(Slides for CS838: Topics in parallel computing, CS1221, Tue+Thu, Feb 2+4, 1999, 8:00-9:15 a.m., Pavel Tvrdik)

Section #5: **Direct interconnection networks I+II**

Direct interconnection network (IN)

=

connected graph
vertices = processing nodes
edges = communication links

processing node

=

one or more processors + local memory +communication router

$\boxed{\textbf{Basic notions and terminology}}$

$\boxed{\textbf{Alphabets and strings}}$

d***-ary alphabet*** : $\mathcal{Z}_d = \{0, 1, .\,.\,, d-1\}$

n***-letter*** d***-ary strings*** : $\mathcal{Z}_d^n = \{x_{n-1} \ldots x_0; x_i \in \mathcal{Z}_d\}$, $n \geq 1$

len(x) = the length of string x

ε = the ***empty*** string

x^i = the i-***fold concatenation*** of x

binary alphabet $\mathcal{B}$

$\overline{b_i} = 1 - b_1$ = the ***inversion*** of bit b_i

$\mathrm{neg}_i(b) = b_{n-1} .\,. b_{i+1}\overline{b_i}b_{i-1} .\,. b_0$

$\boxed{\textbf{Graph theory}}$

vertex and edge set of graph G : $V(G)$, $E(G)$, $N = |V(G)|$

adjacent vertices u and v = ***edge*** $\langle u, v\rangle$

H **=** ***subgraph*** of G, $H \subset G$: $V(H) \subset V(G)$ and $E(H) \subset E(G)$

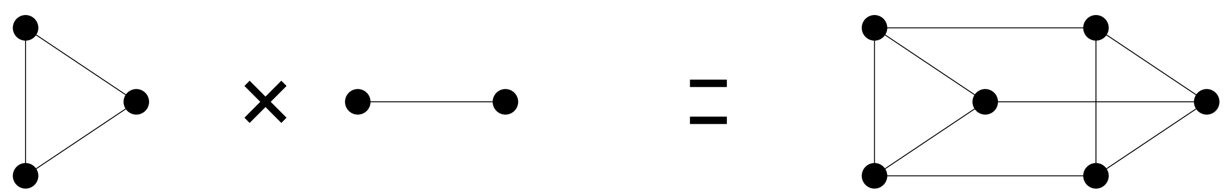
H **=** ***induced subgraph*** of G : ***maximal*** subgraph with $V(H)$

H **=** ***spanning subgraph*** of G : $V(H) = V(G)$

cartesian product $G = G_1 \times G_2$:

$$V(G) = \{(x,y); x \in V(G_1), y \in V(G_2)\}$$

$$E(G) = \{\langle (x_1,y),(x_2,y)\rangle; \langle x_1,x_2\rangle \in E(G_1)\} \cup \{\langle (x,y_1),(x,y_2)\rangle; \langle y_1,y_2\rangle \in E(G_2)\}$$



degree of vertex u, $\deg_G(u)$: $\quad \#$ of neighbors of u

degree set of graph G, $\deg(G)$: $\quad \{\deg_G(u); u \in V(G)\}$

maximum degree of G : $\quad \triangle(G) = \max(\deg(G))$

minimum degree of G : $\quad \delta(G) = \min(\deg(G))$

k***-regular graph*** G : $\quad \triangle(G) = \delta(G) = k$

path length $\mathrm{len}(P(u,v))$: $\quad \#$ of edges in $P(u,v)$

distance $\mathrm{dist}_G(u,v)$: $\quad$ the length of a shortest $P(u,v)$

average distance in N-vertex G : $\quad \Sigma_{u,v}\, \mathrm{dist}_G(u,v)/(N(N-1))$

diameter of G : $\quad \phi(G) = \max_{u,v} \mathrm{dist}_G(u,v)$

vertex-disjoint paths : $\quad V(P_1(u,v)) \cap V(P_2(u,v)) = \{u,v\}$

edge-disjoint paths : $\quad E(P_1(u,v)) \cap E(P_2(u,v)) = \emptyset$

(vertex) connectivity of G, $\kappa(G)$:

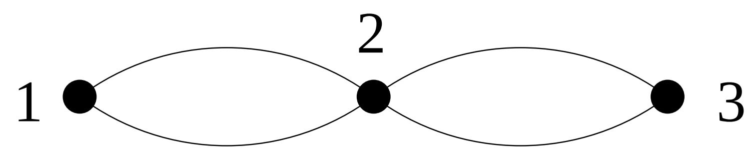
the ***minimum*** # of vertices whose removal disconnects G

edge connectivity of G, $\lambda(G)$:

the ***minimum*** # of edges whose removal disconnects G

$$\kappa(G) \leq \lambda(G) \leq \delta(G)$$

G



$\kappa(G) = 1$, $\lambda(G) = \delta(G) = 2$

k***-connected graph*** : $\kappa(G) = k$

k***-edge-connected graph*** : $\lambda(G) = k$

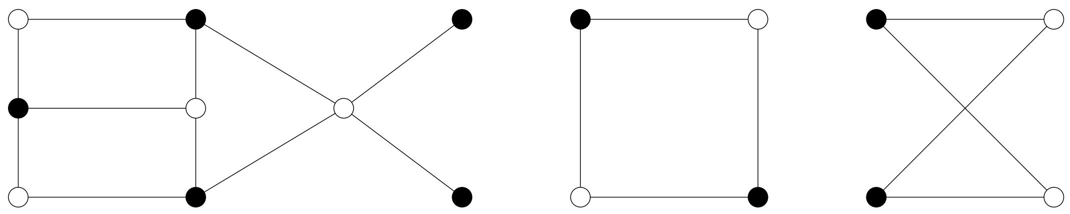
optimal connectivity : $\kappa(G) = \lambda(G) = \delta(G)$

Menger's theorem : *Between any 2 vertices of* G, $\exists$ *at least* $\kappa(G)$ *vertex-disjoint and at least* $\lambda(G)$ *edge-disjoint paths.*

fault distance between u and v : the ***maximum*** over lengths of all shortest vertex-disjoint paths between u and v.

fault diameter : maximum over fault distances.

bipartite and balanced bipartite graph :



hamiltonian path/cycle :

(closed) path over all vertices (vertex permutation)

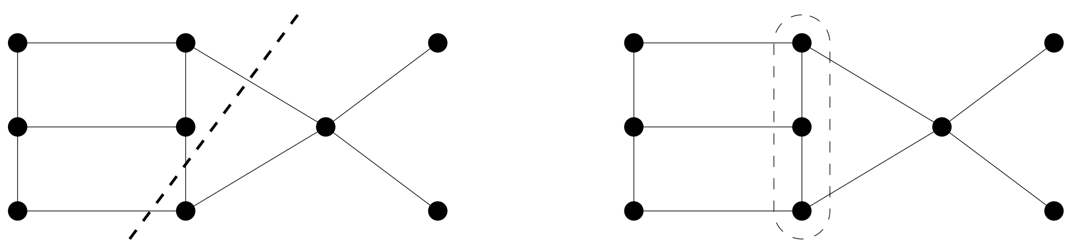
A bipartite graph can have a hamiltonian cycle only if balanced.

