



Operating Systems

Threads and Concurrency

Seyyed Ahmad Javadi

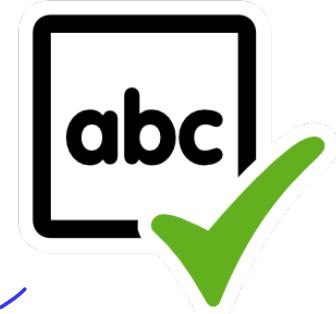
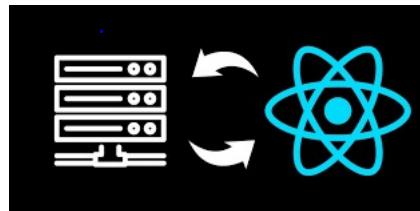
sajavadi@aut.ac.ir

Fall 2021

Motivation

- Most modern applications are multithreaded
- Multiple tasks with the application can be implemented by threads

- Update display
- Fetch data
- Spell checking



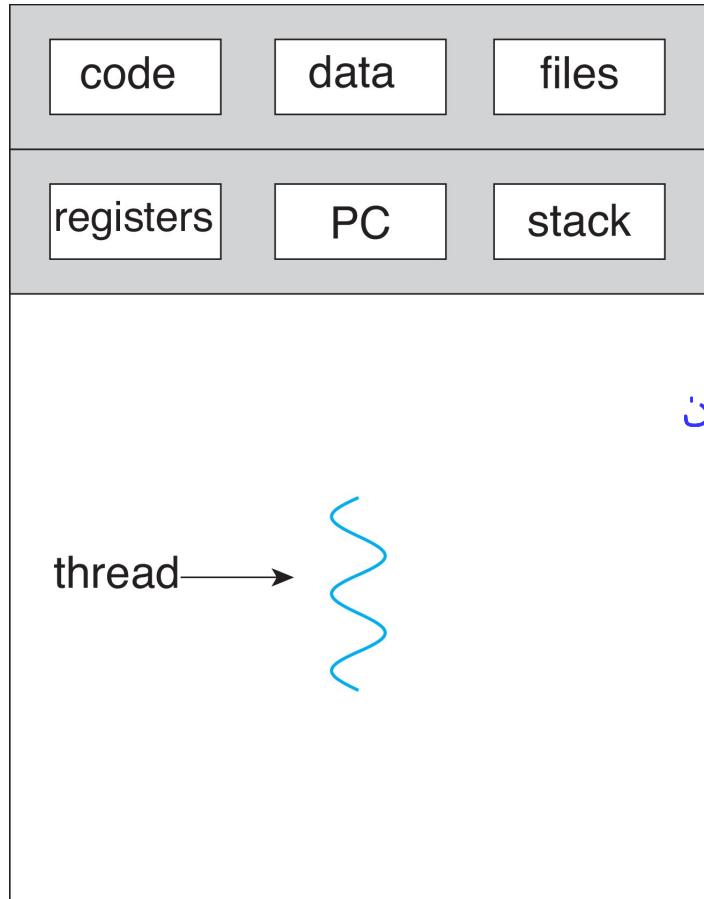
هر کدام از فرآیندهای مختلف را در یک دسکتاپ اجرا می‌کند

- Process creation is **heavy-weight** while **thread** creation is **light-weight**
- Kernels are generally multithreaded

