Network Simulator: Final Presentation

Sharjeel, Chris, Ying-Yu, Emily

Agenda

- Architecture
- Routing
- Congestion Control
- Data Analysis and Plotting
- Test Cases

Tools

Python (2.7)

SimPy: process-based simulation in Python

Github: source control

Hackpad: issue tracking and documentation

Google Slides: presentation

Main Programs

Simulator

- Output logs as simulation progresses.

Analyzer

- Bins logs into intervals and do statistics.

Plotter

- Visualizes the above hard works.

Simulator

Device

- Physical objects e.g. Host, Link, and Router.
- Packet exchange via SimPy

Packet

- Messages e.g. Data, Ack, Routing
- Tells devices what to do on arrival

Flow

- Data transmission and congestion control

Device

Host

- Source holds Flow
- Target acks data packets

Router

- (will explain in detail later)

Link

- Full-duplex (equiv. to two half-duplex)
- Buffer, transmission time (~packet size), latency (const)

Packet

Methods

- reach_host()
- reach_router()

Rationale

- No if/elif/else type testing
- Separate high-level logic from SimPy codes in Device
- New routing algorithm => new packet class(es)

Agenda

- Architecture
- Routing
- Congestion Control
- Data Analysis and Plotting
- Test Cases

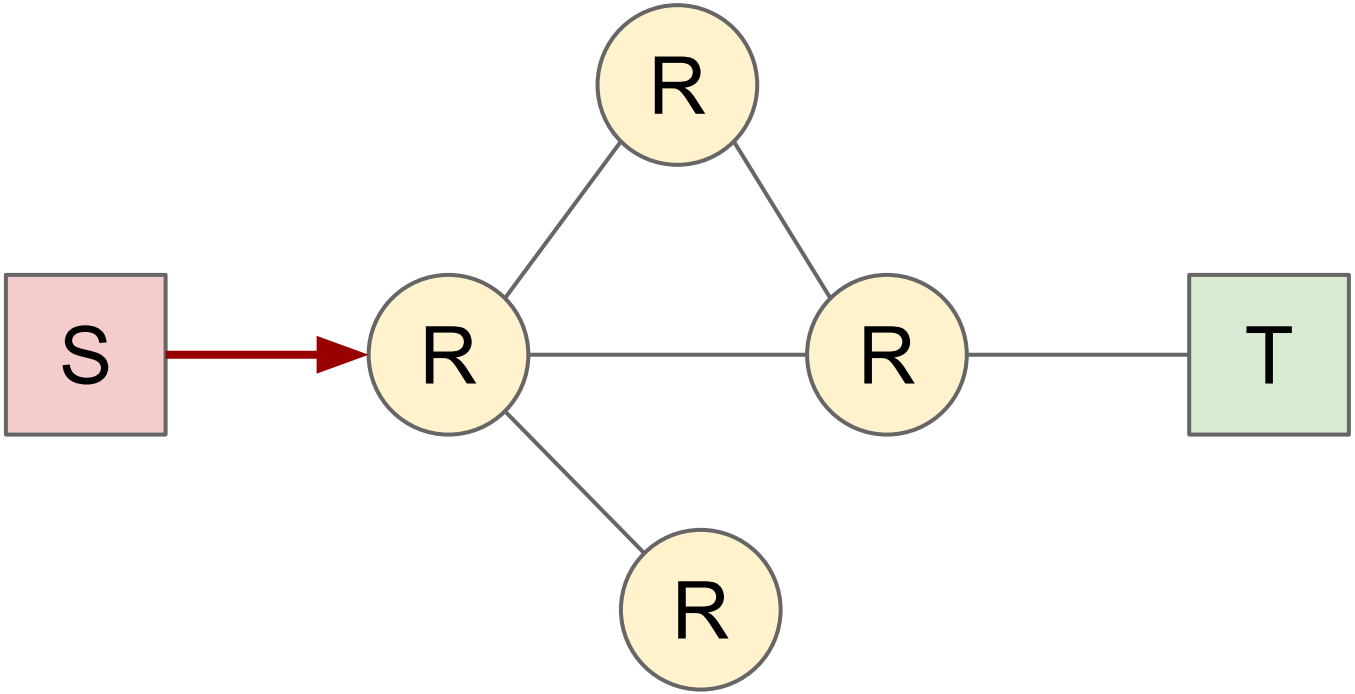
Bellman-Ford Algorithm

1. Packet starts at the router and goes to the host/router
2. Records the time it took to get there.
3. It goes back to the initial router and checks to see if that path was faster than other paths to that host or that router’s hosts

