Entendendo Concorrência e Paralelismo

GDG Ribeirão Preto

Ronaldo Faria Lima
CTO Chiclete Mkt

agenda

O que é concorrência?

O que é paralelismo?

O que é multi-processamento?

Juntando tudo

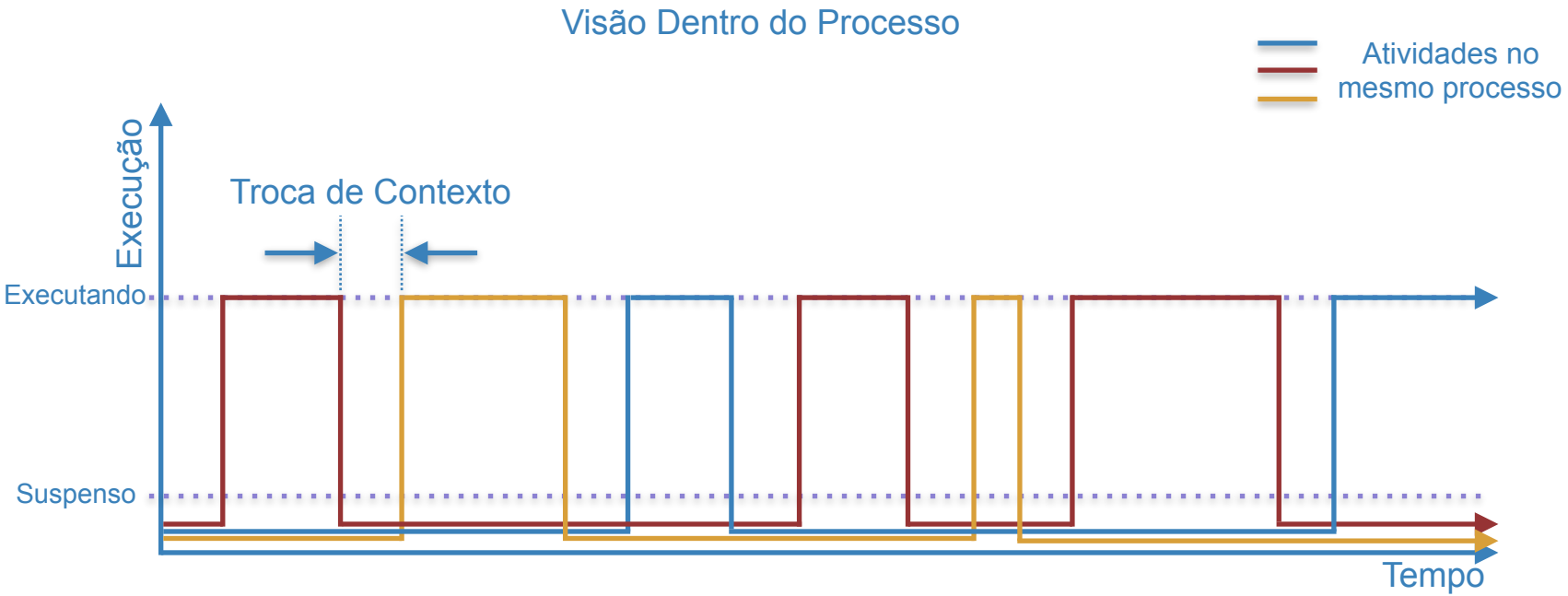
Material de Apoio

Dúvidas?

Qual a importância disso, afinal de contas?

Concorrência:
Responsabilidade por várias atividades

Concorrência



Vantagens

- Tudo acontece no mesmo processo.
- A troca de contexto é extremamente rápida e eficiente.
- Não há necessidade de sincronismo. A programação é bem mais simples.

Desvantagens

- A troca de contexto é responsabilidade do programador.
- A programação precisa ser colaborativa.
- Atividades altamente dependentes de CPU podem sofrer com a performance.
- O scheduler é de implementação complexa.

Quem usa concorrência?