(edge) bisection width of G, $\mathrm{bw_e}(G)$:

the smallest # of edges removal of which splits G into two halves

vertex bisection width of G, $\mathrm{bw_v}(G)$:

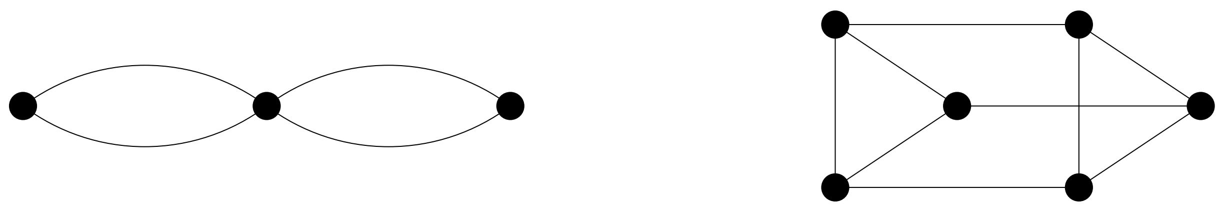
the smallest # of vertices removal of which splits G into two halves (of size at most $\lceil |V|/2 \rceil$ vertices)



isomorphism :

automorphism of G : any isomorphic mapping of G to itself

vertex-symmetric, edge-symmetric graphs :



digraphs = oriented graphs :

- ***arc set*** $A(G)$
- arc $\langle u \to v \rangle$ is ***incident from*** u ***to*** v
- $\deg_G^{\text{in}}(u)$ and $\deg_G^{\text{out}}(u)$: ***in-degree*** and ***out-degree***
- dipaths
- oriented diameters
- strong connectedness

$\boxed{\text{Requirements on interconnection networks}}$

small diameter and small average distance :

- decreases *communication latency* for
  - both *distance sensitive* routing (*store-and-forward*)
  - and *distance insensitive* routing (*wormhole routing*)

small and fixed vertex degree :

low-cost and universal routers $\times$ low connectivity and large distances

The diameter of an N-node fixed-degree network is $\Omega(\log N)$.

Proof: Given N-vertex graph with $\triangle(G) \leq k$, k is constant,

$\Rightarrow$ the # of vertices reachable in G within i steps is $O(k^i)$

$\Rightarrow N = O(k^{\phi(G)})$

$\Rightarrow \phi(G) = \Omega(\log N)$.

large bisection width :

- parallel *binary divide-and-conquer* algorithms need *high bandwidth* between both halves
  - split the input into two halves
  - solve them recursively on both halves of the IN in parallel
  - merge the results from both halves into the final result
- VLSI design: large bisection width ⇒ *many interchip wires*

high connectivity and fault tolerance :

- *redundant* short paths in case of link or node *faults*
- *bypassing congested parts or hotspots*
- *spliting large packets* into smaller chunks sent along *parallel disjoint* paths

small fault average distance and diameter

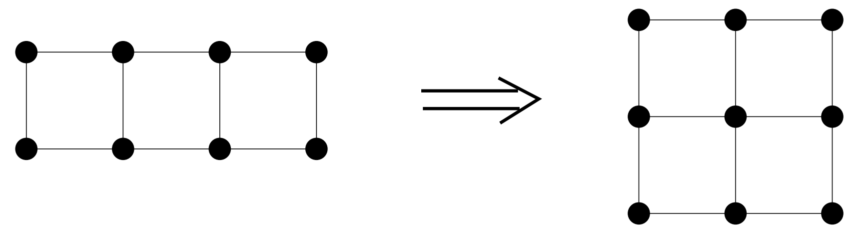
hamiltoniaty : labeling processors with numbers $1, \ldots, p$ (sorting)

hierarchical recursivity :

- ***topology*** =
  set of equally defined graphs of different ***dimensions***
- ***hierarchically recursive topology*** :
  lower-dim. instances are *subgraphs* of higher-dim. instances
- makes the design and manufacturing of INs easier
- allows *inductive design* and mapping of parallel algorithms

incremental extendability and incremental scalability :

- ***extendable*** topology: defined for specific values of N
- ***incrementally extendable*** topology: defined for any N
- ***incrementally scalable*** topology:
  N-vertex instance $\Longrightarrow$ $(N+k)$-vertex instance needs $O(k)$ work



symmetry : makes parallel algorithms design easier

- doesn't matter where the computation starts
- doesn't matter in which directions communication evolves
- VLSI design

support for routing and collective communication :

- shortest-path one-to-one routing
- permutation routing
- one-to-all communication operations
- all-to-all communication operations

embeddability of and into other topologies :

- efficient mapping of the process graph into the topology
- ability to simulate efficiently other topologies

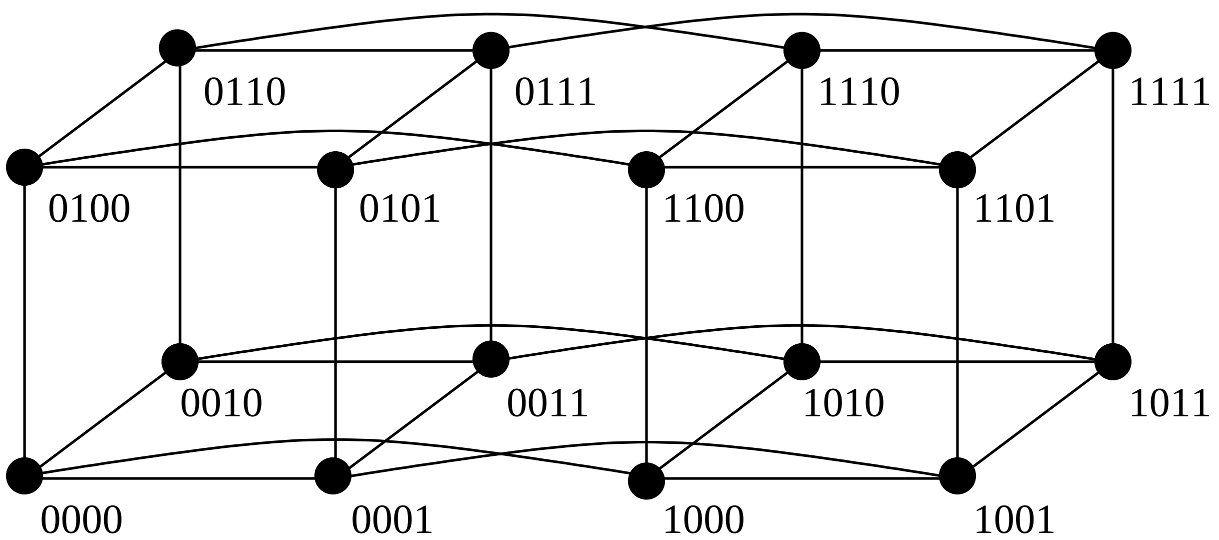
simple VLSI or 3-D layout

Mesh-based (strictly orthogonal) topologies

hypercubes, meshes, and tori

Binary hypercube of dimension n, Q_n

Q_4:



