Técnicas de Paralelização

2 – Particionar e dividir para conquistar

Particionamento

O problema é dividido em partes ou tarefas, cada uma processada individualmente.

Dividir para conquistar

O problema é dividido em subproblemas com a mesma forma que o problema principal. Geralmente prossegue recursivamente a dividir cada subproblema em subproblemas menores.

Exemplos

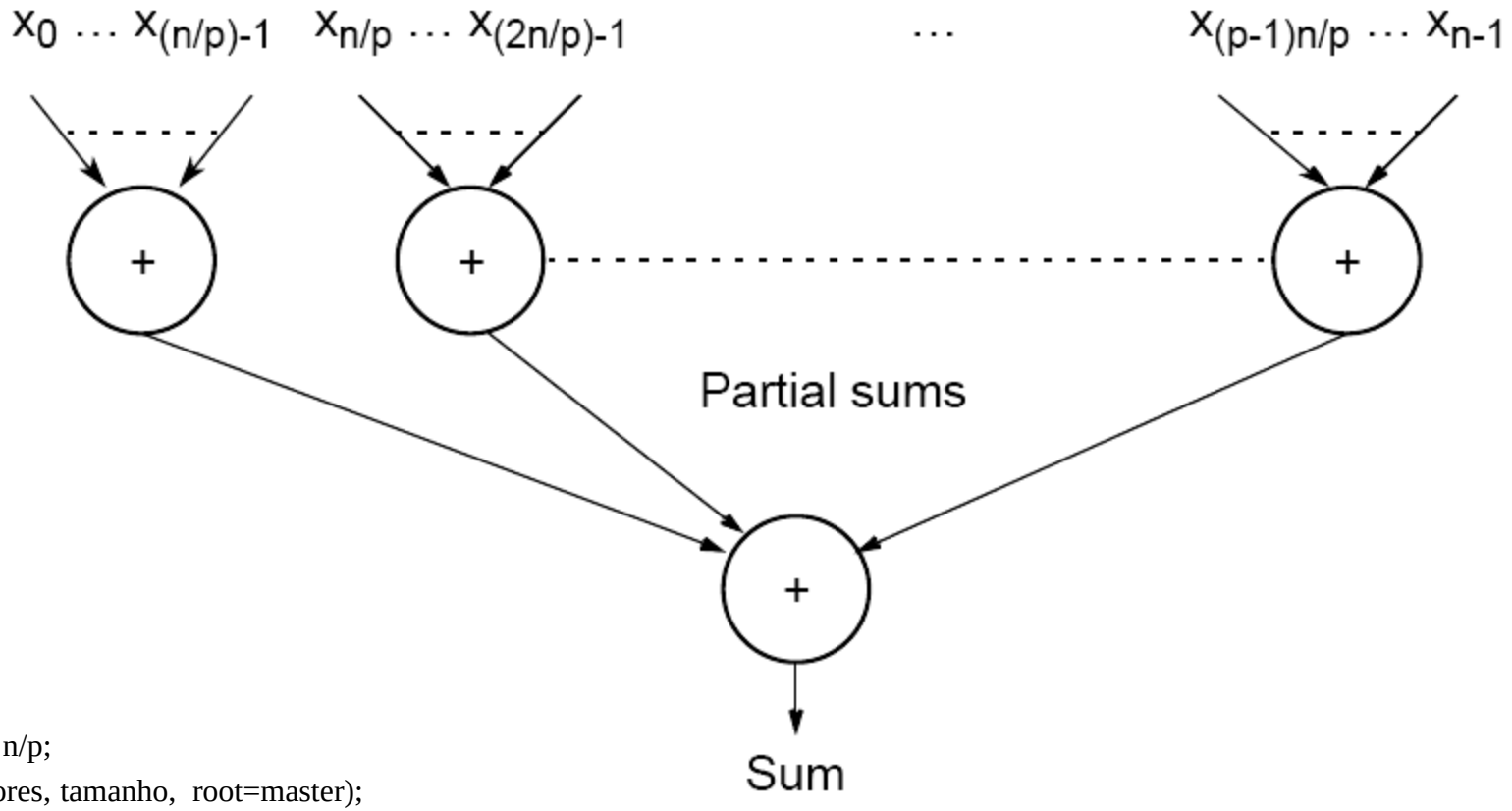
- Operações em sequências de números
- Algoritmos de ordenação
- Integração numérica
- "*N*-body problema"

Particionamento

- Pode ser aplicado aos dados (particionamento de dados)

Pode ser aplicado às funcionalidades de um programa (decomposição funcional)

