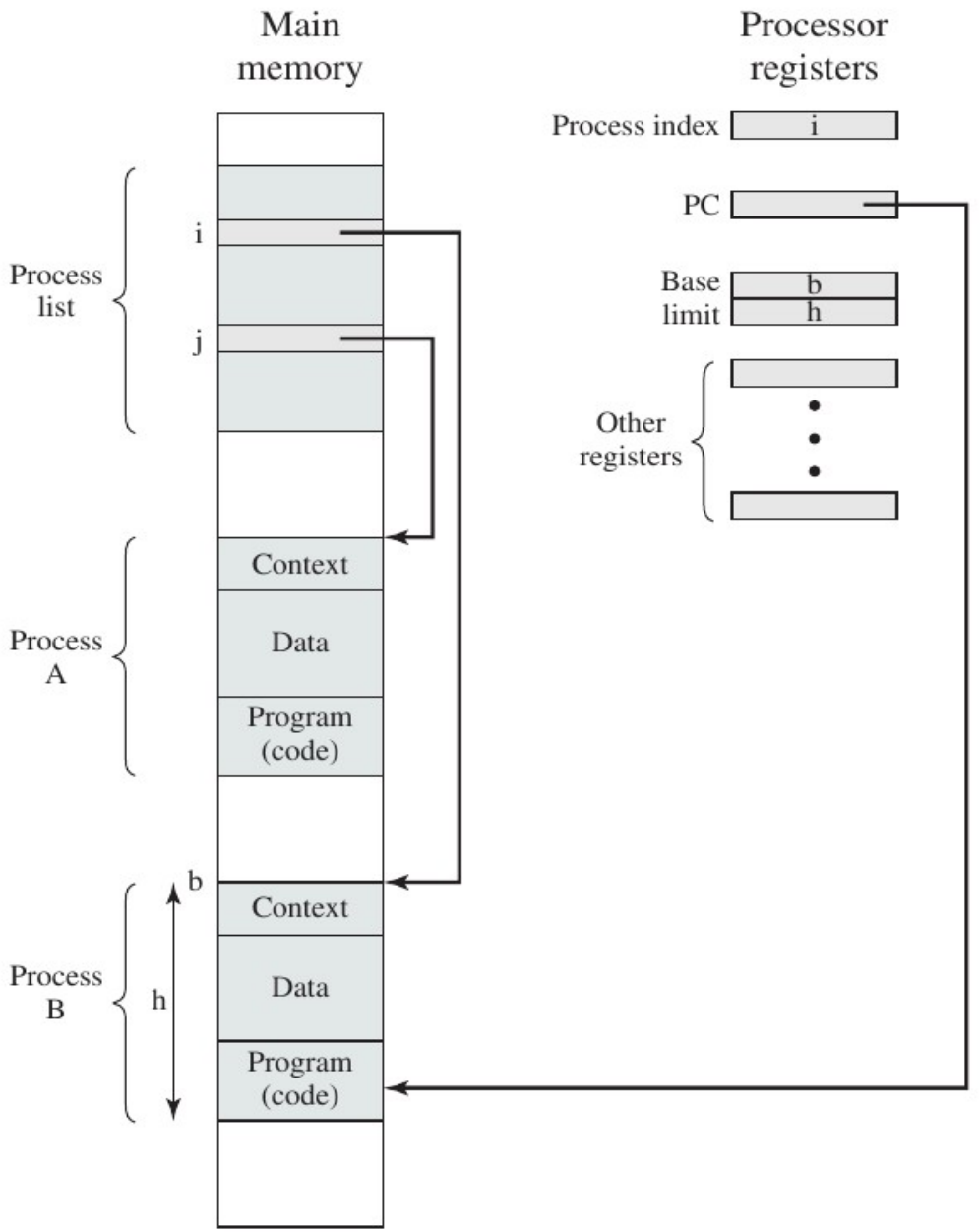
Sistema Operacional

Implementação de Processo e Threads

O mecanismo básico para a criação de processos no UNIX é a chamada de sistema **Fork().** A Figura a seguir ilustra como que o processo e implementado.


Main memory
Processor registers
Process index
i
PC
Base
limit
b
h
Other registers
Process list
i
j
Process A
Context
Data
Program (code)
Process B
b
h
Context
Data
Program (code)

Introdução a Threads

Em sistemas operacionais tradicionais, cada processo tem um espaço de endereçamento e um único *thread* (fluxo) de controle.

