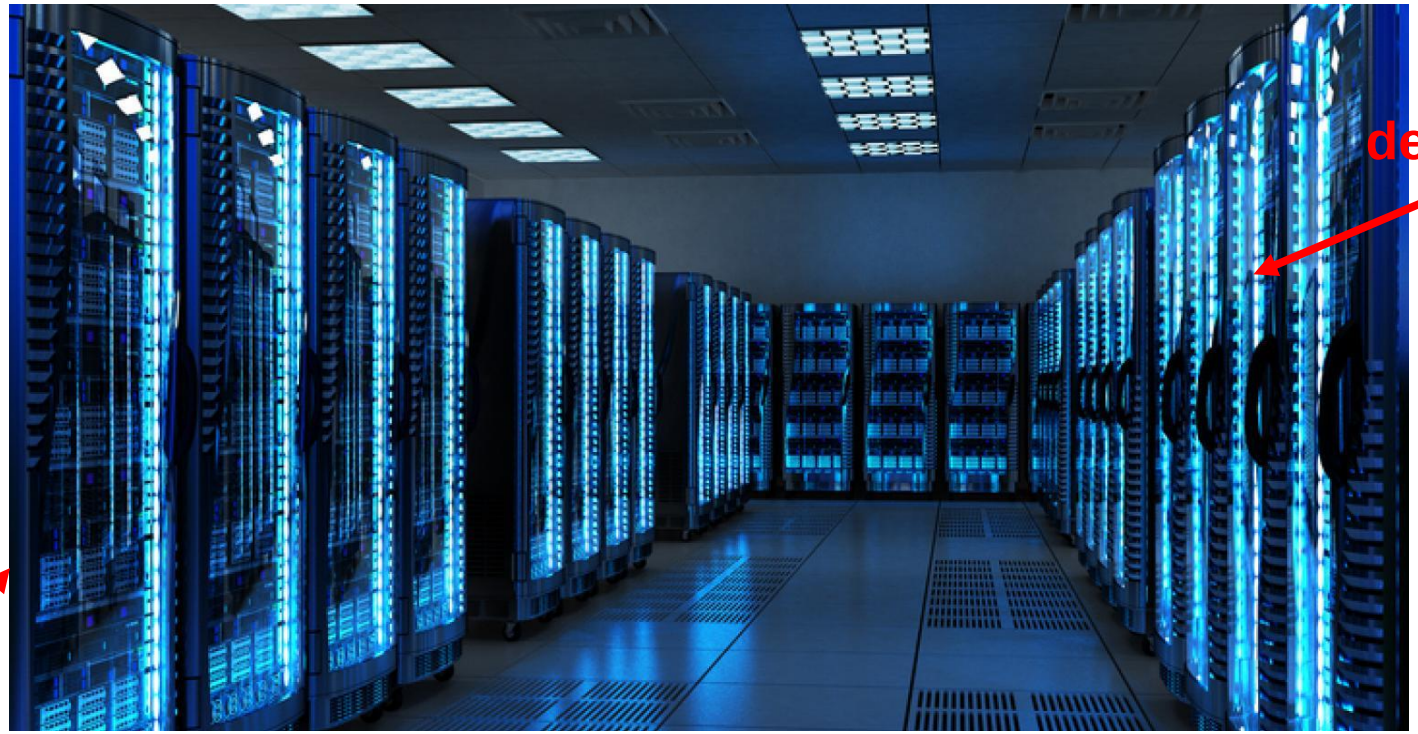
Data Structures
Programming Project #2

Data Center

- A data center consists of multiple severs
- The servers are connected by switches in a local area network

