

Multicomputadores

Tópicos:

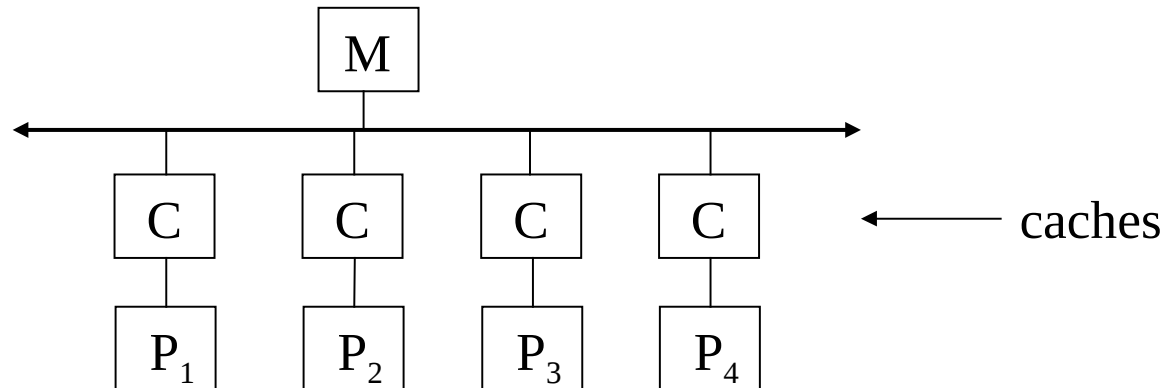
- Multiprocessadores (revisão)
- Multicomputadores
- Comunicação em Multicomputadores

Multiprocessadores

- Sistemas Paralelos de Memória Compartilhada:
 - Popularmente conhecidos como *Multiprocessadores*
 - Todos os processadores têm acesso a uma região comum (compartilhada) de memória
 - A memória comum pode estar num único módulo, ou distribuída pelos vários processadores
 - Nada impede que haja também módulos de memória que sejam particulares a um único processador
 - Em geral, a comunicação entre processadores ocorre através de posições da memória compartilhada
 - Programação: fork/join, pthreads, etc.

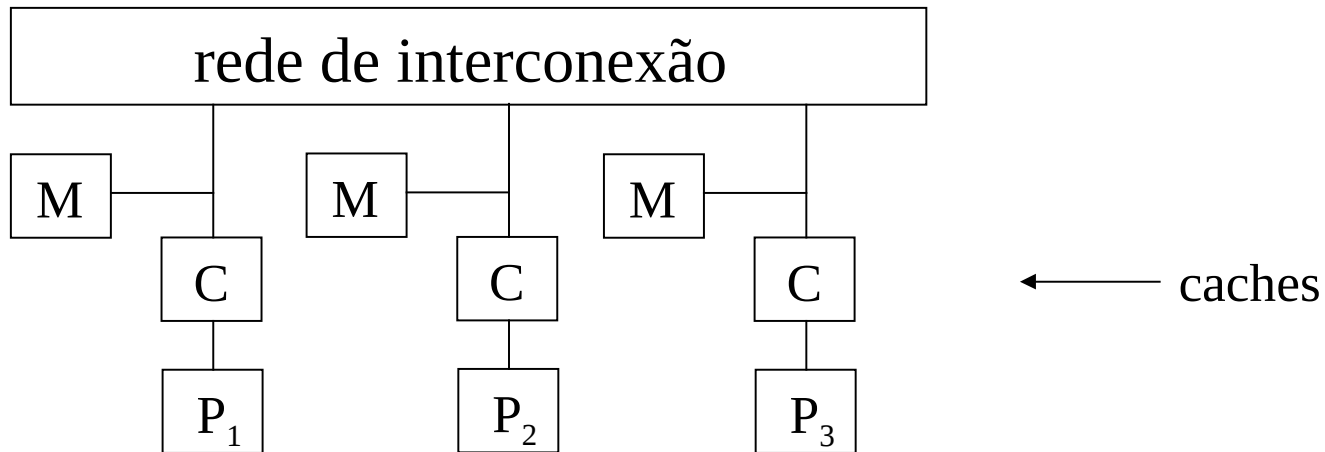
Multiprocessadores de Pequena Escala

- Baseados em Barramento:
 - Barramento comum limita o número máximo de processadores (da ordem de no máximo 30, na prática)
 - Muito populares em ambientes multi-usuários
 - Atualmente, quase todos os sistemas são do tipo *simétrico*
 - Ex: Sparc, PCs, SGI/Power-Challenge