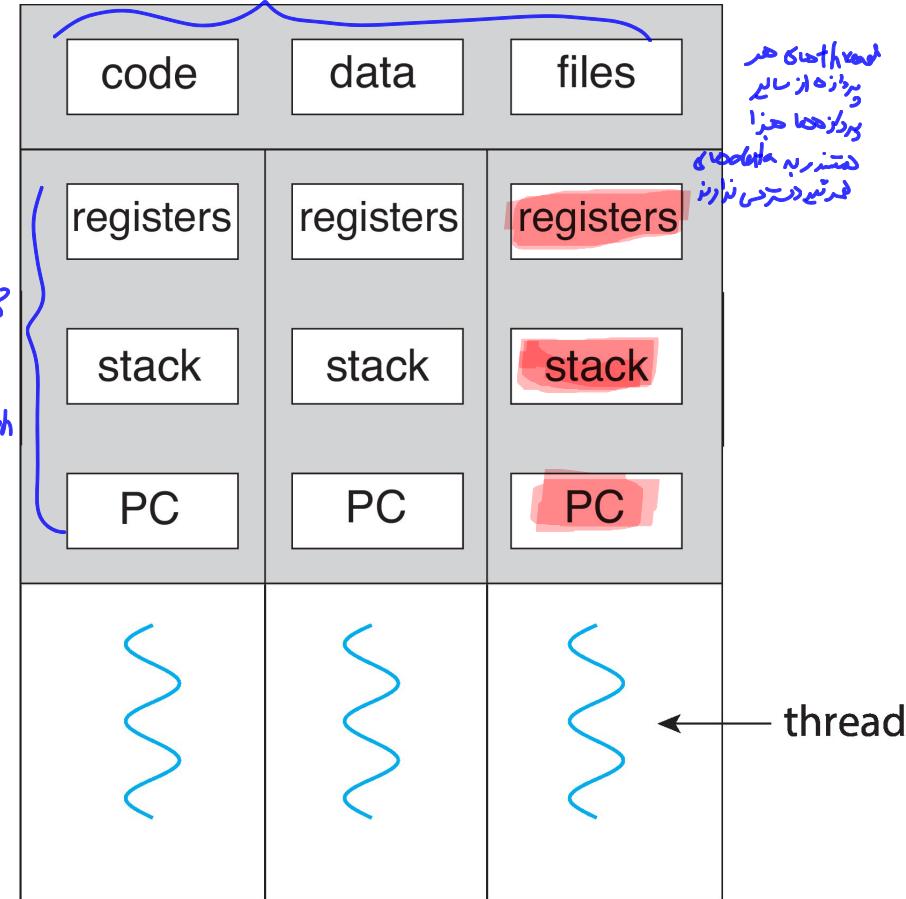
ادنی قیمت
لے بردی همین - weight
و هر چیز را با هم به این شکل می‌گذراند
باید منطقی وسیعی داشته باشد