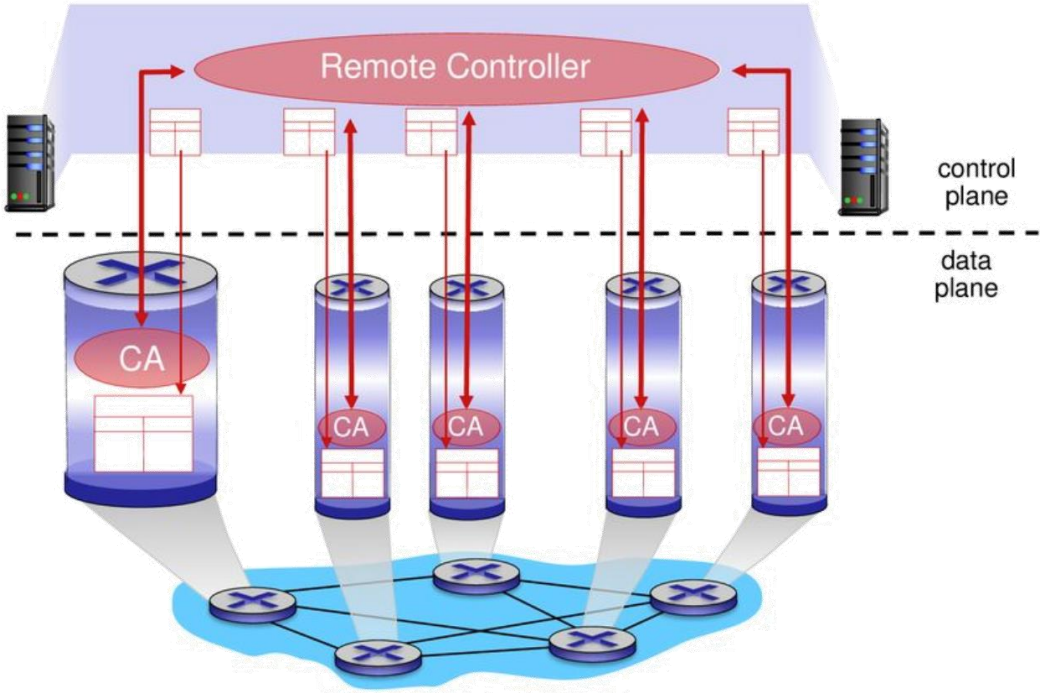


Switches

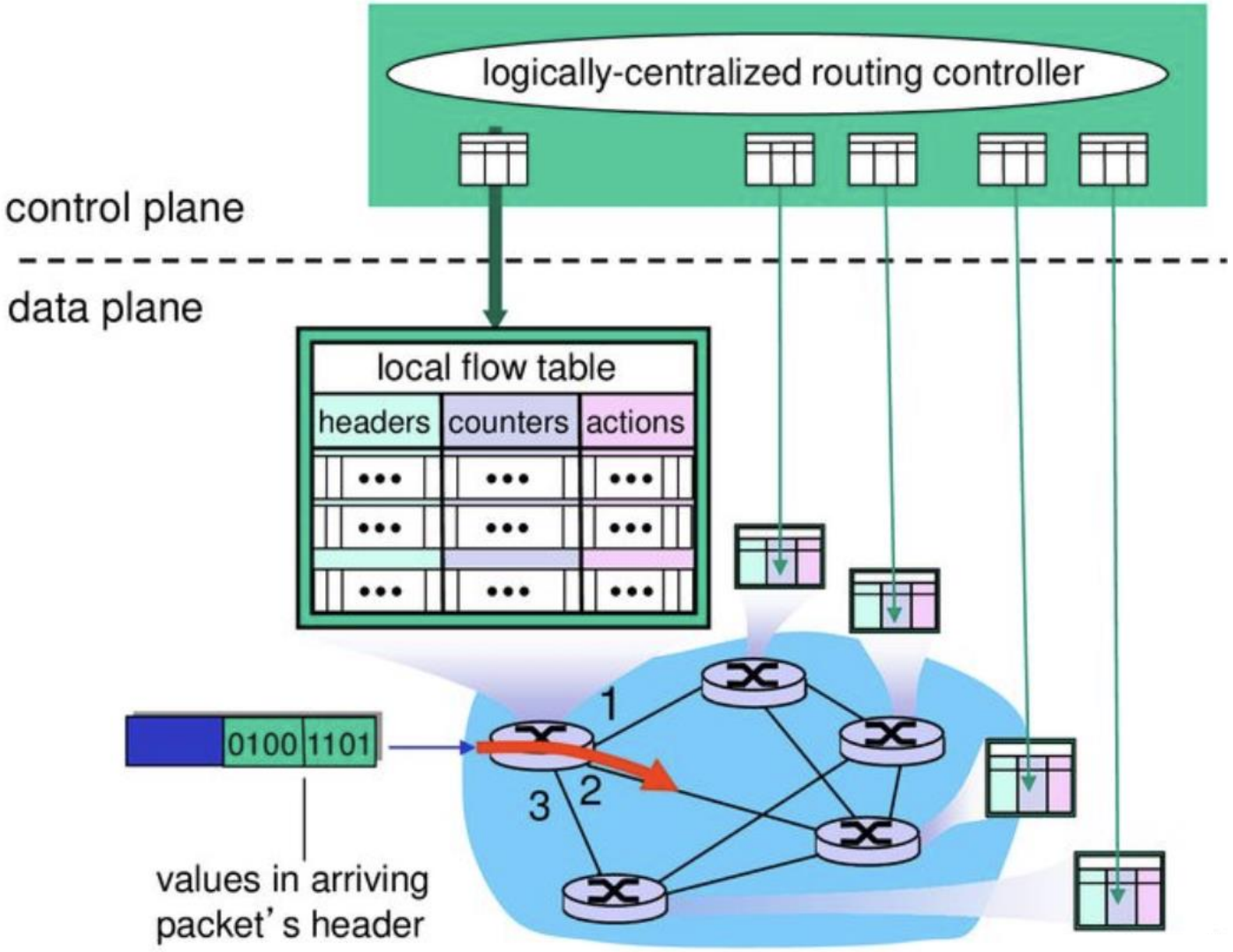
- Each switch has multiple ports
- Receive and forward the packets from a port to another port

SDN-enabled Switches

- A centralized controller is introduced – software-defined networking (SDN)