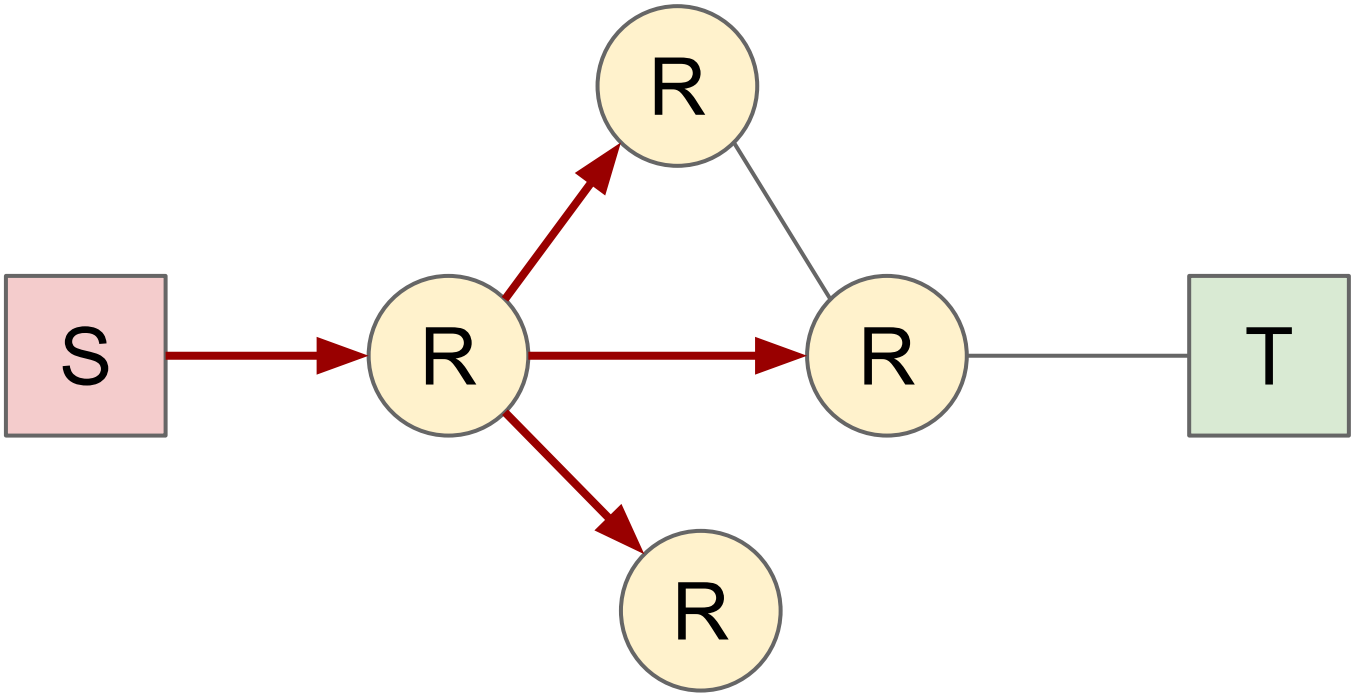
Dijkstra’s Algorithm

1. **Sonar from Source**
    - Explores the network in Dijkstra’s fashion
    - Builds reverse routing table
2. **Echo by Target**
    - Reverse routing table points to the source
    - Builds forward routing table along the way
    - Cost = one-way delay (source => target)

