# 2 Introduction to Network Theory

In this section we will introduce some popular families of networks and will investigate how well they allow to support routing.

## 2.1 Basic network topologies

The most basic network topologies used in practice are trees, cycles, grids and tori. Many other suggested networks are simply combinations or derivatives of these. The advantage of trees is that the path selection problem is very easy: for every source-destination pair there is only one possible simple path. However, since the root of a tree is usually a severe bottleneck, so-called *fat trees* have been used. These trees have the property that every edge connecting a node $v$ to its father $u$ has a capacity that is equal to all leaves of the subtree routed at $v$. See Figure 1 for an example.

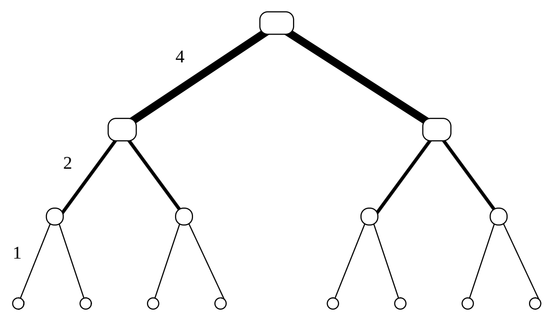


Figure 1: The structure of a fat tree.

Fat trees belong to a family of networks that require edges of non-uniform capacity to be efficient. Easier to build are networks with edges of uniform capacity. This is usually the case for grids and tori. Unless explicitly mentioned, we will treat all edges in the following to be of capacity 1. In the following, $[x]$ means the set $\{0, 1, \ldots, x-1\}$.

**Definition 2.1 (Torus, Mesh)** *Let* $m, d \in \mathbb{N}$. *The* $(m,d)$-mesh $M(m,d)$ *is a graph with node set* $V = [m]^d$ *and edge set*

$$E = \left\{ \{(a_{d-1} \ldots a_0), (b_{d-1} \ldots b_0)\} \mid a_i, b_i \in [m],\ \sum_{i=0}^{d-1} |a_i - b_i| = 1 \right\} .$$

*The* $(m,d)$-torus $T(m,d)$ *is a graph that consists of an* $(m,d)$*-mesh and additionally wrap-around edges from* $(a_{d-1} \ldots a_{i+1} (m-1)\ a_{i-1} \ldots a_0)$ *to* $(a_{d-1} \ldots a_{i+1}\ 0\ a_{i-1} \ldots a_0)$ *for all* $i \in [d]$ *and all* $a_j \in [m]$ *with* $j \neq i$. $M(m,1)$ *is also called a* line*,* $T(m,1)$ *a* cycle*, and* $M(2,d) = T(2,d)$ *a* $d$-dimensional hypercube.

Figure 2 presents a linear array, a torus, and a hypercube.

The hypercube is a very important class of networks, and many derivatives, the so-called *hypercubic networks*, have been suggested for it. Among these are the butterfly, cube-connected-cycles, shuffle-exchange, and de Bruijn graph. We start with the butterfly, which is basically a rolled out hypercube.

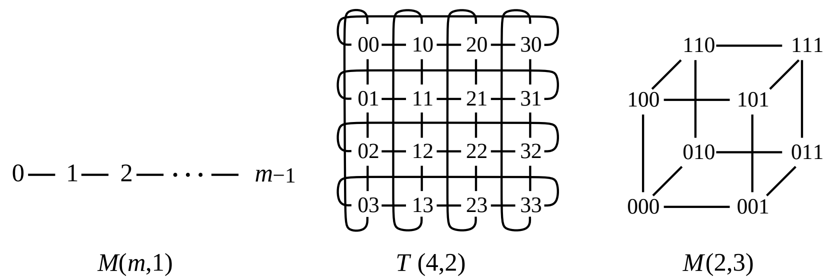


Figure 2: The structure of $M(m,1)$, $T(4,2)$, and $M(2,3)$.

