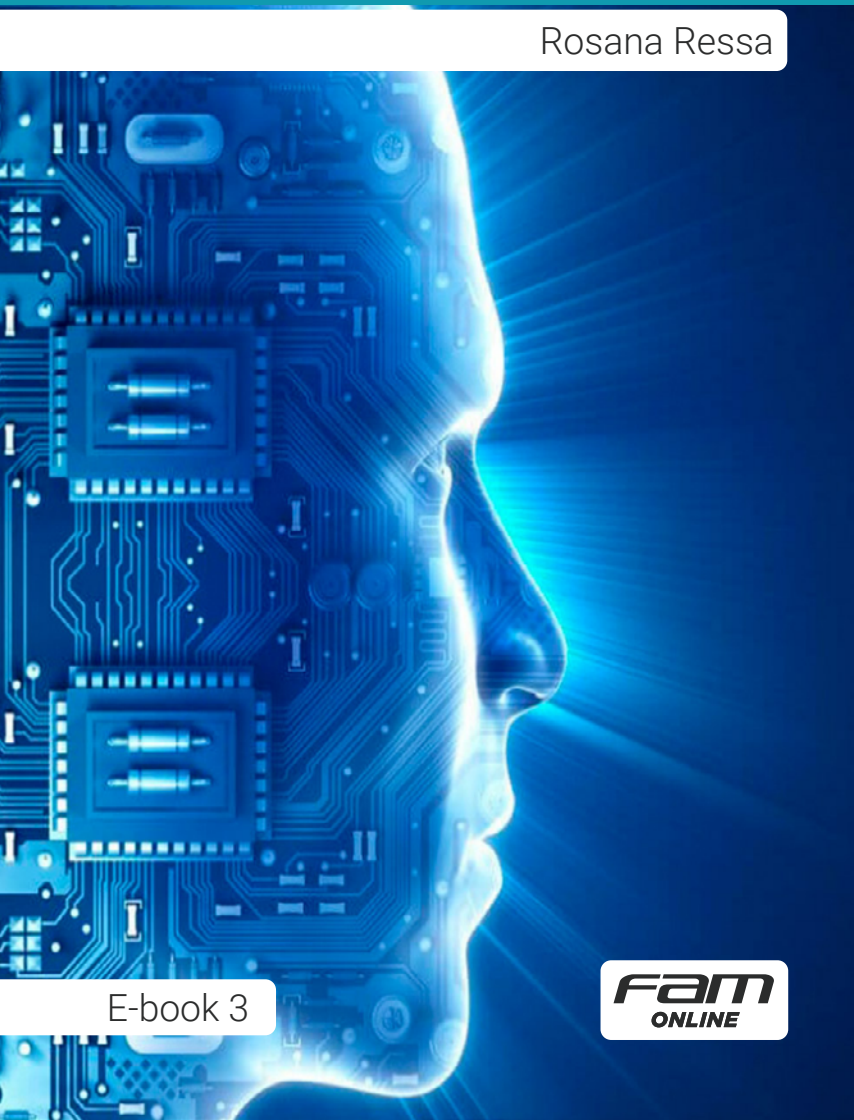
SISTEMAS COMPUTACIONAIS

Rosana Ressa



E-book 3

FAM ONLINE

Neste E-Book:

INTRODUÇÃO

Neste módulo, o conhecimento em sistemas computacionais se ampliará, à medida que apresentarmos alguns tópicos avançados em Arquitetura de Computadores, bem como as arquiteturas computacionais de alto desempenho. Trata-se de um tema mais robusto, que inicialmente fará uma breve exposição sobre os conceitos de arquiteturas paralelas, cujo objetivo de aprendizagem é que você seja capaz de definir paralelismo.

Em seguida, será apresentada a Classificação de Flynn. Este tópico diferencia arquiteturas de computadores com múltiplos processadores, empregando duas dimensões independentes: instruções e dados. O objetivo é que você seja capaz de diferenciar cada classe de processadores.

Depois, abordaremos os Multiprocessadores de Memória Compartilhada (SMP) e conheceremos os multiprocessadores UMA e NUMA. Posteriormente, estudaremos os Processadores Maciçamente Paralelos (MPP), os quais são multicomputadores e podem ser qualificados como NORMA, para você entender os conceitos envolvidos nas duas formas de paralelismo e observar a principal diferença entre os dois, que é a presença ou a ausência de memória compartilhada.

Por fim, abordaremos os Fundamentos de *Clusters* Computacionais, seu funcionamento, suas carac-

terísticas, vantagens e desvantagens. Classificado também como multicomputador, *Cluster* refere-se à arquitetura computacional que une dois ou mais máquinas como se fossem uma. O tópico visa a apresentar devidamente a arquitetura, a partir da definição de conceitos fundamentais e que você perceba que o uso da computação em *Cluster* é bem difundido.

ARQUITETURAS COMPUTACIONAIS DE ALTO DESEMPENHO

A aquisição de alto desempenho não depende apenas da utilização de dispositivos de hardware mais rápidos, mas também de avanços na arquitetura dos computadores e de técnicas de processamento. As arquiteturas avançadas de computadores estão embasadas no conceito de processamento paralelo, sendo que um deles se baseia em pipeline; outros, em arranjo de processadores e multiprocessadores.

O processamento em paralelo é uma forma eficaz de processamento de dados que explora eventos concorrentes no processo de computação. O interesse nesta unidade de aprendizagem é que você possa estudar tecnologias aplicadas ao melhoramento do desempenho dos computadores, bem como analisar as tecnologias do ponto de vista da eficiência e do custo.

CONCEITOS BÁSICOS DE ARQUITETURAS PARALELAS

Ao estudar sua história, observamos que o computador evoluiu muito e tornou-se cada vez mais rápido e eficiente, por conta de técnicas modernas que visam a aumentar a performance dos computadores com base em novas alternativas arquiteturais. Uma forma de atingir esse objetivo foi tornar os chips cada vez mais velozes, ou seja, aumentar sua velocidade de *clock*.

