Network System Capstone
@CS.NYCU

2025.05.08: Lab5

Instructor: Kate Ching-Ju Lin (林靖茹)

Agenda

- **Lab Overview**
- Problem Definition
- Optimization Model
- Tasks
- Report & Result
- Submission

Lab Overview

- **Load balancing routing**
  - Given a network topology and a set of source-destination pairs
  - Assign each SD pair a routing path such that the total throughput of all SD pairs is maximized

- **Approach**
  - Formulate the problem as an optimization model
  - Use OR-Tools to find the optimal solution
  - Design and implement your own algorithm in C++

Agenda

- Lab Overview
- **Problem Definition**
- Optimization Model
- Tasks
- Report & Result
- Submission

Input Network Topology

- Given a set of source-destination pairs
- Consider a directional graph
- Links are **directional** and **independent**
  - Each node has two network interfaces, one as transmitter and the other as receiver
  - (i → j) and (j → i) are independent links with non-shared capacity
  - Traffic carried on each directional link must not exceed its capacity
- Assumption:
  - A node can be a source/destination and, at the same time, a forwarder for other SD pairs

Problem Definition

Objective

- Maximize the total throughput across all SD pairs

Constraints

- Single-path constraint: Each SD pair transmits over a single path (no loop)
- Link capacity: total amount of traffic sent over a link is bounded by its capacity
- Limited transmitter/receiver: Number of outgoing/incoming links is bounded by one
- Flow conservation: total incoming flows equals total outgoing flows

Agenda

- Lab Overview
- Problem Definition
- **Optimization Model**
- Tasks
- Report & Result
- Submission

Model (1/3)

Input:

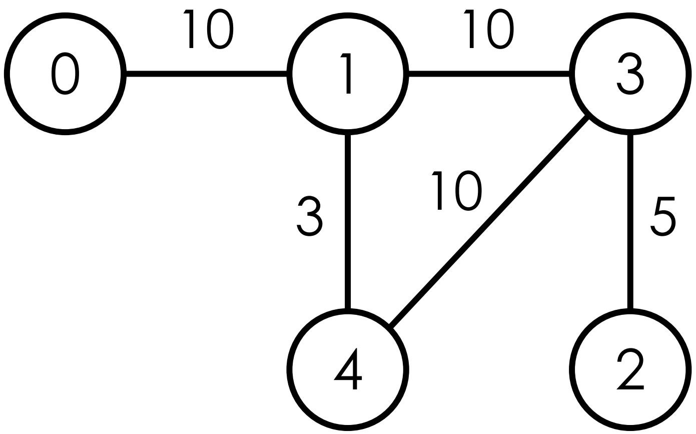
- $\mathcal{F}$: the set of source-destination pairs
  - s_f the source of SD pair $i \in \mathcal{I}$
  - d_f the destination of SD pair $i \in \mathcal{I}$
- $\mathcal{S} = \{s_f : f \in \mathcal{F}\}$ set of sources
- $\mathcal{D} = \{d_f : f \in \mathcal{F}\}$ set of destinations
- $G(\mathcal{U}, \mathcal{E})$: original network consisting of set of nodes $\mathcal{U}$ and set of edges $\mathcal{E}$
- $r_{u,v} \in \mathbb{R}_0^+$: data rate of link $(u, v) \in \mathcal{E}$

Model: Build New Graph

New Graph:

- $\mathcal{E}_s = \{(s, u_s) : s \in \mathcal{S}\}$
  - set of edges connecting a source s to its router u_s
- $\mathcal{E}_d = \{(u_d, d) : d \in \mathcal{D}\}$
  - set of edges connecting a destination d from its router u_d
- $G'(\mathcal{V}, \mathcal{E}')$: transformed network
  - set of nodes $\mathcal{V} = \mathcal{U} \cup \mathcal{S} \cup \mathcal{D}$
  - set of edges $\mathcal{E}' = \mathcal{E} \cup \mathcal{E}_s \cup \mathcal{E}_d$