**Definition 2.2 (Butterfly)** *Let $d \in \mathbb{N}$. The $d$-*dimensional butterfly $BF(d)$ *is a graph with node set $V = [d+1] \times [2]^d$ and an edge set $E = E_1 \cup E_2$ with*

$$E_1 = \{\{(i,\alpha),(i+1,\alpha)\} \mid i \in [d],\ \alpha \in [2]^d\}$$

*and*

$$E_2 = \{\{(i,\alpha),(i+1,\beta)\} \mid i \in [d],\ \alpha,\beta \in [2]^d,\ \alpha \textit{ and } \beta \textit{ differ only at the } i\textit{th position}\}\ .$$

*All node set $\{(i,\alpha) \mid \alpha \in [2]^d\}$ is said to form* level $i$ *of the butterfly. The $d$-*dimensional wrap-around butterfly *W-BF$(d)$ is defined by taking the $BF(d)$ and identifying level $d$ with level 0.*

Figure 3 shows the 3-dimensional butterfly $BF(3)$. The $BF(d)$ has $(d+1)2^d$ nodes, $2d \cdot 2^d$ edges and degree 4. It is not difficult to check that combining the node sets $\{(i,\alpha) \mid i \in [d]\}$ into a single node results in the hypercube.

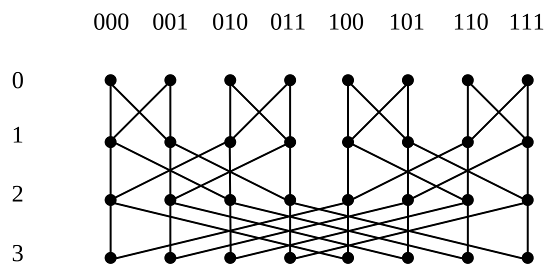


Figure 3: The structure of BF$(3)$.

Next we define the cube-connected-cycles network. It only has a degree of 3 and it results from the hypercube by replacing the corners by cycles.

**Definition 2.3 (Cube-Connected-Cycles)** *Let $d \in \mathbb{N}$. The* cube-connected-cycles *network CCC$(d)$ is a graph with node set $V = \{(a,p) \mid a \in [2]^d, p \in [d]\}$ and edge set*

$$E = \Big\{\{(a,p),(a,(p+1) \bmod d)\} \mid a \in [2]^d, p \in [d]\Big\} \cup \Big\{\{(a,p),(b,p)\} \mid a,b \in [2]^d, p \in [d], a = b \text{ except for } a_p\Big\}$$

Two possible representations of a CCC can be found in Figure 4.

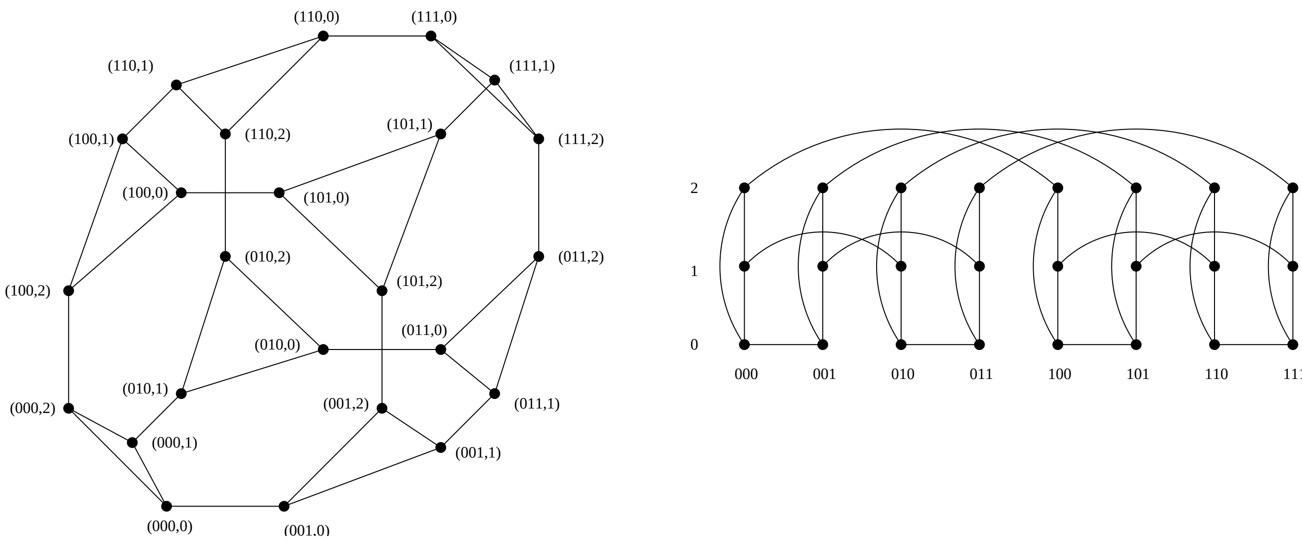


Figure 4: The structure of CCC$(3)$.