Particionar uma sequência de valores em partes e adicionar as partes



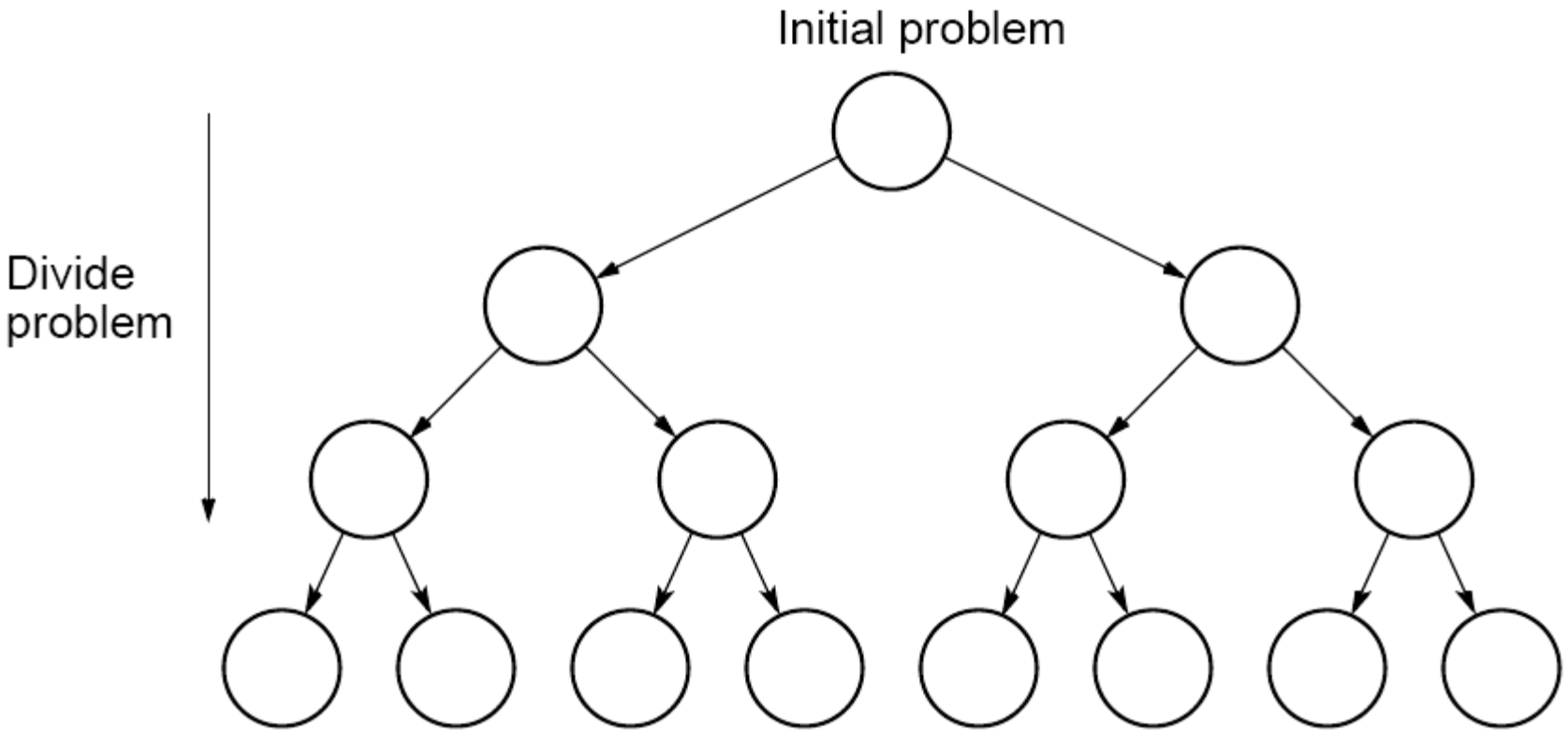
tamanho = n/p;
scatter(valores, tamanho, root=master);
Cacalulcar somas parcias
reduce_add(parcialsum → sum, root=master);

Dividir para conquistar

Somar recursivamente uma lista de números:
(Versão sequencial)

