

Sumário

- Introdução:
- ➔ Etapas para o desenvolvimento e análise de Programas Paralelos;
- Computação Paralela sobre Sistemas Distribuídos;
- Ambientes para Troca de Mensagens;
 - PVM;
 - MPI ;
- Resultados;

Etapas para o Desenvolvimento e Análise de um Programa Paralelo

- **I - Desenvolvimento de um Algoritmo Paralelo**
 - Abordagem do Algoritmo
 - Identificação do Algoritmo e Divisão dos Processos
 - Organização do Trabalho
- **II - Desenvolvimento do Programa Paralelo**
 - Formas de Expressar Paralelismo
 - Comunicação e Sincronismo
 - Linguagens para Programação Paralela

Etapas para o Desenvolvimento e Análise de um Programa Paralelo

■ III - Mapeamento de Processos:

- Escalonamento
- Balanceamento de Carga
- Migração de Processos

■ IV - Teste e Depuração

- Efeito na Inserção de Testes
- Dificuldades com E/S Paralelo

■ V - Avaliação de Desempenho

- *Speedup*
- Eficiência

Introdução

Tarefa para casa

Ler o Capítulo 1 do Livro:

ALMASI, G., GOTTLIEB, A., *Highly Parallel Computing*, Second Edition, The Benjamin/Cummings Publishing Company, 1994.

Desenvolvimento do Algoritmo Paralelo

Abordagem do Algoritmo
Identificação do Algoritmo e Divisão dos Processos
Organização do Trabalho

Desenvolvimento do Algoritmo Paralelo

■ Programa Seqüencial:

- Existência de programa seqüencial com documentação e especificação pobres;
- Utilização de ferramentas;
- Baixa flexibilidade - *Speedup* limitado.

■ Algoritmo seqüencial

- Adaptar algoritmo seqüencial e refazer programa;
- Nem sempre melhor algoritmo seqüencial é o melhor algoritmo paralelo;
- Maior flexibilidade, mas ainda limitada;
- Melhores *speedups*, mas ainda pode não ser ideal.

Desenvolvimento do Algoritmo Paralelo

- Problema a ser resolvido:
 - Análise do problema e proposta de algoritmo;
 - Programador deve conhecer bem o problema;
 - Alta flexibilidade, pode-se obter bom *speedup*;
 - Base em algoritmos paralelos que resolvem outros problemas.

Abordagem do Algoritmo

■ Exemplo: Método de Ordenação "Bolha"

Analizando pelo programa:

```
for (k=1;k<n-1;k++)  
    for (j=1;j<n-k;j++)  
        for (i=1;i<n-1-k;i++)  
            if (A[i]>A[i+1])  
            {  
                aux = A[i+1]  
                A[i+1] = A[i]  
                A[i] = aux  
            }
```

⇒ **Difícil paralelizar!**

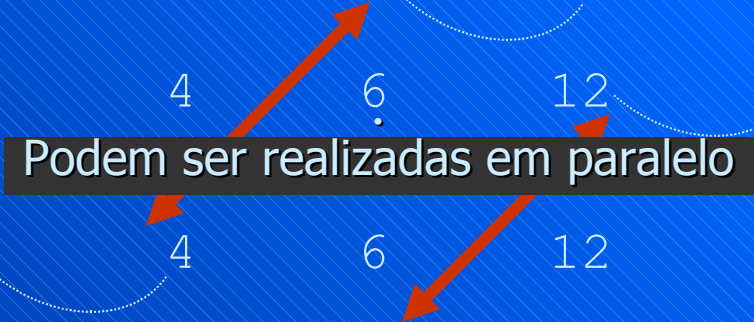
Abordagem do Algoritmo

■ Exemplo: Método de Ordenação "Bolha"

Analizando pelo algoritmo

passos

1.1	12	10	4	6	15	2	8
1.2	10	12	4	6	15	2	8
1.3	10	4	12	6	15	2	8
1.4	10	4	6	12	15	2	8
2.1	10	4	6	12	2	8	15
2.2	4	10	6	12	2	8	15
2.3	4	6	10	12	2	8	15
			⋮				