The shuffle-exchange is yet another way of transforming the hypercubic interconnection structure into a constant degree network.

**Definition 2.4 (Shuffle-Exchange)** *Let* $d \in \mathbb{N}$. *The* $d$-dimensional shuffle-exchange $SE(d)$ *is defined as an undirected graph with node set* $V = [2]^d$ *and an edge set* $E = E_1 \cup E_2$ *with*

$$E_1 = \{\{(a_{d-1} \dots a_0), (a_{d-1} \dots \bar{a}_0)\} \mid (a_{d-1} \dots a_0) \in [2]^d,\ \bar{a}_0 = 1 - a_0\}$$

*and*

$$E_2 = \{\{(a_{d-1} \dots a_0), (a_0 a_{d-1} \dots a_1)\} \mid (a_{d-1} \dots a_0) \in [2]^d\} \ .$$

Figure 5 shows the 3- and 4-dimensional shuffle-exchange graph.

**Definition 2.5 (DeBruijn)** *The* $b$-ary DeBruijn graph of dimension $d$ $DB(b,d)$ *is an undirected graph* $G = (V, E)$ *with node set* $V = \{v \in [b]^d\}$ *and edge set* $E$ *that contains all edges* $\{v, w\}$ *with the property that* $w \in \{(x, v_{d-1}, \dots, v_1) :\ x \in [b]\}$, *where* $v = (v_{d-1}, \dots, v_0)$.

Two examples of a DeBruijn graph can be found in Figure 6.

## 2.2 Direct and indirect networks

Networks are usually separated into *direct* and *indirect* networks. Direct networks are networks in which every node represents a processing unit that can inject and absorb packets, whereas in indirect networks only certain nodes (the so-called *input nodes*) can inject packets and certain nodes (the so-called *output nodes*) can absorb packets.

A very broad class of direct graphs are the so-called vertex-transitive graphs.

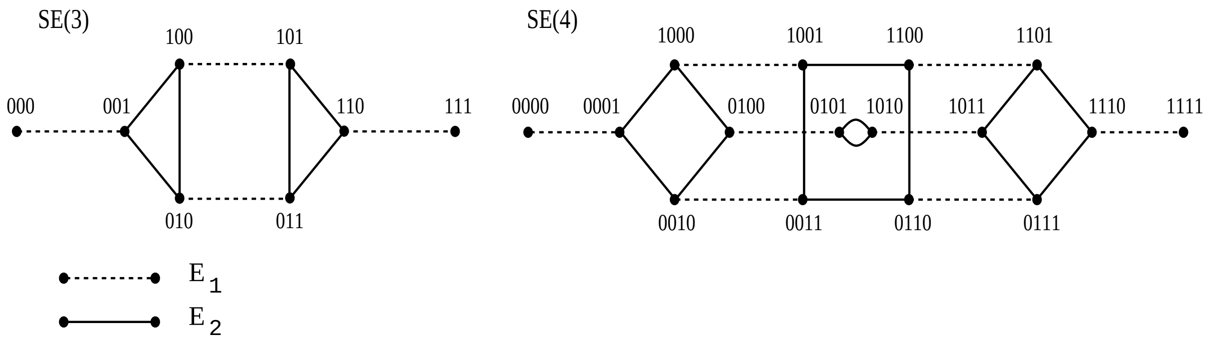


Figure 5: The structure of SE$(3)$ and SE$(4)$.

