Capítulo 9
Multiprocessadores

Introdução

- Processamento paralelo:
  - programa sendo executado por múltiplos processadores simultaneamente
- Questões básicas no projeto de sistemas multiprocessados:
  - Compartilhamento de dados
  - Coordenação entre os processadores
  - Quantidade de processadores

Introdução

- Compartilhamento de dados
  - Processadores com um único espaço de endereçamento = Processadores de memória compartilhada (comunicação via variável comum)
    - UMA – *uniform memory access* ou SMP – *symmetric multiprocessors*
    - NUMA – *nonuniform memory access*
- Processadores de memória não-compartilhada (*private* ou memória distribuída)
  - Comunicação entre processos via troca de mensagens
    - *send* e *receive*

Processadores em Rede

- Modelo de paralelismo mais recente
  - *clusters*
  - computadores ligados por uma rede

Categoria	Tipo		Número de processadores
Modelo de Comunicação	Troca de Mensagens		8-256
	Endereçamento compartilhado	*NUMA*	8-256
		UMA	2-64
Conexão Física	Rede		8-256
	Bus		2-32

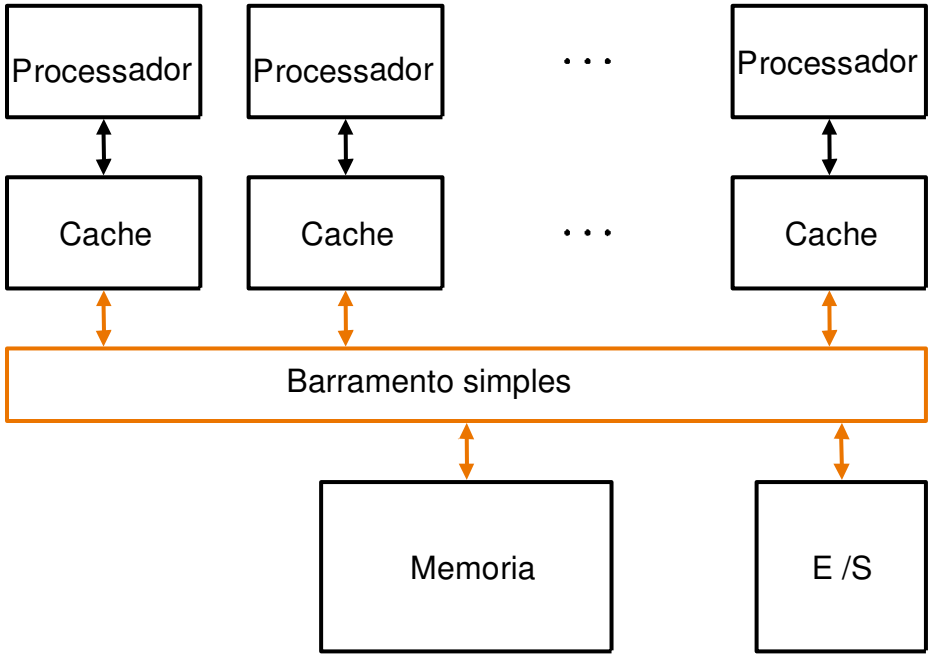
Speedup com Multiprocessamento e a Lei de Amdhal

- Achar o speedup linear com 100 processadores. Que fração da computação original pode ser seqüencial ?
  - Lei de Amdhal:
    - Tempo de execução após melhora = (tempo de execução afetado pela melhora / quantidade de melhora) + tempo de execução não afetado
  - Se buscamos aceleração linear $\Rightarrow$
    - Tempo de execução após melhora / 100 = (tempo de execução afetado pela melhora / 100) + tempo de execução não afetado
    - Tempo de execução não afetado = zero
- Conclusão:
  - para alcançar aceleração (*speedup*) linear com 100 processadores nenhuma operação original pode ser seqüencial
  - ou para alcançar aceleração de 99 em 100 processadores, quantidade de operações lineares < 0,01%

