



Fundação Arco-Íris de Araputanga
FCARP
FACULDADE CATÓLICA
RAINHA DA PAZ

Threads e Escalonamento

Fundamentos de Sistemas Operacionais



Prof. Thiago Dias Negrão | FCARP

Compreender

Diferenciar processo e thread e reconhecer os recursos compartilhados.

Analisar

Entender concorrência, paralelismo, estados e problemas de sincronização.

Aplicar

Comparar algoritmos de escalonamento e calcular tempos de espera simples.

Voçê sabia?

Como o sistema operacional organiza várias tarefas quando há poucos núcleos de CPU?

Processo e thread representam níveis diferentes

03

Processo

Programa em execução.

Possui espaço de endereçamento, arquivos abertos, dados e recursos próprios.

Pode conter uma ou várias threads.

Thread

Fluxo de execução dentro de um processo.

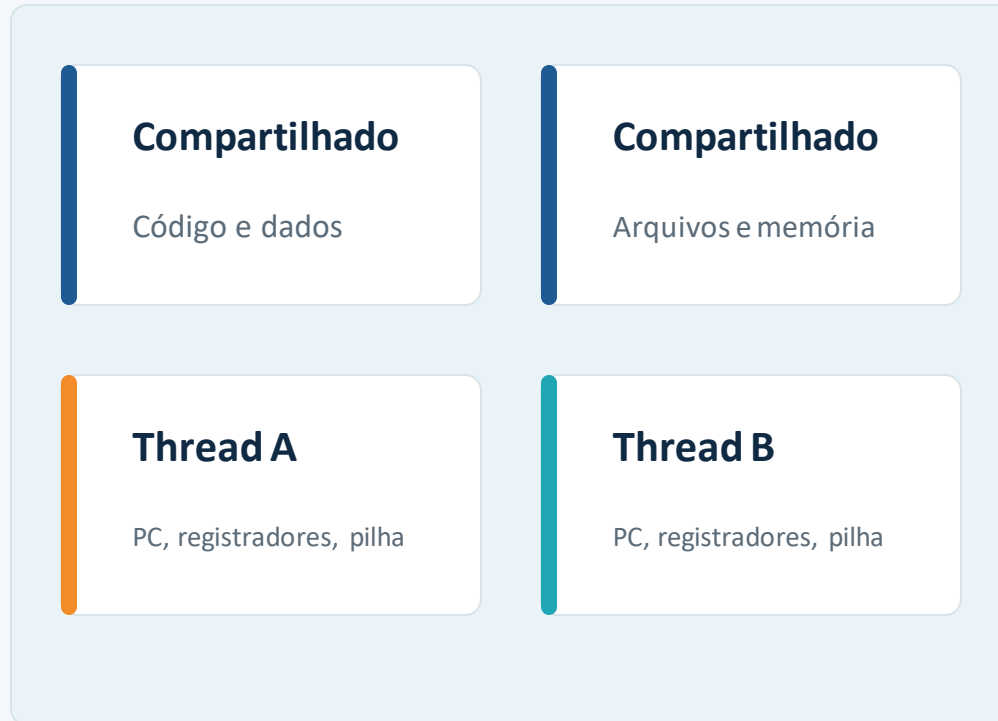
Possui contador de programa, registradores e pilha.

Compartilha código, dados e arquivos com outras threads do processo.

Um processo é o ambiente. A thread é quem executa dentro dele.

O que pertence à thread e o que é compartilhado

Recursos do processo



Resultado

Threads comunicam-se rapidamente porque acessam a mesma memória. Essa vantagem também cria risco de conflitos.

PC = contador de programa

Threads melhoram resposta e aproveitamento da CPU

05

Resposta

Uma tarefa pode continuar enquanto outra aguarda entrada ou saída.

Compartilhamento

Dados comuns ficam disponíveis sem criar mecanismos pesados entre processos.

Paralelismo

Núcleos diferentes podem executar threads ao mesmo tempo.

Economia

Criar e alternar threads costuma exigir menos recursos que processos.

Desafio

O acesso simultâneo a dados exige sincronização correta.

Nível de usuário

Uma biblioteca gerencia as threads.

Criação rápida.

O kernel pode enxergar apenas o processo, dependendo do modelo.

Nível de kernel

O sistema operacional conhece e escalona cada thread.

Permite melhor uso de vários núcleos.

Operações têm maior custo.

Modelos de relação

Muitos para um

Um para um

Muitos para muitos

Sistemas modernos usam principalmente suporte direto do kernel às threads.

Uma thread muda de estado durante a execução

07



Preempção

O sistema retira a CPU e devolve a thread à fila de prontas.

Desbloqueio

Após a E/S, a thread volta à fila de prontas.

Concorrência e paralelismo não são sinônimos

08

Concorrência

As tarefas progridem em períodos intercalados.

Em um único núcleo, a CPU alterna rapidamente entre threads.

Existe a impressão de simultaneidade.

1 núcleo: alternância rápida

Paralelismo

As tarefas executam ao mesmo tempo.

Exige dois ou mais núcleos ou unidades de processamento.

Aumenta o trabalho realizado por intervalo.

Vários núcleos: execução simultânea

A condição de corrida altera o resultado

09

Variável saldo = 100

Thread A: depósito de 50

1. Lê saldo = 100
2. Calcula 150
3. Grava 150

Thread B: retirada de 30

1. Lê saldo = 100
2. Calcula 70
3. Grava 70

Resultado correto: 120 | Resultado possível: 70 ou 150

Sincronização protege a seção crítica

10

Mutex

Permite que apenas uma thread por vez acesse o recurso protegido.