$V(Q_n) = \mathcal{B}^n$

$E(Q_n) = \{\langle x, \mathrm{neg}_i(x)\rangle; x \in V(Q_n), 0 \leq i < n\}$

$\phi(Q_n) = n$

$\mathrm{bw_e}(Q_n) = 2^{n-1}$

$|V(Q_n)| = 2^n$

$|E(Q_n)| = n2^{n-1}$

$\deg(Q_n) = \{n\}$

$\boxed{\textbf{Basic properties of the hypercube}}$

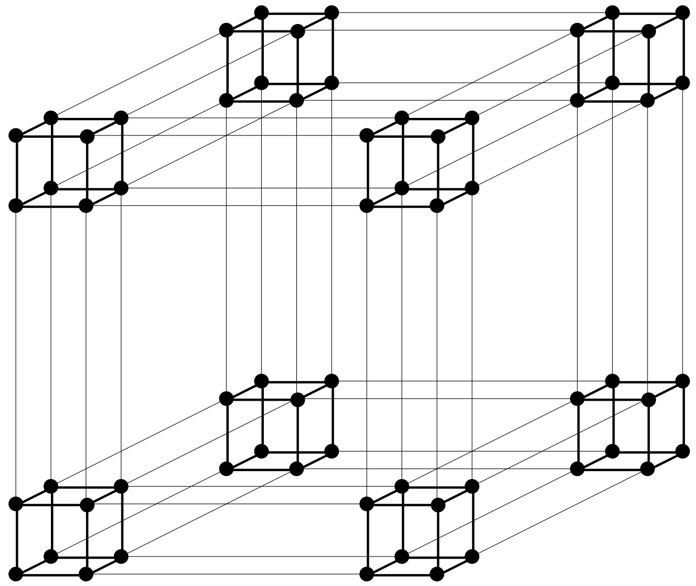
- regular, balanced bipartite, hamiltonian
- *Hamming distance* ϱ
- the average distance in Q_n is $n/2$
- logarithmic degree $\Rightarrow$ ***hypercubic networks***
- the *largest* possible bisection width
- $\lambda(Q_n) = \kappa(Q_n) = n$
- if $\varrho(u, v) = k$, then $\exists\ n$ vertex-disjoint paths $P(u, v)$ among which
  - k are of length k
  - $n - k$ are of length $k + 2$
- the fault diameter of Q_n is n
- $\exists\ k!$ different shortest paths between two vertices in distance k
- hierarchical recursivity

$$Q_n = Q_p \times Q_{n-p} = Q_p \times Q_q \times Q_{n-p-q} = Q_1^n$$

- ***canonical decomposition*** : $Q_n = Q_{n-1}[j = 0] \times Q_{n-1}[j = 1]$

- ***subcubes*** : $s_{n-1} \ldots s_1 s_0$, where $s_i \in \{0, 1, *\}$, $* =$ don't care (subcubes = terms in boolean algebra)
- only extendable $\Rightarrow$ ***incomplete hypercubes***
- vertex- and edge-symmetric: $2^n \times n!$ automorphisms
  - ***translation*** $u \to v$: mapping $\forall x \in V(Q_n)$ $(x \mapsto (u \,\mathrm{XOR}\, v))$
  - ***rotation***: a permutation (renaming) of dimensions
- *e*-***cube*** routing
- *optimal* algorithms for collective communication in almost all communication models
- *simulates efficiently almost any other topology*

$Q_6 = Q_3 \times Q_3$:



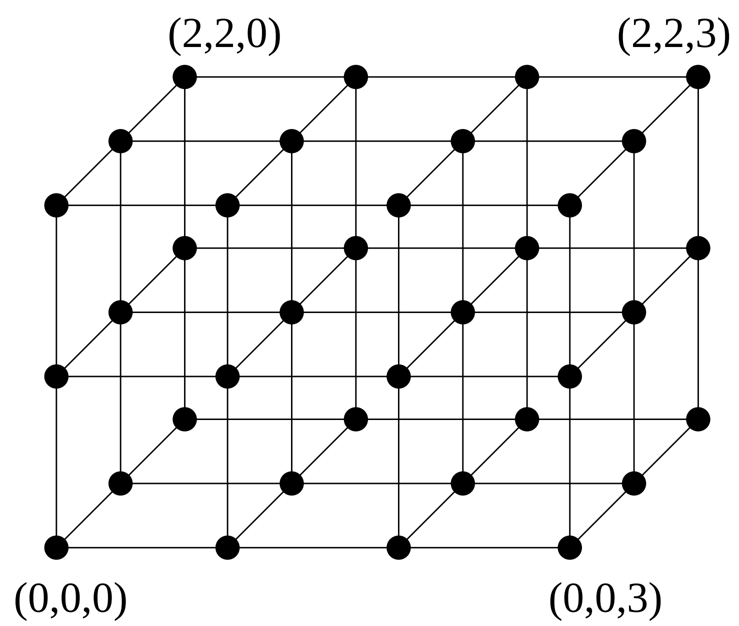
Remarks to the hypercube

- ***subcube allocation problem*** in multiuser hypercube machines:
  - users declare numbers of required nodes for their tasks
  - OS allocates, if possible, subcubes of appropriate sizes
  - OS releases subcubes upon task completion
  - ***fragmentation*** and ***compaction***
- testbed for parallel feasibility of problems on distributed memory architectures
- main drawbacks = logarithmic degree and lack of scalability
- commercial hypercube MPPs: nCUBE 1 and 2, Intel's iPSC/2 and iPSC/860, TMC's CM-2, SGI Origin

n-dimensional mesh of dimensions $z_1, z_2, .., z_n$, $M(z_1, z_2, .., z_n)$

$$V(M(\ldots)) = \{(a_1, a_2, .., a_n); 0 \le a_i \le z_i - 1 \,\forall i \in \{1, .., n\}\}$$
$$E(M(\ldots)) = \{\langle(.., a_i, ..), (.., a_i + 1, ..)\rangle; 0 \le a_i \le z_i - 2\}$$
$$|V(M(\ldots))| = \Pi_{i=1}^{n} z_i \qquad |E(M(\ldots))| = \Sigma_{i=1}^{n}(z_i - 1)\Pi_{\substack{j=1\\j\neq i}}^{n} z_j$$
$$\phi(M(\ldots)) = \Sigma_{i=1}^{n}(z_i - 1) = \Omega(\sqrt[n]{|V(M(\ldots))|})$$
$$\deg(M(\ldots)) = \{n, .., n + j\},\ j = |\{z_i; z_i > 2\}|$$
$$\mathrm{bw_e}(M(\ldots)) = \begin{cases} (\Pi_{i=1}^{n} z_i)/\max_i z_i & \text{if } \max_i z_i \text{ is even,} \\ \Omega((\Pi_{i=1}^{n} z_i)/\max_i z_i) & \text{otherwise.} \end{cases}$$

Mesh $M(3,3,4)$:



$\boxed{\textbf{Basic properties of the mesh}}$

- we assume that the mesh dimension n is *fixed*
- the most practical meshes are *2-D* and *3-D*
- 1-D meshes = ***linear arrays*** (incrementally scalable)
- $M(k, k, .., k) = k$-***ary*** n-***cube***
- bipartite (not necessarily balanced)
- hamiltonian if at least one side has *even* length
- a hamiltonian path always exists
- not regular $\Rightarrow$ not vertex symmetric
- large diameter for small n
- the *exact* value of bisection width is open problem for $n \geq 4$
- easy for 2- or 3-D meshes (very low = perimeter problem)

$$\mathrm{bw_e}(M(11, 9, 6)) = 64$$

- optimal connectivity
- optimal fault diameter
- fault distances = by at most 4 larger than nonfaulty ones

- hierarchical recursiveness

$$M(z_1, z_2, .., z_k) = M(z_1, z_2, .., z_{k-1}) \times M(z_k) = M(z_1, .., z_i) \times M(z_{i+1}, .., z_j) \times M(z_{j+1}, .., z_k) = M(z_1) \times M(z_2) \times \cdots \times M(z_n).$$

