# Arquiteturas Paralelas: máquinas MIMD com memória distribuída

Paulo Sérgio Lopes de Souza
*pssouza@icmc.usp.br*

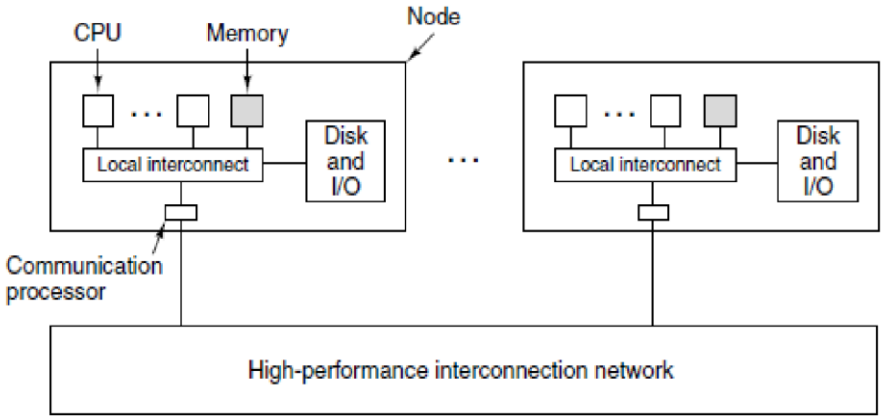
**Universidade de São Paulo / ICMC / SSC – São Carlos**
**Laboratório de Sistemas Distribuídos e Programação Concorrente**

# MIMD com Memória Distribuída - Multicomputadores

- Máquinas MIMD com memória distribuída
  - Diferentes computadores (ou nós)
  - Interconectados por uma rede de conexão
    - Oferece suporta a interação entre processos
  - Nós são unidades independentes e **assíncronas**
    - Têm CPU, memória local e E/S
      - Múltiplas instruções executam assincronamente sobre dados diferentes.
  - Espaços de endereçamento são locais aos processos
    - Processos não acessam memórias de outros processos
    - Interação por primitivas *send* e *receive*
  - Modelo de programação considera
    - **Modelo de memória por troca de mensagens**
  - Exemplos de arquiteturas
    - *Clusters*, MPPs (*Massively Parallel Processors*) e *Grids*

# *Clusters*

- Nós formados usualmente por CPUs *multicore*,
  - Núcleos em um nó compartilham memória e E/S (discos, rede, ou outros)
  - ***Threads* que executam em um mesmo nó podem compartilhar memória**
    - Neste caso, no nó, o modelo de memória é compartilhado
  - Combinação dos modelos de memória formam um modelo híbrido
    - Permite explorar a versatilidade das máquinas MIMD
- Características (qualidade) da rede de interconexão
  - Afetam granulação do paralelismo (normalmente é mais alta em clusters)
  - Replicações de redes fornecem redundância e maior vazão
- Projeto do algoritmo paralelo deve ter como foco um *trade-off*:
  - Maximizar o grau de concorrência e
  - Minimizar sobrecarga de comunicação




Tanenbaum (2013)

Figure 8-36. A generic multicomputer.

# *Clusters*

- Benefícios (Stallings, 2013):
  - **Escalabilidade absoluta**:
    - há possibilidade de máquinas com vários nós
  - **Escalabilidade incremental**:
    - novos nós podem ser adicionados

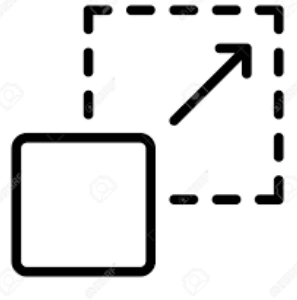

  - Usam sistemas operacionais e outras ferramentas de software **disponíveis para máquinas SISD**

