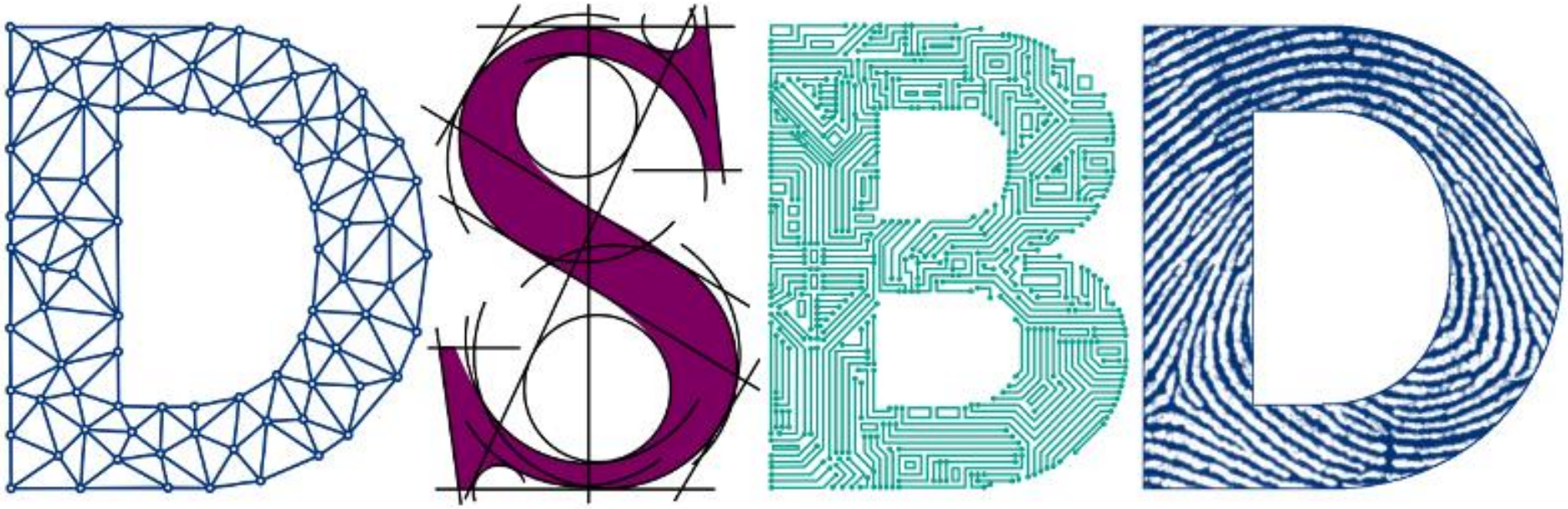
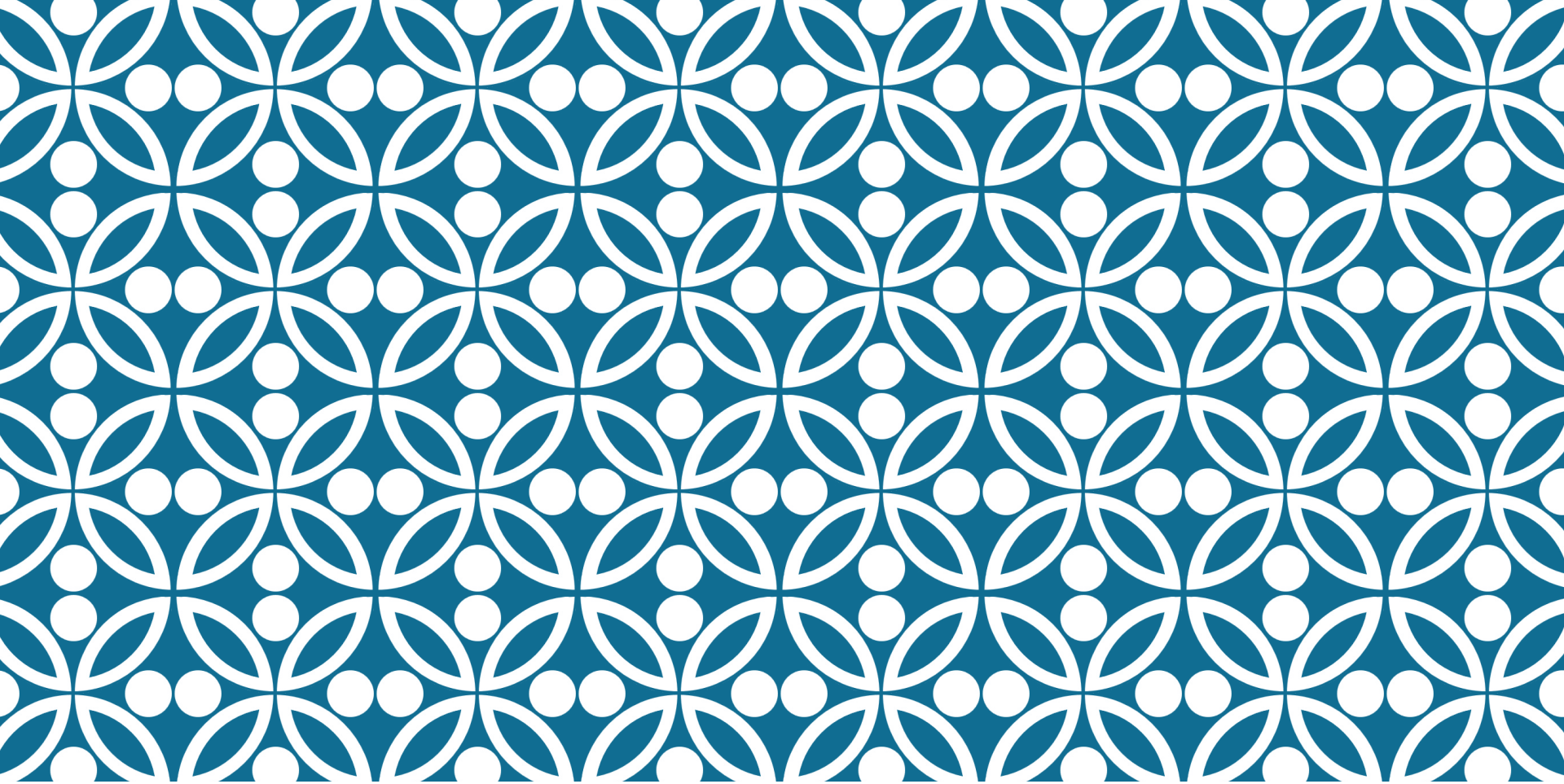


