1

Fundamentos de Processamento de Alto Desempenho

César A. F. De Rose (*PUCRS – derose@inf.pucrs.br*)[1]
Philippe O. A. Navaux (*UFRGS – navaux@inf.ufrgs.br*)[2]

Resumo:

Neste trabalho, apresentam-se os conceitos básicos de arquiteturas paralelas, incluindo as suas diferentes classificações e as principais tendências para a sua construção. Um destaque especial é dado as máquinas agregadas (*clusters*) por se tratar de uma arquitetura cada vez mais acessível tanto em empresas como em universidades e centros de pesquisa.

[1] Doutor em Ciência da Computação pela Universidade de Karlsruhe, Alemanha (1998) nas áreas de Sistemas Operacionais e Processamento Paralelo. Mestre em Ciência da Computação pela UFRGS (1993) nas áreas de Arquitetura de Computadores e Processamento Paralelo e Bacharel em Informática pela PUCRS (1990). Professor do Programa de Pós-Graduação em Ciência da Computação da Pontifícia Universidade Católica do Rio Grande do Sul, atuando nas disciplinas de Arquitetura de Computadores, Sistemas Operacionais e Processamento Paralelo, nos cursos de Graduação e Pós-Graduação. Coordenador do Centro de Pesquisa em Alto Desempenho (CPAD – PUCRS/HP, www.cpad.pucrs.br) Áreas de Interesse: Processamento de Alto Desempenho e Arquiteturas Paralelas.

[2] Doutor em Informática pelo INPG, Universidade de Grenoble, França (1979) na área de Arquiteturas de Computadores. Mestre em Física Aplicada no Instituto de Física da UFRGS (1973) e Engenheiro Eletrônico pela Escola de Engenharia da UFRGS (1970). Professor do Programa de Pós-Graduação em Computação do Instituto de Informática da UFRGS, atuando na área de Arquitetura de Computadores e Processamento de Alto Desempenho, orientador de doutorado e mestrado desde o inicio dos cursos. Áreas de Interesse: Processamento de Alto Desempenho e Arquiteturas de Computadores.

1.1. Introdução

Máquinas paralelas vem tornando-se mais populares em função da demanda sempre crescente por poder computacional. Infelizmente, os sistemas que oferecem a capacidade de processamento para satisfazer esta demanda, muitas vezes, ainda têm um custo muito elevado e/ou são difíceis de programar. O estudo de arquiteturas paralelas contribui para o entendimento e busca de alternativas para os dois problemas.

No caso do custo, um melhor entendimento das alternativas de construção de máquinas paralelas, e a compreensão de como estas decisões de projeto repercutirão no desempenho final da máquina, podem possibilitar a escolha de uma arquitetura de menor custo que ainda obtenha o desempenho desejado.

Como a programação de aplicações paralelas ainda exige o conhecimento de características específicas da máquina para a obtenção de desempenho, sólidos conhecimentos sobre a arquitetura de máquinas paralelas auxiliam em todo o ciclo de desenvolvimento de programas paralelos, desde sua modelagem até a fase de depuração e otimização.

1.2. Classificações de Arquiteturas Paralelas

1.2.1. Classificação de Flynn

Para uma classificação inicial de arquiteturas paralelas pode ser usada a classificação genérica de Flynn [FLY72]. Apesar de ter sua origem em meados dos anos 70, é ainda válida e muito difundida. Baseando-se no fato de um computador executar uma seqüência de instruções sobre uma seqüência de dados, diferencia-se o fluxo de instruções (*instruction stream*) e o fluxo de dados (*data stream*). Dependendo de esses fluxos serem múltiplos ou não, e através da combinação das possibilidades, Flynn propôs quatro classes (Tabela 1-1):

Tabela 1-1: Classificação de Flynn segundo o fluxo de instruções e o fluxo de dados

	SD (*Single Data*)	**MD** (*Multiple Data*)
SI (*Single Instruction*)	SISD Máquinas von Neumann convencionais	SIMD Máquinas *Array* (CM-2, MasPar)
MI (*Multiple Instruction*)	MISD Sem representante (até agora)	MIMD Multiprocessadores e multicomputadores (nCUBE, Intel Paragon, Cray T3D)

Na classe SISD (*Single Instruction Single Data*), um único fluxo de instruções atua sobre um único fluxo de dados. Na Figura 1-1, o fluxo de instruções (linha contínua) alimenta uma unidade de controle (C) que ativa a unidade central de processamento (P). A unidade P, por sua vez, atua sobre um único fluxo de dados (linha tracejada), que é lido, processado e reescrito na memória (M). Nessa classe, são enquadradas as máquinas von Neumann tradicionais com apenas um processador, como microcomputadores pessoais e estações de trabalho.

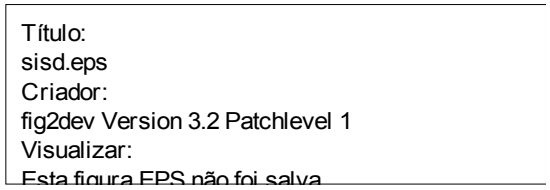

Figura 1-1: Diagrama da classe SISD

A classe MISD (*Multiple Instruction Single Data*) é bastante interessante. Nesse caso, múltiplos fluxos de instruções atuariam sobre um único fluxo de dados. A Figura 1-2 mostra múltiplas unidades de processamento P, cada uma com sua unidade de controle própria C, recebendo um fluxo diferente de instruções. Essas unidades de processamento executam suas diferentes instruções sobre o mesmo fluxo de dados. Na prática, diferentes instruções operam a mesma posição de memória ao mesmo tempo, executando instruções diferentes. Como isso, até os dias de hoje, não faz qualquer sentido, além de ser tecnicamente impraticável, essa classe é considerada vazia [HOC88][ALM89].

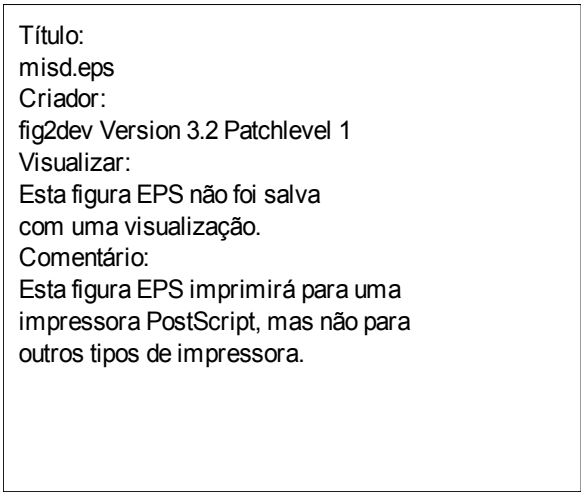

Figura 1-2: Diagrama da classe MISD

As máquinas paralelas concentram-se nas duas classes restantes, SIMD e MIMD. No caso SIMD *(Single Instruction Multiple Data),* uma única instrução é executada ao mesmo tempo sobre múltiplos dados. O processamento é controlado por uma única unidade de controle C, alimentada por um único fluxo de instruções. A mesma instrução é enviada para os diversos processadores P envolvidos na execução. Todos os processadores executam suas instruções em paralelo de forma síncrona sobre diferentes fluxos de dados (Figura 1-3). Na prática, pode-se dizer que o mesmo programa está sendo executado sobre diferentes dados, o que faz com que o princípio de execução

SIMD assemelhe-se bastante ao paradigma de execução seqüencial. É importante ressaltar que, para que o processamento das diferentes posições de memória possa ocorrer em paralelo, a unidade de memória M não pode ser implementada como um único módulo de memória, o que permitiria só uma operação por vez. Nessa classe, são enquadradas as máquinas *Array* como CM-2 [HWA93] e MasPar [HWA93].

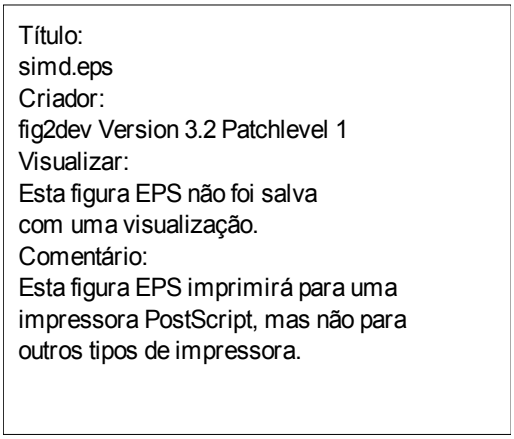

Figura 1-3: Diagrama da classe SIMD

Enquanto, em uma máquina SIMD, só um fluxo de instruções, ou seja, só um programa, está sendo executado, em uma máquina MIMD (*Multiple Instruction Multiple Data*), cada unidade de controle C recebe um fluxo de instruções próprio e repassa-o para sua unidade processadora P para que seja executado sobre um fluxo de instruções próprio (Figura 1-4). Dessa forma, cada processador executa o seu próprio programa sobre seus próprios dados de forma assíncrona. Sendo assim, o princípio MIMD é bastante genérico, pois qualquer grupo de máquinas, se analisado como uma unidade (executando, por exemplo, um sistema distribuído), pode ser considerado uma máquina MIMD. Nesse caso, como na classe SIMD, para que o processamento das diferentes posições de memória possa ocorrer em paralelo, a unidade de memória M não pode ser implementada como um único módulo de memória, o que permitiria só uma operação por vez. Nessa classe, enquadram-se servidores com múltiplos processadores (*dual*, *quad*), as redes de estações e máquinas como CM-5 [HWA93], nCUBE [CUL99], Intel Paragon [CUL99] e Cray T3D [CUL99].