O paralelismo é um meio de se conseguir um desempenho ainda melhor, visto que pode ser introduzido de várias formas. O processamento paralelo, em linhas gerais, é o emprego simultâneo de vários recursos computacionais para resolver um problema, e pode-se acrescentá-lo ao chip da CPU por pipeline e projetos superescalares com várias unidades funcionais. Ainda, pode-se acrescentá-lo por meio de palavras de instrução muito longas com paralelismo implícito.

FIQUE ATENTO

Projetos superescalares: exploram o paralelismo de instrução. O processador realiza várias instruções em um ciclo de *clock*. Nele, há somente um pipeline, mas há múltiplas unidades funcionais. As unidades funcionais que exigem mais ciclos de *clock* são multiplicadas com a intenção de forçar a execução de um estágio em um tempo menor.

As propriedades especiais podem ser acrescentadas à CPU, o que permite que sejam manipuladas diversas *threads* de controle ao mesmo tempo e, enfim, várias CPUs podem ser concentradas no mesmo chip; juntos, esses atributos podem equivaler a 10 vezes mais performance em relação aos projetos sequenciais (TANENBAUM, 2013).

FIQUE ATENTO

Thread, no contexto de processadores, traduz-se em uma ordem de execução com instruções encadeadas que são realizadas uma por vez.

De acordo com Stallings (2017), é importante a distinção entre dois conceitos relacionados ao paralelismo: **em nível de instrução** e **de máquina**. O paralelismo em nível de instrução é aplicado quando as instruções de uma sequência são independentes e, deste modo, podem ser realizadas em paralelo (ao mesmo tempo) por sobreposição. Por sua vez, o paralelismo de máquina tem por habilidade tirar vantagem do paralelismo em nível de instrução; é orientado por um conjunto de pipelines paralelas. Os dois tipos de paralelismo são elementos importantes para melhorar o desempenho de um computador.

Nesse contexto, para o processamento paralelo, são exigidos alguns recursos computacionais, dentre eles está a combinação de uma unidade de processamento (um nó) com várias CPUs e um número ilimitado de computadores conectados por uma rede. Para que

essa rede funcione, os problemas computacionais devem ter atributos que os possibilitem ser divididos em “pedaços”, e que possam ser resolvidos de uma maneira concorrente ou simultânea; ainda, devem permitir a execução de diversas instruções de programa a qualquer momento no tempo de execução, além de possibilitar a solução em um menor espaço de tempo que o equivalente ao serial.

SAIBA MAIS

Saiba mais sobre o funcionamento do paralelismo. Para isso, assista a **Computação paralela: unindo vários computadores para um único propósito**, que está disponível em: *https://www.youtube.com/watch?v=wf0NCVrHFCk*. Acesso em: 18 jul. 2019.

De acordo com Navaux, De Rose e Pilla (2011), a aplicação do paralelismo está presente nos vários **níveis** de um sistema, pois o paralelismo pode ser definido a partir de sua granularidade, indicada pela relação entre a dimensão de cada tarefa e a dimensão total do programa (ou a razão entre computação e comunicação). Dessa maneira, o **Grão de Paralelismo** pode ser grosso, médio ou fino, conforme posto a seguir (NAVAUX; DE ROSE; PILLA, 2011, p. 25-26):

- **Grão grosso:** o trabalho a ser feito pode ser particionado em unidades de trabalho grandes. Mesmo pagando um alto custo de comunicação, é grande a chance de se obter ganho de desempenho delegando estas unidades de trabalho a outras unidades ativas

(o custo do envio é compensado pelo ganho de tempo em atacar o problema com mais unidades).

- **Grão médio:** o trabalho a ser feito só pode ser particionado em unidades de trabalho médias. Em caso de alto custo de comunicação, pode ser difícil obter ganho de desempenho delegando estas unidades de trabalho a outras unidades ativas (o custo do envio não é necessariamente compensado pelo ganho de tempo em atacar o problema com mais unidades).
- **Grão fino:** o trabalho a ser feito só pode ser particionado em unidades de trabalho pequenas. Em caso de alto custo de comunicação, não vale a pena delegar estas unidades de trabalho a outras unidades ativas (o custo do envio não é compensado pelo ganho de tempo em atacar o problema com mais unidades).

Genericamente, o emprego do paralelismo visa sobretudo a reduzir o tempo total de execução, a resolução de grandes problemas e o uso de vários processamentos ao mesmo tempo (concorrência). Além disso, podemos citar outras razões:

1. Tirar proveito dos recursos disponíveis na rede (Internet), disponibilizados por grandes núcleos de processamento de dados (computação em grade).

2. Diminuir os custos de processamento, usando estações de trabalho e processadores comuns no mercado.

3. Recursos de memória.

4. Limitação do processamento em máquinas seriais.

Portanto, com vistas a melhorar cada vez mais o desempenho computacional, os arquitetos de computadores estão recorrendo à computação paralela, motivados por grandes desafios computacionais científicos, como previsão do tempo, genoma humano e simulações físicas. Sem contar as aplicações comerciais que necessitam de processamento de imensas bases de dados, como banco de dados com processamento paralelo, mineração de dados, realidade virtual e tecnologia em multimídia.

Acesse o Podcast 1 em Módulos

CLASSIFICAÇÃO DE FLYNN

Existem várias maneiras de classificar computadores paralelos. Uma das mais usadas, desde 1966, é a classificação de Flynn. Essa classificação caracteriza arquiteturzas de computadores com múltiplos processadores, combinando duas medidas independentes: **Instrução** e **Dados**. Cada medida pode ter apenas dois estados: **Simples** ou **Múltiplos**. Observe o Quadro 1:

	Dado simples	Dado múltiplo
Instrução simples	SISD	SIMD
Instrução múltipla	MISD	MIMD

Quadro 1: Classificação da taxonomia de Flynn. **Fonte:** Elaboração Própria.