# *Clusters*

- Questões de projeto dos Sistemas Operacionais de Clusters
  - Gerenciamento de falhas
    - **Clusters para alta disponibilidade**
      - Oferece alta probabilidade que todos os recursos estejam funcionando
      - Caso haja falha, novas requisições tratadas por outro nó
    - **Clusters com tolerância a falhas**
      - Garante funcionamento mesmo na presença de falhas


- Redundância garante retorno de requisições não retornadas
  - *Failover*: recuperação de falhas para outro dispositivo que esteja apto
  - *Failback*: retorno ao dispositivo original quando este voltar a funcionar

- Escalonamento: é essencial à escalabilidade
  - Políticas e mecanismos de escalonamento distribuem carga de trabalho
    - Balanceamento da carga de trabalho é um objetivo usual

# *Clusters*

- Usos da computação paralela sobre máquinas MIMD com memória distribuída
  - Computação paramétrica
    - Algoritmo sequencial executado em paralelo com parâmetros diferentes
  - Compilação para geração automática de paralelismo
    - Compilador gera porções que podem ser executadas em paralelo
  - Aplicações paralelas
    - Programador desenvolve a aplicação para executar na máquina MIMD

# *Clusters*

- Sistemas básicos para clusters
  - Oferecem suporte ao desenvolvimento e execução de aplicações concorrentes com passagem de mensagens
  - Sistemas operacionais gerenciam cada nó

    - **Linux** é um exemplo
    - **Permitem interação entre processos locais ou remotos**
      - **preste atenção na diferença dos modelos de memória**
  - Compiladores tradicionais são usuais: C, Fortran, Java
    - Permitem o uso de linguagens conhecidas
    - Bibliotecas possibilitam a geração de processos, comunicação e sincronização dos processos.
  - **Middlewares oferecem recursos extra** de software/hardware para:
    - Ponto de entrada único para processos
      - gerenciamento de submissões (escalonamento e balanceamento da carga), espaço único de processos e migração de processos
    - Interface de usuário padronizada, controle único e pontos de verificação (recuperação de falhas)
    - Espaço único para E/S, hierarquia de sistemas de arquivos e rede virtual única

# *Clusters*

- Sistemas básicos para clusters:
  - uma visão gráfica, hierárquica e abstrata da gerência do cluster

