Revisão

Introdução

Modelos de Programação

- Modelos + comuns

- Tópicos clássicos em memória compartilhada

- Tópicos clássicos em passagem de mensagens

- Tópicos clássicos em compiladores //izadores

Hoje

Máquinas/Arquiteturas //s

- Pipelining, super-escalares e VLIW
- SIMD e máquinas vetoriais
- MIMD: multicomputadores e multiprocessadores

MIMD: Multiprocessadores e Multicomputadores

- Redes de interconexão
- Classificação
- Memória compartilhada
  - Implementação
  - Coerência de caches
  - Modelos de consistência
  - Software DSMs
- Exemplos

Redes de Interconexão

Conceitos básicos

- Latência e banda passante
- Grau e diâmetro
- Bisection width
- Redes estáticas/dinâmicas

Redes de Interconexão

Padrões de comunicação

- Broadcast/multicast
- Perfect shuffle
- Comunicação hipercúbica

Padrões Shuffle e Hipercúbico

Figuras

Redes Estáticas

Figuras

Árvores e K-ary N-cubes

Figuras

Redes Estáticas

Tabela

Redes Dinâmicas

Figuras

Redes Multi-Estágios

Figuras

Redes Dinâmicas

Tabela

Controle e Roteamento

Conceitos básicos

- Redes com único/múltiplos estágios
- Rede síncrona/assíncrona
- Redes bloqueantes/não-bloqueantes
- Controle centralizado/distribuído
- Chaveamento por circuito/pacote
- Roteamento determinístico/adaptativo

Roteamento em MINs

Figura roteamento em Omega network

Roteamento em Redes Estáticas

Store-and-forward

Virtual cut-through

Wormhole routing

Figuras

Roteamento em K-ary N-cubes

Dimension-ordered routing

Figura

Classificação de Máquinas MIMD

UMA (Uniform Memory Access)

NUMA (Non-Uniform Memory Access)

- CC-NUMA
- NCC-NUMA
- Network NUMA
- COMA (Cache Only Memory Architecture)

NORMA (NO Remote Memory Access)

Memória Compartilhada

Estratégias de implementação:

- Memória centralizada (UMA)
- Memória compartilhada distribuída (NUMA)
- Memória compartilhada distribuída em software (software DSM)
- UMA como nó de sistema NUMA ou sistema software DSM

Configurações Típicas

Figura UMA e NUMA/NORMA

Coerência de Dados

Coerência dos dados compartilhados é o problema principal !

Conceitos básicos

- Causas de inconsistência: compartilhamento, migração e E/S

- Coerência de blocos de cache

- Implementação em software ou hardware

Causas de Inconsistência em Caches

Causas de Inconsistência em Caches

Coerência em Software

Opções:

- Evitar caching de dados compartilhados
- Limpar a cache em operações de sincronização, E/S ou migração
- Usuário, compilador, biblioteca ou sistema operacional responsável por manter coerência

Coerência em Hardware

- Gerenciamento de estados e controle de coerência em hardware

- Tipos de protocolos: invalidação e atualização

- Formas de manter estado: "snooping" e diretórios

Invalidações e Atualizações com "Snooping"

Exemplo de Protocolo

Protocolo de invalidação para máquina UMA com caches write-back

Protocolos Baseados em Diretórios

Diretórios controlam estado dos blocos

Cada diretório responsável pela memória associada

Figura

Modelos de Consistência de Memória

Consistência define quando uma escrita é "vista" por outros processadores

Programadores estão acostumados a *consistência seqüencial*

```
   P1                    P2
A = 0;                B = 0;
   .                     .
   .                     .
A = 1;                B = 1;
if (B == 0)           if (A == 0)
   SC                    SC
```

Dois testes não podem ser verdadeiros

Problema: ineficiente

Consistência Relaxada

Memória não precisa estar consistente sempre

Consistência relaxada permite sobrepor comunicação e computação

Sincronização tem que ser explícita

Consistência Relaxada

Figura

Consistência Relaxada

Seqüencial: cada acesso completado antes do início do próximo

Processador: escritas em ordem, leituras podem ultrapassar

Fraca: acessos fora de ordem, sincronização seqüencial

Liberação: acessos fora de ordem, liberação seqüencial

Software DSMs

Hardware não permite acessos remotos

É necessario que o software:

- Evite acessos a dados desatualizados
- Consiga dados atualizados quando necessário
- Detecte escritas em dados compartilhados

Essas funções sao transparentes ao software (aplicação, compilador, runtime system, ou mesmo o sistema operacional) em sistemas com coerência em hardware

Software DSMs

Em geral, hardware de memória virtual controla caching e coerência

Características fundamentais:

- Grandes unidades de caching e coerência
- Tratamento de falso compartilhamento
- Consistência relaxada de memória

Sistemas Paralelos Atuais

- Shrimp, AURC (Princeton)
- CM 5 (Thinking Machines)
- Origin e T3E (Silicon Graphics)
- Dash e Flash (Stanford)
- Alewife e Fugu (MIT)
- TreadMarks (Rice)
- NCP2 (COPPE/UFRJ)

Estudos de Caso

DASH

- SMPs (4 procs cada) ligados por malha
- Coerência de caches em hardware
- Protocolo de invalidação com diretório por mapa completo
- Consistência por liberação
- Speedups entre 5 e 26 em 32 procs (Splash)

Estudos de Caso

Alewife

- Um processador por nó
- Coerência de caches em hardware + software
- Protocolo de invalidação com diretório limitado
- Consistência seqüencial
- Speedups entre 12 e 24 em 32 procs (Splash)

Estudos de Caso

NCP2

- Um processador por nó
- Coerência de memoria em software + hardware auxiliar (CP)
- TreadMarks modificado (invalidação)
- Consistência por liberação preguiçosa
- Primeiro protótipo em desenvolvimento

Próxima Aula

- Stenstrom, "A Survey of Cache Coherence Schemes for Multiprocessors", IEEE Computer, vol 23, no 6, Junho 1990

- Protic, Tomasevic, and Milutinovic, "Distributed Shared Memory: Concepts and Systems", IEEE Parallel and Distributed Technology, vol 4, no 2, Summer 1996

Discussão em aula

Aluno responsável por esclarecer dúvidas