SSC-0742
PROGRAMAÇÃO CONCORRENTE

Aula 04 – Revisão de Arquiteturas Paralelas -Parte 2

Prof. Jó Ueyama e Julio Cezar Estrella

Créditos

Os slides integrantes deste material foram construídos a partir dos conteúdos relacionados às referências bibliográficas descritas neste documento

Visão Geral da Aula de Hoje

1. Arquiteturas de Memória
2. Coerência de Cache
3. Exercício e Leitura Recomendada

ARQUITETURAS DE MEMÓRIA

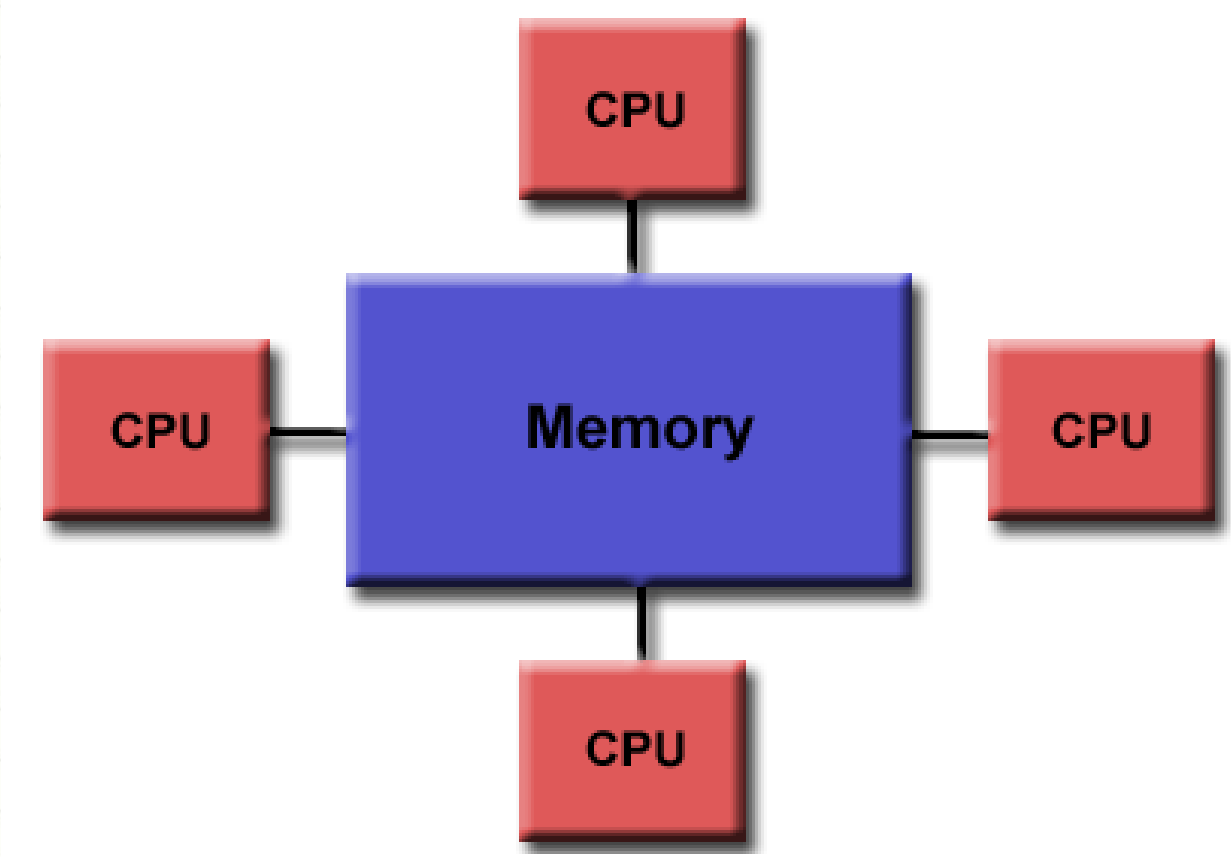
Arquiteturas de Memória

- **Memória Compartilhada**
  - Múltiplos processadores compartilham os mesmos recursos de memória em um espaço de endereçamento global
  - Modificações efetuadas em uma região de memória por um processador são visíveis a todos os outros processadores
  - Divisão em duas classes:
    - **UMA – tempo de acesso uniforme**
    - **NUMA – tempo de acesso não uniforme**