Speedup Via Execução Paralela

- Exemplo: Duas somas:
  - duas variáveis escalares
  - duas matrizes 1000 X 1000
  - Qual o *speedup* para 1000 processadores ?
  - Assumindo tempo para uma soma = *t*
    - 1 soma não afetada pelo paralelismo e 1.000.000 afetadas
    - tempo de execução antes do paralelismo= 1.000.001 t
  - Tempo de execução depois do paralelismo = (tempo de execução afetado pela melhora / quantidade de melhora) + tempo de execução não afetado
  - Tempo de execução depois melhora = (1.000.000/1000)*t* + 1*t* = 1.001*t*
  - Aceleração = 1.000.001/1.001 = 999

Multiprocessadores Conectados por um Barramento Simples



Nome	# max. de proc	Nome do proc.	CK MHz	Max. mem /sist MB	Bw max/sist MB/seg
Compaq Proliant 5000	4	Pentium Pro	200	2.048	540
Digital AlphaServer 8400	12	Alpha 21164	440	28.672	2.150
HP 9000 K460	4	PA-8000	180	4.096	960
IBM RS/6000 R40	8	PowerPC 604	112	2.048	1.800
SGI Power Challenge	36	MIPS R10000	195	16.384	1.200
Sun Enterprise 6000	30	UltraSPARC1	167	30.720	2.600

Um Programa Paralelo

- Exemplo: Somar 100.000 números em um computador com 10 processadores ligados em um único barramento (memória compartilhada)

  Pn é o processador n (de 0 a 9)

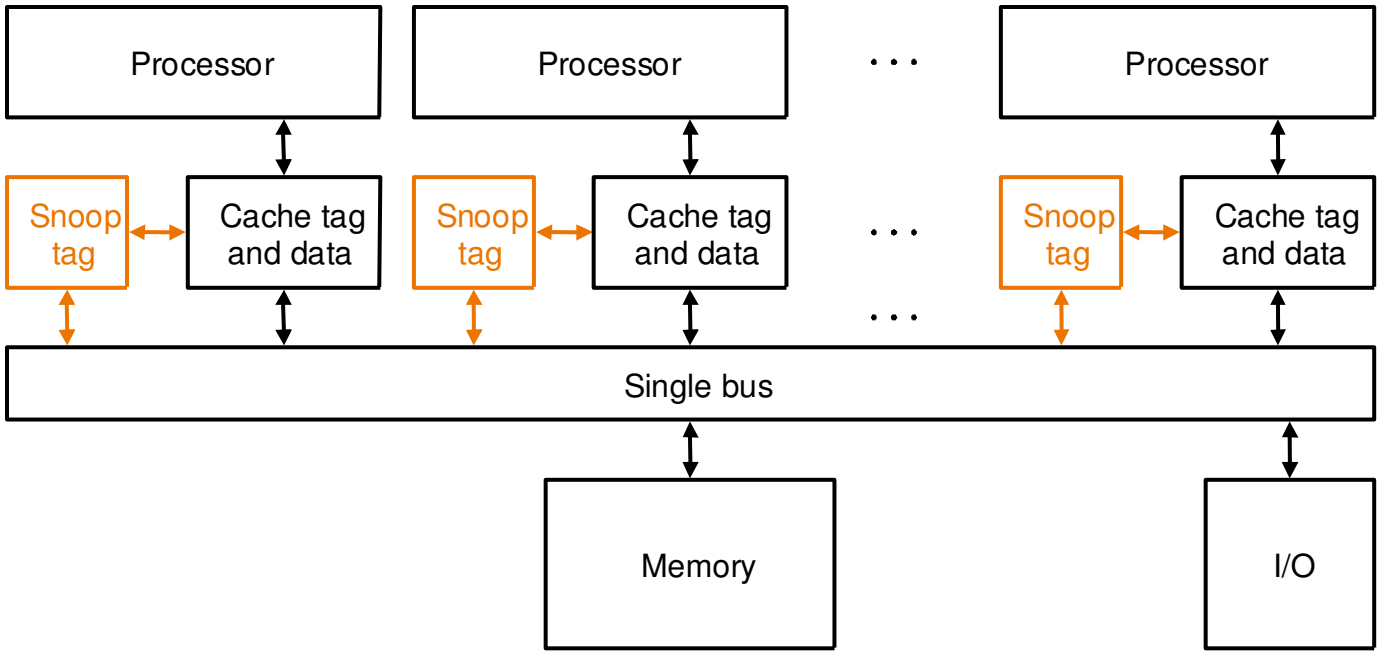
  1. Somar 10.000 números em cada processador (definir faixas de endereços de memória)

```
sum[Pn] = 0;
for (i = 10000*Pn; i < 10000*(Pn+1); i = i + 1;)
sum[Pn] = sum[Pn] + A[i] ; /* soma as regiões alocadas ao processador*/
```