- ***submeshes*** = strings with don't-care symbols
- incremental extendability, but not scalability
- the basic shortest-path routing = ***dimension ordered*** routing
  2- and 3-D meshes : ***XY*** and ***XYZ routing***
- $\exists$ many communication and parallel algorithms *matching* the mesh
  lower bounds (large diameter, low bisection width)
- modifications: ***reconfigurable meshes, meshes of buses***

Remarks on meshes

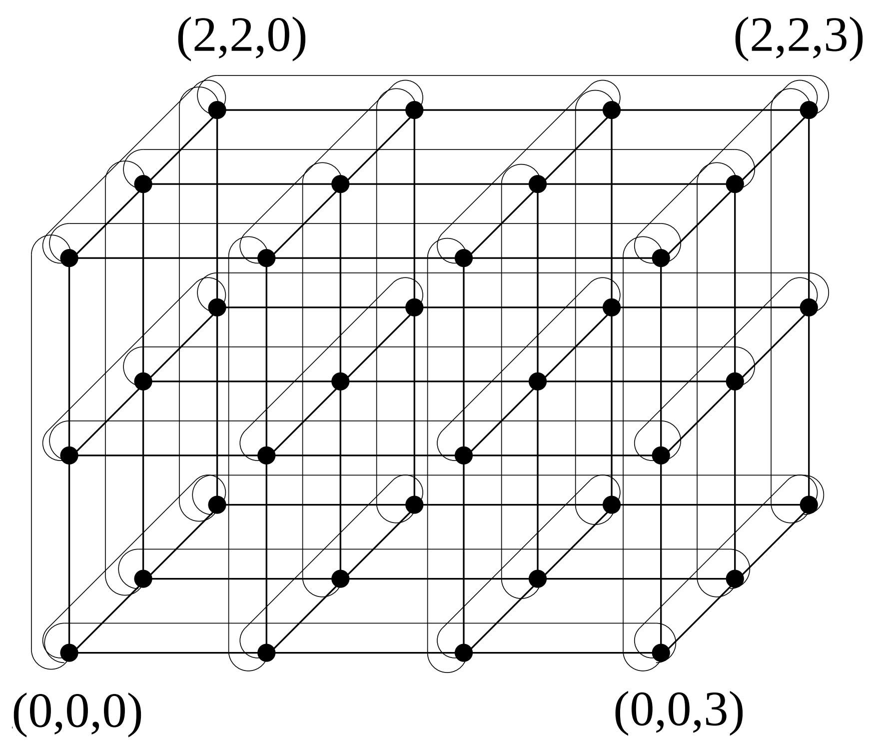
- commercial mesh-based parallel computers: Intel's Paragon (2-D mesh), MIT J-Machine (3-D mesh), transputers (2-D mesh)
- *submesh* allocation, fragmentation and compaction problem

$\boxed{n\textbf{-dimensional torus of dimensions } z_1, z_2, \ldots, z_n\textbf{, } T(z_1, z_2, \ldots, z_n)}$

torus = ***toroidal*** or ***wrapped mesh***

$$V(T(\ldots)) = V(M(\ldots))$$
$$E(T(\ldots)) = \{\langle(\ldots, a_i, \ldots), (\ldots, a_i \oplus_{z_i} 1, \ldots)\rangle; 0 \le a_i < z_i\}$$
$$|E(T(\ldots))| = n \times \Pi_{i=1}^{n} z_i$$
$$\phi(T(\ldots)) = \Sigma_{i=1}^{n} \lfloor z_i/2 \rfloor$$
$$\deg(T(\ldots)) = \{2n\}$$
$$\mathrm{bw_e}(T(\ldots)) = 2\,\mathrm{bw_e}(M(\ldots))$$

Torus $T(3,3,4)$:



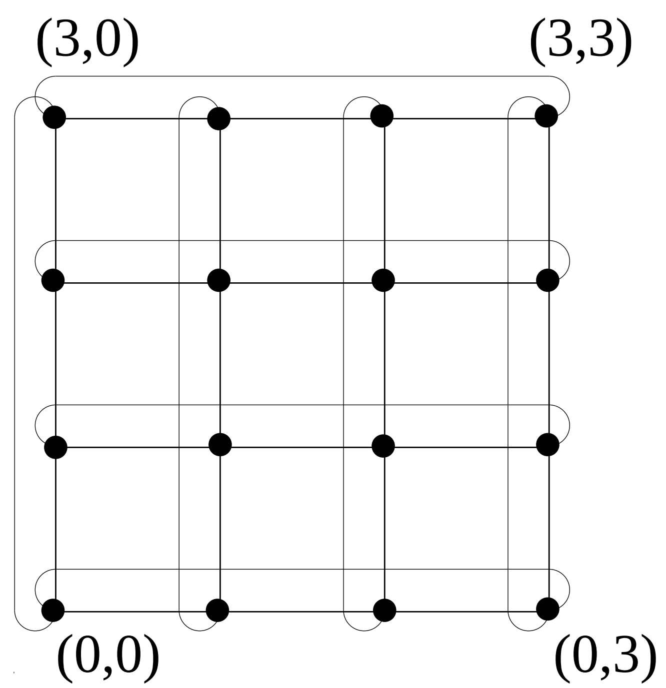
Basic properties of the torus

- a 1-dimensional torus = ***cycle*** or ***ring***
- $T(k, k, \ldots, k) = k$-***ary*** n-***torus***
- bipartite iff all side lengths are even (bipartite $\Rightarrow$ balanced)
- hamiltonian
- regular and vertex symmetric (automorphisms = translations)
- k-ary n-tori are edge symmetric (automorphisms = rotations)
- diameter/average distance drop to 1/2 w.r.t. equal mesh
- connectivity/bisection width double w.r.t. equal mesh
- hierarchically recursive, up to cartesian product of cycles
- scalability properties are even slightly worse comapred to meshes
- *dimension ordered* shortest-path routing (cycles make it more complicated)
- topologically optimal algorithms exist for many fundamental problems on tori
- commercial MPPs: Cray/SGI T3D and T3E, Convex Exemplar (3-D), Intel/CMU iWarp (2-D), KSR (1-D)

Comparisons of hypercubes, meshes, and tori

- $M(2,2,..,2)$ is isomorphic Q_n
- n-dimensional meshes and tori are generalizations of Q_n
- For some k and n, $k < n$, a k-dimensional mesh/torus can be a subgraph of Q_n.

$T(4,4) = Q_4$:



Tradeoff between diameter, number of edges, and bisection width

$M(8,8,4) \subset T(8,8,4) \subset Q_8$ and for all of them, $|V()| = 256$..