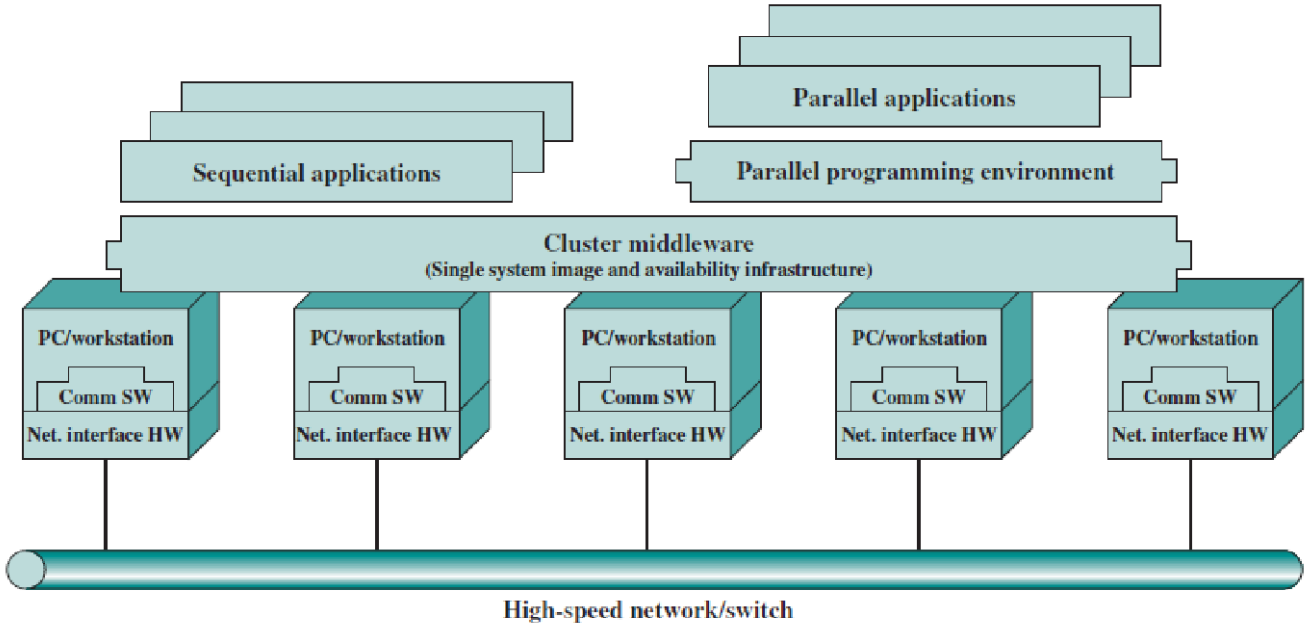
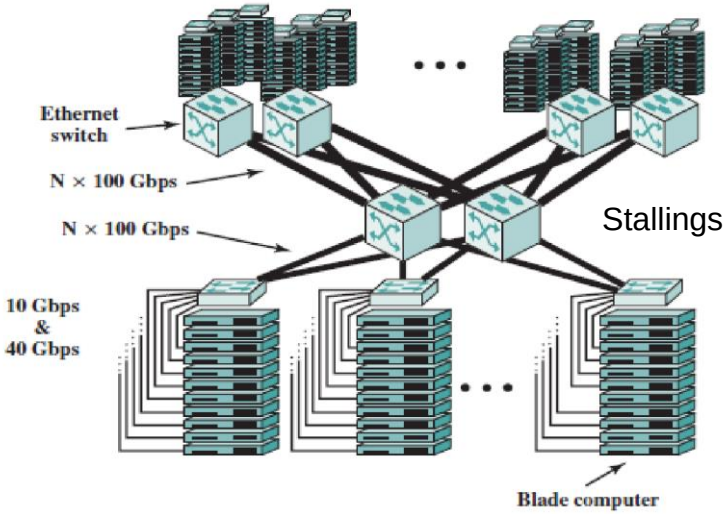



Figure 17.10 Cluster Computer Architecture [BUYY99a]

Stallings (2013)

# *Clusters*

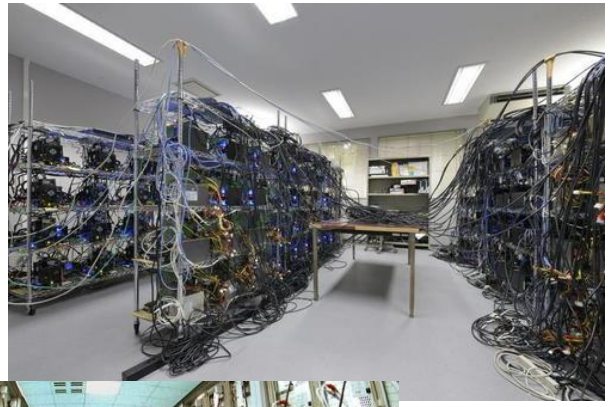
- **Usualmente estão hospedados em racks...**




Stallings (2013)

**Figure 17.11** Example 100-Gbps Ethernet Configuration for Massive Blade Server Site

... mas não precisam estar...



# Clusters

- Exemplo dos Clusters usados pela Google (Tanenbaum, 2013)
  - Indexar bilhões de páginas e pesquisar a base o mais rápido possível
    - Tratar requisições concorrentes dos usuários 24/7
  - Opção por máquinas simples
    - Grande redundância de hardware, inclusive dos links ópticos
  - **Clusters projetados para tolerância a falhas**
    - Hardwares que param de funcionar
    - Bugs de software
    - Falta de energia
    - Problemas externos
      - terremotos, ataques, ...
  - **Replicação para tratar falhas**
    - Previsão de falhas no projeto
    - Manter alta vazão e disponibilidade
  - Otimização da relação custo / benefício