  2. Somar as 10 sub-somas: metade dos processadores somam pares de somas parciais, um quarto somam pares das novas somas parciais, e assim por diante. "half" é uma variável privada

```
half = 10;
repeat
        synch(); /* primitiva de sincronização; espera a soma do bloco anterior*/
        if (half%2 != 0 && Pn ==0) /*condição para obter elemento "perdido" quando half é impar*/
            sum[0] = sum[0] + sum[half –1];
        half = half/2;
        if (Pn < half) sum[Pn] = sum[Pn] + sum[Pn+half];
until (half ==1);
```

Coerência de caches (*snooping*)



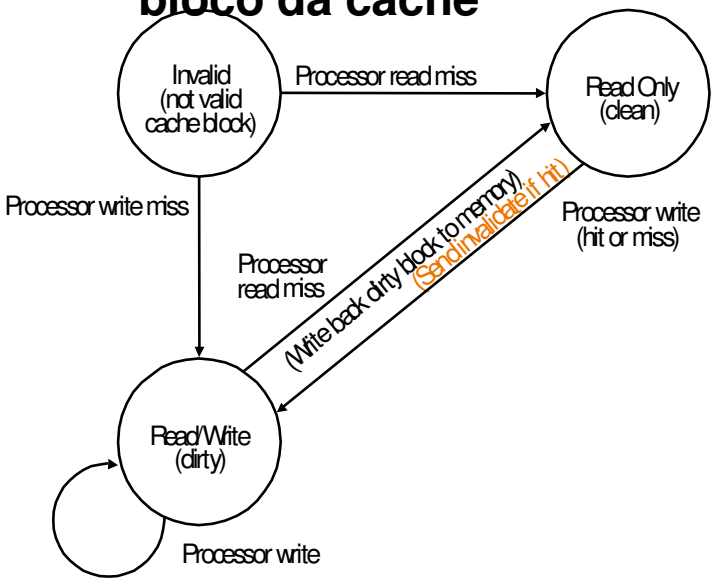
- ***Snoop*: controlador monitora o barramento para determinar se existe ou não uma cópia de um bloco compartilhado**
  - **Manutenção da coerência: problema só na escrita**
  - **Processador tem que ter acesso exclusivo na escrita**
  - **Todos os processadores tem que ter a cópia mais recente após uma escrita**

Protocolo *snooping*

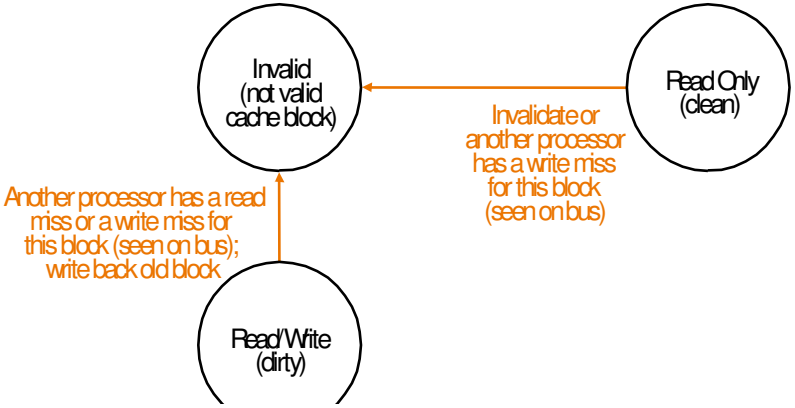
- *write-invalidate*
  - escrita ⇒ todas as cópias em outras caches tornam-se inválidas até a atualização
  - O processador que irá escrever manda um sinal de inválido no barramento e todas as caches verificam para ver se tem uma cópia
    - se sim, tornam o bloco que contem a palavra inválido
- *write-update (write-broadcast)*
  - processador que escreve propaga o novo dado no barramento e todas as caches com cópia são atualizadas

- Variações do protocolo snooping:
  - MSI (*Modified, Shared, Invalidate*)
  - MESI (*Modified, Exclusive, Shared, Invalidate*)