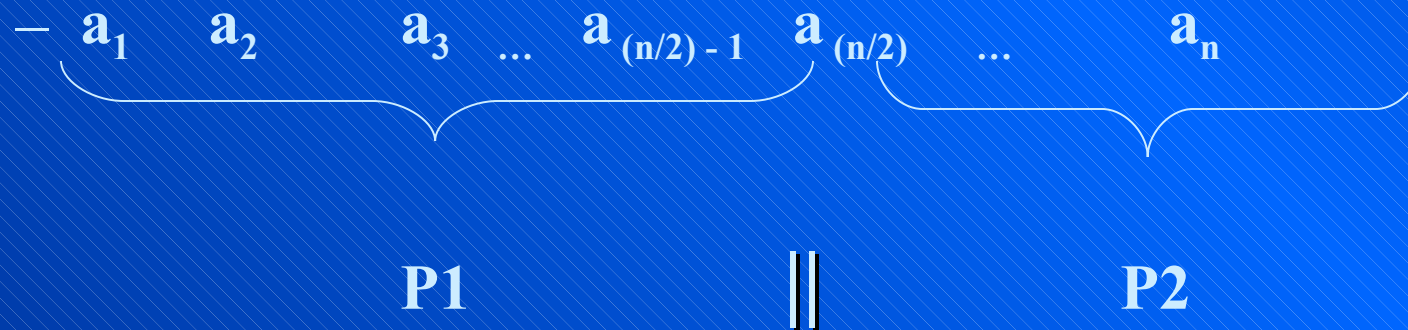
Podem ser realizadas em paralelo

Abordagem do Algoritmo

■ Pode-se concluir que:

- O passo 1.3 pode ser executado em paralelo com 2.1;
- O passo 1.4 pode ser executado em paralelo com 2.2;
- O passo 1.5 pode ser executado em paralelo com 2.3 e 3.1;

■ Pelo Problema:



- Ao final, deve-se realizar um merge.

Desenvolvimento do Algoritmo Paralelo

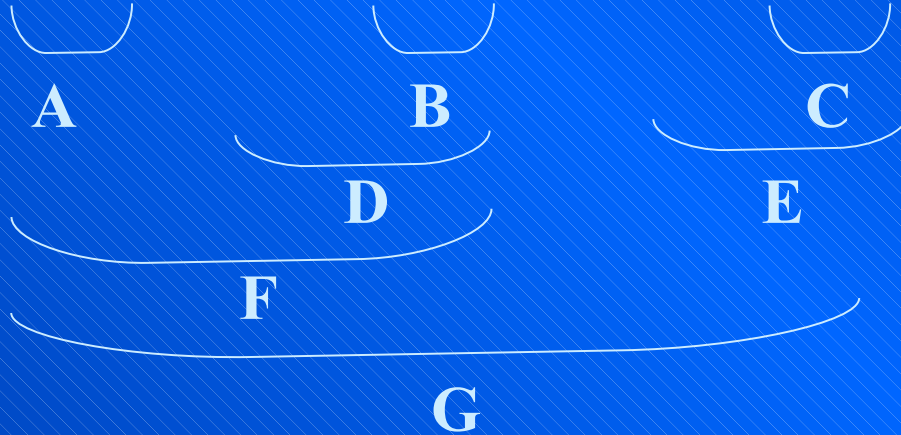
Abordagem do Algoritmo
Identificação do Algoritmo e Divisão dos Processos
Organização do Trabalho

Identificação do paralelismo e divisão dos processos

- Aspectos importantes que devem ser considerados:
 - Arquitetura das máquinas disponíveis;
 - Tipo de comunicação;
 - Granulação;
 - Sincronismo.
- Exemplo: Expressão Aritmética.