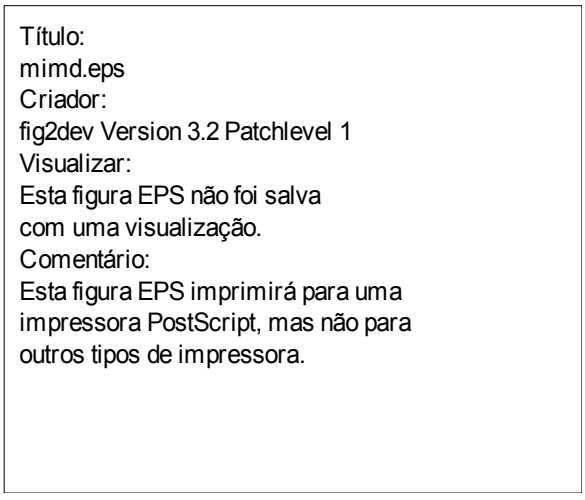

Figura 1-4: Diagrama da classe MIMD

1.2.2. Classificação segundo o compartilhamento de memória

Um outro critério para a classificação de máquinas paralelas é o compartilhamento da memória. Quando se fala em **memória compartilhada** (*shared memory*), existe um único espaço de endereçamento que será usado de forma implícita para comunicação entre processadores, com operações de `load` e `store`. Quando a memória não é compartilhada, existem **múltiplos espaços de endereçamento privados** (*multiple private address spaces*), um para cada processador. Isso implica comunicação explícita através de troca de mensagens com operações `send` e `receive`.

Memória distribuída (*distributed memory*), por sua vez, refere-se à localização física da memória. Se a memória é implementada com vários módulos, e cada módulo foi colocado próximo de um processador, então a memória é considerada distribuída. Outra alternativa é o uso de uma **memória centralizada** (*centralized memory*), ou seja, a memória encontra-se à mesma distância de todos os processadores, independentemente de ter sido implementada com um ou vários módulos [PAT94]. Dependendo de uma máquina paralela utilizar-se ou não de uma memória compartilhada por todos os processadores, pode-se diferenciar:

- **Multiprocessadores**

  Todos os processadores *P* acessam, através de uma rede de interconexão, uma memória compartilhada *M*. Esse tipo de máquina possui apenas um espaço de endereçamento, de forma que todos os processadores *P* são capazes de endereçar todas as memórias *M* (Figura 1-5). A comunicação entre processos é feita através da memória compartilhada de forma bastante eficiente com operações do tipo `load` e `store`. Essas características resultam do fato de esse tipo de máquina paralela ser construída a partir da replicação apenas do componente processador de uma arquitetura convencional (destacados com uma moldura mais escura na Figura 1-5). Daí o nome **múltiplos processadores**.

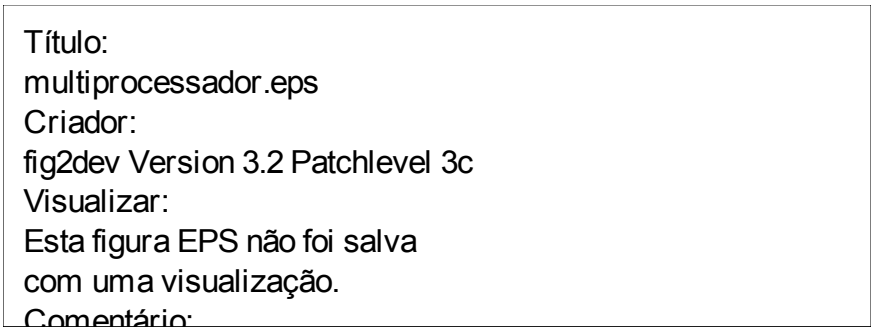

Figura 1-5: Arquitetura de um multiprocessador

Em relação ao tipo de acesso às memórias do sistema, multiprocessadores podem ser classificados como [HWA93]:

- acesso uniforme à memória (*uniform memory access*, UMA)

  A memória usada nessas máquinas é centralizada e encontra-se à mesma distância de todos os processadores (Figura 1-6), fazendo com

que a latência de acesso à memória seja igual para todos os processadores do sistema (uniforme). Como o barramento é a rede de interconexão mais usada nessas máquinas e suporta apenas uma transação por vez, a memória principal é normalmente implementada com um único bloco. É importante ressaltar que máquinas com outras redes de interconexão e com memórias entrelaçadas (implementadas com múltiplos módulos e, dessa forma, permitindo acesso paralelo a diferentes módulos) também se enquadram nessa categoria se mantiverem o tempo de acesso à memória uniforme para todos os processadores do sistema.

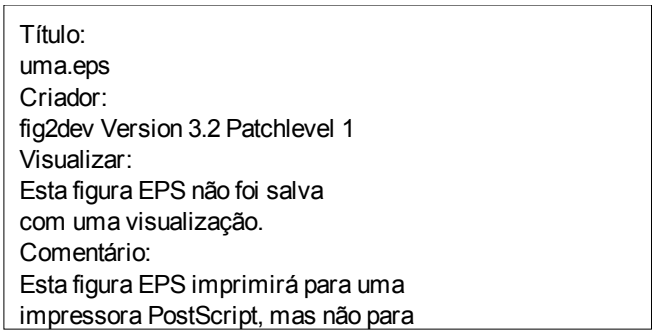

Figura 1-6: Máquina UMA

Muitas dessas máquinas utilizam-se de memórias *cache* para amenizar a diferença de velocidade entre processador e memória principal [PAT94]. A memória *cache* é uma memória especial mais rápida que a memória principal e funciona como uma área intermediária de armazenamento. Para cada consulta feita à memória principal, é mantida uma cópia na *cache*, com o objetivo de acelerar um novo acesso ao mesmo endereço. Como, nessa classe de máquinas, todos os processadores podem endereçar toda a memória do sistema, em um determinado momento, várias cópias da mesma posição da memória principal podem existir em *caches* diferentes. Isso é um problema que precisa ser tratado, pois um processador pode trabalhar com sua cópia local, e essa pode não refletir mais o estado atual da mesma posição na memória principal. Como o ideal é que o conteúdo das memórias *caches* seja **coerente**, esse problema é chamado de coerência de *cache* (*cache coherence*) [CUL99]. A maioria dos multiprocessadores UMA resolve esse problema em hardware.

- acesso não uniforme à memória (*non-uniform memory access*, NUMA)

A memória usada nessas máquinas é distribuída, implementada com múltiplos módulos que são associados um a cada processador (Figura 1-7). O espaço de endereçamento é único, e cada processador pode endereçar toda a memória do sistema. Se o endereço gerado pelo processador encontrar-se no módulo de memória diretamente ligado a ele, dito local, o tempo de acesso será menor que a um módulo que está diretamente ligado a outro processador, dito remoto, que só pode ser acessado através da rede de interconexão. Por esse motivo, essas máquinas possuem um acesso não uniforme à memória (a distância à memória não é sempre a mesma e depende do endereço desejado).

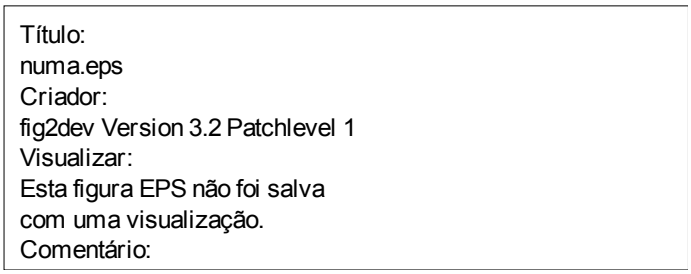

Figura 1-7: Máquina NUMA

Dependendo de o problema da coerência de *cache* ter sido tratado ou não, e de esse tratamento ter sido feito em hardware ou em software, essa classe pode ser subdividida em:

- acesso não uniforme à memória sem coerência de *cache* (*non-cache-coherent non-uniform memory access*, NCC-NUMA)

  Variação de NUMA em que não há coerência de *cache* garantida em hardware.

- acesso não uniforme à memória com coerência de *cache* (*cache-coherent non-uniform memory access*, CC-NUMA)

  Variação de NUMA em que há coerência de *cache* garantida em hardware.

- acesso não uniforme à memória com coerência de *cache* em software (*software-coherent non-uniform memory access*, SC-NUMA)

  Nesse caso, a coerência de *cache* não está implementada em hardware como nas máquinas CC-NUMA, mas em software, de forma transparente ao usuário. Essa camada de software é também conhecida como DSM (*Distributed Shared Memory*) [NIT91] e naturalmente só faz sentido sobre máquinas NCC-NUMA e NORMA que não têm coerência de *cache* em hardware.
  Uma DSM depende de extensões de software para a obtenção de um espaço de endereçamento único, compartilhamento de dados e controle de coerência. Uma alternativa de implementação é chamada SVM (*Shared Virtual Memory*), e nela o mecanismo de gerenciamento de memória de um sistema operacional tradicional é modificado para suportar esses serviços em nível de páginas ou de segmentos. A vantagem é que, dessa forma, não se fazem necessárias alterações nas aplicações. O projeto SHRIMP [LIA98] usa essa abordagem.
  Outra possibilidade é não alterar o sistema operacional e usar compiladores e bibliotecas de funções para converter código desenvolvido para um espaço de endereçamento único em um código que roda em múltiplos espaços de endereçamento. Nesse caso, o código original tem que ser modificado para incluir compartilhamento de dados, sincronização e primitivas de coerência. O projeto TreadMarks [AMZ96] usa essa abordagem.

– arquiteturas de memória somente com *cache* (*cache-only memory architecture*, COMA)

Em uma máquina COMA, todas as memórias locais estão estruturadas como memórias *cache* e são chamadas de COMA *caches* [HWA98] (Figura 1-8). Essas *caches* têm muito mais capacidade que uma *cache* tradicional. Arquiteturas COMA são as únicas que têm suporte de hardware para a replicação efetiva do mesmo bloco de *cache* em múltiplos nós (nas arquiteturas anteriores, os blocos são invalidados, e não atualizados). A memória principal dessas máquinas é composta pelas *caches* COMA, e o hardware de suporte tem que integrar a gerência das *caches* e a gerência de memória. Essa complexidade faz com que essas arquiteturas sejam mais caras de implementar que as máquinas NUMA.

