COMUNICAÇÃO POR MENSAGENS - MPI

DCE540 - Computação Paralela e Distribuída

Atualizado em: 25 de novembro de 2021

Iago Carvalho

Departamento de Ciência da Computação

MPI

Message Passing Interface (MPI)

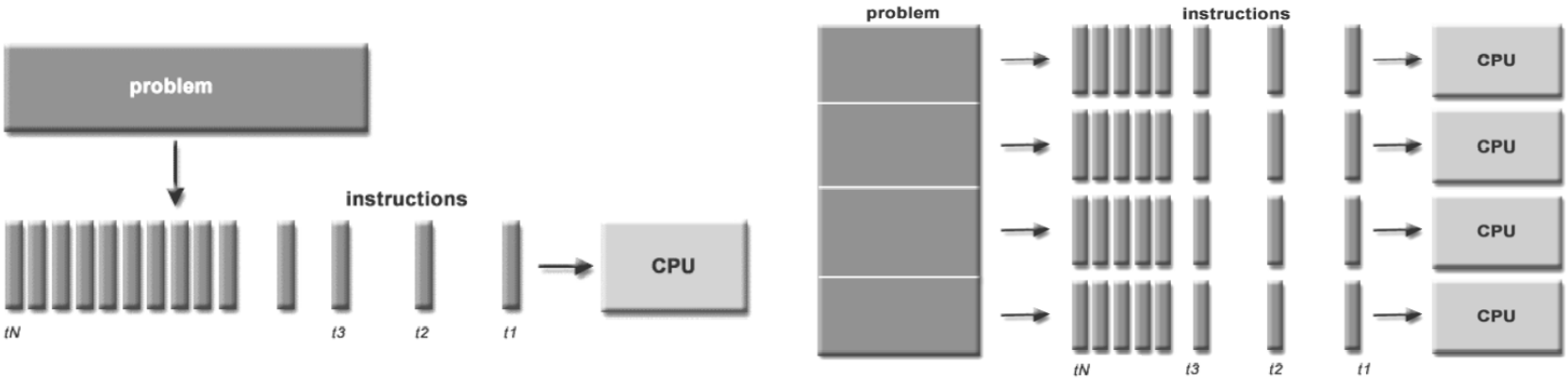
Troca de mensagens em clusters, servidores e dispositivos de computação de alta performance

- Uma forma simples de fazer computação paralela
- Diferentes formas de *buffering* e sincronização

Comunicação transiente

- Implementações síncrona e assíncrona

PROCESSAMENTO PARALELO



MPI

MPI não é necessariamente um protocolo de transmissão

Também não é iniciativa de nenhum orgão regulador

- IEEE, ISO, . . .