PROGRAMAÇÃO PARALELA MÉTRICAS DE DESEMPENHO

Infraesrutura Computacional Pt.3

Marco A. Z. Alves

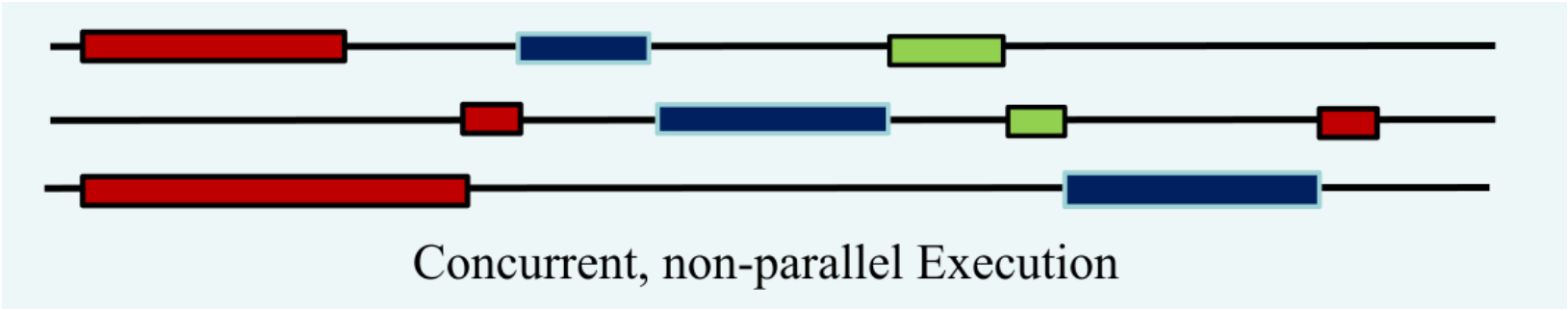


DEFINIÇÕES BÁSICAS

CONCORRÊNCIA VS. PARALELISMO

Duas definições importantes:

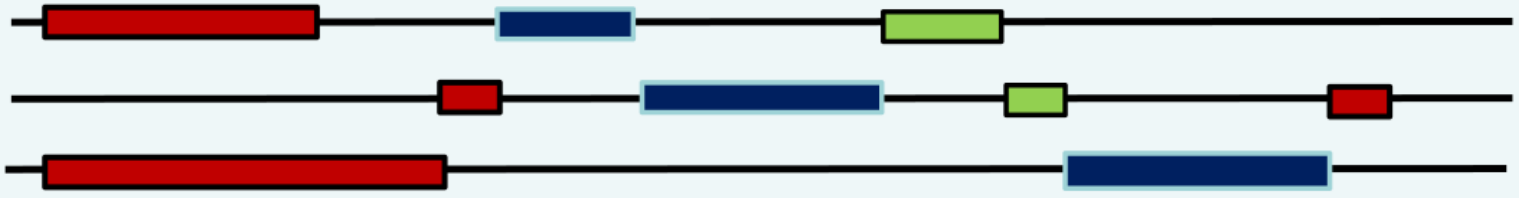
Concorrência: A propriedade de um sistema onde múltiplas tarefas estão logicamente ativas ao mesmo tempo



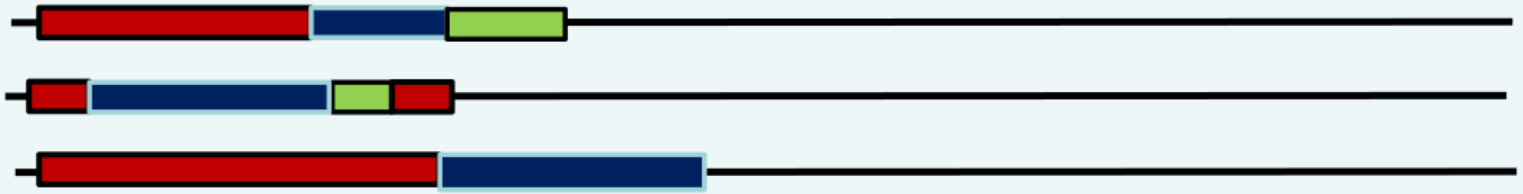
CONCORRÊNCIA VS. PARALELISMO

Duas definições importantes:

Paralelismo: A propriedade de um sistema onde múltiplas tarefas estão realmente ativas ao mesmo tempo