Conjunto de estados em cada bloco da cache

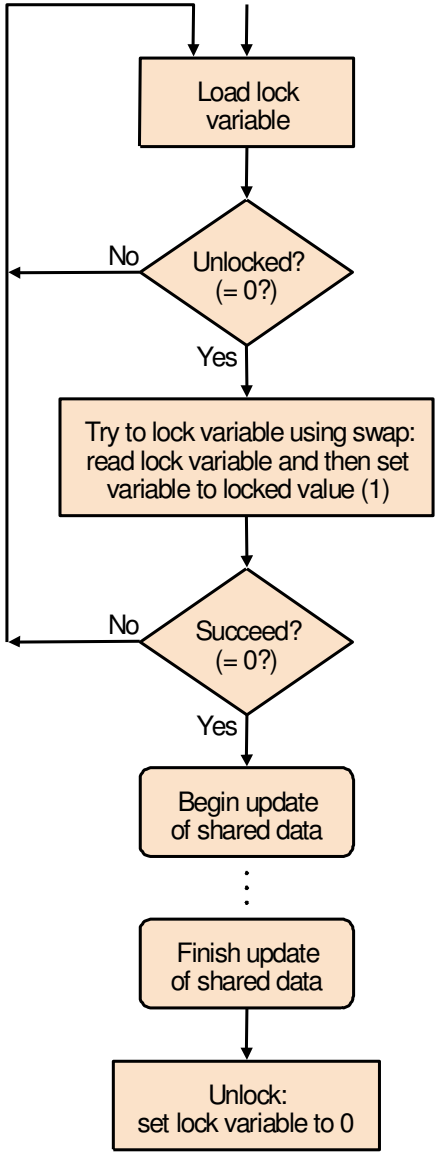


a) Transições de um bloco da cache de acordo com os sinais do processador

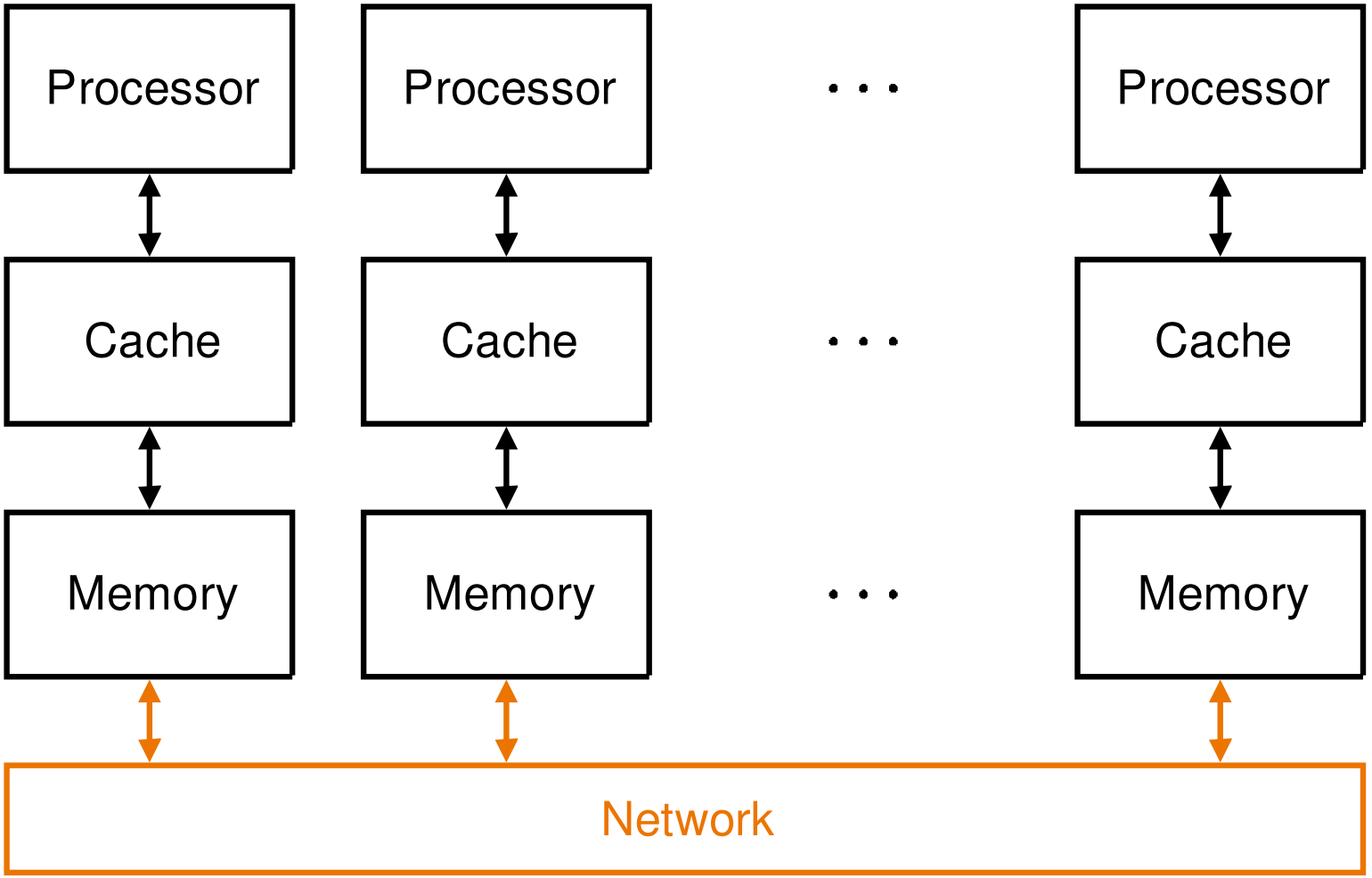


b) Transições de um bloco da cache de acordo com os sinais do barramento

Sincronização Usando Coerência



Multiprocessadores Conectados em Rede



Programa Paralelo em Rede

