Programação Concorrente em Java

Aula 03

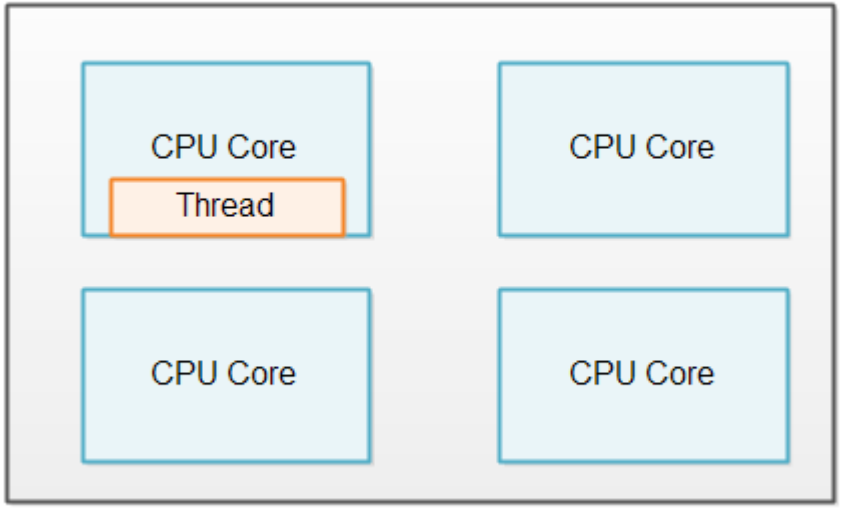
Same-Threading
Concorrência VS Paralelismo

Same-Threading

- Same-threading é um modelo de concorrência onde sistemas **single-thread** são escalonados para sistemas **N single-threads**.
- Isso resulta em N sistemas single-threads rodando em paralelo.
- Contêm múltiplas threads mas cada thread funciona sistema de single thread.

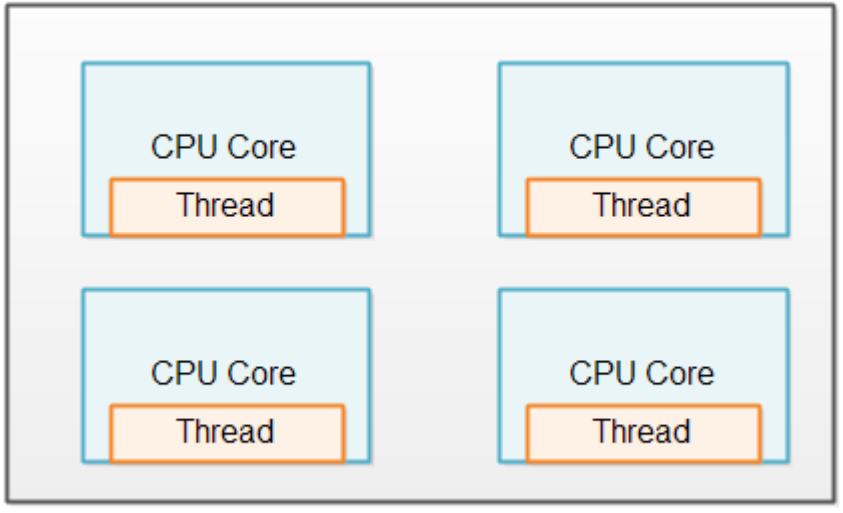
Single-thread

- São mais simples de implementar que sistemas de múltiplas threads.
- Não compartilham dados com outras threads, possibilitando o uso de estruturas de dados não concorrentes.