	$M(8,8,4)$	$T(8,8,4)$	Q_8
$\phi()$	17	10	8
$\|E()\|$	640	768	1024
$\mathrm{bw_e}()$	32	64	128

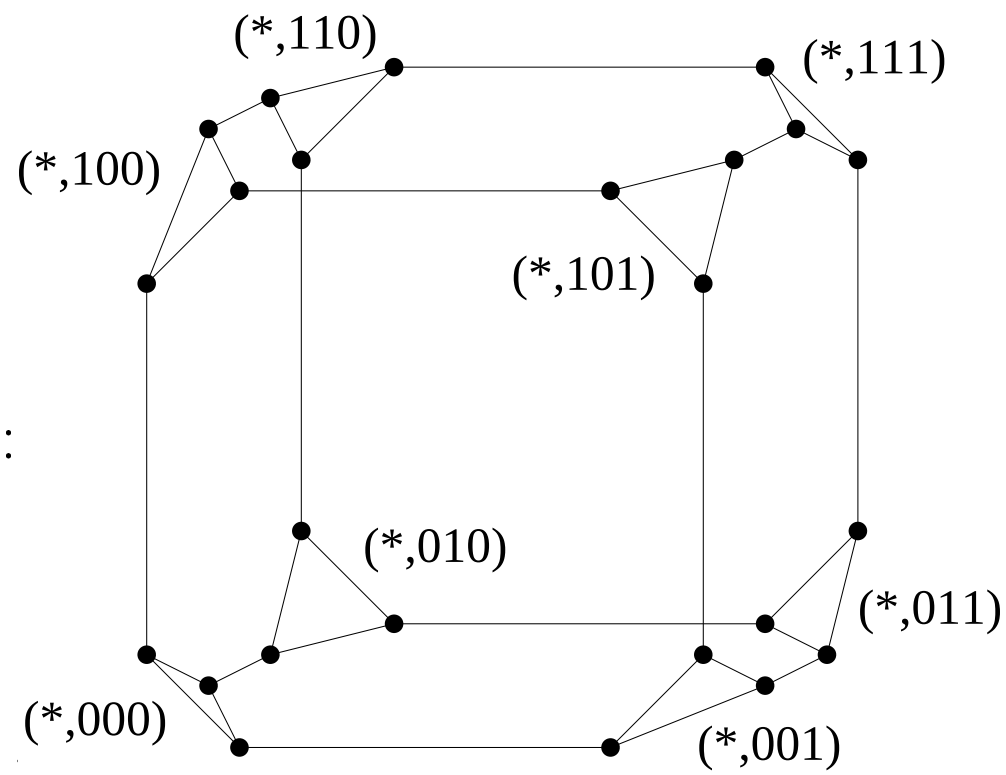
$\boxed{\textbf{Hypercubic topologies}}$

- logarithmic-diameter constant-degree derivatives of the hypercube
- worse extendability: the # of vertices is $n2^n$ or similar
- bisection width $\Omega(N/\log N)$
- two main representants: *cube-connected cycles* and *butterflies*

$\boxed{\textbf{Cube-connected cycles of dimension } n\textbf{, } CCC_n}$

$V(CCC_n) = \{(i, x); 0 \leq i < n \wedge x \in \mathcal{B}^n\}$
$E(CCC_n) = \{\langle(i, x), (i \oplus_n 1, x)\rangle, \langle(i, x), (i, \mathrm{neg}_i(x))\rangle; (i, x) \in V(CCC_n)\}$
$|V(CCC_n)| = n2^n$
$|E(CCC_n)| = n2^{n-1} + n2^n$
$\phi(CCC_n) = (2n - 2) + \lfloor n/2 \rfloor$ for $n > 3$, $\phi(CCC_3) = 6$
$\deg(CCC_n) = \{3\}$
$\mathrm{bw_e}(CCC_n) = 2^{n-1}$

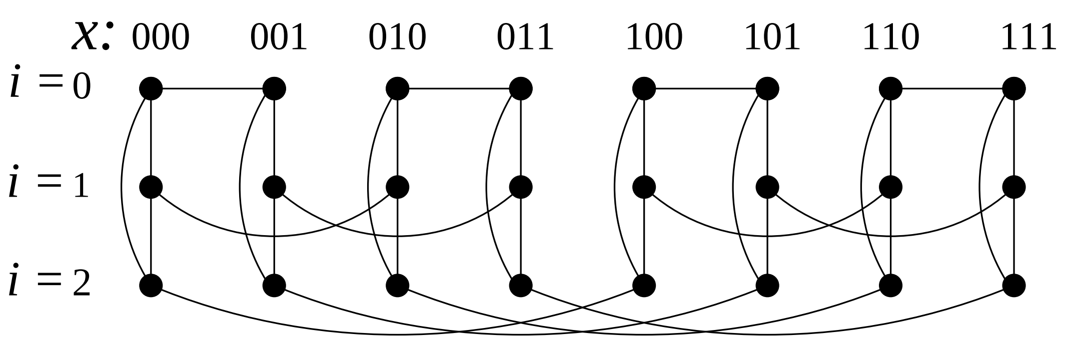
Cube Connected Cycles CCC_3:



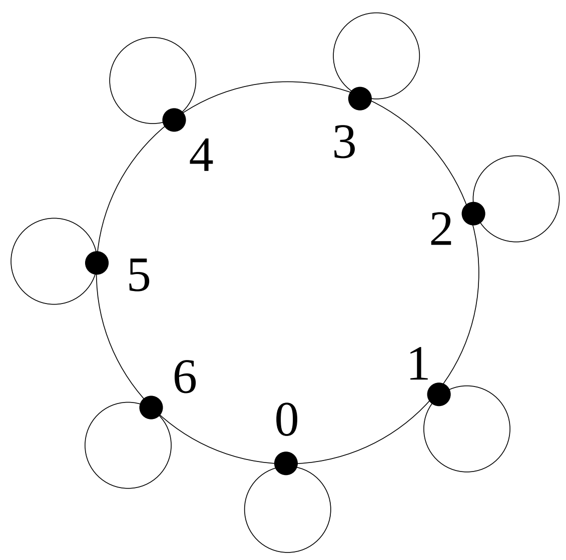
$\boxed{\textbf{Basic properties of } CCC}$

- CCC_n is balanced bipartite iff n is even
- regular, vertex-symmetric, hamiltonian
- not edge-symmetric (***hypercube*** and ***cycle*** edges)
- optimal connectivity 3 (between two vertices, $\exists$ at most two shortest paths)
- shortest-path routing : based on e-cube routing
- not hierarchically recursive