Memória Compartilhada - UMA

- **Uniform Memory Access (UMA)**
  - Comum em processadores simétricos (SMP)
  - Tempo de acesso igual para toda a memória
  - Coerência de Cache
    - Se um processador atualiza um dado na memória compartilhada, isso tem que ser propagado para os outros processadores (cache interna à CPU)
    - Normalmente feita em hardware

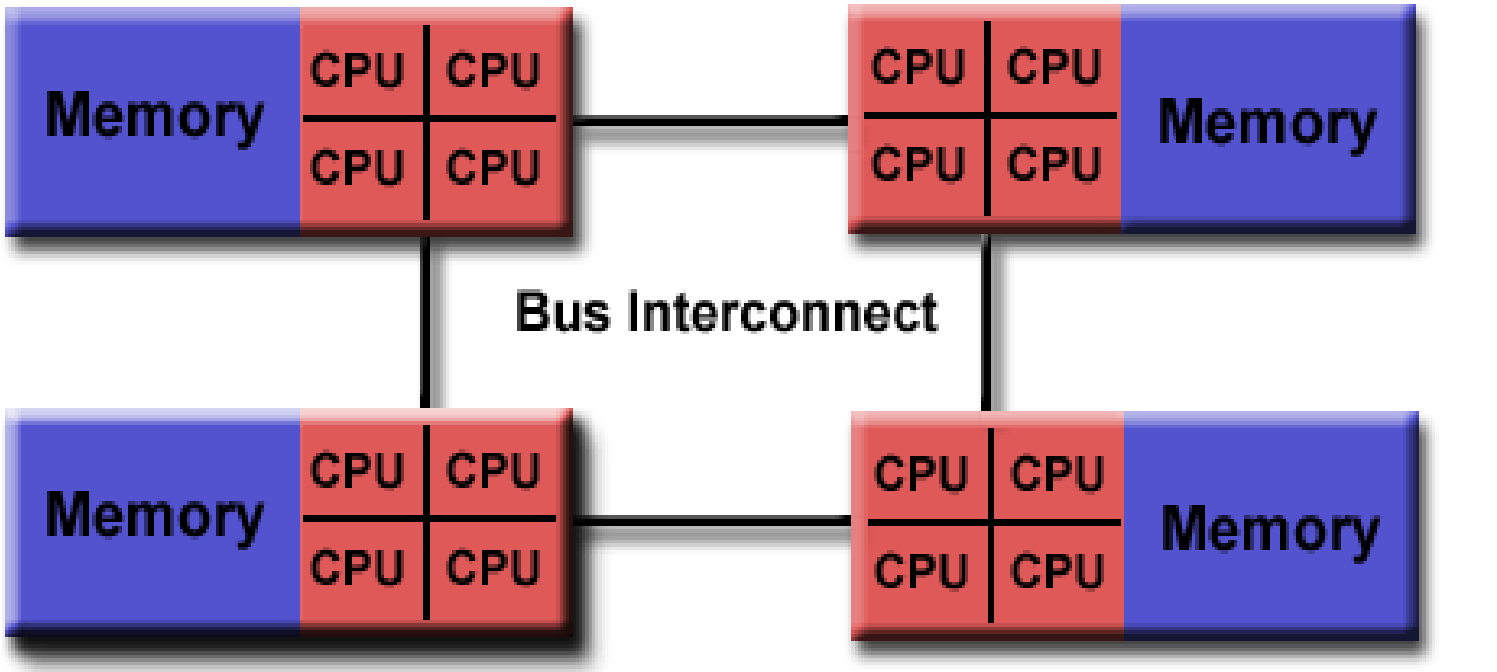
Memória Compartilhada - UMA



Memória Compartilhada - NUMA

- **Non-Uniform Memory Access**
  - SMPs conectados por canais de comunicação
  - O espaço de endereçamento ainda é global e um SMP pode acessar diretamente a memória do outro
  - O tempo de acesso à uma posição de memória varia dependendo se ela pode ser acessada localmente ou em um processador remoto através dos canais de comunicação
  - Se existe Coerência de Cache, a arquitetura é chamada de CC-NUMA (Cache Coeherent NUMA)

