# Lecture notes: Interconnection networks

Master M1: Parallel Algorithms and Programming

2022

This lecture presents interconnection network for parallel distributed memory systems. It introduces the main characteristics of interconnection networks and presents some major network topologies.

## 1 Introduction

We consider a distributed memory system: each processor has its own private memory; processor communicate by exchanging messages. In such a system, the processor are connected through an interconnection network that is in charge of transferring the data between the processors.

An interconnection network is composed of:

- **Endpoints:** Endpoints are the sources and destinations of messages, in our case processors.
- **Switches:** A switch is a device connected to a fix set of links. It transmits a received packed to one or multiple links.
- **Links:** A *wire* that is used to transfer data between endpoints and switches, and between switches.

Note that in the case of distributed memory systems, the nodes (endpoints) are equipped with a network interface card that is in charge of sending data to and receiving data from the network on behalf of the main processor.

There are two main kinds of interconnection networks: Direct and Indirect.

- **Direct network:** In a direct network, the nodes *sit inside* the network. It means that a node is both a endpoint and a switch.
- **Indirect network:** In an indirect network, endpoints are connected indirectly via switches.

Both direct and indirect networks are used in today's systems, even if indirect networks are way more common. IBM Blue Gene machines are examples of *recent* supercomputers that use a direct network interconnect.

Direct network are also called *static* as the nodes are physically connected through static physical links. On the other hand, indirect networks are called *dynamic* as the connections between nodes are build dynamically at runtime through the configuration of the switches.

The *network topology* describes the structure used to interconnect the processors, the switches and the links.

The interconnection network is a major topic in parallel systems because data movements are the most costly operations in such systems in terms of performance.

**Note about shared-memory systems** In this lecture, we consider distributed memory systems. However the concept of interconnection network also applies to shared-memory systems. Indeed, in a multicore processor a *Network-on-Chip* (NoC) is used to move data between the cores, and to interconnect the cores and the memory sub-systems (memory controllers and caches).

**Routing techniques** In addition to the topology, the routing techniques are important properties of a network. Namely, the routing technique defines how a message goes from a source to a destination. It includes three main aspects:

- **The routing algorithm**: The routing algorithm determines the path from the source to the destination. The algorithm can be deterministic or adaptive. Adaptive routing algorithms try to take into account network contention when deciding on the path for a message.
- **The switching strategy:** The switching strategy determines how a message is transmitted along a selected path. It specifies the properties of the network with respect to the following questions: Is the message divided into packets? Are the packets routed independently? What is the buffering strategy at the level of switches? etc.
- **The flow control mechanisms:** The flow control mechanisms should deal with the following questions: How to deal with the case where multiple messages should be sent over the same link? How to ensure the reliable delivery of messages despite packets loss? How can a node in the network inform other nodes that it currently does not have space in its memory buffers to receive new packets?

Studying routing techniques in details is outside the scope of this lecture.

## 2 Static networks

### 2.1 Network characteristics

A static network can be represented as an undirected graph where the vertices are the nodes and the edges are the network links. The graph is *undirected* because, in general, messages can be sent in both directions at the same time over a network link. The *distance* between two nodes of the graph is the length of the shortest path between the two nodes. The length is the number of edges on the path from the source to the destination. The interconnection network ensures that there is a path between any two nodes in the system.

Some properties of the graph are important to study to analyze the characteristics of the network. Among those properties, we are mostly interested in the 2 following ones:

- **Diameter:** The diameter ($D$) of a graph is the maximal distance between any two nodes in the graph.
- **Bisection bandwidth:** The bisection bandwidth ($B$) is the minimum number of edges that must be removed to partition the network into two sub-networks of equal size.

These two properties are important metrics of the performance that can be achieved by the interconnection network. The diameter gives information about the maximum latency that can be observed on the network when there is no contention. The bisection bandwidth tells us about the maximum throughput that can be achieved when $n/2$ communications are initiated simultaneously in a system included $n$ processors. Note that to compute the effective bisection bandwidth, one should multiply the number of links by the capacity of the links.

An efficient network is one for which the diameter is small and the bisection bandwidth is large. However, in practice, the designer has to find a good trade-off between optimizing these properties and the cost of implementing the network. The *degree* of the graph, *i.e.*, the maximum number of incident edges for vertices in this graph, is a good metric of the hardware cost for a given network.

