INSTITUTO FEDERAL
RIO GRANDE DO SUL

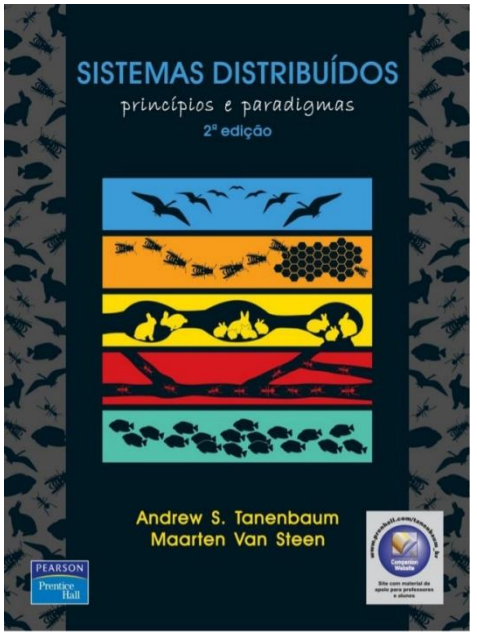
Programação com Objetos Distribuídos

- Programação Paralela

Março/ 2022

Bibliografia

- TANENBAUM, Andrew S.; STEEN, Maarten Van. Sistemas distribuídos: princípios e paradigmas. São Paulo: Pearson Prentice Hall, 2010. x, ISBN 9788576051428.
- (Capítulo 03 – processos (threads))



• Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído ERAD 2011 22-25 de março de 2011 ISBN 2177-0085
Philippe O. A. Navaux (Universidade Federal do Rio Grande do Sul), César A. F. De Rose (PUCRS), Laércio L. Pilla (Universidade Federal do Rio Grande do Sul)

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

- Introdução
  - O aumento na capacidade de processamento dos computadores só podia ser obtido através de processadores mais velozes ou através do aumento do número de processadores empregados em conjunto.
  - O aumento da velocidade dos processadores esbarrava no custo e no limite da capacidade tecnológica para obtenção de circuitos rápidos.
  - A solução tendeu para o emprego de vários processadores trabalhando em conjunto na obtenção de uma maior capacidade de processamento.
  - A área é conhecida também como Processamento de Alto Desempenho - PAD (em inglês, High Performance Computing - HPC).

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

- Máquinas para Processamento de Alto Desempenho
  - Previsão do tempo
  - Procura de petróleo
  - Simulações físicas
  - Matemática computacional
  - Inteligência Artificial
  - Aprendizagem de Máquina

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

- Máquinas para Processamento de Alto Desempenho
  - Previsão do tempo
  - Procura de petróleo
  - Simulações físicas
  - Matemática computacional
  - Inteligência Artificial
  - Aprendizagem de Máquina

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

Tipos de Concorrência numa Arquitetura

- **Concorrência temporal,** que resulta nas arquiteturas pipeline, onde existe uma sobreposição na execução temporal dos vários estágios que compõem uma instrução;
- **Concorrência de recursos síncrona**, que resulta nas arquiteturas SIMD, também conhecidas como arquiteturas matriciais (array), nas quais a concorrência existe entre elementos de processamento que executam em paralelo, de forma simultânea, a mesma operação;
- **Concorrência de recursos assíncrona**, que resulta nas arquiteturas MIMD, em que processadores atuam em paralelo para resolver uma tarefa, porém cada um executando dentro do seu ordenamento e tempo

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

Processamento Paralelo (PP) Definição: várias unidades ativas colaborando na resolução de um mesmo problema. As várias unidades ativas cooperam para resolver o mesmo problema, atacando cada uma delas uma parte do trabalho e se comunicando **para a troca de resultados intermediários** ou no mínimo para a divisão inicial do trabalho e para a junção final dos resultados.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

Motivação para o uso de processamento paralelo

Desempenho:
- reduzir o tempo de execução devido a utilização de diversas unidades ativas na resolução do problema.

Tolerância a falhas:
- reduzir a probabilidade de falhas no cálculo pois cada unidade ativa calcula o mesmo problema e faço uma eleição no final.

