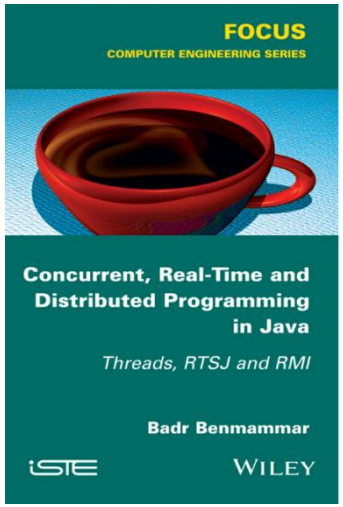
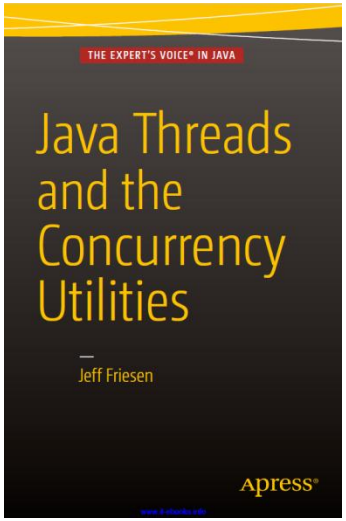
Programação Paralela e Concorrente

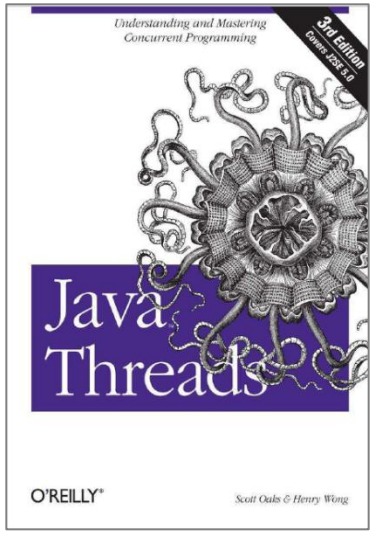
Unidade 1 - Introdução

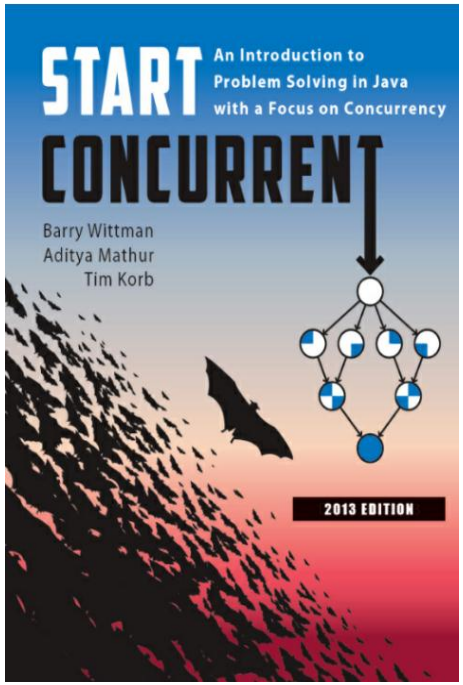
Prof. Aparecido V. de Freitas
Doutor em Engenharia
da Computação pela EPUSP
aparecidovfreitas@gmail.com

Bibliografia





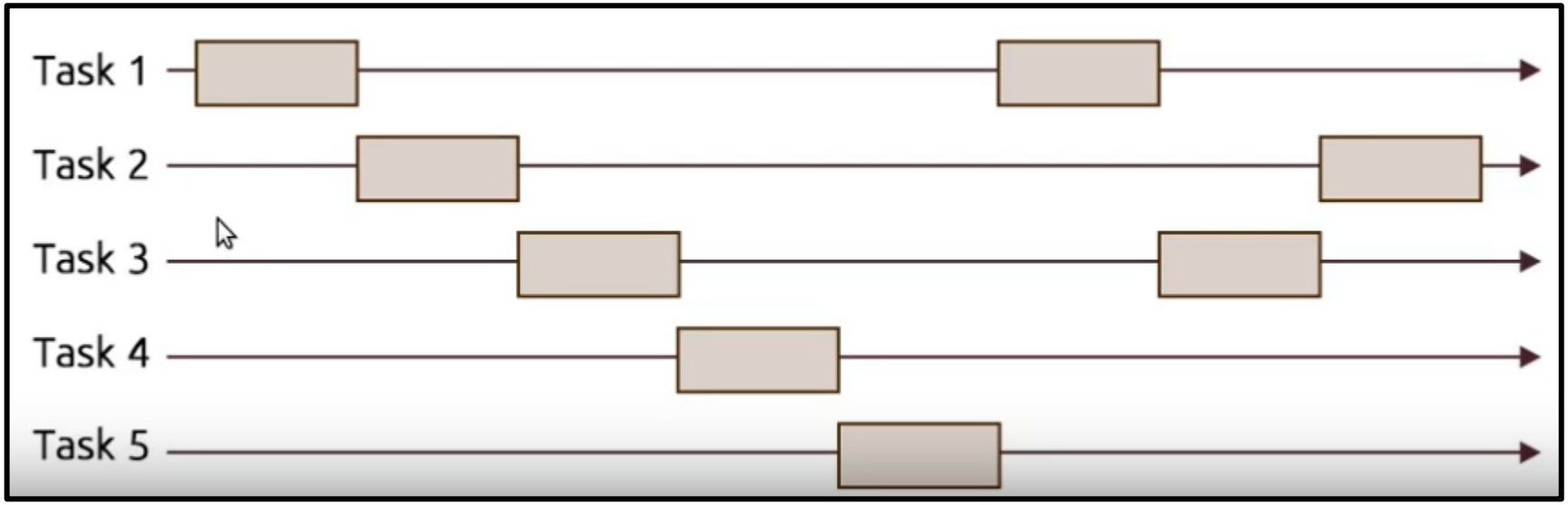




O que é Programação Concorrente ?

Programação Concorrente

- Paradigma de programação no qual o programa pode ser **decomposto** em **partes** que podem ser executadas de **forma não sequencial**.