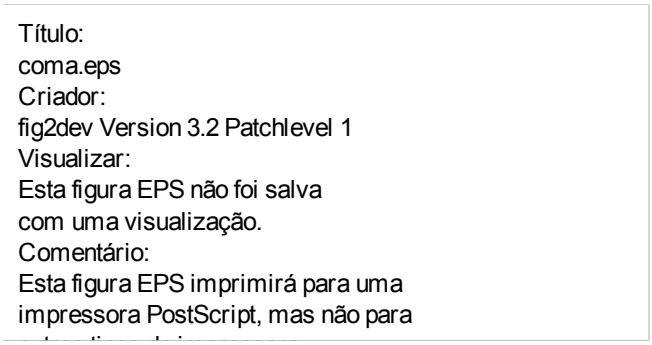

Figura 1-8: Máquina COMA

- **Multicomputadores**

Cada processador *P* possui uma memória local *M*, à qual só ele tem acesso. As memórias dos outros processadores são consideradas memórias remotas e possuem espaços de endereçamento distintos (um endereço gerado por P_1 só é capaz de endereçar M_1). Como não é possível o uso de variáveis compartilhadas nesse ambiente, a troca de informações com outros processos é feita por envio de mensagens pela rede de interconexão (Figura 1-9). Por essa razão, essas máquinas também são chamadas de sistemas de troca de mensagens (*message passing systems*). Essas características resultam do fato de esse tipo de máquina paralela ser construído a partir da replicação de toda a arquitetura convencional, e não apenas do componente processador como nos multiprocessadores (destacados com uma moldura mais escura na Figura 1-9). Daí o nome **múltiplos computadores**.

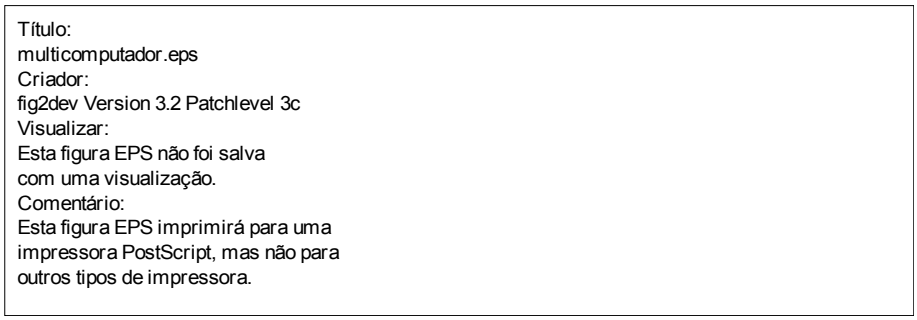

Figura 1-9: Arquitetura de um multicomputador

Em relação ao tipo de acesso às memórias do sistema, multicomputadores podem ser classificados como:

- sem acesso a variáveis remotas (*non-remote memory access*, NORMA)

  Como uma arquitetura tradicional inteira foi replicada na construção dessas máquinas, os registradores de endereçamento de cada nó só conseguem endereçar a sua memória local.

A Figura 1-10 apresenta uma visão geral da classificação segundo o compartilhamento de memória com exemplos de máquinas comerciais em cada classe [HWA98]. A linha tracejada indica que as máquinas das classes NCC-NUMA e NORMA podem ser transformadas em máquinas SC-NUMA através da inclusão de uma camada de software que implemente coerência de *cache*.

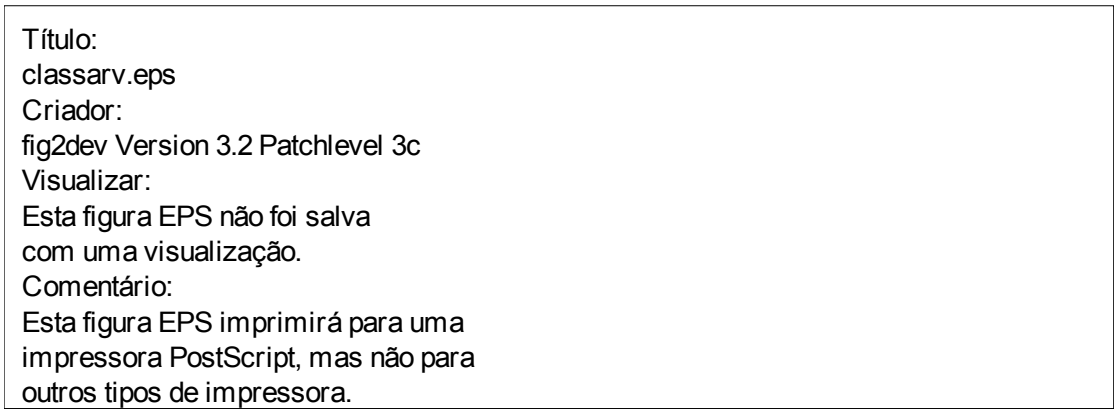

Figura 1-10: Visão geral da classificação segundo o compartilhamento de memória [HWA98]

1.3. Tendências na Construção de Máquinas Paralelas

Nesta seção são apresentados os principais modelos físicos de máquinas paralelas que constituem atualmente as principais tendências para a construção destes sistemas. Para cada modelo são apresentadas suas principais características e sua classificação segundo a terminologia definida na Seção 1.2. No final do capítulo são apresentadas tabelas e diagramas comparativos entre os modelos.

Cabe destacar que, segundo a classificação de Flynn, todos os modelos aqui apresentados pertencem à classe MIMD.

1.3.1. Processadores vetoriais paralelos (PVP)

Processadores vetoriais paralelos (PVP - *Parallel Vector Processors*) são sistemas constituídos de poucos processadores vetoriais poderosos (PV) que são especialmente desenvolvidos para esse tipo de máquina. A interconexão dos processadores a módulos de memória compartilhada (MC) é feita, na maioria dos casos, por uma matriz de chaveamento (*crossbar*) de alta vazão, também especialmente desenvolvida para essas máquinas (Figura 1-11).

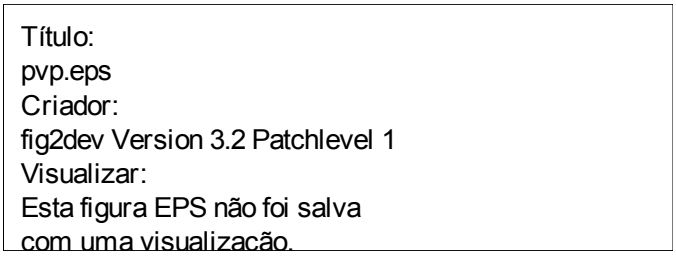

Figura 1-11: Arquitetura de um PVP (*Parallel Vector Processor*)

A implementação da memória compartilhada com blocos entrelaçados e o uso de uma rede de interconexão não bloqueante permite que os processadores acessem a memória em paralelo.

A comunicação entre os processadores é feita através da memória compartilhada que possui apenas um espaço de endereçamento que engloba todos os módulos de memória (todos os processadores podem endereçar todos os módulos de memória). Como o tempo de acesso à memória compartilhada é uniforme, essas máquinas são classificadas como multiprocessadores UMA. Esse tipo de máquina normalmente não se utiliza de memórias *cache*, usando para essa função um grande número de registradores vetoriais e um buffer de instrução.

São exemplos de PVP o Cray C-90, Cray T-90, Cray Y-MP, Fujitsu VP 2000, Fujitsu VPP 500 e NEC Sx-4.

1.3.2. Multiprocessadores simétricos (SMP)

Multiprocessadores simétricos (SMP - *Symmetric Multiprocessors*) são sistemas constituídos de processadores comerciais, também denominados "de prateleira" (*of the shelf*), conectados a uma memória compartilhada (MC) na maioria dos casos através de um barramento de alta velocidade (Figura 1-12). Como a maioria dos processadores comerciais encontrados no mercado utiliza-se amplamente de memórias *cache*, tanto no chip quanto fora dele, os processadores foram representados na figura por P/C. Como conseqüência, o barramento utilizado nessas máquinas implementa coerência de *cache* através do protocolo *snoopy*.

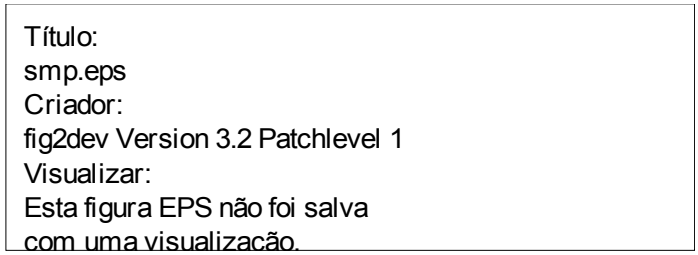

Figura 1-12: Arquitetura de um SMP (*Symmetric Multiprocessor*)

O adjetivo simétrico refere-se ao fato de que todos os processadores têm igual acesso ao barramento e à memória, não ocorrendo privilégios por parte do sistema operacional a nenhum dos processadores no atendimento de requisições. Isso resulta em um maior grau de paralelismo do que em uma arquitetura assimétrica (com um processador mestre, que possui vários privilégios no acesso aos recursos, e vários processadores escravos) [HWA98].

Devido à existência de uma memória compartilhada por todos os processadores, o que caracteriza essas máquinas como multiprocessadores UMA, o paradigma natural de comunicação nesses sistemas é o de memória compartilhada. Por se tratar de um paradigma mais próximo da programação feita em sistemas convencionas, a programação desses sistemas é considerada mais fácil do que em máquinas que comunicam por troca de mensagens. Os programas que definem múltiplos fluxos de execução, por exemplo, aproveitam-se automaticamente dos múltiplos processadores desse tipo de sistema (por exemplo, programas Java que possuam várias Threads).

Um fator que compromete a escalabilidade dessas máquinas é o uso de um barramento como rede de interconexão. O barramento impede a construção de multiprocessadores simétricos com um grande número de processadores por se tratar de um canal compartilhado que só permite uma transação por vez. Isso faz com que o seu desempenho e conseqüentemente o desempenho global da máquinas caia, a medida que aumenta a disputa por seu acesso (ou seja, que aumenta o número de processadores ligados a ele). As maiores máquinas SMP encontradas hoje no mercado possuem em torno de 50 processadores.

São exemplos de SMP o IBM R50, SGI Power Challenge, SUN Ultra Enterprise 10000, HP/Convex Exemplar X-Class e DEC Alpha Server 8400.