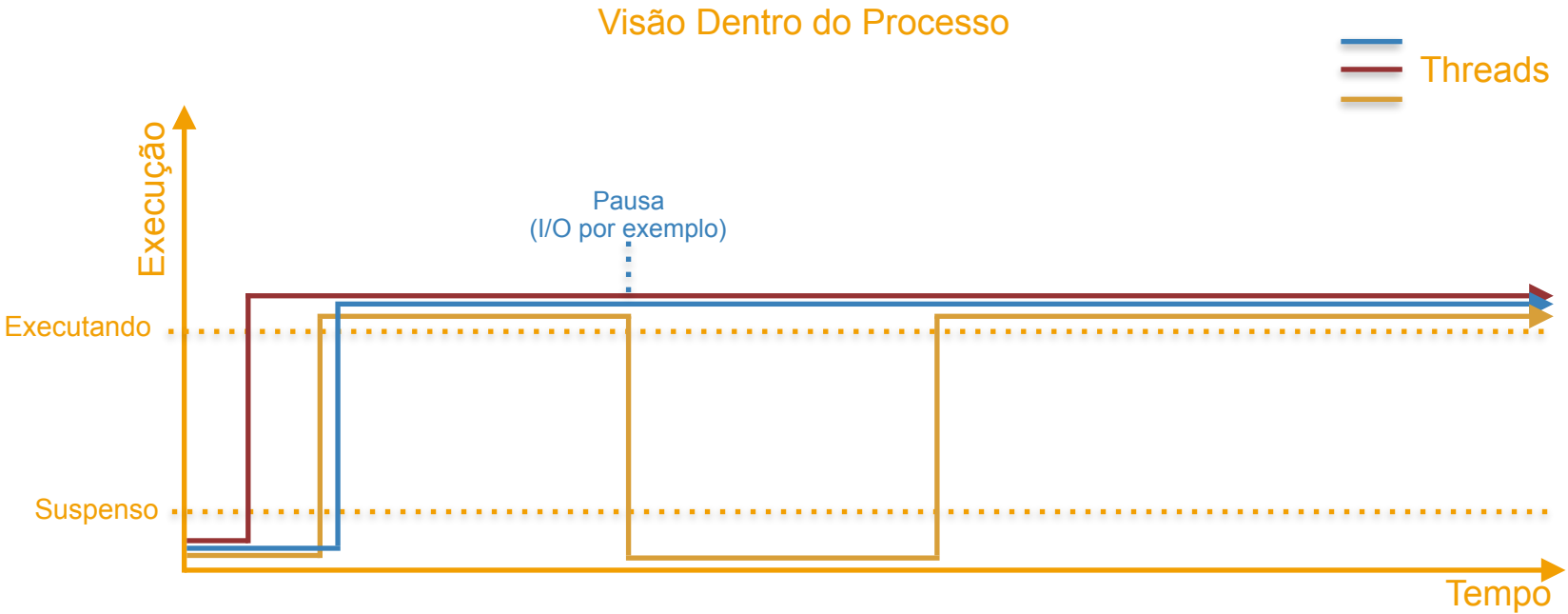
- Populares:
  - Go
  - NodeJS
  - Swift e Objective C
- Menos Conhecidas:
  - Alef
  - Crystal
  - Ease
  - FortranM
  - XC

Bibliotecas

- libdill e libmill em C
- Foundation para Swift e Objective C
- Python CSP
- Ruby CSP

Paralelismo:
Fazer muita coisa ao mesmo tempo

Paralelismo



Características

- A thread é uma entidade do kernel do sistema operacional.
- Quem cuida do scheduling é o sistema operacional.
- A troca de contexto está fora do controle do programador.
- A programação é assíncrona por natureza. Nunca se deve assumir quando algo acontece.
- O mesmo trecho de código pode executar ao mesmo tempo em várias threads diferentes.
- O paralelismo é determinado pela quantidade de processadores. Em sistemas com um único núcleo, as threads funcionam como um sistema baseado em co-rotinas, porém sem a simplicidade de processamento serial.

Vantagens

- Melhor aproveitamento de vários processadores
- Há um certo isolamento: uma thread não impede o funcionamento de outra.
- Não há necessidade de programação colaborativa.
- Ideal para processamento altamente dependente de CPU.