$\boxed{\textbf{Another view of } CCC}$



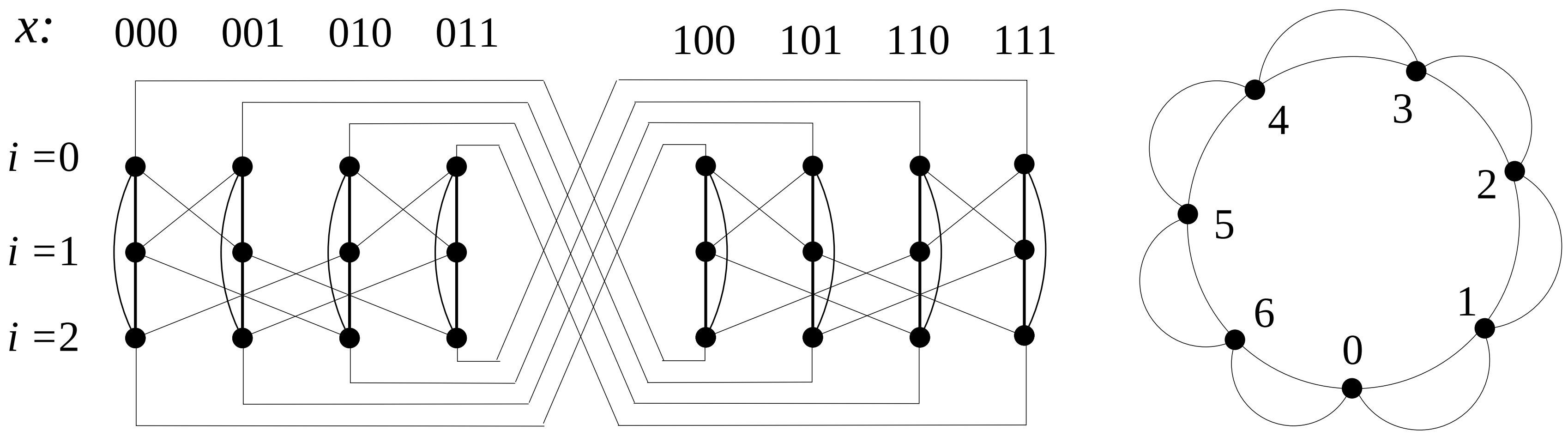
(a) CCC_3



(b) CCC_7

Wrapped butterfly of dimension n, wBF_n

$$V(wBF_n) = \{(i,x); 0 \le i < n \wedge x \in \mathcal{B}^n\}$$
$$E(wBF_n) = \{\langle(i,x),(i \oplus_n 1, x)\rangle, \langle(i,x),(i \oplus_n 1, \mathrm{neg}_i(x))\rangle \mid (i,x) \in V(wBF_n)\}$$
$$|V(wBF_n)| = n2^n$$
$$|E(wBF_n)| = n2^{n+1}$$
$$\phi(wBF_n) = n + \left\lfloor \frac{n}{2} \right\rfloor$$
$$\deg(wBF_n) = \{4\}$$
$$\mathrm{bw_e}(wBF_n) = 2^n$$



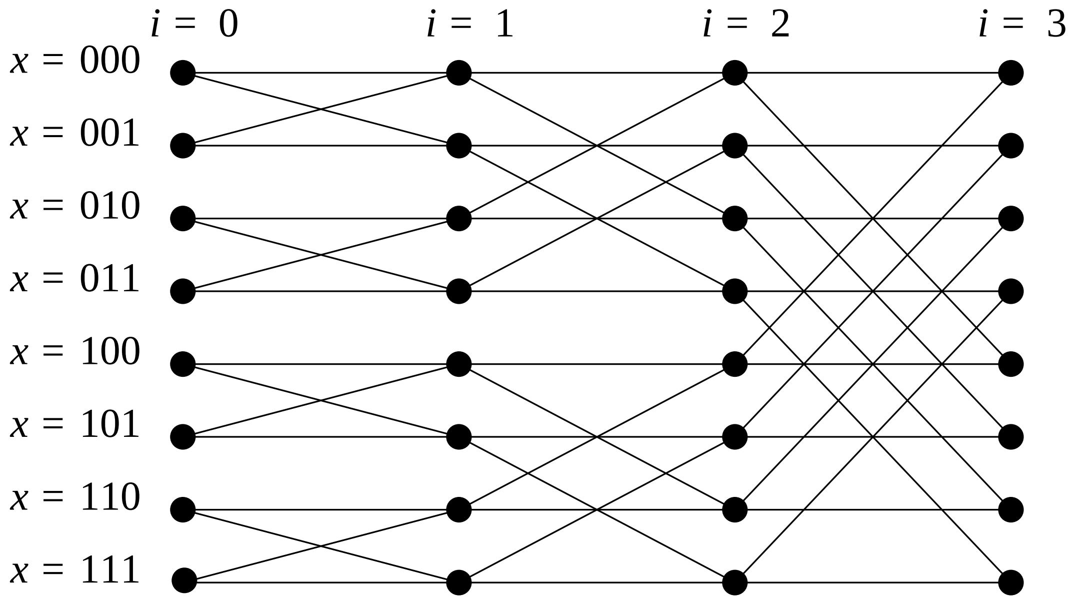
(a) wBF_3 (b) wBF_7

Basically the same properties as CCC, except that: more edges, larger bisection width, and lower diameter.

$\boxed{\textbf{Ordinary butterfly of dimension } n\textbf{, } oBF_n}$

$$V(oBF_n) = \{(i,x); 0 \le i \le n \wedge x \in \mathcal{B}^n\}$$
$$E(oBF_n) = \{\langle (i,x),(i+1,x)\rangle, \langle (i,x),(i+1,\mathrm{neg}_i(x))\rangle \mid i < n\}$$
$$|V(oBF_n)| = (n+1)2^n$$
$$|E(oBF_n)| = n2^{n+1}$$
$$\phi(oBF_n) = 2n$$
$$\deg(oBF_n) = \{2,4\}$$
$$\mathrm{bw_e}(oBF_n) = 2^n$$

oBF_3:



Basic properties of the ordinary butterfly

- oBF_n is organized into ***columns*** (***stages***) $0 \leq i \leq n$ and ***rows*** $0 \leq x \leq 2^n - 1$
- two kinds of edges: ***straight*** and ***cross*** (***hypercube***) edges
- not vertex symmetric and not regular
- not hamiltonian
- hierarchically recursive: oBF_n contains two oBF_{n-1} as subgraphs
- optimal connectivity 2
- $\exists$ only one shortest path between $(0, x)$ and (n, y) (= e-cube routing)

Tree-based topologies

- ***fat tree***
- ***pyramidal machines***
- ***mesh of trees***

$\boxed{\textbf{2-D mesh of trees of height } n\textbf{, } MT_n}$

For integer $n \geq 1$, let $N = 2^n$.

$V(MT_n) = (\mathcal{B}^n \times \cup_{i=0}^{n} \mathcal{B}^i) \cup (\cup_{i=0}^{n} \mathcal{B}^i \times \mathcal{B}^n)$

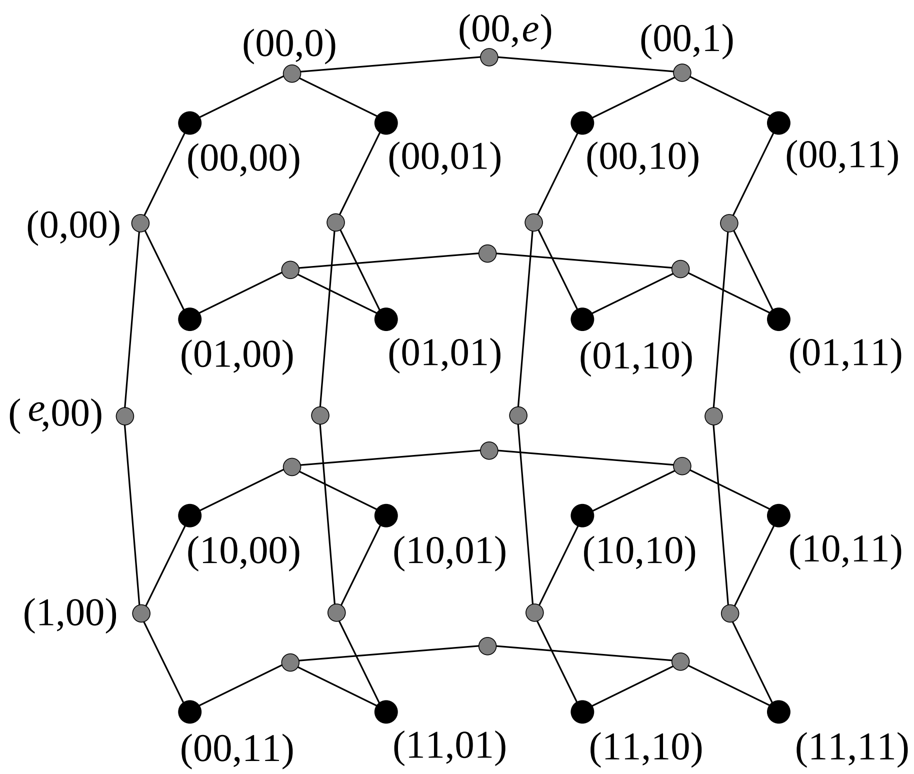
$E(MT_n) = \{\langle (x,y),(xa,y)\rangle; \mathrm{len}(x){<}n \wedge a \in \mathcal{B}\}$
$\qquad \cup \{\langle (x,y),(x,ya)\rangle; \mathrm{len}(y) < n \wedge a \in \mathcal{B}\}$

$|V(MT_n)| = 3N^2 - 2N$ $\qquad$ $|E(MT_n)| = 4N(N-1)$

$\deg(MT_n) = \{2,3\}$ $\qquad$ $\phi(MT_n) = 4\log N = 4n$

$\mathrm{bw_e}(MT_n) = N = \Theta(\sqrt{|V(MT_n)|})$

Mesh-of-trees MT_2:



MT can be defined for any 2-D mesh $M(2^k, 2^l)$ and for higher dimensions. We will discuss only the 2-D square mesh case.