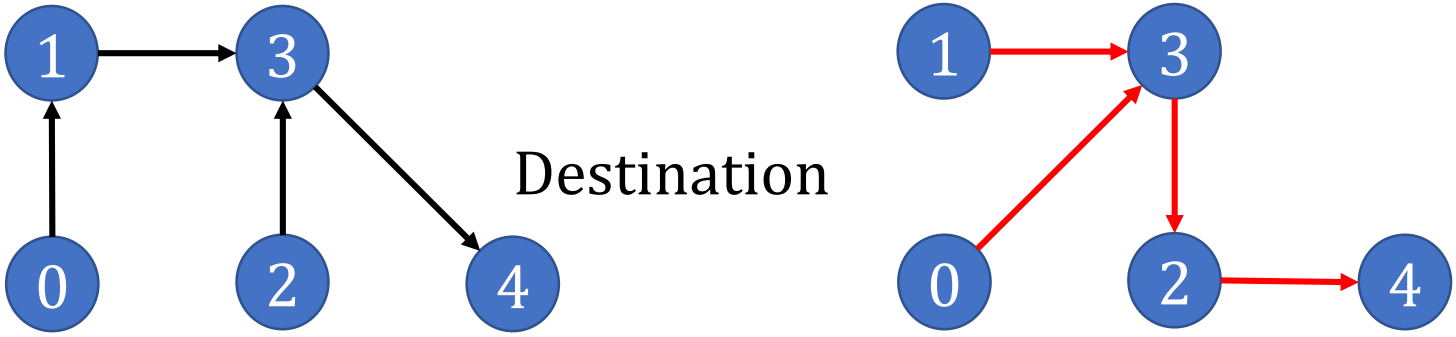


Installing Rules in the SDN-enabled Switches



Routing Path Update (aka Network Update)

- Given:
  the **old** and **new** destination-based routing tree
- Update the destination-based routing tree

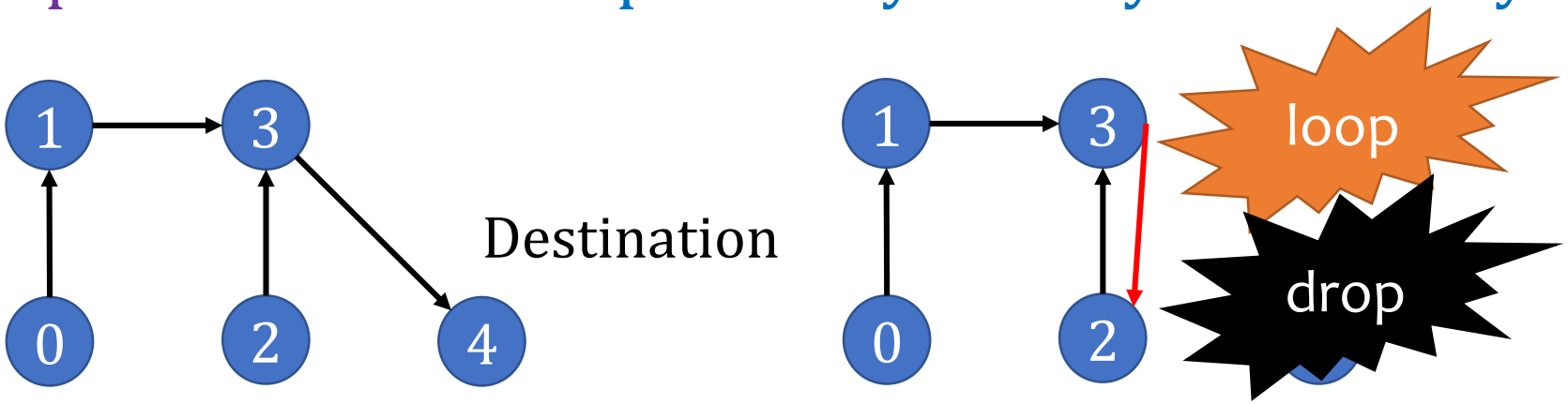


Node ID	0	1	2	3	4
To	1	3	3	4	-1

Node ID	0	1	2	3	4
To	3	3	4	2	-1

Difficulty of Network Update in SDN

- The controller is logically-centralized
- However, the underlying mechanism is distributed
- Each switch receives the update message and updates its rule independently and asynchronously



