



DESENVOLVIMENTO DE SOFTWARE MULTIPLATAFORMA

Disciplina: ISO-011 Sistemas Operacionais

Aula: Processos e Threads

Data 20/05/2023

Prof. Me. Anderson Silva Vanin

Diferença entre Programa, Processos e Threads

Programa

- Em geral um programa gera um executável que pode ser chamado diretamente ou através de outros executáveis.
- Programa é uma sequência de instruções codificadas (escritas) a serem executadas pelo computador.
- Thread é uma sequência de instruções sendo executadas.
- Nesse contexto talvez programa esteja sendo usado como sinônimo de executável.

Diferença entre Programa, Processos e Threads

Processo

- Todo executável chamado diretamente rodará em um processo. Executáveis chamados dentro de um processo podem rodar explicitamente em outro processo ou no mesmo processo (quando isso for possível), em geral usando uma DLL.
- O processo é controlado pelo sistema operacional tanto o tempo de processador que ele tem à disposição quanto à memória disponível (que pode ir sendo solicitada pelo processo). Além, é claro, de permissões de acesso a outros recursos que ficam vinculados ao processo como um todo.
- Um processo tem um espaço de endereçamento de memória virtual só para ele. É como se ele rodasse sozinho no computador. Claro que isso não ocorre na realidade, mas ele se comporta como se fosse. Mas isso é outro assunto. O mesmo executável pode rodar em vários processos.

Diferença entre Programa, Processos e Threads

Thread

- A/O thread se assemelha ao processo porque ele tem uma nova linha de execução (e tudo que está relacionado a isto) e o sistema operacional o trata de forma igual ao processo para alocação de tempo de processamento. Mas em termos de memória é responsabilidade da aplicação controlar o acesso compartilhado por todo o processo.
- É comum que as aplicações possuam uma stack para cada thread, mas apenas um (já vi casos de ter mais que um) espaço de heap para todo processo. Por isso costuma-se dizer que é complicado programar com threads, compartilhar estado é difícil. Threads estão vinculadas ao processo, até porque o processo principal em si não deixa de ser uma thread.
- Algumas aplicações possuem um controle próprio de threads internas não controladas pelo processador/sistema operacional. Isso tem algumas vantagens (principalmente evitar troca de contexto do sistema operacional que é algo relativamente caro) e desvantagens (principalmente não podem usar outros processadores). Costumam ser chamadas de soft, light ou green threads.

A execução de um Programa

- Basicamente, a execução de um programa dá-se, em um primeiro instante, em uma ação do sistema operacional. Quando o usuário abre um aplicativo, o sistema operacional interpreta a ação e requisita que os arquivos relacionados a esse software sejam executados.
- **Antes que um programa esteja aberto** e realmente requisite o trabalho em massa da CPU, ele é **apenas carregado na memória RAM**, o que não exige uma atividade do processador.
- Ao efetuar o carregamento de um programa, o sistema operacional trabalha com processos. Cada software possui um processo (alguns utilizam árvores de processos), cada qual com respectivas instruções para o processador saber como proceder na hora de efetuar os cálculos.

- Os chamados “**processos**” são módulos executáveis, os quais contêm linhas de código para que a execução do programa seja realizada apropriadamente. Isso quer dizer que **o processo é uma lista de instruções**, a qual informa ao processador que passos devem ser executados e em quais momentos isso acontece.
- **Uma CPU com dois núcleos, por exemplo, pode trabalhar com dois processos simultaneamente.**

Gerenciador de Tarefas

Arquivo Opções Exibir

Processos Desempenho Histórico de aplicativos Inicializar Usuários Detalhes Serviços