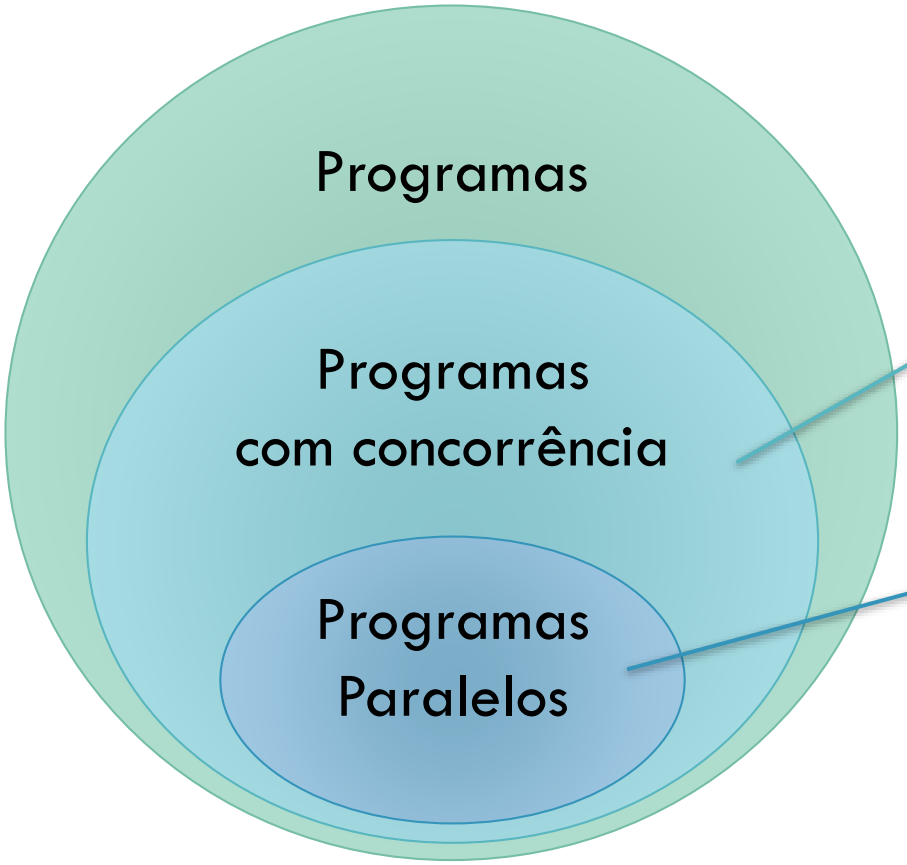


Concurrent, non-parallel Execution



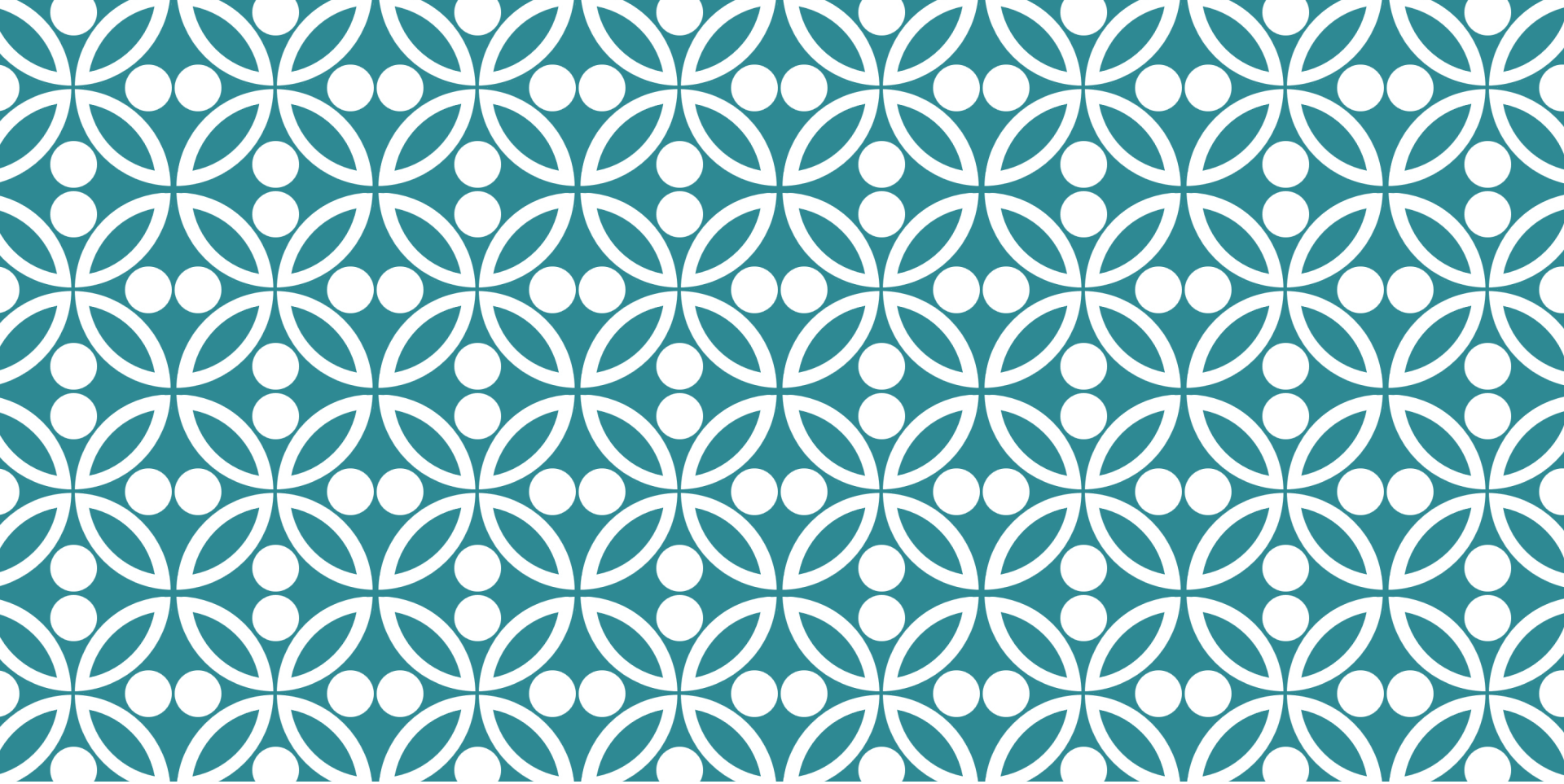
Concurrent, parallel Execution

CONCORRÊNCIA VS. PARALELISMO



Problemas fundamentalmente concorrente:
Ex. Web services
Banco de dados

Queremos otimizar a execução (não necessariamente concorrente)
- Tempo ou + Dados
Ex. Programas sequenciais



LIMITES DO DESEMPENHO

LIMITES AO DESEMPENHO

Limites Arquiteturais: ???

Limites Algorítmicos: ???