Node ID	0	1	2	3	4
To	1	3	3	4	-1

Node ID	0	1	2	3	4
To	1	3	3	2	-1

Difficulty of Network Update in SDN

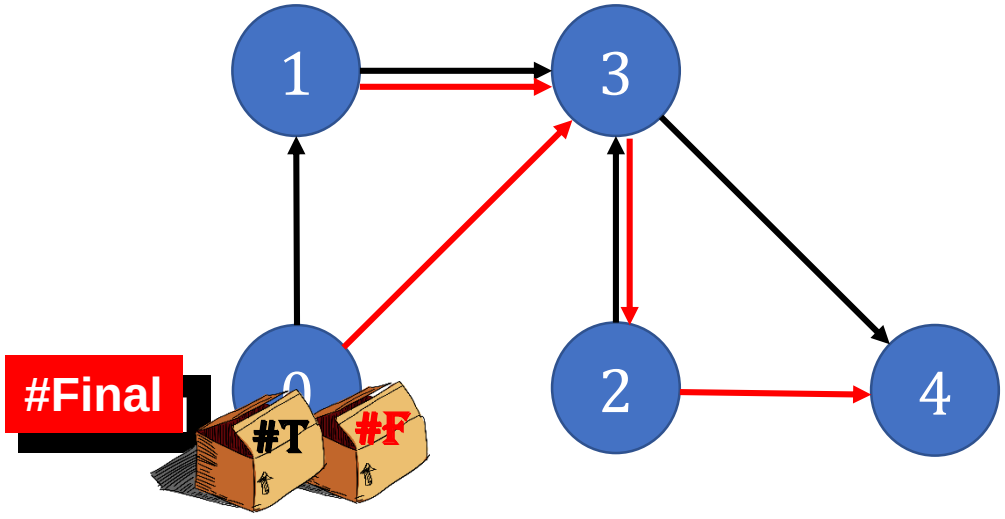
- The controller is logically-centralized
- However, the underlying mechanism is distributed
- Each switch receives the update message and updates its rule independently and asynchronously

- How to solve the issue?
  - Two-phase commit (SIGCOMM 2012)
  - Round-based update for routing trees (HotNets, 2013)
  - Round-based update for single paths (TON, 2018)

"Survey of Consistent Software-Defined Network Updates",
in IEEE Communications Surveys & Tutorials, 2019

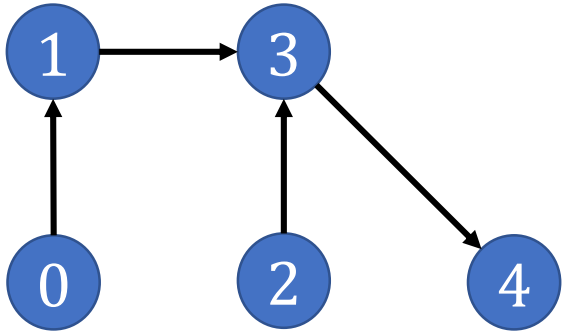
Difficulty of Network Update in SDN

- Two-phase commit (SIGCOMM 2012)
- Drawback: waste the TCAM size during the update

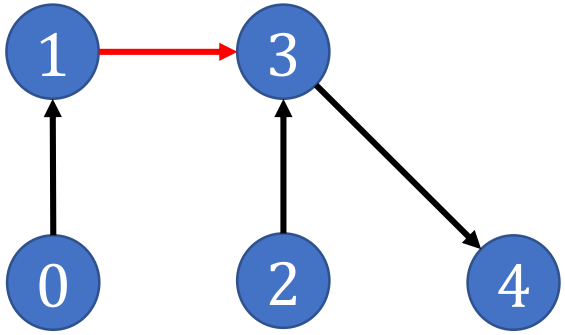


Difficulty of Network Update in SDN

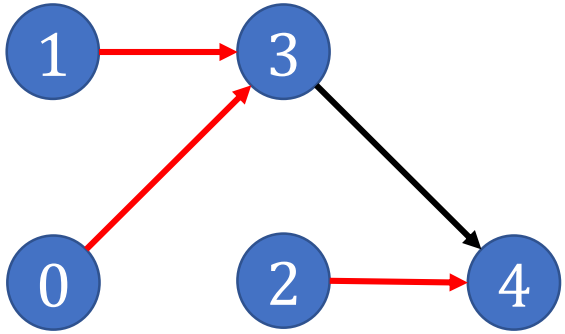
- Round-based update (單純概念介紹)



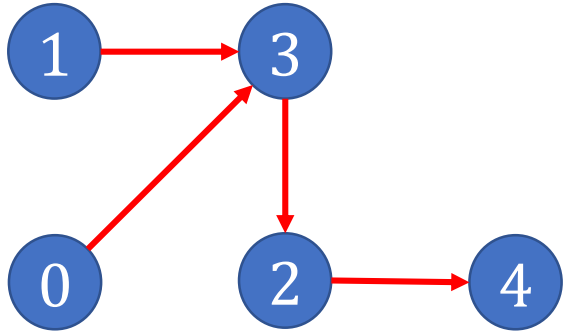
Step 1: No update 1



Step 2: Update 0 and 2

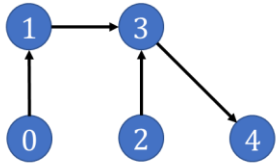


Step 3: Update 3

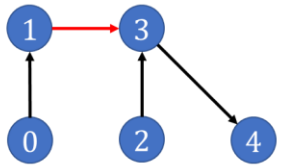


Programming Project #2: Minimize the number of update rounds

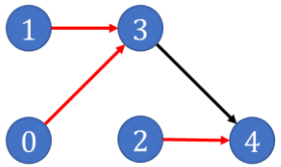
- Input:
  - Numbers of nodes
  - Nodes in old and new routing trees
- Procedure:
  - Minimize the rounds of update
- Output:
  - Rules of each switch in each round
- The grade is inversely proportional to the number of rounds