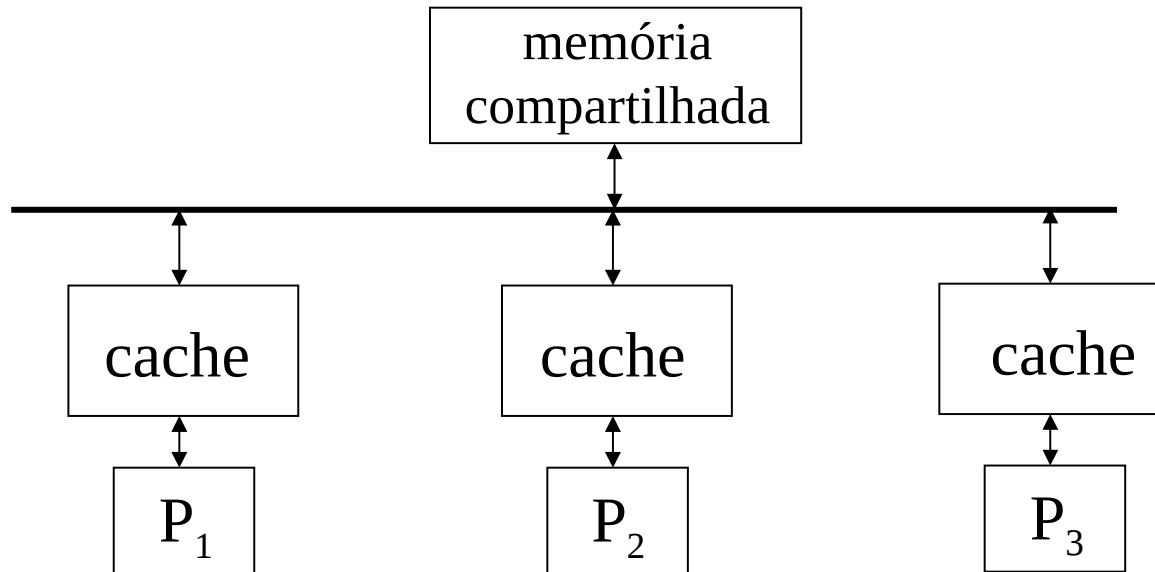
Multiprocessadores de Média Escala

- Baseados em Redes de Interconexão:
 - Como o uso da rede não é exclusivo, número de processadores pode ser maior
 - Acessos à memória podem gerar tráfego pela rede
 - Ex: SGI/Origin , Cray/T3E, etc.