Single and Multithreaded Processes

با اینکه چند نویز !
با هم یک پروسس



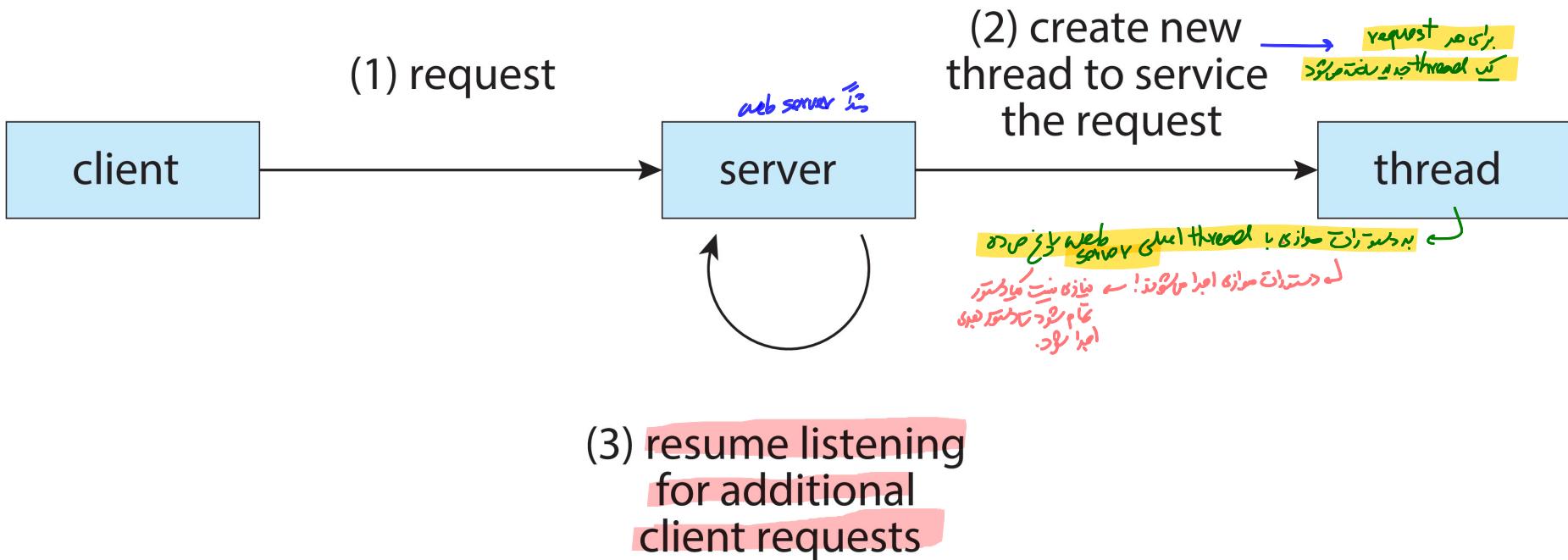
single-threaded process



multithreaded process

با هم هر کدامی
و با هم سایر
برگزشها همراه
لسترنون همراه
قدرتی داشتند

Multithreaded Server Architecture



Benefits

■ Responsiveness

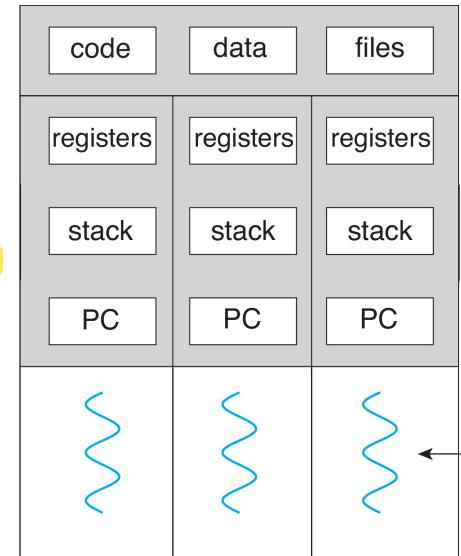
- Allow continued execution if part of process is blocked
- Especially important for user interfaces → 

چنین زمانی از لغت داری که آخرین بار بت در میان
تا زمانی که اولین بار بت در راسته شروع شد!
آنچه می‌گذرد زمان **Response time** است!

■ Resource Sharing

- Threads share resources of process
- Easier than shared memory or message passing.

می‌توانید چند کاری که نباید داشت را
اعلمیدم (race) باشد می‌توانید همان متنرا:
shared memory را درین نظر!



Benefits (cont.)

Economy

- Cheaper than process creation
- Thread switching lower overhead than context switching.

لذیع پردازی از جهت اینکه در هر موردی که چندین کارهای مجزا را باید اجرا کرد، ممکن است مساحتی بزرگ در فضای ذخیره سازی ایجاد شود.

بجزیع پردازی
پردازی پنهان



Scalability → Core i7, i9, ...

- Process can take advantage of multicore architectures.

استفاده از هر منابعی که برای این بزرگی ممکن باشد!
scalable
بسیار! من چنین چیزی را نمی‌دانم!

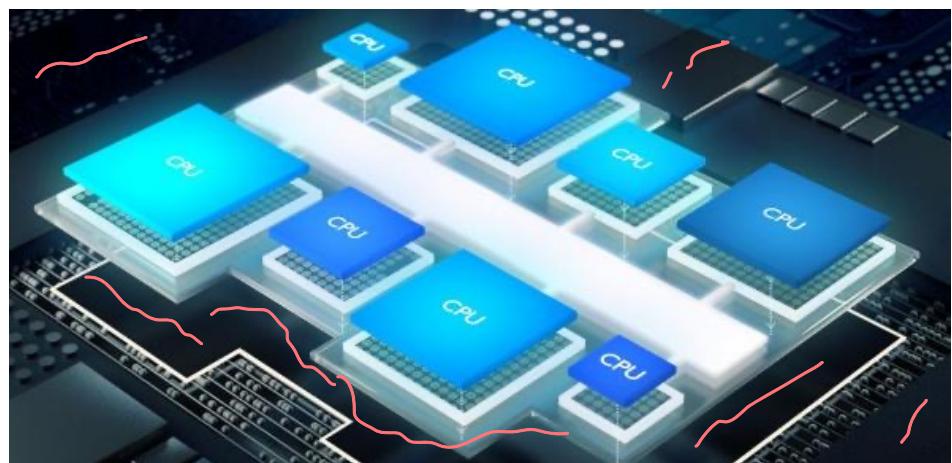


Multicore Programming

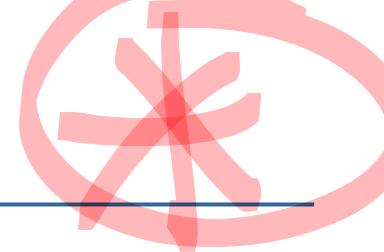
- Multicore or multiprocessor systems putting pressure on programmers, challenges include:

- Dividing activities → *thread مجزاً منفصل thread بـ*
- Balance: ensuring that the tasks perform **equal work of equal value.**
حجم کار رهم ارزشی کار نیسان
- Data splitting → *آنچه داده را تقسیم کنیم کمترین تفاوت را بین رایج برای هر Thread داشته باشیم! مثلاً از شعبه ای خاص استفاده نکنید!*
- Data dependency → *اگر قسمت هایی مختلف به هم وابسته باشند، دین 1 بنتیجه 5 و 2 بنتیجه 6 نیز نباشد*
- Testing and debugging

↳ **thread-Safe**
پس از ایجاد خطا این اشتباه را که برخانده شد

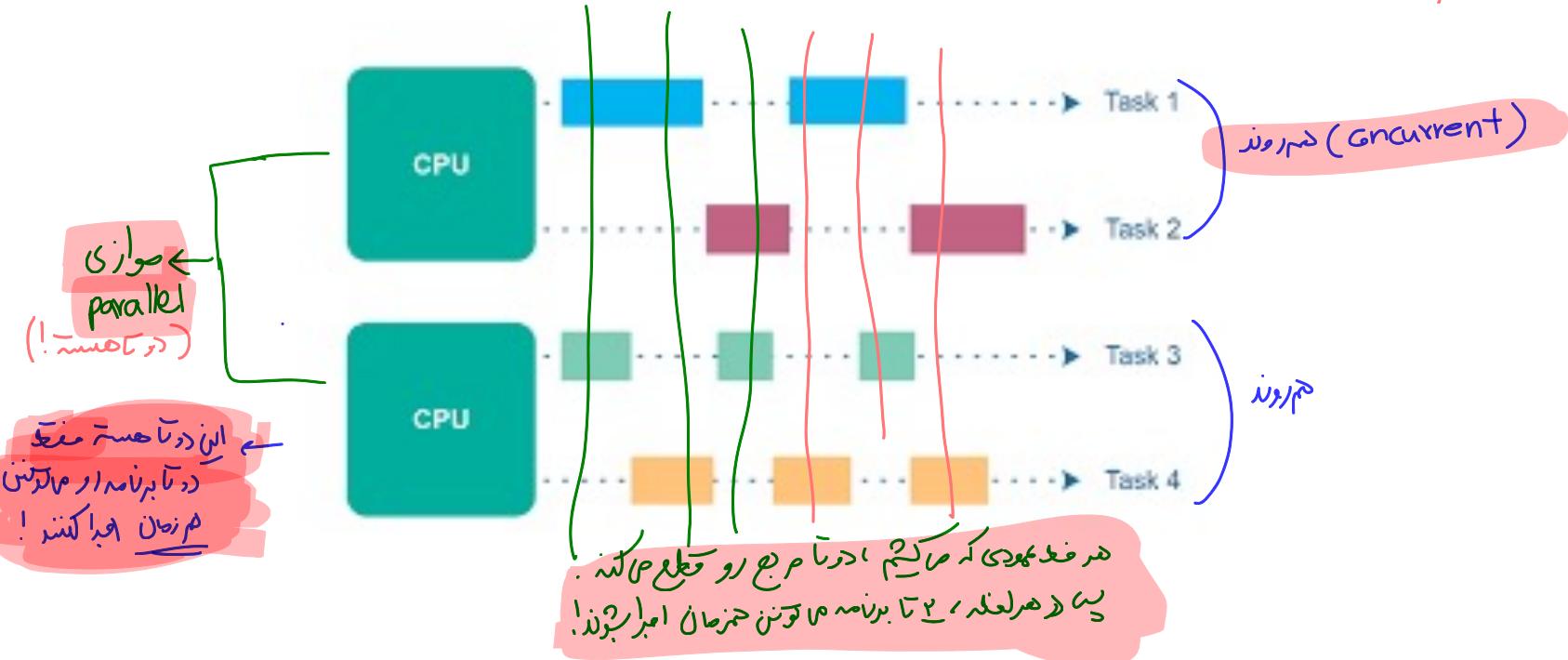


Multicore Programming (cont.)



