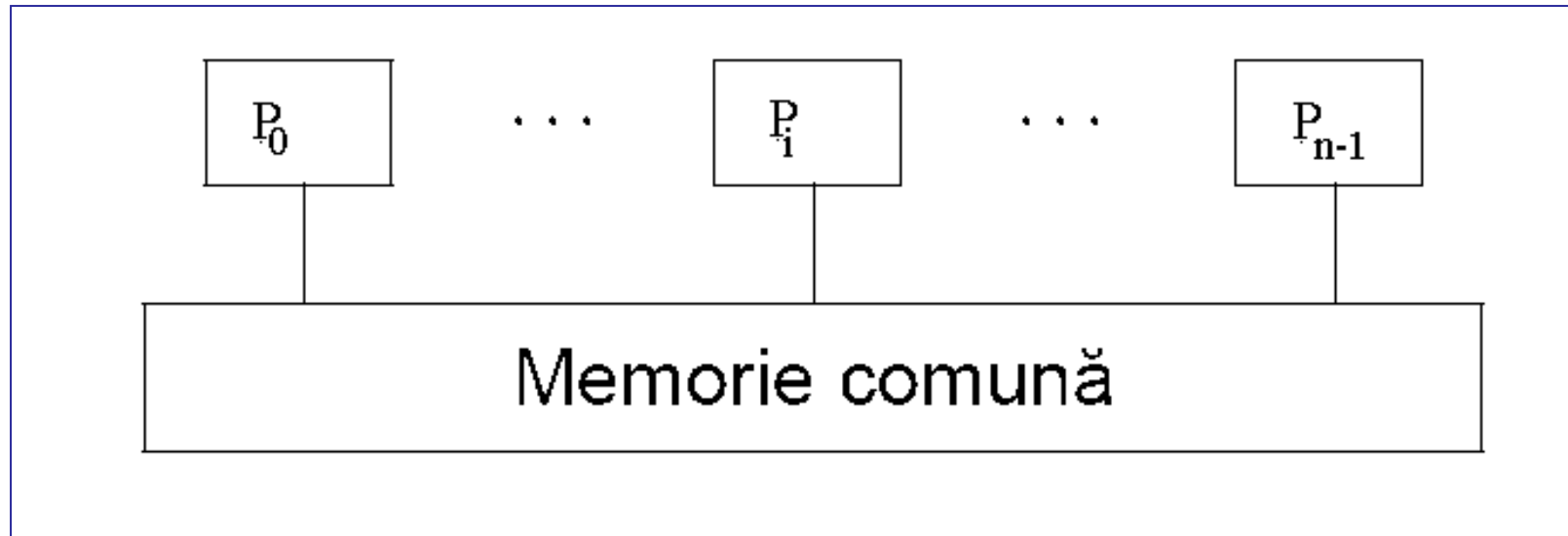
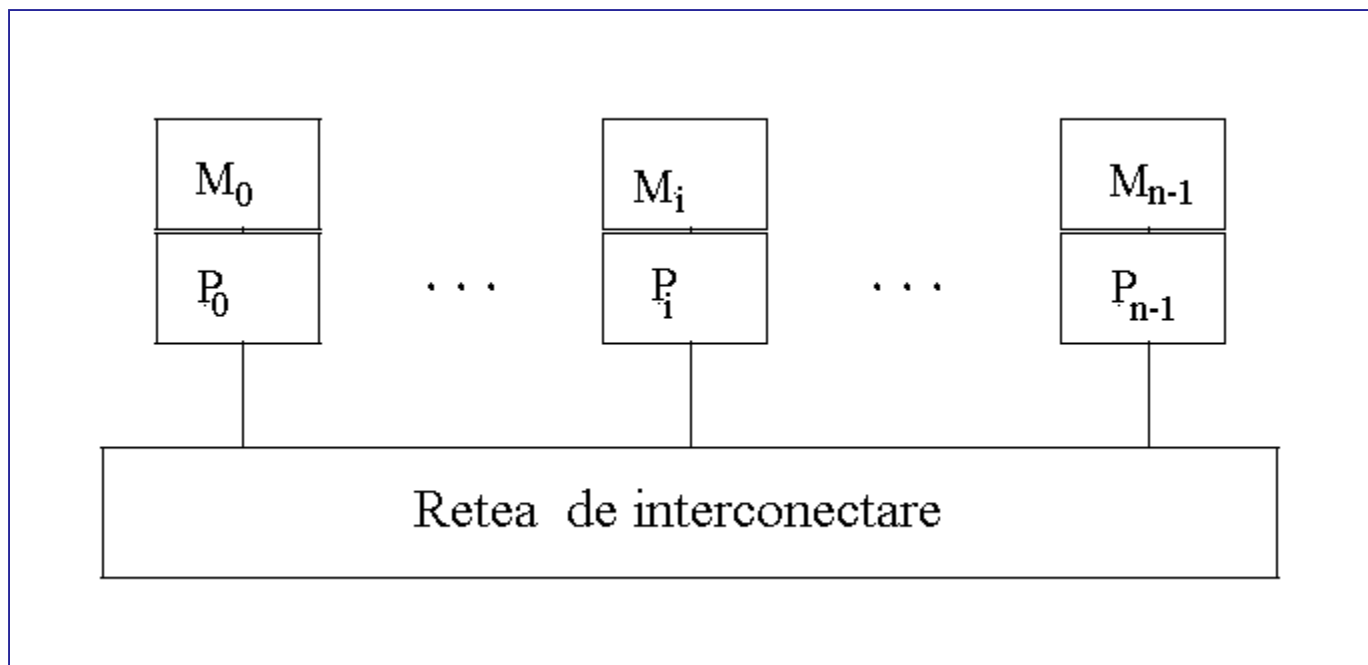