Identificação do paralelismo e divisão dos processos

■ $(a * b + c * d^2) * (g + f * h)$



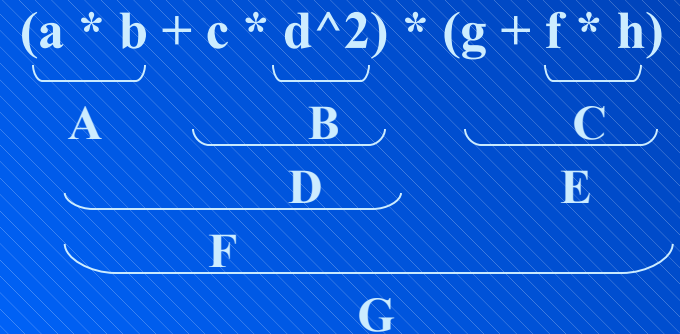
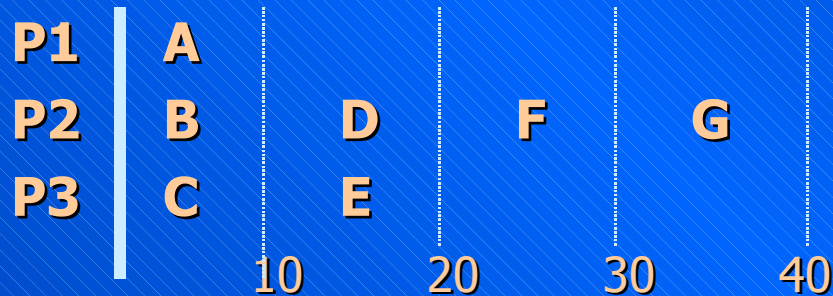
■ Considerando Toperação = 10 ut, tem-se que:

$$T_{seq} = 70 \text{ ut}$$

$$T_{par} = ?$$

Identificação do paralelismo e divisão dos processos

- Utilizando três processadores:



- Ao final da execução, verifica-se que:

$T_{par} = 40 \text{ ut}$

(sem considerar comunicação!)

- Calculando o *speedup* e a eficiência...

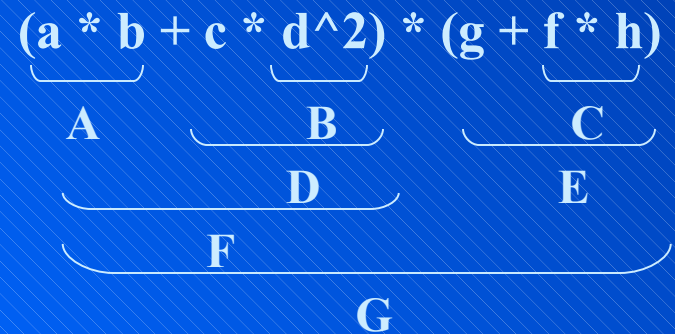
$Sp = 70 / 40 = 1.75$

$Ef = 1.75 / 3 = 58\%$

Identificação do paralelismo e divisão dos processos

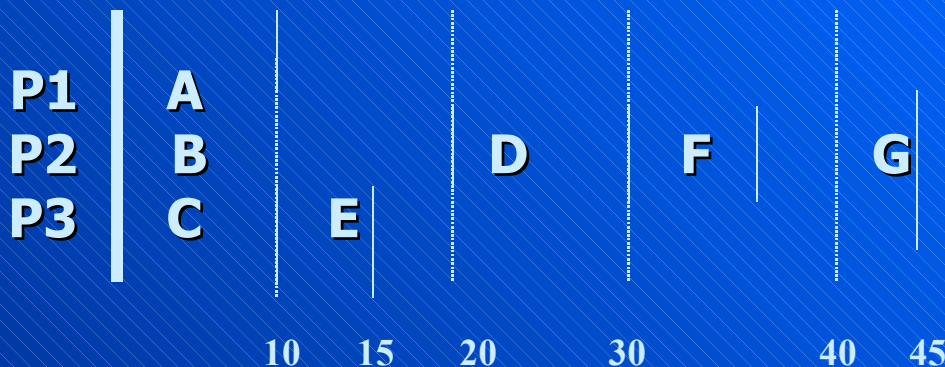
■ Agora, considerando:

- $T^{\wedge} = 20 \text{ ut}$
- $T^* = 10 \text{ ut}$
- $T^+ = 5 \text{ ut}$



■ $T_{\text{seq}} = 70 \text{ ut}$

■ $T_{\text{par}} = 45 \text{ ut}$

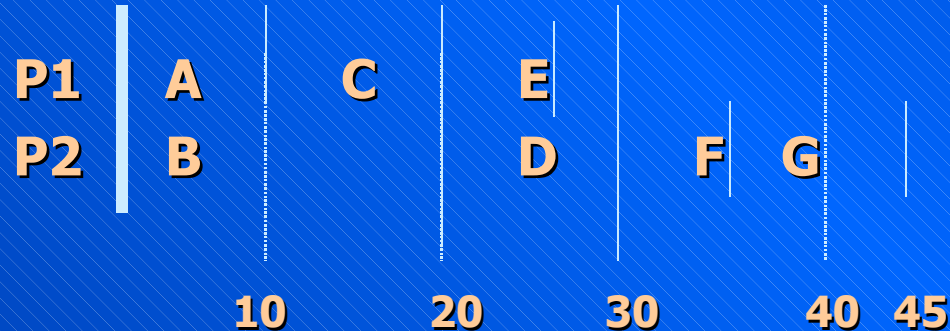


$$Sp = 70 / 45 = 1,55$$

$$Ef = 1,55 / 3 = 52\%$$

Identificação do paralelismo e divisão dos processos

- Analisando o diagrama, nota-se que A pode ser executado por P3 ou C e E por P1



O tempo de execução
ainda continua sendo 45 ut

- Assim:

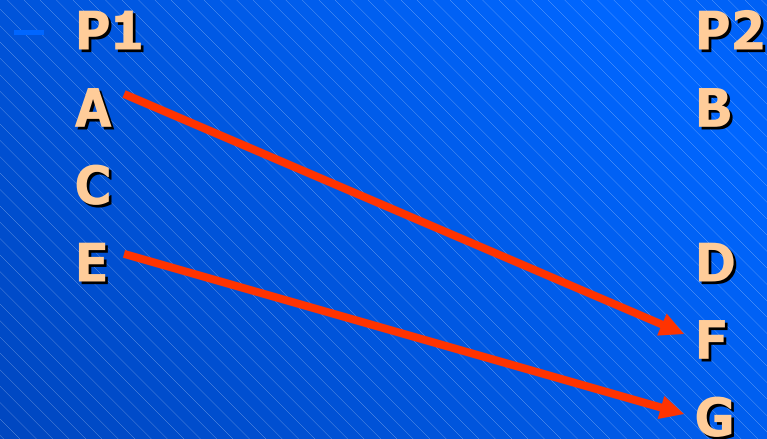
$$S_p = 70/45 = 1,55$$

$$E_f = 1,55 / 2 = 77\%$$



Identificação do paralelismo e divisão dos processos

■ Considerando comunicação...



$$\underbrace{(a * b)}_A + \underbrace{c * d^2}_B \underbrace{)}_{D} * \underbrace{(g + f * h)}_E \underbrace{)}_{G}$$

■ Assim: $T_{par} = 45 + 2 * T_{comunicação}$

Comunicação = sobrecarga!

Abordagem do Algoritmo

■ Exemplo 2 - Soma dos Elementos de um Vetor

$$- a_1 + a_2 + a_3 + \dots a_{(n/2)-1} + a_{(n/2)} + \dots + a_{n-1} + a_n$$

Desenvolvimento do Algoritmo Paralelo

Abordagem do Algoritmo
Identificação do Algoritmo e Divisão dos Processos
Organização do Trabalho

Organização do Trabalho

- Especifica o modelo de concorrência a ser utilizado;
- Depende da arquitetura considerada:
 - SIMD (*Single Instruction Multiple Data*)
 - MIMD (*Multiple Instruction Multiple Data*)
 - **Memória compartilhada**
 - **Memória distribuída**
- Duas Abordagens:
 - Paralelismo por Dado;
 - Paralelismo por Controle.