1.3.3. Máquinas maciçamente paralelas (MPP)

Máquinas maciçamente paralelas (MPP - *Massively Parallel Processors*) são multicomputadores construídos com milhares de processadores comerciais (P/C) conectados por uma rede proprietária de alta velocidade (Figura 1-13). A expressão "maciçamente paralela" indica a proposta dessas máquinas: a obtenção de alto desempenho através da utilização de um grande número de processadores, os quais, devido ao fator custo, acabam sendo processadores de poder computacional médio ou pequeno. Essa é uma alternativa à proposta dos PVPs, por exemplo, em que o alto desempenho é obtido através da utilização de poucos processadores vetoriais de grande poder computacional.

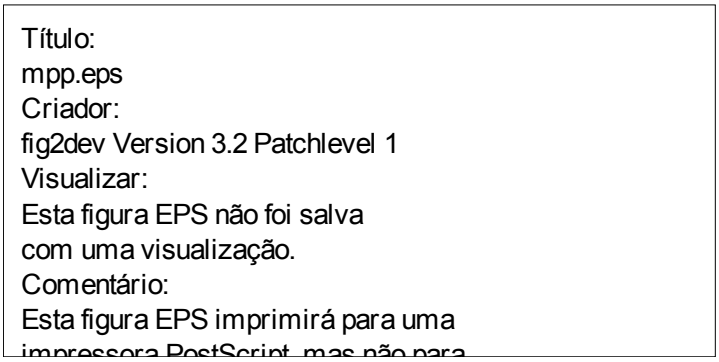

Figura 1-13: Arquitetura de um MPP (*Massively Parallel Processor*)

Para que essas máquinas possam ser altamente escaláveis, suportando tantos processadores, cada nó possui sua memória local (ML) com um espaço de endereçamento próprio. Dessa forma, não é possível o acesso à memória de nós vizinhos, caracterizando um multicomputador NORMA. A ligação desses nós à rede de interconexão é feita através de um adaptador de rede (AR).

Como conseqüência direta dessa organização de memória distribuída com vários espaços de endereçamento locais, a comunicação nessas máquinas é realizada através de troca de mensagens. Esse paradigma é considerado de mais difícil programação do que o paradigma de memória compartilhada, por ser mais distante da programação feita em sistemas tradicionais.

São exemplos de MPP o Intel Paragon, Connection Machine CM-5 e IBM SP2.

1.3.4. Máquinas com memória compartilhada distribuída (DSM)

Máquinas com memória compartilhada distribuída (DSM – *Distributed Shared Memory*) são sistemas em que, apesar de a memória se encontrar-se fisicamente distribuída através dos nós, todos os processadores podem endereçar todas as memórias. Isso se deve à implementação de um único espaço de endereçamento. Essa implementação pode ser feita em hardware, em software ou ainda com a combinação dos dois. A distribuição da memória, por sua vez, pode ser resultado da escolha de uma arquitetura multiprocessada com memória entrelaçada distribuída (máquinas NUMA) ou de uma arquitetura de multicomputador com memórias locais (máquina NORMA). Em ambos os casos, a máquina resultante é considerada CC-NUMA se tiver coerência de *cache* implementada em hardware ou SC-NUMA se a implementação for em software.

A Figura 1-14 apresenta a arquitetura de uma máquina com memória compartilhada distribuída derivada de um multicomputador NORMA. Nesse caso, os vários nós com suas memórias locais (ML) são interligados através de adaptadores de rede (AR) a uma rede de interconexão específica. O que diferencia essa arquitetura da máquina maciçamente paralela da Figura 1-13 é a possibilidade do acesso às memórias remotas e a utilização de um diretório (DIR) para a implementação da coerência de *cache*. É importante destacar que, em alguns casos, a coerência de *cache* não é implementada com um diretório, como no caso do Cray T3D que usa uma combinação de hardware especial e extensões de software para garantir a coerência em nível de bloco (o tamanho do bloco pode variar de poucas palavras a páginas inteiras). Outra alternativa é a construção de uma máquina DSM a partir de uma máquina NORMA, utilizando-se apenas de extensões de software como o TredMarks [AMZ96].

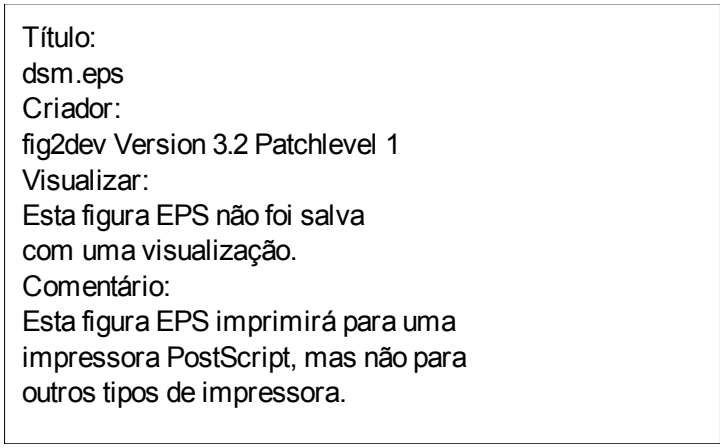

Figura 1-14: Arquitetura de uma máquina DSM (*Distributed Shared Memory*)

São exemplos de DSM o Stanford DASH, o Cray T3D e estações de trabalho rodando TreadMarks.

1.3.5. Redes de estações de trabalho (NOW)

Redes de estações de trabalho (NOW – *Network of Workstations*) são sistemas constituídos por várias estações de trabalho interligadas por tecnologia tradicional de rede como Ethernet [SOA95] e ATM [HAN98]. Na prática, uma rede local de estações que já existe é utilizada na execução de aplicações paralelas. Sob o prisma das arquiteturas paralelas, a rede local pode ser vista como uma máquina paralela em que vários processadores com suas memórias locais (estações de trabalho) são interligados por uma rede, constituindo assim uma máquina NORMA de baixo custo (ou sem nenhum custo, caso a rede já exista).

A Figura 1-15 apresenta a arquitetura dessas máquinas. A diferença para a arquitetura de um MPP (Figura 1-13) reside na hierarquia de barramento utilizada nas estações, na presença de um disco local (DL) nos nós e na rede de interconexão utilizada. Essas diferenças advêm do fato de a rede local ter sido concebida para um tipo de utilização bem diferente do que a execução de aplicações paralelas (compartilhar arquivos e acessar periféricos remotos como a impressora). A arquitetura de barramentos hierárquica dos nós foi adotada para suportar um grande número de periféricos locais sem prejudicar o desempenho do processador. Periféricos mais lentos e de menor importância são conectados a barramentos de E/S mais distantes do processador, que são endereçados através de adaptadores de barramento (AB). Nesse contexto, a placa de rede que, no caso da rede local, não tem uma grande importância, é conectada a um desses barramentos de E/S. No entanto, a rede tem um papel muito importante em uma máquina paralela, e esse acoplamento fraco (*loosely coupled*) do adaptador de rede (AR) implica uma grande perda de desempenho nas comunicações entre nós. O disco local, por sua vez, está presente nos nós porque as estações de trabalho foram concebidas para executar programas de forma autônoma, e o disco é utilizado como memória secundária. Em uma máquina paralela, os nós trabalham de forma cooperativa e uma memória secundária só seria estritamente necessária na máquina hospedeira. Outra dificuldade encontrada nessa troca de função é a rede de interconexão. Redes tradicionais como Ethernet e ATM não são otimizadas para as

operações de comunicação de uma aplicação paralela. O resultado é uma alta latência nessas operações que compromete o desempenho da máquina como um todo.

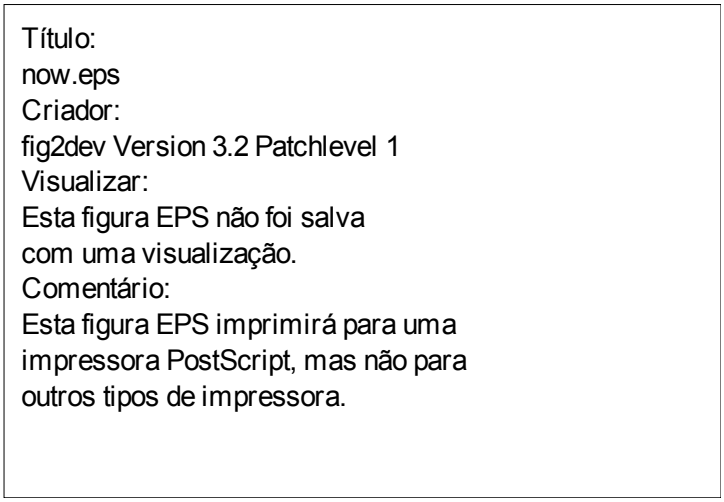

Figura 1-15: Arquitetura de uma máquina NOW (*Network of Workstations*)

Na prática, devido ao baixo desempenho da rede de interconexão, máquinas NOW são utilizadas como ambientes de ensino de processamento paralelo e distribuído ou na execução de aplicações em que a comunicação entre os nós não é muito intensa.

O exemplo mais comum de NOW são estações de trabalho interligadas por tecnologia Ethernet.

1.3.5.1. Estudo de casos: Rede local Ethernet, Fast-Ethernet e Gigabit Ethernet

O protocolo Ethernet padrão se tornou um sinonimo de interligação em rede de estações de trabalho. Esta tecnologia é amplamente utilizada tanto nas universidades como nas empresas. Seus 10 Mbits/s porém não são mais suficientes para a utilização em ambientes onde a densidade do tráfico é muito grande ou quando são necessárias transferências de grandes arquivos.

Uma versão melhorada do mesmo protocolo, conhecida como Fast-Ethernet atinge 100 Mbits/s e foi desenvolvida para permitir uma fácil transição para instalações Ethernet já existentes. Como o mesmo tipo de cabo é usado em ambos os casos, é possível efetuar a troca do *hub* ou *switch* para o padrão Fast-Ethernet e só trocar as placas de rede das máquinas onde o aumento de vazão se faz mais necessário. Sendo assim, ambos os padrões podem coexistir até que a atualização da placa de rede seja efetuada em todas as máquinas da rede.

