

SISTEMAS OPERACIONAIS

AULA 6 – PROCESSOS

Prof.^a Sandra Cossul, Ma.

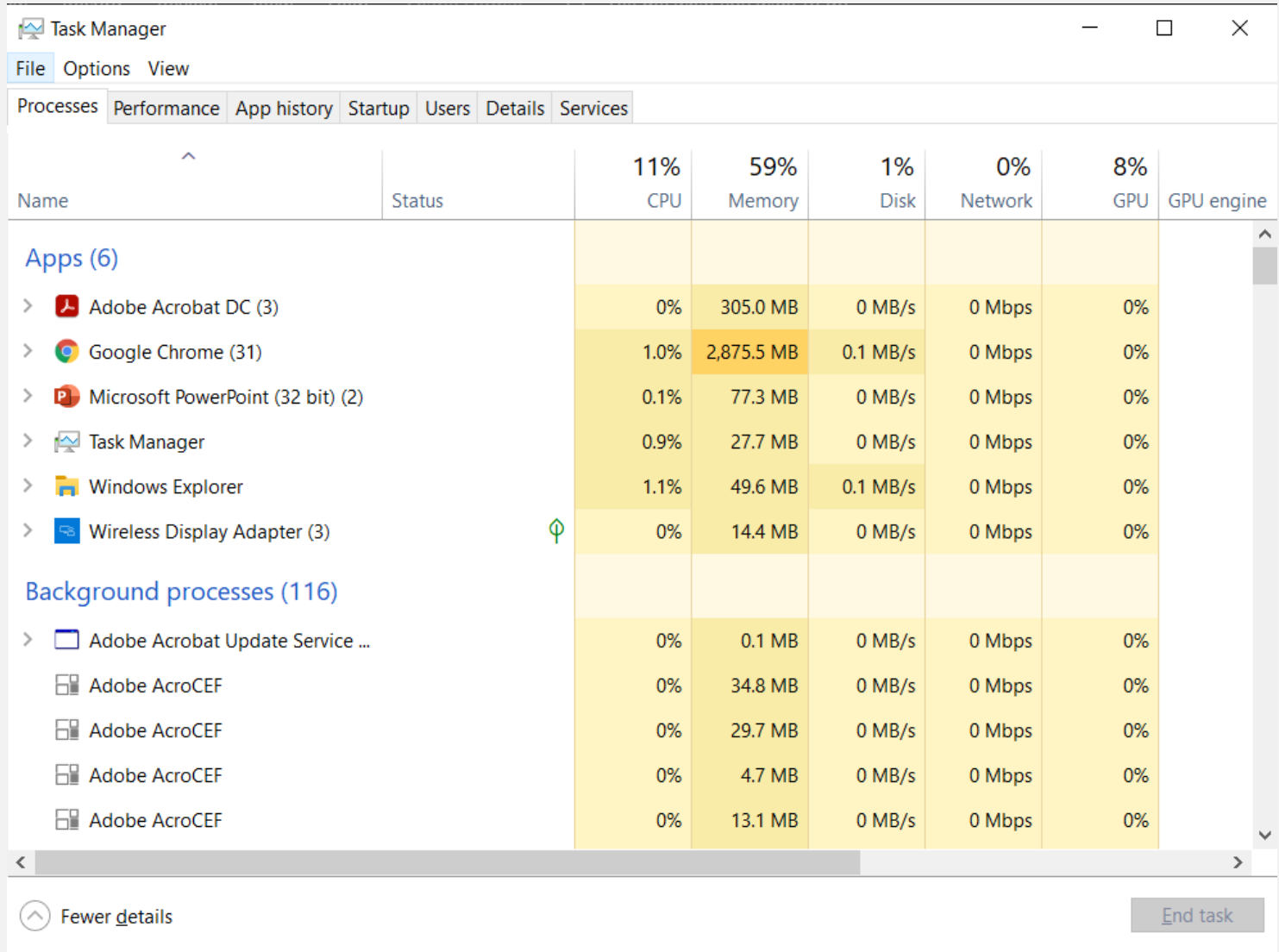


PROCESSOS - INTRODUÇÃO

- Todos os sistemas operacionais multiprogramados são “construídos” em torno do conceito de **processo**.
- Os computadores fazem várias tarefas ao mesmo tempo
- Num sistema **multiprogramado**, o CPU alterna entre processos de forma muito rápida (milissegundos)

PROCESSOS - INTRODUÇÃO

- Exemplo de processos em um computador:



The screenshot shows the Windows Task Manager window with the 'Processes' tab selected. The window title is 'Task Manager'. The menu bar includes 'File', 'Options', and 'View'. The tabs at the top are 'Processes', 'Performance', 'App history', 'Startup', 'Users', 'Details', and 'Services'. The main area displays a list of processes with columns for Name, Status, CPU, Memory, Disk, Network, GPU, and GPU engine. The processes are grouped into 'Apps (6)' and 'Background processes (116)'. The 'Apps (6)' group includes Adobe Acrobat DC (3), Google Chrome (31), Microsoft PowerPoint (32 bit) (2), Task Manager, Windows Explorer, and Wireless Display Adapter (3). The 'Background processes (116)' group includes Adobe Acrobat Update Service ..., Adobe AcroCEF (5 instances), and Adobe Acrobat DC (3). The CPU usage is 11%, Memory is 59%, Disk is 1%, Network is 0%, and GPU is 8%.