## 2.2 Static networks

There are several famous static network topologies: ring, clique, mesh, torus, hypercube, etc. We are studying some of them in this section.

**Ring** Figure 1 shows an example of a ring topology. The main properties of a ring topology for a network including $n$ nodes are:

- Degree $= 2$
- Diameter $= \lfloor \frac{n}{2} \rfloor$
- Bisection bandwidth $= 2$

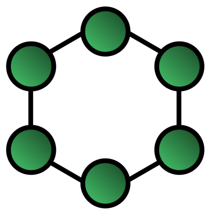


Figure 1: A ring topology

The ring topology is not costly to implement but it cannot be used for large scale systems (*i.e.*, when $n$ is large), as the diameter increases with the number of nodes while the bisection bandwidth does not.

**Fully-connected network** Figure 2 shows a fully connected network (also called *clique*). The main properties of this network, assuming $n$ nodes, are:

- Degree $= n - 1$
- Diameter $= 1$
- Bisection bandwidth $= \lfloor \frac{n}{2} \rfloor \times \lceil \frac{n}{2} \rceil$

This is the optimal network from performance point of view. However, it is very costly to implement as the number of links increases quadratically with the number of nodes. It is not usable in practice for large scale systems.

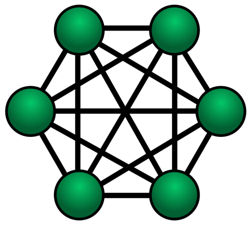


Figure 2: A fully-connected network

**D-dimensional mesh** Figure 3 shows an example of a 2-D mesh. Meshes with more dimensions (mostly 3-D) may be implemented in practice. The main properties of a 2-D mesh are:

- Degree $= 4$
- Diameter $= 2 \times (\sqrt{n} - 1)$
- Bisection bandwidth $= \sqrt{n}$

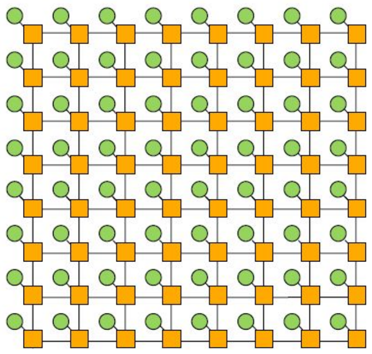


Figure 3: A 2-D mesh

Note that Figure 3 provides a clear representation of a static network (direct network) where each node in the graph is at the same time an endpoint (the processors – green circles) and a switch (yellow squares).

A 2-D mesh is well suited to run computations that correspond to 2-D space problems and where communication are mostly *local*, such as processing images. However for other problems, it can be difficult to deal with nodes at the edges of the network that have a lower degree of connectivity.

**D-dimensional torus** A D-dimensional torus is similar to a D-dimensional mesh except that additional links interconnect the first and the last nodes in each dimension. Figure 4 shows an example of a 2-D torus. The main properties of a 2-D torus are:

- Degree $= 4$

- Diameter $= 2 \times \lfloor \frac{\sqrt{n}}{2} \rfloor$
- Bisection bandwidth $= 2 \times \sqrt{n}$

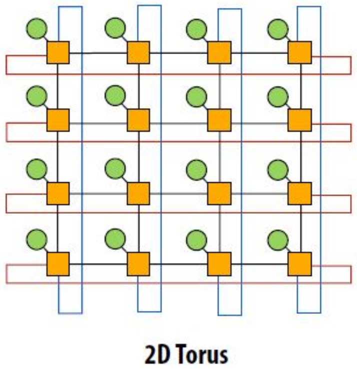


Figure 4: A 2-D torus

Compared to a mesh, a torus topology offers a smaller diameter (halved) and a larger bisection bandwidth (multiplied by two) for almost the same cost in terms of network links. D-dimensional Torus have good scalability properties, and thus, have been used in several very large scale parallel systems.