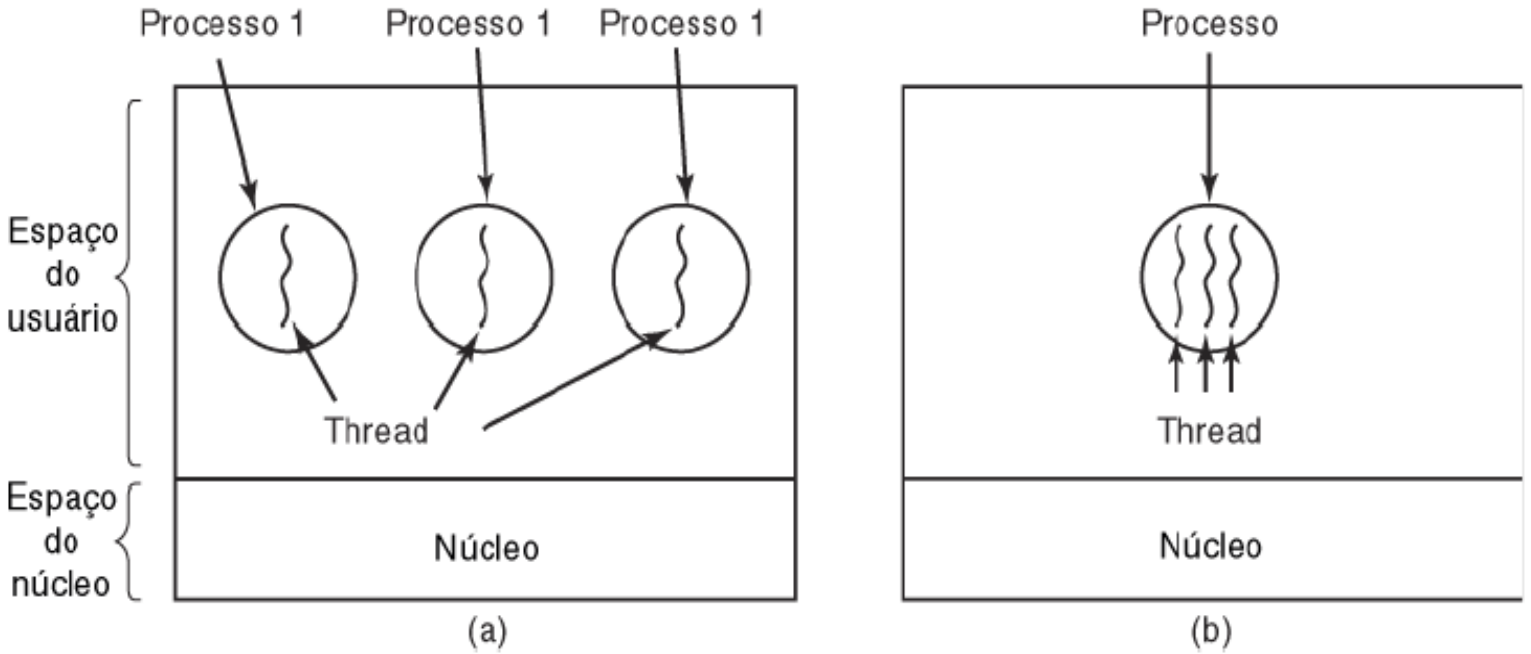
Contudo, frequentemente há situações em que é desejável ter múltiplos *threads* de controle no mesmo espeço de endereçamento executando em quase paralelo, como se fossem processos separados .

O modelo de Thread

As threads são utilizadas para que multiplas execuções ocorram no mesmo ambiente do processo com um grande grau de independência uma da outra.

O modelo de Thread

Ter multiplas *threads* executando em paralelo é análogo a ter vários múltiplos processos executando em paralelo em um único computador.