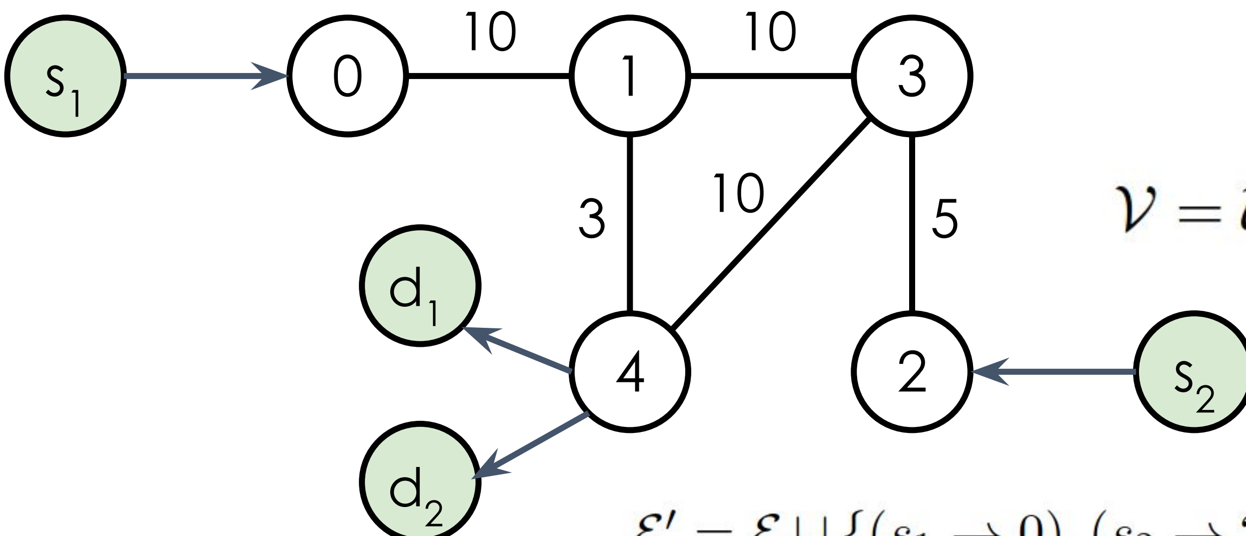
Build Graph Example



$$\mathcal{F} = \{(0,4),(2,4)\}$$

$$\mathcal{U} = \{0,1,2,3,4\}$$

$$\mathcal{E} = \{(0,1),(1,3),(1,4),(3,4),(3,2)\}$$



$$\mathcal{V} = \mathcal{U} \cup \{s_1, s_2, d_1, d_2\}$$

$$\mathcal{E}' = \mathcal{E} \cup \{(s_1 \to 0),(s_2 \to 2),(4 \to d_1),(4 \to d_2)\}$$

Model (2/3)

Variables:

- $X_{f,u,v} \in \mathbb{R}_0^+$: the data rate assigned to SD pair $f \in \mathcal{F}$ on link $(u,v) \in \mathcal{V}^2$
- $Y_{f,u,v} \in \{0,1\}$: binary indicator deciding whether the link $(u,v) \in \mathcal{U}^2$ is assigned to SD pair $f \in \mathcal{F}$
- $Z_{u,v} \in \{0,1\}$: binary variable indicating whether link $(u,v) \in \mathcal{U}^2$ is used

Model (3/3)

- **Integer Linear Programming (ILP)**

$$\max \sum_{f\in\mathcal{F}} X_{f,s_f,u_{s_f}}$$

subject to

$$\left.\begin{aligned}&\sum_{v\in\mathcal{U}} Y_{f,u,v} \le 1, \quad \forall f \in \mathcal{F}, \forall u \in \mathcal{U}\\&\sum_{v\in\mathcal{U}} Y_{f,v,u} \le 1, \quad \forall f \in \mathcal{F}, \forall u \in \mathcal{U}\end{aligned}\right\} \text{Single path}$$

$$\left.\begin{aligned}&\sum_{v\in\mathcal{V}} X_{f,u,v} = \sum_{v\in\mathcal{V}} X_{f,v,u}, \quad \forall f \in \mathcal{F}, \forall u \in \mathcal{U}\\&X_{f,s_f,u_{s_f}} = X_{f,u_{d_f},d_f}, \forall f \in \mathcal{F}\end{aligned}\right\} \text{Flow conservation}$$

$$\sum_{f\in\mathcal{F}} X_{f,u,v} \le r_{u,v}, \quad \forall (u,v) \in \mathcal{E} \quad \} \text{ Link capacity limit}$$

$$X_{f,u,v} \le Y_{f,u,v} \cdot r_{u,v}, \quad \forall f \in \mathcal{F}, \forall (u,v) \in \mathcal{E} \quad \} \text{ Variable binding}$$