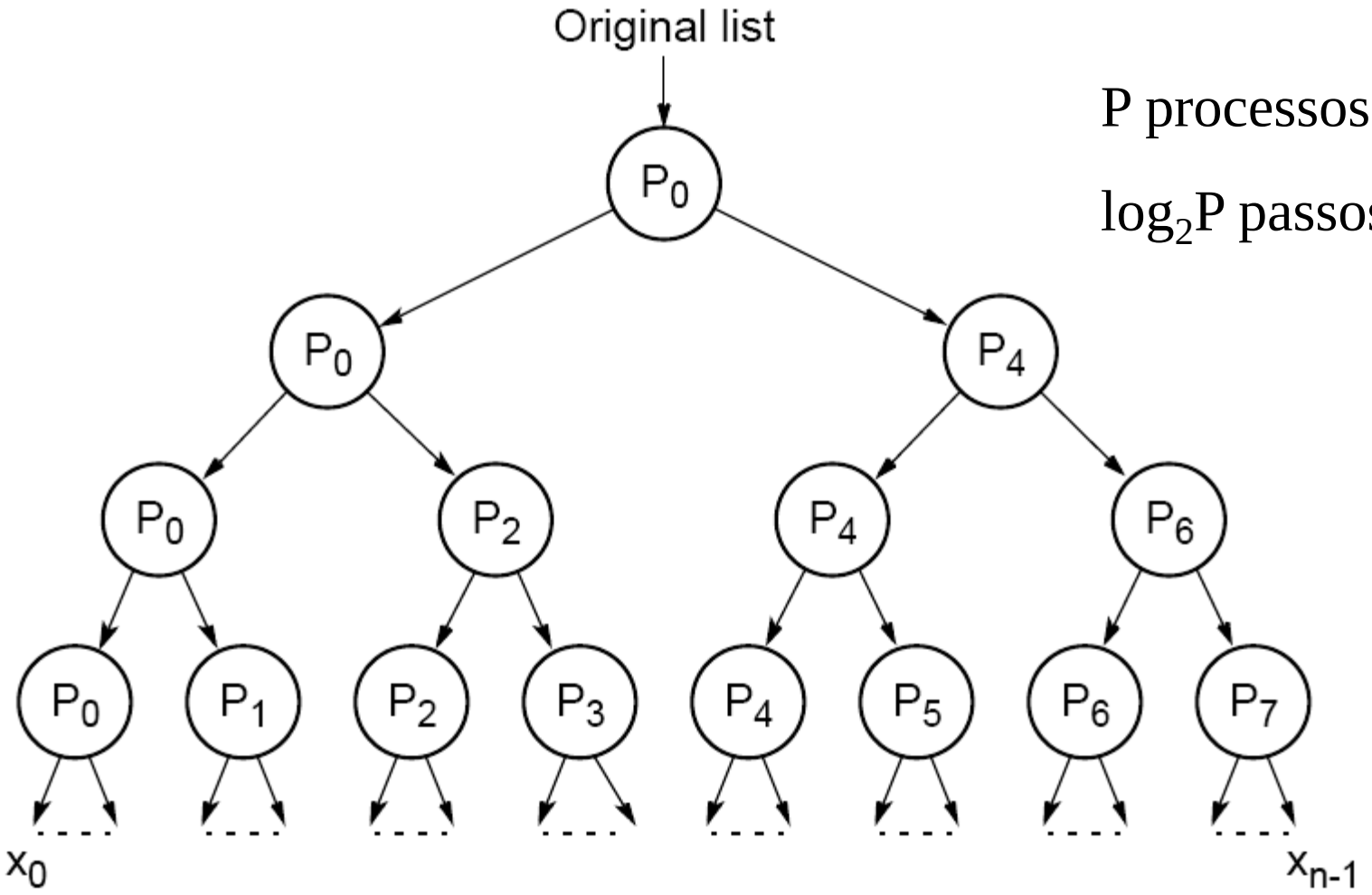
```
def soma( ar ):
    if (ar.size == 1):
        return ar[0]
    if (ar.size == 2):
        return ar[0]+ ar[1]
    s1 = ar[0:ar.size/2]
    s2 = ar[ar.size/2: ar.size]