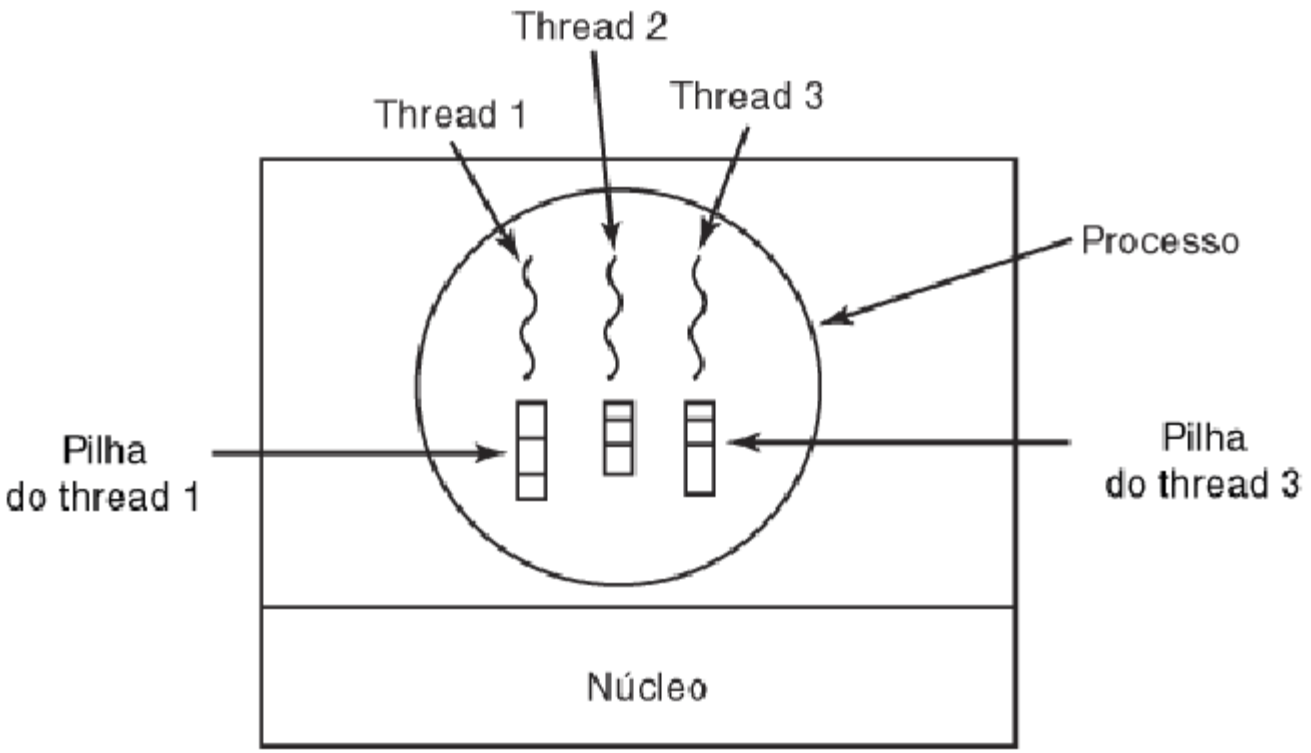
O modelo de Thread

Como cada *thread* pode ter acesso a qualquer endereço de memória dentro do espaço de endereçamento do processo, uma *thread* pode ler, escrever ou até mesmo apagar completamente a pilha de outra *thread*.

A primeira coluna da Figura abaixo mostra alguns itens compartilhados por todas as *threads* em um processo. A segunda coluna mostra alguns itens privativos de cada *thread*.