Hoje o estado da arte em tecnologia Ethernet é o padrão Gigabit Ethernet. Novamente foram mantidas as características do protocolo Ethernet para facilitar a migração e a vazão foi elevada para 1 Gbit/s.

A Figura 1-16 apresenta resultados de latência e vazão para a comunicação com a biblioteca MPI (*Message Passing Interface* [GRO96]) entre duas máquinas interligadas por rede Fast-Ethernet. Foi utilizado um agregado NOW com máquinas baseadas em processadores Intel Pentium III de 550MHz rodando o sistema operacional Linux (Slackware 8.0) e se utilizando de placas Fast-Ethernet interconectadas por uma *switch*.

Figura 1-16: Latência e vazão do MPI sobre uma conexão Fast-Ethernet para diferentes tamanhos de mensagem

A maior vazão obtida (10,24 Mbytes/s) está muito próxima do limite teórico de 12,5 Mbytes/s (100 Mbits/s) mas a menor latência (165µs) se encontra bem acima do limite teórico de 7µs (indicado na documentação da *switch* utilizada). Essa diferença se explica pelo custo adicionado pelas várias camadas do protocolo TCP/IP, todas executadas em software, além do custo do próprio MPI. Otimizações do protocolo TCP/IP que eliminam estes vários níveis, como por exemplo o protocolo VIA [VIA98], reduzem essa latência para algo em torno de 60µs, mais próximo do limite teórico da rede Fast-Ethernet.

Outro grande problema das redes Ethernet, que afeta sua utilização em máquinas NOW e COW com aplicações que gerem muita comunicação, é sua grande sensibilidade ao aumento de tráfego. Como o protocolo Ethernet se utiliza de uma arbitragem baseada em detecção de colisão, tanto a latência como a vazão são muito afetadas quando do aumento de tráfego na rede. A utilização de uma *switch* em vez de um *hub* para interligação das máquinas ameniza esse problema.

1.3.6. Máquinas agregadas (COW)

Máquinas agregadas (COW – *Cluster of Workstations*) podem ser vistas como uma evolução das redes de estações de trabalho. Como nas NOW, também são constituídas por várias estações de trabalho interligadas, mas com a diferença de terem sido projetadas com o objetivo de executar aplicações paralelas. Dessa forma, a máquina pode ser otimizada para esse fim e, na maioria dos casos, as estações que servem de nó não possuem monitor, teclado e mouse, sendo denominadas estação de trabalho "sem cabeça" (*headless wokstation*). A máquina resultante também é chamada de "pilha de computadores pessoais" (*Pile-of-PC´s*). Mas as principais otimizações são feitas no software. Como essas estações não vão ser usadas localmente, o sistema operacional pode ser "enxugado", e vários servidores, desabilitados. Várias camadas de

rede podem ser simplificadas, ou até mesmo totalmente eliminadas, pois as necessidades de comunicação em máquinas paralelas são diferentes das necessidades em redes locais (em aplicações paralelas científicas, por exemplo, 87% das mensagens enviadas são menores do que 1Kbyte [VAN94]). Na prática, são aproveitadas as principais vantagens dos sistemas NOW, que são o ótimo custo/benefício e a grande flexibilidade de construção, e tenta-se amenizar a principal desvantagem, que é a alta latência das comunicações, construindo uma NOW **dedicada** ao processamento paralelo e distribuído.

Já que o sistema está sendo concebido para se dedicar à execução de aplicações paralelas, a rede de interconexão também pode ser otimizada para esse fim. Aqui encontramos atualmente duas tendências:

- **Agregados interligados por redes padrão**

  Esta tendência é impulsionada pelos grandes fabricantes como HP, IBM e Dell, que estão interessados na construção de máquinas paralelas poderosas, agregando milhares de estações de trabalho de baixo custo (*low end*). Para a interligação de tantas máquinas, fica muito caro investir em uma rede especial, sendo usada, para esse fim, uma rede padrão como a rede Ethernet. Para melhorar um pouco a latência da rede, são usados grandes chaveadores (*switches*) em contraste aos *Hubs* que funcionam como grandes barramentos, solução mais comum em redes locais por causa do menor custo. O enfoque aqui é a obtenção de desempenho com muitos nós de pequeno poder computacional e de preferência com aplicações que não tenham muita necessidade de comunicação. Essas máquinas são claramente NORMA, já que não é possível o acesso à memória de nós remotos e o paradigma de comunicação utilizado é troca de mensagens.

- **Agregados interligados por redes de baixa latência**

  Esta tendência é impulsionada por pequenas empresas que fabricam placas de interconexão especificamente para máquinas agregadas. Essas placas implementam protocolos de rede de baixa latência otimizados para as características de comunicação de aplicações paralelas. Como o custo dessas placas é mais alto do que placas de rede padrão, fica muito caro construir máquinas com muitos nós (mais do que algumas centenas de nós). Para compensar o menor número de nós, são usados nós mais poderosos, muitas vezes servidores de médio porte com vários processadores (multiprocessadores SMP). A máquina resultante fica mais equilibrada na relação poder de processamento do nó e desempenho da rede, obtendo um bom desempenho, mesmo com aplicações que necessitem de muita comunicação. A maioria dessas máquinas é NORMA, já que, a princípio, não é possível o acesso à memória de nós remotos. Porém algumas das placas implementam um espaço de endereçamento global entre os nós, permitindo o acesso de memórias remotas. Nesse caso, a máquina resultante é NUMA, e são suportados em hardware tanto a troca de mensagens como o acesso a áreas de memória compartilhada.

A Figura 1-17 apresenta a arquitetura dessas máquinas. A diferença para a arquitetura de um NOW (Figura 1-15) é muito pequena, já que se trata de uma evolução do mesmo princípio. Destaca-se na figura, com linhas tracejadas, a possibilidade de inclusão de mais de um processador (P/C). A principal diferença é que a rede de interconexão utilizada pode ser tanto padrão como uma rede de baixa latência.

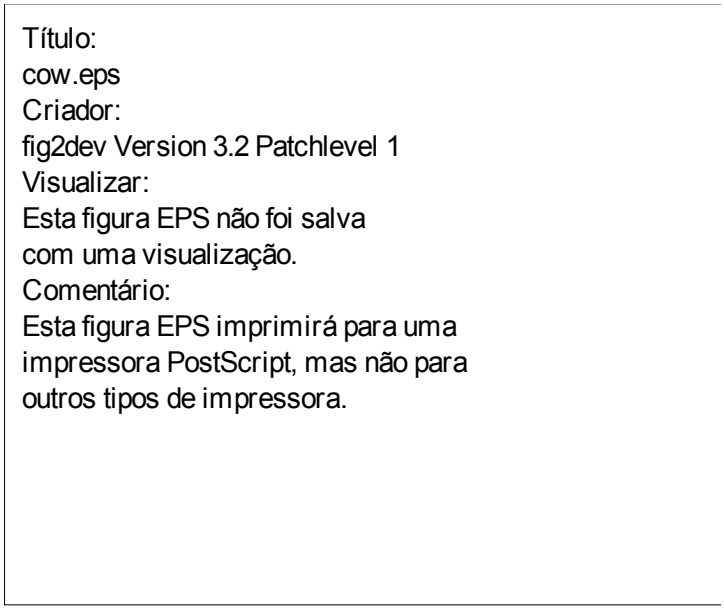

Figura 1-17: Arquitetura de uma máquina COW (*Cluster of Workstations*)

Entre as principais vantagens na utilização de agregados, temos:

- Relação custo × benefício

  Apresenta uma ótima relação custo × benefício, possuindo um dos menores custos por Mflops/s (milhões de instruções de ponto flutuante por segundo) em comparação às outras arquiteturas paralelas. Isso se deve principalmente ao fato de serem utilizados, na sua confecção, basicamente "componentes de prateleira", que, por serem produzidos em grande escala, possuem custo reduzido.

- Configurabilidade

  A arquitetura possui um alto grau de configurabilidade, por tratar-se de uma arquitetura aberta. A definição de seus componentes (nós e rede de interconexão) é generalizada, possibilitando diversas configurações.

- Custo de Manutenção

  A arquitetura possui um baixo custo de manutenção, pois os "componentes de prateleira" utilizados são facilmente encontrados no mercado e, como já vimos, possuem custo reduzido.

Devido a essa ótima relação custo × benefício, fabricantes de MPPs estão se utilizando de alguns conceitos da construção de agregados para diminuir o custo de seus produtos. Sendo assim, as diferenças entre essas máquinas diminuíram bastante a ponto de ficar difícil sua classificação. Um exemplo disso é a família SP2 da IBM . Essa máquina é considerada um MPP, mas, na sua construção, foram interligadas estações de trabalho RS6000 através de uma rede de baixa latência, resultando em uma arquitetura de máquina agregada. As principais diferenças entre agregados e MPPs são:

- Em um agregado, o adaptador de rede é fracamente acoplado (*loosely coupled*) ao processador, pois reside em um barramento E/S que é ligado ao barramento processador memória através de um adaptador. Isso é uma conseqüência da utilização de uma estação de trabalho como nó que possui vários níveis de barramento para suportar vários periféricos. Em um MPP, o nó possui normalmente só um nível de barramento (Figura 1-13), e a placa de rede é ligada diretamente nesse barramento, caracterizando um acoplamento forte (*tightly coupled*). Essa maior proximidade com o processador melhora o desempenho do subsistema de comunicação.
- Um disco local (P/C) está sempre presente no nó de um agregado e pode estar ausente no nó de um MPP.
- Um sistema operacional completo é instalado em cada nó de um agregado, enquanto os nós de um MPP possuem normalmente apenas o núcleo do sistema (*microkernel*).

São exemplos de COW o iCluster do HP Labs de Grenoble com rede Fast-Ethernet, o Primergy Server do PC2 em Paderborn com rede rápida SCI [HEL99] e a máquina Amazônia do CPAD com rede rápida Myrinet [BOD95].