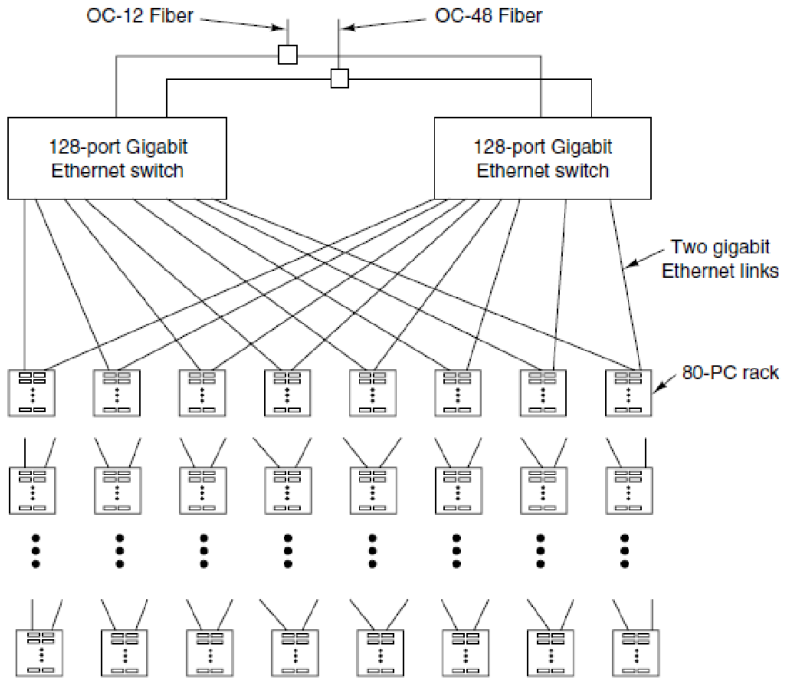



Figure 8-44. A typical Google cluster.

Tanenbaum (2013)

# *Massively Parallel Processors* (MPP)

- **MPP são supercomputadores MIMD com memória distribuída** como clusters
  - Mas orçados na faixa de milhões de dólares (Tanenbaum, 2013)
  - Usados para problemas que requerem maior desempenho

- Normalmente usam tecnologias escalares/vetoriais já conhecidas e redes proprietárias de alto desempenho (mais eficiência).
- Exemplo: BlueGene (IBM)

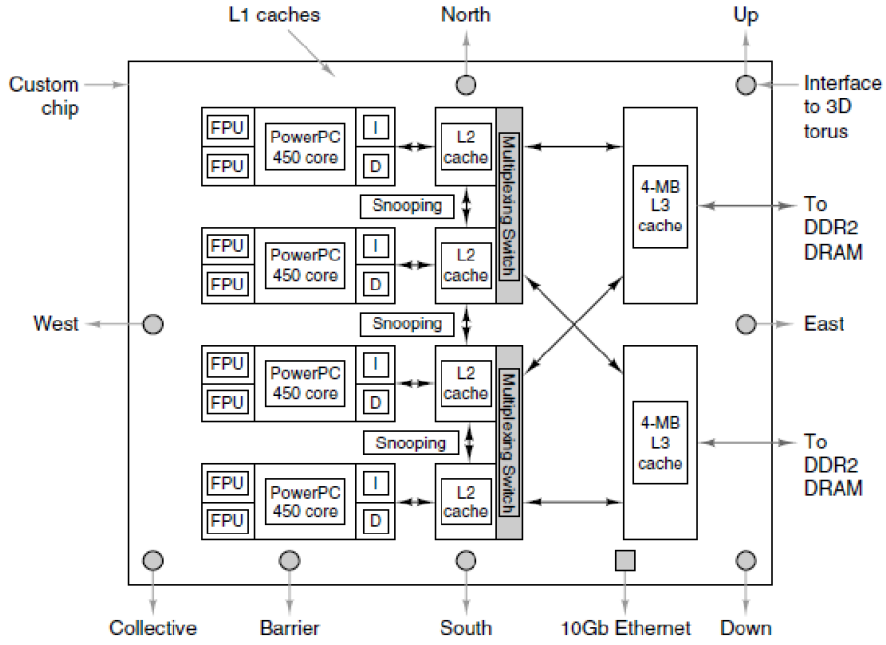



Figure 8-38. The BlueGene/P custom processor chip.