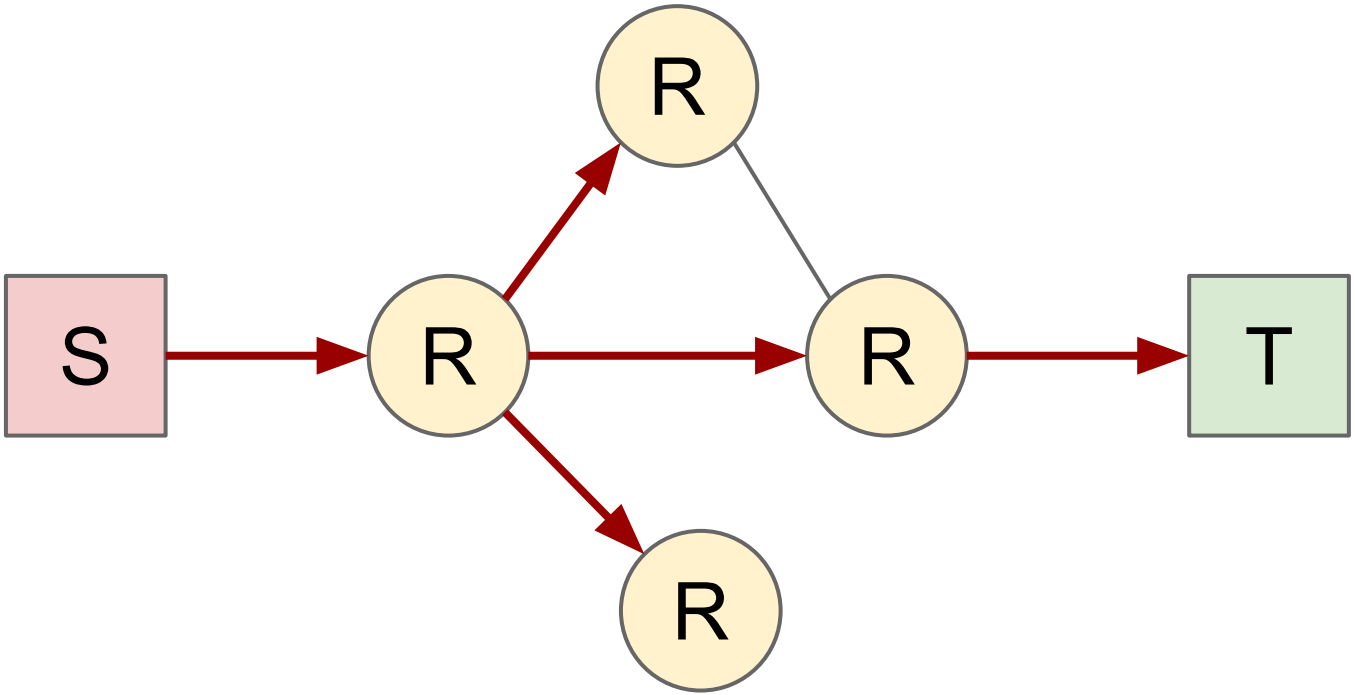
Sonar Phase



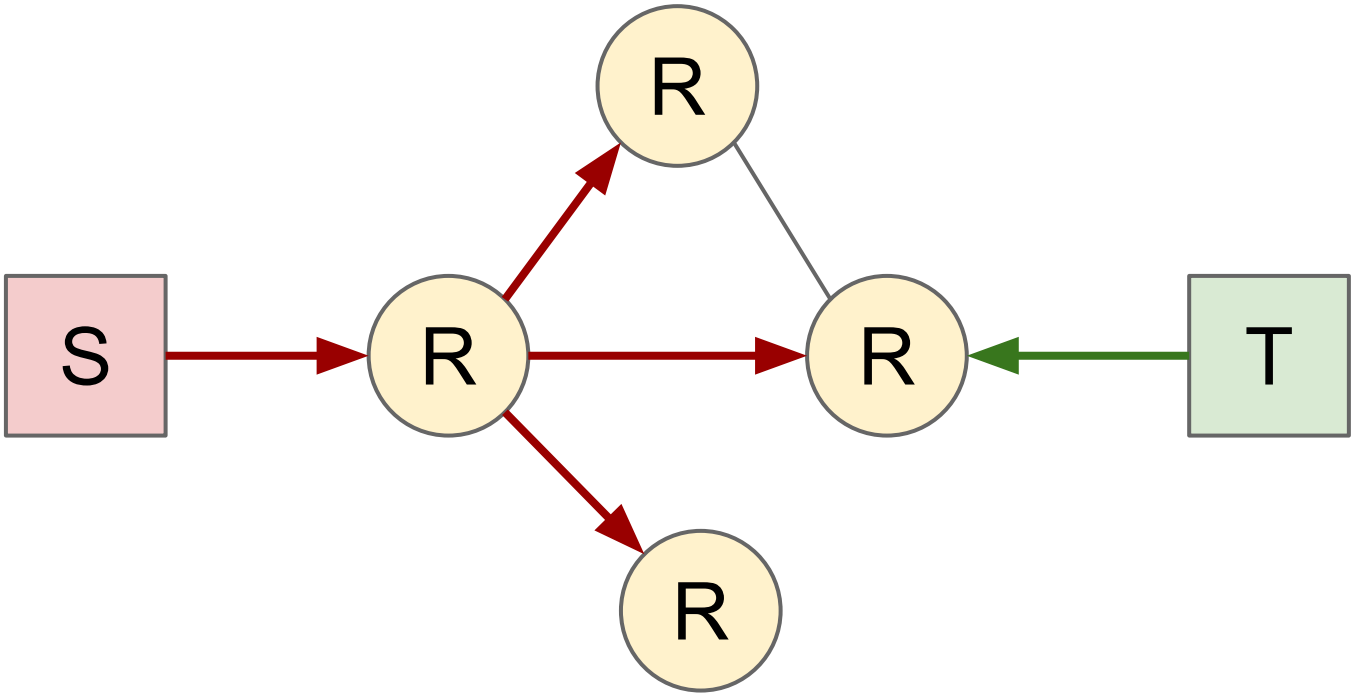
Sonar Phase



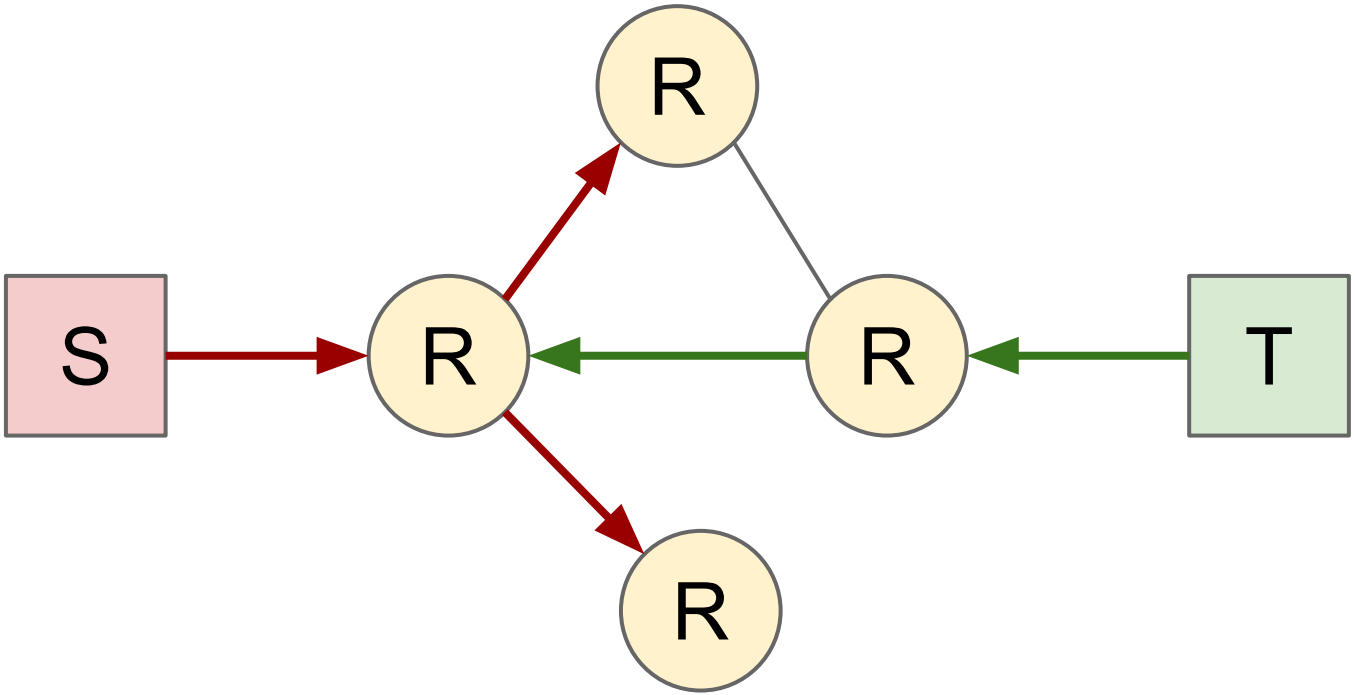
Sonar Phase



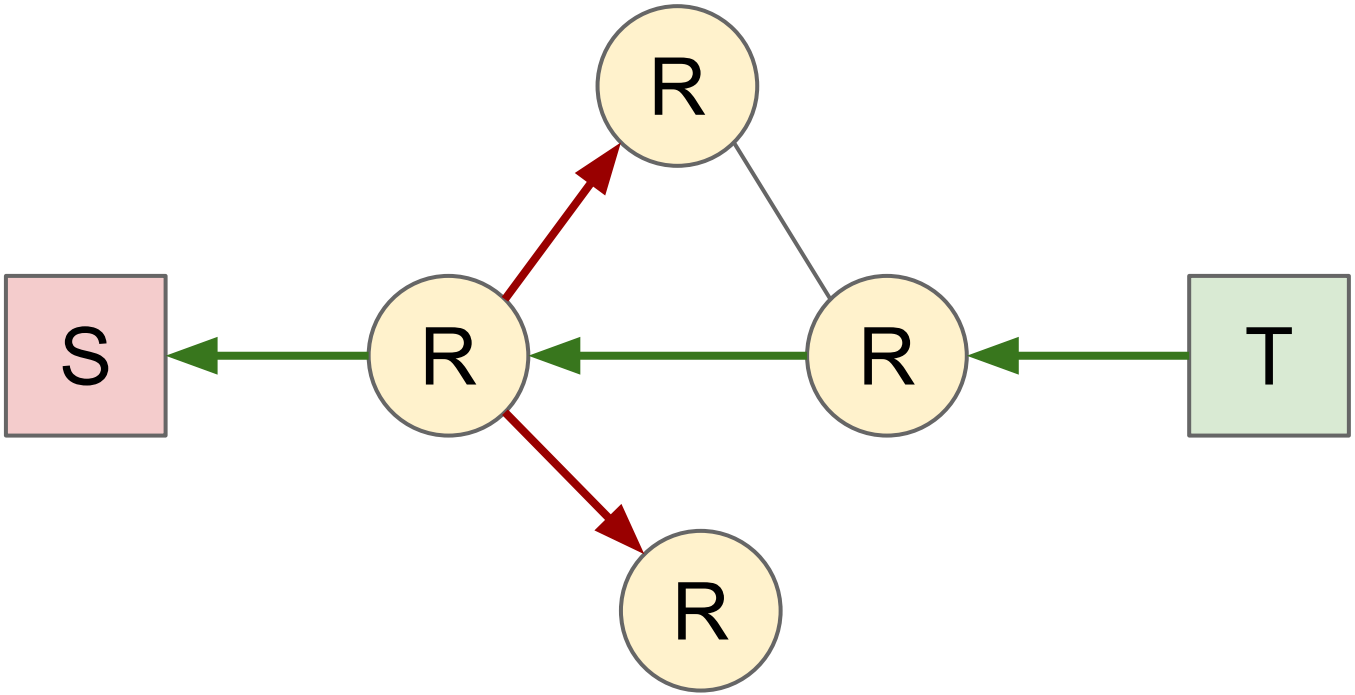