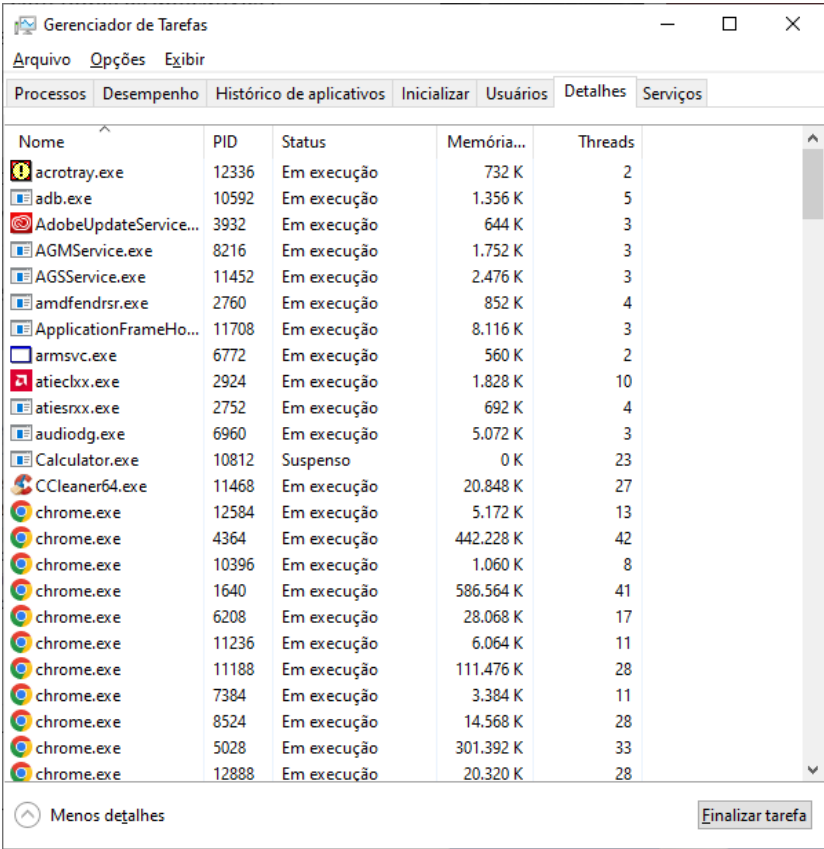
Nome	Status	6% CPU	44% Memória	0% Disco	0% Rede	8% GPU	Mecanismo de GPU	Uso de energia	Tendência de uso de ener...
Aplicativos (6)									
> Gerenciador de Tarefas		0,2%	25,0 MB	0 MB/s	0 Mbps	0%	GPU 0 - 3D	Muito baixo	Muito baixo
> Google Chrome (36)		3,8%	2.655,1 MB	0,1 MB/s	0,1 Mbps	3,5%		Moderada	Muito baixo
> Microsoft PowerPoint (2)		0%	78,8 MB	0 MB/s	0 Mbps	0%		Muito baixo	Muito baixo
> Microsoft Screen Magnifier		0,1%	13,1 MB	0 MB/s	0 Mbps	0%		Muito baixo	Muito baixo
> Teacher Application (32 bits)		0%	7,9 MB	0 MB/s	0 Mbps	0%		Muito baixo	Muito baixo
> Windows Explorer (2)		0,1%	54,4 MB	0 MB/s	0 Mbps	0%		Muito baixo	Muito baixo
Processos em segundo plano (...)									
> Acrobat Update Service (32 bits)		0%	0,5 MB	0 MB/s	0 Mbps	0%		Muito baixo	Muito baixo
AcroTray (32 bits)		0%	0,7 MB	0 MB/s	0 Mbps	0%		Muito baixo	Muito baixo
adb (32 bits)		0%	1,3 MB	0 MB/s	0 Mbps	0%		Muito baixo	Muito baixo
> Adobe Genuine Software Integri...		0%	2,4 MB	0 MB/s	0 Mbps	0%		Muito baixo	Muito baixo
> Adobe Genuine Software Monit...		0%	1,7 MB	0 MB/s	0 Mbps	0%		Muito baixo	Muito baixo
> Adobe Update Service (32 bits)		0%	0,6 MB	0 MB/s	0 Mbps	0%		Muito baixo	Muito baixo
> AMD Crash Defender Service		0%	0,8 MB	0 MB/s	0 Mbps	0%		Muito baixo	Muito baixo

Menos detalhes

Finalizar tarefa

Os processos e as threads

- A princípio, a presença de múltiplos núcleos era suficiente para a maioria dos usuários. Todavia, a evolução dos softwares e dos componentes de hardware requisitou uma divisão ainda melhor das tarefas. As linhas de instruções dos processos adquiriram características únicas, que possibilitaram separá-las para execuções em diferentes núcleos.