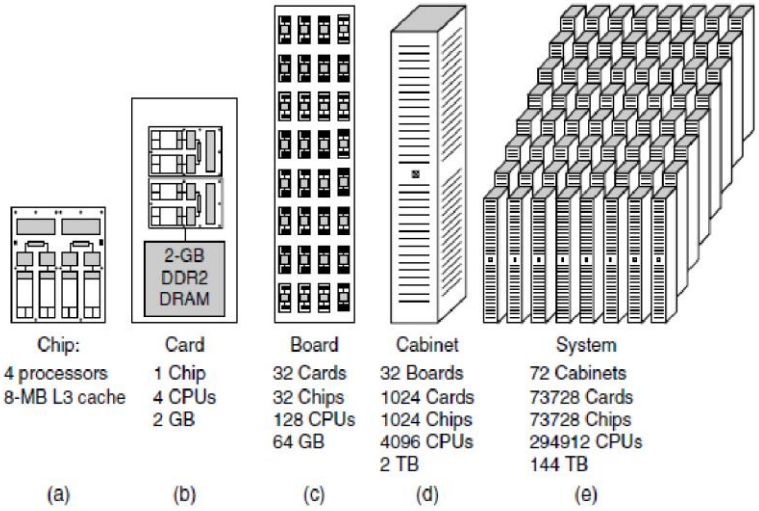


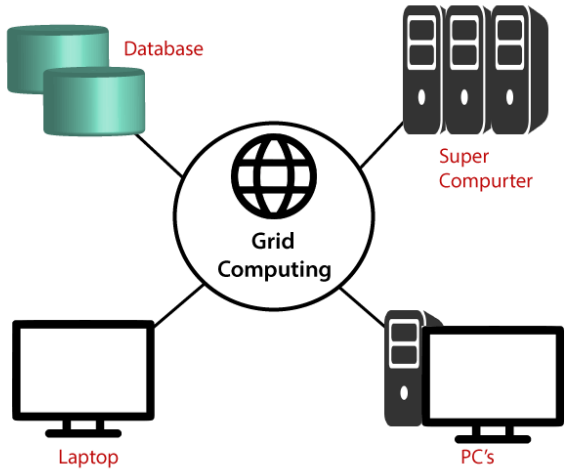

Figure 8-39. The BlueGene/P: (a) chip. (b) card. (c) board. (d) cabinet. (e) system.

Tanenbaum (2013)

# Grades (*Grids*)

- Desafios atuais têm uma escala grande e são interdisciplinares
  - Ciência, engenharias, indústria, meio ambiente, ...
    - Missões à Marte
    - Consórcios para construção aviões, barragens, ...
    - Equipes internacionais trabalhando em novas vacinas
- Necessitam que organizações individuais com seus recursos trabalhem juntas
  - Soma de esforços, recursos e dados para atingirem um objetivo comum

- **Grids representam sistemas e tecnologias usadas para conectar computadores isolados geograficamente**
  - Grid é usualmente um cluster heterogêneo, fracamente acoplado e internacional
- Fornece uma infraestrutura técnica para que organizações compartilhem um objetivo comum, formando uma organização virtual
  - Flexível, com muitos membros que mudam
  - Dá suporte trabalho colaborativo de diferentes áreas
  - Permite controle sobre os próprios recursos