Modelagem:
- reduzir a complexidade da modelagem e como resultado da implementação da aplicação utilizar uma linguagem que expresse paralelismo (em situações onde o problema é em sua essência paralelo).

Aproveitamento de recursos:
- aproveitar melhor os recursos disponíveis na rede executando uma aplicação com múltiplos processos.

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

Computação paralela é uma forma de computação em que vários instruções são realizados simultaneamente.

Várias unidades ativas colaborando na resolução de um mesmo problema.

Este conceito opera sob o princípio de que grandes problemas geralmente podem ser divididos em problemas menores, que então são resolvidos concorrentemente (em paralelo).

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

- Tradicionalmente, o software tem sido escrito para ser executado sequencialmente (procedural).

- Para resolver um problema, um algoritmo é construído e implementado como um fluxo serial de instruções.

- Tais instruções são então executadas por uma unidade central de processamento de um computador.

- Somente uma instrução pode ser executada por vez; após sua execução, a próxima então é executada.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

• A computação paralela faz uso de múltiplos elementos de processamento simultaneamente para resolver um problema.

• Isso é possível ao quebrar um problema em partes independentes de forma que cada elemento de processamento pode executar sua parte do algoritmo simultaneamente com outros.

• Os elementos de processamento podem ser diversos e incluir recursos como um único computador com múltiplos processadores, diversos computadores em rede, hardware especializado ou qualquer combinação dos anteriores

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

- Teoricamente, o aumento de velocidade com o paralelismo deveria ser linear, de forma que dobrando a quantidade de elementos de processamento, reduz-se pela metade o tempo de execução.

- Entretanto, são poucos algoritmos paralelos atingem essa situação ideal.

- A maioria deles possui aumento de velocidade quase linear para poucos elementos de processamento, tendendo a um valor constante para uma grande quantidade de elementos.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Como Paralelizar

- Uma aplicação escrita em C ou Java com várias *threads.* (cap. 03 – pag 41)

- Uma aplicação escrita em Java usando RMI (*Remote Method Invocation).*

- Uma aplicação escrita em C que foi quebrada em vários processos que se comunicam por *sockets.*

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Como Paralelizar

Um programa que não foi **preparado para executar com várias unidades ativas** (implementado com apenas um processo que não dispara múltiplas *threads)* ***não*** executa mais rápido em uma máquina dual!!!