    parcial1 = soma(s1)
    parcial2 = soma(s2)
    return parcial1 + parcial2
```

Construção de uma árvore



Dividir a lista em partes:

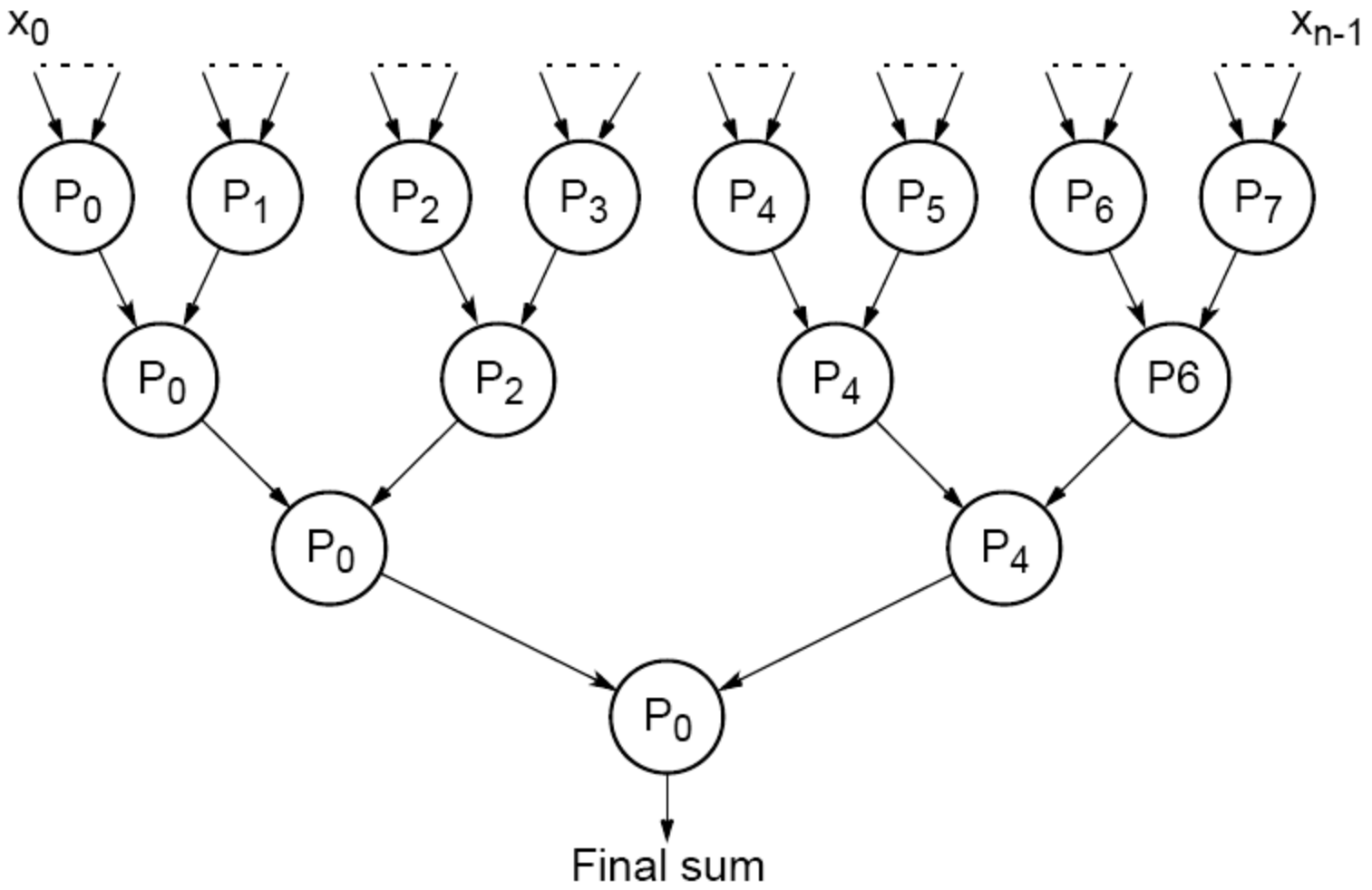
(reusar processos em todos os níveis da árvore)