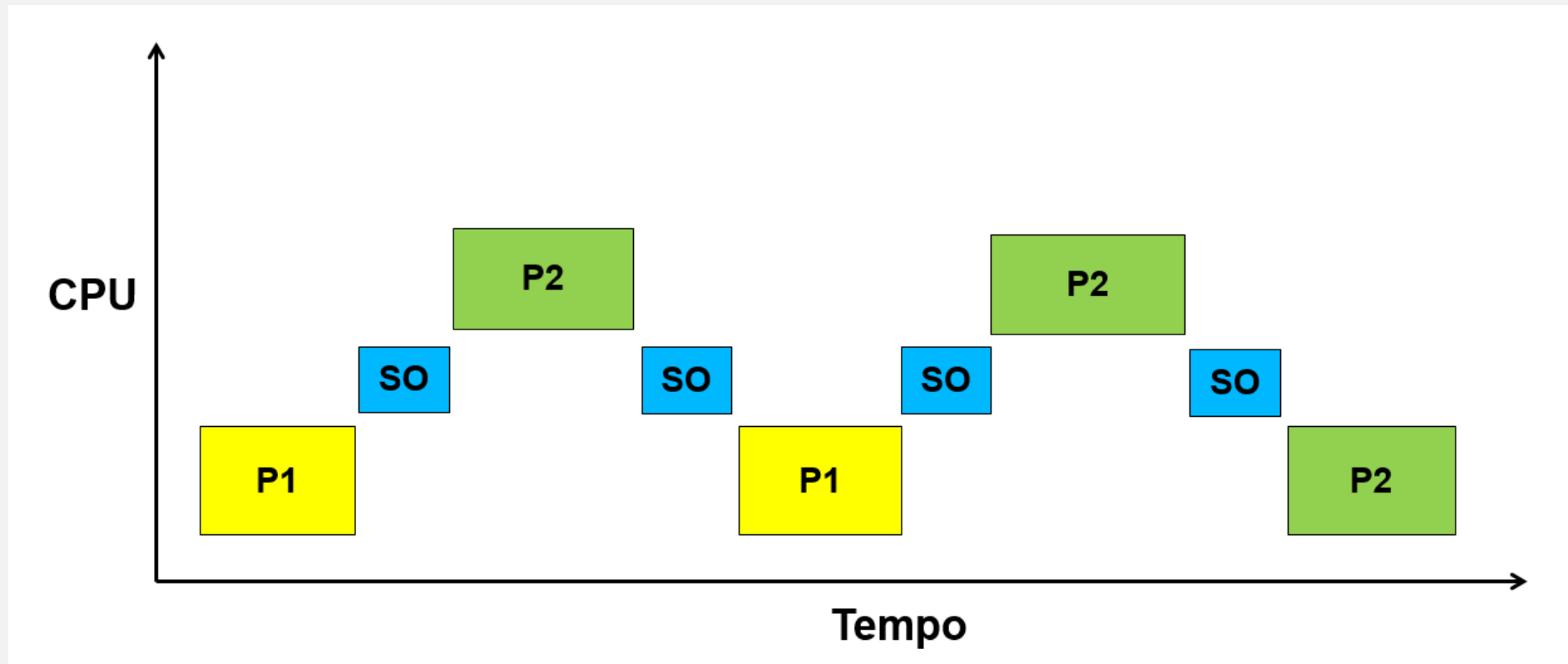
Name	Status	11% CPU	59% Memory	1% Disk	0% Network	8% GPU	GPU engine
Apps (6)							
> Adobe Acrobat DC (3)		0%	305.0 MB	0 MB/s	0 Mbps	0%	
> Google Chrome (31)		1.0%	2,875.5 MB	0.1 MB/s	0 Mbps	0%	
> Microsoft PowerPoint (32 bit) (2)		0.1%	77.3 MB	0 MB/s	0 Mbps	0%	
> Task Manager		0.9%	27.7 MB	0 MB/s	0 Mbps	0%	
> Windows Explorer		1.1%	49.6 MB	0.1 MB/s	0 Mbps	0%	
> Wireless Display Adapter (3)		0%	14.4 MB	0 MB/s	0 Mbps	0%	
Background processes (116)							
> Adobe Acrobat Update Service ...		0%	0.1 MB	0 MB/s	0 Mbps	0%	
Adobe AcroCEF		0%	34.8 MB	0 MB/s	0 Mbps	0%	
Adobe AcroCEF		0%	29.7 MB	0 MB/s	0 Mbps	0%	
Adobe AcroCEF		0%	4.7 MB	0 MB/s	0 Mbps	0%	
Adobe AcroCEF		0%	13.1 MB	0 MB/s	0 Mbps	0%	

At the bottom of the window, there is a 'Fewer details' button and an 'End task' button.