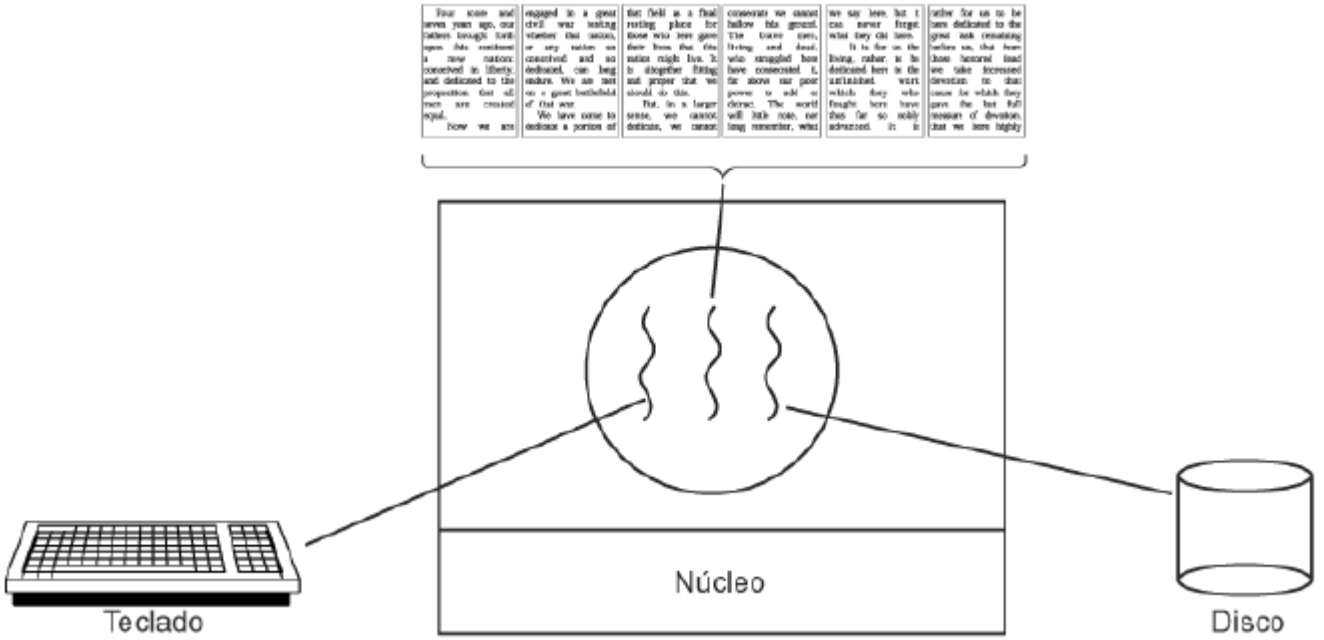
Itens por processo	Itens por thread
Espaço de endereçamento	Contador de programa
Variáveis globais	Registradores
Arquivos abertos	Pilha
Processos filhos	Estado
Alarmes pendentes	
Sinais e manipuladores de sinais	
Informação de contabilidade	