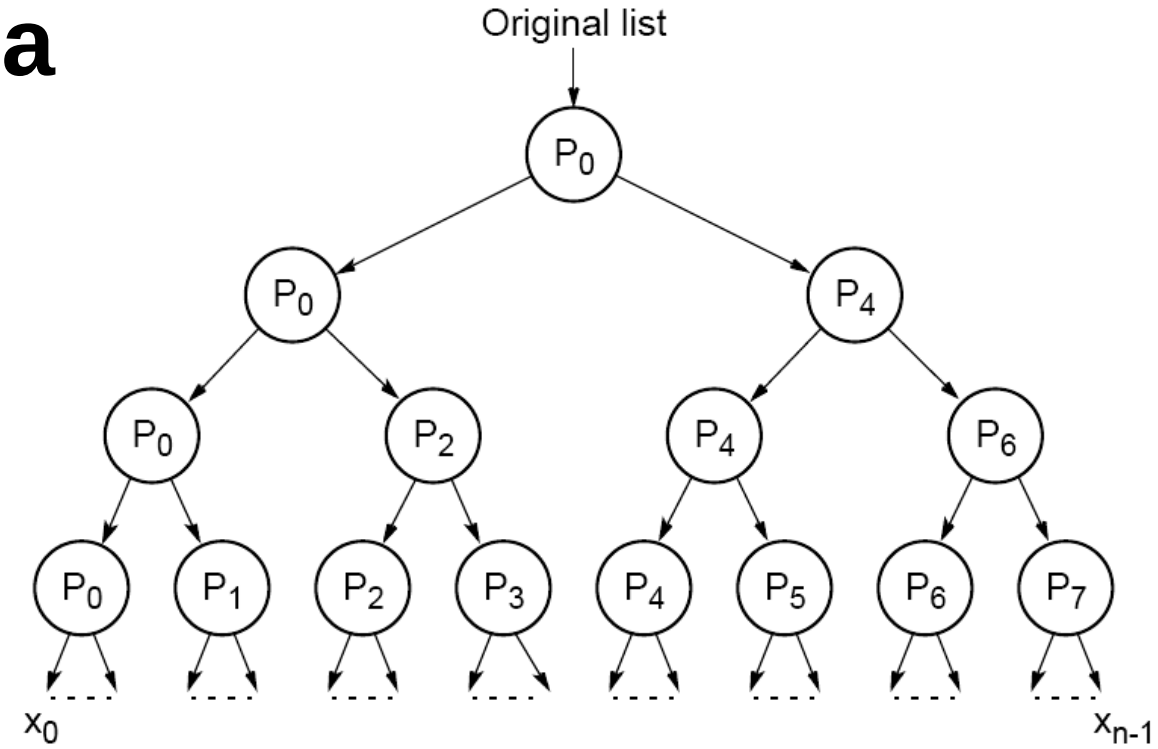
P processos

$\log_2 P$ passos

Resultados parciais



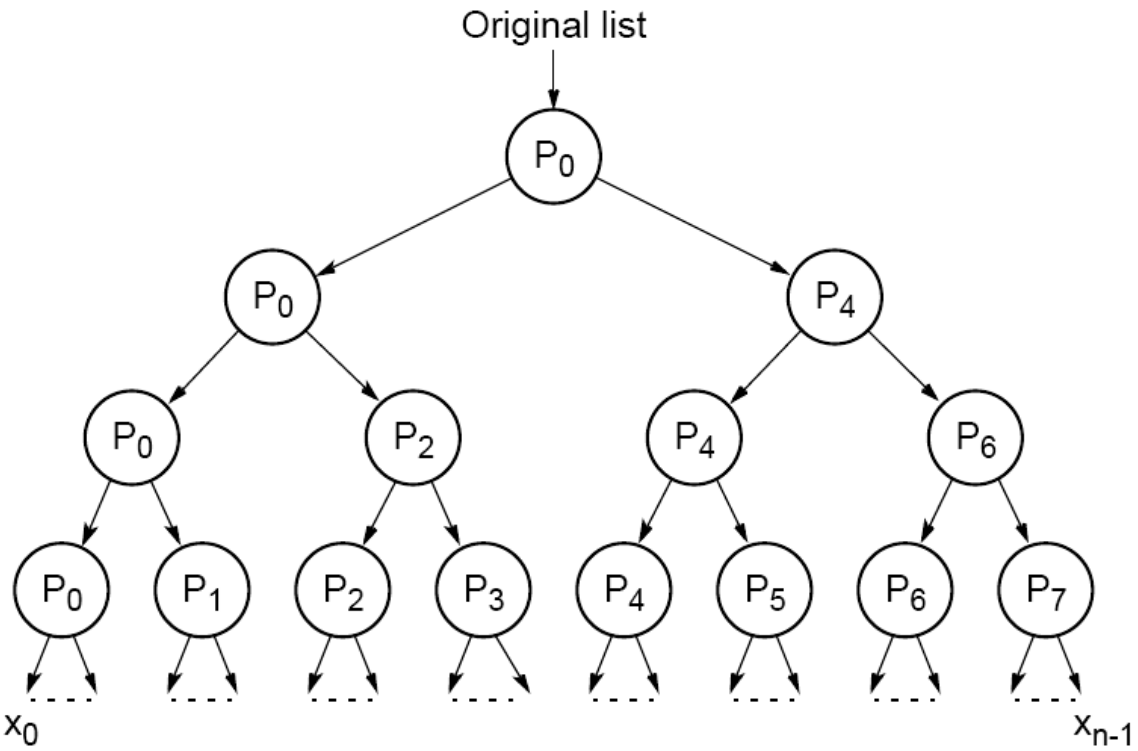
Versão paralela



Processo P0

```
dividir (s1, s1, s2)
send (s2, P4)
dividir (s1, s1,s2)
send ( s2, P2)
dividir (s1, s1,s2)
send ( s2, P1)
soma_parcial = soma(s1)
recv ( soma_parcial1, P1)
soma_parcial += soma_parcial1
recv ( soma_parcial1, P2)
soma_parcial += soma_parcial1
recv ( soma_parcial1, P4)
soma_parcial += soma_parcial1
```