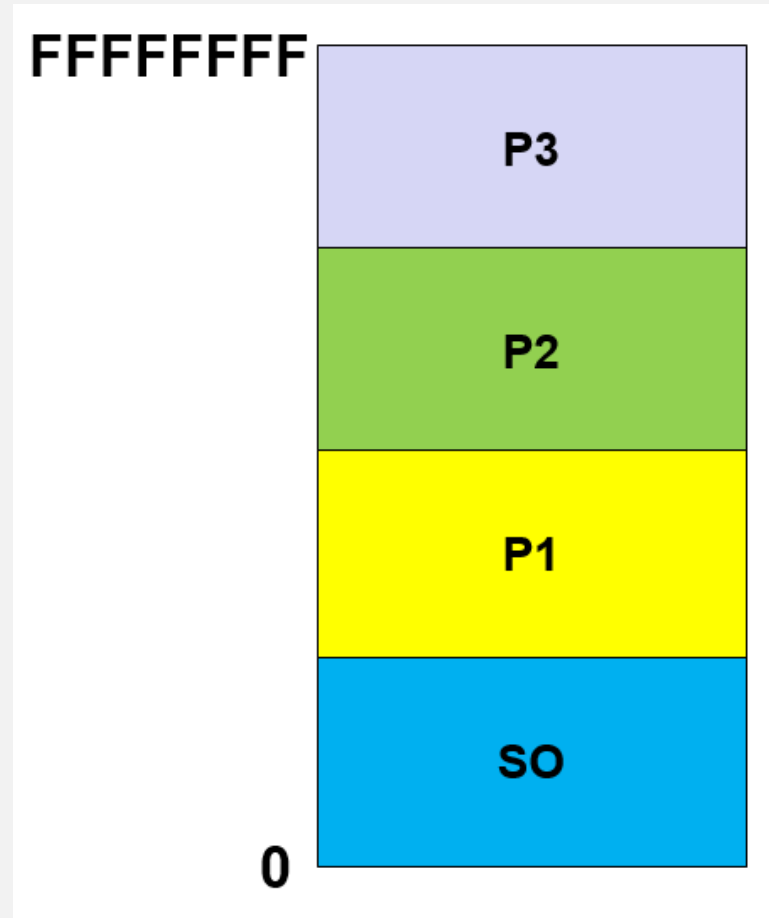
PROCESSOS – USO DA CPU X TEMPO

- Pela velocidade da CPU e pelo pouco tempo que um processo fica esperando a **CPU** (milissegundos), os usuários tem a ilusão de que trabalham em paralelo (**pseudoparalelismo**)
- Em um determinado instante, a CPU executa apenas um programa
- Em um período de tempo, vários processos são executados

PROCESSOS – USO DA CPU X TEMPO



PROCESSOS – USO DA CPU X TEMPO



Alocação de memória

PROCESSOS – IMPLEMENTAÇÃO MULTIPROGRAMAÇÃO

- **Compartilhamento de tempo (*time-sharing*)** – para cada processo que o processador recebe é definido um tempo de processamento, denominado fatia de tempo ou *quantum*.
- Esgotado o *quantum*, o processo em execução perde o CPU e volta para uma fila de processos “prontos”, que estão em memória aguardando sua chance de executar.
- **Preempção** – retirar um recurso “à força” de um processo (CPU)
- **Sistemas preemptivos**

RELEMBRANDO...

- **O computador consiste de uma série de recursos de hardware** (processador, memória, módulos de entrada e saída, timers, etc.)
- **Software são desenvolvidos para uma aplicação** (recebem dados, processam e geram dados de saída)
- **É ineficiente os softwares serem escritos diretamente para uma plataforma de hardware**
 - Aplicações podem compartilhar rotinas para acessar recursos do HW
 - É necessário um software para gerenciar o compartilhamento do processador e outros recursos das várias aplicações rodando ao mesmo tempo
 - Quando múltiplos processos estão acontecendo ao mesmo tempo, é necessário proteger os dados, uso de E/S, etc...

RELEMBRANDO...

- **O SO foi desenvolvido como uma plataforma conveniente, segura e consistente para prover uma interface para uso das aplicações.**
 - Camada de software entre as aplicações e o hardware do computador
- **O SO pode ser visto como uma abstração dos recursos que podem ser requisitados e acessados pelos programas**
 - Recursos incluem memória, interfaces de rede, sistemas de arquivos, etc.
- **O SO faz o gerenciamento destes recursos, permitindo compartilhamento e proteção.**

TÓPICOS A SEREM DISCUTIDOS

- **Como o SO consegue gerenciar todos os programas de forma que:**
 - Os recursos ficam disponíveis para as múltiplas aplicações (programas)?
 - O CPU é “chaveado” dentre as múltiplas aplicações de forma que pareça que todas estão sendo processadas ao mesmo tempo?
 - O CPU e os dispositivos de E/S possam ser usados de forma eficiente ?