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Como Paralelizar

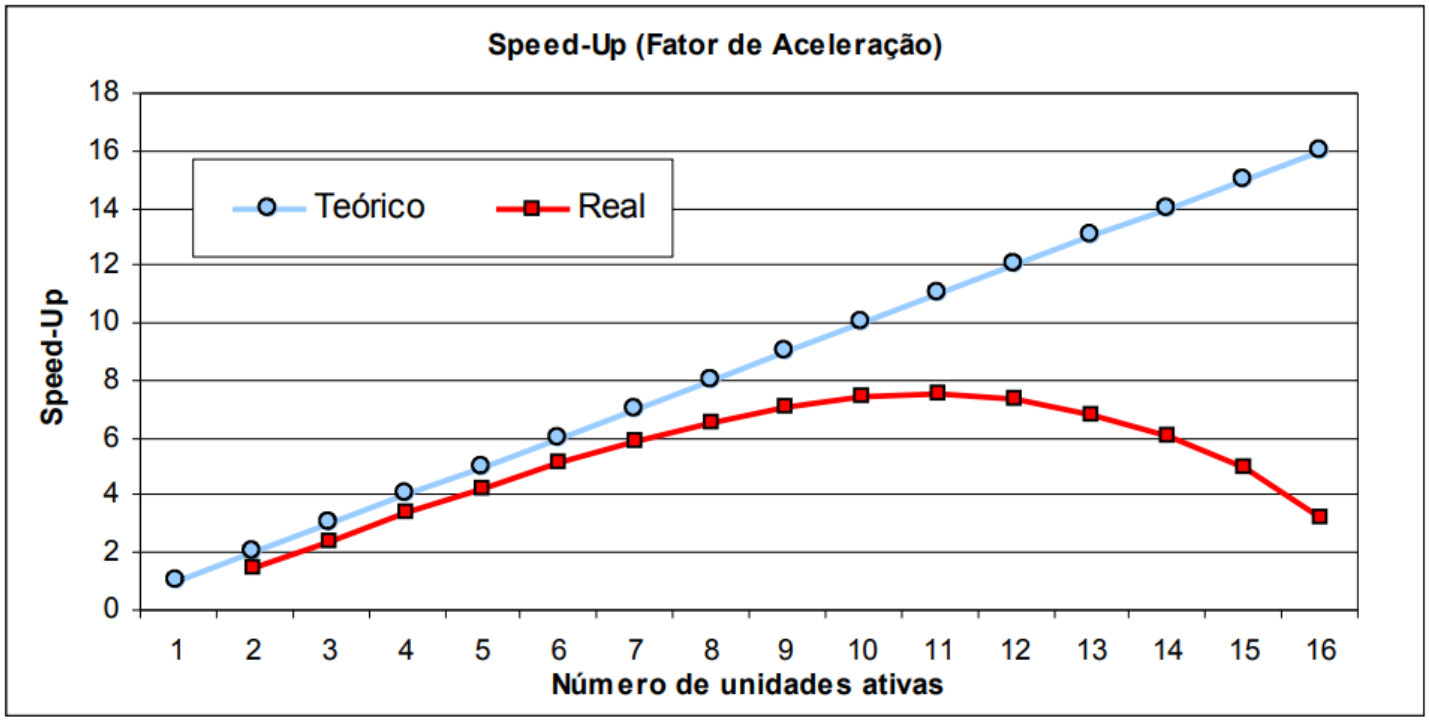


Figura 2. Comparação entre fator de aceleração teórico e obtido.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Questões de Desempenho

Desempenho da aplicação

- quantas vezes o programa paralelo ficou mais rápido que a versão sequencial.

Desempenho da rede de interconexão

- É o tempo necessário para enviar uma mensagem através da rede de interconexão.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Níveis de Paralelismo

Grão de Paralelismo: é um conceito muito importante pois seu entendimento é fundamental para a modelagem de programas paralelo.

Níveis de Paralelismo

Grão grosso:

- O trabalho a ser feito pode ser particionado em unidades de trabalho grandes. (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 médio. (Em caso de um alto custo de comunicação)

Grão fino:

- O trabalho a ser feito só pode ser particionado em unidades de trabalho pequenas. (o custo do envio não é compensado pelo ganho de tempo em atacar o problema com mais unidades).

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Níveis de Paralelismo

Aplicação	• Utilização de processos leves (*threads)* • Invocação de métodos remotos (Java RMI) • Utilização de processos remotos
Sistema Operacional	• Multiprocessamento • Multiprogramação
Arquitetura	• Vários processadores • Unidades de E/S ativa • Hierarquia de memória • Hierarquia de barramentos
Processador	• Múltiplas unidades funcionais • Pipeline de instrução

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Processamento Paralelo x Distribuído

Processamento Paralelo

- **a motivação foi o ganho de desempenho e** as unidades ativas estão normalmente na mesma máquina resultando em custos de comunicação menores (baixa latência).

Processamento Distribuído

- **a motivação foi a modelagem e o** aproveitamento de recursos e as unidades ativas estão normalmente mais afastadas (em uma rede local ou até na Internet) resultando em custos de comunicação maiores (alta latência).