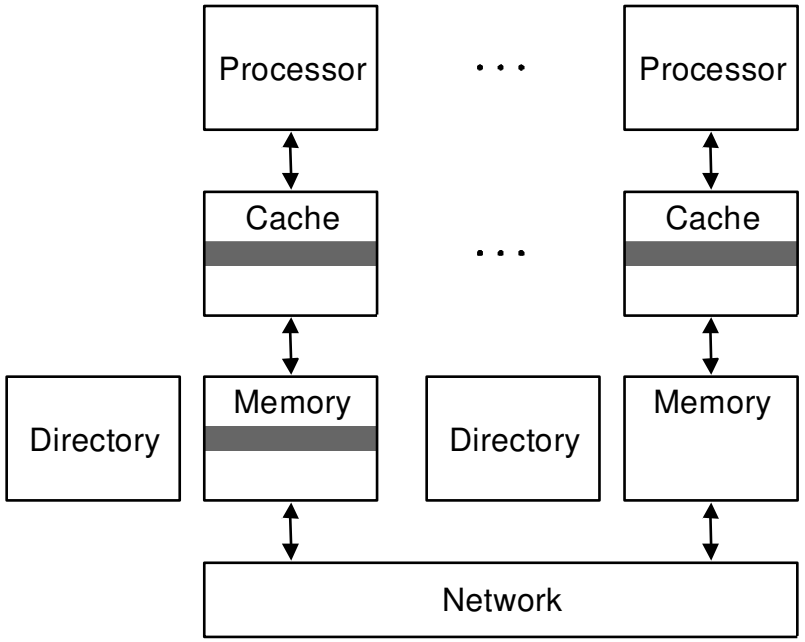
- **Programação Paralela e Troca de Mensagens:**
  - **somar 100.000 números em um computador com 100 processadores (com espaços de endereçamento privados) ligados a uma rede**
    - distribuir os 100 conjuntos de dados nas memórias locais e somar

```
sum = 0;
for (i = 0; i < 1000; i = i + 1;)
        sum = sum + A[i] ; /* soma arrays locais*/
```