Cada *thread* possui sua pilha própria

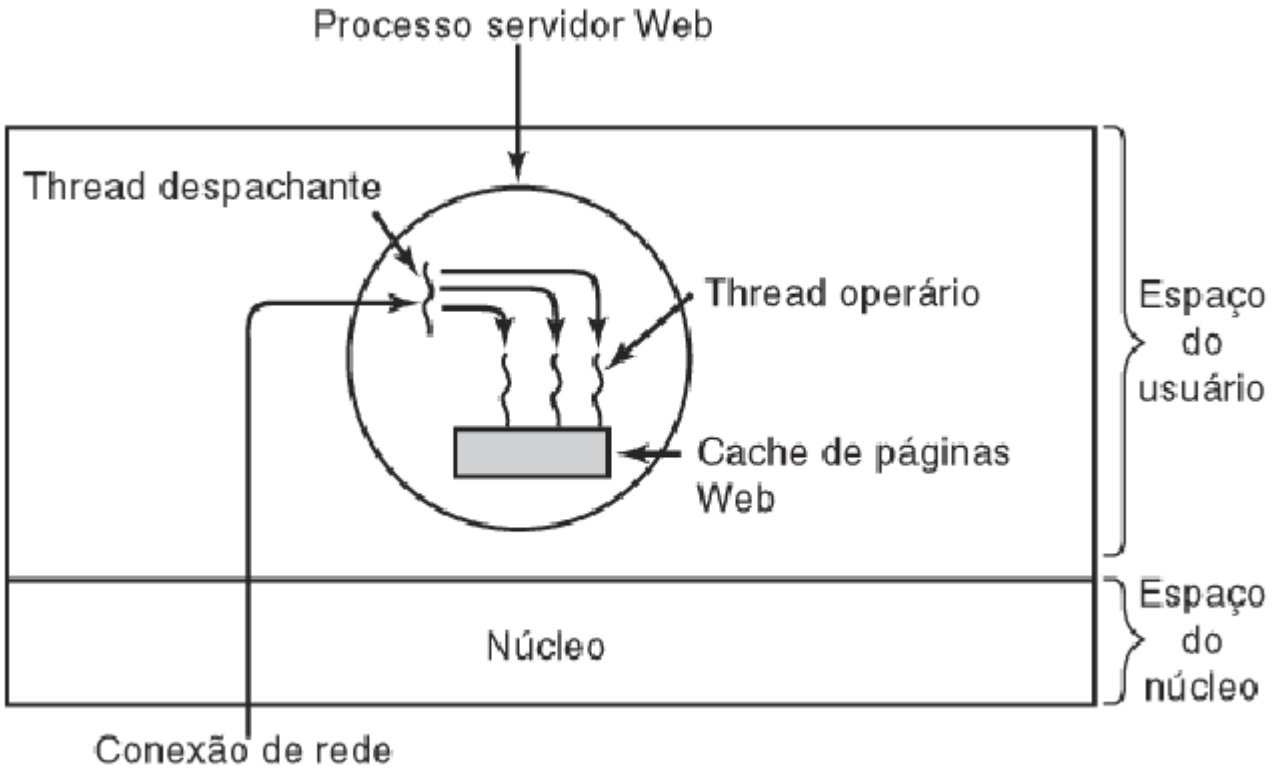


As *threads* são muito utilizadas em aplicações clientes servidor, como também em aplicações que necessitam de trabalho de execução em paralelo.

Um processador de texto com três *threads.*



Um servidor Web com multiplas *threads*.



Implementação de threads de usuário

Há dois modos principais de implementar threads: no espaço do usuário e no espaço do núcleo.

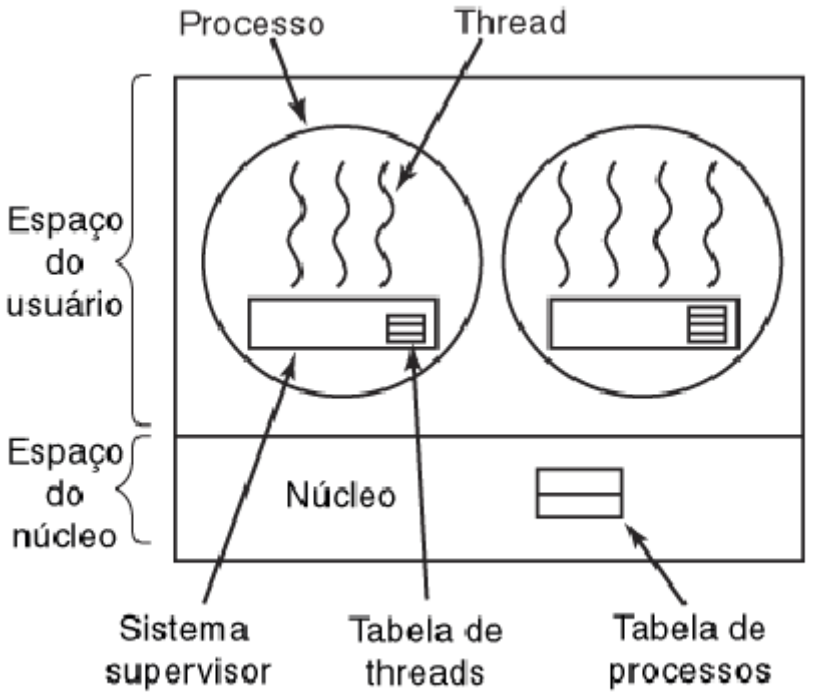
Threads executando dentro do espaço do usuário são chamados de *threads* de usuário.

Neste caso, sem poder contar com o suporte do sistema operacional para a criação de *threads*, os desenvolvedores contornaram este problema desenvolvendo bibliotecas para gerenciar *threads* dentro de cada processo.

Implementação de threads de usuário

Usando estas bibliotecas, uma aplicação pode lançar mão de várias *threads* conforme sua necessíade, mas o núcleo do sistema irá sempre perceber (e gerenciar) apenas um fluxo de execução naquele processo.

Por essa razão, esta forma de implementação de *threads* é nomeada de Modelo ***Threads* N**. A figura abaixo exemplifica este modelo:



Implementação de threads de usuário

O modelo de *threads* N:1 é muito utilizado e defácil implementação. A carga de gerenciemento imposta ao núcleo do sistema é constante e independe do número de *threads*.