Echo Phase



Echo Phase



Echo Phase



Routing Strategies

- **Dijkstra**
  - Host starts the process
  - Fast network discovery
  - Measures the whole trip
- **Bellman-Ford**
  - Router takes the initiative (decentralized)
  - Reacts to individual links (fast)
  - Oscillations?

Agenda

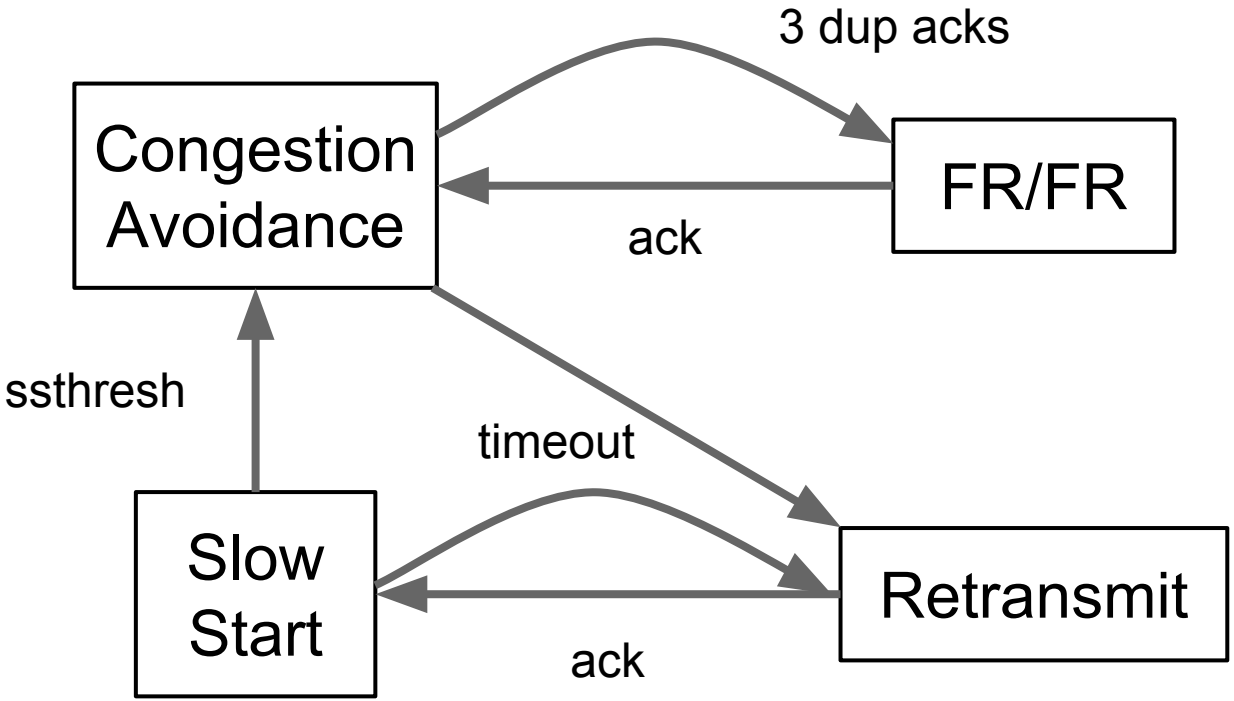
- Architecture
- Routing
- Congestion Control
- Data Analysis and Plotting
- Test Cases

Flow Functionality

- Send data packets sequentially
- Maintain transmission window
- Detect timeout
- Process good acks, dup acks