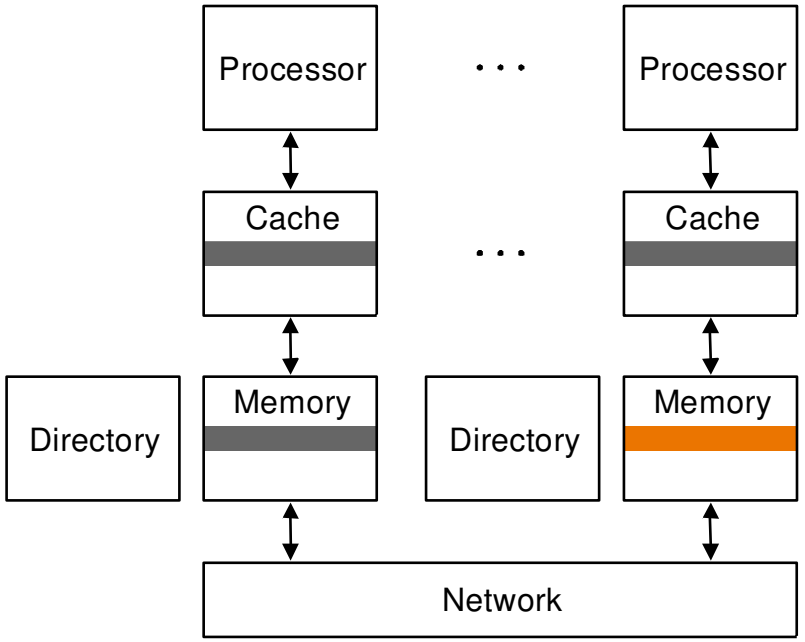
Somar as 100 sub-somas ⇒ send (x,y) onde x = processador Pn e y o valor

```
        half = 100;
        limit=half+1;
        repeat
                half =ceil((half+1)/2);
                if (Pn >= half && Pn < limit) send (Pn – half, sum)
                if (Pn < ceil(limit/2) -1) sum = sum + receive();
                limit = half;
        until (half ==1);
```

Endereçamento em Processadores Paralelos de Larga Escala



a.



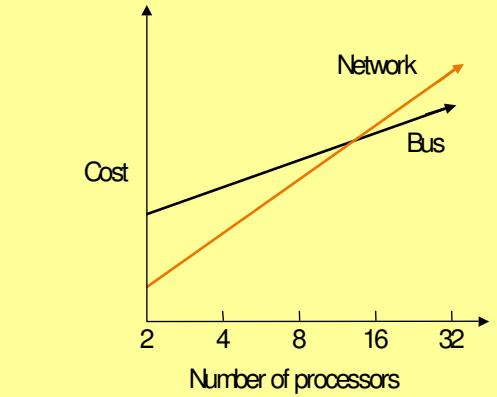
b.

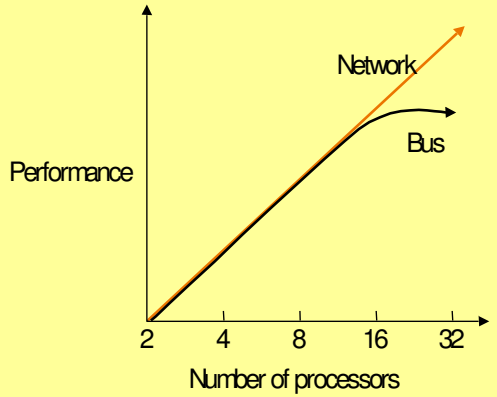
Coerência no nível de cache usando diretórios; dados originais na memória e cópias replicadas apenas nas caches

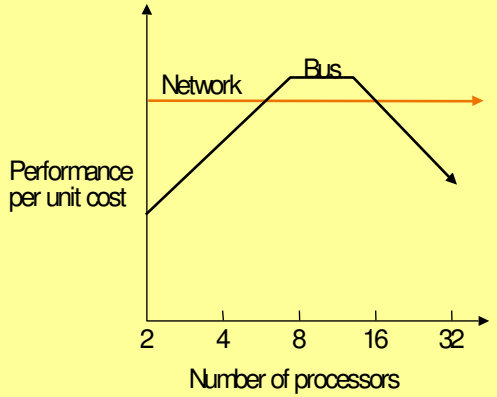
Coerência no nível de memória usando diretórios; cópias replicadas em memória remota (cor) e nas caches

Desempenho de Sistemas com Barramentos e Sistemas de Interconexão

Custo, desempenho e custo/desempenho em processadores conectados via barramento ou rede