Organização do Trabalho

- Paralelismo por Dado:
 - Executa as mesmas instruções simultaneamente em um conjunto de dados distintos.
- Paralelismo por Controle:
 - Executa instruções diferentes sobre dados diferentes.
- Exemplo: *Sieve of Eratosthenes*
 - Algoritmo para procurar números primos em um conjunto de números naturais.

Organização do Trabalho

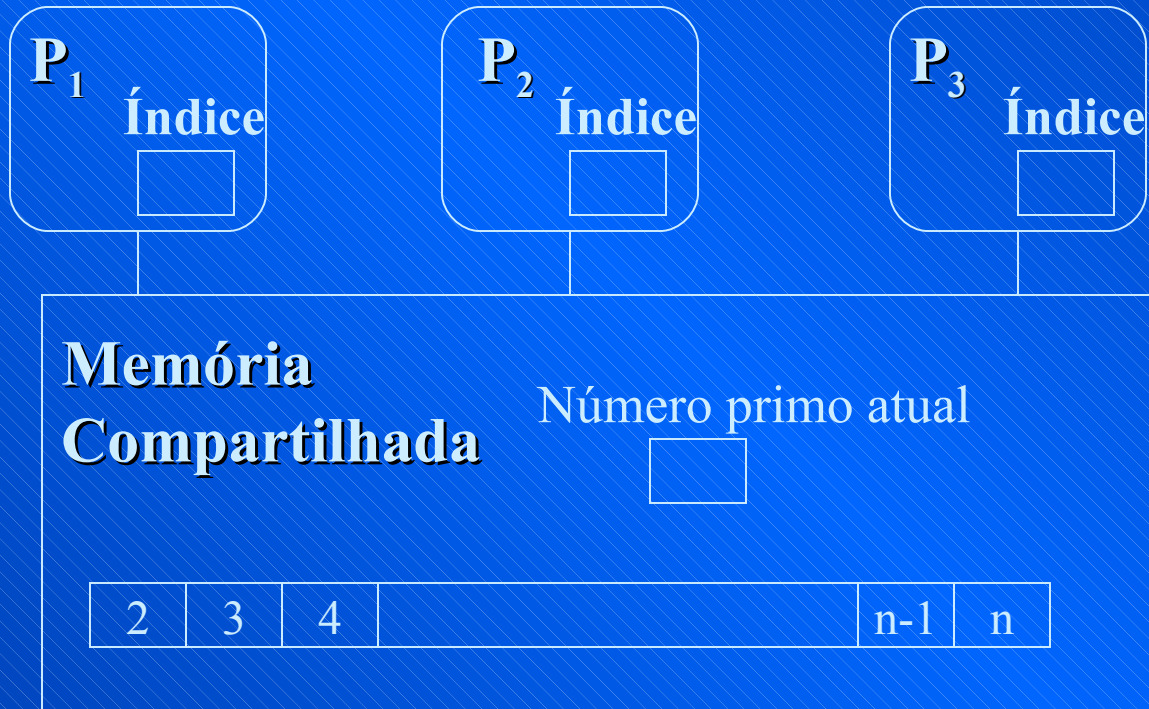
■ Algoritmo *Sieve of Eratosthenes*

- Marcar os múltiplos de um conjunto de números até n ;
- Termina a execução quando for atingir um número maior que \sqrt{n}