Versão paralela

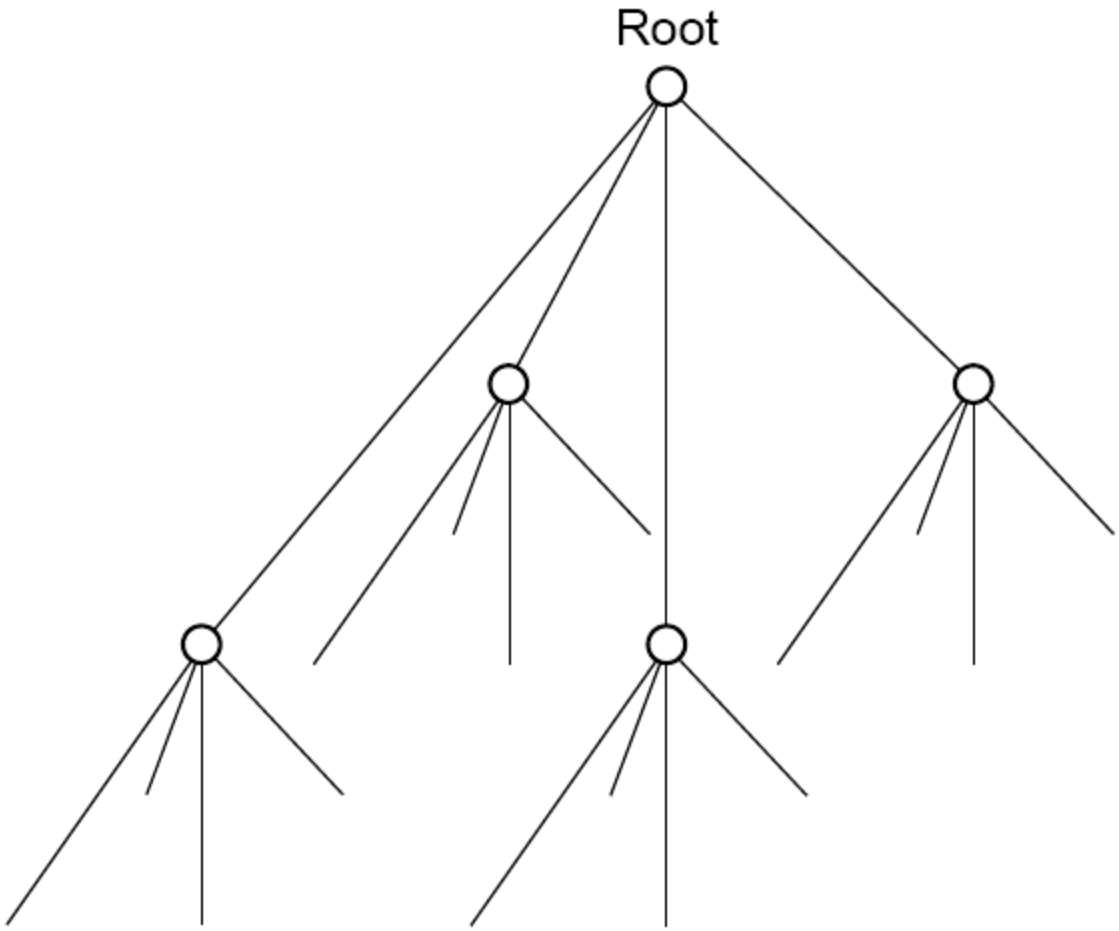


Processo P4

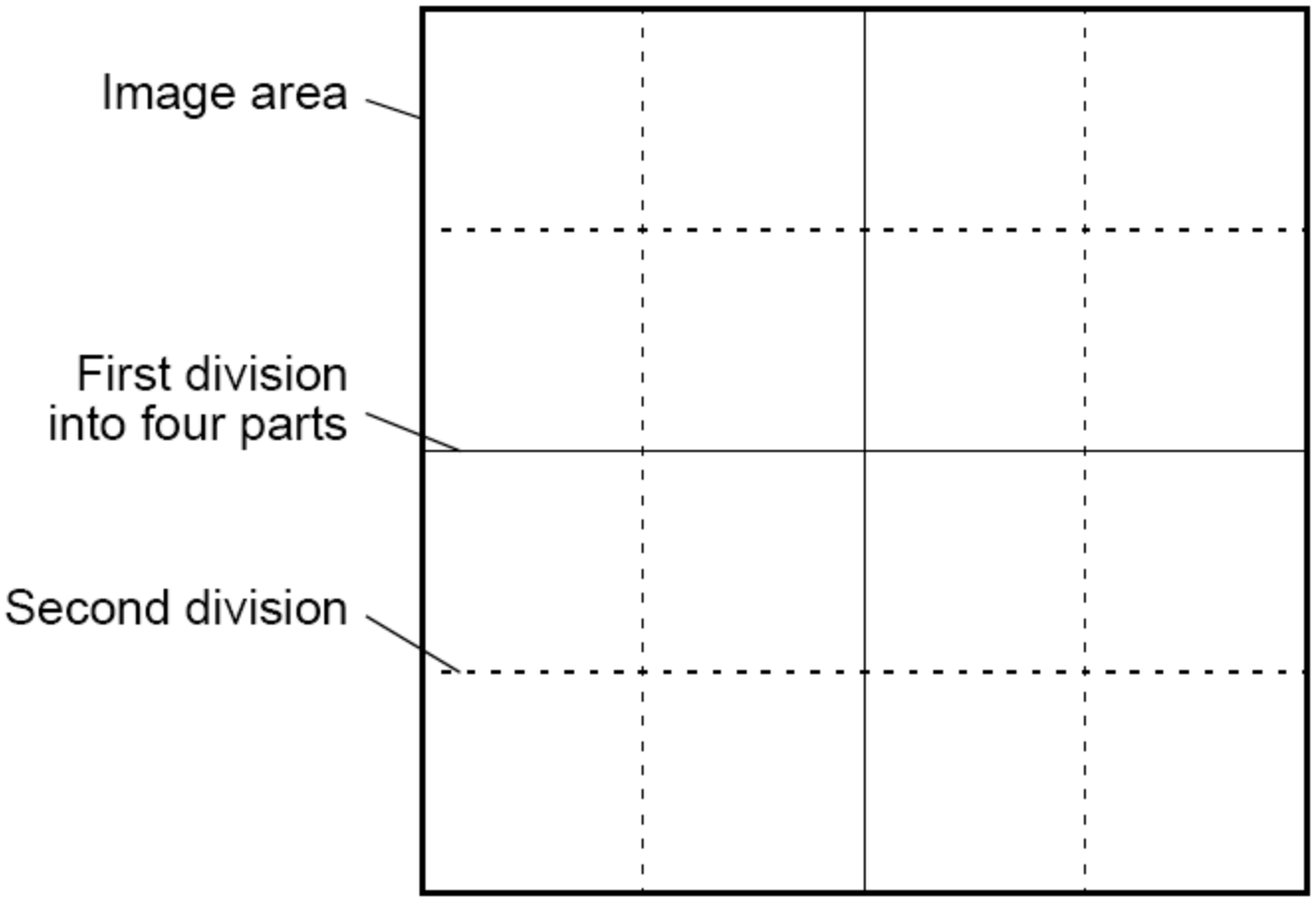
```
recv ( s1, P0)
dividir (s1, s1, s2)
send (s2, P6)
dividir (s1, s1,s2)
send ( s2, P5)
soma_parcial = soma(s1)
recv ( soma_parcial1, P5)
soma_parcial += soma_parcial1
recv ( soma_parcial1, P6)
soma_parcial += soma_parcial1
send ( soma_parcial, P0)
```