Clusters

N máquinas têm N memórias independentes e N cópias do SO

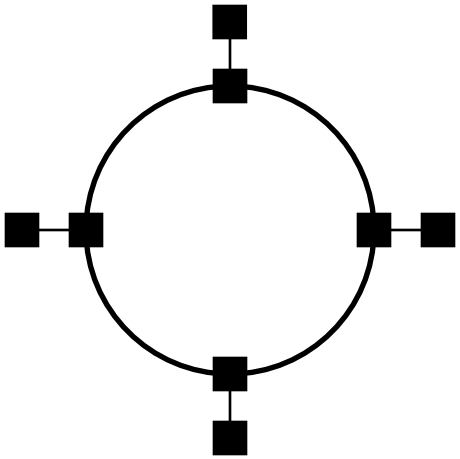
Nome	N° max proc.	Proc.	CK (MHz)	Max MemSize / Syst (MB)	BW / link	Nó	N° max Nós
HP 9000 EPS21	64	PA-8000	180	65.356	532 MB/s	4-way SMP	16
IBM RS/6000 HACMP R40	16	PowerPC 604	112	4.096	12 MB/s	8-way SMP	2
IBM RS/6000 SP2	512	Power2 SC	135	1.048.576	150 MB/s	16-way SMP	32
SUN Enterprise 6000 HA	60	Ultra SPARC	167	61.440	100 MB/s	30-way SMP	2
Tandem NonStop Himalaya S70000	4096	MIPS R10000	195	1.048.576	40 MB/s	16-way SMP	256

Topologia de Redes (Sistemas) de Interconexão

- **Dois extremos:**
  - ***single bus*** ⟷ **totalmente conectado (grafo completo)**

- Rede: grafo, links são bidirecionais, podem existir chaves (switches)

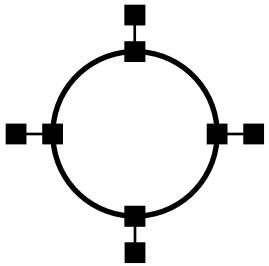
- Exemplo: Anel, mais de uma transação simultânea

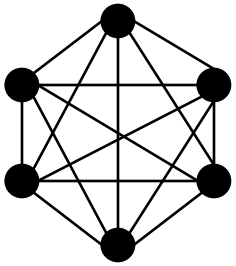


Medidas de Desempenho de Rede:

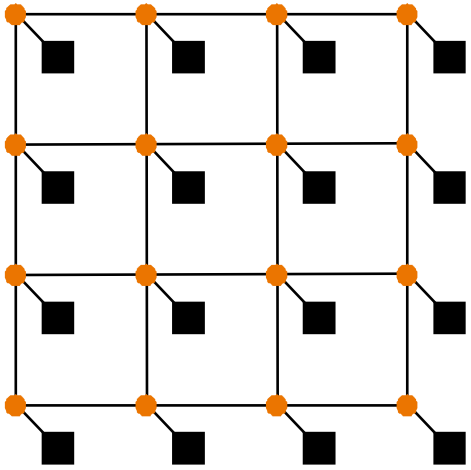
- *Total Network Bandwidth* (largura de banda) = soma dos *bandwidths* de todos os *links* de comunicação
  - *single bus* = 1
  - anel = N //Número de nós (processadores) interconectados
  - Totalmente conectado (grafo completo) = N (N - 1) / 2

- *Bisection Bandwidth:* dividir a rede em duas sub-redes com N/2 nós (da maneira mais pessimista); Medida = BW entre as sub-redes
  - single bus = 1
  - anel = 2
  - fully connected = $(N/2)^2$