A classificação Single-instruction-single-data (**SISD**), que significa instrução-simples-dado-simples, representa um programa totalmente sequencial. Nessa classe, um único fluxo de instruções atua sobre um único fluxo de dados, uma propriedade bastante característica do modelo von Neumann. Embora os programas sejam dispostos por meio de instruções sequenciais, podem ser realizadas de forma sobreposta em diferentes estágios (pipelining).

A categoria Single-instruction-multiple-data (**SIMD**), que significa instrução-simples-dados-múltiplos, apresenta o processamento de vários dados sob

o comando de uma única instrução. Na arquitetura SIMD, o programa segue ainda uma organização sequencial. Um exemplo de seu uso é em aplicações que precisam processar vetores, como no tratamento de imagens (NAVAUX; DE ROSE; PILLA, 2011).

A classe Multiple-instruction-single-data (**MISD**), que significa instrução-múltipla dado-simples, é uma classificação raramente empregada, pois a ideia é que um único conjunto de dados seja processado por múltiplas CPUs. Alguns estudiosos consideram o pipeline um exemplo dessa categoria, mas não há um consenso (FERNANDEZ, 2015).

Já a classificação Multiple-instruction-multiple-data (**MIMD**), que significa múltiplas-instruções-múltiplos--dados, é a mais genérica e aplicada à grande parte dos programas paralelos. Ela abrange o processamento de múltiplos dados por parte de múltiplas instruções. Os servidores multiprocessados representam esta classe. As arquiteturas MIMD podem ser divididas em dois grandes grupos: Memória Compartilhada e Memória Distribuída (falaremos nos tópicos seguintes). Com a classe MIMD, "os processadores são de uso geral; cada um é capaz de processar todas as instruções necessárias para efetuar transformação de dados apropriada" (STALLINGS, 2017, p. 524).

De acordo com Fernandez (2015), a maior parte das máquinas é híbrida com essas classes, e esse modelo clássico persiste até hoje porque é simples, fácil

de entender e dá ao aluno uma primeira visão dos sistemas paralelos.

REFLITA

O paralelismo é marcado pelo uso de diversos processadores para executar uma computação mais rapidamente. Pensando nesse cenário, reflita e responda à seguinte questão:

Dado que um único processador consegue resolver um problema em *n* segundos, *n* processadores conseguem resolver o mesmo problema em 1 segundo?

SAIBA MAIS

Saiba mais sobre o tópico ao assistir a **Arquiteturas Paralelas**: **Taxonomia de Flynn**, disponível em: *https://www.youtube.com/watch?v=LcMW1R2fYzw*. Acesso em: 18 jul. 2019.

MULTIPROCESSADOR SIMÉTRICO

Os conceitos de paralelismos podem ser praticados de diferentes maneiras, segundo a arquitetura existente e os fins específicos da arquitetura. Outra forma de classificar máquinas paralelas é pelo compartilhamento da memória, formato no qual todos os processadores têm acesso a uma memória compartilhada e no qual múltiplos processadores podem atuar independentemente. Entretanto, os processadores compartilham os mesmos recursos de memória.

Denomina-se **multiprocessador** um sistema computacional paralelo no qual todas CPUs compartilham uma memória comum. A vantagem dessa estrutura é que todos os processos podem conectar igualmente os dados compartilhados sem a necessidade de explicitar a troca de dados entre os processos. Em contrapartida, há pontos negativos: o gargalo no barramento de memória e a consistência de cache.

FIQUE ATENTO

A consistência de cache indica quando um valor escrito será devolvido por uma leitura.

A forma mais comum dessa arquitetura é o **multiprocessador simétrico** (SMP). O termo SMP faz referência a uma arquitetura de hardware computacional, bem como ao comportamento do sistema

operacional que reflete essa arquitetura. O emprego de SMP é mais popular do que se imagina e bem mais fácil de programar. Esse tipo de computador é encontrado naturalmente na maioria das organizações de hoje; além disso, vem ganhando espaço em áreas menores, à medida que os custos baixam (STALLINGS, 2017). Um SMP pode ser determinado como um sistema de computação independente com as seguintes particularidades, conforme observa Stallings (2017, p. 526):

> *1. Há dois ou mais processadores semelhantes de capacidade comparável.*
>
> *2. Esses processadores compartilham a mesma memória principal e recursos de E/S, e são interconectados por um barramento ou algum outro esquema de conexão interna, de tal forma que o tempo de acesso à memória é aproximadamente igual para cada processador.*
>
> *3. Todos os processadores compartilham acesso dispositivos de E/S, ou pelos mesmos canais ou por canais diferentes que fornecem para o mesmo dispositivo.*
>
> *4. Todos os processadores desempenham as mesmas funções (daí o termo simétrico).*
>
> *5. O sistema é controlado por um sistema operacional integrado que fornece interação entre processadores e seus programas em nível de trabalhos, tarefas, arquivos ou elementos de dados.*

No SMP, a função do sistema operacional é fazer o agendamento de *threads* através de todos os processadores. Ao contrário de uma organização uniprocessadora (máquinas SISD), uma organização SMP tem benefícios como:

> **Desempenho:** *se o trabalho a ser feito por um computador pode ser organizado de tal forma que algumas partes do trabalho podem ser feitas em paralelo, então, um sistema com vários processadores vai atingir um desempenho melhor do que um com um único processador do mesmo tipo.*
>
> **Disponibilidade:** *em um multiprocessador simétrico, como todos os processadores podem efetuar as mesmas funções, a falha de um único processador não trava a máquina. Em vez disso, o sistema pode continuar a funcionar com desempenho reduzido.*
>
> **Crescimento incremental:** *o usuário pode melhorar o desempenho de um sistema acrescentado por um adicional.*
>
> **Escalabilidade:** *fornecedores podem oferecer uma série de produtos com diferentes preços e características de desempenho com base no número de processadores configurado no sistema (STALLINGS, 2017, p. 526-527).*

É importante salientar que essas vantagens são potenciais e não garantidas. O sistema operacional precisa ser dotado de instrumentos e funções que permitam explorar o paralelismo de sistema SMP. Outro ponto interessante desse sistema é que a existência de vários processadores se apresenta de forma transparente ao usuário. Além de todas as peculiaridades, os sistemas com memória compartilhada podem ser divididos em duas categorias fundamentadas no número de acessos e no intervalo de acesso à memória: UMA e NUMA.