O MPI nasceu a partir de uma discussão em uma conferência científica em 1991

- Se tornou o padrão para comunicação entre processos

MPI

MPI define três coisas

1. Sintaxe
2. Semântica
3. Métodos

Sendo assim, existem diversas diferentes implementações de MPI

- Diversas linguagens de programação

OPERAÇÕES PADRÃO DE MPI

Existem 7 operações básicas para se trabalhar com MPI

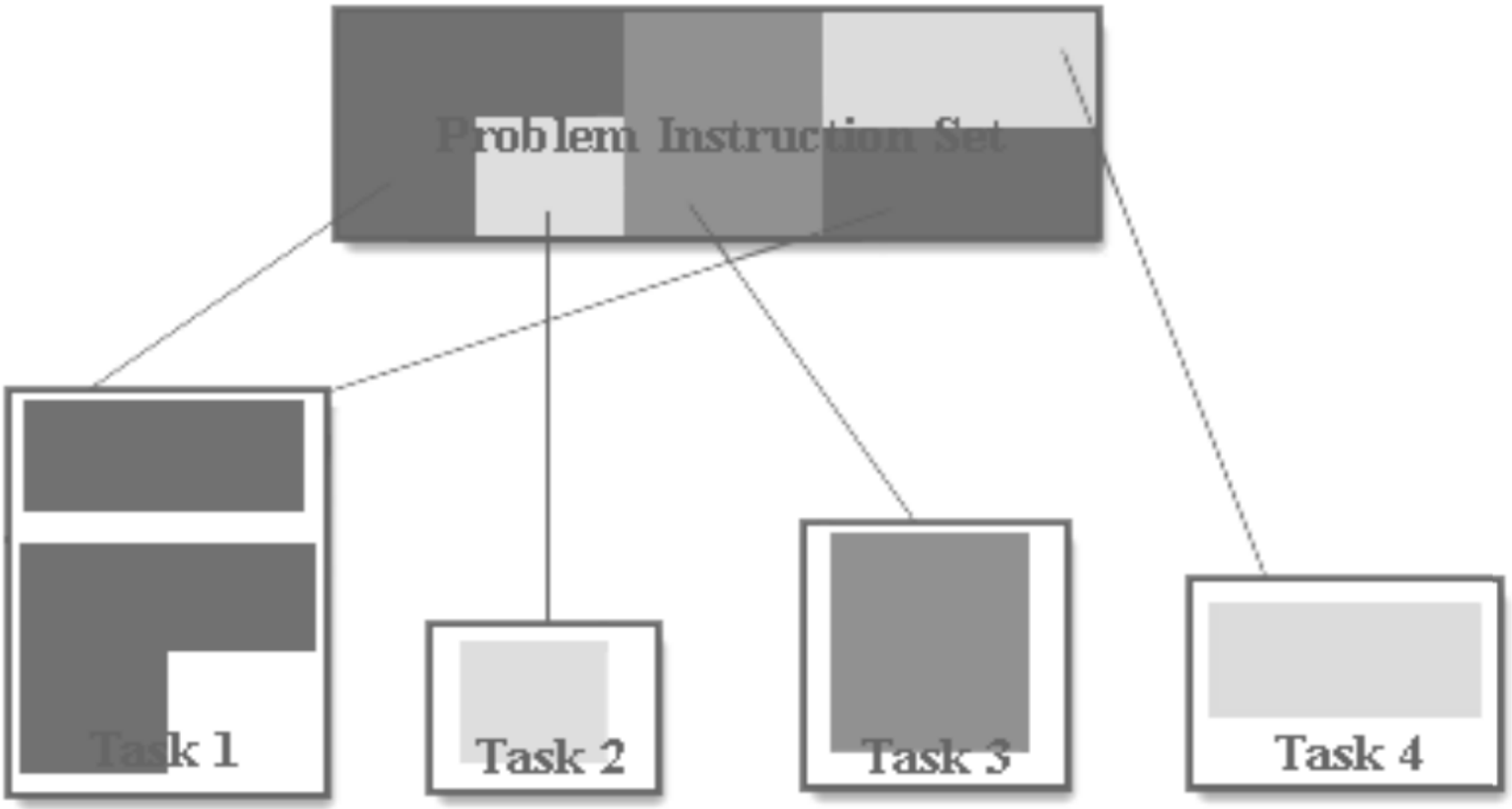
1. MPI_send ← envia uma mensagem
2. MPI_ssend ← envia uma mensagem e espera o início da transmissão
3. MPI_bsend ← adiciona uma mensagem para um buffer
4. MPI_isend ← envia uma referência para uma mensagem
5. MPI_sendrecv ← envia uma mensagem e espera a resposta
6. MPI_recv ← recebe uma mensagem; bloqueia caso não exista nenhuma
7. MPI_irecv ← recebe uma referência para uma mensagem

Entretanto, existem diversos métodos avançados ▸ Link

- Mais de 440 métodos (MPI 3.0)

