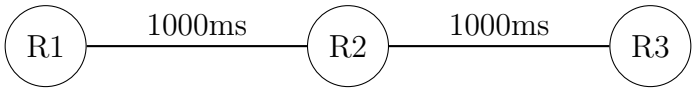
Problem Formulation

In this assignment, we will explore how to create forwarding tables for various network topologies. Here, the term network topology refers to how nodes in a network are connected (e.g., edges in the graph). We will make many simplifying assumptions in order to focus on the high-level concept of network topology. Here are some of the assumptions:

- Packet size is not considered. Assume the entire packet content is its destination.
- No packet loss. No variability in message transmission time.
- We will use integer IDs for each router instead of IP addresses.
- Router bandwidth is not considered (just latency; see next bullet).
- Latency is defined per edge. For instance, in the network below, the edge between R2 and R3 has a latency of 1000ms. We will assume this latency is fixed for each message being sent. We will also assume a router can only send one message at a time. Practically, if all edges have an 1000ms latency, then R2 can only transmit a message once per second.
- Forwarding tables are static. These will be set by you!

Unlike the prior assignments that used a real networking library (asio), for this assignment we will use a simple network topology simulator that I wrote. This assignment will have a significantly smaller coding portion as most of the code is written for you! (Hopefully, you don't mind having less code to write.) Please go here to download the starter code for this assignment. There is currently a simple network topology in `main.cpp` which is illustrated below.

Simple network with 3 nodes



R1 Table

Dest	Next Hop
R2	R2
R3	R2

R2 Table

Dest	Next Hop
R1	R1
R3	R3

R3 Table

Dest	Next Hop
R1	R2
R2	R2

Each `Router` is represented as a class instance in the simulator. For example, creating R1 in code is done as follows:

```
Router r1(1, &manager, {{2, 1000}});
```

where the `1` is the router id (R1), `manager` is a class in charge of carrying out the simulation, and the final argument are the edges (links with delays) for R1. For routers with multiple links (such as R2) there are multiple key-value pairs for links and delays (e.g., `{{1, 1000}, {3, 1000}}`).

The forwarding tables are then set up for each router matching the above tables as follows:

```
r1.set_forwarding_tables({{2, 2}, {3, 2}});
r2.set_forwarding_tables({{1, 1}, {3, 3}});
r3.set_forwarding_tables({{1, 2}, {2, 2}});
```

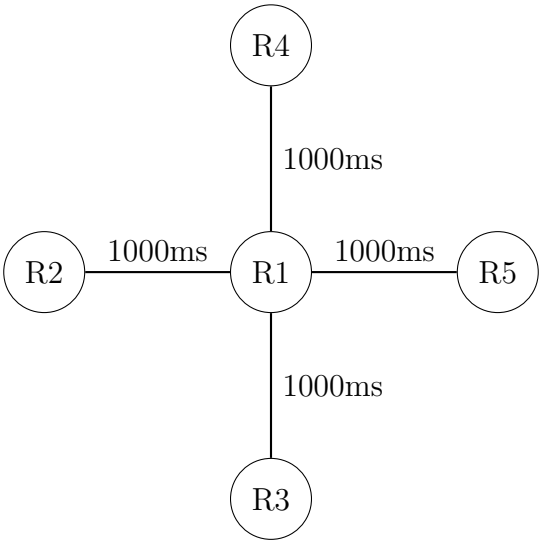
After setting up the network and forwarding tables, `main.cpp` sends two messages (packets) through the network via the `Manager`, which is in charge of the simulation. These messages are printed at each hop along the network. Once all the messages have been received by the destination, the program exits. You will observe that two messages are received at R2 at the same time, but only one message is forwarded at a time. For simplicity, we use the link delays to regulate how frequently each router can transmit a message.