Memória Compartilhada - NUMA



Memória Compartilhada

- **Vantagens**
  - O espaço de endereçamento global permite um modelo de progamação mais amigável (conhecido)
  - O compartilhamento de dados entre as tarefas é rápido

Memória Compartilhada

- **Desvantagens**
  - Falta de escalabilidade. Aumentando o número de CPUs, aumenta o tráfego entre memória e CPU (bottleneck de von Neumann)
  - Para sistemas com coerência de cache
    - Aumento de tráfego associado com o gerenciamento de memória/cache
  - É de responsabilidade do programador a sincronização dos acessos à memória global compartilhada
  - Custo: alto para máquinas com o número elevado de processadores

Memória Distribuída

- **Memória Distribuída**
  - Sistemas de memória distribuída possuem uma rede de comunicação para interconectar os processadores
  - Cada processador tem sua própria memória local
  - Não existe um espaço de endereçamento global, um processador tem acesso somente à sua memória
  - Coerência de cache não se aplica. Por que?

Memória Distribuída

- É tarefa do programador determinar quando e como um dado é propagado de um processador para outro
- A sincronização das tarefas é também função do programador

Memória Distribuída

- **Vantagens**
  - A memória é escalável com o número de processadores
  - Mais Processadores => Maior Quantidade Memória
  - Acesso rápido a memória do próprio processador sem interferência e sobrecarga da coerência da cache
- **Desvantagens**
  - O programador precisa cuidar de muitos detalhes associados à comunicação de dados entre os processadores
  - Dificuldade de mapear estruturas de dados projetadas para uma memória global com esta organização de memória

Arquiteturas de Memória

- **Memória Compartilhada e Distribuída**
  - Os computadores maiores e mais rápidos do mundo empregam ambas as arquiteturas de memória: compartilhada e distribuída.
  - O componente de memória compartilhada é em geral um SMP com coerência da cache
  - O componente de memória distribuída é uma rede de SMPs.

Compartilhada e Distribuída

- Processadores em um SMP (várias CPU e uma memória) só endereçam diretamente a sua memória
- É necessária comunicação em rede para mover dados de um SMP para outro

- **Vantagens e Desvantagens**
  - Compartilha as vantagens e desvantagens das arquiteturas de memória compartilhada e distribuída, conforme o uso