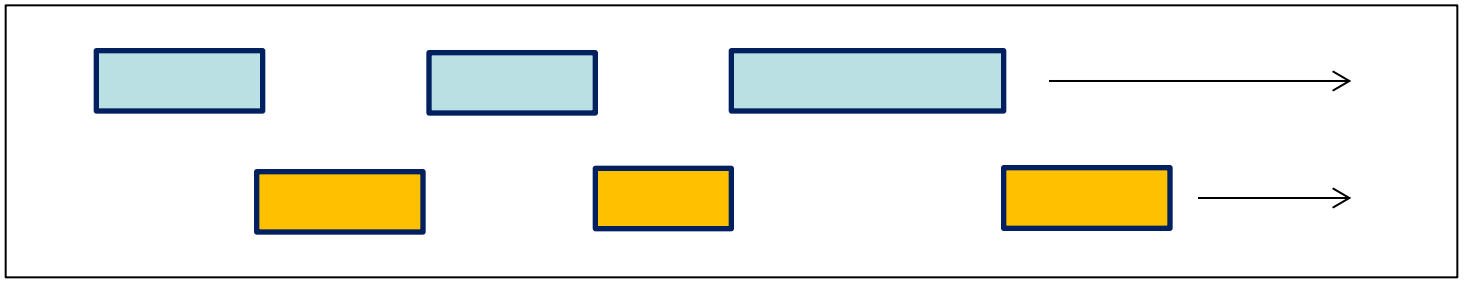
APARECIDOFREITAS

Socrative Student

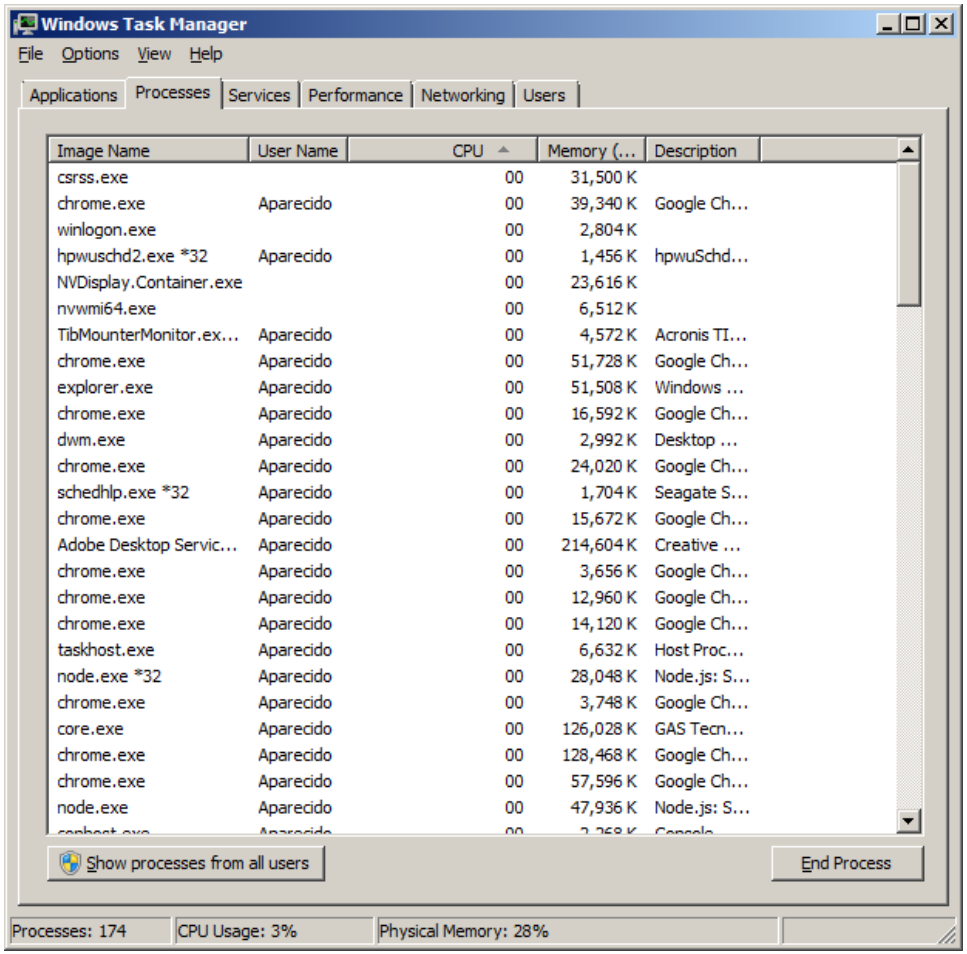
É possível criar-se Programas Concorrentes em máquinas com apenas 1 processador ?

Programação Concorrente

- **Não** necessariamente requer múltiplos processadores;
- Tarefas iniciam, executam e são completadas em períodos de tempo intercalados (**overlapping**);
- Com apenas **um** processador, tem-se uma abstração de paralelismo (pseudo paralelismo).



Programação Concorrente



- É por meio da **Concorrência** que uma máquina com apenas um processador ou apenas um núcleo (core) consegue processar concorrentente diversos processos "ao mesmo tempo".

APARECIDOFREITAS

Socrative Student

Porque estudar Programação Concorrente ?

Porque estudar Programação Concorrente ?

- O mundo à nossa volta funciona de modo **concorrente**;
- Exemplo: **Ao mesmo tempo** estamos...
  - Respirando;
  - Lendo;
  - Escrevendo;
  - Assistindo a esta aula, etc....

Porque estudar Programação Concorrente ?