Assignment Objective

In this assignment, you will need to do the following:

- Create a function in main to carry out simulations for each of the topologies given below. For instance, you may call the function `client_server` for the first topology.
- Determine the forwarding tables for each topology using the link-state routing algorithm. This can be done by hand or programmatically - your choice.
- Add some timing code to determine how long it takes for messages to propagate through the network using your forwarding tables.
- Send the supplied messages through the network (look at `main.cpp` for an example). These should all be sent at the same time (as is the case in `main.cpp`).
- In addition to the code, submit a write-up file (e.g., `writeup.txt`) that explains why these running times make sense to you based on the derived forwarding tables and simulator assumptions.

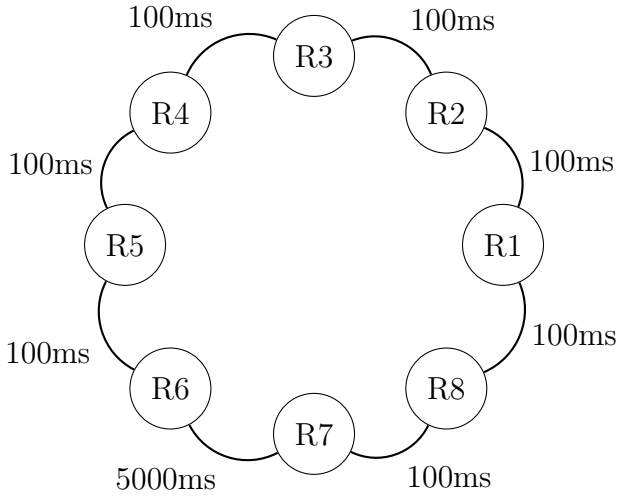
Client-Server Topology



Messages to send:

- R2 to R1
- R5 to R1
- R4 to R1

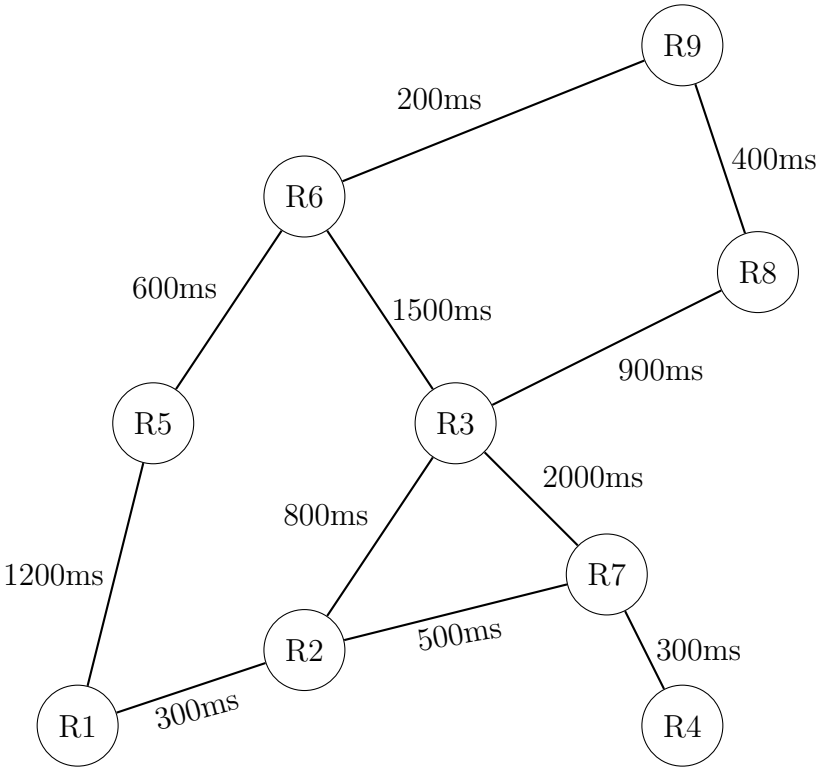
Ring Topology



Messages to send:

- R1 to R3
- R8 to R4
- R6 to R7

Complex Network



Messages to send:

- R8 to R6
- R1 to R9
- R4 to R5

Submission

When you are finished, zip up your project (excluding the `build/` directory) into `assignment4.zip` and submit via Canvas. Make sure it includes a `writeup.txt` file that discusses the running times of each topology based on your forwarding tables and the network latencies.