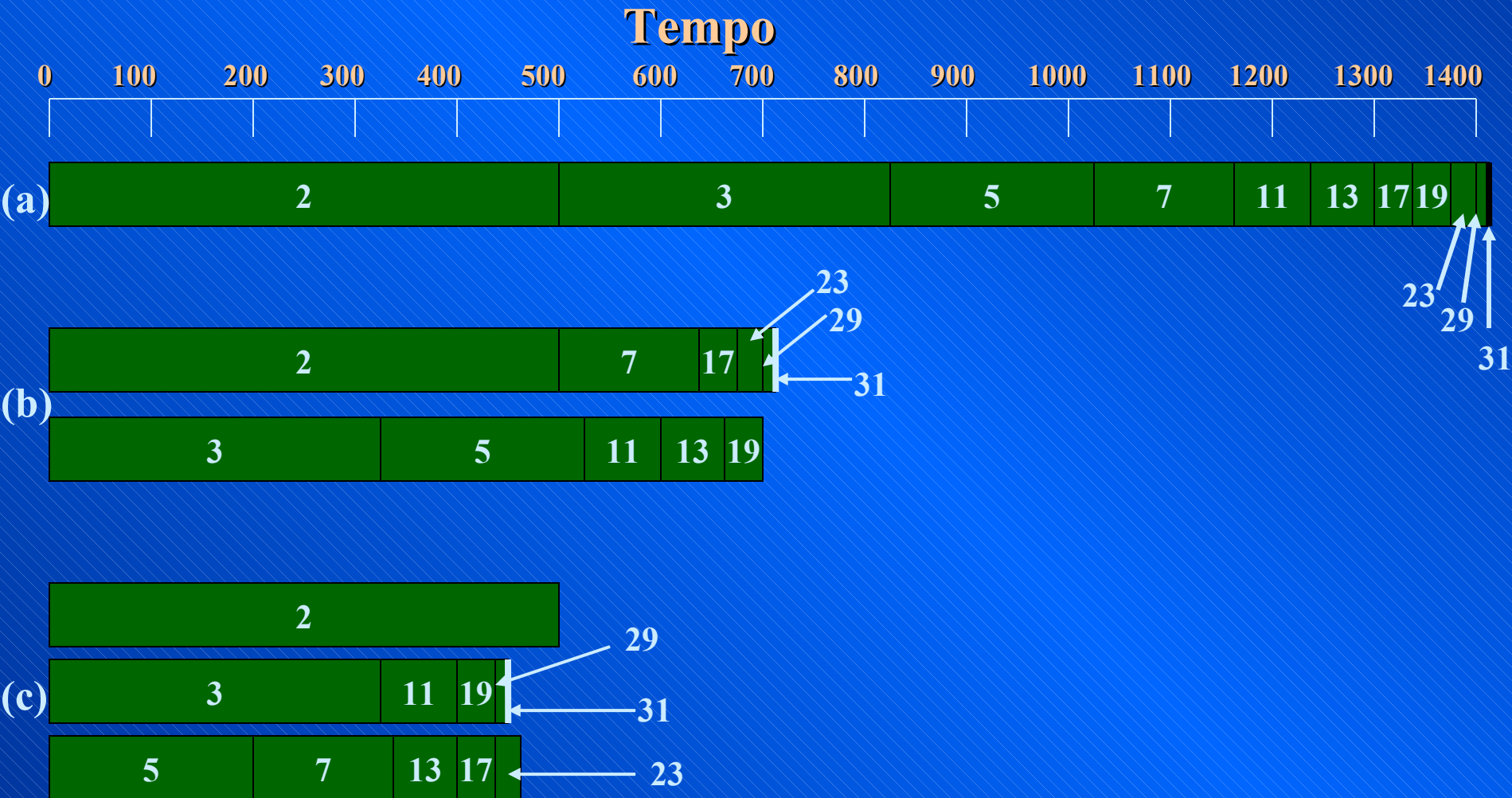
Organização do Trabalho

■ *Silve of Eratosthenes*: Paralelismo por Controle



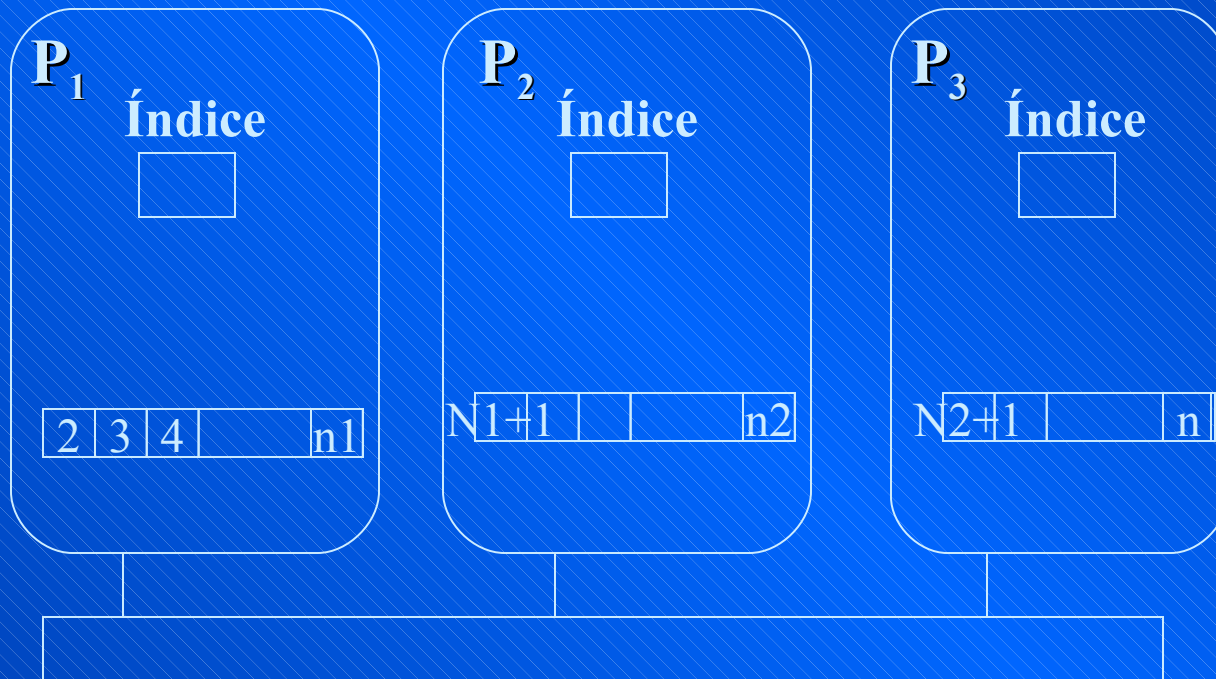
Organização do Trabalho

■ *Sieve of Eratosthenes*: Paralelismo por Controle



Organização do Trabalho

- *Sieve of Eratosthenes*: Paralelismo por Dado



Organização do Trabalho

- Paralelismo por Dado e Controle podem ser divididos em:
 - Abordagem *Processor Farm*;
 - Abordagem *Pipeline* ou Especialista;
 - Abordagem Geométrica ou Resultado.

Organização do Trabalho

■ Abordagem *Processor Farm*:

- Cada processador é designado para ajudar no atual item da pauta;
- Geralmente, há um processador mestre que envia o item para cada processador participante;
- Problema: Sobrecarga para o processador mestre;
- Exemplo: Multiplicação de Matrizes $A[n][k] * B[k][m]$.

Organização do Trabalho

■ Abordagem *Processor Farm*:

- Mestre: envia aos escravos ociosos a próxima posição da matriz produto a ser calculada;
- Escravo: solicita posição ao mestre, determina produto, envia resposta ao mestre.



– Vantagens:

- Flexibilidade quanto ao número de processadores;
- Balanceamento de carga automático.