PROCESSOS - DEFINIÇÃO

- Processo é um programa em execução.
- É uma **instância** do programa rodando no computador
- A **entidade** que pode ser atribuída e executada em um processador
- Uma **unidade de atividade** caracterizada pela execução de uma sequência de instruções, um estado atual e um conjunto associado de recursos do sistema.
- **Ex.:** associar com receita (programa – algoritmo) e execução da receita (processo)
- **Obs.:** Se um programa roda duas vezes, conta como dois processos.

PROCESSOS – BLOCO DE CONTROLE

- Cada processo é representado e gerenciado pelo SO como um **bloco de controle de processo**.
- Elementos de cada bloco:
 - **Identificador** – para identificar cada processo (único)
 - **Estado** – estado atual do processo
 - **Prioridade** – nível de prioridade relativo aos outros processos
 - **Contador de programa** – endereço da próxima instrução do programa a ser executada

PROCESSOS – BLOCO DE CONTROLE

- Elementos de cada bloco:
 - **Ponteiros de memória** – ponteiros para o código do programa e dados associados com este processo, além de blocos de memória compartilhados com outros processos
 - **Dados do contexto** – dados presentes nos registradores enquanto o processo está em execução
 - **Status E/S** – inclui requisições de E/S, dispositivos de E/S em uso no processo, lista de arquivos em uso no processo, etc.
 - **Informações de status** – quantidade de tempo e clock do CPU, tempo utilizado, limites de conta, etc.

PROCESSOS – BLOCO DE CONTROLE

- **Observações:**

- Um bloco de controle de processo tem informações suficientes para interromper um processo em execução e, mais tarde, continuar a execução normalmente;
- É a chave principal que permite ao SO suportar múltiplos processos (**multiprocessamento**)
- Um **Processo** consiste de:
- **Código do programa + dados associados + bloco de controle de processo**

CRIAÇÃO DE PROCESSOS

- Existem quatro **eventos** que causam a **criação de processos**:
 - **Inicialização do Sistema**
 - Um processo chama uma **system call** de criação de processos
 - Um **usuário** pede para criar um **novo processo**
 - Um **job batch** executa um comando que cria um processo
 - Batch: Arquivo utilizado para automatizar tarefas
 - É um conjunto de comandos rodados sequencialmente

CRIAÇÃO DE PROCESSOS

- Chamadas de sistemas para gestão de processos:

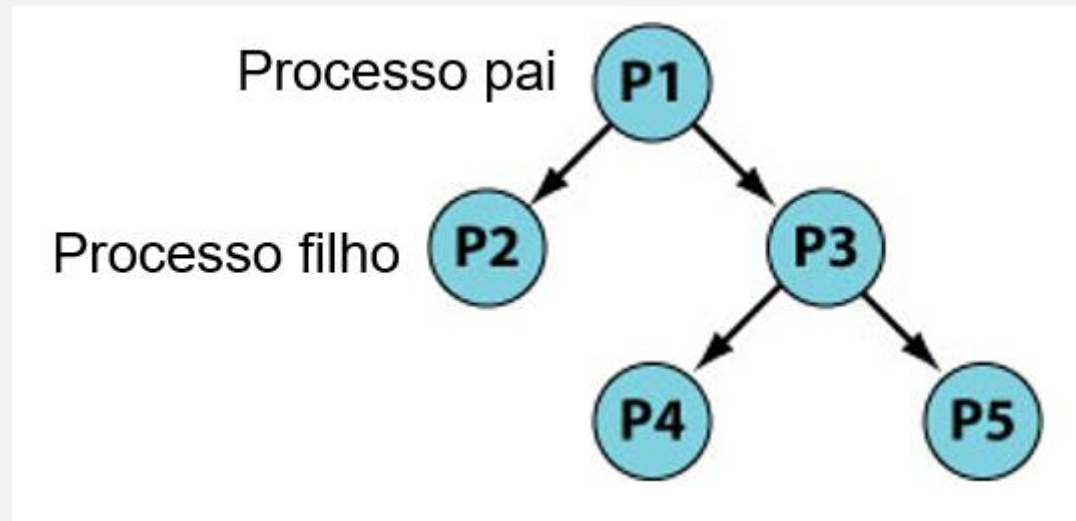
Ação	Windows	Linux
Criar um novo processo	CreateProcess()	fork(), execve()
Encerrar o processo corrente	ExitProcess()	exit()
Encerrar outro processo	TerminateProcess()	kill()
Obter o ID do processo corrente	GetCurrentProcessId()	getpid()

TÉRMINO DE PROCESSOS

- Após a realização de seu trabalho, um processo deve ser **finalizado**
- Existem **quatro condições** para o encerramento de um processo:
 - **Exit normal** (voluntário) - O processo encerrou sua tarefa
 - **Exit por erro** (voluntário) - O processo descobre que não tem condições para continuar executando (bug)
 - **Erro fatal** (involuntário) – processo não existe/não consegue executar
 - **Encerrado por outro processo** (involuntário)

HIERARQUIA DE PROCESSOS

- Em alguns sistemas, quando um processo cria outro processo, o **processo pai** e o **processo filho** continuam associados.
- O processo filho pode criar novos processos, formando uma **hierarquia de processos** (árvore de processos).