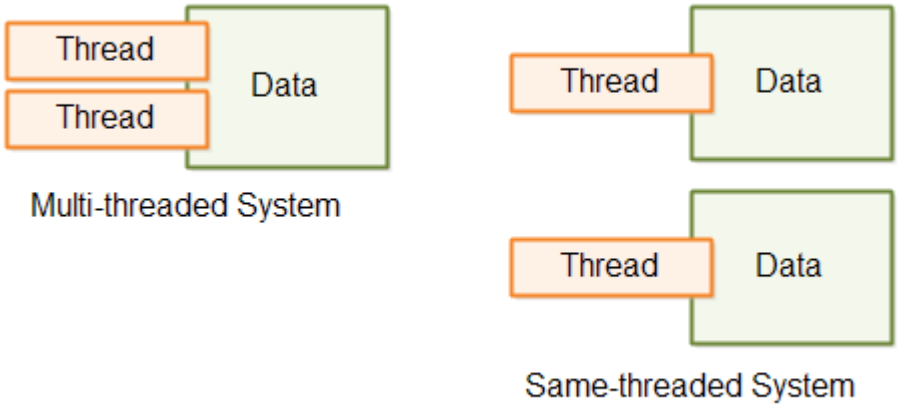
Multiple Same Threads

- Vário sistemas single-thread em CPUs diferentes.
- Uma thread por CPU.
- Paralelismo REAL



Sem estado compartilhado

- Um sistema same-thread não compartilha estado.
- Não há memória compartilhada.
- Não há concorrência entre estruturas de dados.

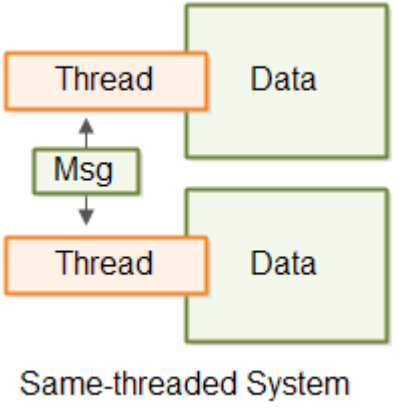


Distribuição de Carga

- Um sistema single-threaded deve distribuir a carga de trabalho entre as threads participantes.
- Como distribuir essa carga?
  - **Microservices**: cada micro serviço é um componente independente, tendo seu próprio estado e estruturas.
  - **Serviços com dados “despedaçados” (sharded)**: se o seu sistema realmente precisa de dados compartilhados, uma opção são os bancos de dados “sharded”. “Sharding” significa que os dados estão espalhados em vários databases.

Comunicação entre Threads

- Passagem de mensagens entre Threads.
- Uma thread A envia dados para uma thread B através da cópia dos mesmos. Não importa o que B faça com o dados, o estado de A não mudará.
- A comunicação pode ser feita:
  - Filas
  - Pipes
  - Sockets
  - Etc.

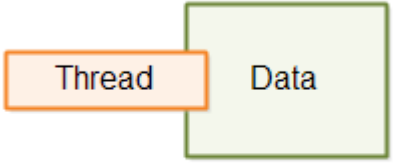


Same-threaded System

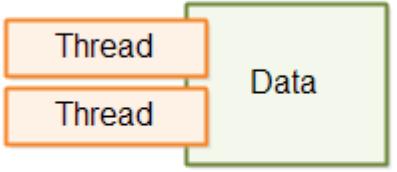
Modelo simples de concorrência

- Cada sistema executando sua única thread em um sistema same-threaded pode ser implementado como um single thread.
- O modelo interno de concorrência se torna muito simples comparado com o compartilhamento de dados.
- Você não precisa se preocupar com estruturas compartilhadas.