MIMD cu memorie comuna



- Fiecare procesor executa instructiuni proprii
- Datele sunt pastrate in memoria comuna
- De obicei fiecare procesor are de asemenea o mica memorie proprie (nu este reprezentata in schema simplificata)
- Operatiile care se pot executa cu usurinta: operatii paralele, la nivel de bloc
- Avantaj: comunicarea se face simplu, prin intermediul memoriei comune (partajate);
- Dezavantaj: cu cat creste numarul de procesoare, cu atat creste probabilitatea conflictelor de tip acces la memorie; drept consecinta, viteza de calcul va scadea

MIMD cu memorie distribuita



- In aceasta structura, fiecare procesor are o memorie proprie;
- Comunicarea se face printr-o retea de interconectare, prin intermediul unor mesaje explicite;
- Operatiile care se pot efectua cu usurinta sunt operatii paralele, la nivel de bloc;
- Avantajul unei astfel de structuri: reseaua de interconectare se poate optimiza pentru a deservi cat mai bine un anumit algoritim;
- Dezavantaj: comunicatia prin mesaje necesita implementarea unor algoritmi dedicati;

MIMD cu memorie distribuita: topologii

- MIMD cu memorie distribuita fixa – se implementeaza diferite topologii ale retelei de comunicatie:
 - Inel;
 - Grid;
 - Tor;
 - Star;
 - Hypercub;
 - Fat tree;
- Exista (pentru mai multe limbaje) biblioteci de functii de comunicatii;
- In calculatoarele actuale, topologia este de obicei invizibila pentru utilizator;
- Numarul de procesoare poate sa fie foarte mare, de ordinul zecilor si sutelor de mii de procesoare;
- Puterea de calcul poate sa fie de peste 10 petaflops / secunda;