O Uniform Memory Access (**UMA**) significa acesso uniforme à memória; a memória utilizada nesses sistemas é concentrada e localizada à mesma distân-

cia de todos os processadores, fazendo com que a latência de acesso à memória seja distribuída uniformemente para todos os processadores do sistema. 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 (NAVAUX; DE ROSE; PILLA, 2011). Vale lembrar ainda que as máquinas com outras redes de interconexão e com memórias entrelaçadas também se enquadram nessa classe se conservarem o tempo de acesso à memória uniforme para todos os processadores do sistema.

No Non-uniform memory access (**NUMA**), que significa acesso não uniforme à memória, é empregado o conceito de que os processadores em um diretório podem alcançar a memória desse diretório com latência menor do que a memória de acesso na memória de outro processador. Isso indica que o espaço de endereçamento é exclusivo, pois cada processador pode endereçar toda a memória do sistema (FERNANDEZ, 2015).

Suas características indicam que, no ambiente paralelo, existe a ligação física entre dois ou mais sistemas SMPs e que cada máquina SMP tem acesso direto à memória das demais máquinas SMPs. Outro ponto é que o número de acessos à memória não é particionado igualmente entre todos os processadores do ambiente, além do acesso à memória de um SMP distinto ser mais lento.

Por esse motivo, tais máquinas possuem um acesso não uniforme à memória, uma vez que, a distância até a memória nem sempre é a mesma, além de estar sujeita ao endereço desejado. Enfim, devido a tudo o que foi apresentado, pode-se dizer que os multiprocessadores simétricos são muito convenientes por conta da sua boa relação custo-benefício, desde que haja memória e largura de banda suficiente.

SAIBA MAIS

Aprofunde seu conhecimento a respeito dos multiprocessadores, assistindo ao vídeo **Multiprocessadores**, disponível em: *https://www.youtube.com/watch?v=BwB8K84YRQw*. Acesso em: 18 jul. 2019.

PROCESSADORES MASSIVAMENTE PARALELOS

Em qualquer sistema de computação paralela, CPUs que atuam em partes distintas do mesmo serviço precisam se comunicar entre si para trocar informações. Diferentemente do que acontece em SMP, uma outra forma de classificar o paralelismo é pelo **sistema de memória distribuída**, cuja arquitetura "é um projeto no qual toda CPU tem sua própria memória privada, acessível somente a ela mesma e a nenhuma outra" (TANENBAUM; AUSTIN, 2013, p. 462).

Esse aspecto é denominado multicomputador; fundamentalmente, o que o distingue de multiprocessadores é que ele tem sua memória local privada, a qual só pode ser acessada pela execução de instruções LOAD e STORE. Mas que nenhuma outra CPU pode acessá-lo usando as mesmas instruções. Dessa forma, multiprocessadores têm um único espaço de endereço físico compartilhado por todas as CPUs, enquanto os multicomputadores têm um espaço de endereço físico para cada CPU.

FIQUE ATENTO

Instrução **LOAD** é uma instrução de transferência de dados da memória para o registrador (*load word*).

Instrução **STORE** é uma instrução de transferência de dados do registrador para a memória (*store word*).

Os sistemas de memória distribuída precisam de uma rede de interconexão que conecte os processadores entre si; por sua vez, o endereçamento de memória de um processador não endereça a memória de outro processador, pois cada processador atua de modo independente.

As modificações em um endereço de memória de um dado processador não são informadas para outros processadores, já que não há o conceito de coerência de cache. Assim, cada processador tem sua própria hierarquia de memória. Os pontos positivos dessa arquitetura estão na possibilidade de construir grandes computadores com o mesmo número de CPUs que os multiprocessadores, porém mais simples e mais barato. Na escalabilidade, a memória do sistema só aumenta com o aumento do número de processadores. Por outro lado há desvantagens, pois essa arquitetura é difícil de programar, e o programador é responsável por todos os pormenores associados a comunicação entre os processadores (NULL; LOBUR, 2011).

FIQUE ATENTO

O problema de coerência de cache apenas existe em arquiteturas com memória compartilhada para multiprocessadores. A estrutura de cache concebe a ideia da memória virtual em que os dados armazenados ficam mais próximos do processador do que da memória principal. Uma arquitetura multiprocessada é coerente se, exclusivamente, uma leitura de uma posição de memória executada por qualquer processador devolver o valor mais recente desse endereço.

Há vários formatos e tamanhos de multicomputadores, o que dificulta sua classificação. Entretanto, há dois estilos que se sobressaem: Massively Parallel Processors (MPP) e *clusters*.

Quanto à categoria MPP,

> *[...] são imensos supercomputadores de muitos milhões de dólares. Eles são usados em ciências, engenharia e na indústria para cálculos muito grandes, para tratar números muito grandes de transações por segundo ou para data warehousing (armazenamento e gerenciamento de imensos bancos de dados) (TANENBAUM; AUSTIN, 2013, p. 490).*

Em grande parte, os MPPs sobrepuseram máquinas SIMD, supercomputadores vetoriais e processadores matriciais do topo da cadeia alimentar digital. O destaque de MPPs é o uso que fazem de uma rede de interconexão proprietária de performance muito alta, projetada para mover mensagens de baixa latência

e alta largura de banda. Essas duas propriedades são bastante significativas, pois a maior parte das mensagens é de tamanho pequeno, mas o tráfego total é predominantemente causado por grandes mensagens.