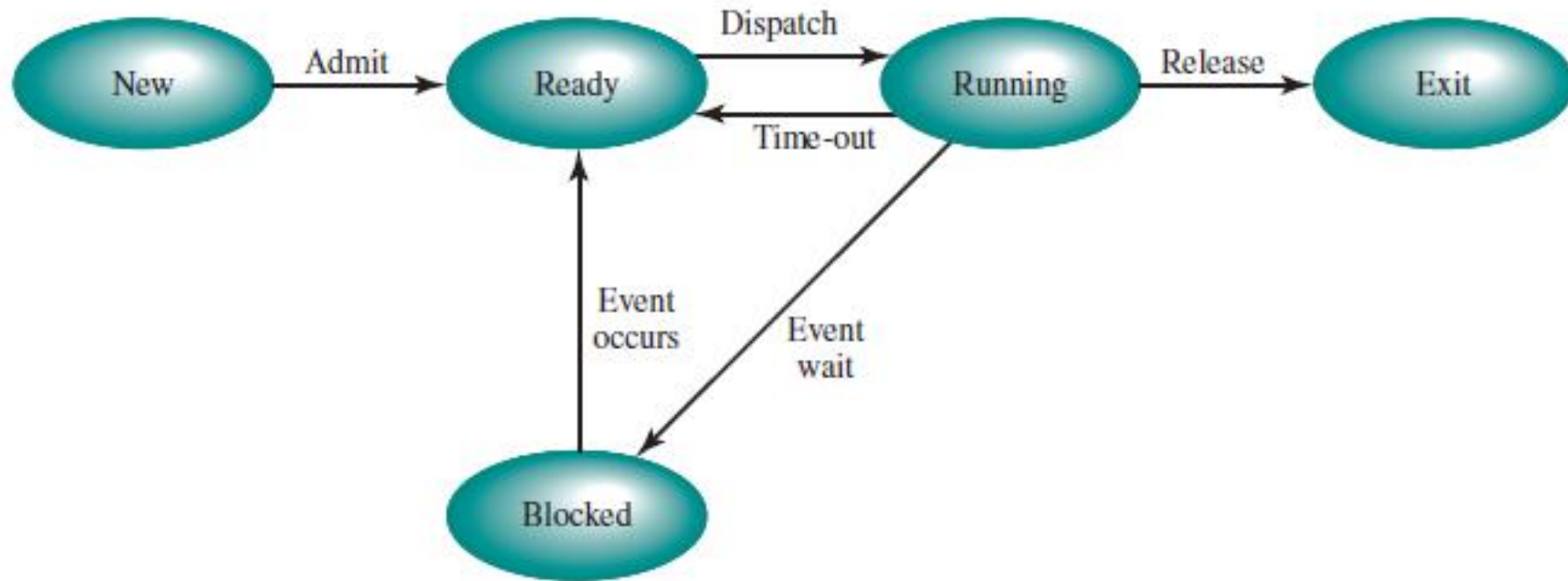
HIERARQUIA DE PROCESSOS

- No **Unix**, um pai e todos os seus filhos formam um **grupo de processos**
- No **Windows**, não há conceito de hierarquia. Portanto, cada processo é criado **independente** de seu pai

ESTADOS DE PROCESSOS

- Existem **5 estados possíveis** para um processo:
 - **Execução**
 - Usando a CPU nesse instante
 - **Pronto**
 - Pode executar, apenas aguardando a disponibilidade da CPU
 - **Bloqueado**
 - Incapaz de executar pela espera de algum evento externo ou sincronização ou aguardando
 - **Novo**
 - processo recém criado pelo SO e ainda não foi carregado. Seu bloco de controle de processo já foi inicializado
 - **Saída**
 - processo que foi liberado da lista dos processos em execução pelo SO ou porque foi parado ou abortado.