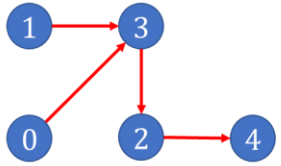
Step 1: No update 1



Step 2: Update 0 and 2

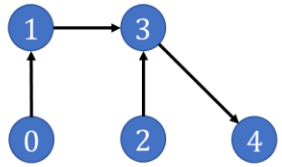


Step 3: Update 3

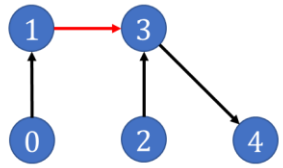


Programming Project #2: Minimize the number of update rounds

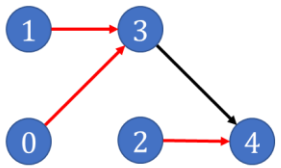
- Input:
  - Numbers of nodes
  - Nodes in old and new routing trees
- Procedure:
  - Minimize the rounds of update
- Output:
  - Rules of each switch in each round
- The grade is inversely proportional to the number of rounds



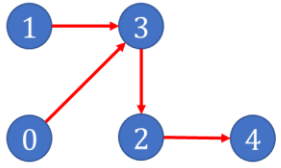
Step 1: No update 1

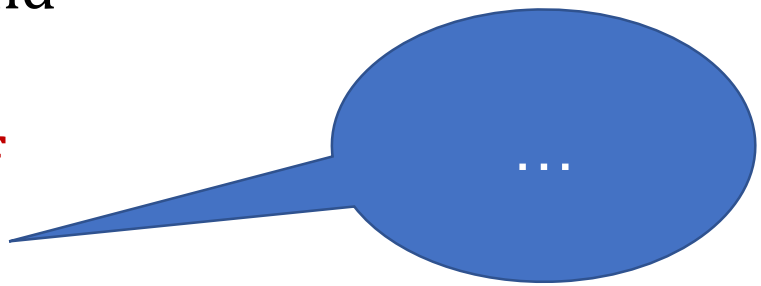


Step 2: Update 0 and 2



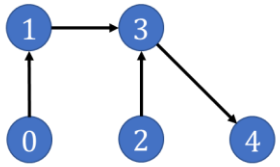
Step 3: Update 3



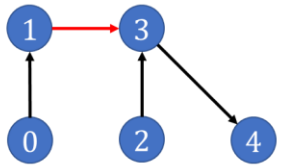


Programming Project #2: Minimize the number of update rounds

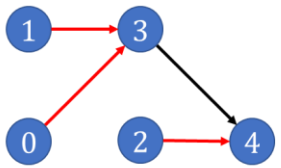
- Input:
  - Numbers of nodes
  - Nodes in old and new routing trees
- Procedure:
  - Minimize the rounds of update
- Output:
  - Rules of each switch in each round
- The grade is inversely proportional to the number of rounds



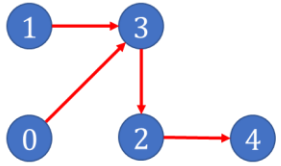
Step 1: No update 1



Step 2: Update 0 and 2



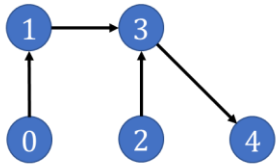
Step 3: Update 3



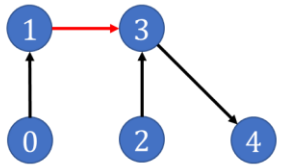
怎麼辦

Programming Project #2: Minimize the number of update rounds

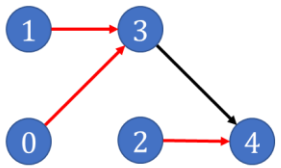
- Input:
  - Numbers of nodes
  - Nodes in old and new routing trees
- Procedure:
  - Minimize the rounds of update
- Output:
  - Rules of each switch in each round
- The grade is inversely proportional to the number of rounds



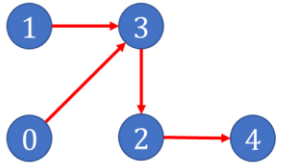
Step 1: No update 1



Step 2: Update 0 and 2



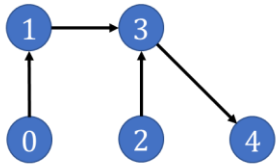
Step 3: Update 3



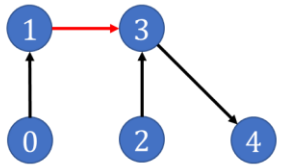
早知道不加簽了

Programming Project #2: Round-based network update

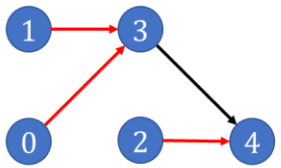
- Input:
  - Numbers of nodes
  - Nodes in old and new routing trees
- Procedure:
  - Minimize the rounds of update
- Output:
  - Rules of each switch in each round
- Implement a given algorithm.