Organização do Trabalho

- Abordagem Pipeline ou Especialista:
 - Cada processador é responsável por um tipo específico de trabalho;
 - Para uma tarefa → Nenhum paralelismo!
 - Sincronismo é essencial, pois tarefas posteriores dependem das anteriores;
 - Eficiência: Depende do tamanho da tarefa;
 - Exemplo: Regra do Trapézio
 - Resolução de N Integrais → N tarefas;

Organização do Trabalho

- Abordagem Pipeline ou Especialista:
 - As integrais da Regra do Trapézio são formadas pela equação:

$$\int_a^b f(x) dx \cong \frac{(b-a) [f(a) + f(b)]}{2}$$

Diagram illustrating the stages of the Trapezoidal Rule formula:

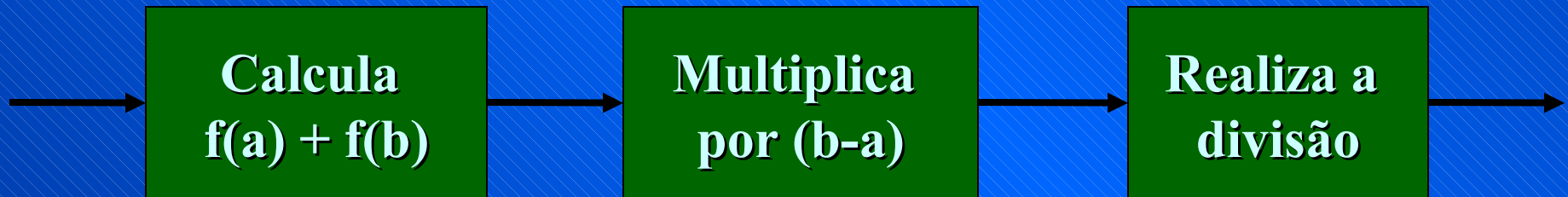
- 2º Estágio**: Points to the entire right-hand side of the equation.
- 1º Estágio**: Points to the term $[f(a) + f(b)]$.
- 3º Estágio**: Points to the entire fraction $\frac{(b-a) [f(a) + f(b)]}{2}$.

Organização do Trabalho

- Abordagem Pipeline ou Especialista:

$$\int_a^b f(x) dx \cong \frac{b-a [f(a) + f(b)]}{2}$$

- Com isso, o *pipeline* pode ser organizado conforme:



- Problemas:

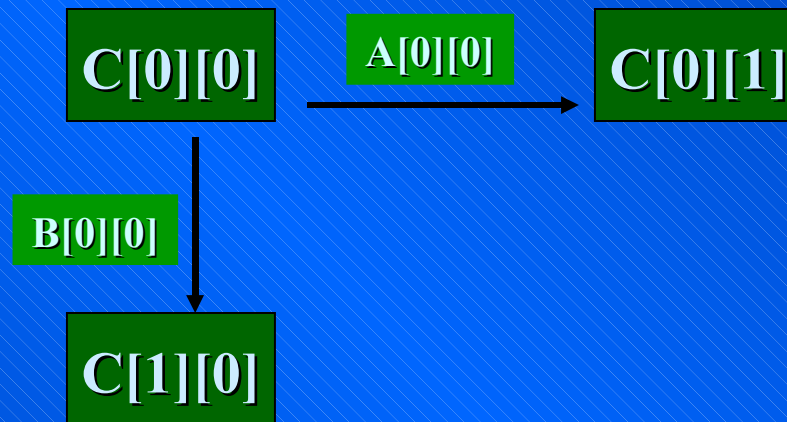
- Pouco flexível;
- Tempo de latência.

Organização do Trabalho

- Abordagem Geométrica ou pelo Resultado:
 - Cada processador é designado para produzir um pedaço do produto final;
 - Deve-se ponderar: Comunicação, Sincronismo e Balanceamento;
 - Exemplo: Multiplicação das Matrizes: $A[n][k] * B[k][m]$
 - Cada processador pode ser responsável por um elemento da matriz resultante;
 - Os valores utilizados na multiplicação são enviados para outros processadores.

Organização do Trabalho

- Abordagem Geométrica ou pelo Resultado:



- Obtenção de um alto grau de paralelismo;
- São necessários $N \times M$ processadores;
- Sobrecarga na comunicação;
- Granulação fina.