Gerenciador de Tarefas

Arquivo Opções Exibir

Processos Desempenho Histórico de aplicativos Inicializar Usuários Detalhes Serviços

Nome	PID	Status	Memória...	Threads
acrotray.exe	12336	Em execução	732 K	2
adb.exe	10592	Em execução	1.356 K	5
AdobeUpdateService...	3932	Em execução	644 K	3
AGMSvc.exe	8216	Em execução	1.752 K	3
AGSSvc.exe	11452	Em execução	2.476 K	3
amdfendrsr.exe	2760	Em execução	852 K	4
ApplicationFrameHo...	11708	Em execução	8.116 K	3
armsvc.exe	6772	Em execução	560 K	2
atieclxx.exe	2924	Em execução	1.828 K	10
atiesnxx.exe	2752	Em execução	692 K	4
audiodg.exe	6960	Em execução	5.072 K	3
Calculator.exe	10812	Suspenso	0 K	23
CCleaner64.exe	11468	Em execução	20.848 K	27
chrome.exe	12584	Em execução	5.172 K	13
chrome.exe	4364	Em execução	442.228 K	42
chrome.exe	10396	Em execução	1.060 K	8
chrome.exe	1640	Em execução	586.564 K	41
chrome.exe	6208	Em execução	28.068 K	17
chrome.exe	11236	Em execução	6.064 K	11
chrome.exe	11188	Em execução	111.476 K	28
chrome.exe	7384	Em execução	3.384 K	11
chrome.exe	8524	Em execução	14.568 K	28
chrome.exe	5028	Em execução	301.392 K	33
chrome.exe	12888	Em execução	20.320 K	28

Menos detalhes Finalizar tarefa

As threads nos processadores

- A thread é uma divisão do processo principal de um programa. Todavia, nem todos os processos são divididos em múltiplas threads, assim como nem todos os processadores são capazes de trabalhar “tranquilamente” com uma enormidade de threads.
- Vamos tomar como exemplo o processador **Intel Core i7 2600**. Verificando no site da fabricante, temos a informação de que esse modelo vem com quatro núcleos e tem suporte para trabalhar com até oito threads.

Especificações da CPU

Número de núcleos ?	4
Nº de threads ?	8
Frequência turbo max ?	3.80 GHz
Frequência da Tecnologia Intel® Turbo Boost 2.0 ⁺ ?	3.80 GHz
Frequência baseada em processador ?	3.40 GHz
Cache ?	8 MB Intel® Smart Cache
Velocidade do barramento ?	5 GT/s
TDP ?	95 W

Processador Intel® Core™ i7-2600
Cache de 8M, até 3,80 GHz

Threads

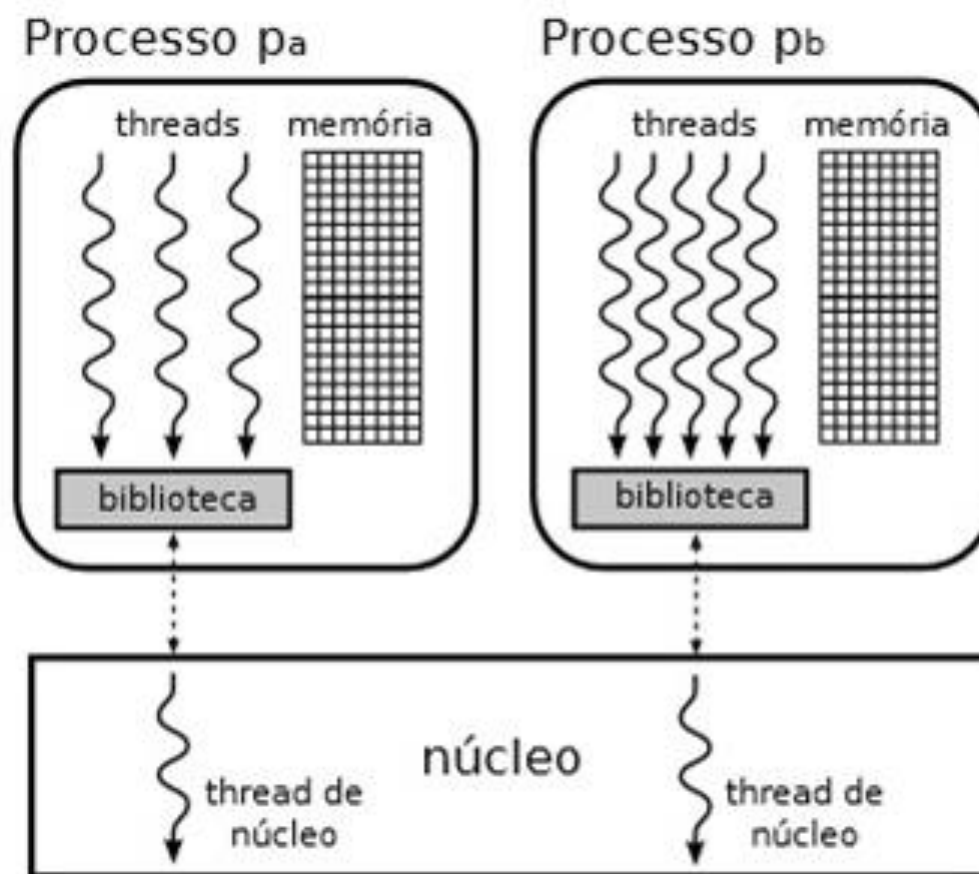
- Threads estão relacionadas na maneira que o computador gerencia as tarefas em demanda através de processos para tratá-las. Mais especificamente, “**cada fluxo de execução do sistema, seja associado a um processo ou no interior do núcleo, é denominado thread**”.
- As threads podem ser divididas em **threads de usuário**, as quais são executadas dentro de um processo e correspondem às tarefas executadas, e **threads de núcleo**, que correspondem à fluxos de execução reconhecidos e gerenciados pelo núcleo do sistema, ou também chamadas de kernel threads.