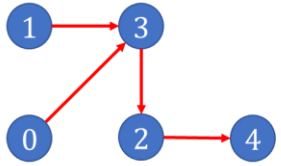
Step 1: No update 1



Step 2: Update 0 and 2



Step 3: Update 3



好啦~
騙你的

Round-Based Update Algorithm (HotNets, 2013)

Black line: old tree

Red line: new tree

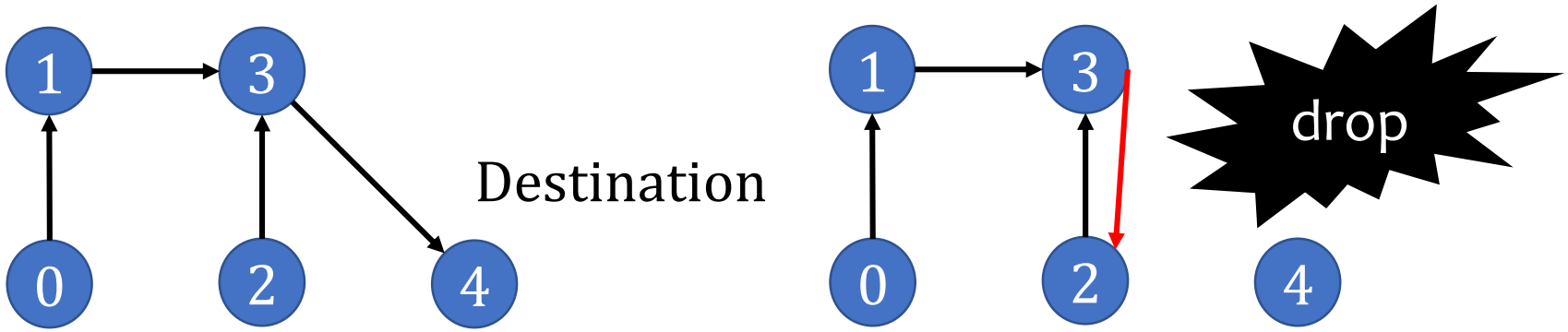
Node ID	0	1	2	3	4
To	1	3	3	4	-1

Node ID	0	1	2	3	4
To	3	3	4	2	-1

Round-Based Update Algorithm (HotNets, 2013)

Black line: old tree

Red line: new tree



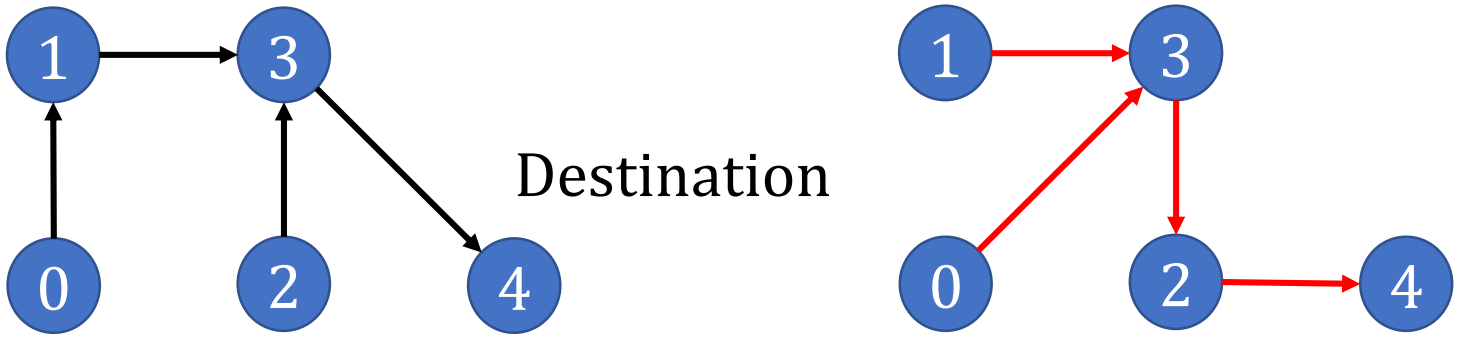
Node ID	0	1	2	3	4
To	1	3	3	4	-1

Node ID	0	1	2	3	4
To	3	3	3	2	-1

Round-Based Update Algorithm (HotNets, 2013)