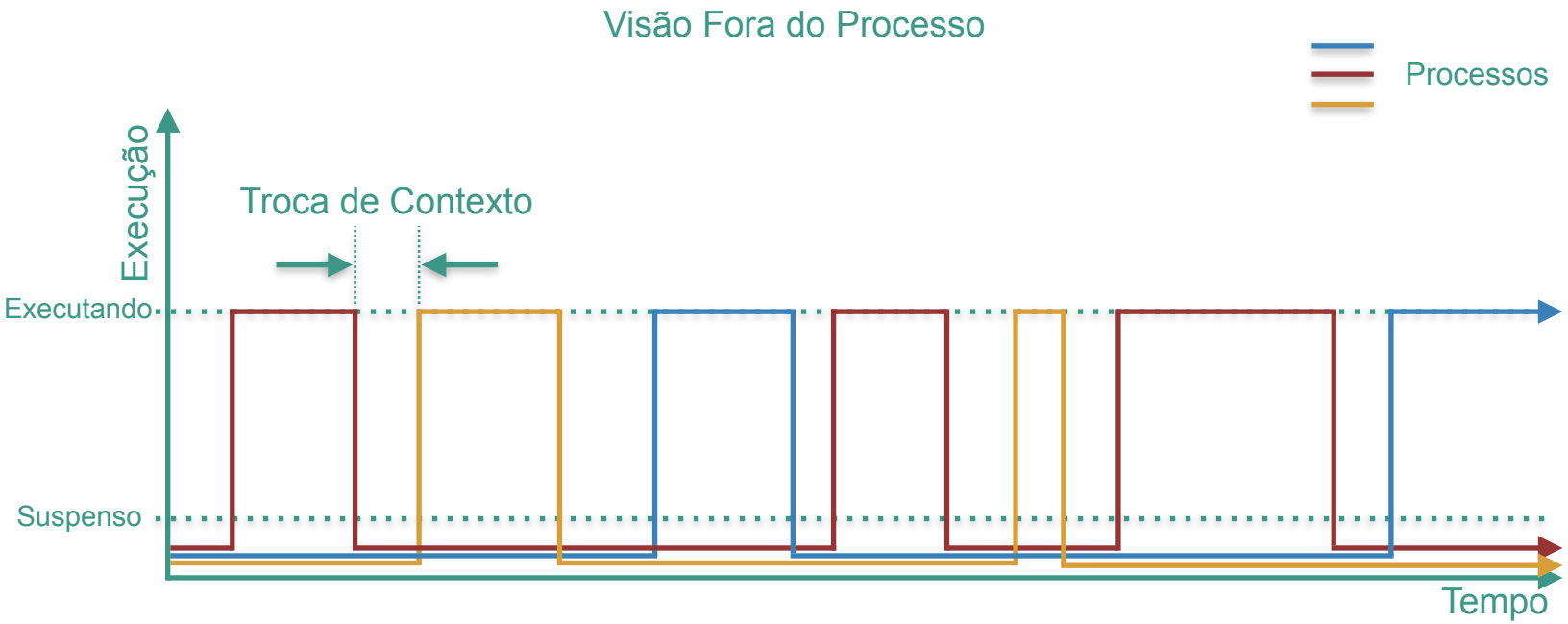
Desvantagens

- Modelo de programação complexo. Tudo acontece de forma assíncrona.
- Necessário sincronismo para proteção de invariantes usadas em várias threads.
- Sincronismo pode levar a dead-locks, o que leva um programa ao travamento.
- Não fica explícito no código o que está em paralelo e o que não está.
- Como tudo executa no mesmo processo, uma thread que corromper o address space do processo, corrompe todas as outras threads.

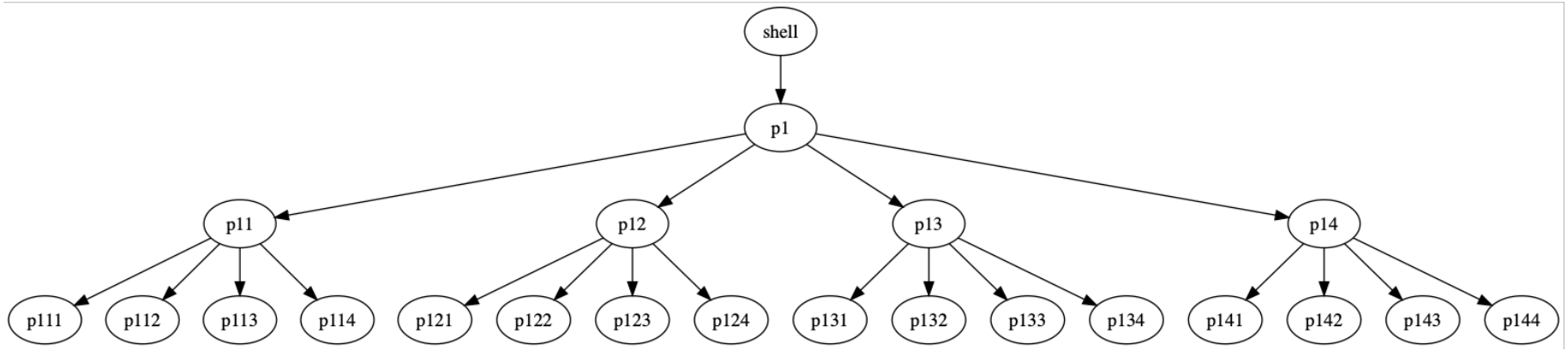
Nota Importante!
A “Seção Crítica”

Multi-processamento:
Paralelismo realmente isolado

Multi-processamento



A Árvore de Processos



Vantagens

- O isolamento entre processos torna o software menos suscetível a falhas.
- Atinge-se o paralelismo mesmo com código totalmente serial em cada processo.
- Não é necessária a colaboração entre os processos para que executem suas atividades em paralelo.

Desvantagens

- A criação de um processo é cara em termos de consumo de recursos.
- A troca de contextos ainda ocorre, sob controle do kernel do sistema operacional.
- A comunicação inter-processos é um desafio por si só.
- É necessário usar vários recursos do kernel do sistema operacional: semáforos, sinais, etc.

Juntando Tudo:
Levando sua CPU a 100% de uso

Como juntar tudo?

- Use concorrência quando houver muito I/O
- Use paralelismo em threads quando for necessário processar paralelamente dados compartilhados no mesmo processo.
- Use paralelismo em processos quando os dados puderem ser processados independentemente em paralelo.
- Combine os modelos de processamento para utilizar de forma mais eficiente os recursos do seu sistema.

Material de Apoio:
https://github.com/ronflima/processos

OBRIGADO!

GDG Ribeirão Preto