MIMD cu memorie distribuita: grafuri

Grafuri

- O retea de comunicatie se modeleaza in mod natural printr-un graf neorientat; in acest graf, procesoarele sunt nodurile, iar liniile de comunicatie intre procesoare sunt arcele;
- Un graf neorientat $G = (V, E)$ (vertices, edges) – cu:
 - V – numarul de noduri;
 - E – numarul de arce;
- Numarul de noduri ale unui graf se numeste **ordinul grafului**: in acest caz este numarul de procesoare;
- **Graf simplu**: nu are bucle;
- Doua noduri sunt **adiacente** daca sunt unite de un arc;
- **Gradul unui nod**: numarul de conexiuni ale acelui nod, adica numarul de arce care pleaca din acel nod;
- **Gradul unui graf** (Δ): maximul gradelor nodurilor componente;
- **Graf regulat**: gradele tuturor nodurilor sunt egale;
- Numarul de arce ale unui graf regulat este: $n \cdot \Delta / 2$ (Ex. cub: $N = 12 = 8 \cdot 3 / 2 = 12$)
- **Drum** (sau cale) intre doua noduri x, y din V : este o secventa de noduri x_0, x_1, \dots, x_k astfel incat doua noduri consecutive sunt adiacente si $x = x_0$ si $y = x_k$.
- **Drum elementar**: nodurile din secventa drumului sunt distincte;

MIMD cu memorie distribuita: grafuri

- Intr-o arhitectura cu memorie partajata, un drum este o cale de comunicatie intre doua procesoare;
- **Lungimea unui drum**: numarul de arce ale sale; in notatia de mai sus, lungimea este k . Pentru o pereche de noduri, (x, y) , pot exista unul sau mai multe drumuri;
- **Distanta** intre doua noduri x si y : $dist(x, y)$ este drumul minim intre cele doua noduri;
- **Diametrul grafului**: maximum tuturor distantelor intre perechi de noduri ale grafului, adica distanta dintre cele mai departate noduri.
- **Ciclu**: un drum elementar pentru care nodul initial este identic cu nodul final;
- **Graf conex**: exista cel putin un drum intre oricare doua noduri ale sale;
- **Arbore**: graf conex fara cicluri;
- **Ciclu hamiltonian**: ciclu care trece (o singura data) prin fiecare nod al grafului; lungimea sa este egala cu ordinal grafului (numarul de noduri ale grafului); cardinalitatea arborelui grafului este egala cu $n-1$, unde n este numarul de noduri (ordinul grafului). Astfel, un ciclu hamiltonian va completa un arbore cu o coarda in asa fel incat ultimul nod sa fie unit cu primul nod.
- Vom prezenta in cele ce urmeaza mai multe topologii uzuale in arhitectura sistemelor paralele de tip MIMD cu memorie distribuita.

Cerintele pentru topologii

Topologii: cerinte hardware

- Graf conex
- Graf mic si regulat: echivalent cu un numar mic de procesoare, cu doar cateva canale de comunicatie, si usor de implementat fizic
- Numarul total de comunicatii este cat mai mic, conexiunea fiind realizata fie prin trasee electrice pe o placa fie prin conexiuni intre placi de retea;

Topologii: cerinte software

- Diametru mic al grafului in asa fel incat comunicatiile intre oricare doua procesoare sa fie cat mai facile si necostisitoare din punct de vedere al calculelor;

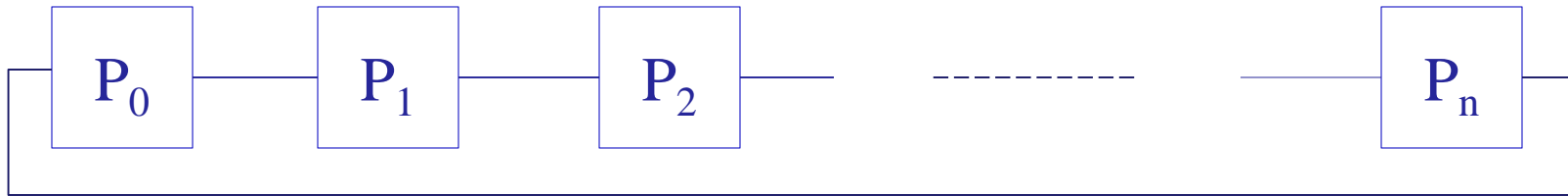
Discutie:

Graful de dimensiune mica presupune un numar mic de arce, ceea ce inseamna marirea diametrului (cu cat sunt mai putine arce, cu atat sansele sa existe drumuri scurte intre noduri scad);

Exista doua extreme:

- Graful total interconectat; diametrul $D = 1$; $\Delta = n-1$; numarul de arce: $n(n-1)/2$;
- Inelul; $D = n-1$; $\Delta = n-1$; numarul de arce: n ;

Topologii: inel



In aceasta topologie, fiecare procesor este conectat cu doua alte procesoare, vecinii sai din stanga si din dreapta; exceptie fac primul si ultimul procesor, care sunt conectate intre ele;

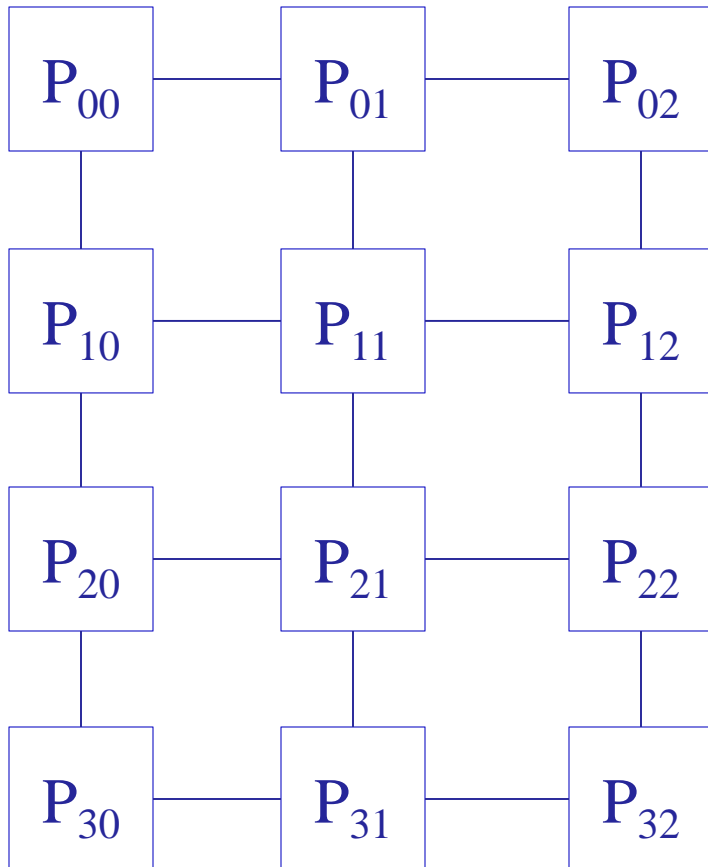
Gradul: $\Delta = 2$ (fiecare procesor este conectat cu alte doua);

Graful este regulat;

Numarul de arce: n (egal cu numarul de noduri);

Diametrul: $D = n / 2$ (unde n este numarul de noduri);