Black line: old tree Red line: new tree

A node can safely update to new rules after its parent has switched



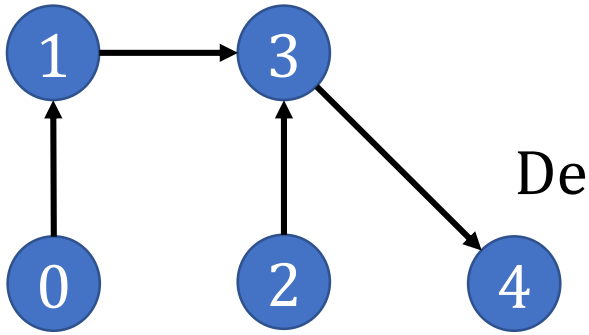
Node ID	0	1	2	3	4
To/Parent	1	3	3	4	-1

Node ID	0	1	2	3	4
To/Parent	3	3	4	2	-1

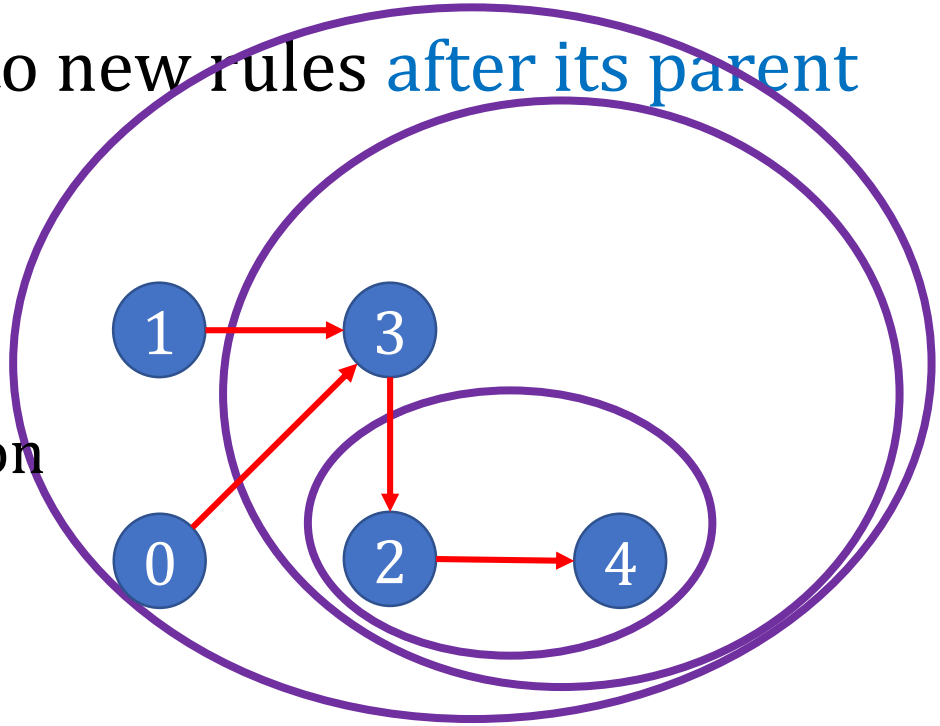
Round-Based Update Algorithm (HotNets, 2013)

Black line: old tree Red line: new tree

A node can safely update to new rules after its parent has switched



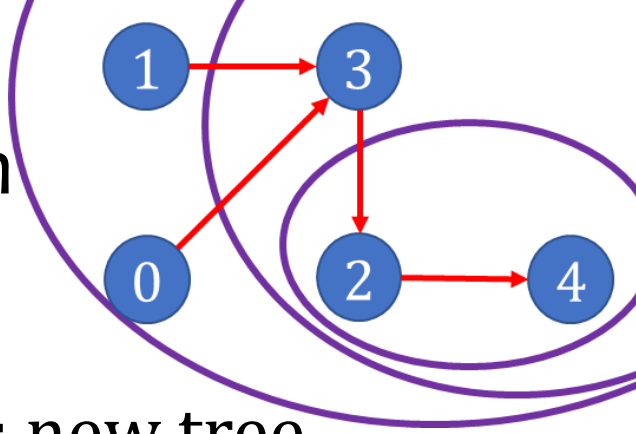
Destination



Node ID	0	1	2	3	4
To/Parent	1	3	3	4	-1

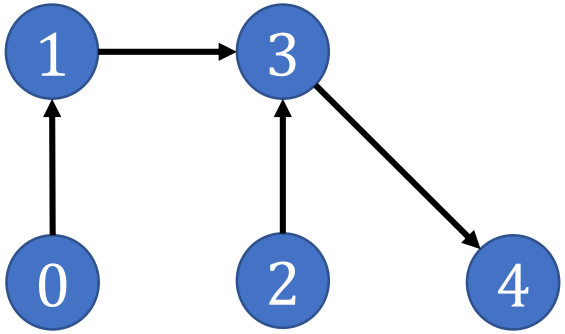
Node ID	0	1	2	3	4
To/Parent	3	3	4	2	-1

Round-Based Update Algorithm (HotNets, 2013)