ESTADOS DE PROCESOS



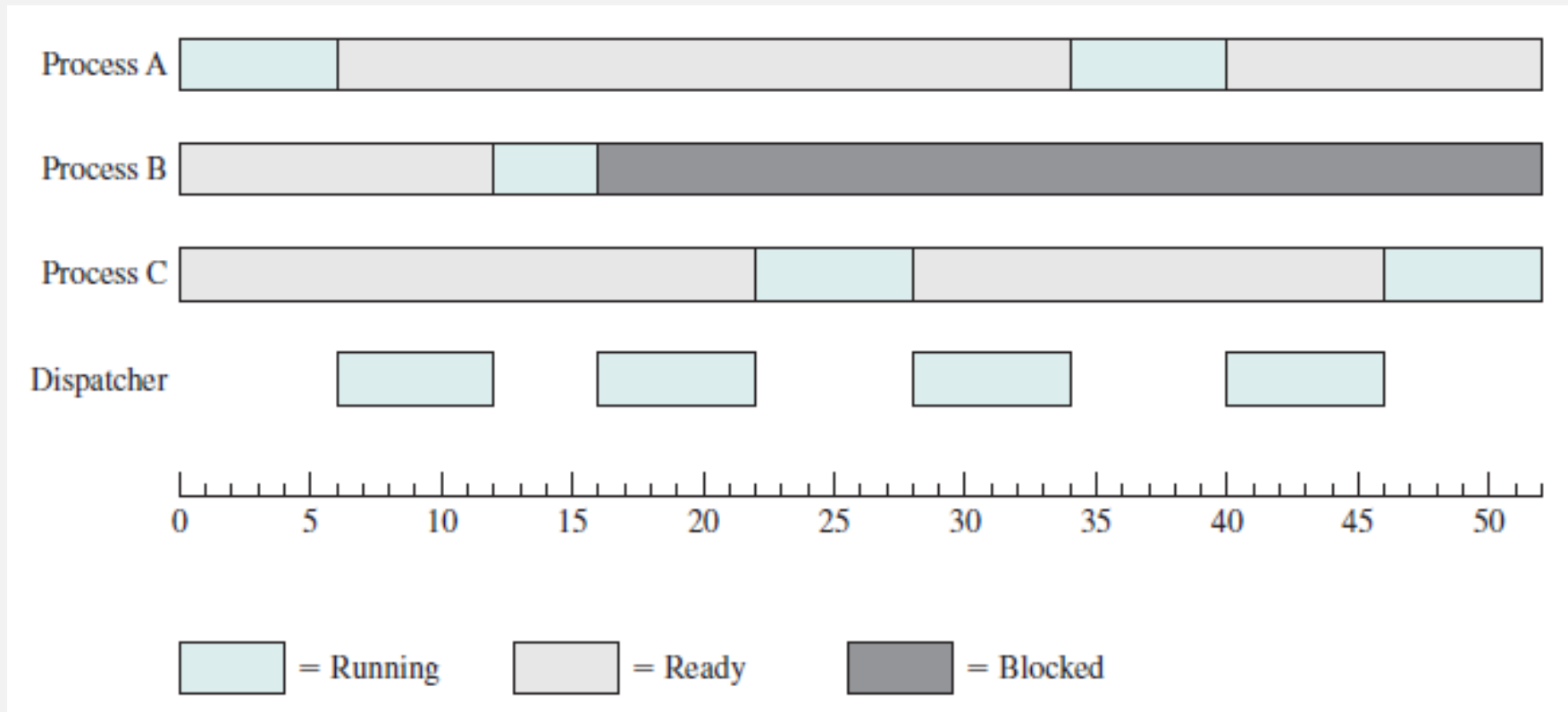
ESTADOS DE PROCESSOS - TRANSIÇÕES

- **Null → Novo:** um novo processo é criado para executar um programa
- **Novo → Pronto:** quando o SO está preparado para executar mais um processo adicional; quando o processo é carregado para a memória (alguns sistemas tem limitação do número de processos em execução)
- **Pronto → Execução:** o escalonador do SO seleciona o processo para executar
- **Execução → Exit:** o processo atual em execução é finalizado pelo SO se o processo indica que finalizou ou se é abortado por algum erro.

ESTADOS DE PROCESSOS - TRANSIÇÕES

- **Execução → Pronto:** o processo atual em execução atingiu o tempo máximo de execução e o escalonador escolhe outro processo (conceito da multiprogramação).
- **Execução → Bloqueado:** o processo está esperando uma entrada de dados que foi solicitada (por meio de uma chamada de serviço)
- **Bloqueado → Pronto:** quando o evento (ou recurso) que estava sendo esperado acontece e pode voltar para ser executado
- **Pronto → Saída:** um processo pai pode finalizar um processo filho (não mostrado no diagrama)
- **Bloqueado → Saída:** um processo pai pode finalizar um processo filho (não mostrado no diagrama)