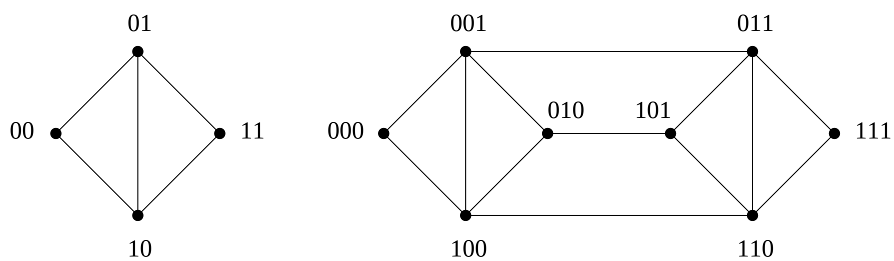


Figure 6: The structure of $DB(2, 2)$ and $DB(2, 3)$.

**Definition 2.6 (Vertex-Transitive Graph)** *A graph $G = (V, E)$ is called vertex-transitive if for any pair of nodes $u, v \in V$ there exists an isomorphism $\varphi : V \to V$ with $\varphi(u) = v$ such that the graph $G_\varphi = (V, E_\varphi)$ with $E_\varphi = \{\{\varphi(x), \varphi(y)\} \mid \{x, y\} \in E\}$ is equal to $G$.*

Intuitively, vertex-transitivity means that a graph looks the same from any node. Vertex-transitive graphs form a very general class and include many of the standard networks such as the $d$-dimensional torus, the wrap-around butterfly, the hypercube, etc. Another important class of graphs are the so-called expanders.

**Definition 2.7 (Expander)** *A graph is called* expander *if it has a constant edge expansion.*

Note that there exist expanders of constant degree. So far, the best expanders that have an explicit construction are all vertex-transitive (see, e.g., [2, 3, 4]).

An important subclass of indirect networks are the so-called leveled graphs.

**Definition 2.8 (Leveled Graph)** *A graph $G = (V, E)$ is called* leveled with depth $D$ *if the nodes of $G$ can be partitioned into $D + 1$* levels $L_0, \ldots, L_D$ *such that every edge in $E$ connects nodes of consecutive levels. Nodes in level 0 are called* inputs, *and nodes in level $D$ are called* outputs. *If, in addition, $|L_0| = |L_D|$ and $L_0$ is identified with $L_D$, then $G$ is called a* wrapped leveled graph with depth $D$.

Examples of leveled graphs are the fat tree and the butterfly, and an example of a wrapped leveled graph is the wrap-around butterfly. In a butterfly it is usually assumed that the nodes in $L_0$ represent the input nodes and the nodes in level $L_D$ represent the output nodes. In a fat tree the nodes in level $L_D$ are usually both input and output nodes.

## 2.3 The diameter

One important goal in choosing a topology for a network is that it has a small diameter. The following theorem presents a lower bound for this.

**Theorem 2.9** *Every graph of maximum degree $d > 2$ and size $n$ must have a diameter of at least $\lfloor (\log n)/(\log(d-1)) \rfloor - 1$.*

**Proof.** Suppose we have a graph $G = (V, E)$ of maximum degree $d$ and size $n$. Start from any node $v \in V$. In a first step at most $d$ other nodes can be reached. In two steps at most $d \cdot (d-1)$ additional nodes can be reached. Thus, in general, in at most $k$ steps at most

$$1 + \sum_{i=0}^{k-1} d \cdot (d-1)^i = 1 + d \cdot \frac{(d-1)^k - 1}{(d-1) - 1} \leq \frac{d \cdot (d-1)^k}{d-2}$$

nodes (including $v$) can be reached. This has to be at least $n$ to ensure that $v$ can reach all other nodes in $V$ within $k$ steps. Hence,

$$(d-1)^k \geq \frac{(d-2) \cdot n}{d} \quad \Leftrightarrow \quad k \geq \log_{d-1}((d-2) \cdot n/d) \ .$$

Since $\log_{d-1}((d-2)/d) > -2$ for all $d > 2$, this is true only if $k \geq \lfloor \log_{d-1} n \rfloor - 1$. $\sqcap$

Theorem 2.9 uses as a construction for the lower bound a complete $(d-1)$-ary tree with a root of degree $d$. However, it is easy to see that in this tree there are two nodes (see the leaves $v$ and $w$ in Figure 7) with a distance of approximately $2 \log_{d-1} n$, which is by a factor of 2 larger than the lower bound. Can networks with a better diameter be constructed? The next theorem gives an answer to this.