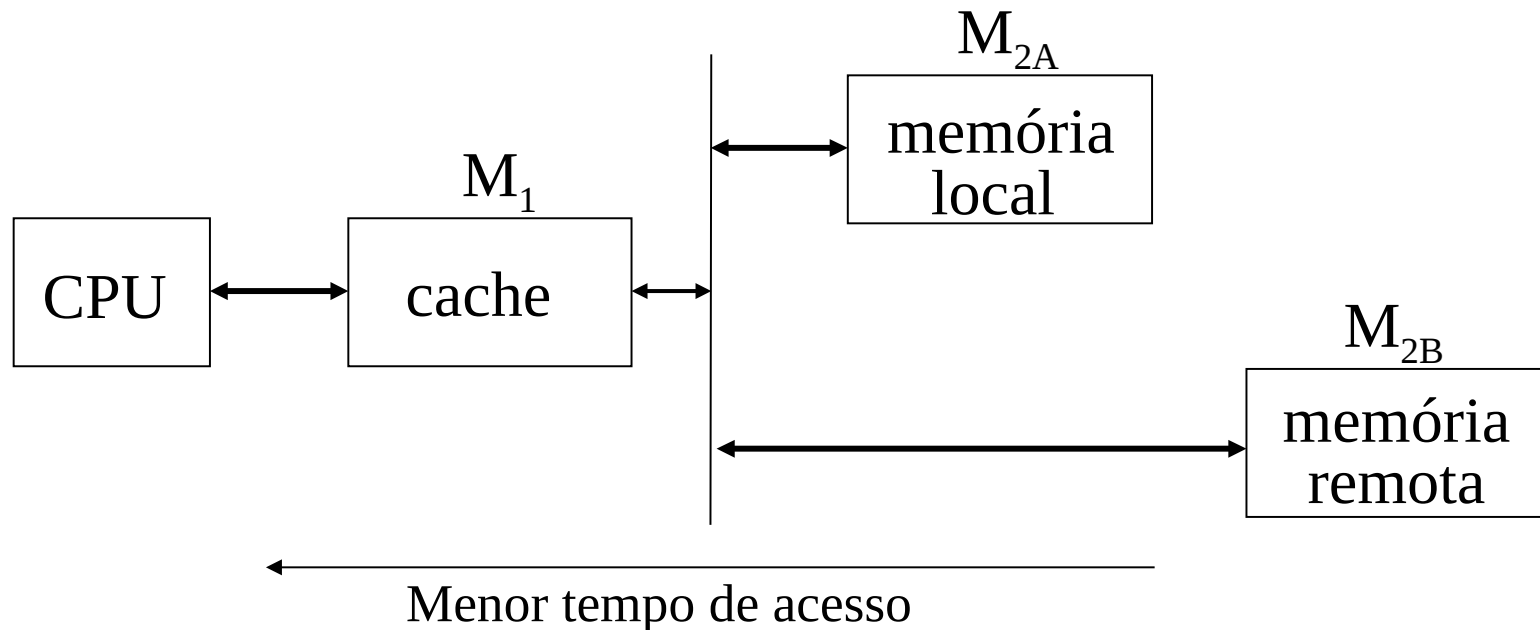
Organização de Multiprocessadores

- Visão lógica da memória em Multiprocessadores:



Organização de Multiprocessadores (cont.)

- Visão física da memória em Multiprocessadores:



*OBS: Todos os acessos são através de *read / write* (*load / store*)

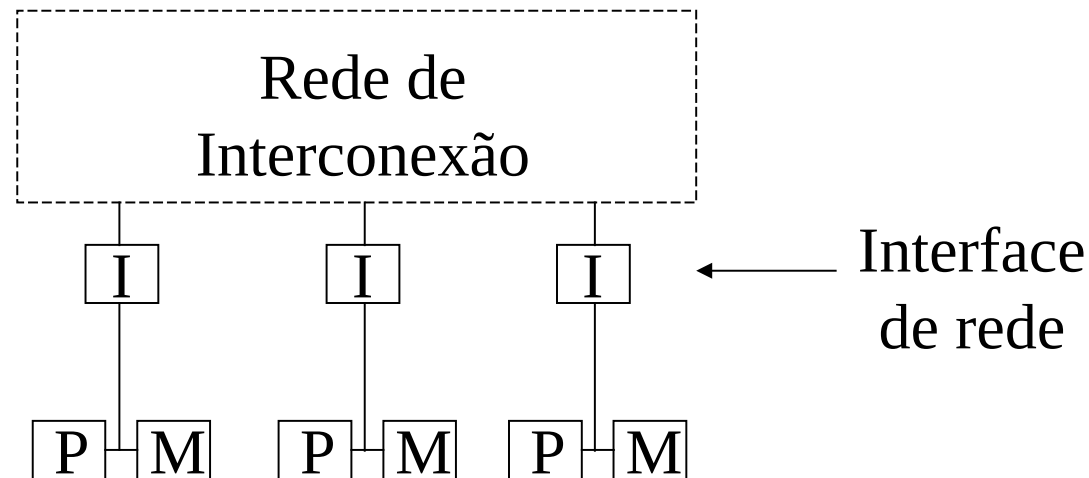
Multicomputadores

- Sistemas Paralelos de Memória Distribuída:
 - Popularmente conhecidos como *Multicomputadores*
 - Cada um dos processadores têm acesso a um espaço único de endereçamento de memória privativa
 - Um módulo da memória pode ser acessado diretamente por apenas um dos processadores
 - Em geral, a comunicação entre processadores ocorre através de *troca de mensagens*:
 - Um processador de origem *envia* dados (executa a função *send*)
 - Um processador de destino *recebe* dados (executa a função *receive*)