Código de P1, P2, P3, P5, P6, P7…

Quadtree



Dividir uma imagem



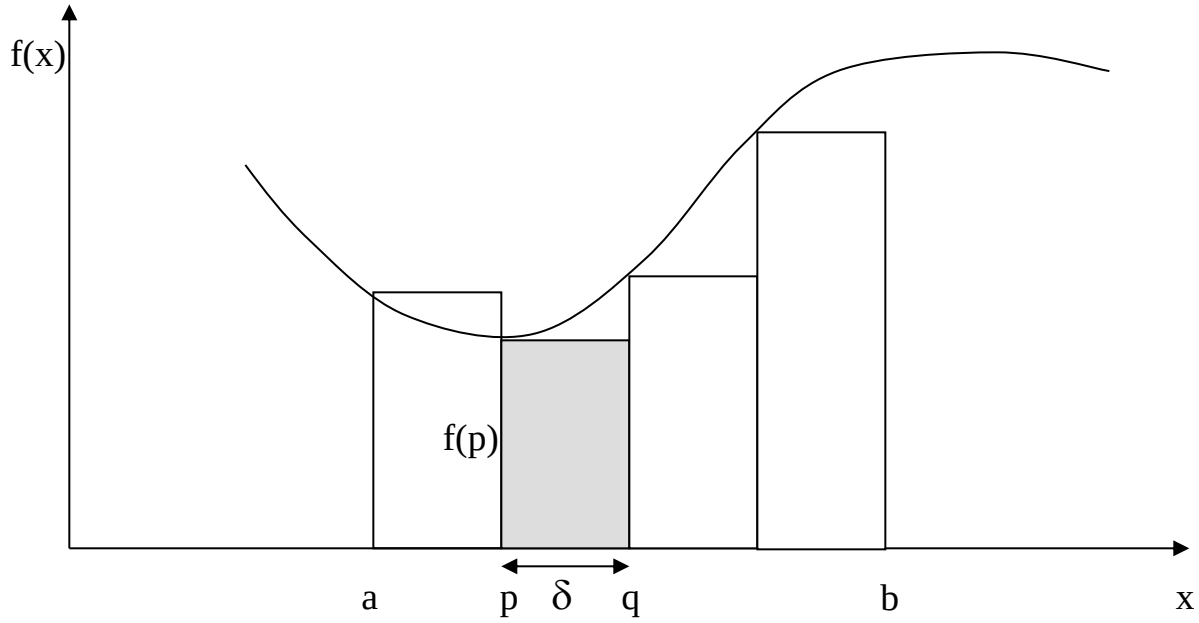
Integração numérica

- Uma técnica geral de divisão e conquista consiste em dividir a região continuamente em partes e existe uma função de optimização que decide quando esta está suficientemente dividida

- Exemplo:
  - Integração numérica: divide a área em várias partes e cada uma delas pode ser calculada por um processo separado