Modelos de threads

- N:1
- 1:1
- N:M

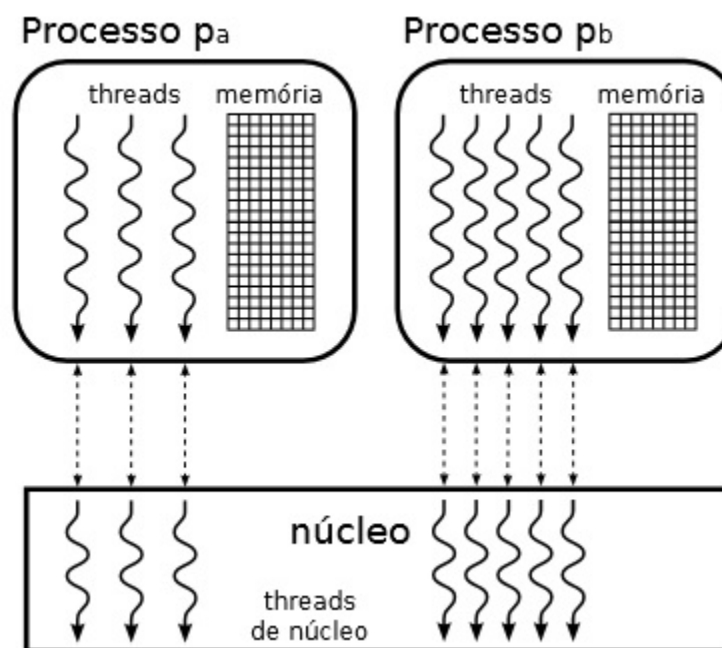
Threads N:1

- Os primeiros sistemas operacionais não tinham suporte a threads, fazendo com que os desenvolvedores elaborassem uma biblioteca, na qual eram definidas as threads para a aplicação desejada.



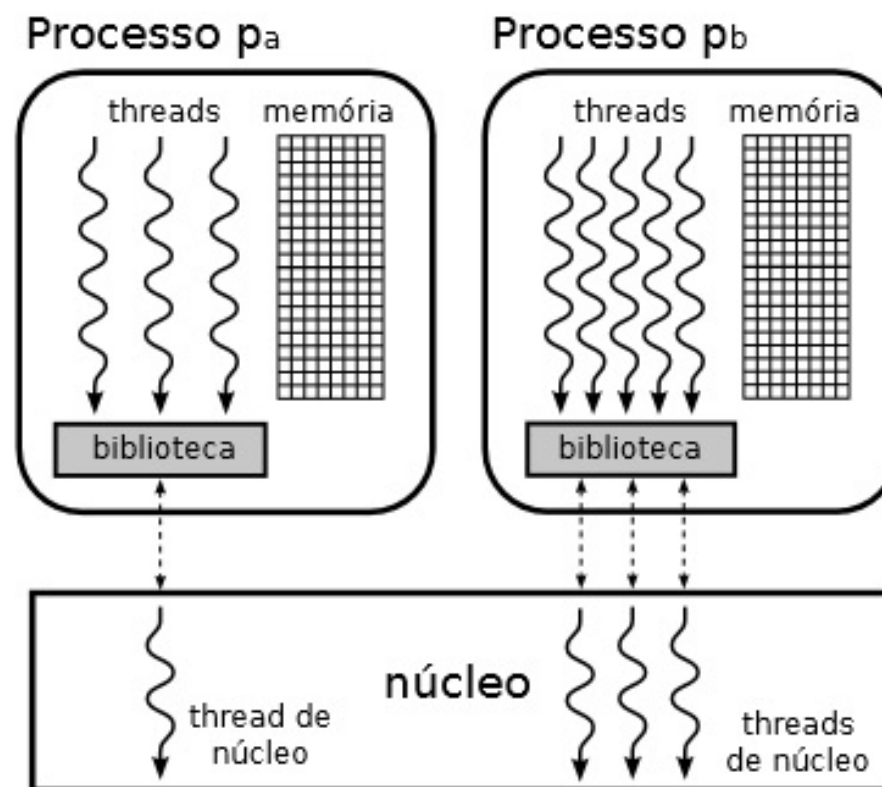
Threads 1:1

- Nesse modelo, cada thread de usuário possui uma thread correspondente no núcleo. Essa inovação fez com que não fosse mais necessária a implementação de bibliotecas para as threads de usuário.