**Theorem 2.10** *For every even $d > 2$ there is an infinite family of graphs $G_n$ of maximum degree $d$ and size $n$ with a diameter of at most $(\log n)/(\log d - 1)$.*

**Proof.** The proof is part of the assignment. $\sqcap$

## 2.4 The edge expansion

We start with an upper bound on the edge expansion that must hold for all networks.

**Theorem 2.11** *For every graph $G = (V, E)$ with non-negative edge capacities, the edge expansion can be at most 1.*

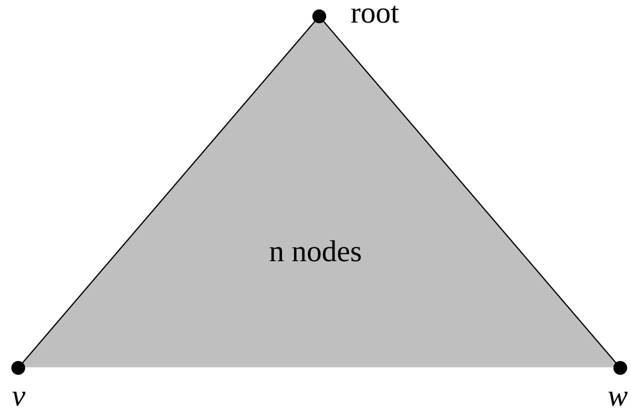


Figure 7: Nodes with highest distance in a tree.

**Proof.** For every set $U \subseteq V$ let $E_U = \{\{v,w\} \in E \mid v \in U\}$, where an edge appears twice in $E_U$ if both $v$ and $w$ are in $U$. Certainly, $(U,\bar{U}) \subseteq E_U$. Since $c(U) = c(E_U)$ it must therefore hold that $c(U,\bar{U}) \leq c(U)$. Equivalently, it must also hold that $c(U,\bar{U}) = c(\bar{U},U) \leq c(\bar{U})$. Hence, $c(U,\bar{U}) \leq \min\{c(U), c(\bar{U})\}$ and therefore

$$\alpha(G) = \min_{U \subseteq V} \frac{c(U,\bar{U})}{\min\{c(U), c(\bar{U})\}} \leq 1 \ .$$

$\sqcap$

Interestingly, for any $d \geq 3$ there are graphs that can achieve a constant edge expansion. These are the so-called expanders. Unfortunately, it is very complicated to construct and analyze graphs that have a constant edge expansion. Also, for the classes of graphs we presented above the edge expansion is somehow complicated to compute. Therefore, we just give some results here.

**Theorem 2.12** *The $d$-dimensional hypercube, cube-connected-cycles, butterfly, shuffle-exchange, and DeBruijn graph with uniform edge capacities all have an edge expansion of* $\Theta(1/d)$.

Using the fact that for these networks $d = \Theta(\log n)$, where $n$ is the number of nodes in the network, it follows that all of these networks have an edge expansion of $\Theta(1/\log n)$.

Next we give a relationship between the edge expansion and the diameter. Since the flow number of a network is an upper bound on its diameter, it follows from Theorem 1.13:

**Corollary 2.13** *For every graph with edge expansion $\alpha$ the diameter is at most* $O(\alpha^{-1} \log n)$.

From Theorem 2.9 it follows that this bound is exact for constant degree expanders. However, for hypercubic networks it is off by a logarithmic factor, as will come out in the next section.

## 2.5 The flow number

First we explore limitations on the flow number.

**Theorem 2.14** *For every network with diameter $D$ its flow number must be at least $D$. Also, for every network with edge expansion $\alpha$ its flow number must be at least $\alpha^{-1}$*

The first property is obvious and the second follows immediately from Theorem 1.13. Do there exist networks where the flow number is in $O(\max\{D, \alpha^{-1}\})$? The next theorem lists some.

