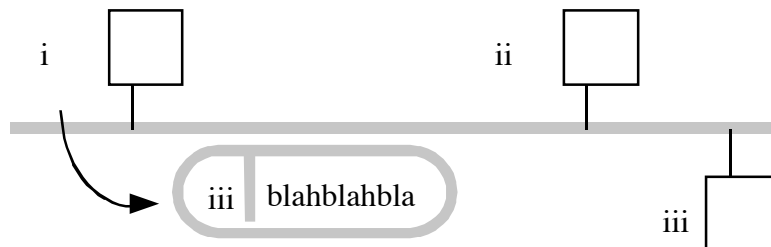
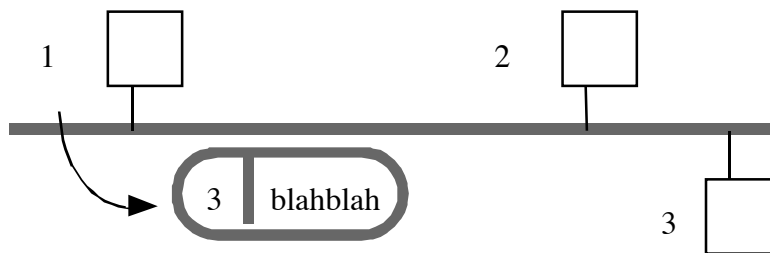
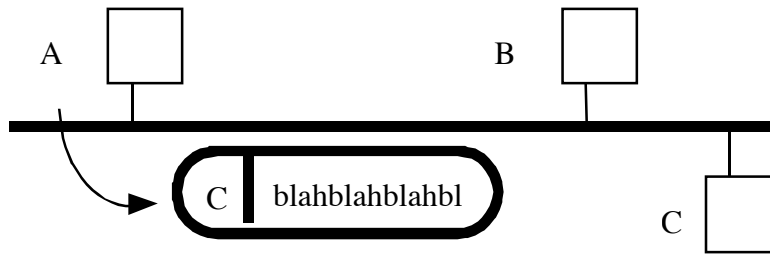


# IP Routing

## Babel

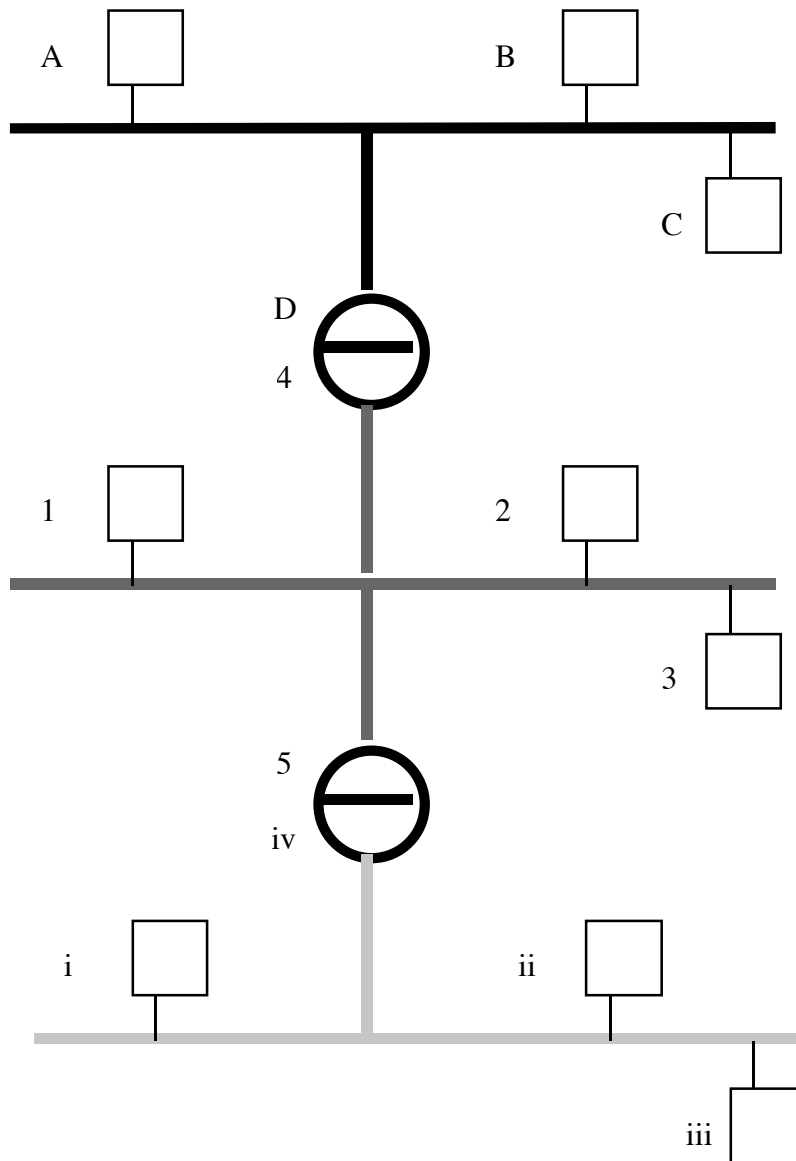
Once upon a time, there were three separate “physical” LANs, each with their own addressing scheme, protocol, and format. And host computer “A” was very sad, because it was unable to communicate with its dearest love in the world, host “iii” which was on a different LAN network.