Threads N:M

- O modelo N:M pode ser visto como uma forma híbrida, pois possui características dos dois modelos anteriores. Nele as threads de usuário são gerenciadas por uma biblioteca, passando-os em uma ou mais threads de núcleo. Tornando possível, assim, ajustes dependo da aplicação.

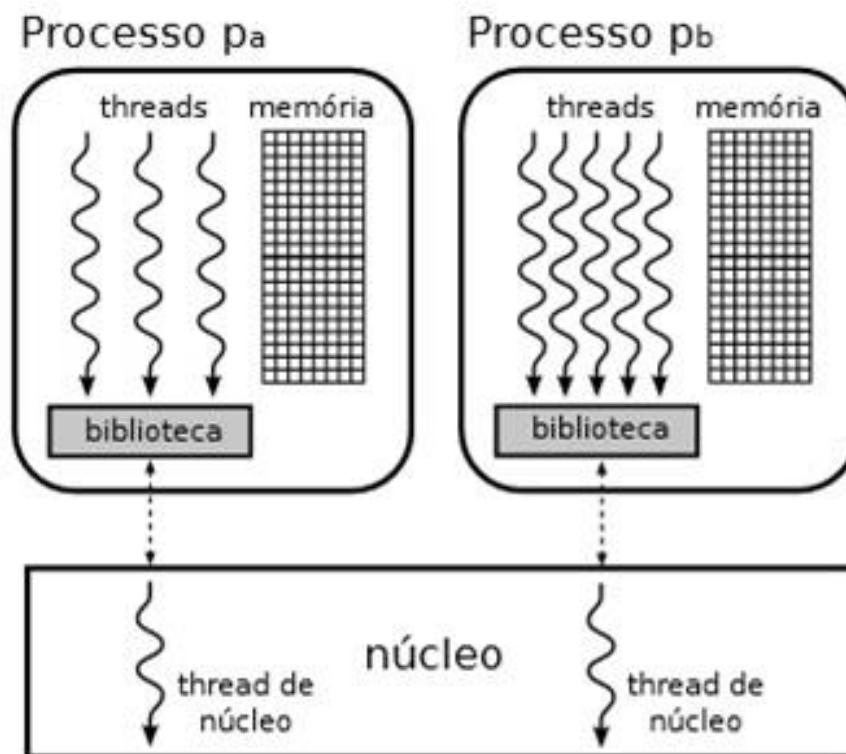


Threads - Comparativo

Modelo	N:1	1:1	N:M
Resumo	Todos os <i>N threads</i> do processo são mapeados em um único <i>thread</i> de núcleo	Cada <i>thread</i> do processo tem um <i>thread</i> correspondente no núcleo	Os <i>N threads</i> do processo são mapeados em um conjunto de <i>M threads</i> de núcleo
Local da implementação	bibliotecas no nível usuário	dentro do núcleo	em ambos
Complexidade	baixa	média	alta
Custo de gerência para o núcleo	nulo	médio	alto
Escalabilidade	alta	baixa	alta
Suporte a vários processadores	não	sim	sim
Velocidade das trocas de contexto entre <i>threads</i>	rápida	lenta	rápida entre <i>threads</i> no mesmo processo, lenta entre <i>threads</i> de processos distintos
Divisão de recursos entre tarefas	injusta	justa	variável, pois o mapeamento <i>thread</i> → processador é dinâmico
Exemplos	GNU Portable Threads	Windows XP, Linux	Solaris, FreeBSD KSE

As threads nos processadores

- Isso quer dizer que essa CPU pode trabalhar com quatro processos indivisíveis simultaneamente (um em cada núcleo) ou com até oito linhas de execução (threads) – as quais podem ou não ser de um mesmo processo.



Prática em Python

Para demonstrar a execução de Processos e Threads, vamos utilizar a linguagem Python e visualizar tempos de execução com e sem o uso de Threads.

Para a execução dos códigos em python vamos utilizar o ambiente Google Colab (<https://colab.research.google.com/>).