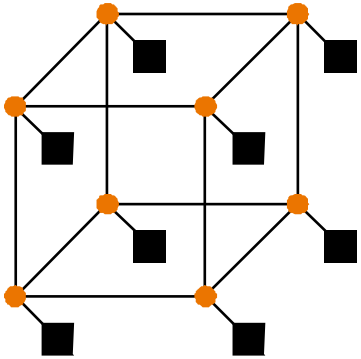




Duas Topologias de Interconexão: Mesh e Cubo



a. 2D grid or mesh of 16 nodes

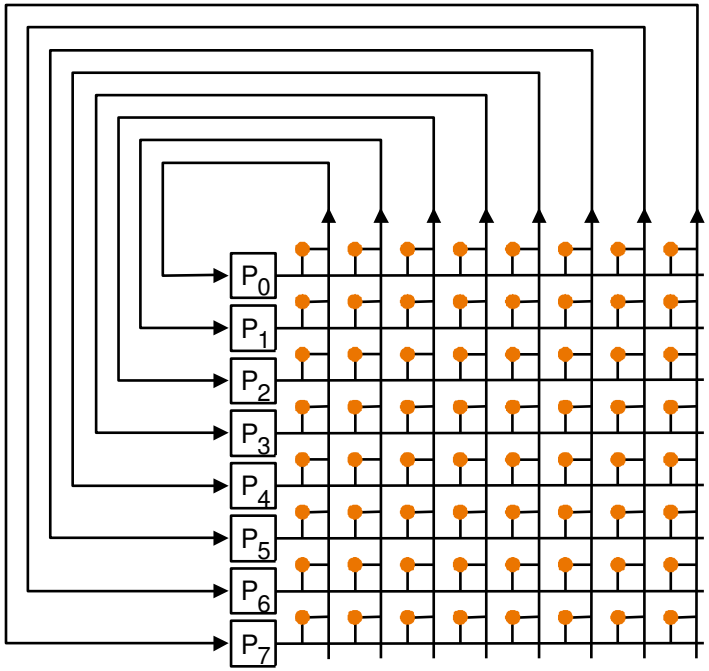


b. n-cube tree of 8 nodes ($8 = 2^3$ so $n = 3$)

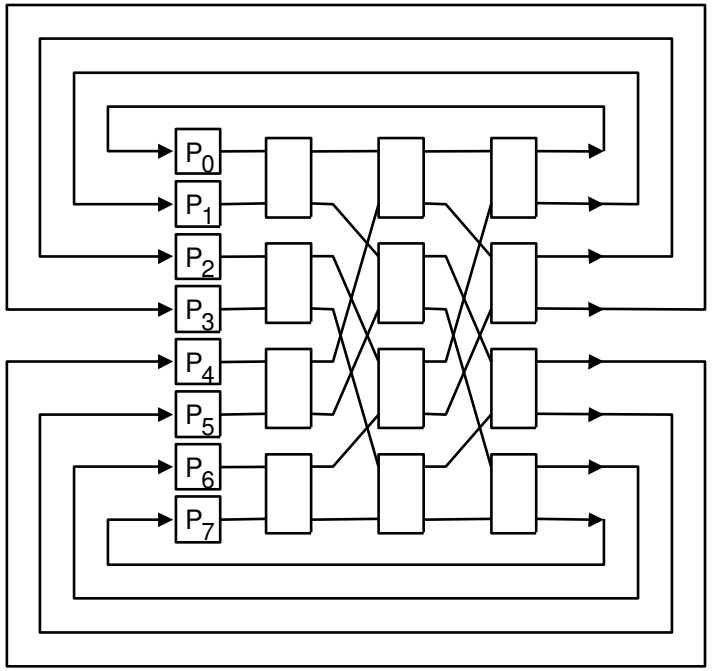
Bandwidths Para Redes com 64 Nós

		Barramento	Anel	Grade 2D	Cubo-6	Fully Connected
Desempenho	Total BW	1	64	112	192	2016
	Bisection BW	1	2	8	32	1024
Custo	Ports / switch	na	3	5	7	64
	Total # links	1	128	176	256	2080

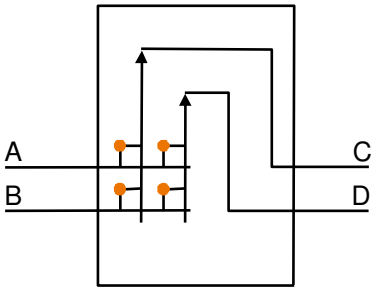
Topologia Multi-Estágio



a. Crossbar



b. Omega network



c. Omega network switch box

Classificação de Michael Flynn

- SISD: *Single Instruction (stream) Single Data (stream)*
  - processador único
  - single = o que?
  - soma paralela? pipeline?
  - referência para Flynn: single hoje = MIPS dos capítulos 5 e 6 (monociclo e multiciclo)
- SIMD:*Single Instruction Multiple Data*
  - operam sobre vetores de dados
- MISD: *Multiple Instruction Single Data*
  - difícil de relacionar com sistemas práticos
- MIMD: *Multiple Instruction Multiple Data*
  - multiprocessadores conectados via barramento ou rede