Uso comum: exclusão mútua.

Semáforo

Controla o acesso por meio de um contador.

Pode permitir uma ou várias entradas.

Monitor

Reúne dados compartilhados e operações sincronizadas em uma abstração de alto nível.

Cuidado: bloqueios mal planejados podem causar deadlock e starvation.

O escalonador escolhe quem usa a CPU

11

Fila de prontas

Processos e threads aguardam a oportunidade de executar.

ESCALONADOR

aplica uma política

seleciona a próxima tarefa

CPU

Executa a tarefa escolhida durante o tempo definido pela política.

Despachante: realiza a troca de contexto e entrega a CPU.

Três níveis de decisão organizam a carga

12

Longo prazo

Decide quais processos entram no sistema.

Controla o grau de multiprogramação.

Executa com menor frequência.

Médio prazo

Suspende e retoma processos.

Ajuda a controlar memória e carga.

Relaciona-se ao swapping.

Curto prazo

Escolhe a próxima tarefa da fila de prontas.

Executa muitas vezes por segundo.

Precisa ser rápido.

Cada política prioriza critérios diferentes

13

Utilização da CPU

Manter o processador ocupado.

Throughput

Concluir mais tarefas por intervalo.

Tempo de retorno

Reduzir o tempo entre chegada e conclusão.

Tempo de espera

Reduzir a permanência na fila de prontas.

Tempo de resposta

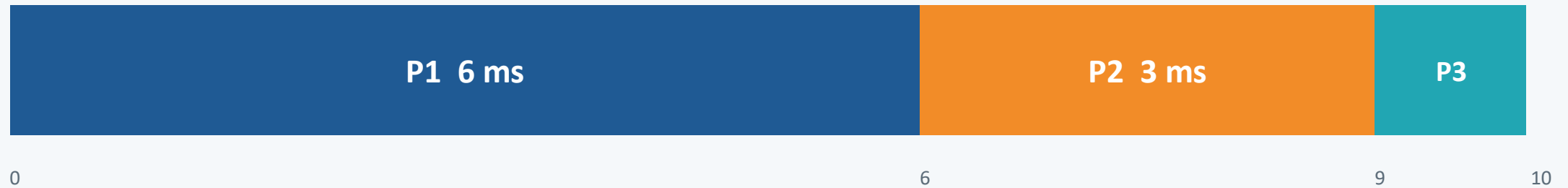
Responder rapidamente à primeira solicitação.

Não existe um algoritmo perfeito para todos os tipos de sistema.

FCFS executa na ordem de chegada

14

Exemplo: P1 = 6 ms P2 = 3 ms P3 = 1 ms



Vantagem

Simple e previsível.

Limitação

Um processo longo pode atrasar todos os demais.

SJF favorece a menor duração estimada

15

SJF

Shortest Job First.

Escolhe o menor próximo ciclo de CPU.

Versão não preemptiva.

Minimiza o tempo médio de espera quando as durações são conhecidas.

SRTF

Shortest Remaining Time First.

Versão preemptiva do SJF.

Uma tarefa menor pode interromper a atual.

Pode causar starvation em tarefas longas.

Desafio prático: o sistema precisa estimar a duração do próximo ciclo de CPU.

Round Robin divide a CPU em fatias de tempo

Quantum = 2 ms



Quantum pequeno

Melhora a resposta, mas aumenta o número de trocas de contexto.

Quantum grande

Reduz trocas, mas aproxima o comportamento do FCFS.

Prioridades

A CPU atende primeiro a tarefa mais prioritária.

Pode haver preempção.

Aging aumenta gradualmente a prioridade de quem espera.

Filas multinível

Separa tarefas por categoria.

Exemplo: sistema, interativas e lote.

Cada fila pode usar uma política própria.

Filas com realimentação

Permite mudar uma tarefa de fila.

Favorece tarefas curtas e interativas.

Penaliza uso prolongado da CPU.

Aging reduz o risco de starvation.

Comparação dos principais algoritmos

18

Algoritmo	Preemptivo?	Ponto forte	Limitação	Uso típico
FCFS	Não	Simplicidade	Efeito comboio	Lote
SJF	Não	Menor espera média	Exige estimativa	Lote
SRTF	Sim	Favorece tarefas curtas	Starvation	Misto
Round Robin	Sim	Boa resposta	Custo de trocas	Interativo
Prioridade	Pode ser	Atende urgência	Starvation	Diversos

Atividade em sala: escalone quatro processos

19

Todos chegam no tempo 0

P1

8 ms
Prioridade 3

P2

4 ms
Prioridade 1

P3

2 ms
Prioridade 4

P4

1 ms
Prioridade 2

Em grupos, construam os diagramas de Gantt para:

1. FCFS 2. SJF 3. Round Robin com quantum de 2 ms 4. Prioridade: 1 é a maior

Depois calculem o tempo médio de espera e comparem os resultados.

Síntese da aula

Threads

Fluxos de execução que compartilham recursos do processo.

Sincronização

Protege dados compartilhados contra acessos conflitantes.

Escalonamento

Define a ordem e o tempo de uso da CPU.

Referências básicas

MACHADO, Francis B.; MAIA, Luiz P. Arquitetura de sistemas operacionais. 5. ed. São Paulo: Grupo GEN, 2013.

SILBERSCHATZ, Abraham; GALVIN, Peter B.; GAGNE, Greg. Fundamentos de sistemas operacionais. São Paulo: Grupo GEN, 2015.

Pergunta final: qual política você escolheria para um computador usado em sala de aula?