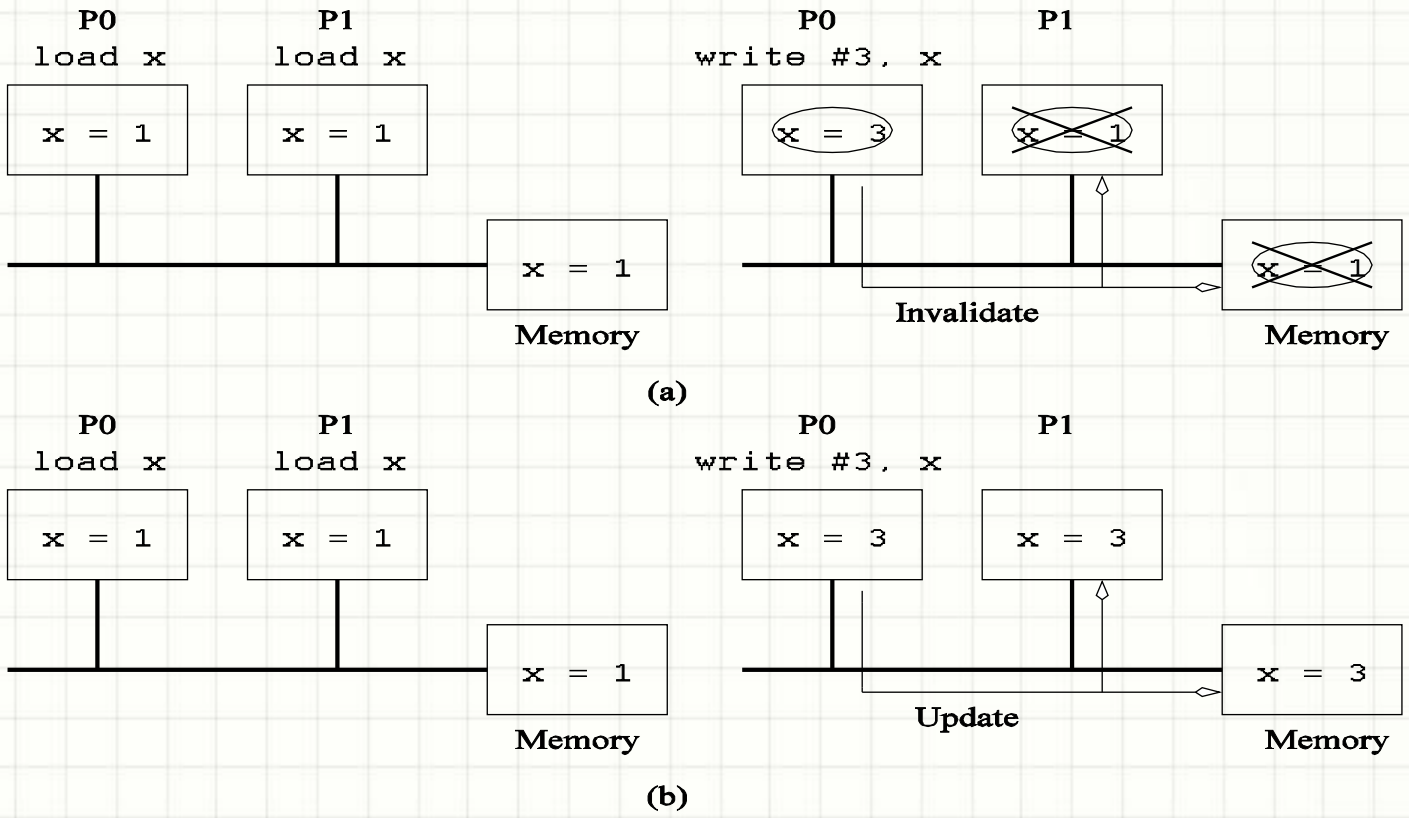
Coerência de Cache

Coerência de Cache

- Coerência de cache se refere à manutenção de informações coerentes nas caches dos diversos processadores que compõem uma arquitetura paralela
- No caso das máquinas com endereçamento global, um endereço de hardware adicional é necessário para coordenar o acesso a dados que podem ter várias cópias na rede

Protocolos de Validação e Invalidação

- Quando o valor de uma variável é modificado, as cópias podem ser anuladas ou atualizadas



Protocolos de Validação e Invalidação

- **Invalidação e Atualização**
  - Se um processador lê um valor apenas uma vez e não precisa dele novamente, um protocolo de atualização impõe um overhead
  - Se dois processadores fazem testes intercalados com atualizações em uma variável, um protocolo de atualização é melhor.
  - Ambos os protocolos sofrem falsas sobrecargas de compartilhamento (duas palavras que não são compartilhadas, porém ficam na mesma linha de cache).
  - A maioria das máquinas atuais usam protocolos de invalidação

Protocolos de Coerência de Cache

- Cada cópia de um item de dado é associada a um estado, por exemplo:
  - Compartilhado: Existem outras cópias do item de dado igualmente válidas
  - Sujo: É a única cópia válida
  - Inválido: A cópia de dados é inválida. Existe outra cópia mais atual