**Theorem 2.15** *The $d$-dimensional hypercube, cube-connected-cycles, butterfly, shuffle-exchange, and DeBruijn graph with uniform edge capacities all have a flow number of $\Theta(d)$.*

For proofs see, for example, [1] or [5]. Thus, for these networks it actually holds that $F = \Theta(\alpha^{-1})$, i.e. the edge expansion describes very well the routing ability of the network. It also follows from the bound that all must have a diameter of $O(\log n)$.

## 2.6 The robustness

We end this section with the definition of one more parameter: the *robustness* of a graph $G = (V, E)$. The robustness $R$ measures how well a graph can sustain node failures. It is defined as the maximum ratio between the capacity lost in $G$ due to node failures and the still available capacity that guarantees that $G$ still contains a connected component $G' = (V', E')$ with $c(V') \geq c(V)/2$. The next theorem gives a lower bound for $R$.

**Theorem 2.16** *For every graph $G = (V, E)$ with edge expansion $\alpha$, $R \geq \frac{2\alpha}{1-\alpha}$.*

**Proof.** Suppose that so many node failures have occurred that every connected component $U$ of working nodes fulfills $c(U) \leq c(V)/2$. This would be bad in our case, so we want to determine what kind of failure rate would be needed to achieve this. Let $U_1, \ldots, U_k$ be the sequence of all connected components that are still left in $G$. We assume the $U_i$s to be largest possible in a sense that all outside neighbors of nodes in every $U_i$ have failed. Since $c(U_i) \leq c(V)/2$ for every $i$ it follows from the definition of the edge expansion that $c(U_i, \bar{U}_i) \geq \alpha \cdot c(U_i)$ for every $i$. Because in $c(U_i)$ all edges having both endpoints in $U_i$ are counted twice, the total working edge capacity in $U_i$ is

$$\tfrac{1}{2}(c(U_i) - c(U_i, \bar{U}_i)) \leq \tfrac{1}{2}(c(U_i) - \alpha \cdot c(U_i)) = \tfrac{1-\alpha}{2} \cdot c(U_i) \ .$$

Hence, if $V'$ is the set of all working nodes, then the total working edge capacity over all nodes is at most

$$\sum_i \tfrac{1-\alpha}{2} \cdot c(U_i) = \tfrac{1-\alpha}{2} \cdot c(V') \ . \tag{1}$$

On the other hand, for every set $U_i$ the set $(U_i, \bar{U}_i)$ consists of edges not working due to failed nodes. Since none of these edges can be contained in any other set $(U_j, \bar{U}_j)$, the total failed edge capacity is at least

$$\sum_i c(U_i, \bar{U}_i) \geq \sum_i \alpha \cdot c(U_i) = \alpha \cdot c(V') \ . \tag{2}$$

Hence, combining (1) with (2) the ratio between the failed and working edge capacities must be at least

$$\frac{\alpha \cdot c(V')}{\frac{1-\alpha}{2} \cdot c(V')} = \frac{2\alpha}{1-\alpha}$$

to make sure that there is no connected component $U$ any more with $c(U) > c(V)$. $\sqcap$

As an example, consider the line of $n$ nodes. It has an edge expansion of $\alpha = 1/(n-1)$. Hence, $R \geq (2/(n-1))/(1-1/(n-1)) = 2/(n-2)$, which is true, because it takes the deletion of at least 2 of the $n-1$ edges (or a ratio of $2/(n-1)$) to make sure that for all connected components $U$, $c(U) \leq c(V)/2$.

# References


[1] F. Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays · Trees · $Hypercubes$*. Morgan Kaufmann Publishers (San Mateo, CA), 1992.

[2] A. Lubotzky, R. Phillips, and R. Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988.

[3] G. Margulis. Explicit group theoretical constructions of combinatorial schemes and their application to the design of expanders and superconcentrators. *Problems Inform. Transmission*, 11:39–46, 1988.

[4] M. Morgenstern. Existence and explicit constructions of $q+1$ regular Ramanujan graphs for every prime power $q$. *Journal of Combinatorial Theory, Series B*, 62:44–62, 1994.

[5] C. Scheideler. *Universal Routing Strategies for Interconnection Networks*, volume 1390 of *Lecture Notes in Computer Science*. Springer, 1998.