Basic properties of the mesh-of-trees topology

- MT_n is a hybrid topology: it is a 2-D mesh $M(N, N)$ whose linear rows and columns are replaced with ***row*** and ***column*** complete binary ***trees*** of height n, CBT_n
- $M(N, N)$ has *large diameter* $O(N)$. CBT_n has extremely small *connectivity* and *bisection width* equal to 1. MT_n has $(\log N)$-diameter of CBT_n and $\sqrt{N}$-bisection width of $M(N, N)$.
- mesh vertices = *leaves* in intersections of row and column trees
- the additional vertices = the internal vertices of the trees
- $MT_n \subset CBT_n \times CBT_n$
- bipartite, but *not* balanced, therefore it is *not* hamiltonian
- hierarchically recursive (it forms a cluster of disjoint submeshes-of-trees connected via row- or column-tree roots)
- neither regular neither vertex symmetric
- poor extendability

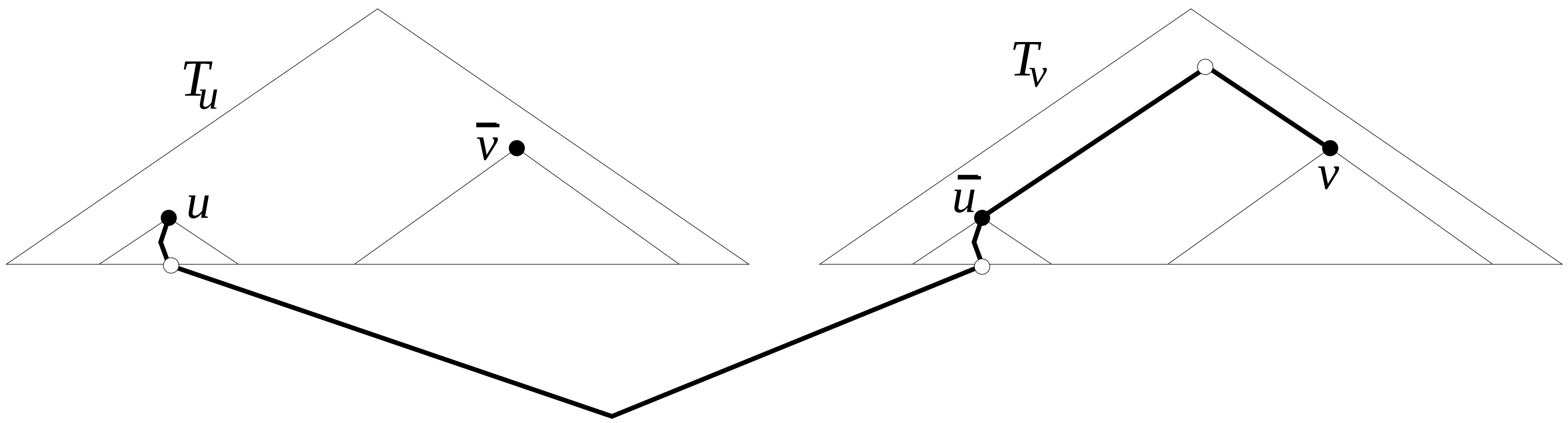
- simple shortest-path routing, it alternates routing in 2-D meshes and trees. Assume source vertex u and destination vertex v.

  1. If u is in a *row* tree and v in a *column* tree, the shortest path from u to v is uniquely defined via the intersection leaf of both trees.
  2. If u and v are in two *different row* trees T_u and T_v,
     (a) compare the level of u and v in their trees.
     (b) if u is lower than v, then (see the figure bellow)
        i. go down in T_u to any leaf in the subtree
        ii. use the corresponding column tree to get to T_v,
        iii. use tree routing in T_v to get to v by passing through the root of the smallest common subtree of v and of the homothetic image $\overline{u}$ of u in T_v
     (c) If u is higher than v, then
        i. go first to the image of v in T_u
        ii. via any leaf get to T_v then (see the figure with vertices u and v exchanged).

  The number of different shortest paths depends on the heights of the vertices.

3. The routing between two internal nodes of column trees is similar.

- MT is optimal for parallel solving of many problems similarly to the hypercube.
- No commercial machine has been built upon this topology yet.



$\boxed{\textbf{Shuffle-based topologies}}$

- *orthogonal* and *hypercubic* topologies: the adjacency is based on *incrementing* and/or *inverting* coordinates.
- *shuffle* topologies: the adjacency is based on ***shifting*** the whole strings. *Rich* and *dence*, but also rather *complicated* topologies.

$\boxed{\textbf{Common properties of shuffle-based topologies}}$

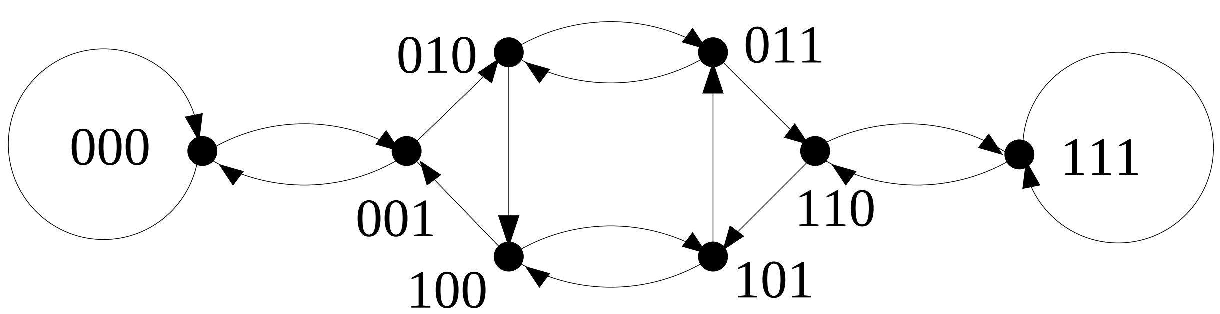
- *logarithmic* diameter and *constant* vertex degree
- *optimal* or nearly optimal connectivity
- fault diameter *nearly* the same as non faulty one, similarly for fault average distances
- *large* bisection width of order $N/\log N$
- *simple* shortest-path routing
- they are *neither* symmetric *nor* hierarchically recursive
- they are rich of *cycles* of various lengths, since the rotation induces equivalence classes of vertices called ***necklaces***
- at the same time, they are rich of *subtrees* and spanning trees, since the general shift operation induces subtrees