FIQUE ATENTO

Redes de Interconexão (*Interconnection Networks*) são redes de altíssima taxa de transferência, arquitetadas para interconectar processadores e memórias em um sistema paralelo. A ordem de magnitude da largura de banda é de *Gbytes/sec*, e o retardo na casa dos *n* segundos.

Vale salientar que o que distingue os MPP é sua enorme habilidade com E/S. Problemas suficientemente grandes defendem o emprego de MPPs, uma vez que têm coleções maciças de dados a processar com bastante frequência e muitas vezes da ordem de *terabytes*. Esses dados são distribuídos entre muitos discos e movidos pela máquina a grande velocidade.

Para Tanenbaum e Austin (2013), outro ponto específico que envolve MPPs é seu cuidado com a tolerância a falhas. Em outras palavras,

> *Com milhares de CPUs, várias falhas são inevitáveis. Abortar uma execução de 18 horas porque uma CPU falhou é inaceitável, em especial quando se espera ter uma falha dessas toda semana. Assim, grandes MPPs sem ter hardware e software especiais para monitorar o sistema, detectar falhas e recuperar-se delas facilmente. (TANENBAUM e AUSTIN, 2013, p. 490).*

Em relação à forma de acesso às memórias do sistema, multicomputadores podem ser qualificados como Non-remote memory access (NORMA). O termo indica o acesso a variáveis remotas que funcionam tradicionalmente por troca de mensagens, uma vez que os processadores necessitam fazer explicitamente operações de Entrada/Saída de mensagens. Tal atributo resulta do fato de esse tipo de máquina paralela ser estabelecido a partir da replicação de toda a arquitetura convencional, e não apenas do componente processador como nos multiprocessadores (NAVAUX; DE ROSE; PILLA, 2011).

Não há muitos princípios em arquiteturas com MPP. No fim das contas, é um conjunto de nós de computação mais ou menos padronizados, integrados por uma interconexão muito rápida que mira o alto desempenho através do uso de um amplo número de processadores comerciais, os quais, em razão do fator do custo, acabam sendo processadores de baixo ou médio poder computacional.

SAIBA MAIS

Leia **Modelos Realísticos de Computação Paralela**, de Amaury Antônio de Castro Junior *et al.*, e conheça outros modelos de paralelismo.

Disponível em: *http://www.dcc.ufla.br/infocomp/index.php/INFOCOMP/article/view/43*. Acesso em: 19 jul. 2019.

FUNDAMENTOS DE CLUSTERS COMPUTACIONAIS

Outro estilo de multicomputador é o computador de *Cluster*. Segundo Fins (2001), a ciência por trás da clusterização permite a solução de vários problemas que abarcam um grande volume de processamento. O *Cluster* é um conceito que se destacou e fortaleceu recentemente, pois muitas vezes é arquitetado a partir de computadores convencionais, os quais são conectados em rede e se comunicam por meio do sistema, operando como se fossem uma única máquina de grande porte.

A distinção entre um MPP e um *Cluster* é equivalente à diferença de um *maiframe* e um PC. Ambos têm um RAM, discos, um sistema operacional e assim por diante. No entanto, os itens do *mainframe* são mais rápidos, talvez com diferenças na execução do sistema operacional. Eles são considerados díspares, utilizados e gerenciados de modo diferente, em termos qualitativos. Essa mesma diferença vale para MPPs se comparado ao *Cluster* (TANENBAUM; AUSTIN, 2013).

Todo computador pertencente a um *Cluster* recebe o nome de **nó.** Não existe limite máximo de nós, porém, independentemente do número de máquinas que o constituem, o *Cluster* deve ser "transparente", isto é, ser visto pelo usuário ou por outro sistema

que precisa deste processamento como um único computador. Assim,

> *um fator relevante para a intensificação da utilização dos Clusters é a melhoria das redes locais (LANs – Local Área Network) que proporcionam equipamentos trabalhando em altas velocidades (de 1 Gbps a 10 Gbps), juntamente com a evolução dos sistemas em nuvem (ELLER JUNIOR, 2013, p. 34).*

Podemos classificar os *Clusters* como homogêneos ou heterogêneos, de acordo com seu hardware e seus instrumentos de rede. Compreende-se por *Cluster* heterogêneos os que contêm distintas configurações em seus nós ou na rede de comunicação a que estão conectados. Em outras palavras, os nós e a rede diferenciam-se por periféricos e conexões.

Opostamente, o *Cluster* homogêneo tem hardware e rede de comunicação semelhantes em seus nós, o que o torna bem raro de ser encontrado por conta da troca ou inclusão de novos equipamentos, bem como o avanço da tecnologia. Uma preocupação em relação a *Clusters* heterogêneos é o balanceamento de carga que cada nó precisará receber.

A tendência é a divisão de forma igual de todas as tarefas pelo número de **nós**. Contudo, deve-se realizar um diagnóstico da capacidade de processamento de cada **nó** e empregar um algoritmo de balanceamento para disseminar uma carga compatível com a capacidade de cada **nó** (PITANGA, 2008). Por ventura, é importante ressaltar que não é imperativo existir um conjunto de hardware precisamente igual em cada

nó. Em contrapartida, é indispensável que todas os computadores usem o mesmo sistema operacional, assegurando que o software que controla o *Cluster* possa coordenar todas as máquinas que o integram. Também é possível classificar os *Clusters* de acordo com suas características de trabalho.

Tipos de clusters

Embora os *Clusters* não sejam a solução para todos os problemas computacionais existentes, eles auxiliam as organizações a resolverem uma boa fração dos problemas que abrangem a demanda de processamento, visto que eles tentam elevar ao máximo a utilização dos recursos existentes. Nesse sentido, o *Cluster* pode revelar-se uma solução viável, desde que o tipo mais apropriado seja selecionado. Existem alguns tipos de Cluster, porém os principais são: Cluster de alto desempenho (High Performance Computer), Cluster de Alta Disponibilidade (High Availability) e Cluster de Balanceamento de Carga (Load Balancing).

