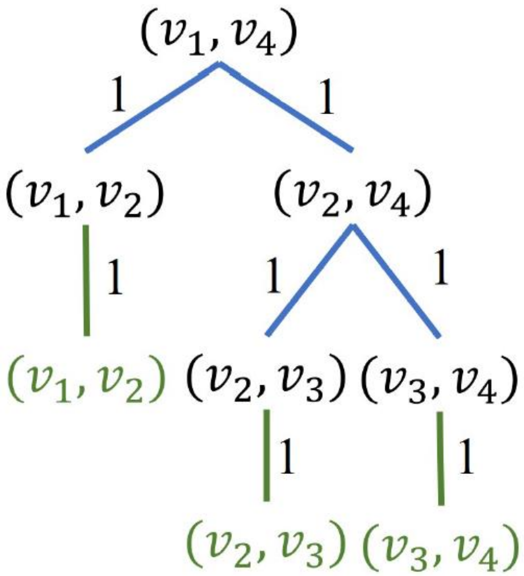
Strategy Tree vs Numerology

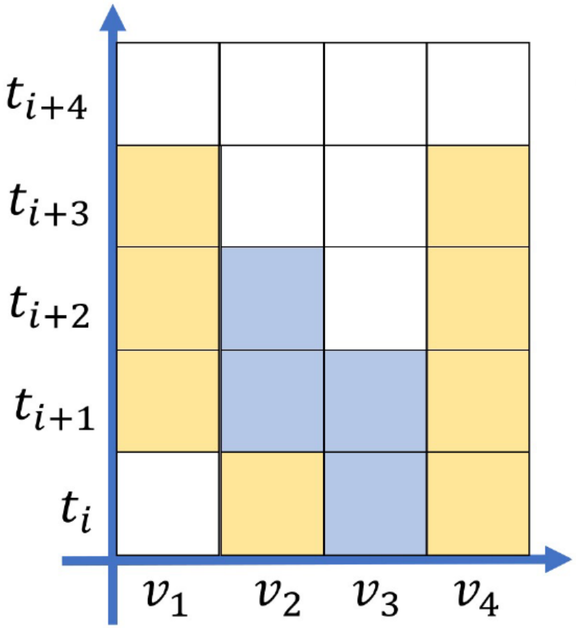
- Different trees lead to different numerologies



Strategy Tree vs Numerology

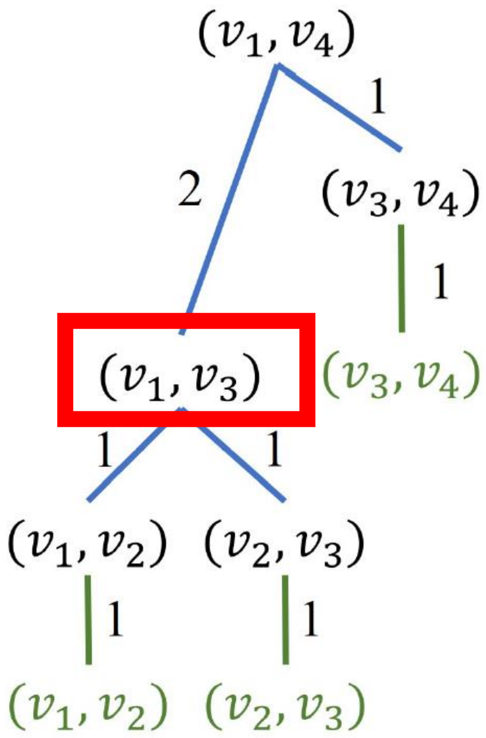
- Different trees lead to different numerologies