Organização de Multicomputadores

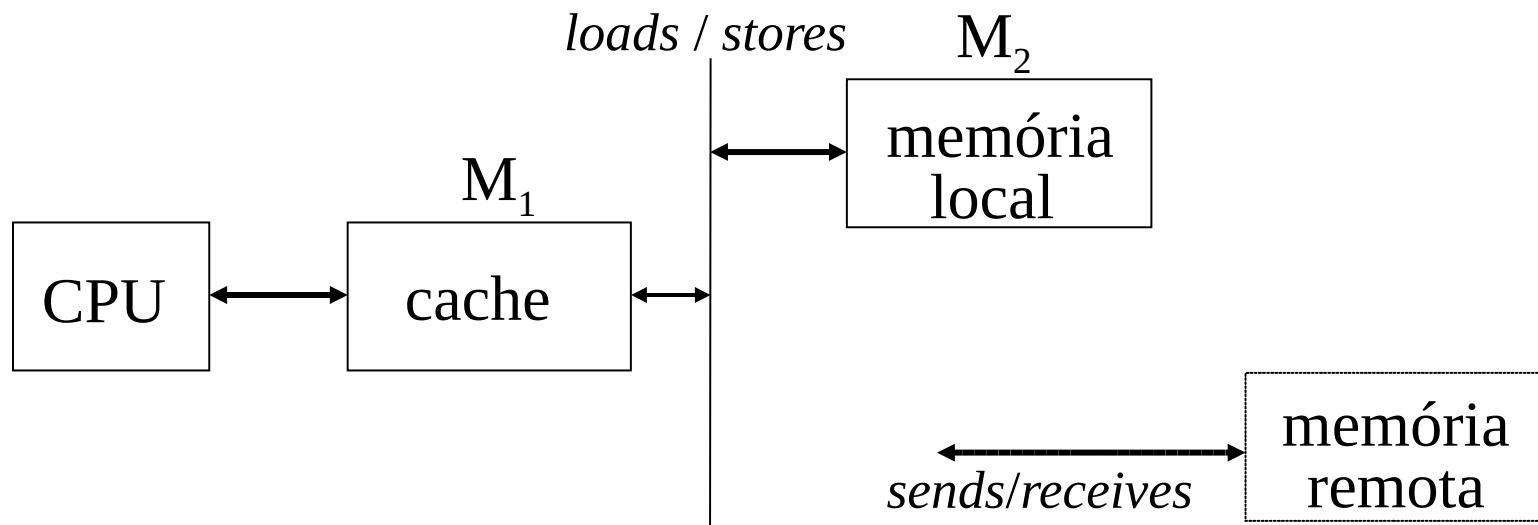
Memórias *locais* apenas:

- Acessos a dados locais: *reads/writes*
- Acessos a dados remotos: *sends/receives*



Organização de Multicomputadores

- Visão da memória em Multicomputadores:



Comunicação em Multicomputadores

- *Sends/Receives*:
 - Acessos à interface de rede
 - Em geral, chamadas ao Sistema Operacional
 - Deve haver proteção entre processos
 - Sistema Operacional gerencia a comunicação física
 - · Há um alto overhead de software associado!
- Otimização:
 - Agrupar dados em *mensagens* com vários itens
 - Fazer comunicação mem-local/mem-remota via DMA
 - CPU apenas inicia comunicação e aguarda término

Comunicação em Multicomputadores (cont.)

- Roteamento de mensagens pela rede:
 - *store-and-forward*:
 - Cada nó intermediário armazena toda a mensagem e a reenvia
 - Método utilizado nos primeiros multicomputadores
 - *cut-through*:
 - Dados não são armazenados pelos nós intermediários
 - Método mais utilizado atualmente (quase todos os sistemas)
 - É comum haver hardware especial de suporte

Obs: Normalmente, há armazenamento parcial de mensagens em trânsito, através do Sist.Operacional

Comunicação em Multicomputadores (cont.)

- Desempenho da comunicação:
 - Modelo mais utilizado:
 - tempo linearmente proporcional ao comprimento (n) das mensagens: $T(n) = a + b \cdot n$
 - onde:
 - a: latência efetiva (hardware + software)
 - b: custo por item adicional, após envio do primeiro item
 - *sends/receives*:
 - $T_{\text{Send}}(n) = a_s + b_s \cdot n$
 - $T_{\text{Recv}}(n) = a_r + b_r \cdot n$