*Three separate physical networks, black, dark gray, and light gray. Each with their own packet formats, packet sizes, and addressing schemes.*

## Routers

So one day, the networks decided to connect with a couple routers. Each router is like a translator capable of participating in either of the two networks to which it is connected.

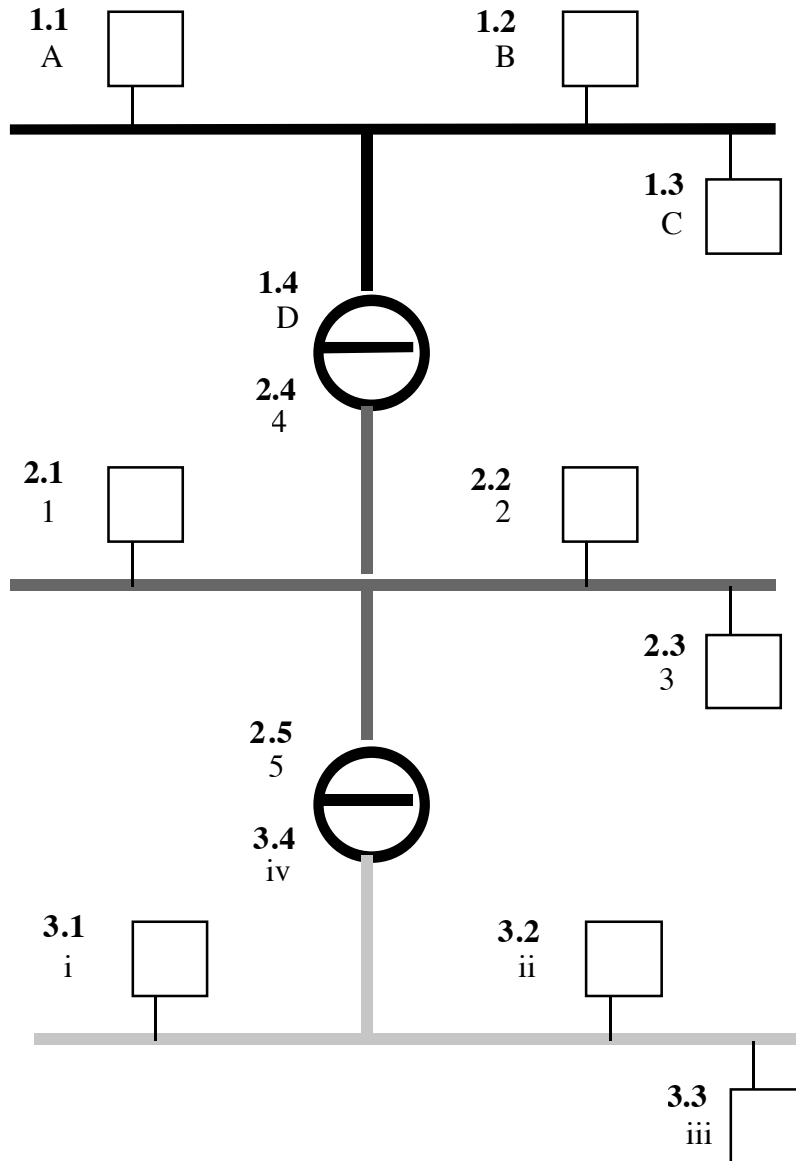


*Introduce two routers. Each router is on two physical networks— each router has an address and can send and receive packets on both networks. Need a consistent naming scheme to identify hosts on the various networks.*