Protocolos de Coerência de Cache

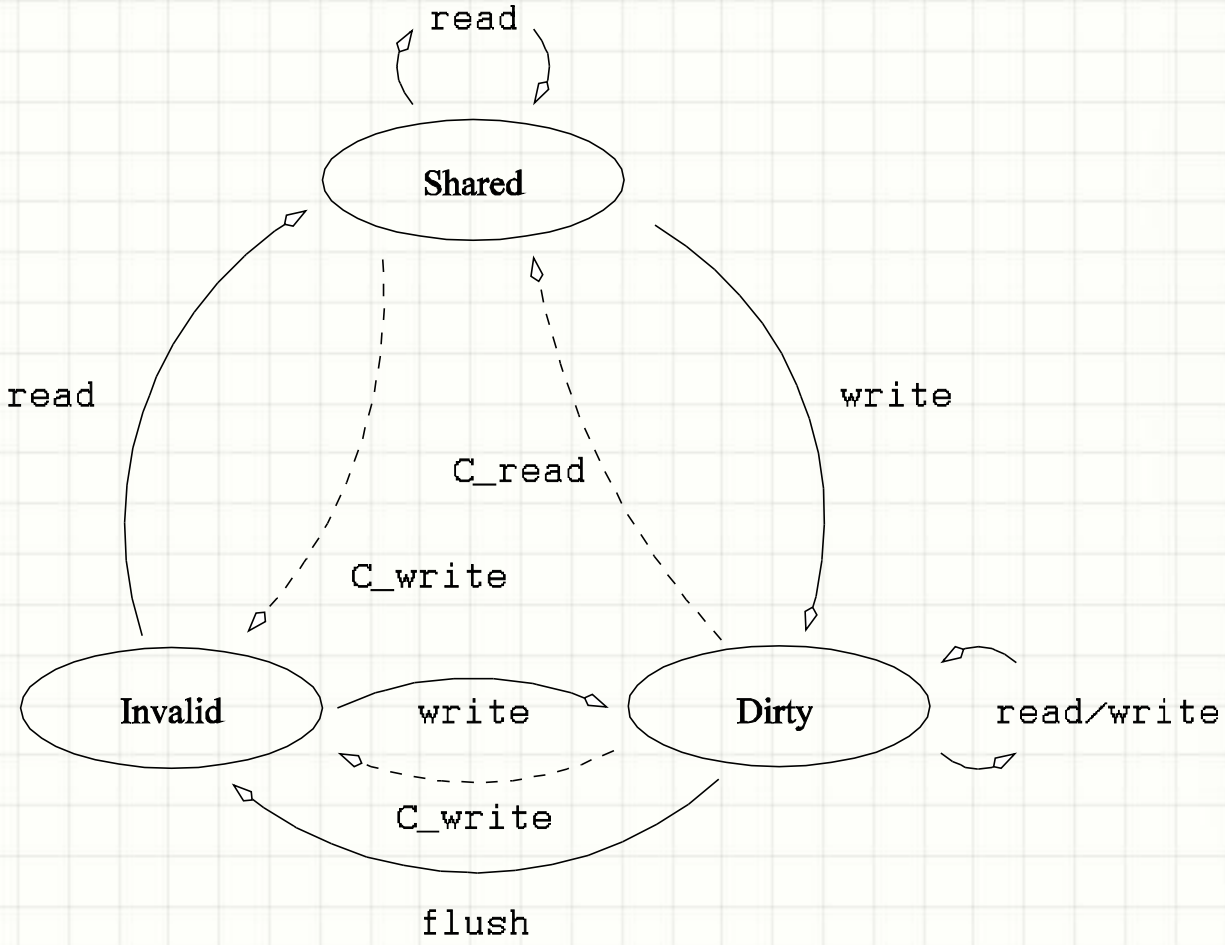


Diagrama de estados para um protocolo de coerência de três estados

Coerência de Cache

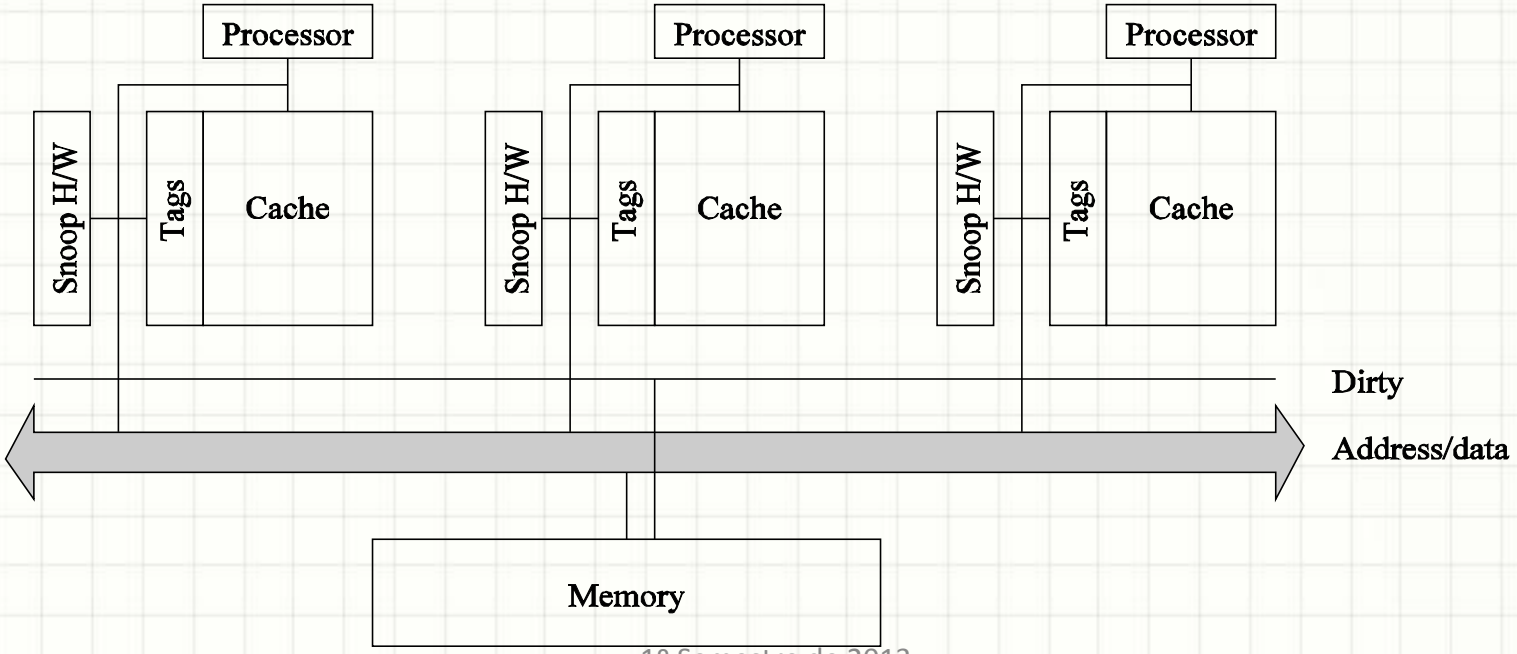
Time ↓

Instruction at Processor 0	Instruction at Processor 1	Variables and their states at Processor 0	Variables and their states at Processor 1	Variables and their states in Global mem.
				x = 5, D y = 12, D
read x		x = 5, S		x = 5, S
	read y		y = 12, S	y = 12, S
x = x + 1		x = 6, D		x = 5, I
	y = y + 1		y = 13, D	y = 12, I
read y		y = 13, S	y = 13, S	y = 13, S
	read x	x = 6, S	x = 6, S	x = 6, S
x = x + y		x = 19, D	x = 6, I	x = 6, I
	y = x + y	y = 13, I	y = 19, D	y = 13, I
x = x + 1		x = 20, D		x = 6, I
	y = y + 1		y = 20, D	y = 13, I

Exemplo de execução de um programa com o protocolo de coerência de três estados

Protocolos de Cache Snoopy

- Como as invalidações são enviadas aos processadores corretos?
  - Em um protocolo Snoopy, um hardware específico usa o barramento para monitorar quais são as operações que estão sendo feitas na memória, alterando o cache local de forma apropriada