LIMITES AO DESEMPENHO

Limites Arquiteturais (veremos mais adiante)

- Latência e Largura de Banda (Rede / Memória)
- Coerência dos Dados
- Capacidade de Memória

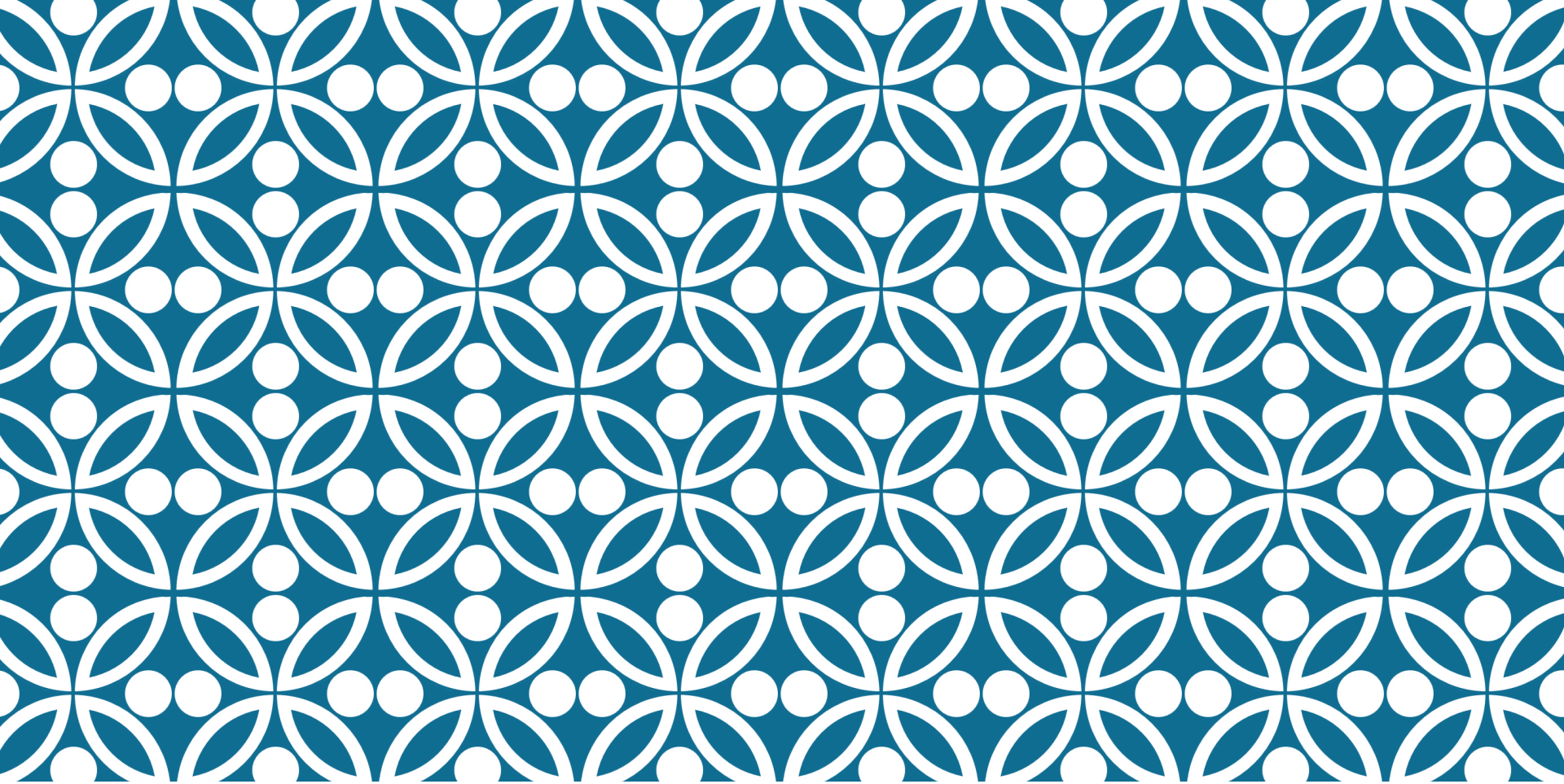
Limites Algorítmicos

- Falta de Paralelismo (código sequencial/concorrência)
- Frequência de Comunicação
- Frequência de Sincronização
- Escalonamento Deficiente (granularidade das tarefas/balanceamento de carga)

COMO MEDIR O DESEMPENHO?

Qual carga de trabalho?

Qual métrica representa melhor o desempenho?



MÉTRICAS DE DESEMPENHO

MÉTRICAS DE DESEMPENHO PARA APLICAÇÕES PARALELAS

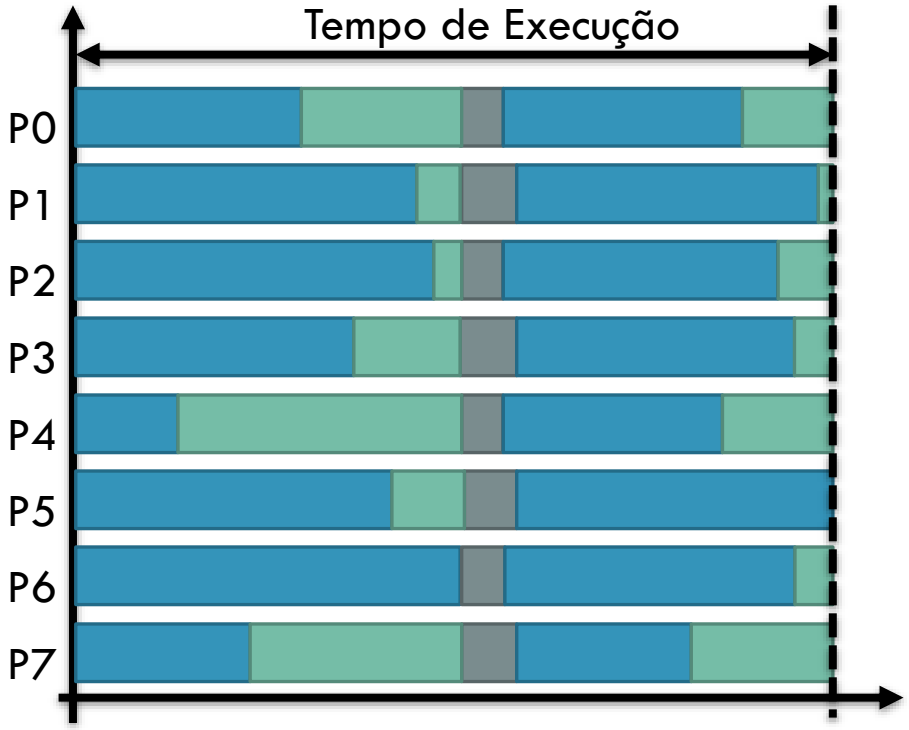
Existem várias medidas que permitem medir/avaliar o desempenho duma aplicação paralela. As mais conhecidas são:

- **Speedup** (*), **Overhead**, **Eficiência**, Custo
- Isoeficiência, Scaled speedup, Redundância, Utilização, Qualidade, Métrica de Karp-Flatt, outras