Task 1 : Add 3 constraints to satisfy #tranceiver constraint

Model (3/3)

- **Integer Linear Programming (ILP)**

$$\max \sum_{f \in \mathcal{F}} X_{f,s_f,u_{s_f}}$$

subject to

$$\sum_{v \in \mathcal{U}} Y_{f,u,v} \leq 1, \quad \forall f \in \mathcal{F}, \forall u \in \mathcal{U}$$

Single path

Task 2 : Solve ILP with OR-Tools

ion

$$X_{f,s_f,u_{s_f}} = X_{f,u_{d_f},d_f}, \forall f \in \mathcal{F}$$

$$\sum_{f \in \mathcal{F}} X_{f,u,v} \leq r_{u,v}, \quad \forall (u,v) \in \mathcal{E}$$

Link capacity limit

$$X_{f,u,v} \leq Y_{f,u,v} \cdot r_{u,v}, \quad \forall f \in \mathcal{F}, \forall (u,v) \in \mathcal{E}$$

Variable binding

Task 1 : Add 3 constraints to satisfy #tranceiver constraint

Design Heuristic Algorithm

Solving ILP by OR-tools may not run in polynomial time

Design your own algorithm(MyAlgo) to solve the problem

- Output of MyAlgo must be feasible
  - Meet all constraints in ILP
- MyAlgo should run in polynomial time
  - Do not use brute force
- The solution does not need to be optimal, but should outperform the baseline we provide
  - Baseline algorithm is an algorithm without considering load balance

Baseline Algorithm (Given)

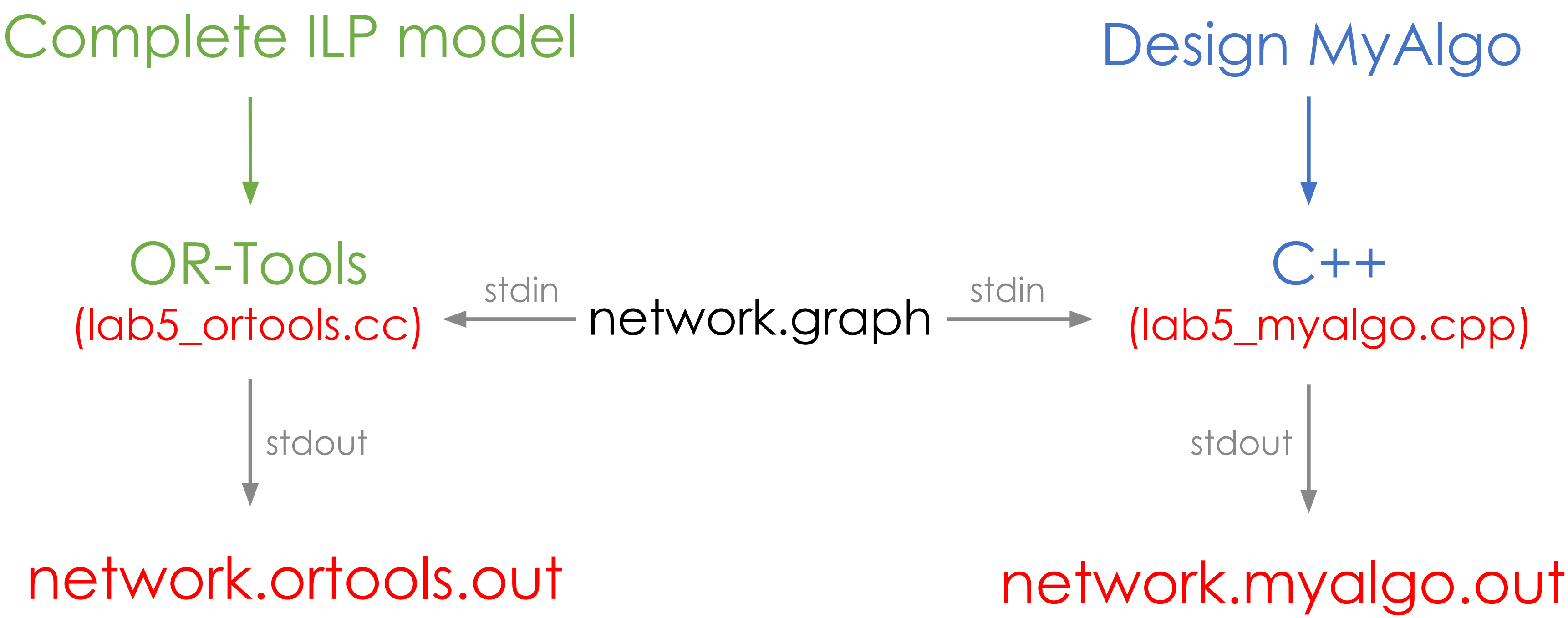
Just transmit the data along the shortest path