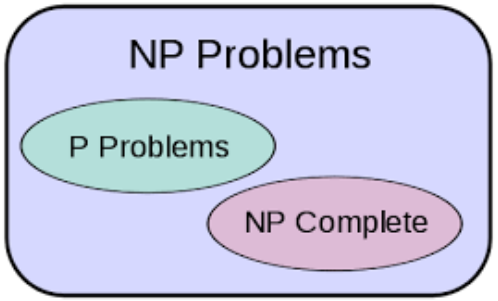
- **Processadores** disponíveis atualmente:
  - ✓ Muito rápidos;
  - ✓ Relativamente baratos; (custo/benefício)
  - ✓ Podem ser utilizados em paralelo

- Pode-se evitar "**ociosidade**" de **Hardware**;

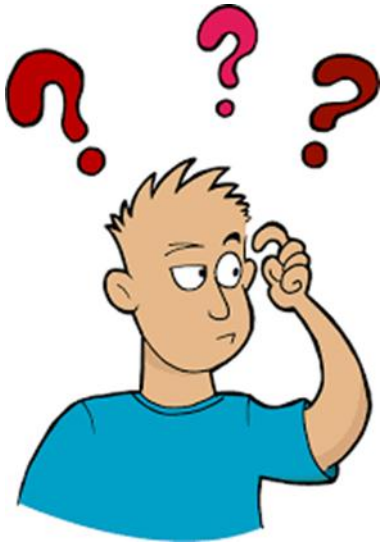
Porque estudar Programação Concorrente ?

- Redução do **tempo** total de processamento de um programa:
  - ✓ O **desempenho** de um programa **pode** ser melhorado caso seja executado em **mais** de um processador;

- Determinados **problemas** demoram muito para serem resolvidos de forma sequencial, seja devido ao **tempo de execução** muito longo ou à necessidade de se alocar muita **memória**.



Que problemas são viáveis para Programação Concorrente?



Que problemas são viáveis para Programação Concorrente?

- **Física Astronômica:** Movimentação de Corpos Celestes;
- **Meteorologia**: Previsão do Tempo;
- **Genética**: Genoma Humano;
- **Farmacologia**: Produção de novos medicamentos;
- Processamento de **Imagens**;
- **Big Data.**

APARECIDOFREITAS

Socrative Student

Os computadores atuais também trabalham de forma concorrente?

Os computadores atuais também trabalham de forma concorrente?

- Ao **mesmo tempo**, computadores:
  - **Imprimem** um documento;
  - **Exibem** na **tela** um texto;
  - **Salvam** informações em **disco**, etc

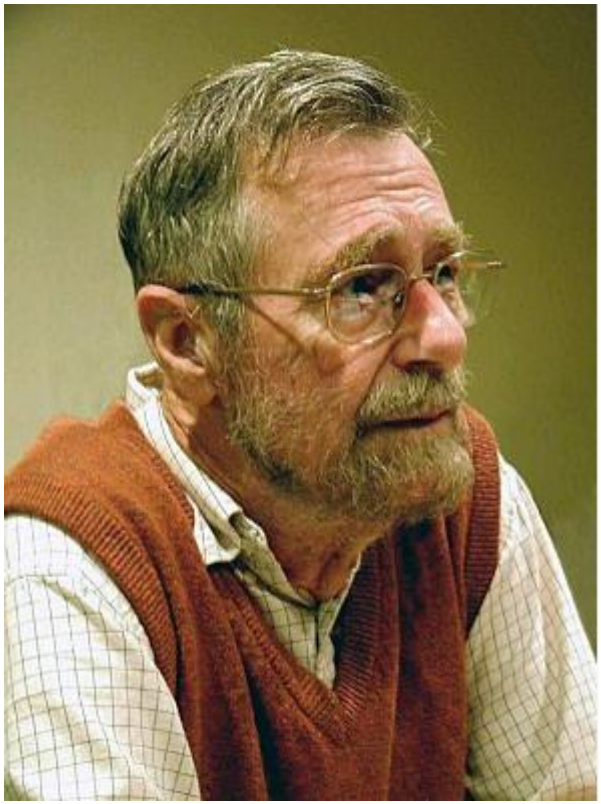
APARECIDOFREITAS

Socrative Student

Desde quando se estuda Programação Concorrente?

Histórico da Programação Concorrente

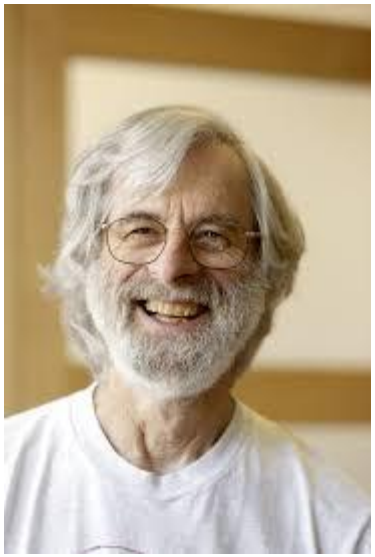
- **1968** – E. W. **Dijkstra**: **Cooperando Processos** Sequenciais;
- **1971** – E. W. **Dijkstra**: Ordem Hierárquica de **Processos** Sequenciais;
- **1973** – C. L. **Liu** e J. W. **Layland**: Algoritmos de Escalonamento para **Multiprogramação** em Ambiente de Tempo Real.



Histórico da Programação Concorrente

- **1974** – Leslie **Lamport**: Uma nova solução para o problema da **programação concorrente** de Dijkstra;

- **1974** – C. A. R. **Hoare**: **Monitores** – Conceito para estruturar Sistemas Operacionais.



Histórico da Programação Concorrente

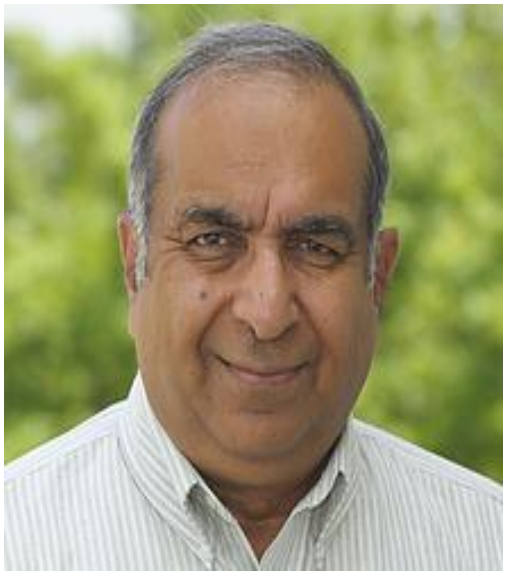
- **1976** – J. H. **Howard**: Provando **Monitores**;
- **1976** – S. **Owicki** e D. **Gries**: Verificando propriedades dos **Programas Paralelos**: uma abordagem axiomática;
- **1977** – P. Brinch **Hansen**: A arquitetura de **Programas Concorrentes**.