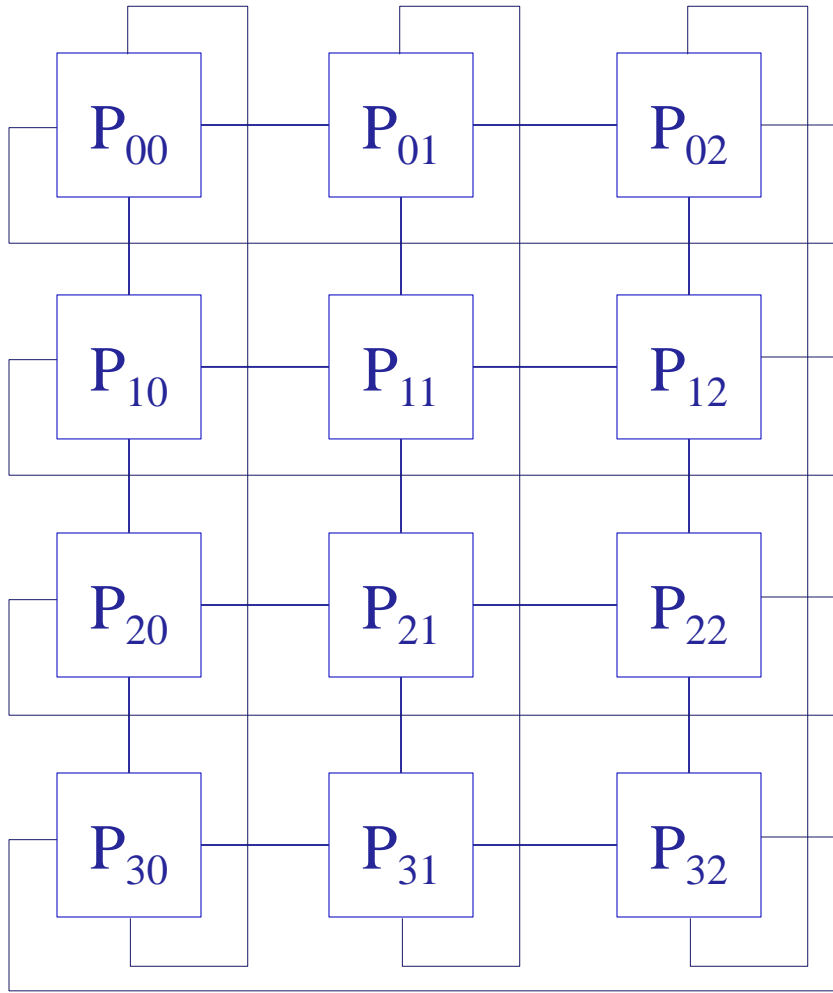
Topologii: grid



Fie un grid cu dimensiunile (in noduri) $n \times m$.

- Gradul: $\Delta = 4$ (fiecare procesor este conectat cu alte 2,3 sau 4);
- Graful este neregulat;
- Numarul de arce: $(n-1)(m-1)$;
- Diametrul: $D = (n+m-2)$ (unde n si m sunt numerele de noduri de pe cele doua directii);