O *Cluster* de Alto Desempenho (ou *Cluster* de alta performance) comporta a ocorrência de um grande volume de processamento com alta carga de operações de ponto flutuante em computadores comuns, além de proporcionar resultados satisfatórios em tempo hábil, usando sistema operacional gratuito, o que baixa seu custo. A principal finalidade desse *Cluster* é otimizar a resolução de problemas de complexidade alta, o que permite grande poder compu-

tacional com um custo baixo se confrontado com supercomputadores (PITANGA, 2008).

Os *Clusters* de Alta Disponibilidade são fundamentados em redundância, onde há nós que podem executar as tarefas de outros nós no caso de falhas, permitindo que seus sistemas continuem ativos por um longo período e em plena condição de uso, independentemente de falhas de hardware (ELLER JUNIOR, 2013). Nesta estrutura de *Cluster*, a arquitetura deve considerar possíveis vulnerabilidades, uma vez que elas podem ocasionar falhas e, quando realizados, esses levantamentos devem avaliar as medidas de redundância, como a utilização de servidores interligados. São exemplos de *Clusters* de alta disponibilidade os servidores de banco de dados, serviços para WEB (e-mail, páginas, FTP) e *backup*.

No *Cluster* de Balanceamento de Carga, as tarefas de processamento são difundidas entre os nós mais uniformemente possível. Este tipo tem como característica a redução da latência e o aumento da capacidade de execução de tarefas (MARTINEZ, 2009). O objetivo é a distribuição apropriada de processos ao longo dos nós do agrupamento de computadores, com a ajuda de algoritmos de escalonamento (PITANGA, 2008). O balanceamento de carga pode ser aproveitado em diversos tipos de aplicações; no entanto, seu emprego é muito comum na internet, já que soluções deste tipo têm maior tolerância ao aumento instantâneo do número de requisições, exatamente por conta do equilíbrio proveniente da distribuição de tarefas.

Outro tipo de *Cluster* bem difundido e implementado em computação paralela é o *Beowulf*. Trata-se de um *Cluster* de alto desempenho que foi desenvolvido pela Nasa com base na utilização de computadores com hardware comuns e softwares livres, como o sistema operacional Linux, e demais softwares de controle e paralelização cujas licenças são *Open Source*. O objetivo era atender à crescente necessidade de capacidade de processamento utilizado nas várias áreas do conhecimento, sobretudo na área científica, com vistas a implementar sistemas computacionais com um baixo custo e cada vez mais poderosos (ELLER JUNIOR, 2013).

FIQUE ATENTO

Open Source é um programa com código-fonte aberto. Ele permite que o usuário possa adaptá-lo de acordo com suas necessidades.

SAIBA MAIS

Para complementar este tópico, assista a **O que é e como funciona um CLUSTER (Computador)**, que está disponível em *https://www.youtube.com/watch?v=WEsdVMdS5Mo*. Acesso em: 18 jul. 2019.

O nome *Beowulf* tem como origem um poema em Língua Inglesa, que descreve atos de um herói dos Gautas, o qual libertou os dinamarqueses de dois

monstros porque era dono de grande força e coragem. Basicamente, um *Cluster Beowulf* é um conjunto de computadores cujas características fundamentais são:

> *Nós compostos por hardwares comuns.*
>
> *Todos os nós do agregado e a rede de interconexão utilizados exclusivamente para as funções do cluster.*
>
> *Utilização do cluster somente para realização de computação de alto desempenho.*
>
> *Uso de softwares livres em todos os nós (TOLOUEI, 2010, Parte 3, p. 2).*

Com estas propriedades, os *Clusters Beowulf* foram bastante disseminados, por apresentarem um custo-benefício muito bom e atender às demandas por computação paralela.

Funcionamento básico dos clusters

Para que um *Cluster* seja estruturado, é indispensável usar determinados elementos. O primeiro já foi apresentado: os equipamentos a serem usados como *nós*. Desse modo, podem-se empregar computadores arquitetados especificamente para trabalhar como nós. Diante disso, as máquinas devem ter apenas componentes de hardware necessários ao C*luster*. Sob essa ótica, também há a possibilidade de utilização de desktops para fins domésticos ou para uso em escritório. Por exemplo, uma universidade ou empresa pode utilizar computadores que foram trocados por modelos mais atuais para criar

um *Cluster* e, por ventura, economizar com a compra de servidores (ALECRIM, 2013).

Os nós de um *Cluster* podem ser dedicados ou não dedicados. No primeiro caso, o nó é empregado apenas para este intuito, fazendo com que dispositivos como teclados e monitores sejam supérfluos. Se, por alguma razão, for imprescindível acessar um computador em particular, pode-se fazê-lo via terminal, por exemplo, a partir do nó principal. No segundo caso, cada máquina que faz parte do *Cluster* não opera exclusivamente nele.

Outro ponto importante é o sistema operacional. Conforme já mencionado, os nós não necessitam ser precisamente iguais no que se refere ao hardware. Entretanto, é essencial que as máquinas usem o mesmo sistema operacional. Essa conformidade é crucial para abreviar a complexidade de configuração e manutenção do sistema e para certificar-se de que os processos rotineiros ao *Cluster* sejam executados constantemente, como monitorização, distribuição de tarefas e controle de recursos. Para reforçar tais atributos, podemos até mesmo considerar os sistemas operacionais dispostos especialmente para *Clustering* (PITANGA, 2008).

Analisando sob a perspectiva do software, o *Cluster* tem um artifício que leva em conta o papel de um *middleware*, que é um sistema que possibilita o controle do *Cluster* e está fortemente ligado ao sistema operacional, funcionando como uma camada de abs-

tração ou mediação entre diferentes sistemas de aplicações (ELLER JUNIOR, 2013).