ESTADOS DE PROCESSOS - TRANSIÇÕES



GERENCIAMENTO DOS PROCESSOS

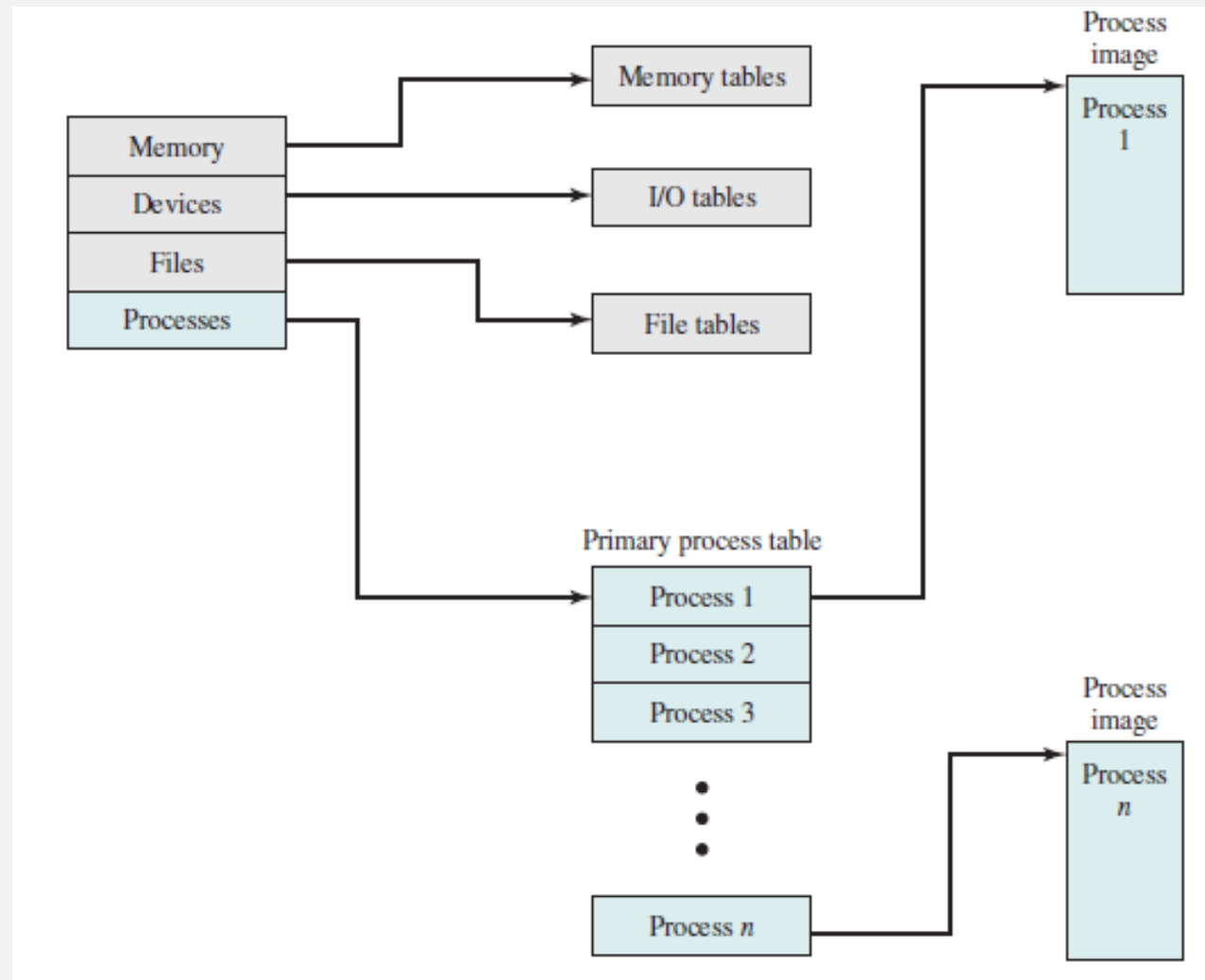
- Num ambiente de multiprogramação, tem um certo número de processos que foram criados.
- Cada processo, ao longo da execução, precisa de acesso a certos recursos do sistema (CPU, unidades de E/S e memória)
- Então, **quais informações o SO precisa para controlar estes processos e gerenciar os recursos para eles ?**

Informação sobre o **estado atual** de cada processo e cada recurso!

GERENCIAMENTO DOS PROCESSOS

- O SO cria e atualiza **tabelas** com informações sobre todas as “entidades” que está gerenciando.
- **Tabelas do SO:**
 - Memória
 - E/S
 - Arquivos
 - Processos

GERENCIAMENTO DOS PROCESSOS



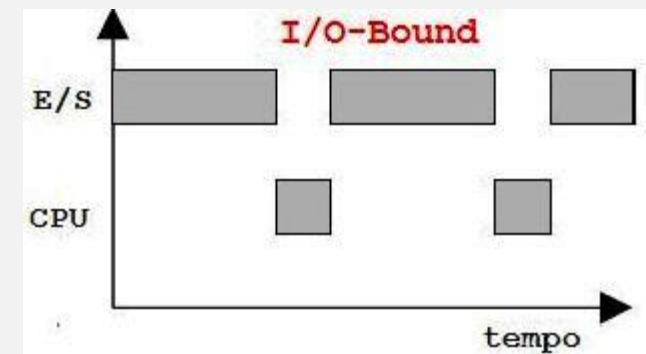
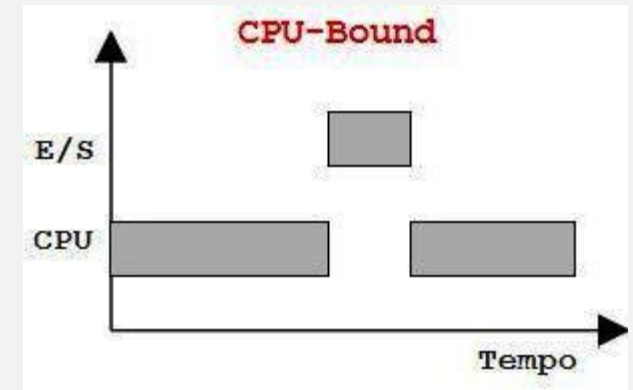
GERENCIAMENTO DOS PROCESSOS