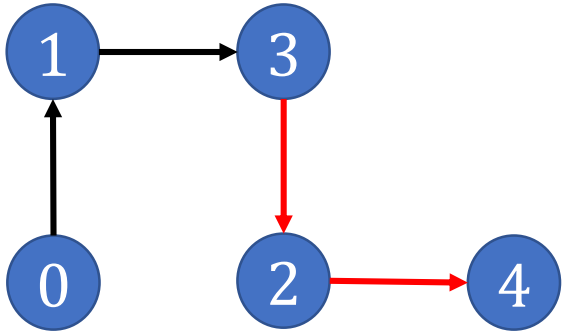


Black line: old tree

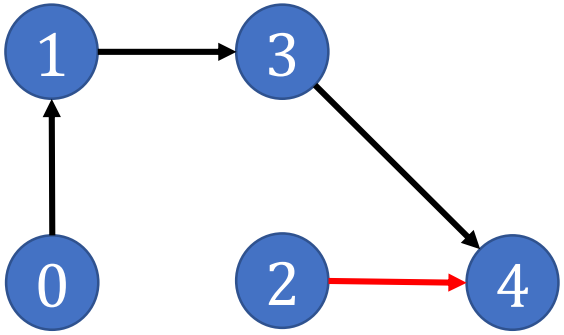
Red line: new tree



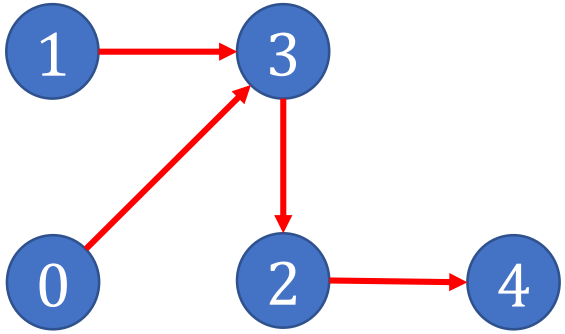
Step 2: Update 3



Step 1: Update 2



Step 3: Update 0 and 1



Requirement

- You have to:
- Use a linked list to save the children for each node
- Use a pointer to save the parent for each node
- You can use the structure like this:

```
typedef struct node* NodePtr;
typedef struct Node {
        int id;
        ListNodePtr children;
        NodePtr parent;
};
```

```
typedef struct ListNode* ListNodePtr;
typedef struct ListNode {
        NodePtr data;
        ListNodePtr link;
};
```

```
NodePtr switches = malloc (sizeof(Node)*num_of_nodes);
```

Discussion

- Minimizing the number of update rounds is NP-hard
- You cannot find an efficient algorithm for this problem unless NP = P

- There are many heuristic algorithms
- "Loop-Free Route Updates for Software-Defined Networks," in IEEE/ACM TON 2018 (上一屆實作的)
- "On Consistent Updates in Software Defined Networks", in ACM HotNets 2013. (你們要實作的)
- …

Input Sample:

use scanf

Format:

#Nodes

Tree1

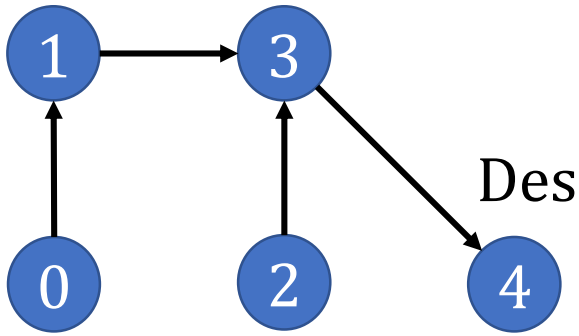
Tree2

e.g.,

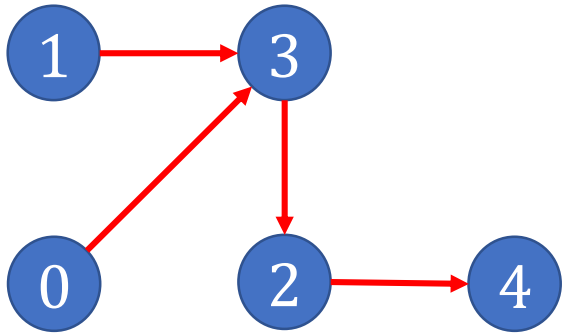
5

1 3 3 4 -1

3 3 4 2 -1



Node ID	0	1	2	3	4
To/Parent	1	3	3	4	-1



Node ID	0	1	2	3	4
To/Parent	3	3	4	2	-1

Output Sample:
use printf

Format:

#Rounds

Tree1

Tree2

…

e.g.,

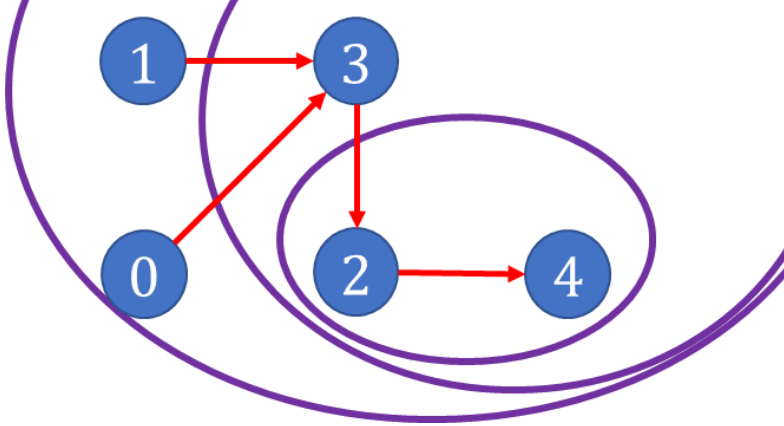
4

1 3 3 4 -1

1 3 4 4 -1

1 3 4 2 -1

3 3 4 2 -1



Node ID	0	1	2	3	4
To/Parent	1	3	3	4	-1

Node ID	0	1	2	3	4
To/Parent	1	3	4	4	-1

Node ID	0	1	2	3	4
To/Parent	1	3	4	2	-1

Node ID	0	1	2	3	4
To/Parent	3	3	4	2	-1

Note

- Superb deadline: 10/27 Tue
- Deadline: 11/03 Tue
- Submit your code to E-course2
- Demonstrate your code in 工院1館 401B
- **C Source code (not C++; compiled with gcc, not g++)**
- Show a good programming style