Histórico da Programação Concorrente

- **1978** – C. A. R. Hoare: Comunicação de Processos Sequenciais;
- **1978** – E. W. **Dijkstra**, L. Lamport, A. J. Martin, C. S. Sholten e E. F. Steffens: Um exercício em cooperação para "Garbage Collection";
- **1980** – E. W. **Dijkstra** e C. S. Sholten: Detecção e Terminação.

Histórico da Programação Concorrente

- **1981** – G. **Ricart** e A. **Agrawala**: Um algoritmo ótimo para Exclusão Mútua Distribuída;
- **1981**– G. L. **Peterson**: O problema da **Exclusão Mútua**;
- **1982** – J. **Misra** e K. M. **Chandy**: Detecção de terminação em Communicating Sequencial Processes.



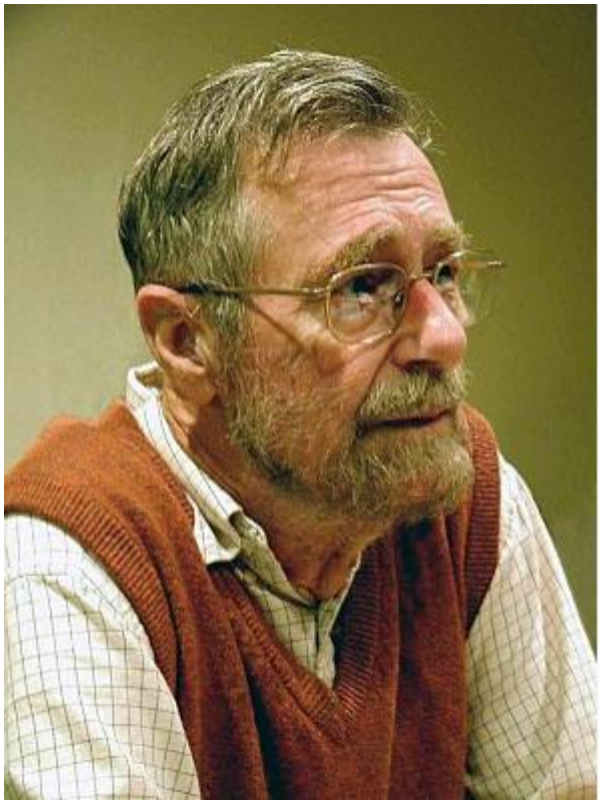
Histórico da Programação Concorrente

- **1983** – G. **L. Peterson**: Uma nova solução para o problema de **Programação Concorrente** de Lamport usando variáveis compartilhadas;
- **1983**– **DoD**, USA: Linguagem de Programação **Ada**;
- **1985** – D. Gelernter: A Linguagem **Linda**.

Em qual aplicação ocorreu o primeiro uso da Concorrência ?

Em qual aplicação ocorreu o primeiro uso da Concorrência ?

- **1968** – E. W. **Dijkstra**: **Cooperando Processos** Sequenciais;
- **1971** – E. W. **Dijkstra**: Ordem Hierárquica de **Processos** Sequenciais;
- **1973** – C. L. **Liu** e J. W. **Layland**: Algoritmos de Escalonamento para **Multiprogramação** em Ambiente de Tempo Real.



Em qual aplicação ocorreu o primeiro uso da Concorrência ?

- **Em Sistemas Operacionais**

Concorrência em Sistemas Operacionais

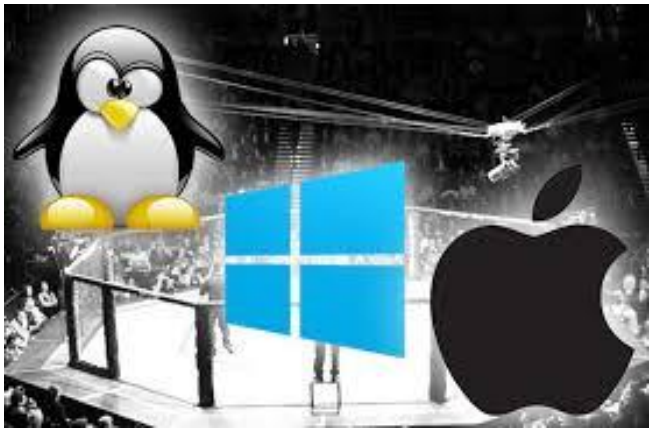
- Nos primeiros Sistemas Operacionais, o controle de Entrada e Saída **não** podia ser feito concorrentemente com outras operações;

- Mas, a **evolução** do hardware e dos SO's, fez surgir a **Concorrência**, retirando do processador principal, alguns microssegundos necessários para controlar **I/O**.

Concorrência em Sistemas Operacionais

- Por muitos anos, a programação concorrente foi considerada como um componente no estudo dos **Sistemas Operacionais**;

- Mas, felizmente, existem hoje diversas **linguagens de programação** onde se pode aplicar programação concorrente em desenvolvimento de aplicações.



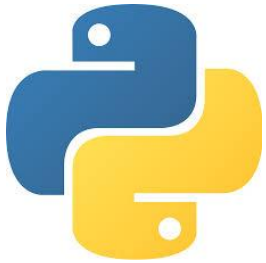
APARECIDOFREITAS

Socrative Student

Que linguagens disponíveis atualmente permitem o emprego de Programação Concorrente ?

Linguagens – Programação Concorrente











Quantas operações devem ser executadas ao se avaliar a expressão: (a*b + c*d**2)*(g+f*h) ?

Quantas operações devem ser executadas ao se avaliar a expressão: (a*b + c*d**2)*(g+f*h) ?

- Multiplicação (a * b)
- Exponenciação (d **2)
- Multiplicação (c * d ** 2)
- Adição (a*b + c*d**2)
- Multiplicação (f * h)
- Adição (g + f * h)
- Multiplicação (a * b + c * d** 2) * (g + f * h)

- A avaliação da expressão requer assim **7** operações.

