Chapter 3

Communication

Contents

3.1 Computer networks

3.1.1 Network protocol stacks

A **protocol stack** is an implementation of a **protocol suite**, which defines the precise set of rules by which computers communicate over a network. Network protocols are modeled as a stack of **layers**, each designed for a specific purpose. There are two primary models of network protocols in common use. The **internet** (aka TCP/IP) model consists of four layers:

4. **Application layer**
3. **Transport layer**
2. **Internet layer**
1. **Link layer**

The **OSI** (Open Source Interconnection) model consists of seven layers:

7. **Application layer**
6. **Presentation layer**
5. **Session layer**
4. **Transport layer**
3. **Network layer**
    a) Subnetwork Access
    b) Subnetwork Dependent Convergence
    c) Subnetwork Independent Convergence
2. **Data link layer**
    - Logical link control (LLC) sublayer
    - Medium access control (MAC) sublayer
1. **Physical layer**
    physical signaling sublayer
    - Physical coding

3.1.2 Wired network topologies

The **topology** of a "wired" computer network (interconnected by electrical cables or optical fibers) is a design criterion that may be tailored to best suit the specific applications for which the system is intended, such as:

- internet connectivity of general-purpose computers,
- centralized fault-tolerant coordination of machines,
- distributed computation of multidimensional PDEs (via spectral, finite difference, or finite element methods),
- weather and climate forecasting [using ensemble methods for forecast uncertainty quantification (UQ)],
- navigation and manipulation of transactional databases (airline ticket sales, large-scale search, …),
- deep learning for medical diagnostics,
- DNA sequencing, etc.

Links between computers (aka nodes) in any such network can be half-duplex (able to maintain communication in one direction at a time only) or full-duplex (able to maintain communication in both directions simultaneously), with the connecting cables attaching to the nodes via a network interface controller (NIC). If links are physically too long to reliably deliver a signal, one or more repeaters can be used. Possible logical network topologies[1] include the following:

A) The prototypical example of a physically-dedicated **point-to-point** link is a tin-can telephone. Circuit-switching technology, as used in conventional telephony, allows temporary dedicated point-to-point electrical connections to be set up when needed in settings incorporating many nodes.

B) Each pair of nodes in a **daisy chain** is connected via a cable with a NIC at each end; if a received message is not intended for that node, it is simply retransmitted down the chain. A **linear** (with two ends) or **ring** configuration can be used for such a chain; a ring provides, at modest additional cost, an alternate direction to pass any given message, which is useful if the other direction is substantially longer, broken, or busy.

C) Each node may be connected (via a NIC) to a single central **bus** (aka backbone or trunk), and all data carried on the bus can simultaneously be received (or, ignored) by any connected node. A **linear bus** has two endpoints; a **distributed bus** has branches, and thus multiple endpoints. All endpoints of a bus must be terminated (see §??) to prevent reflections.

D) The sending of signals out to all nodes in a **star** configuration is coordinated by a central hub [in which an input signal on one port is repeated as an output signal on all other ports] or switch [in which an input signal (say, on port A) is routed only towards its specific destination (say, on port B), allowing simultaneous communication traffic between the various other ports as needed (say, from port C to D, etc.)].

E) At least some nodes in a **mesh** network have more than two NICs, and thus can themselves act simultaneously as both nodes and switches (note that certain nodes at high-traffic junctions may be replaced by dedicated switches). This topological class is very versatile. A small network with n nodes can be:

E.1) **fully connected**, with a point-to-point link between every pair of nodes [that is, with $n-1$ NICs per node, and $(n-1)!$ total links…which quickly becomes unmanageable for fairly small n], or

E.2) a single d-dimensional (aka dD) **hypercube**, with $n = 2^d$ nodes, d NICs per node, and $d\,2^{d-1}$ total links; with this paradigm, all nodes are within d "hops" from any starting node in the network.

For larger n, a mesh forms some sort of (partially connected) d-dimensional interconnect **grid**, such as:

E.3) A d-dimensional **cartesian grid**, usually with periodic connections in each coordinate direction (thus dubbed a dD **torus**), with $n = n_1 n_2 \cdots n_d$ nodes, and 2^d NICs per node. At the cost of more NICs per node, an dD grid has two important performance advantages over a 1D ring. First, for unstructured data flow, a key performance metric is how many additional nodes are reached per "hop" from any starting

[1] Of course, physically, such networks are usually laid out quite differently (leveraging standardized server racks, etc.).

node in the network; after $r \gg 1$ hops, the number of new nodes reached with one more hop is roughly proportional to r^{d-1} (that is, a dD grid spreads data faster with increased d). Second, with its more numerous, well-structured pathways, an dD grid can more quickly "transpose" a large multidimensional grid that is distributed over the cluster[2]. Note also that circuit switching (see topology A) can be used at times to temporarily partition a single periodically-connected grid into a number of smaller periodically-connected grids to better run smaller jobs.

E.4) An d-dimensional **noncartesian grid**, also with periodic connections, formed by a rare sphere packing (see *RP*), with fewer NICs per node for a given dimension d than a cartesian grid[3]. Examples include the 2D uniform hexagonal tiling, with 3 NICs per node, the 3D diamond packing D_3^+, with 4 NICs per node, and the D_d^+ **hyperdiamond** packing (a d-dimensional generalization), with $d+1$ NICs per node.

E.5) An **unstructured 2D grid** interconnecting many computers that are sparsely separated over a large physical area, such as a factory floor.

Networks may also be arranged logically as a **hybrid** combination of the above basic topologies, such as:

E.6) An **extended star**, given by a star (topology D) connecting to additional stars.

E.7) A **tree**, given by a high-speed bus (aka trunk; topology C) connected to stars (aka branches).