Binary shuffle-exchange

- ***shuffle*** of binary string x, $\sigma(x)$ = ***left rotation by one bit***.
- ***exchange*** operation on x = ***inversion*** of $LSB(x)$

$$V(SE_n) = \mathcal{B}^n$$
$$A(SE_n) = \{\langle x \to \mathrm{neg}_0(x)\rangle; x \in \mathcal{B}^n\} \cup \{\langle x \to \sigma(x)\rangle; x \in \mathcal{B}^n\}$$
$$|V(SE_n)| = 2^n$$
$$|A(SE_n)| = 2^{n+1}$$
$$\deg^{\mathrm{in}}(SE_n) = \deg^{\mathrm{out}}(SE_n) = \{2\}$$
$$\phi(SE_n) = 2n - 1$$
$$\mathrm{bw_e}(SE_n) = \Theta(\tfrac{2^n}{n})$$

Digraph SE_3:



Basic properties of binary shuffle-exchange topology

- $\langle x \to \sigma(x) \rangle$ = ***shuffle arc***, $\langle x \to \mathrm{neg}_0(x) \rangle$ = ***exchange arc***.
- $d_{SE}(0^n, 1^n) = 2n - 1 = \phi(SE_n)$
- $d_{SE}(u, v)$: (1) find the longest substring α s.t. $u = u'\alpha$ and $v = \alpha v'$, (2) convert prefix u' into postfix v' by shuffling and/or exchanging
- 2-regular digraph
- Necklace with vertex u has length i = *periodicity* of u.
  For example, if $|\mathrm{necklace}(111010)| = 6$ and $|\mathrm{necklace}(101101)| = 3$.
- due to necklaces, SE_n does not contain as subgraphs any SE_i, $i < n$.
- Also *not* vertex symmetric
- The exact value of $\mathrm{bw_e}(SE_n)$ is known up to a small constant multiplicative factor.
- Undirected SE is similar: Strings can be rotated to the right by ***unshuffle operation*** $\sigma^{-1}(x)$. The diameter, bisection width, and necklaces remain the same. Average distance drops. For example, $u = 1011101010$ and $v = 1000010111$, the unoriented distance is only 8 instead of 13.

De Bruijn digraph of degree d and dimension n, $dB_{d,n}$

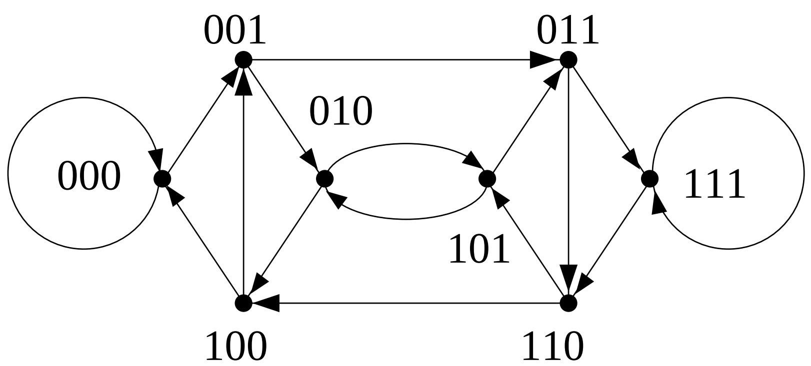
- adjacency operation = general ***left shift*** by 1 position:
  $\lambda(x,a) = x_{n-2} \ldots x_0 a$

$$V(dB_{d,n}) = \mathcal{Z}_d^n \qquad |V(dB_{d,n})| = d^n$$

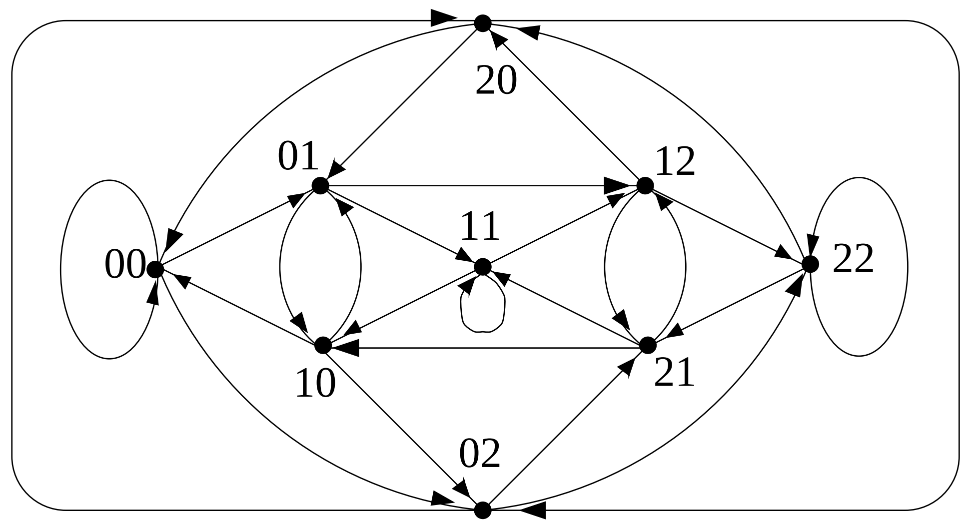
$$E(dB_{d,n}) = \{\langle x \rightarrow \lambda(x,a)\rangle; x \in \mathcal{Z}_d^n, a \in \mathcal{Z}_d\} \qquad |E(dB_{d,n})| = d^{n+1}$$

$$\deg^{\mathrm{in}}(dB_{d,n}) = \deg^{\mathrm{out}}(dB_{d,n}) = \{d\} \qquad \phi(dB_{d,n}) = n$$

$$\mathrm{bw_e}(dB_{d,n}) = \Theta(d^n/n)$$



(a) $dB_{2,3}$



(b) $dB_{3,2}$

Basic properties of de Bruijn topology

- vertex u is adjacent to v if the *first* $n-1$ letters of v equal to the *last* $n-1$ letters of u
- d-regular
- $dB_{2,n}$ is the *state diagram* of a shift register
- *extremely low* diameter: Any vertex can be obtained from another one by at most n left shifts, hence the diameter. $dB_{2,n}$ has the same diameter as Q_n with 4 arcs per vertex instead of $\log N$
- BUT: the ***average distance*** is very *close* to the diameter (in contrast to mesh-based topologies)
- the structure of necklaces depends on periodicity of strings, similarly to shuffle-exchange topology
- no recursivity. BUT: $dB_{d,n}$ is decomposable into *isomorphic universal building blocks* (project Galileo at JPL, $dB_{2,16}$ machine)
- no hierarchical recursivity. BUT: $dB_{d,n}$ is the ***line digraph*** of $dB_{d,n-1}$, i.e., if we replace all arcs in $dB_{d,n-1}$ with vertices and preserve adjacency, we get $dB_{d,n}$. It can be used for optimal *simulations* of large de Bruijn networks on small ones

- the connectivity is ***nearly*** optimal: $\kappa(dB_{d,n}) = \lambda(dB_{d,n}) = d - 1$ (loops at vertices a^n)
- the fault diameter of $dB_{d,n}$ is only $n + 1$
- the exact value of $\mathrm{bw}_{\mathrm{e}}(dB_{d,n})$ is known up to a small constant multiplicative factor
- no incremental extendability or scalability. BUT: there exist incrementally extendable ***generalized*** de Bruijn networks and incrementally scalable ***partial line digraphs***