É o *middleware* que lida, por exemplo, com as bibliotecas que realizam a comunicação do *Cluster*, sendo que uma delas é o padrão Message Passing Interface (MPI), além de atuar como administrador do *Cluster*. Por padrão, "o *middleware* é instalado em uma máquina chamada de nó controlador (ou nó mestre)" (ALECIM, 2013).

FIQUE ATENTO

MPI significa Interface de Passagem de Mensagem e trata da comunicação entre nós. Os processos do MPI são executados paralelamente e possuem um espaço de endereçamento independente.

A denominação deixa evidente que se trata do já citado nó principal, que controla o *Cluster*, efetivamente, através da distribuição de tarefas, do monitoramento e de procedimentos relacionados.

Por fim, a comunicação entre os nós é realizada a partir de uma tecnologia de rede local. Neste caso, o padrão Ethernet é muito usado exatamente por ser mais comum e, por conseguinte, mais bem suportado e menos custoso.

FIQUE ATENTO

O padrão Ethernet é um modelo de conexão para redes locais (LAN) fundamentado no envio de pacotes. O padrão determina como se transmitem os dados por meio dos cabos da rede.

Vantagens e desvantagens dos clusters

Agora que você já sabe o que é um *Cluster*, como funciona e quais tipos existem, podemos focar nas vantagens e desvantagens que a clusterização traz. Comecemos pelas desvantagens.

Como toda tecnologia, *Clusters* de computadores também têm algumas desvantagens que o arquiteto ou projetista necessita fazer o levantamento dos prós e contras quando for implantar um sistema nesse formato, por exemplo:

> **Manutenção de equipamento:** *Por o cluster ser facilmente expansível, o sistema computacional pode se tornar muito grande, e a manutenção do sistema pode se tornar uma tarefa imensamente grande, pois cada máquina em um clusters deve ter todos os seus componentes em perfeito estado de funcionamento.*
>
> **Monitoração dos nós:** *Monitorar as informações trocadas em cada nó pode ser um problema dependendo como foi configurado o cluster, levando em consideração a expansibilidade do cluster.*
>
> **Gargalos de troca de informações:** *Como a comunicação de clusters de computadores ocorre por uma tecno-*

> *logia de rede, a troca de informação se transforma no principal gargalo, uma vez que a transmissão de rede é bem lenta se comparada à troca de informação com um barramento de um sistema de memória compartilhada, entretanto, é possível realizar ajustes de granulosidade para diminuir esse problema (BACELLAR, 2010, p. 4).*

Em contraponto, a implantação de *Clusters* pode trazer vários benefícios, e suas principais vantagens são:

> **Alto Desempenho:** *possibilidade da utilização da paralelização de programas complexos que diminui o tempo de processamento e resolução do problema.*
>
> **Escalabilidade:** *possibilidade de que novos componentes sejam à medida que cresce a carga de trabalho.*
>
> **Tolerância a falhas:** *aumento da confiabilidade do sistema como um todo, caso alguma parte falhe.*
>
> **Baixo custo:** *redução do custo para se obter processamento de alto desempenho utilizando PCs simples.*
>
> **Independência de fornecedores:** *utilização de hardware aberto, software de uso livre e independência de fabricantes e licenças de uso (PITANGA, 2008, p. 33).*

Note que, com essas propriedades, é possível ter um C*luster* computacional poderoso, visto que atualmente a grande demanda por alta capacidade de processamento deixou de ser um luxo para se tornar algo cada vez mais comum em vários ambientes computacionais. Nesse sentido, os *Clusters* se tornaram uma alternativa muito eficaz e economicamente viável, devido ao barateamento e à evolução dos computadores pessoais e das redes de interconexão ao longo dos anos.

SAIBA MAIS

Para saber mais, leia **Ambientes de Clusters e Grids Computacionais: Características, Facilidades e Desafios,** de Taís Appel Colvero, Mar Dantas e Daniel Pezi da Cunha. Disponível em: *http://periodicos.unesc.net/sulcomp/article/view/798*. Acesso em: 27 jul. 2019.

Acesse o Podcast 2 em Módulos

CONSIDERAÇÕES FINAIS

A computação de alto desempenho motiva arquitetos a explorar técnicas de paralelização com a intenção de obter execuções cada vez mais velozes. Um dos desafios do paralelismo se refere à maneira como os processos se comunicam. Arquiteturas paralelas têm como intuito auxiliar a programação e diminuir custos, para conseguir uma melhor performance em menor tempo o resultado para algum problema, na área computacional, de grande complexidade.

Nesse sentido, inicialmente expomos os conceitos de arquiteturas paralelas. Estudamos que o processamento paralelo, de modo simples, é o emprego simultâneo de vários recursos computacionais para resolver um problema; aprendemos que existem dois conceitos distintos relacionados ao paralelismo, que são **em nível de instrução** e **de máquina**. Em seguida, conhecemos a Classificação de Flynn e as quatro classes de arquitetura de computadores: **SISD**, **SIMD**, **MISD** e **MIMD**, baseadas nos fluxos de instruções e dados.

Saímos do cenário acima abordando **SMP** e **MPP**, sendo o primeiro um multiprocessador e o segundo um multicomputador. Estudamos a diferença fundamental entre os dois, que é caracterizada pela presença ou pela ausência de memória compartilhada. Ainda, foram examinados os tipos de máquinas inerentes a cada de arquitetura: **UMA**, **NUMA** e **NORMA**.

No último tópico, abordamos os Fundamentos de *Clusters* Computacionais. O intuito é compreender que eles são compostos por máquinas comuns, denominadas nós e que atuam, em geral, dedicadamente porque é para executar no menor tempo possível as especificas aplicações em um ambiente computacional com ampla demanda de processamento.