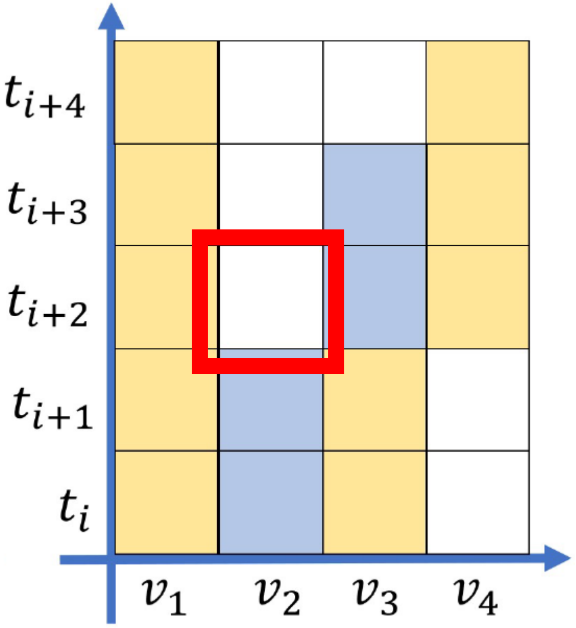




Strategy Tree vs Numerology

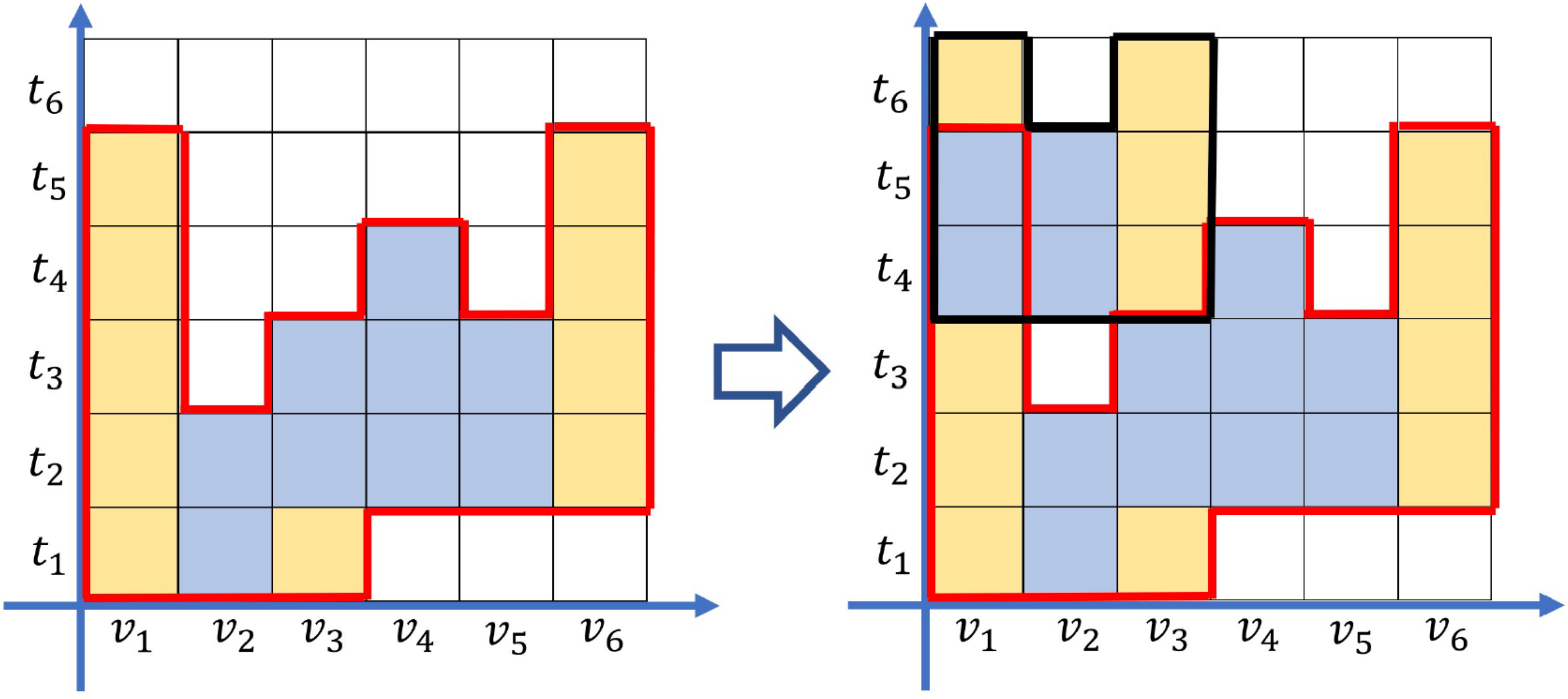
- Different trees lead to different numerologies





Quantum Resource Allocation Problem

- Multiple numerologies can overlap within T if the number of memory units is sufficient at that time