1.3.6.1. Estudo de caso: iCluster do HP Labs de Grenoble (Rede Ethernet)

O padrão Fast-Ethernet, originalmente desenvolvido para a interligação de máquinas em redes locais (ver Seção 1.3.5.1), também é bastante utilizado na construção de máquinas agregadas. O desempenho da comunicação com o MPIch [GRO96] rodando sobre o sistema operacional Linux nesse tipo de rede atinge uma vazão de 10 Mbytes/s e uma latência de 100 μs aproximadamente.

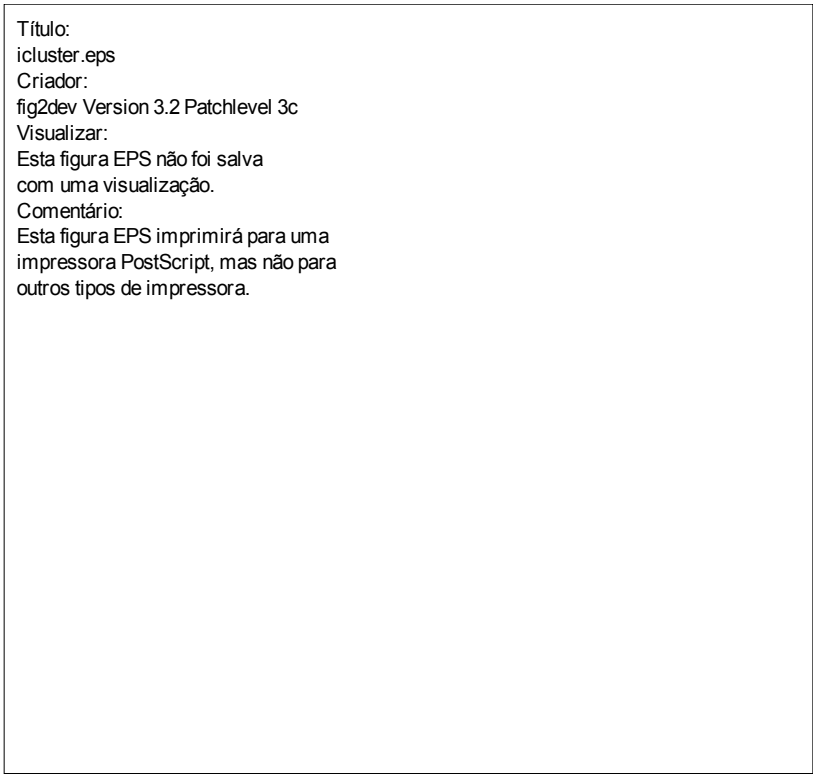

Figura 1-18: Arquitetura do agregado iCluster do centro HP Labs de Grenoble composto de nós monoprocessados interligados por uma rede totalmente interconectada de chaveadores Fast-Ethernet

Um exemplo desse tipo de agregado é o iCluster do centro HP Labs de Grenoble (França). A 18 apresenta a arquitetura dessa máquina, que é composta de 216 nós monoprocessados. Esses nós são agrupados em 5 redes Fast-Ethernet chaveadas (HP Procurve 4000 *switch*), sendo os chaveadores interligados através de uma rede totalmente interconectada de 1 Gbit/s. Cada nó possui um processador Intel Pentium III de 733 MHz e 256 Mbytes de memória principal. As máquinas executam o sistema operacional Linux (Mandrake 7.1). Essa máquina encontrava-se na posição 385 na lista de junho de 2001 das 500 mais rápidas do mundo (www.top500.com) com um desempenho de 81,6 Gflops/s rodando o teste de desempenho Linpack.

1.3.6.2. Estudo de caso: Primergy Server do PC^2 em Paderborn (Rede SCI)

SCI (*Scalable Coherent Interface* [HEL99]) é um padrão IEEE (1596-1992) que foi desenvolvido com o objetivo de prover um barramento de alta velocidade para a interligação de processadores em máquinas paralelas. A comunicação é realizada através do acesso à memória dos nós remotos através da implementação em hardware de uma memória distribuída compartilhada (DSM – *Distributed Shared Memory*) de baixa latência e alta vazão [BUY99]. A coerência das *caches* do sistema é garantida em hardware, sendo o sistema resultante uma máquina CC-NUMA.

A implementação desse padrão em placas PCI possibilitou sua utilização na construção de agregados. Os nós da máquina são interligados por conexões ponto-a-ponto que formam um anel unidirecional. Uma operação de escrita em uma memória remota é automaticamente direcionada para a placa SCI no barramento PCI da máquina de origem, que encaminha a operação para a placa SCI da máquina destino através do anel. Um anel comporta a ligação de até 12 máquinas. O roteamento no anel é feito em hardware e a placa permite a conexão de um segundo anel para a implementação de um *torus* (um anel horizontal e outro vertical – Figura 1-19), permitindo a ligação de até 144 nós. A construção de máquinas maiores pode ser realizada através da ligação de vários agregados SCI com a utilização de *switches*. A vazão teórica desse anel é de 6,4 Gbits/s. Cabe destacar que essa vazão é compartilhada por todas as operações que circulam no anel, e que o barramento PCI da maioria das máquinas atualmente encontradas no mercado (largura de 32 bits e freqüência de 33 MHz) limita a vazão máxima ponto-a-ponto a aproximadamente 130 Mbytes/s.

Título:
maquina.eps
Criador:
fig2dev Version 3.2 Patchlevel 3c
Visualizar:
Esta figura EPS não foi salva
com uma visualização.
Comentário:
Esta figura EPS imprimirá para uma

Figura 1-19: Arquitetura do agregado Primergy Server do centro PC2 em Paderborn composto de nós biprocessados interligados por um *torus* SCI

A versão PCI da placa SCI não implementa coerência de *cache* em hardware de forma que as operações de comunicação não podem aproveitar-se das *caches* do sistema (*non-cachable*). Como as placas de E/S, em arquiteturas modernas, não são mais ligadas ao barramento principal da máquina, onde se encontra o processador e suas caches, a placa SCI não consegue "farejar" as operações da *cache*, não conseguindo implementar o protocolo *snoopy* previsto no padrão.

O desempenho teórico de uma placa SCI fica em torno de uma latência de 3μs para operações em memórias remotas. Uma versão da biblioteca MPI para essas placas, o ScaMPI [HUS99], obtém uma latência em torno de 5 μs no envio de uma mensagem de 1 byte.

Um exemplo desse tipo de agregado é o Primergy Server do centro PC2 em Paderborn (Alemanha). A Figura 1-19 apresenta a arquitetura dessa máquina, que é composta de 96 nós biprocessados interligados por um *torus* bidimensional SCI de dimensão 12 × 8. Na Figura 1-19, foi representada apenas parte dessa rede de interconexão (8 × 4). Cada nó possui dois processadores Intel Pentium III de 840 MHz e 512 Mbytes de memória principal. As máquinas executam o sistema operacional Linux (RedHat 6.2), e o desempenho da comunicação com o ScaMPI atinge uma vazão de 85 Mbytes/s e uma latência de 5.1 μs. Essa encontrava-se na posição 351 na lista de novembro de 1999 das 500 mais rápidas do mundo (www.top500.com) com um desempenho de 41,45 Gflops/s rodando o teste de desempenho Linpack.

1.3.6.3. Estudo de caso: Amazônia do CPAD - PUCRS/HP (Rede Myrinet)

Myrinet [BOD95] é uma rede de interconexão full-duplex de 1,28 Gbits/s com latência nominal de 3 μs desenvolvida e comercializada pela empresa norte-americana Myricom. É uma tecnologia proprietária para a implementação de uma rede de baixa latência que é implementada em placas PCI para facilitar a construção de agregados. Essas placas são interligadas por matrizes chaveadoras que implementam o protocolo de roteamento *cut-through*. Por serem modulares, essas matrizes podem ser configuradas com o número desejado de portas, garantindo a escalabilidade da arquitetura.

O desempenho da comunicação com o MPIch [GRO96] rodando sobre o sistema operacional Linux nesse tipo de rede atinge uma vazão de 100 Mbytes/s e uma latência de 8 μs aproximadamente.

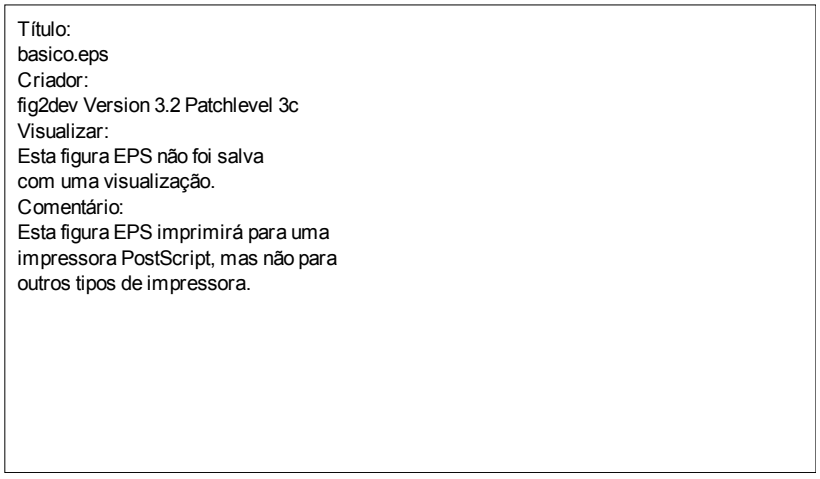

Figura 1-20: Arquitetura do agregado Amazônia do Centro de Pesquisa em Alto Desempenho (CPAD – PUCRS/HP) composto de nós biprocessados interligados por uma matriz de chaveamento Myrinet

Um exemplo desse tipo de agregado é a máquina Amazônia do Centro de Pesquisa em Alto Desempenho (CPAD – PUCRS/HP). A Figura 1-20 apresenta a arquitetura dessa máquina, que é composta de 16 nós biprocessados interligados por uma matriz de chaveamento Myrinet. Cada nó possui dois processadores Intel Pentium III de 550 MHz e 256 Mbytes de memória principal. As máquinas executam o sistema operacional Linux (Slackware 8). Essa máquina tem um desempenho aproximado de 12 Gflops/s rodando o teste de desempenho Linpack.