Suponha que você tenha uma máquina com 64 processadores. Posso avaliar a a expressão usando 100% de paralelismo?

Suponha que você tenha uma máquina com 64 processadores. Posso avaliar a a expressão usando 100% de paralelismo?

- **Não**, pois as operações são **interdependentes**;

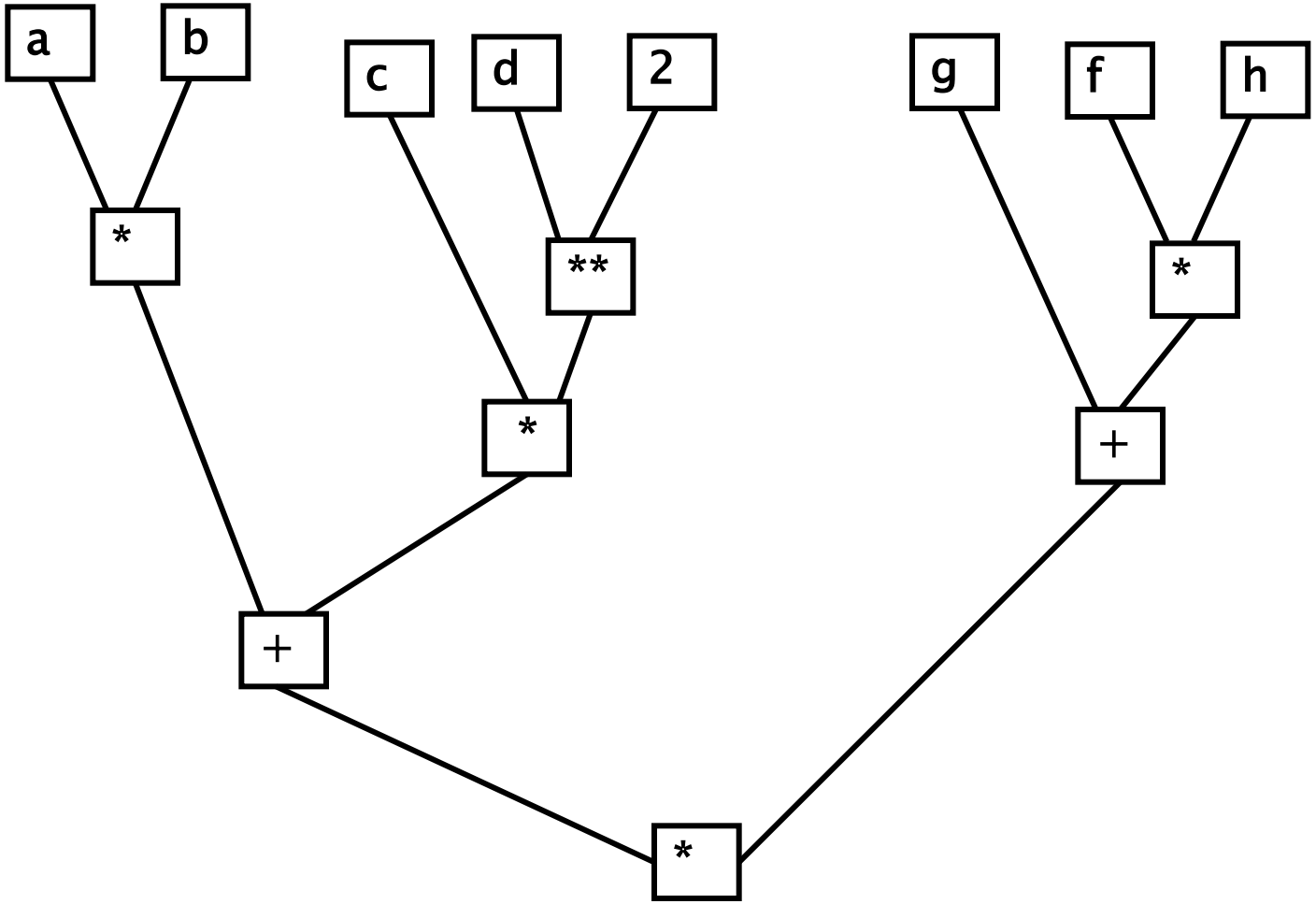
- Ou seja, não se consegue concorrência (paralelismo) total, em função da dependência das operações que provoca **sequencialidade**.

A expressão (a*b + c*d**2)*(g+f*h)

poderia ser avaliada de forma concorrente?

Se sim, de que forma ?