- Parallelism implies performing more than one task simultaneously

همزبان



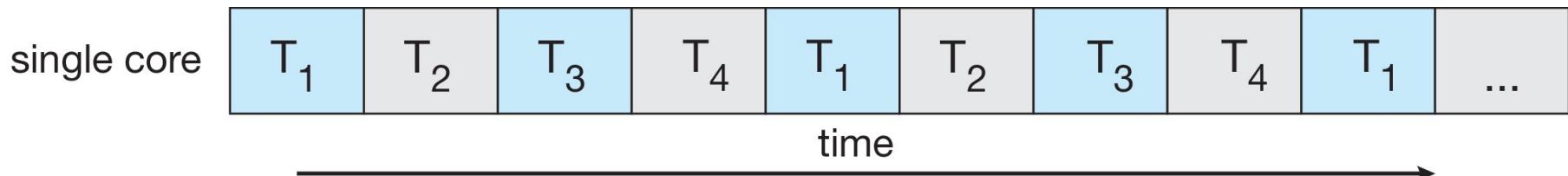
- Concurrency supports more than one task **making progress**

- Single processor or core, scheduler providing concurrency

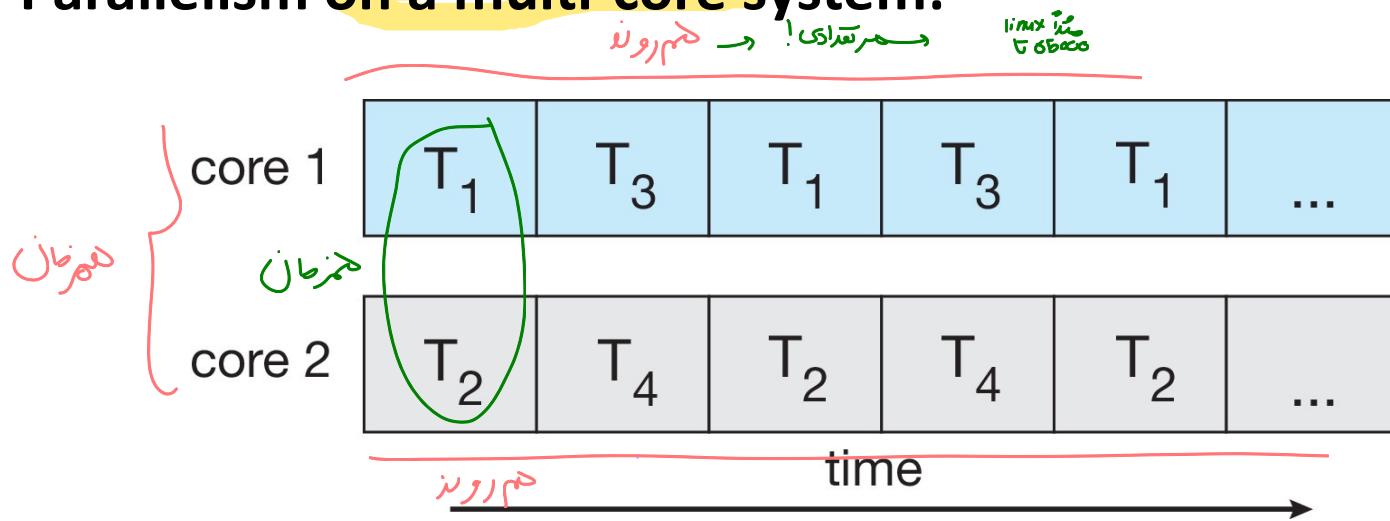
برای یک پردازنده مخصوصی که 65000 Thread دارد این اجرا می شود

Concurrency vs. Parallelism

- Concurrent execution on single-core system:



- Parallelism on a multi-core system:



Multicore Programming-Types of Parallelism

■ Data parallelism

- Distributes subsets of the same data across multiple cores, same operation on each
- Example: summing the contents of an array of size N.