1.3.7. Comparação entre os modelos

1.3.7.1. Desempenho

A demanda por desempenho tem sido o principal motivador para a construção de máquinas paralelas. Portanto, é bastante natural que a primeira pergunta que surja após a apresentação das tendências na construção de máquinas paralelas seja em relação a qual dos modelos apresentados teria o melhor desempenho. Infelizmente, a pergunta, colocada dessa forma, não tem uma resposta na medida em que não se pode fazer uma comparação de desempenho, analisando apenas a arquitetura desses modelos e sem informações sobre o tipo de aplicação que será utilizado.

De forma resumida, os modelos vistos são classificados em relação ao número de processadores, à forma de ligação destes processadores à memória, e à existência de uma memória compartilhada no sistema. A escolha das tecnologias que implementam esses modelos é que, por um lado, vai determinar o desempenho final de uma máquina. Ou seja, mesmo que esteja definido que os processadores serão interligados a uma memória compartilhada através de um barramento, ainda existem muitas variações possíveis na fase de implementação como: a escolha do padrão de barramento utilizado, sua freqüência, sua largura, se será aplicada multiplexação, a opção por múltiplos barramentos, se a memória usada será entrelaçada, e assim por diante. O estudo de casos feito neste capítulo ilustra bem esse ponto na medida em que máquinas construídas a partir do mesmo modelo podem acabar tendo um desempenho bem diferente.

Outro fator muito importante na questão do desempenho de uma máquina paralela é definição da aplicação utilizada. Aplicações paralelas podem possuir diferentes características, e a forma como a máquina paralela atende às peculiaridades de cada caso tem um impacto muito grande no desempenho final do sistema. O mesmo raciocínio vale para as máquinas convencionais nas quais a relação entre a quantidade de cálculo e a quantidade de entrada e saída (*CPU bound*, *I/O bound*) influencia o desempenho final do sistema. Como, em uma máquina paralela a operação de entrada e saída pode resultar em comunicação, que é muitas vezes o gargalo do sistema, o problema acentua-se de tal forma, que é difícil que uma máquina paralela seja considerada genérica, ou seja, tenha um bom desempenho para qualquer tipo de aplicação.

Sendo assim, a pergunta inicial tem que ser reformulada para que possa ter uma resposta. Como vimos, faz-se necessário que os detalhes de implementação do modelo escolhido e as características da aplicação que vai ser executada sejam conhecidos para que possa ser estimado o desempenho obtido por uma determinado sistema.

Muitos usuários, porém, não possuem conhecimento para analisar detalhes de implementação e muito menos as características de suas aplicações quando paralelizadas. Nesses casos, podem ser usados testes de desempenho padronizados (*benchmarks*) que permitem a comparação entre diferentes sistemas.

Um *benchmark* é um programa de teste de desempenho que analisa as características de processamento e de movimentação de dados de um sistema de computação com o objetivo de medir ou prever seu desempenho e revelar os pontos fortes e fracos de sua arquitetura. *Benchmarks* podem ser classificados de acordo com a classe de aplicação para a qual são voltados como, por exemplo, computação científica, serviços de rede, aplicações multimídia, processamento de sinais, etc. Também podem ser divididos em *macrobenchmarks* e *microbenchmarks*. *Macrobenchmarks* medem o desempenho de um sistema como um todo. Comparam sistemas diferentes de acordo com uma classe de aplicação e auxiliam o comprador na escolha do melhor sistema para sua aplicação. Porém, não revelam porque um sistema tem um desempenho bom ou ruim. *Microbenchmarks*, por sua vez, medem aspectos específicos de um sistema, como o desempenho do processador, do acesso à memória, do subsistema de E/S, e do sistema operacional. *Benchmarks* podem ser ainda aplicações completas ou simplesmente núcleos, que são programas muito mais simples e de menor tamanho extraídos de aplicações mas mantendo as características principais. A aplicação utilizada pode ser uma aplicação que execute uma tarefa real ou um programa sintético especialmente desenvolvido para ser um teste de desempenho. Microbenchmarks são normalmente programas sintéticos. Mais de uma centena de testes de desempenho já foram propostos e se encontram em uso. A Tabela 1-2 apresenta alguns dos *benchmarks* mais conhecidos com suas principais características.

Testes de desempenho podem ser usados para dar uma idéia de desempenho comparativo entre os modelos apresentados neste capítulo. A Figura 1-21 apresenta a evolução de desempenho nas últimas duas décadas das 500 máquinas mais rápidas divididas por modelo. A comparação é feita com o teste de aplicação LINPACK que foi criado e é mantido por Jack Dongarra da universidade do Tennessee. Apesar de ser utilizado como teste de desempenho o LINPACK é constituído na verdade por um conjunto de rotinas na linguagem Fortran que analisam e resolvem equações lineares, e é amplamente utilizado na resolução de problemas computacionais reais. É simples e fácil de usar, e um bom indicador da capacidade de um sistema de resolver problemas de computação numérica.

Tabela 1-2: Principais características dos *Benchmarks* mais conhecidos [HWA98]

Tipo	Nome	Classe de Aplicação
Microbenchmark	LINPACK	Computação numérica (álgebra linear)
	LMBENCH	Chamadas de sistema e movimentação de dados em sistemas Unix
	STREAM	Largura de banda da memória
Macrobenchmark	NAS	Processamento Paralelo
	PARKBENCH	Processamento Paralelo
	SPEC	Família de testes incluindo várias classes
	Splash	Processamento Paralelo
	STAP	Processamento de sinais
	TCP	Aplicações comerciais

É importante destacar que a figura só apresenta a representatividade de cada modelo entre as 500 máquinas mais rápidas e a ordem de apresentação dos modelos eixo vertical não indica a posição desses modelos entre os mais rápidos. As listas completas desde 1994, com duas edições por ano (junho e novembro), pode ser encontrada na página http://www.top500.org.

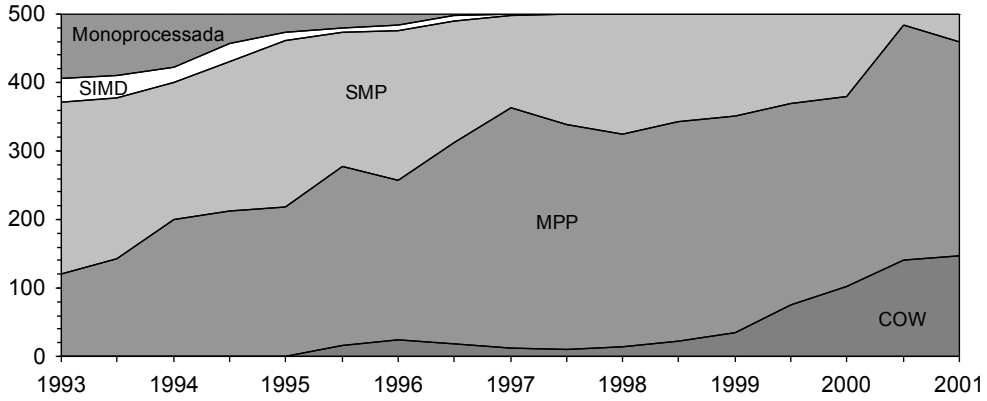

Figura 1-21: Representatividade de cada tendência na lista das 500 máquinas mais rápidas das últimas décadas (www.top500.org)

A classificação dos modelos usada nessas listas não é exatamente igual à apresentada neste capítulo, incluindo os modelos PVP e DSM dentro dos SMP. Além disso, um modelo chamado de *constellation* é usado para referenciar máquinas construídas a partir da replicação de nós SMP quando o número de processadores do nó é maior que o número de nós do sistema (por exemplo uma máquina com 64 procesadores formada pela ligação de quatro multiprocessadores cada um com 16 processadores). Na Figura 1-21, essas máquinas foram incluídas nos sistemas SMP.

A Figura 1-22 apresenta a representatividade de cada modelo na lista dos 500 sistemas mais rápidos em junho de 2001 segundo a classificação apresentada neste capítulo.

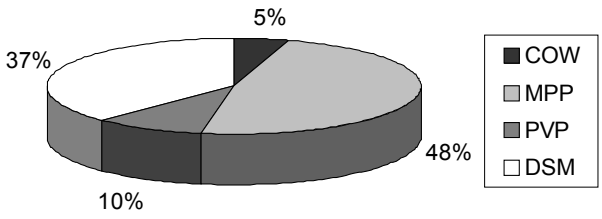

Figura 1-22: Representatividade de cada tendência apresentada neste capítulo na lista das 500 máquinas mais rápidas de junho de 2001 (www.top500.org)

Nos últimos anos, as máquinas mais rápidas da lista das top500 vêm sendo COWs e MPPs por causa de sua grande escalabilidade que permite a construção de máquinas com milhares de nós. Isso advém do fato de ser possível, no teste de desempenho LINPACK, que se aumente a carga do problema a ser resolvido de modo que todo o hardware disponível pode ser aproveitado. É importante destacar que, apesar dos resultados apresentados acima, aplicações de difícil distribuição ainda obtêm um melhor desempenho em máquinas PVP e SMP (DSM).

1.3.7.2. Relação custo × benefício

Como a principal motivação do uso de máquinas paralelas é a obtenção de desempenho, uma comparação dos modelos baseada na relação entre seu custo e o benefício resultante acaba se transformando em uma relação entre o custo e o desempenho obtido.

Mesmo com as dificuldades de se comparar o desempenho entre os modelos, no caso da relação custo/desempenho, é muito clara a vantagem dos agregados (COW) em relação aos outros modelos. A combinação do baixo custo de aquisição e manutenção desses sistemas, por causa da utilização de componentes de prateleira, com as opções de redes de baixa latência, que aproximam o desempenho a níveis de MPPs, tem aumentado o interesse por essas máquinas nos últimos anos. Outra conseqüência disso foi a redução dos preços dos MPPs, devido à incorporação de algumas tecnologias de agregados em sua construção. Como já vimos, em muitos casos, já fica difícil definir a fronteira entre MPP e COW e, conseqüentemente, classificar algumas máquinas, como no caso da máquina SP da IBM .