Esta característica o torna útil na construção de aplicações que necessitam de várias *threads*, como por ex emplo, jogos, servidor web *multithread* etc. Um exemplo de biblioteca utilizada é para a criação deste tipo de thread é a biblioteca *P_Thread*.

Porém, este modelo apresenta alguns problemas, por exemplo, o núcleo do sistema gerencia o tempo de processamento entre os processos. Como o núcleo não conhece as threads ex istentes em um processo, ele dividirá o tempo igual para cada processo.

Implementação de threads de usuário

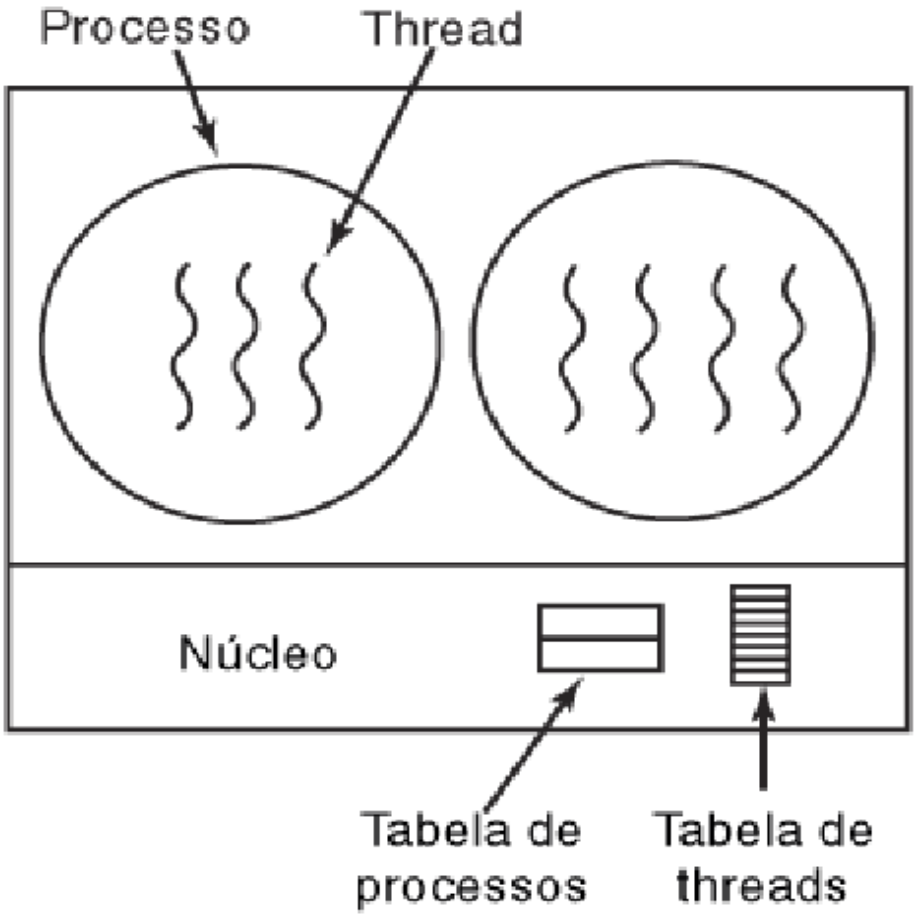
A necessidade de suportar aplicações com várias *threads* (*multithread*) levou os desemvolvedores de sistemas operacionais a incorporar o gerenciamento de *thread* em modo núcleo.

Para cada *thread* de usuário foi então definido uma *thread* correspondente dentro do núcleo.

No caso de alguma *thread* efetuar uma instrução bloqueante, apenas esta *thread* será bloqueada, e no caso do computador possuir vários processadores, várias *threads* poderão ser executadas em paralelo (real paralelismo).

Implementação de threads de usuário

A figura a seguir e emplifica as *threads* de núcleo.



Implementação de threads de usuário

No caso de grandes servidores, a grande quantidade de *threads* irá sobrecarregar o núcleo do sistema. Para resolver este problema, alguns sistemas operacionais implementam omodelo híbrido de *thread*.

Neste novo modelo, uma biblioteca gerencia as *threads* em modo usuário e estas são mapeadas em duas ou mais *threads* de núcleo. Este modelo é chamado de modelo N:M, onde *N threads* se usuário são mapeados em M < = N *threads* de núcleo.

Implementação de threads de usuário