Integração numérica usando rectângulos

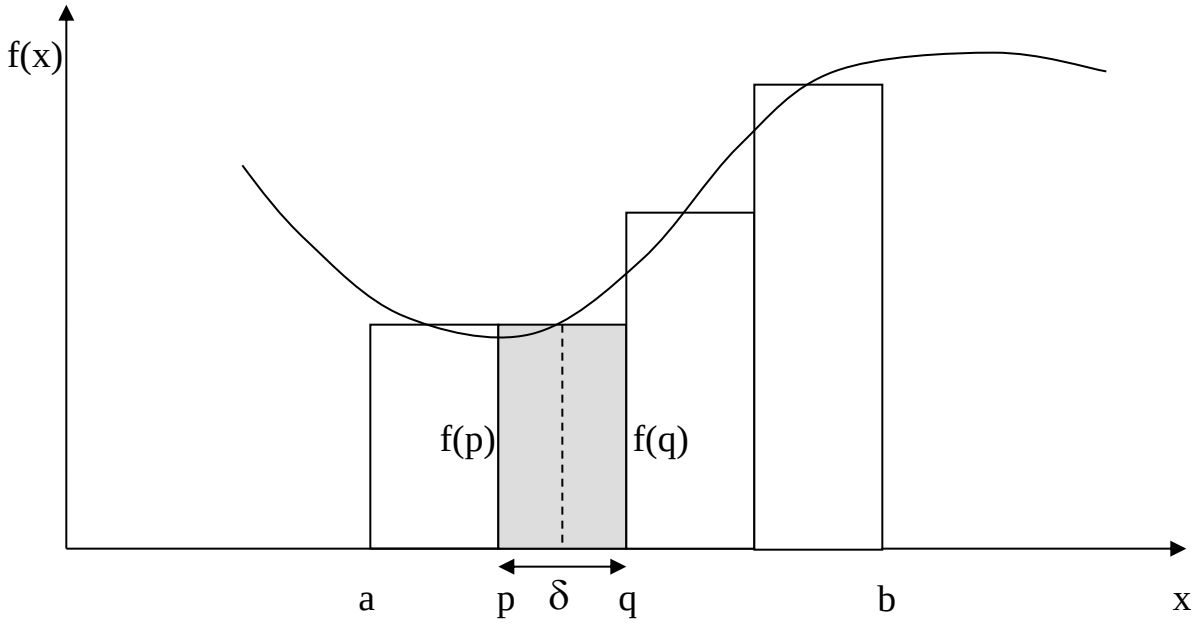
- Cada região pode ser calculada por uma aproximação utilizando retângulos - fixados à esquerda



Area=$\partial \times f(p)$

Integração numérica usando rectângulos

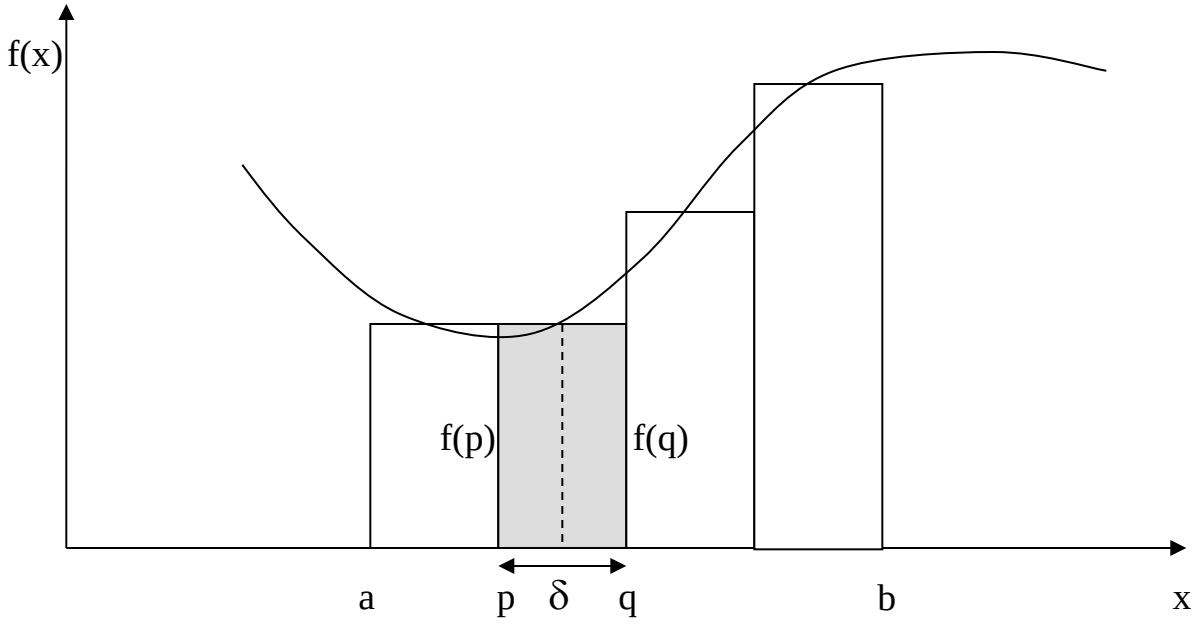
- Retângulo centrado



$$\text{Area} = \partial \times f\left(\frac{(p+q)}{2}\right)$$

Integração numérica usando rectângulos

- Regra do trapézio



$$\text{Area} = 0.5 \times \partial \times (f(p)+f(q))$$

Integração numérica usando rectângulos

- Decomposição Estática

Pseudo código

Obter intervalo [a,b] e numero de intervalos

Fazer Broadcast

Dividir intervalo - particionamento de numero de intervalos.

Cada processo cálcula area atribuída

Calcular área global usando operação de redução com operação de somar

Pseudo código SPMD

Processo P_i

```
if (i == master) {
    printf ("Introduza o número intervalos ");
    scanf ("%d", &n);
}
bcast(&a,&b,n, P_group);
region = (b-a)/p;
start=a + region * i;
end = start + region;
d=(b-a)/n;
area=0.0;
for ( x = start;  x <end;  x = x+d)
    area = area +f(x) + f(x+d);
area=0.5 * area * d;

reduce_add(&integral, &area, P_group);
```