Existem igualmente várias leis/métricas que permitem balizar o comportamento duma aplicação paralela face ao seu potencial desempenho. As mais conhecidas são:

- **Lei de Amdahl**, Lei de Gustafson-Barsis, Lei de Grosch, Conjectura de Minsky, outras

FONTES DE OVERHEAD EM PROGRAMAS PARALELOS

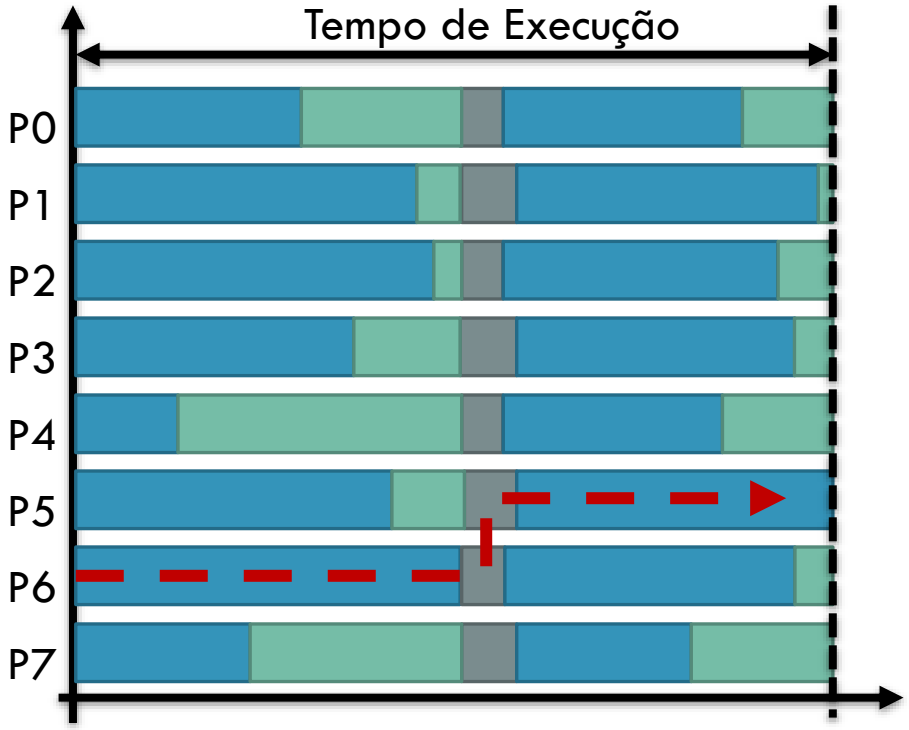


Processo inativo

Comunicação inter-processadores

Computação essencial ou excessiva

FONTES DE OVERHEAD EM PROGRAMAS PARALELOS



Inativo: Acontece devido a desbalanceamento e/ou sincronização

Comunicação: Trata-se de troca de mensagens ou acesso a dados compartilhados e sincronização

Processamento: O melhor algoritmo sequencial pode ser difícil ou impossível de ser paralelizado. A diferença de trabalho entre o processor paralelo e o serial é chamado é considerada excessivo. A inicialização/divisão do trabalho também pode significar overhead.

Caminho Crítico: O caminho que define o maior tempo de execução

OVERHEAD

O overhead (T_O) pode ser medido como a diferença entre o tempo sequencial e o a soma de todos os (p) tempos paralelo (com todos os sobre-custos)

$$T_O = p \times T_p - T_s$$

T_s é o tempo de execução do algoritmo sequencial

T_p é o tempo de execução do algoritmo paralelo com p processadores

SPEEDUP

O speedup é uma medida do grau de desempenho. O speedup mede a razão entre o tempo de execução de duas soluções que resolvem o mesmo problema.

$$Speedup = \frac{T_{antigo}}{T_{novo}}$$

T_{antigo} é o tempo de execução de uma solução base

T_{novo} é o tempo de execução da nova solução proposta

SPEEDUP EM COMPUTAÇÃO PARALELA

O speedup é uma medida do grau de desempenho. O speedup mede a razão entre o tempo de execução sequencial e o tempo de execução em paralelo.

$$S(p) = \frac{T_p(1)}{T_p(p)}$$

$T_p(1)$ é o tempo de execução com um processador

$T_p(p)$ é o tempo de execução com p processadores

	1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs
T(p)	1000	520	280	160	100
S(p)	**1**	**1,92**	**3,57**	**6,25**	**10,00**

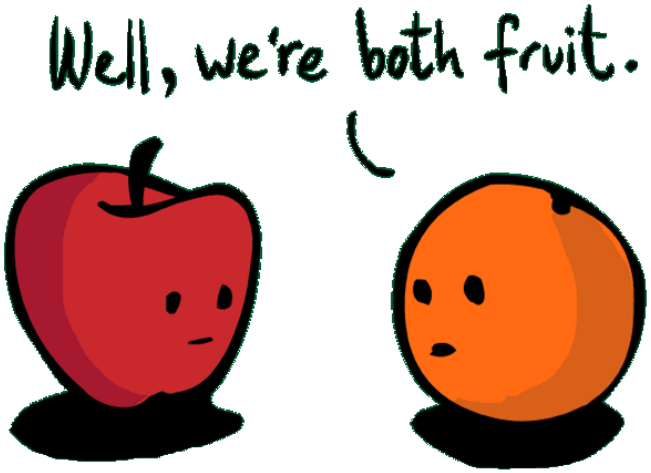
SPEEDUP - ARMADILHAS

Qual o speedup do $T_{odd-even}(4)$ considerando os seguintes tempos:

$T_{bubble-sort}(1) = 150$ seg.

$T_{quick-sort}(1) = 30$ seg.

$T_{odd-even}(1) = 120$ seg. → $T_{odd-even}(4) = 40$ seg.



SPEEDUP - ARMADILHAS

Qual o speedup do $T_{odd-even}(4)$ considerando os seguintes tempos:

$T_{bubble-sort}(1) = 150$ seg.

$T_{quick-sort}(1) = 30$ seg.

