Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

FS 2019

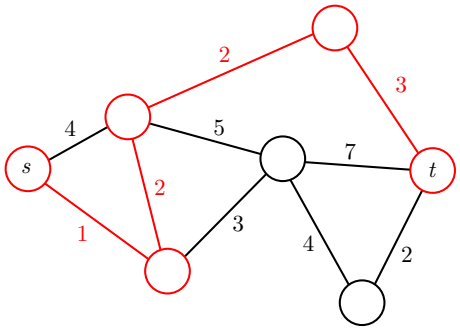
Prof. R. Wattenhofer

Computer Engineering II

Solution to Exercise Sheet Chapter 2

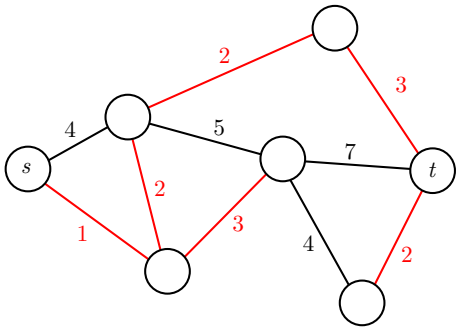
1 Quiz Questions

a) $1+2+2+3=8$:



In general, a shortest path can be found using Dijkstra's algorithm.

b) $1+2+2+2+3+3=13$:



In general, a minimum spanning tree (MST) can be found using Kruskal's algorithm.

c) That is indeed possible. However, the `127.*.*.*` (and not only `127.0.0.1`!) is reserved to be used as a *loopback address*. Thus, if a computer A is assigned the address `127.0.0.1` no other computer will be able to reach A; i.e., if a computer B sends packets to `127.0.0.1`, all those packets will directly be routed back to B itself (without ever leaving B).

d) Any network interface that supports IPv6 has always at least one IPv6 address: The *link-local* IPv6 address. To distinguish them from other IPv6 addresses, they always start with the prefix `fe80::/64`. The maximum number of addresses is OS-specific and can be checked in `/proc/sys/net/ipv6/conf/default/max_addresses` on Linux, often it is 16.

e) There are many use-cases, this is a non-exhaustive list:

- The computer is running virtual machines that have to be reachable using own addresses.
- The computer is acting as a proxy/gateway for other machines connected on a second interface.
- The computer administrator has set up virtual interfaces, and different applications use different virtual interfaces.

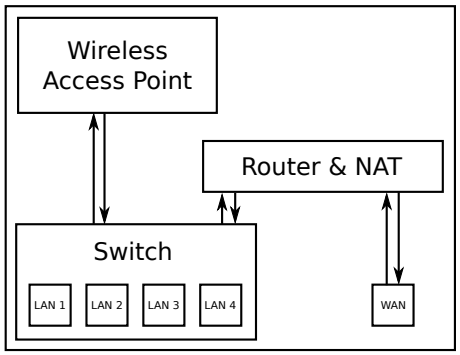
2 Basic IP Networking

a) Routers, as presented in the lecture, are devices which are between different subnets. They operate on the network layer (OSI layer 3, see *OSI Model*), i.e., they work with IP addresses.

A similar device is a *switch*; however, a switch operates on a lower level (OSI layer 2), i.e., does not work with IP addresses. Hence, a switch *switches* packets *within* a subnet.

A typical "home router" is a multi-functional device: It includes a router, but usually also a switch, a wireless access point, a NAT component, ...

A high level view of such a device is as follows:



b) Depending on your operating system and installed software there are many possibilities. In Linux you can for example run `ip address` on the console, which will you list the IP addresses of all available interfaces.

c) There are many ways to determine this address:

- In a typical home network, all traffic that is not sent to a destination within the network is sent "outwards". In order for you computer to know where "outwards" is, your computer has an entry called *default gateway*. This entry is used if there is no other route available for a packet. As the "router" is responsible for those packets, this address is the address of your "router". You can inspect the routing table to determine the *default gateway*. On Linux you can determine the default gateway e.g. with `ip route`.
- Your home router typically runs a DHCP server to assign IP addresses. Hence, the router address will be the same as the one which runs the DHCP server. On Linux you can determine the DHCP server e.g. using `dhclient -v` (needs root).

d) No, it is not a good idea. If you use `8.8.8.*` as local addresses, you will not be able to access the computers which legitimately using these addresses. I.e., you have a collision: There are more than one machine which is using the same address. Why do you not run into any problems with e.g. `192.168.0.*`? Because these addresses are reserved for the very purpose of being used as local network addresses, and no server on the Internet will ever use these addresses.

e) No, the server will see the public address of the home router. Many routers will display this address somewhere in the user interface. However, it is often easier to access a website which tells you the IP address with which it got accessed – for example, google "what's my IP".

3 Using an SSH Tunnel

If you run Linux, you can run `ssh -D 9988 nethz_name@slab1.ethz.ch`, which will create the tunnel binding at the local port 9988, and configure your browser such that it uses a SOCKS proxy on this port. In Firefox for instance, you have to go to `Preferences` → `Advanced` → `Network` → `Settings`, select manual proxy configuration and enter the address 127.0.0.1 and the port 9988 in the *SOCKS Host* fields. Please refer to the many short tutorials available on the Internet for further information.

4 Addressing

a) There are three subnets in the network: [1,2], [3,4,5], [6]:

Nodes 1 and 2 share the same subnet address, because the first 17 bits are the same. We can write the subnet address of nodes 1 and 2 in binary as: 0100 1110 0000 0011 1XXX XXXX XXXX XXXX.