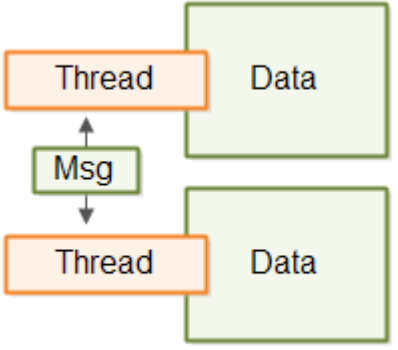
Recapitulando



Single-threaded System



Multi-threaded System



Same-threaded System

Concorrência VS Paralelismo

Concorrência VS Paralelismo

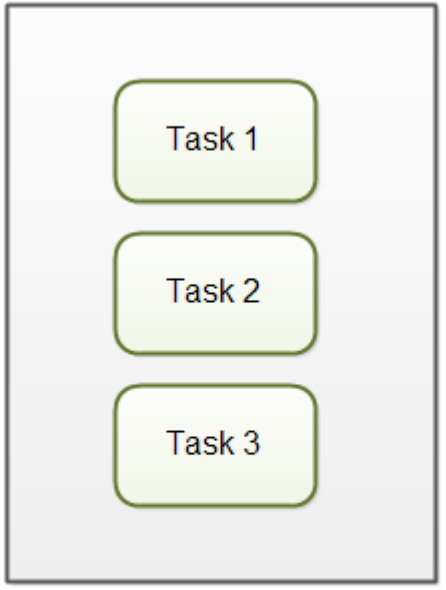
- Os termos concorrência e paralelismo geralmente são usados em sistemas multi-thread.
- Mas, serão eles sinônimos?
  - O que é concorrência?
  - O que é paralelismo?

Concorrência

- Significa que uma aplicação está fazendo progresso em mais de uma tarefa ao mesmo tempo (concorrentemente).
- Se o computador tem apenas uma CPU (com um único núcleo), na realidade a aplicação não estáo fazendo progresso exatamente ***ao mesmo tempo***.
- No entanto, mais de uma tarefa está sendo processada “ao mesmo tempo” dentro da aplicação.

Concorrência

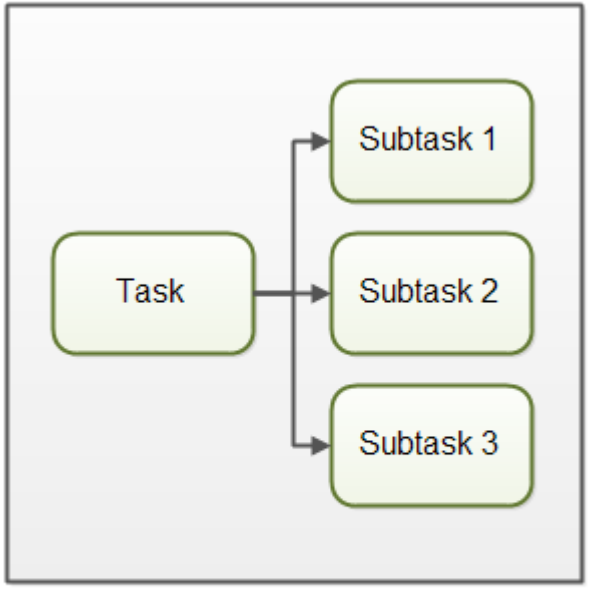
- Ilustração



Concurrency:
Multiple tasks makes progress at the same time.

Paralelismo

- Significa que uma aplicação pode quebrar uma tarefa em subtarefas as quais podem ser processadas em paralelo **ao mesmo tempo**.



Parallelism:
Each task is broken into subtasks which can be processed in parallel.

Em detalhes...

- **Concorrência** está relacionado com como a aplicação trata múltiplas tarefas. Ela pode processar uma tarefa por vez (sequencialmente) ou múltiplas tarefas ao mesmo tempo (concorrentemente).
- **Paralelismo**, do outro lado, está relacionado em como uma aplicação trata uma tarefa individual. Um aplicação pode executar a tarefa do começo ao fim de form serial ou quebrar a tarefa em subtarefas que executem em paralelo.

Em detalhes...

- Uma aplicação pode ser concorrente, mas não obrigatoriamente paralela. Ou seja, ela pode processar mais de uma tarefa “ao mesmo tempo” mas as tarefas não são quebradas em subtarefas.
- Uma aplicação também pode ser paralela e não ser concorrente. Ou seja, cada subtarefa executa de forma paralela sem concorrer por recursos.

Em detalhes...

- Uma aplicação pode ser nem concorrente e nem paralela, ou seja, há apenas uma tarefa sendo executada por vez e ela nunca é quebrada em subtarefas.
- Finalmente, uma aplicação pode ser concorrente e paralela. Ou seja, ela tem múltiplas tarefas sendo executadas “ao mesmo tempo” e também as quebra em subtarefas que são executadas em paralelo.

Referências

- http://tutorials.jenkov.com/java-concurrency/same-threading.html
- http://tutorials.jenkov.com/java-concurrency/concurrency-vs-parallelism.html