$T_{odd-even}(1) = 120$ seg. → $T_{odd-even}(4) = 40$ seg.

$$S = \frac{T_{bubble-sort}(1)}{T_{odd-even}(4)} = 3.75$$ (Enganoso)

$$S = \frac{T_{odd-even}(1)}{T_{odd-even}(4)} = 3.00$$ (Justo, compara o mesmo algoritmo)

$$S = \frac{T_{quick-sort}(1)}{T_{odd-even}(4)} = 0.75$$ (Justo, compara com o melhor algoritmo)

SPEEDUP SUPERLINEAR

"Se um único processador consegue resolver um problema em N segundos, podem N processadores resolver o mesmo problema em tempo $\begin{cases} > 1 segundo \\ = 1\ segundo \\ < 1\ segundo \end{cases}$?"

SPEEDUP SUPERLINEAR

O speedup diz-se superlinear quando a razão entre o tempo de execução sequencial e o tempo paralelo com p processadores é maior do que p.

$$\frac{T(1)}{T(p)} \geq \mathrm{p}$$

Alguns dos fatores que podem fazer com que o speedup seja superlinear são: ???

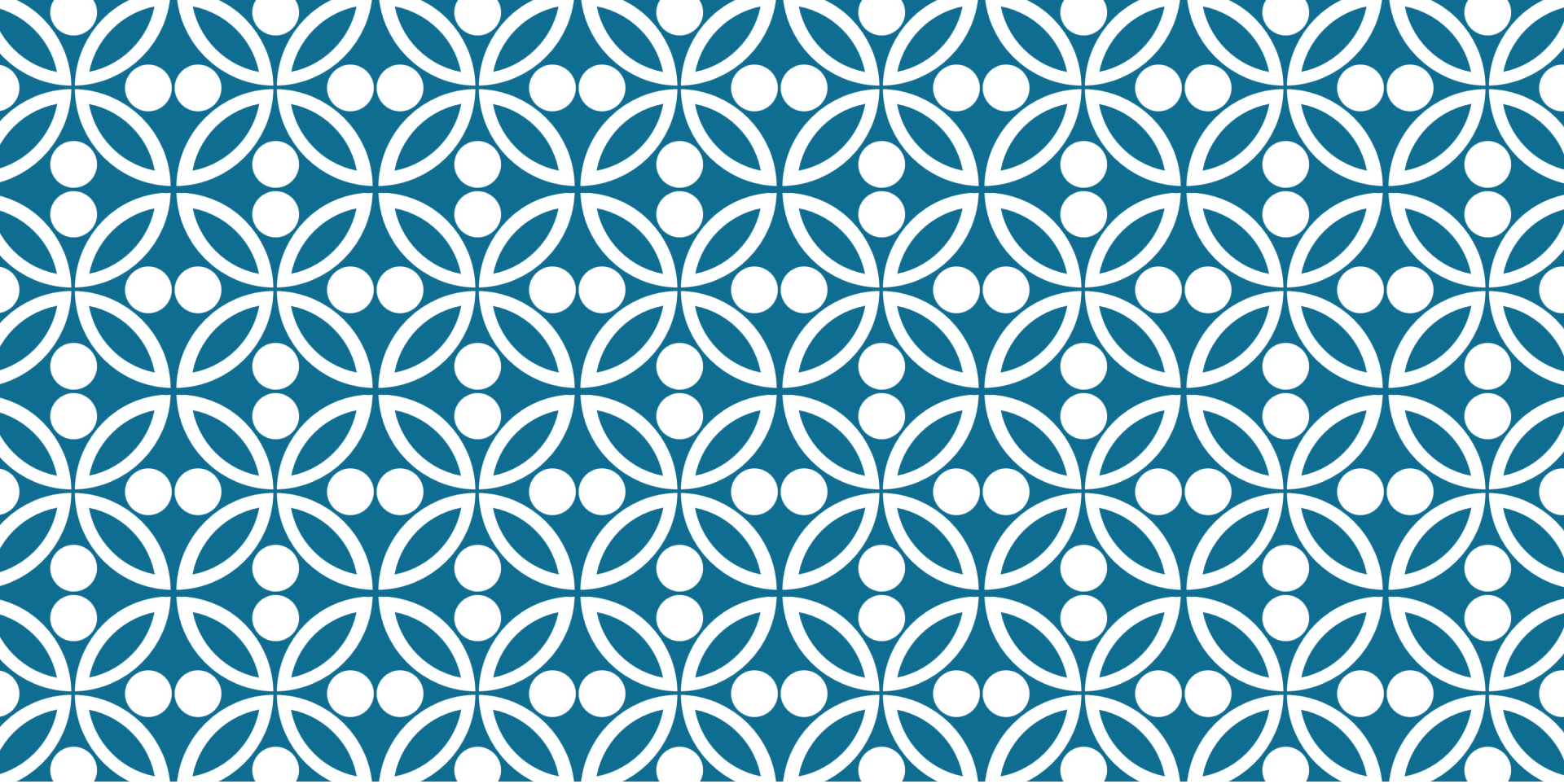
SPEEDUP SUPERLINEAR

O speedup diz-se superlinear quando a razão entre o tempo de execução sequencial e o tempo paralelo com p processadores é maior do que p.

$$\frac{T(1)}{T(p)} \geq \mathrm{p}$$

Alguns dos fatores que podem fazer com que o speedup seja superlinear são:

- Aumento da capacidade de memória (o problema passa a caber todo em memória).
- Subdivisão do problema (tarefas menores geram menos cache misses).
- Aleatoriedade da computação em problemas de optimização ou com múltiplas soluções (ex. achar solução para o problema de n-rainhas).