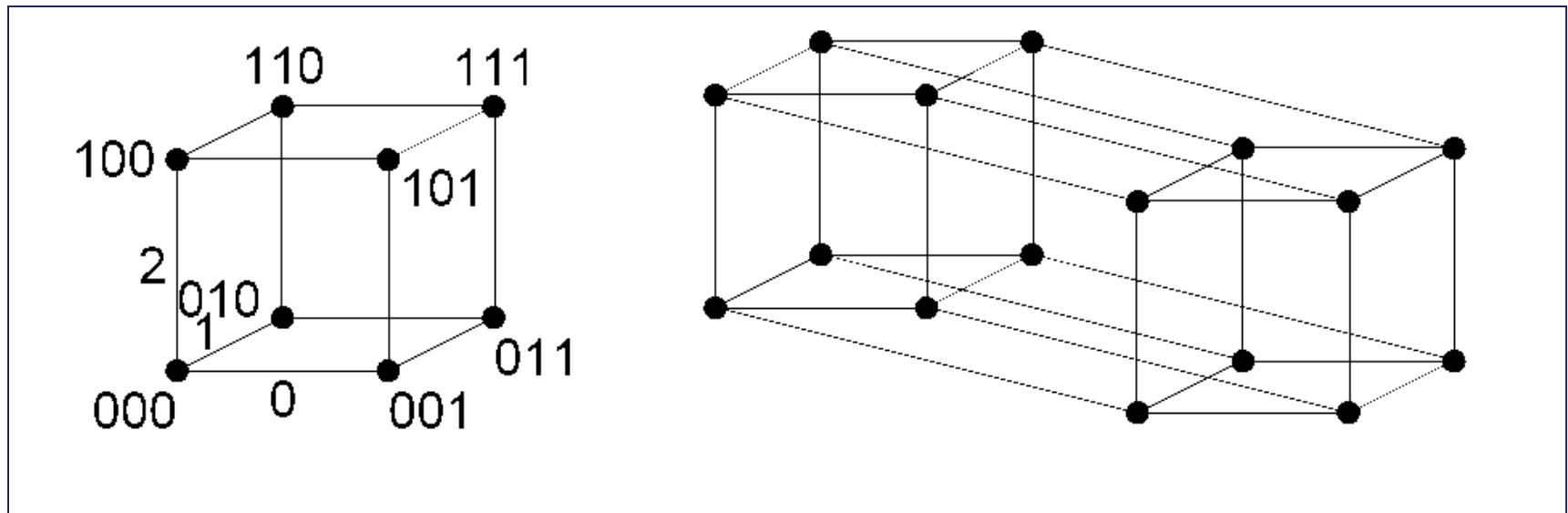
Topologii: tor



Fie un tor cu dimensiunile (in noduri) $n \times m$.

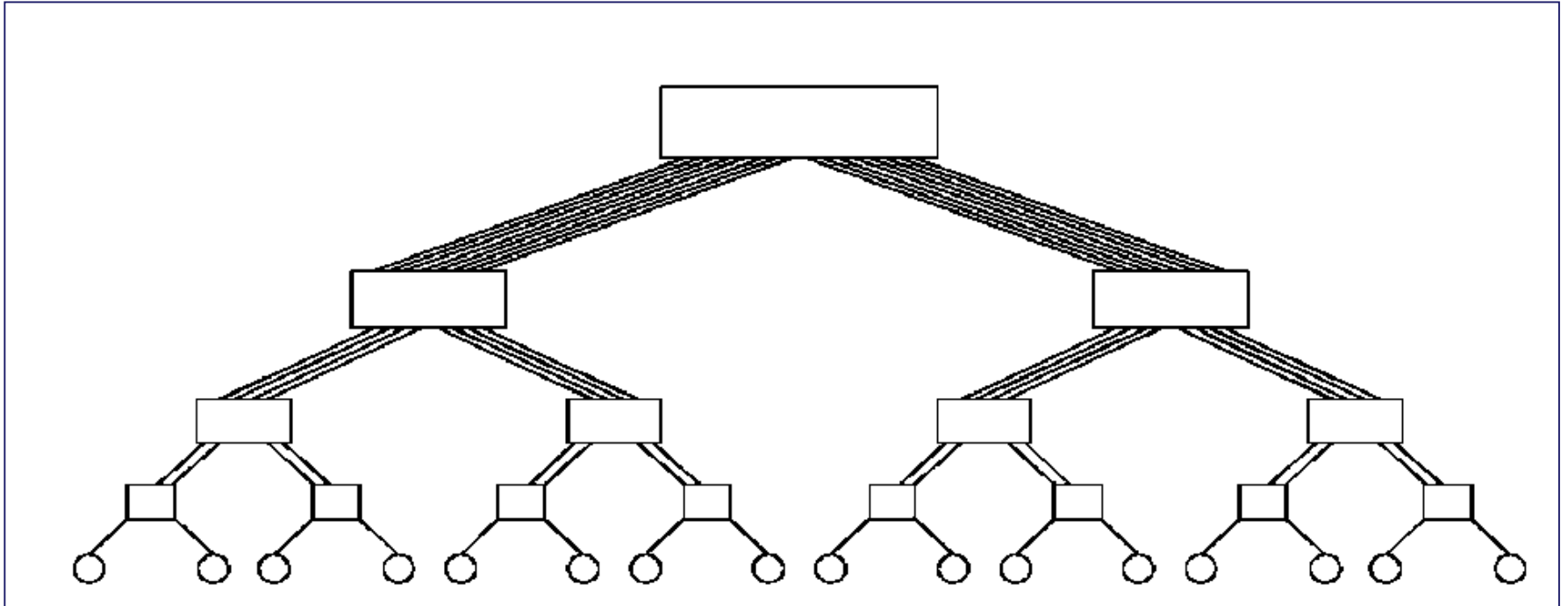
- Gradul: $\Delta=4$ (fiecare procesor este conectat cu alte 4);
- Graful este regulat;
- Numarul de arce: $n \times m$;
- Diametrul: $D = (n+m)/2$ (unde n si m sunt numerele de noduri de pe cele doua directii);
- Pentru un tor patrat:
 - Gradul: $\Delta=4$;
 - Numarul de arce: $n \times n$;
 - Diametrul: $D=n$;