Complicated enough. To make things worse…

Flow States


3 dup acks
Congestion Avoidance
FR/FR
ack
ssthresh
timeout
Slow Start
Retransmit
ack

Lego-like Design

a.k.a. State Pattern

BaseFlow

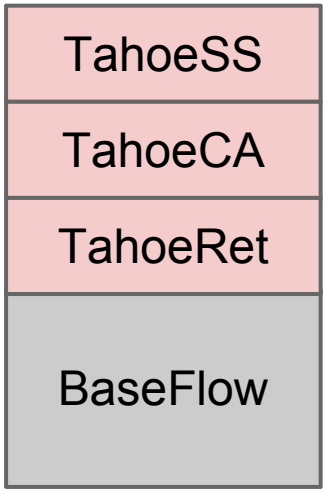
- Basic functionality + API
- Holds constructors for class FlowState

FlowState

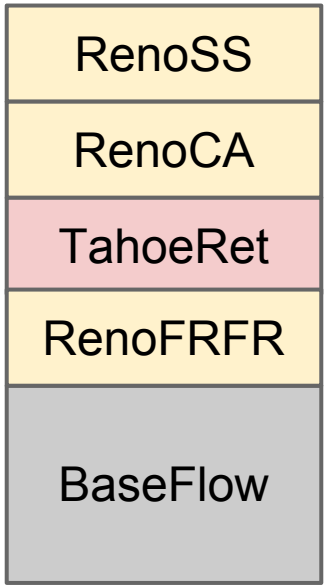
- Implements state-dependent event handlers
- State transition

Building Examples

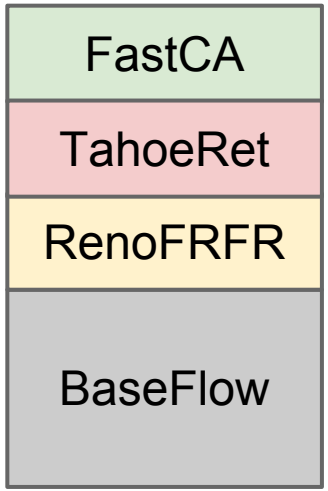
TCP Tahoe



TCP Reno



FastTCP



Congestion Control Algorithms

- TCP Tahoe

ACK: W += 1
LOSS: W = W/2

- TCP Reno

ACK: W += 1/W
LOSS: W = W/2

- FAST-TCP

$$W = \frac{\text{baseRTT}}{\text{RTT}} W + \alpha$$

- CUBIC-TCP

$$W_{cubic} = C(t-K)^3 + W_{\max}$$

Agenda

- Architecture
- Routing
- Congestion Control
- Data Analysis and Plotting
- Test Cases

Network Input

Sections

- Host
- Router
- Link
- Flow

Example ==>

S1

T1

-

R1

-

L0 S1 R1 12.5 10 64

L1 R1 T2 12.5 10 64

-

F1 S1 T1 20 0.5

Raw Output

Network simulator gives

- **send_data:** data packet leaves source
- **receive_data:** data packet reaches target
- **packet_loss:** packet dropped by link due to full buffer
- **packet_rtt:** RTT reported when source receives ack
- **buffer_diff:** packet enters/leaves link buffer
- **transmission:** packet enters cable section of link

Processed Output

Calculated from raw logs:

- **link rate**: [time] link_flow_rate [link_id] [flow_rate]
- **buffer occupancy**: [time] buf_level [link_id] [level_kb]
- **packet loss**: [time] pkt_loss [link_id] [loss_rate]
- **flow rate**: [time] flow_send_rate [flow_id] [rate]
- **window size**: [time] window_size [flow_id] [average_size]
- **packet delay**: [time] packet_rtt [flow_id] [average_delay]

Graphics

Tkinter

- General gui package for python

matplotlib

- Draw specialized charts

Networkx

- Graph drawing in gui generator

Agenda

- Architecture
- Routing
- Congestion Control
- Data Analysis and Plotting
- Test Cases

Demos

- Test case 0
- Test case 1
- Test case 2