EFICIÊNCIA E ESCALABILIDADE

EFICIÊNCIA

A eficiência é uma medida do grau de aproveitamento dos recursos computacionais. A eficiência mede a razão entre o grau de desempenho e os recursos computacionais disponíveis.

$$\mathrm{E}(p) = \frac{S(p)}{p} = \frac{T_p(1)}{p \times T_p(p)}$$

S(p) é o speedup para p processadores

	1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs
T(p)	1000	520	280	160	100
S(p)	1	1,92	3,57	6,25	10,00
E(p)	**1**	**0,96**	**0,89**	**0,78**	**0,63**

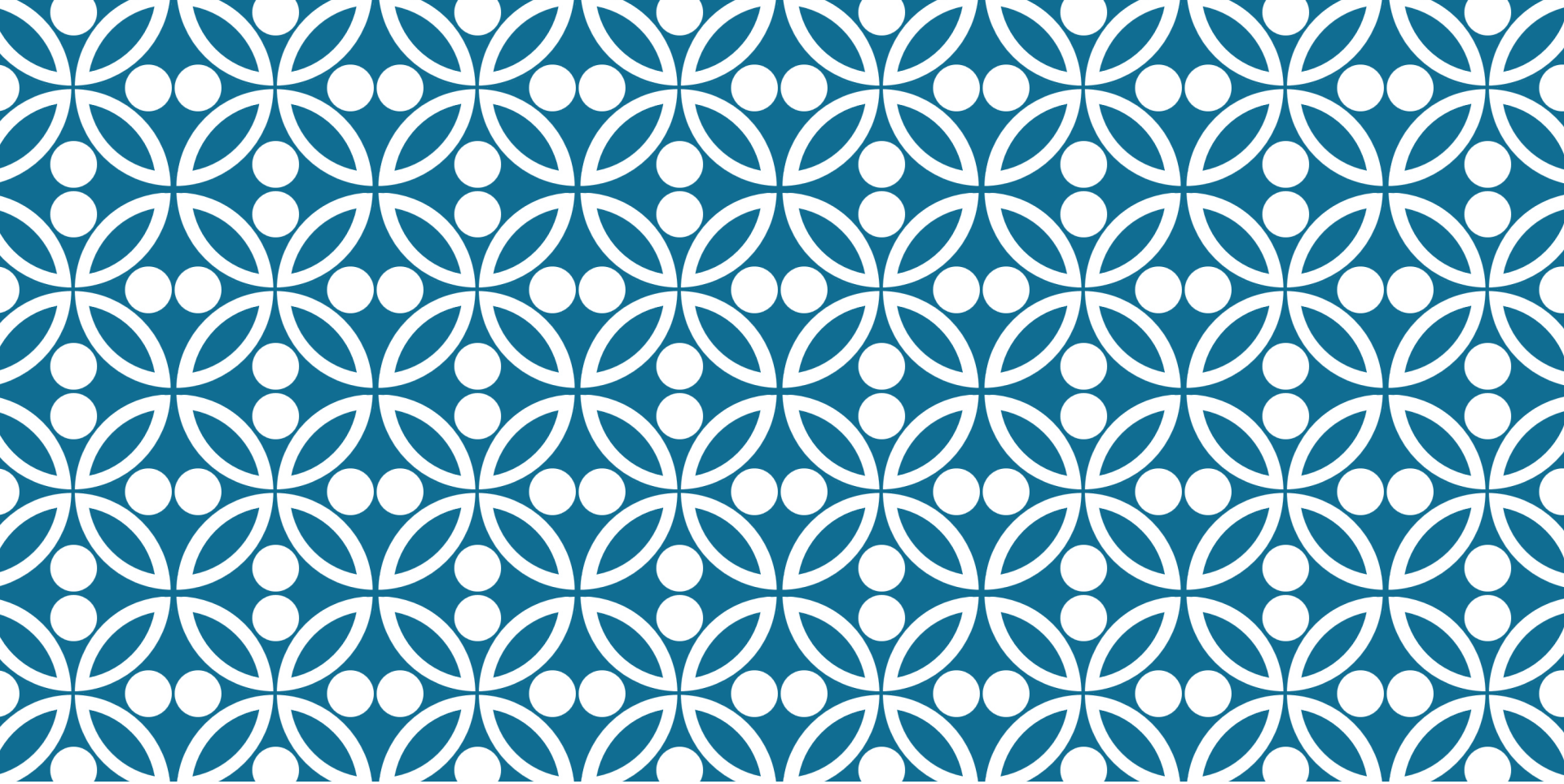
EFICIÊNCIA E ESCALABILIDADE

$$\text{Eficiência}(p) = \frac{S(p)}{p}$$

Uma aplicação é dita de **fortemente escalável** quando demonstra a capacidade de manter a **mesma eficiência** à medida que o número de processadores aumenta (P).

Uma aplicação é dita de **fracamente escalável** quando demonstra a capacidade de manter a **mesma eficiência somente quando** o número de processadores (P) e a dimensão do problema (N) aumentam proporcionalmente.

		1 CPU	2 CPUs	4 CPUs	8 CPUs	16 CPUs
Eficiência	N=10.000	1	0,81	0,53	0,28	0,16
	N=20.000	1	0,94	0,80	0,59	0,42
	N=40.000	1	0,96	0,89	0,74	0,58



LEI DE AMDAHL (1967)

LEI DE AMDAHL

A computação realizada por uma aplicação paralela pode ser divididas em 3 classes:

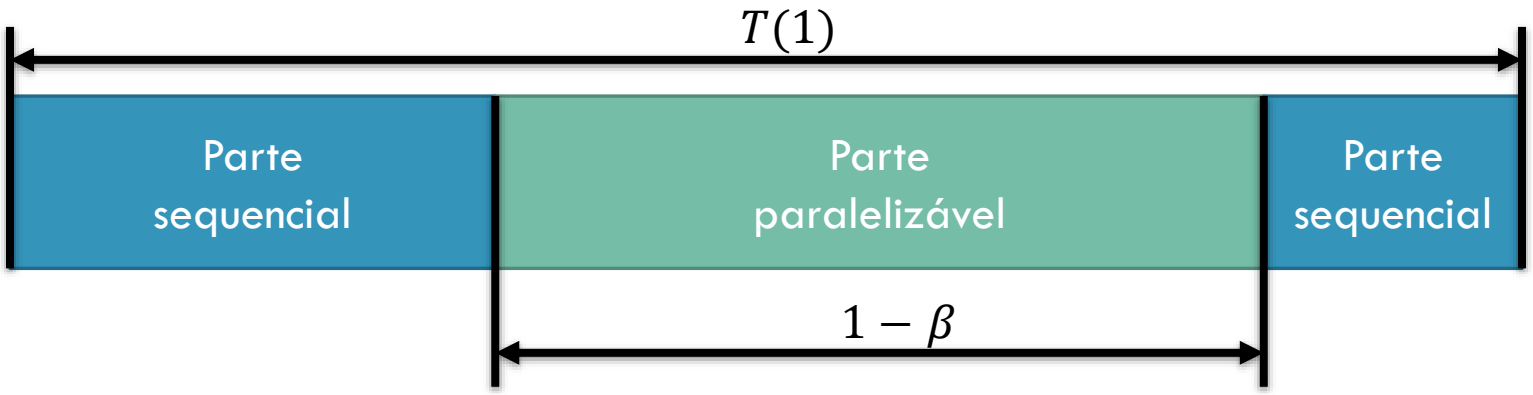
- C(seq): computações que só podem ser realizadas sequencialmente.
- C(par): computações que podem ser realizadas em paralelo.
- C(com): computações de comunicação/sincronização/iniciação.