Similary, the subnet address of nodes 3,4 and 5 in binary is: 0100 1110 0000 0011 0XXX XXXX XXXX XXXX.

Finally, the subnet address of node 6 is: 0100 1110 0000 0100 0XXX XXXX XXXX XXXX.

b) Yes, it is indeed possible. If we change the prefix length of all the nodes to 13 or less, it is clear from the binary addresses that all nodes would have the same subnet address: 0100 1110 0000 0XXX XXXX XXXX XXXX XXXX.

c) The TTL value is reduced by one after each hop. The source and destination address do not change. In this way we would have:

Pacekt	Source address	Destination address	TTL
1	78.3.128.1/17	78.3.3.6/17	16
2	78.3.3.6/17	78.3.128.1/17	14

d) The corresponding IPv6 addresses of the nodes are:

1: ::ffff:4e03:8001/113

2: ::ffff:4e03:8102/113

3: ::ffff:4e03:7f03/113

4: ::ffff:4e03:7f04/113

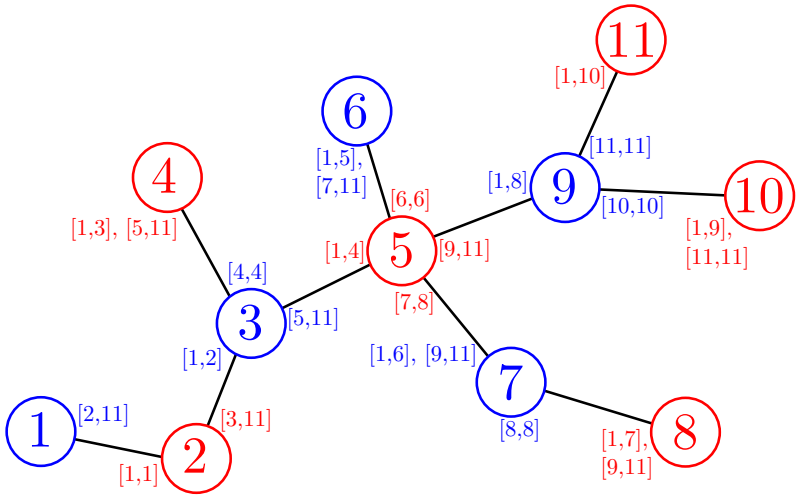
5: ::ffff:4e03:306/113

6: ::ffff:4e04:305/113

e) Following the compression rules in the script, the address bc12:1:b:2::4 can be expanded to bc12:0001:000b:0002:0000:0000:0000:0004.

5 Simple Routing

a) We assigned the addresses as follows (the red/blue coloring is just to improve readability). Note that we also added the intervals stored for each node/edge to the respective node/edge:



b) We can assign the addresses by performing a depth-first search (DFS) on the graph, and we assign addresses to the nodes in the order in which we visit them.

Let us show that assigning the addresses in DFS-manner will require us to put at most two intervals on each edge. For that, we look at any arbitrary node which will be visited during the DFS, and we will argue that the claim holds.

Let u be the addresses of the node, which is assigned to it once it is visited. The DFS continues to assign addresses by following all edges of u that have not yet been visited. The first node in the first subtree is assigned $u+1$. Once the subtree is completely assigned, the DFS returns to u. We can then look at the largest address in that subtree, which we call v. Since all nodes in this subtree are given addresses with values from $u+1$ to v, we can simply use that as the interval for this outgoing edge of u. This argument also applies to all subsequent subtrees, except that the first address of the following subtree is not $u+1$, but $v+1$, where v is the maximum assigned address in the previously visited subtree.

After assigning addresses to all subtrees and adding the according interval, there is only one edge which has no intervals yet: The one on which u was reached when the DFS reached u for the first time. We can put (at most) two intervals to that edge: All addresses in $[1, u-1]$ (they have been assigned before u was reached), and all addresses in $[v_{\max}+1, \infty]$, where $v_{\max}$ is the maximum address assigned in the subtree rooted at u (these addresses will be assigned afterwards).

Remark: The assignment of the labels is not unique: It depends on the sequence in which the nodes are visited during the DFS - but the reasoning applies to all possible orders, as long as the DFS is executed correctly.

6 Routing Loops

a) In RIP, the distance vectors are sent to the neighbors every 30 seconds and the network has a maximal diameter of 15 hops. Therefore, it can take a few minutes until every node knows about the change in the network. Additionally, the messages are sent using UDP and can therefore get lost.

b) A node knows it is part of an STRL when it sees the same packet a second time. (Or if it forwards the packet back to the previous node.)

c) An STRL gets resolved when the new link-state has been propagated to all nodes that are part of the STRL and the routing tables have been recomputed.

d) The link between the nodes v_1 and v_2 fails. v_2 sends packets for v_1 to v_3 and v_3 sends them back. Routing tables v_2 and v_3:

Routing table of v_2	
Destination	Next node
v_1	v_3
v_2	deliver
v_3	v_3
v_4	v_4

Routing table of v_3	
Destination	Next node
v_1	v_2
v_2	v_2
v_3	deliver
v_4	v_2

e) Yes. In a graph where a link failure leads to an STRL, an additional node can be added in this link. If this node fails, it has the same effect as if the original link failed. E.g., you can add a node v_5 between v_1 and v_2 with links of costs 0.5 to v_1 and v_2.