System Model & Problem Formulation

- Given:
- A quantum network with nodes and edges
- Each node has limited memory units
- (Assume fiber channels are always sufficient)
- Multiple source-destination (SD) requests
- Number of time slots (T: $t_1 \sim t_n$)

- Goal: maximize # accepted requests

- Constraints:
- Find a path for each accepted SD request
- Find a numerology for each accepted SD request
- Each node's memory size cannot be violated

Programming Project #3: Entanglement Routing & Numerology Selection

- Input:
  - A node-weighted network $G = (V, E)$
  - Multiple SD requests
  - A number of time slots T
- Procedure:
  - Accept or reject each SD request
  - Find a path with a numerology for accepted one
- Output:
  - The accepted SD requests
  - Their paths and numerologies
- The grade is proportional to # accepted SD requests

The Competition

- The grade is proportional to # accepted SD requests
- Basic: 60 (deadline)
  - A baseline solution (see the following pages)
  - (Meet the coding style requirements)
- Performance ranking (decided after the deadline)
  - [0%, 30%) (bottom): +0
  - [30%, 50%): + 5
  - [50%, 75%): + 10
  - [75%, 85%): + 15
  - [85%, 90%): + 20
  - [90%, 95%): + 25
  - [95%, 100%] (top): + 30
- Homework assistant (superb deadline)
  - +10

The Competition

- The grade … requests
- Basic: 60 (
  - A baselin
  - (Meet the
- Performan
  - [0%, 30%
  - [30%, 50
  - [50%, 75
  - [75%, 85
  - [85%, 90
  - [90%, 95
  - [95%, 10

We have TIME LIMIT!

- Homework assistant (superb deadline)
  - +10

The Competition

- The grade ...) requests
- Basic: 60 (
  - A baselin
  - (Meet the
- Performan ... eadline)
  - [0%, 30%
  - [30%, 50
  - [50%, 75
  - [75%, 85
  - [85%, 90
  - [90%, 95
  - [95%, 10

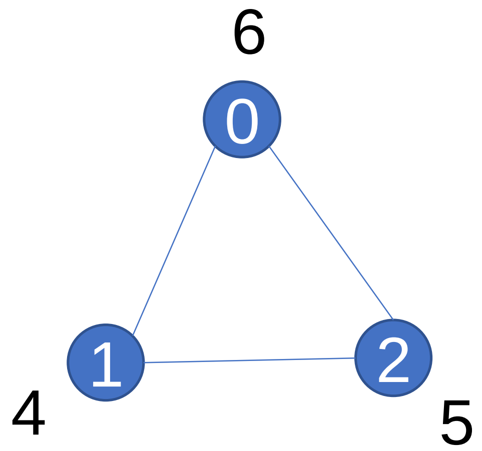
- Homework assistant (superb deadline)
  - +10

相信你們在做完作業以後

The Baseline Algorithm

- Sequentially find the minimum-hop path for each input SD request via BFS (on the original graph)
- If there is a tie, select the one that visits the node with a smaller ID first
  - 10 -> 4 -> 11 -> 15 -> 12 vs 10 -> 4 -> 9 -> 17 -> 12
  - Select 10 -> 4 -> 9 -> 17 -> 12
- Only use complete binary tree (recall that in Chap. 5) where the bottom leaf(s) is at time t_1 and side length $= 1$
- Examine whether the above path and numerology can be accommodated in the network
  - If yes, then accept the SD request
  - Otherwise, reject the SD request
- Repeat the above actions until all SD requests are examined

Input Sample: use scanf



Format:

#Nodes #Links #TimeSlots #Req
NodeID #QuantumMemories
...
LinkID LinkEnd1 LinkEnd2
...
ReqID ReqSrc ReqDst

Ex:

3 3 7 2
0 6
1 4
2 5
0 0 1
1 0 2
2 1 2
0 0 2
1 1 2

Output Sample: use printf

Format:

#AccReq

ReqID ReqSrc Rep1 Rep2… ReqDst

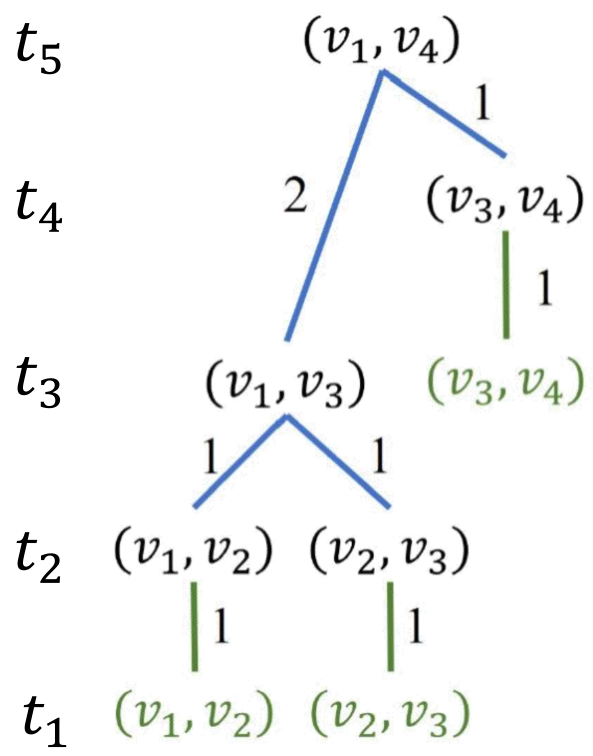
Tree Structure (detailed in the next slide)

…

Output Sample: use printf

Format:

Tree Structure:
use a postorder traversal




```
v1_ID v2_ID t2_time
V2_ID v3_ID t2_time
v1_ID v3_ID v1_ID v2_ID v2_ID v3_ID t3_time
v3_ID v4_ID t4_time
v1_ID v4_ID v1_ID v3_ID v3_ID v4_ID t5_time
```

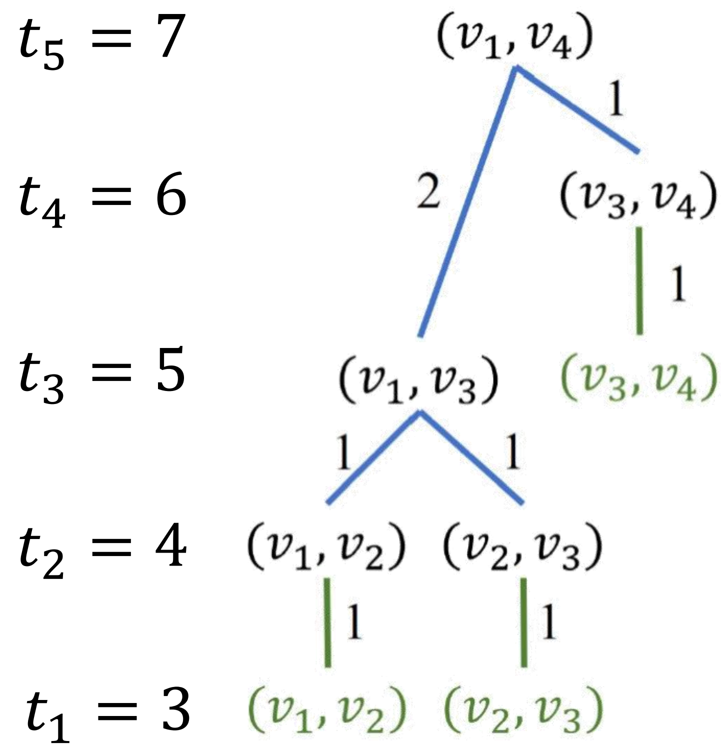
Output Sample: use printf

Format:

Tree Structure:

use a postorder traversal

```
10 8 4
8 15 4
10 15 10 8 8 15 5
15 20 6
10 20 10 15 15 20 7
```




```
v1_ID = 10
V2_ID = 8
v3_ID = 15
v4_ID = 20
```

Note

- Superb deadline: 12/14 Thu
- Deadline: 12/21 Thu
- Pass the test of our online judge platform
- Submit your code to E-course2
- Demonstrate your code remotely or in person with TA
- **C Source code (i.e., only .c)**
- Show a good programming style

It is worth noting that…

- The processes of entangling and swapping on the path may fail (although we don't consider it)
- In addition, strategy tree height also affects quality of entangled pair (although we don't consider it)

- For example:
- The success probability of entangling is $p = e^{-\alpha \cdot L}$, where α is a constant of the fiber material and L is the distance between the two nodes
- The success probability of swapping is q, which is related to the technique