Usando estas 3 classes, o speedup de uma aplicação pode ser definido do seguinte modo:

$$S(p) = \frac{T(1)}{T(p)} = \frac{C(seq) + C(par)}{C(seq) + \frac{C(par)}{p} + C(com)}$$

LEI DE AMDAHL

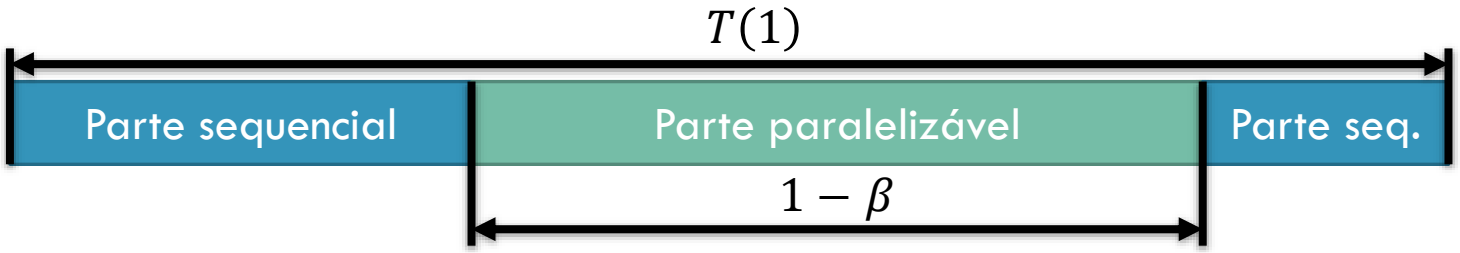
Tempo de execução em um único processador:



$\beta =$ fração de código que é puramente sequencial

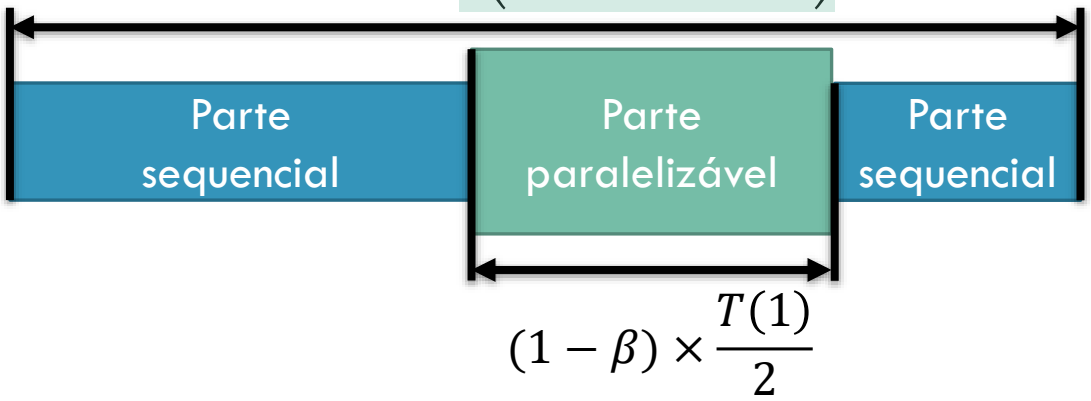
LEI DE AMDAHL

Tempo de execução em um único processador



$$T(2) = T(1) \times \beta + \left((1-\beta) \times \frac{T(1)}{2} \right)$$

Tempo de execução em 2 processadores



LEI DE AMDAHL

Seja $0 \leq \beta \leq 1$ a fração da computação que só pode ser realizada sequencialmente.

A lei de Amdahl diz-nos que o speedup máximo que uma aplicação paralela com p processadores pode obter é:

$$S(p) = \frac{1}{\beta + \frac{(1-\beta)}{p}}$$

A lei de Amdahl também pode ser utilizada para determinar o limite máximo de speedup que uma determinada aplicação poderá alcançar independentemente do número de processadores a utilizar (limite máximo teórico).

EXERCÍCIO: LEI DE AMDAHL

Suponha que pretende determinar se é vantajoso desenvolver uma versão paralela de uma determinada aplicação sequencial.

Por experimentação, verificou-se que 90% do tempo de execução é passado em procedimentos que se julga ser possível paralelizar.

Qual é o speedup máximo que se pode alcançar com uma versão paralela do problema executando em 8 processadores?

Qual é o speedup máximo considerando infinitos processadores?

$$S(p) = \frac{1}{\beta + \frac{(1-\beta)}{p}}$$

LEI DE AMDAHL

Suponha que pretende determinar se é vantajoso desenvolver uma versão paralela de uma determinada aplicação sequencial. Por experimentação, verificou-se que 90% do tempo de execução é passado em procedimentos que se julga ser possível paralelizar. Qual é o speedup máximo que se pode alcançar com uma versão paralela do problema executando em 8 processadores?

$$S(p) \leq \frac{1}{0{,}1+\frac{(1-0{,}1)}{8}} \approx 4{,}71$$

Qual é o speedup máximo considerando infinitos processadores?

$$\lim_{p\to\infty}\left(\frac{1}{0{,}1+\frac{(1-0{,}1)}{p}}\right) = 10$$

0

LIMITAÇÕES DA LEI DE AMDAHL

A lei de Amdahl ignora o custo das operações de comunicação/sincronização associadas à introdução de paralelismo numa aplicação.

Por esse motivo, a lei de Amdahl pode resultar em predições pouco realistas para determinados problemas.

A lei de Amdahl também supõe que o problema se manterá inalterado ao aumentar os recursos computacionais.

Ou seja, considera que a porcentagem puramente sequencial se manterá mesmo modificando o tamanho do problema.