Topologii: hipercub



- Un hipercub \mathcal{H}_d de dimensiune d are un numar de noduri $n = 2^d$. Exemplu: un hipercub de dimensiunea 3 va avea $n = 8$ noduri (cubul in spatiul 3-dimensional); numarul de vecini ai fiecarui nod este d ;
- Un nod i este legat de un nod j daca reprezentarile binare ale celor doua noduri difera doar cu un singur bit; Ex: 001 este vecin cu 011, 101 si cu 000 ($d=3$); 0001 este vecin cu 0011, cu 0101, 1001 si cu 0000 ($d = 4$). Dist. intre 2 noduri este distanta Hamming intre adresele lor;
- Gradul: $\Delta = d = \log n$;
- Diametrul $D = d$;
- Numarul de arce: $E = n \cdot \Delta / 2 = n \cdot \log n / 2 = n \cdot d / 2$;
- Un hipercub \mathcal{H}_d se obtine din conctarea nodurilor a doua hipercuburi \mathcal{H}_{d-1} .

Topologii: fat tree

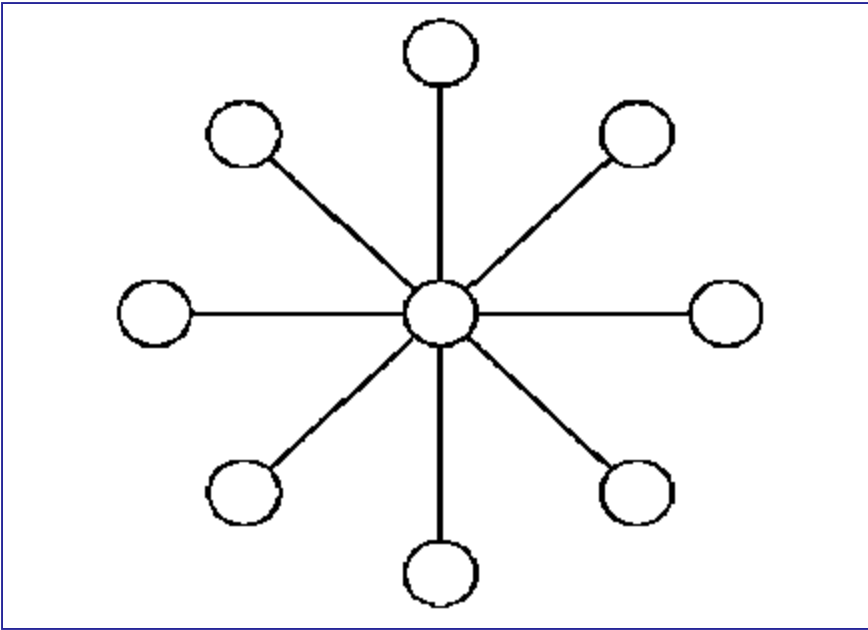


In aceasta topologie, nodurile de la nivelele superioare au un numar de conexiuni egal cu numarul total de noduri din frunzele ramurilor dependente;

Gradul: $\Delta = 2^{h+1}$ (unde h este adancimea arhorelui);

Numarul de arce: $E = h^2$ (egal cu numarul de noduri);

Topologii: star



Aceasta topologie este specifica unei arhitecturi statice cu procesoare cu o memorie partajata

Gradul: $\Delta = n-1$ (n e numarul de noduri);

Numarul de arce: $n-1$;

Diametrul: $D = 2$;