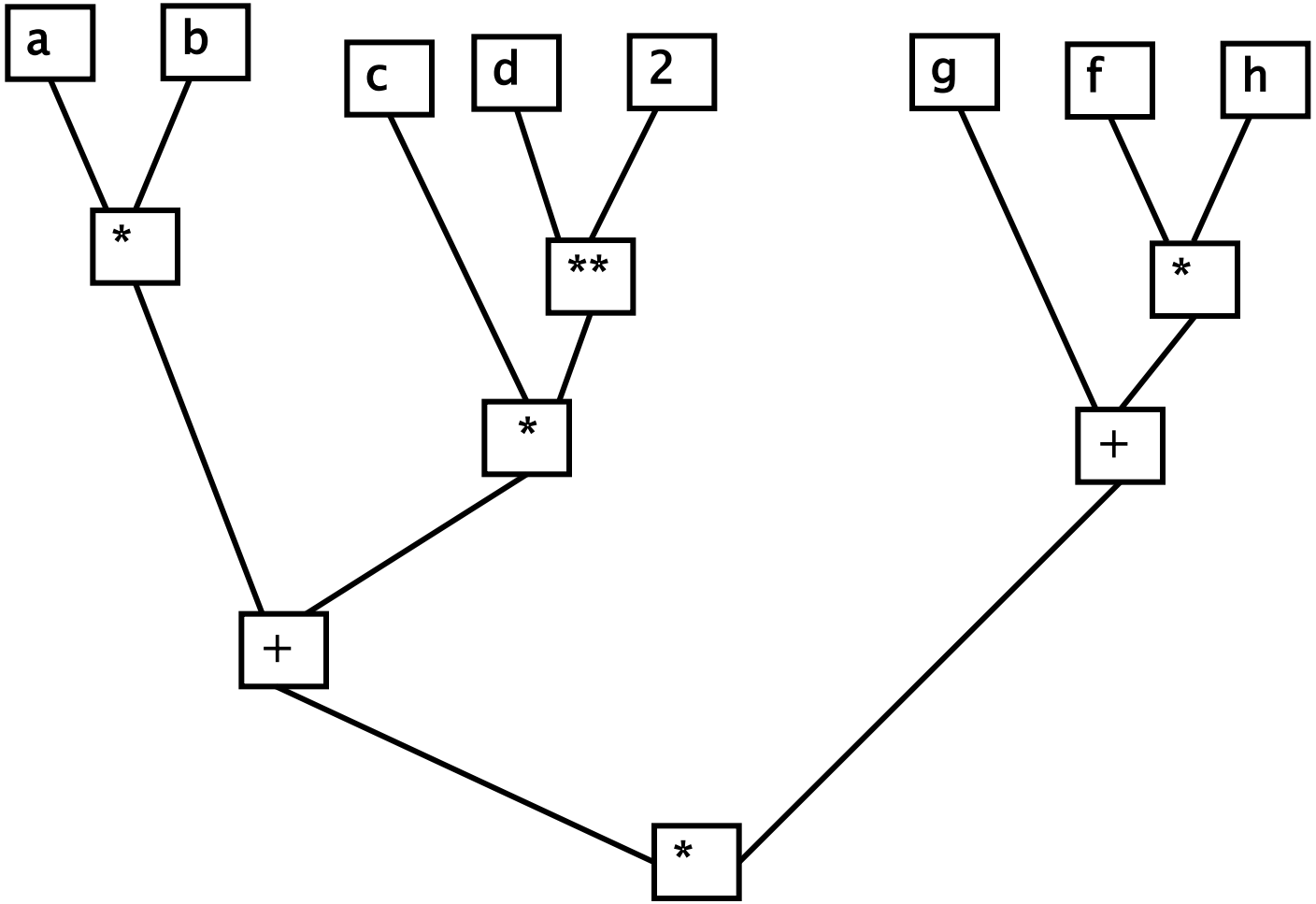
Árvore de Execução



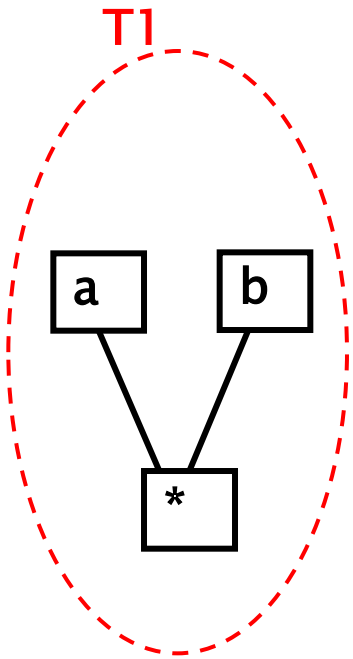
Qual o grau de interdependência da expressão

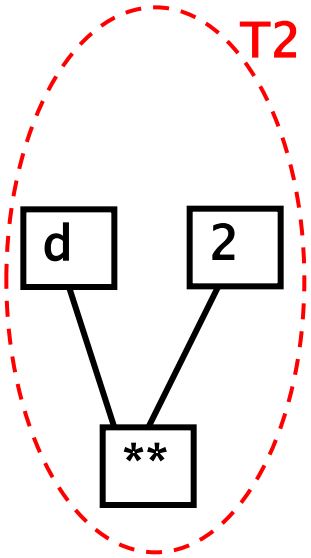
(a*b + c*d**2)*(g+f*h) ?