Por causa da grande demanda do mercado de servidores máquinas SMP de pequeno porte (até 4 processadores), tem havido redução de seu custo, o que resulta em uma melhora na sua relação custo/desempenho. No entanto, sistemas SMP de médio e grande porte não têm tanta saída e ainda não têm acompanhado a redução de preços dos sistemas menores.

As máquinas PVP ainda estão entre as de maior custo tanto de aquisição como de manutenção. Apesar de se basearem no modelo SMP e possuírem normalmente apenas algumas dezenas de nós, a arquitetura dos nós vetoriais difere bastante das arquiteturas convencionais. Dessa forma, muito pouco das tecnologias que são produzidas em maior escala pode ser aproveitado na confecção desses nós, o que aumenta o seu custo.

1.3.7.3. Escalabilidade

O termo escalabilidade, nesse contexto de comparação entre modelos, refere-se a capacidade de crescimento do sistema, ou seja, à capacidade de inclusão de nós ou de processadores sem que o funcionamento do sistema seja comprometido (por exemplo, se esse crescimento resultar no surgimento de um gargalo). Essa capacidade de crescimento, por sua vez, depende basicamente do tipo de rede utilizado, de sua topologia e vazão, e da forma com que a memória do sistema foi implementada, centralizada ou distribuída.

Por utilizarem, na maioria dos casos, redes estáticas com topologias regulares como anéis e malhas e implementarem a memória do sistema de forma distribuída, os MPPs, COWs e DSMs resultam normalmente na construção de máquinas com boa escalabilidade.

A utilização de barramentos para a interligação de processadores e a implementação de uma memória centralizada, como é normalmente o caso em PVPs e SMPs, tende a comprometer a escalabilidade dos sistemas.

A escalabilidade de um sistema influencia diretamente a escolha do poder de processamento de seus processadores. Quando os modelos não são muito escaláveis, a tendência é equipá-los com processadores de maior desempenho, seguindo o enfoque da obtenção de desempenho com poucos nós poderosos. Se o sistema tem boa escalabilidade, a tendência é a utilização de nós mais baratos de menor desempenho, seguindo o enfoque da "união faz a força".

1.3.7.4. Imagem única

O conceito de imagem única (*SSI – Single System Image*) refere-se a forma como um usuário percebe a máquina paralela como um todo. O caso ideal seria que o usuário abstraísse a forma como os recursos foram replicados e interligados e visse o sistema como uma única grande máquina, com uma memória compartilhada, um grande poder de processamento e um grande disco.

A Figura 1-23 apresenta a posição de cada um dos modelos em relação a uma visão única de sistema [HWA98]. Como podemos ver, os modelos baseados em multiprocessadores como SMPs e PVPs estão mais próximos dessa visão única, enquanto os modelos baseados em multicomputadores estão mais distantes. Isso é bastante natural se considerarmos que, em multiprocessadores, apenas o recurso processador foi replicado em uma arquitetura convencional enquanto, em um multicomputador, toda a arquitetura convencional foi replicada.

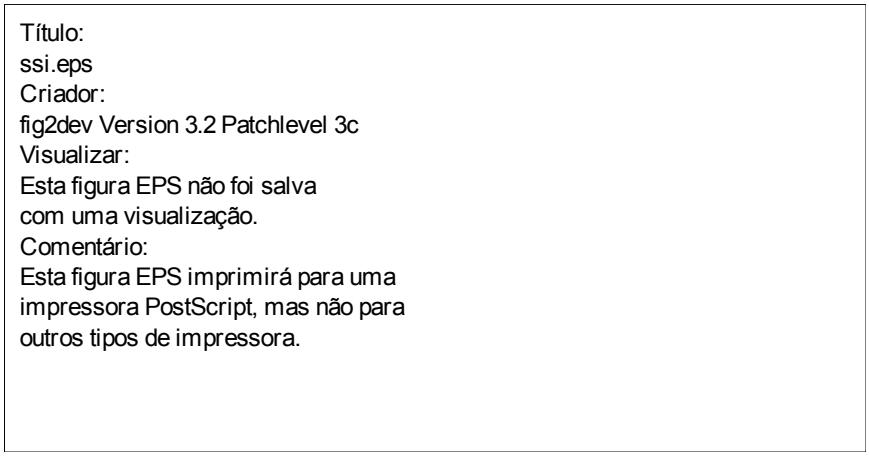

Figura 1-23: Modelos apresentados em relação a uma visão única de sistema

Essa questão é bastante significativa pois facilita a utilização desses sistemas. Em sistemas que fornecem uma visão única, o usuário não necessita levar em consideração se um determinado recurso foi implementado de forma distribuída, não precisando particionar sua aplicação. Caso necessário, esse particionamento é feito automaticamente pelo sistema.

1.3.8. Resumo das principais características

A Tabela 1-3 apresenta um resumo das características dos modelos apresentados neste capítulo.

Tabela 1-3: Comparação entre os modelos físicos para construção de máquinas paralelas

Tipo	**PVP**	**SMP**	**MPP**	**DSM**	**NOW**	**COW**
Estrutura	MIMD	MIMD	MIMD	MIMD	MIMD	MIMD
Comunicação	memória compart.	memória compart.	troca de mensagem	memória compart.	troca de mensagem	Troca de mensagem
Interconexão	crossbar	barramento ou crossbar	rede específica	rede específica	rede comum	rede rápida
Número de nós	~10	~50	100 a 5000	10 a 1000	4 a 5000	4 a 1000
Endereçamento	único	único	múltiplo	único	múltiplo	múltiplo
Processador	específico	comum	comum	específico ou comum	comum	comum
Acesso à memória	UMA	UMA	NORMA	NUMA	NORMA	NORMA

1.4. Agradecimentos

Inicialmente, gostaríamos de agradecer a Editora Sagra-Luzzatto por ter concordado com a reprodução de partes do livro: Arquiteturas Paralelas [DER02] (a ser publicado dentro da série Livros Didáticos do Instituto de Informática da UFRGS). Agradecemos, ainda, a Capes, a FAPERGS e a HP do Brasil pelo suporte econômico aos projetos de pesquisa dos nossos Grupos, que viabilizaram o desenvolvimento desse trabalho e de muitos outros.

1.5. Bibliografia

[AGE95] Agerwala, T. *et.all.* SP2 System Architecture. *IBM Systems Journal*, Vol. 34, No. 2, 1995. Pp. 152-184.

[ALM89] Almasi, G. S.; Gottlieb, A. *Highly Parallel Computing.* Benjamin/Cummings Publ. Comp., Redwood City, Cal., 1989.

[AMZ96] Amza, C.; Cox, A. L.; Dwarkads, S.; Keleher, P.; Rajamony, L.; Yu, W.; Zwaenepoel. W. Trademarks: shared memory computing on networks of workstations. *IEEE Computer*, 29 (2): 18-28, February, 1996.

[BOD95] Boden, N. *et.all.* Myrinet: A gigabit-per-second local-area network. *IEEE Micro*, 15(1):29-36, February, 1995.

[BUY99] Buyya, Rajkumar. *High Performance Cluster Computing: Architectures and Systems*, Prentice Hall, 1999.

[CUL99] Culler, David E.; Pal, Singh J.; Gupta, A. *Parallel Computer Architecture: a hardware/software approach.* Morgan Kaufmann Publishers. 1999.

[DER02] De Rose, César A. F.; Navaux, Philippe O. A. *Arquiteturas Paralelas*. Editora Sagra-Luzzatto, 2002 (a ser publicado).

[FLY72] Flynn, M. J. Some Computer Organizations and their Effectiveness. *IEEE Transaction on Computers* 21, v. 21, n. 9, pp. 948-960, 1972.

[GRO96] Gropp, W.; Lusk, E. *User´s Guide for mpich, a Portable Implementation of MPI*. 1996.

[HAN98] Haendel, M.; Huber, M.; Shroeder, S. *ATM Networks: Concepts Protocols Aplications.* Addison-Wesley, 1998. (terceira edição)

[HEL99] Hellwanger, H.; Reinfeld, A. *SCI: Scalable Coherent Interface*. : architecture and software for high-performance compute clusters. Springer-Verlag, 1999. 490p. (Lecture Notes in Computer Science, v.1734).

[HOC88] Hockney, R. W.; Jesshope, C. R. *Parallel Computers 2.* Adam Hilger, Bristol and Philadelfia, 1988.

[HUS99] Huse, L. *et.all.* ScaMPI – Design and implementation. In *SCI: Scalable Coherent Interface: Architecture and Software for High-Performance Computer Clusters*, volume 1734 of Lecture Notes in Computer Science, pages 249-261. Springer, 1999.

[HWA93] Hwang, K. *Advanced Computer Architecture: Parallelism, Scalability, Programmability.* McGraw-Hill. 1993.

[HWA98] Hwang, K.; Xu, Z. *Scalable Parallel Computing: technology, architecture, programming.* McGraw-Hill. 1998.

[LIA98] Liao, C.; Jiang, D.; Iftode, L.; Martonosi, M.; Clark W. *Monitoring Shared Virtual Memory Performance on a Myrinet-base PC Cluster.* Proceedings of International Conference on Supercomputing, July 1998.

[NIT91] Nitzberg, B.; Lo, V. Distributed Shared Memory: A Survey of Issues and Algorithms. *IEEE Computer*, Vol. 24, No. 8, pp. 52-60. 1991.

[PAT94] Patterson, David A.; Hennessy, John L. *Computer Organization & Design: The Hardware Software Interface.* Morgan Kaufmann Publishers, Inc. 1994.

[SOA95] Soares, L. F.; Lemos, G.; Colcher, S. *Redes de Computadores - das LANs, MANs e WANs às Redes ATM*. Editora Campus, 1995.

[VAN94] Van Voorsst, B.; Seidel, S.; Barszcz, E. Profiling the Communication Workload of na iPSC/860. *Proc. Scalable High Performance Computing Conf.*, 1994.

[VIA98] *Virtual Interface ArchitectureHome Page*. http://www.viarch.org/, 1998.