## IP Addresses

The first step in allowing all the of the different hosts to communicate with each other is to introduce a uniform naming scheme so that every host can have a well defined name. Give every host an IP address which identifies the network and host. Every host has a “physical” name, such as “A” which is used by its LAN, as well as a global standard IP name such as “1.1”. The left part of the IP address identifies the network and the right part identifies the host on that network. For simplicity, we will use only two-byte IP

addresses. Remember that real addresses are of the form W.X.Y.Z, where each part is a byte-sized number that spans 0 to 255.

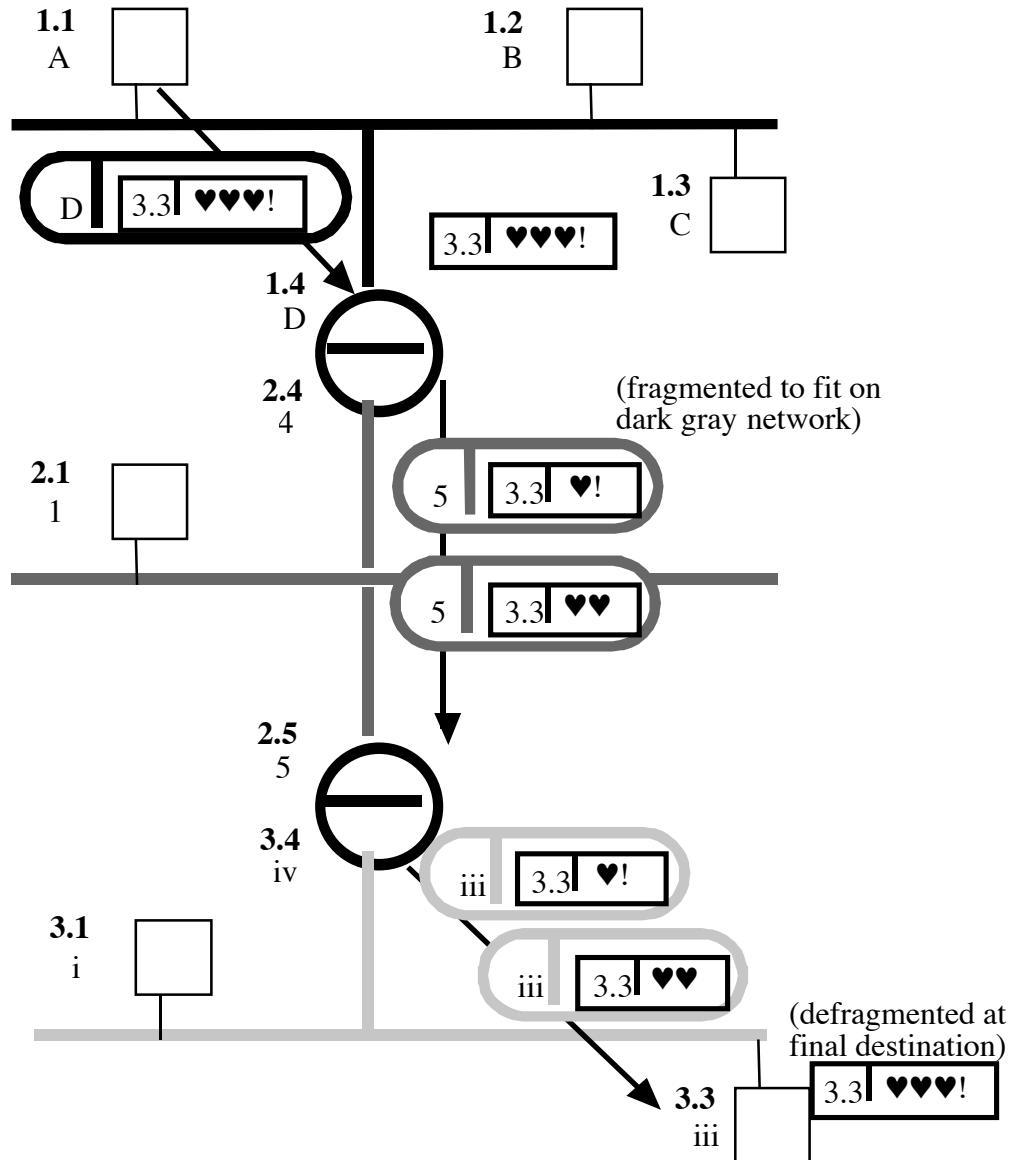


*Give each host and router an IP address for each of its connections. The IP addresses are consistent across all the networks. Each IP address identifies the host and the network it is on.*