https://www.tutorialandexample.com/grid-computing-vs-cloud-computing/

# Grades (*Grids*)

- Grids têm características multilaterais que consideram seus pares
  - Diferente de modelos já consolidados como cliente-servidor ou *peer-to-peer*
- Por isso, o desenvolvimento de Grids envolve
  - Novos serviços, ferramentas e protocolos para habilitar o funcionamento dessas organizações virtuais
- Grids precisam de acesso a diferentes recursos
  - Pertencem a organizações distintas com infraestruturas diferentes
    - Organizações decidem o que compartilhar, por quanto tempo e para quem
- No fim, Grids focam em gerir o acesso a recursos compartilhados
  - A gerência pode ser decomposta em camadas

| Layer | Function |
|---|---|
| Application | Applications that share managed resources in controlled ways |
| Collective | Discovery, brokering, monitoring and control of resource groups |
| Resource | Secure, managed access to individual resources |
| Fabric | Physical resources: computers, storage, networks, sensors, programs and data |

**Figure 8-52.** The grid layers.

Tanenbaum (2013)

# Grades (*Grids*)

- Segurança é um aspecto chave para o sucesso dos Grids
  - Credenciais têm um peso importante na segurança
    - Autenticação de credenciais, assinaturas digitais, ...

- Interoperabilidade entre diferentes organizações segue padrões
  - *Global Grid Forum* gerencia o processo de padronização dos sistemas
  - Framework OGSA (*Open Grid Services Architecture*)
    - Posiciona vários padrões em desenvolvimento
  - Padrões já consolidados também são usados, como o
    - WSDL (*Web Services Definition Language*) descrevem serviços OGSA

  - Diferentes categorias de serviços são padronizados:
    - Infraestrutura (comunicação entre recursos)
    - Gerenciamento de recursos (reserva e liberação de recursos)
    - Dados (movimentação e replicação de dados)
    - Contexto (descrição de recursos e políticas de uso)
    - Informação (disponibilidade de recursos)
    - Auto gerenciamento (suporte a uma qualidade de serviço declarada)
    - Segurança (garantir políticas de segurança)
    - Gerenciamento de execução (gerenciar o fluxo de trabalho)

# Referências

Stallings, W.; Computer Organization and Architecture: Designing for Performance. Ninth Edition. Pearson. 2013.

Tanenbaum, A. S.; Austin, T.; Structured Computer Organization. Sixth Edition. Pearson. 2013.

Patterson, D. A.; Hennessy, J. L.; Computer Organization and Design: the hardware / software interface. Fith Edition. Elsevier, 2014.

Rauber, T.; Rünger, G.; Prallel Programming for Multicore and Cluster Systems. Second Edition. Springer. 2013.

# Material Complementar

Material sobre Consistência de Memória e Coerência de Cache

https://www.cs.utexas.edu/~bornholt/post/memory-models.html

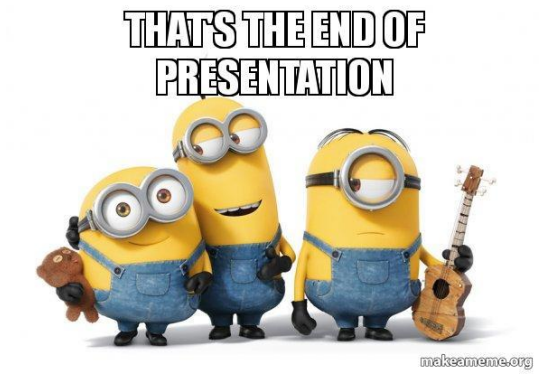
https://scs.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=694972dd-dac1-4635-8939-b6abe3f99749

https://www.cs.colostate.edu/~cs551/CourseNotes/Consistency/TypesConsistency.html

# Arquiteturas Paralelas: máquinas MIMD com memória distribuída

Paulo Sérgio Lopes de Souza
*pssouza@icmc.usp.br*

**Universidade de São Paulo / ICMC / SSC – São Carlos**
**Laboratório de Sistemas Distribuídos e Programação Concorrente**




CNPq Conselho Nacional de Desenvolvimento Científico e Tecnológico
CAPES
FAPESP