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

- Paralelismo na Arquitetura de Chips
  - **Arquiteturas empregando a Técnica Pipeline**
    - uma tarefa é dividida em subtarefas que são executadas cada uma por um dos estágios, de forma a ter na saída da cadeia do pipeline a tarefa completa executada
  - **Arquiteturas Superescalares**
    - Um processador superescalar típico busca e decodifica várias instruções por ciclo, enquanto as arquiteturas escalares buscam e executam apenas uma instrução por ciclo
  - **Arquiteturas Multithread**
    - permite o processamento de mais de um fluxo de instruções em paralelo pela arquitetura do processador.
  - **Arquiteturas Multi-core**
    - divisão das tarefas em operações concorrentes e distribuídas entre várias unidades de processamento, conhecidas como cores ou núcleos.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Acesso a Memória

- Em um **sistema paralelo**, 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. Daí o nome **múltiplos processadores.**

- Em relação ao tipo de acesso às memórias do sistema, multiprocessadores podem ser classificados como:
  - UMA
  - NUMA
  - COMA

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Acesso a Memória - *Multiprocessadores*

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, fazendo com que a latência de acesso à memória seja igual para todos os processadores do sistema (uniforme).

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.

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. Essas caches têm* muito mais capacidade que uma *cache tradicional.*

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Acesso a Memória - *Multicomputadores*

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.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

Classificação de Flynn: (fluxo de instruções / fluxo de dados)

- Pode ser usada a classificação genérica de Flynn.
- Origem em meados dos anos 70, é ainda válida e muito difundida.
- Baseando-se no fato de um computador executar uma sequência de instruções sobre uma sequência de dados, diferenciam-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:

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Fundamentos das Arquiteturas para Processamento Paralelo e Distribuído

	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)

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Introdução ao Processamento Paralelo e ao Uso de Clusters de Workstacions em Sistemas GNU/LINUX

O termo **cluster de workstacions,** refere-se a utilização de diversos computadores conectados em rede para realização de processamento paralelo

O objetivo de um cluster de workstatcions é possibilitar o uso de **computadores ligados em rede** para execução de processamento com alto desempenho, permitindo a realização de simulações avançadas.

O processamento paralelo consiste em dividir uma tarefa em suas partes independentes e na execução de cada uma destas partes em diferentes processadores

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Introdução ao Processamento Paralelo e ao Uso de Clusters de Workstacions em Sistemas GNU/LINUX

Threads
são múltiplos caminhos de execução que rodam concorrentemente na memória compartilhada e que compartilham os mesmos recursos e sinais do processo pai. Uma thread é um processo simplificado, mais leve ou “light”, custa pouco para o sistema operacional, sendo fácil de criar, manter e gerenciar. O padrão de implementação de threads é o POSIX 1003.1c threads standard.

MPI (Message Passing Interface)
É um método que inclui conceitos novos como rank (cada processo tem uma identificação única, crescente), group (conjunto ordenado de processos) e communicator (uma coleção de grupos), que permitem um gerenciamento mais complexo (e inteligente) do uso de cada máquina do cluster.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Introdução ao Processamento Paralelo e ao Uso de Clusters de Workstations em Sistemas GNU/LINUX Parte II: Processos e Threads

Processos

Os computadores e os sistemas operacionais modernos suportam multitasking (múltiplos processos sendo executados simultaneamente).

Se o computador tem mais de um processador, o sistema operacional divide os processos entre os processadores.

No GNU/Linux e nas variantes do Unix, um processo pode ser clonado com a função fork().

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Introdução ao Processamento Paralelo e ao Uso de Clusters de Workstations em Sistemas GNU/LINUX Parte II: Processos e Threads

Um processo pode estar num dos seguintes estados:

• idle/new:Processo recém criado (sendo preparado).

• ready: Processo sendo carregado.

• standby: Cada processador tem um processo em standby (o próximo a ser executado).

• running: O processo esta rodando (ativo).

• blocked: Esperando evento externo (entrada do usuário) (inativo).

suspended-blocked: Processo suspenso.

• zoombied: O processo acabou mas não informou seu pai.

• done-terminated: O processo foi encerrado e os recursos liberados.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Introdução ao Processamento Paralelo e ao Uso de Clusters de Workstations em Sistemas GNU/LINUX Parte II: Processos e Threads

Processos sincronizados e não sincronizados:

Quando o processo pai espera o encerramento do processo filho, ele tem uma execução sincronizada. Quando os processos rodam independentemente eles são não sincronizados.

A comunicação entre processos pode ser realizada utilizando-se variáveis de ambiente, pipes, ou memória compartilhada (shared memory).