TABELA DE PROCESSOS

Gerenciamento de processos	Gerenciamento de memória	Gerenciamento de arquivos
ID do Processo	Ponteiro para o segmento de código	Diretório-raiz
Grupo do processo	Ponteiro para o segmento de dados	Diretórios de trabalho
Estado do processo	Ponteiro para o segmento de pilha	Descritores de arquivos
Prioridade		Identificador (ID) do usuário
Processo pai (ID)		Identificador (ID) do grupo
Registradores		
Contador do programa		
Palavra de status do programa		
Ponteiro de Pilha		
Tempo em que o processo iniciou		
Tempo de CPU utilizado		
Momento do próximo alarme		
Ponteiros		

CLASSIFICAÇÃO DE PROCESSOS

- **CPU-bound** (ligado à CPU):
 - Maior parte do tempo em estado de execução
 - Ou seja, usando o processador
- **I/O-bound** (ligado à E/S):
 - Maior parte do tempo em estado de bloqueado
 - Ou seja, fazendo E/S

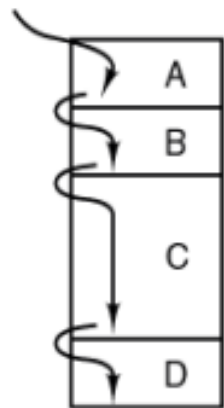


EXERCÍCIO I

- (**F**) Na multiprogramação os processos executam ao mesmo tempo.
- (**F**) Para a implementação de multiprogramação, é necessário que o sistema seja multiprocessado (tenha dois ou mais CPUs)
- (**V**) Existe a necessidade de haver 4 contadores de programa lógicos para a execução de 4 processos na multiprogramação
- (**V**) A execução de uma chamada ao sistema pode causar a criação de um processo
- (**V**) Uma requisição de usuário pode causar a criação de um processo

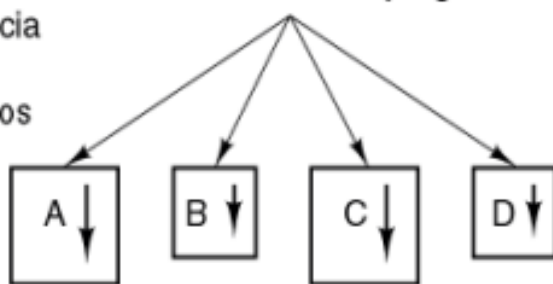
EXERCÍCIO I

Um contador de programa

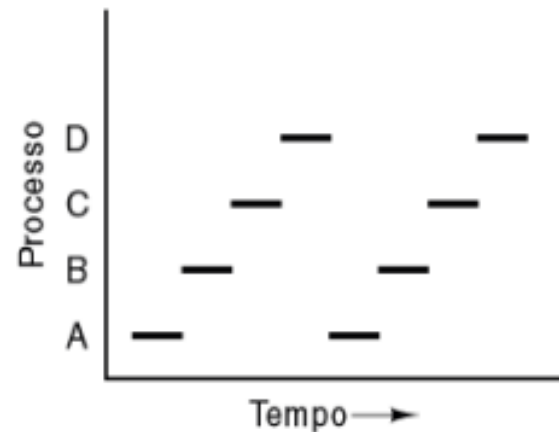


(a)

Quatro contadores de programa



(b)



(c)

- a) **Multiprogramação** de quatro programas
- b) Modelo conceitual de 4 processos sequenciais, independentes, mas
- c) Somente um processo está ativo a cada momento (compartilhamento de tempo/**time sharing**) ⇒ **escalonamento**

Obs.:

(b) Cada processo com seu fluxo de controle (e seu contador de programa lógico).

No entanto, existe apenas um contador de programa físico.

EXERCÍCIO 2

- **Em um sistema operacional, quais das transições de estado listadas nas alternativas a seguir não é uma transição possível ?**

- A. Do estado de pronto para executando
- B. Do estado de pronto para bloqueado**
- C. Do estado de executando para pronto
- D. Do estado de executando para bloqueado

PRÓXIMA AULA

- Threads

BIBLIOGRAFIA

- Tanenbaum, A. S. **Sistemas Operacionais Modernos**. Pearson Prentice Hall. 3rd Ed., 2009.
- Silberschatz, A; Galvin, P. B.; Gagne G.; **Fundamentos de Sistemas Operacionais**. LTC. 9th Ed., 2015.
- Stallings, W.; **Operating Systems: Internals and Design Principles**. Prentice Hall. 5th Ed., 2005.