Protocolo Snoopy de coerência de cache

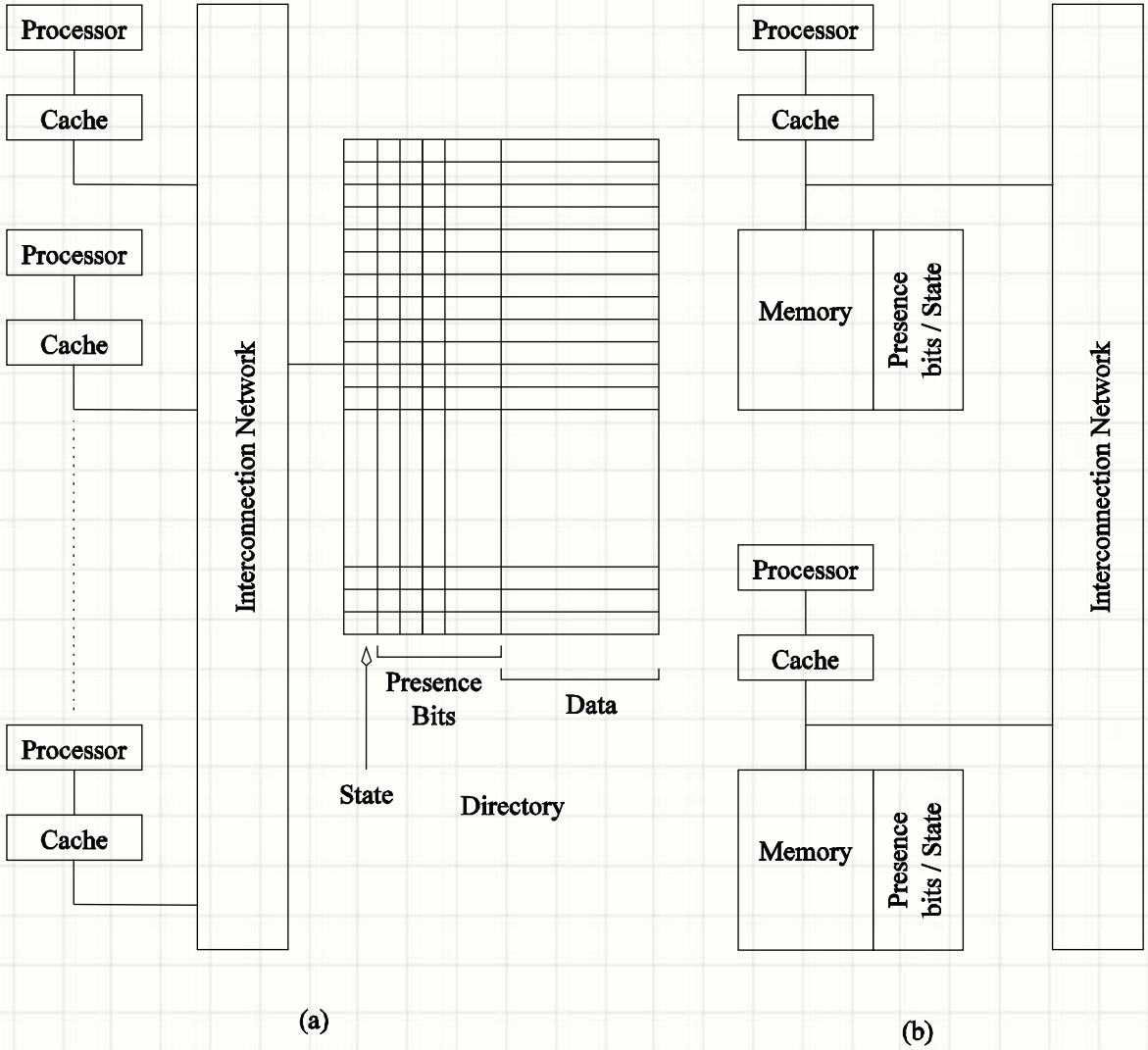
Protocolos de Cache Snoopy

- Cópias de dados marcadas como sujas → Operações subsequentes podem ser executadas localmente no cache
  - Sem tráfego externo
- Se item de dados lido por um número de processadores → Transição para o estado compartilhado na cache e todas a subsequentes operações de leitura tornam-se locais
- Se um processador atualiza dados, ele gera requisições de coerência no barramento

Protocolos Baseados em Diretórios

- Nos protocolos snoopy, cada operação de coerência é enviada para todos os processadores
  - Limitação – aumento da comunicação no barramento
- Por que não enviar requisições de coerência somente para os processadores que precisam ser notificados?
  - Uso de um diretório centralizado ou vários diretórios distribuídos para a manutenção das informações de coerência de cache

Protocolos Baseados em Diretórios



Arquitetura típica de um sistema baseado em diretórios
(a) diretório centralizado, (b) diretório distribuído.

Protocolos Baseados em Diretórios

- Os bits adicionais para armazenar o diretório pode aumentar o overhead
- A rede básica deve ser capaz de realizar todos os pedidos de coerência
- O diretório centralizado é um ponto de contenção

EXERCÍCIO E LEITURA RECOMENDADA

Exercício

- Acessar o Moodle

Leitura Recomendada

- Introduction to Parallel Computing, Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar - 2ª ed., Addison Wesley

Bibliografia

- Introduction to Parallel Computing, Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar - 2ª ed., Addison Wesley
- Introduction to Parallel Computing
  - https://computing.llnl.gov/tutorials/parallel_comp/

Dúvidas

Próxima Aula...

- Modelos de Programação Paralela
- Terminologia Geral de Programação Paralela