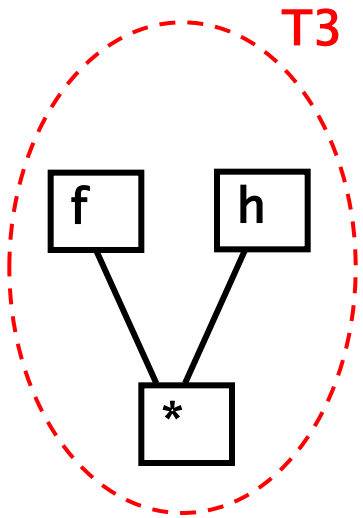
Árvore de Execução



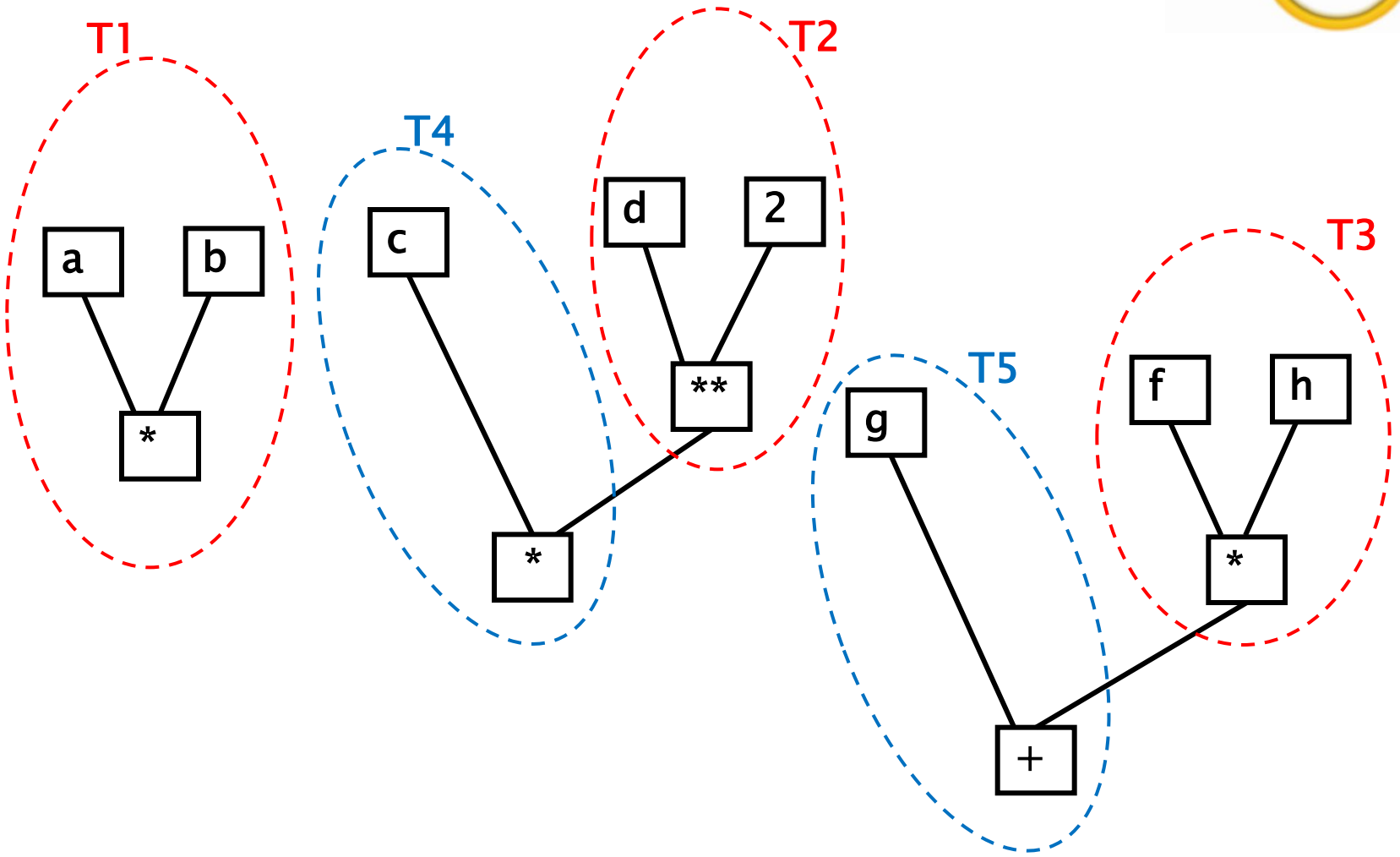
Grau de Interdependência







Grau de Interdependência



Grau de Interdependência

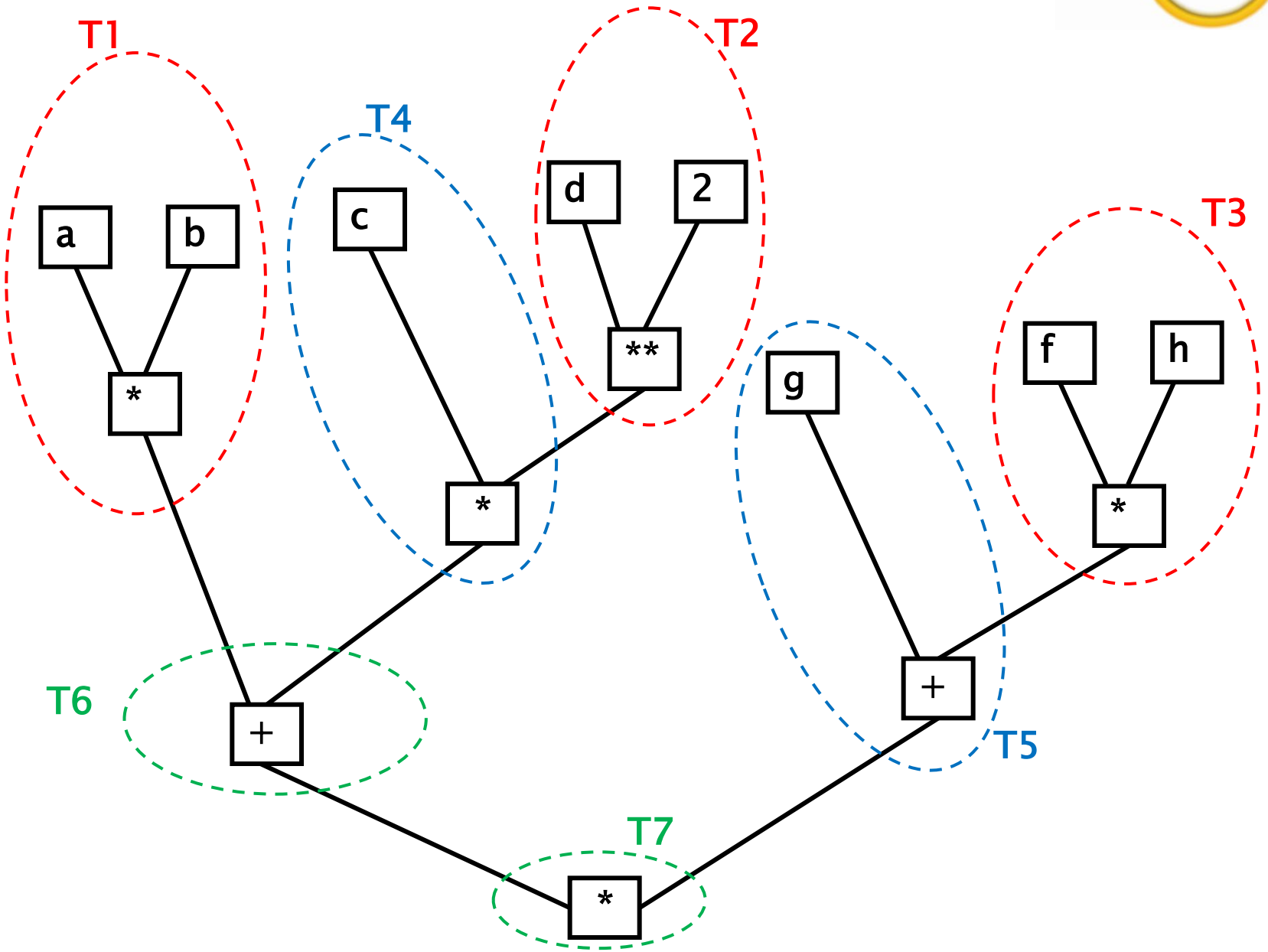
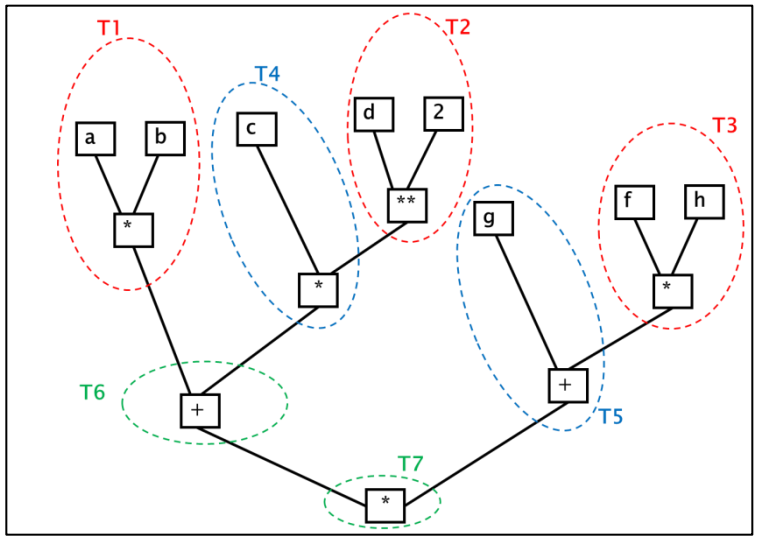
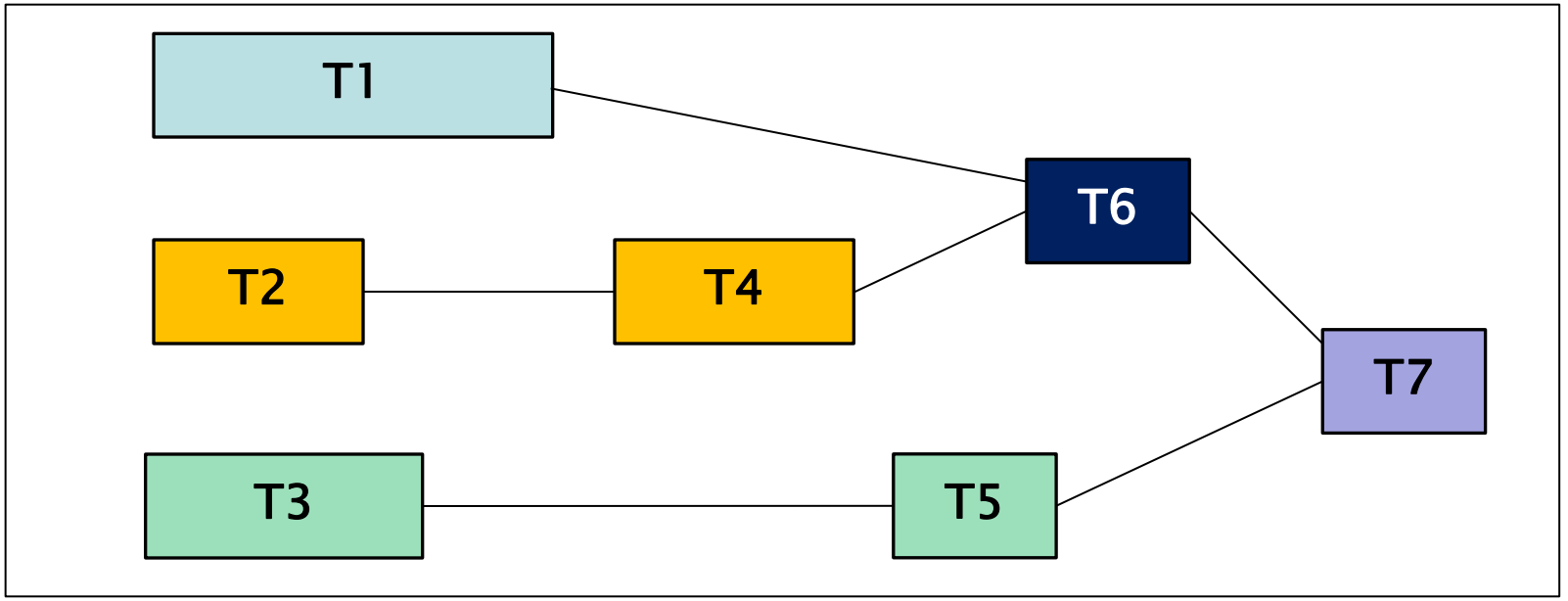


Diagrama de Fluxo





Concorrência em Programas

- Geralmente qualquer programa sequencial **não** é totalmente sequencial;
- Ao se avaliar as operações executadas pelo programa, usualmente pode-se estruturá-lo em **blocos** nos quais a concorrência pode ser implementada.

Exercício - 1

- **Desenhe** a **árvore** que representa a avaliação da expressão aritmética:

```
(a**2 + ( c + d ) * e ) + (f + g * h)
```

Exercício - 2

- Dada a expressão:

```
(a**2 + ( c + d ) * e ) + (f + g * h)
```

- Assumindo-se que cada operação gasta **1** milissegundo, qual o tempo necessário para se avaliar a expressão, considerando-se:

  A) Processamento sequencial;

  B) Processamento concorrente, assumindo-se que haja processadores suficientes para se obter a máxima concorrência.

Exercício - 3

- Dada a expressão:

```
(a*2  +  ( c**3 + d ) * e )  + (f * g + h**2)
```

- Assumindo-se que as operações gastam o sequinte tempo:
  - **Adição** – 1 milissegundo
  - **Multiplicação** – 2 milissegundos
  - **Exponenciação** – 4 milissegundos

- Qual o tempo necessário para se avaliar a expressão, considerando-se:

  A) Processamento sequencial;

  B) Processamento concorrente, assumindo-se que haja processadores suficientes para se obter a máxima concorrência.