SÍNTESE

SISTEMAS COMPUTACIONAIS

Este módulo aprofundou os temas relacionados à Arquitetura e Organização de Computadores, mais especificamente em Arquiteturas de Alto Desempenho. Primeiro, foram abordados os conceitos básicos de arquiteturas paralelas e mostrado que o paralelismo pode ser dividido **em nível de instrução e de máquina**. E ainda que a aplicação do paralelismo está presente nos vários níveis de um sistema. Em seguida, apresentou-se a **Classificação de Flynn**, que é baseada em fluxos de instruções e dados e divide em quatro classes de arquiteturas de computadores: **SISD**, **SIMD**, **MISD** e **MIMD**.

Nos tópicos seguintes, foram discutidas as arquiteturas SMP e MPP. A arquitetura **SMP** (Multiprocessador Simétrico) surge como uma forma diferente de classificar arquiteturas paralelas cuja característica principal é o **compartilhamento da memória.**

Esses sistemas podem ser divididos em duas categorias fundamentadas no número de acessos e no intervalo de acesso à memória: **UMA** e **NUMA**. Um sistema computacional paralelo no qual todas as CPUs compartilham uma memória comum é denominado **multiprocessador**.

A arquitetura **MPP** (Processadores Massivamente Paralelos) também é uma forma de computação paralela e é caracterizada pelo **sistema de memória distribuída**. Esse aspecto é denominado **multicomputador** e, em relação à forma de acesso às memórias do sistema, os multicomputadores podem ser qualificados como **NORMA**. Mostramos que a diferença fundamental entre os dois modelos é caracterizada pela presença ou a ausência de memória compartilhada.

Por fim, estudamos os **Fundamentos de Clusters Computacionais**, bem como os seus conceitos básicos e, em seguida, os tipos principais: ***Cluster* de alto desempenho, *Cluster* de Alta Disponibilidade** e o ***Cluster* de Balanceamento de Carga**, logo depois o ***Cluster* Beowulf.**, apresentando o funcionamento básico dos *Clusters* e depois suas vantagens e desvantagens.

Referências Bibliográficas & Consultadas

ALECRIM, E. **Cluster**: conceito e características. *In:* INFOWESTER. 2013. Disponível em: *https://www.infowester.com/cluster.php*. Acesso em: 21 jul. 2019.

BACELLAR, H. V. **Cluster:** Computação de Alto Desempenho. Instituto de Computação da Universidade Estadual de Campinas. Campinas, 2010. Disponível em: *http://www.ic.unicamp.br/~ducatte/mo401/1s2010/T2/107077-t2.pdf*. Acesso em: 26 jul. 2019.

CÓRDOVA JUNIOR, R. S.; SANTOS, S. C. B.; KISLANSKY, P. **Fundamentos computacionais**. Porto Alegre: SAGAH, 2018.

CORRÊA, A. G. D. **Arquitetura e Organização de computadores**. São Paulo: Pearson Education do Brasil, 2016. [Biblioteca Virtual]

CRISTO, E. F.; PREUSS, E.; FRANCISCATTO, F. **Arquitetura de computadores.** Universidade Federal de Santa Maria, Colégio Agrícola de Frederico Westphalen, 2013.

DELGADO, J.; RIBEIRO, C. **Arquitetura de computadores**. 5. ed. Rio de Janeiro: LTC, 2017. [Minha Biblioteca]

ELLER JUNIOR, E. **Estudo de tecnologias para computação paralela e distribuída:** implementação de um Cluster Beowulf. Dissertação (Mestrado em Modelagem Computacional em Ciência e Tecnologia) 163f. Universidade Federal Fluminense. Volta Redonda, 2013.

FERNANDEZ, M. P. **Informática: Arquitetura de Computadores**. 3. ed. Fortaleza: Editora UECE, 2015. Disponível em: *http://www.uece.br/computacaoead/index.php/downloads/doc_download/2123--arquiteturadecomputadores*. Acesso em: 08 jul. 2019.

FINS, B. **O que é cluster e como funciona?** Disponível em: *http://faqinformatica.com/o-que-e--cluster-e-como-funciona/*. Acesso em: 20 jul. 2019.

MARTINEZ, I. F. **Creacion y validacion de um cluster de computacion cientifica baseado em rocks**. Escuela Politecnica Superior, Leganes, Universidad Carlos III de Madrid, 2009. Disponível em: *https://core.ac.uk/download/pdf/29400187.pdf*. Acesso em: 20 jul. 2019.

NAVAUX, P. O. A; DE ROSE, C. A. F; PILLA, L. L. Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído. *In*: ERAD.

Escola Regional de Alto Desempenho (1.: 22-25, mar. 2011: Porto Alegre). **Anais**... Porto Alegre: SBC, 2011.p. 22-59.

NULL, L; LOBUR. J. **Princípios Básicos de Arquitetura e Organização de Computadores.** 2. ed. Porto Alegre: Bookman, 2011.

PITANGA, M. **Construindo Supercomputadores com Linux**. 3. ed. Rio de Janeiro: Brasport, 2008.

STALLINGS, W. **Arquitetura e organização de computadores**. 10. ed. São Paulo: Pearson Education do Brasil, 2017.

TANENBAUM, A. S.; AUSTIN, T. **Organização estruturada de computadores**. 6.ed. São Paulo: Pearson Prentice Hall, 2013.

TOCCI, R. J.; WIDMER, N. S.; MOSS, G. L. **Sistemas digitais**: princípios e aplicações. 10. ed. São Paulo: Pearson Prentice Hall, 2007. [Biblioteca Virtual].

TOLOUEI, D. V. L. **Clusters computacionais de alto desempenho**. Monografia (Especialização em redes de computadores) - Escola Superior Aberta do Brasil. Vila Velha, 2010. Disponível em: *https://www.monografias.com/pt/trabalhos3/clusters-computacionais-alto-desempenho/clusters-computacionais-alto-desempenho.shtml*. Acesso em: 21 jul. 2019.


fam
ONLINE