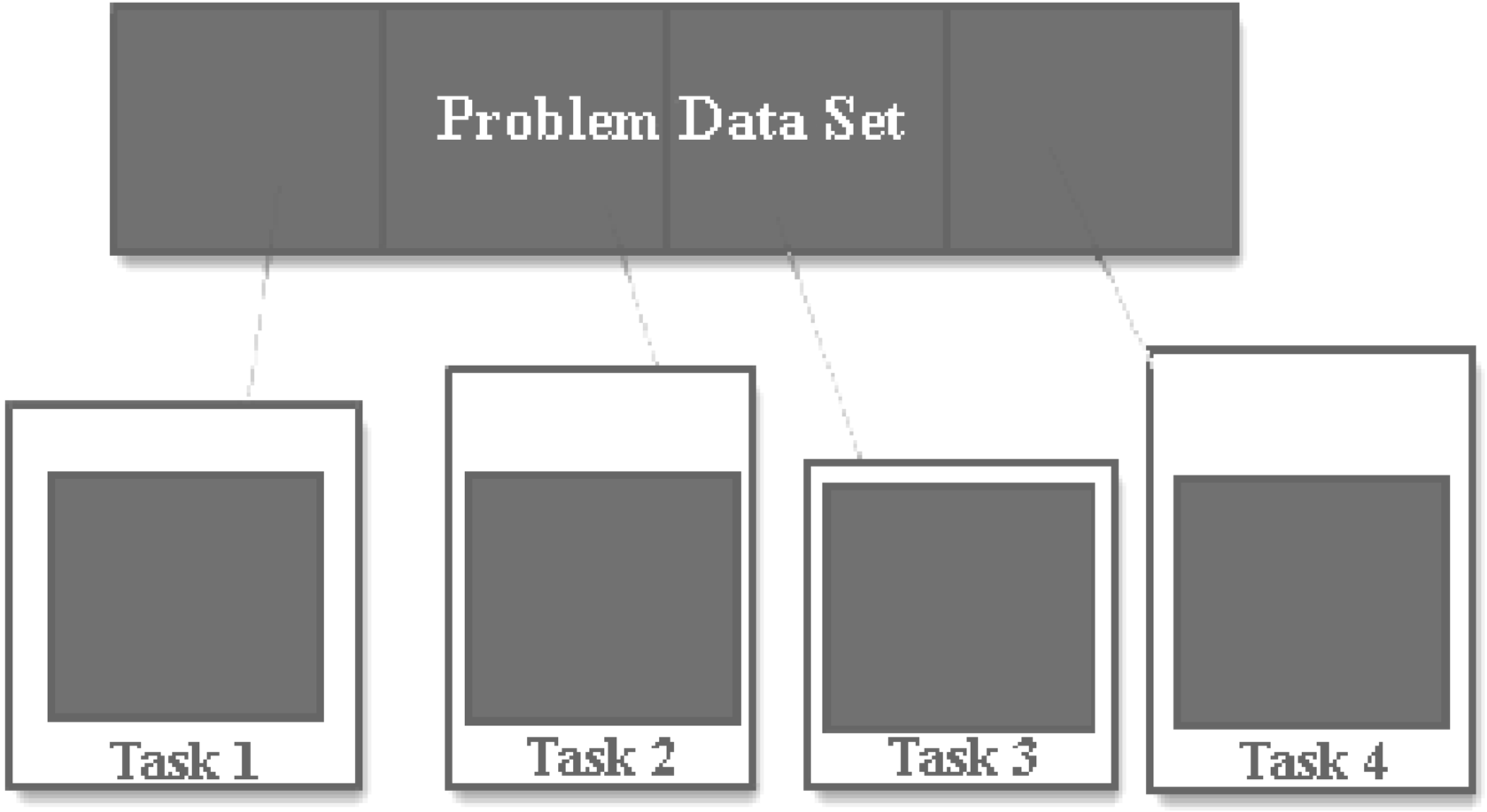
DECOMPOSIÇÃO FUNCIONAL

O problema é decomposto em diferentes tarefas, gerando diversos programas, que serão distribuídos por entre múltiplos processadores para execução simultânea



DECOMPOSIÇÃO DE DOMÍNIO

Os dados são decompostos em grupos, que serão distribuídos por entre múltiplos processadores que executarão, simultaneamente, um mesmo programa



SPEED-UP E LEI DE AMDAHL

Speed-up é o nome que damos ao ganho de performance de um algoritmo ao utilizarmos computação paralela

- Representa qual é o ganho de velocidade de execução de um algoritmo

No geral, o *speed-up* esperado de um algoritmo pode ser calculado utilizando a Lei de Amdahl

TEMPO DE PROCESSAMENTO PARALELO

$$T(n) = T(1)\left((1 - B) + \frac{1}{n}B\right),$$

onde

- n representa o número de threads
- $T(n)$ representa o tempo esperado de computação utilizando n threads paralelas
- B é a fração paralela de um algoritmo

SPEED-UP TEÓRICO - LEI DE AMDAHL

$$S(n) = \frac{T(1)}{T(n)} = \frac{T(1)}{T(1)\left((1-B) + \frac{1}{n}B\right)} = \frac{1}{(1-B) + \frac{1}{n}B}$$

SPEED-UP TEÓRICO - LEI DE AMDAHL

	B			
n	0,25	0,50	0,75	0,99
2	1,14	1,33	1,60	1,98
10	1,29	1,82	3,08	9,17
50	1,32	1,96	3,77	33,56
100	1,33	1,98	3,88	50,25
1000	1,33	2,00	3,99	90,99
100000	1,33	2,00	4,00	99,90

SPEED-UP NA PRÁTICA

A Lei de Amdahl representa um limite teórico para o *speed-up*

Na prática, não é bem isso o que acontece

- *Speed-up* é sub-linear (em relação a *n*)
- Tempo de barramento
- Tempo de acesso a memória
- Tempo para "juntar"as informações

Em raríssimos casos, pode acontecer um *speed-up* super-linear!

- Diferentes velocidades de memória cache dos processadores