1. For each SD pair, find the shortest path (minimum hop count) by BFS
2. For each finding path:
   - If the path does not violate the Limited transmitter/receiver constraint
     - Transmit the maximum possible data rate along it
     - Update the residual link capacity
   - Otherwise, skip and consider the next path

Agenda

- Lab Overview
- Problem Definition
- Optimization Model
- **Tasks**
- Report & Result
- Submission

Task Overview



Task 1: Complete the ILP model

- Add 3 constraints to complete the model
  - Y-Z binding
    - Hint: Each link is **used (occupied)** if it is assigned to any SD pair
  - Single transmitter per node
    - Hint: Each node can use at most one outgoing link
  - Single receiver per node
    - Hint: Each node can use at most one incoming link

Task 2: OR-Tools Program

- Write your OR-Tools program
  - Write the code in lab5_ortools.cc based on the ILP model built in task 1
  - **Do not** submit OR-Tools program to codeforces

- We also provide sample.graph/sample.out for you to test your OR-tools program
  - Your output does not need to match sample.out exactly — however, the total throughput must be the same
  - That is, there could be multiple optimal solutions, but their objective values should all be optimum

Task 3: Design & Implement MyAlgo

- Design MyAlgo: a load balancing algorithm
  - The output total throughput must be better than baseline

- Implement MyAlgo in C++
  - Write the code in lab5_myalgo.cpp
  - Submit lab5_myalgo.cpp on codeforces
    - If you get an **Accepted**, the correctness of your code is verified
    - You will get the Wrong Answer if
      - Your output solution violates some constraints
      - The average throughput of all the testcases is not greater than baseline

I/O format

- Both OR-Tools and MyAlgo program should follow the same I/O format
  - Please see the statement on codeforces (you need to login first)

- Please store your program output into files
  - Input file: network.graph
  - Output files
    - OR-tools: network.ortools.out
    - MyAlgo: network.myalgo.out

- Your program can just use **stdin/stdout** and run the following command to do the file redirection

```
./program < input.txt > output.txt
```

Agenda

- Lab Overview
- Problem Definition
- Optimization Model
- Tasks
- **Report & Result**
- Submission

Report

- Name as report.pdf
- Explain how you implement your lab step by step for each commit version
- Questions
  1. Write down the 3 constraints you add in task 1 and briefly explain it
  2. Calculate the average throughput ratio between network.myalgo.out and network.ortools.out

$$\text{ratio} = \frac{\text{average throughput of network.myalgo.out}}{\text{average throughput of network.ortools.out}}$$

  3. Briefly explain the main idea of MyAlgo
  4. Analyze the time complexity of MyAlgo

Agenda

- Lab Overview
- Problem Definition
- Optimization Model
- Tasks
- Report & Result
- **Submission**

Submission

- Add your own studentID to studentID.txt (same as lab1)
- Push only the following specified files to GitHub
  - Please **do not** include any other files

```
.
├── lab5_myalgo.cpp
├── lab5_ortools.cc
├── network.graph
├── network.myalgo.out
├── network.ortools.out
├── report.pdf
├── sample.graph
├── sample.out
└── studentID.txt
```

Due

- **May. 22 (Thu.) 23:59, 2025**
- Don’t need to submit to E3
- Commit **your flies** to your Github repository
  - Should have at least **3 commits** by **yourself** (commit by github-classroom[bot] is not included)
  - One version should be at least **1 day** after another
- **Notice: You will get penalty with wrong file structure and naming**

Grading Policy

- Grade
  - Code correctness – 20%
  - Report – 50%
  - Result – 30%
- Late Policy
  - (Your score) * 0.8^D, where D is the number of days overdue
- Cheating Policy
  - Academic integrity: Homework must be your own – cheaters share the score
  - Both the cheaters and the students who aided the cheater equally share the score