## 3 Dynamic networks

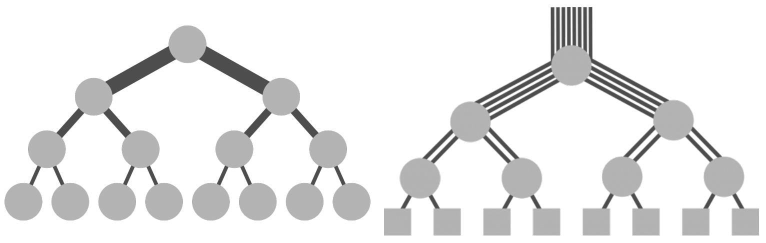
In a dynamic network (indirect network), the processors are not connected directly but communicate through intermediate switches. Several topologies exists for indirect networks: Crossbar, Dragonfly, Fat-Tree, etc. In this course, we focus on the popular Fat-Tree topology.

### 3.1 The Fat-Tree topology

A Fat-Tree topology is presented in Figure 5. A Fat-Tree topology is a tree topology. The processors are the leaves of the tree and all other nodes in the tree are switches. The main characteristic of a Fat-Tree is that each switch in the Fat-Tree has the same number of links going up and going down in the topology, as depicted in Figure 5b. This can be understood as the links getting *fatter*, *i.e.* having a larger capacity, towards the root of the tree, as depicted in Figure 5a.

A Fat-Tree such as the ones described in Figure 5 can be costly to implement in practice because the switches towards the root of the tree becomes larger, and such high-end switches have a very high cost. An alternative solution exists. It only relies on a set of switches having all the same capacity. Such a solution is illustrated in Figure 6. Note that such a solution is actually a folded Clos topology. Discussing more generally about Clos topologies goes beyond the scope of this course.

Fat-Trees are extensively used in parallel systems (supercomputer and data centers) because it can theoretically ensure that any permutation traffic can traverse the network at maximum bandwidth. As such it can implement any communication pattern very efficiently.



(a) Aggregated view (b) Detailed view

Figure 5: A Fat-Tree topology

Another important property of Fat-Trees is that multiple paths exist from any source to any destination, which makes such a topology fault tolerant.

The network described in Figure 6 is a 2-level Fat-Tree. It means that the tree features two levels of switches. Such a Fat-Tree is very popular in practice because of its relatively low cost from hardware point of view. However, the maximum size of such a network is limited by the number of ports of the individual switches in use. To increase the size of the network, more levels can be added to the tree, but the cost in terms of network cables and number of switches also increases.

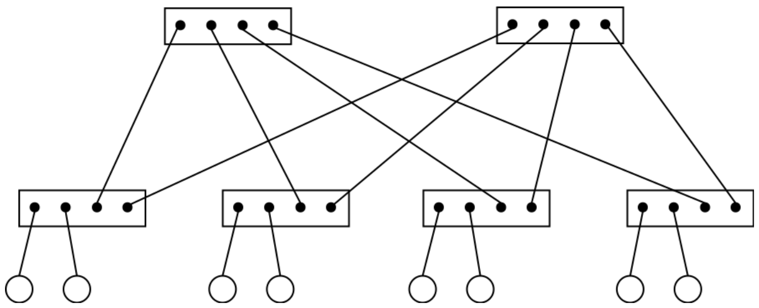


Figure 6: A 2-level Fat-Tree

Here are some properties of a 2-Level Fat-Tree topology built out of switches with $k$ ports:

- Number of switches at the upper level $= \frac{k}{2}$
- Total number of switches $= k + \frac{k}{2} = \frac{3 \times k}{2}$
- Max number of nodes $n = k \times \frac{k}{2} = \frac{k^2}{2}$

For such a 2-level Fat-Tree, the diameter and the bisection bandwidth of the corresponding graph are:

- Diameter $= 4$
- Bisection bandwidth $= \lfloor \frac{n}{2} \rfloor$

Such a network has good properties: a low diameter and a high bisection bandwidth. Some recent works have proposed topologies that can reduce even more the diameter of the network (*e.g.*, dragonfly network), but studying such networks goes beyond the scope of this course.

## References

Some references can complement the material presented in these lecture notes:

- Section 2.5 of the book "Parallel Programming: For Multicore and Cluster Systems" (by Rauber and Runger) about direct and indirect network topologies.
- Section 3.1 of the book "Parallel Algorithms" (by Casanova, Robert, and Legrand) also about direct and indirect network topologies.
- The web page `https://packetpushers.net/demystifying-dcn-topologies-clos-fat-trees-part2/` for a discussion on Fat-Trees.