Note that in the figure above, each router now has two IP addresses that map to it, and two LAN addresses. This is important, because each router has to be addressable by the hosts on its local network. For example, computer “A” cannot talk to a router named “4” because it does not understand that naming scheme. Thus, it must talk to the router named “D.” Fortunately, “D” and “4” are one and the same router.

## IP Datagram

Now define the "IP Datagram" — a standard packet format. The IP datagram is forwarded hop by hop **inside** the packets of each LAN. At each hop, the IP datagram is unpacked from the physical packet that transported it. Now host "A" (aka 1.1) is able to send a love note to host "iii" (aka 3.3), and they all lived happily ever after.



*Host "A" wishes to send a love note to host "iii". Introduce the IP datagram as a common unit of transfer. IP datagrams use IP addresses. The datagram always keeps the IP address of its ultimate destination. Hosts and routers forward the IP datagram one hop at a time towards its destination. On each hop, the datagram is encapsulated inside a physical network packet with a physical network address. The datagram may need to be "fragmented" if it is too big to fit in a single physical network packet.*

## Low Level

- The IP datagram and IP addressing and routing schemes are umbrella standards that all the hosts and routers use. At a small scale, hosts and routers use their physical networks to send packets hop by hop. IP is the overall, universal standard.
- Hosts and routers use the IP address for everything, and mostly just the network part of the IP address. The physical address of the final destination is only computed on the very last hop.
- The original IP datagram may be fragmented along the way, but it always keeps the IP address of its final destination. All routing decisions are in terms of IP addresses.
- The datagram is defragmented only when it reaches the final destination (a design decision which helps keep the routers simple).
- Each host and router is only concerned with what the "next hop" is for a particular IP address. The overall design tries to keep the hosts simple, leaving the complexity for the routers. So for example, the top-left **1.1** host would keep a "next hop" table which looked like..

<u>IP addr of destination</u>	<u>Next Hop IP addr</u>
1.*	Same phys. network as me-- "direct connection" compute physical address and send directly
2.*	1.4
3.*	1.4

Every host and router needs another (much smaller and simpler) table which maps IP addresses to physical address for things on its physical network

<u>IP addr</u>	<u>Physical Addr</u>
1.1	A
1.2	B
1.3	C
1.4	D

- Routers have the same sort of "next hop" table, but theirs are typically more complex since they know about paths for farther away networks. The routing tables are partly determined by the administrator for the router, and partly the routers have their own complex protocol they use dynamically to exchange routing information with each other. IP routing dynamically adapts to traffic patterns and network availability.

For this example, the routing table for the upper router (1.4/2.4) would look like....

<u>IP addr of destination</u>	<u>Next Hop IP addr</u>
1.*	Direct connection
2.*	Direct connection
3.*	2.5

## Miscellaneous

- Flow Control – so far we have looked at routing. Another problem IP tries to solve is "flow control" – control the sending rate of the sender so that the recipient (and all the routers along the way) can cope with the flow of packets. There is a reverse flow of tiny “Acknowledge” packets sent from the destination back to the source – these help throttle back the sender if it is going too fast.
- Business School Thesis Topic – the free market is not good at coming up with standards such as TCP/IP, and yet they appear to be quite valuable. It's hard for the vendors to cooperate to make a standard, even when it might make them all better off.
- “Datagram Service” – just a best effort delivery service to send one packet
- “Connection Oriented Service” – an error-free stream of data from the sender to the recipient.
- TCP/IP and sockets are built on datagrams, but they try to give the appearance of connection oriented to the higher layers. TCP/IP re-assembles and error-checks the packets at the recipient to re-create the in-order stream of data at the recipient. The flow will still be "bursty" however since really it's built on packets.