E.8) A **spoked ring**, given by a ring (topology B) with a central switch connected (as in a star) to a sparse subset of nodes distributed around the ring. The ring facilitates fast simultaneous nearest-neighbor communications (in 1D), and the spokes facilitate fast communication farther over the network.

E.9) A **spoked grid**, generalizing the spoked ring to a d-dimensional grid (topology E.3, E.4, or E.5), with a central switch connected to a sparse subset of nodes distributed over the grid, to facilitate both fast simultaneous nearest-neighbor communications (in dD), and fast communication over longer distances.

Illustrations of several such logical network topologies are given in Figure 3.1. Note that topologies E.6 through E.9 are especially well suited for algorithms that have a coordinating "central" node that needs especially fast access to all other "compute" or "machine coordination" nodes.

A well-designed structured grid network (topology E.3 or E.4 above) with $2h$ NICs per node and $n \gg 1$ nodes sometimes has embedded within it h **nonoverlapping Hamiltonian circuits**; that is, h entirely nonoverlapping pathways that reach every other node in the network; a couple of examples are illustrated in Figure 3.1, which can be useful for certain data sharing tasks within a network. Consider, for example, a difficult **all-to-all** data transfer problem, in which each node has a certain (large) amount of data that needs to be transferred to all other nodes. Splitting the data to be transferred on each node into h equal-sized pieces and directing each piece along one of the Hamiltonian circuits (from each node simultaneously) gets all of the data where it needs to go in exactly n hops, utilizing each communication link in the network with maximum efficiency.

Once one of the several above network topologies is selected, and set up correctly, the medium access control (MAC) sublayer of the network protocol stack (see §3.1.1) handles all of the low-level rules for determining which links to use to actually route packets across the network in any given situation. This significantly streamlines the coding task for the embedded programmer, requiring simply the calling of the appropriate one-to-one, one-

[2]Consider, for example, a high-resolution x-y-z discretization of a 3D field defined over a cube using a 2D cartesian network topology. Each node in the network can contain the discretized values of the field at all z gridpoints (for a certain range of x and y gridpoints), which substantially accelerates numerical algorithms involving implicit solves or FFTs in the z coordinate (only). Transferring data in one set of directions over the 2D network allows one to quickly perform a sort of matrix transpose, putting onto each node the discretized values of the field at all y gridpoints (for a certain range of x and z gridpoints), thus facilitating fast implicit solves or FFTs in the y coordinate; transferring data in the other set of directions puts onto each node the discretized values of the field at all x gridpoints (for a certain range of y and z gridpoints), thus facilitating fast implicit solves or FFTs in the x coordinate.

[3]The cost of mesh network is proportional to the number of NICs per node. Note that 4 NICs per node form a 2D cartesian or 3D diamond grid, and 6 NICs per node form a 3D cartesian or 5D hyperdiamond grid, clearly favoring noncartesian grid topologies for certain applications (particularly, those with unstructured message passing and a very large number of nodes).

?

Figure 3.1: Illustration of several logical network topologies.

to-all, or all-to-all data transfers in the numerical code, and leaving it to the protocol stack to sort out which links to use to actually complete the requested data transfer.

3.1.3 Ad hoc wireless networks

3.2 Short-range wired communication protocols

blah.

3.2.1 USB

3.2.2 I2C / I3C

3.2.3 SPI / QSPI

3.2.4 UART / USART

3.2.5 PWM

PWM (Pulse Width Modulation) can be used for both driving an H-Bridge directly, and signalling to a Servo or Electronic Speed Controller (ESC).

3.3 Long-range wired communication protocols

3.3.1 RS485

3.3.2 CAN

3.3.3 Ethernet

3.3.4 Infiniband

(medium range)
(alternative to Ethernet for HPC)
InfiniBand

3.4 Wireless communication protocols

review

3.4.1 RFID / NFC

3.4.2 Bluetooth / BLE

generation	standard	adoption	frequency band(s)	MIMO	max datarate
Wi-Fi 1	**802.11b**	1999	2.4 GHz		11 Mbps
Wi-Fi 2	**802.11a**	1999	5 GHz		54 Mbps
Wi-Fi 3	**802.11g**	2003	2.4 GHz		54 Mbps
Wi-Fi 4	**802.11n**	2009	2.4 and 5 GHz	✓	600 Mbps
Wi-Fi 5	**802.11ac**	2014	5 GHz	✓	6.933 Gbps
Wi-Fi 6 / 6E	**802.11ax**	2019	2.4 and 5 GHz / 2.4, 5, and 6 GHz	✓	9.607 Gbps

Table 3.1: Commonly used variants of the Wi-Fi standard.

3.4.3 Wi-Fi

Commonly used variants of the Wi-Fi standard are listed in Table 3.1; note that the maximum practical throughput that an application can expect to achieve is about 53% of the max data rate using TCP, and about 64% of the max data rate using UDP.

Wall-powered Wi-Fi routers operating at 2.4 GHz are typically effective up to about 46 m indoors and 92 m outdoors, whereas routers operating at 5 GHz are typically effective over only about a third of these distances, though they can be pushed to significantly higher data rates. Unfortunately, many other household products operate in the 2.4 GHz band, including Bluetooth (see §3.4.2), microwave ovens, and baby monitors. Due to such (often, frustrating) interference issues on the 2.4 GHz band, Wi-Fi 4 and later protocols do not rely exclusively on the 2.4 GHz band. Notably, Wi-Fi 6 and 6E specifically address channel congestion and interference issues, as well as significantly reducing the power required by client devices.

3.4.4 3G/4G/5G cellular

3.4.5 Satellite

3.4.6 Zigbee / Zwave

3.4.7 LoRa / SigFox

LoRaWAN, Symphony Link

3.4.8 LPWAN / NBIOT / LTE-M