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Introdução ao Processamento Paralelo e ao Uso de Clusters de Workstations em Sistemas GNU/LINUX Parte II: Processos e Threads

O que são threads ?
Threads são múltiplos caminhos de execução que rodam concorrentemente na memória compartilhada e que compartilham os mesmos recursos e sinais do processo pai. Uma thread é um processo simplificado, mais leve ou "light", custa pouco para o sistema operacional, sendo fácil de criar, manter e gerenciar.

Porque usar threads ? Em poucas palavras é o pacote definitivo para o desenvolvimento de programação em larga escala. O padrão é o POSIX 1003.1C, o mesmo é compatível com outros sistemas operacionais como o Windows.

Casos em que o uso de threads é interessante: Para salvar arquivos em disco. Quando a interface gráfica é pesada. Quando existem comunicações pela internet. Quando existem cálculos pesados.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Introdução ao Processamento Paralelo e ao Uso de Clusters de Workstations em Sistemas GNU/LINUX Parte II: Processos e Threads

O que são threads ?
Threads são múltiplos caminhos de execução que rodam concorrentemente na memória compartilhada e que compartilham os mesmos recursos e sinais do processo pai. Uma thread é um processo simplificado, mais leve ou "light", custa pouco para o sistema operacional, sendo fácil de criar, manter e gerenciar.

Porque usar threads ? Em poucas palavras é o pacote definitivo para o desenvolvimento de programação em larga escala. O padrão é o POSIX 1003.1C, o mesmo é compatível com outros sistemas operacionais como o Windows.

Casos em que o uso de threads é interessante: Para salvar arquivos em disco. Quando a interface gráfica é pesada. Quando existem comunicações pela internet. Quando existem cálculos pesados.

INSTITUTO FEDERAL
RIO GRANDE DO SUL

Introdução ao Processamento Paralelo e ao Uso de Clusters de Workstations em Sistemas GNU/LINUX Parte II: Processos e Threads

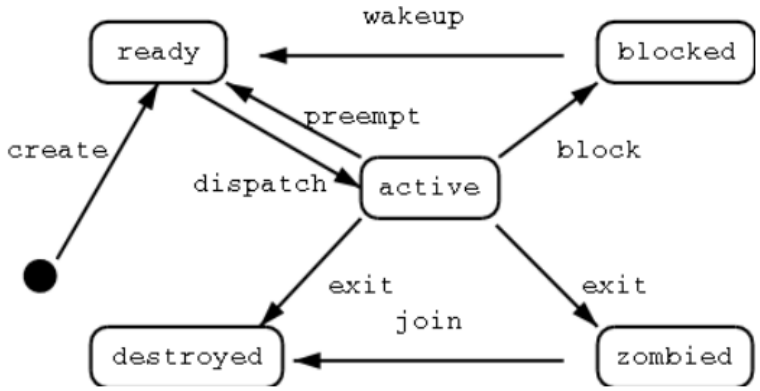
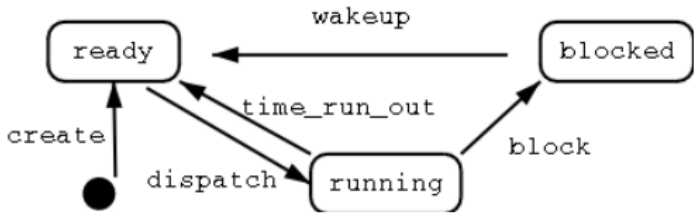


Figura 3: Estados de uma thread.



INSTITUTO FEDERAL
RIO GRANDE DO SUL

Introdução ao Processamento Paralelo e ao Uso de Clusters de Workstations em Sistemas GNU/LINUX Parte II: Processos e Threads

Existem diferentes categorias de sincronização, dentre as quais pode-se destacar:

Sincronização de dados: Threads concorrendo no acesso de uma variável compartilhada ou a arquivos compartilhados.

Sincronização de hardware: Acesso compartilhado a tela, impressora.

Sincronização de tarefas: Condições lógicas