thread چند جایزه، هر کدام یکی است

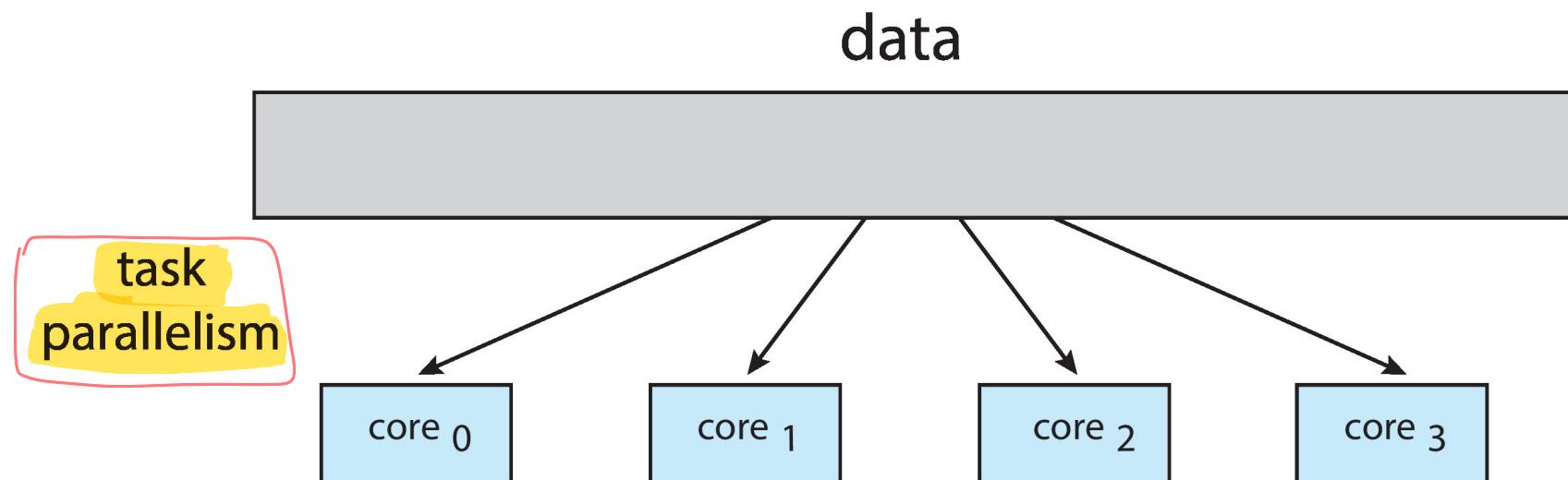
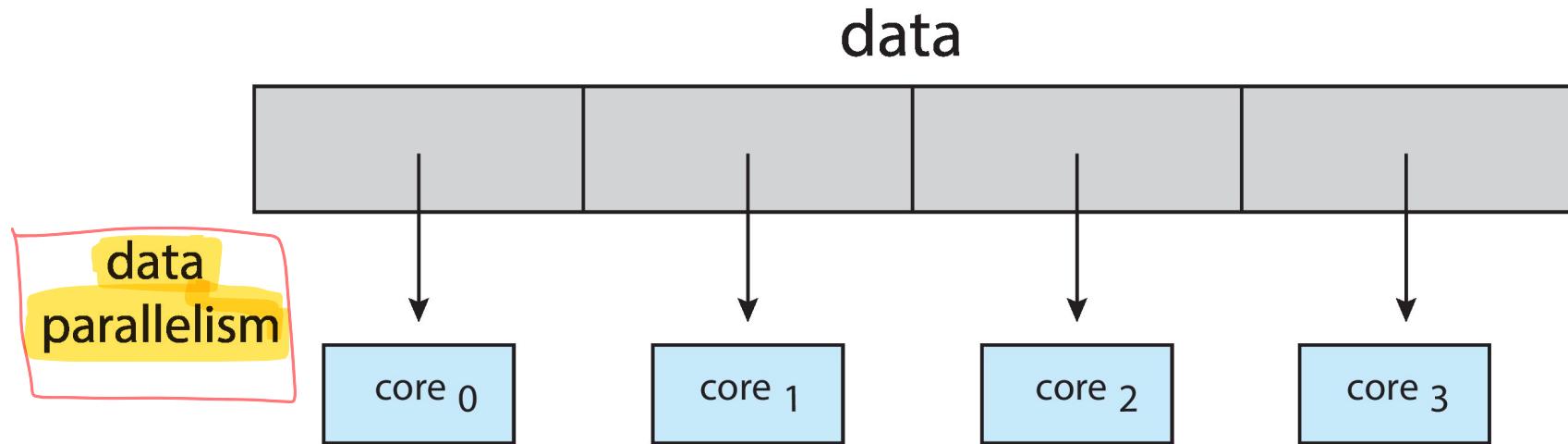
■ Task parallelism

- Distributing threads across cores, each thread performing unique operation
- Example: Unique statistical operation on the array of elements.

میان سنجی، sum، min، max

این میان سنجی را که اخیراً در متدهای Thread استفاده شده است
و هم درینجا نیست! databus، memory، I/O

Data and Task Parallelism



User Threads and Kernel Threads

- **User threads:** management done by user-level threads library.

ادن رهشانه! از باخنه نه خبر ندارد!
data

پیوپیوی (U)! جس پیوپیوی منت!

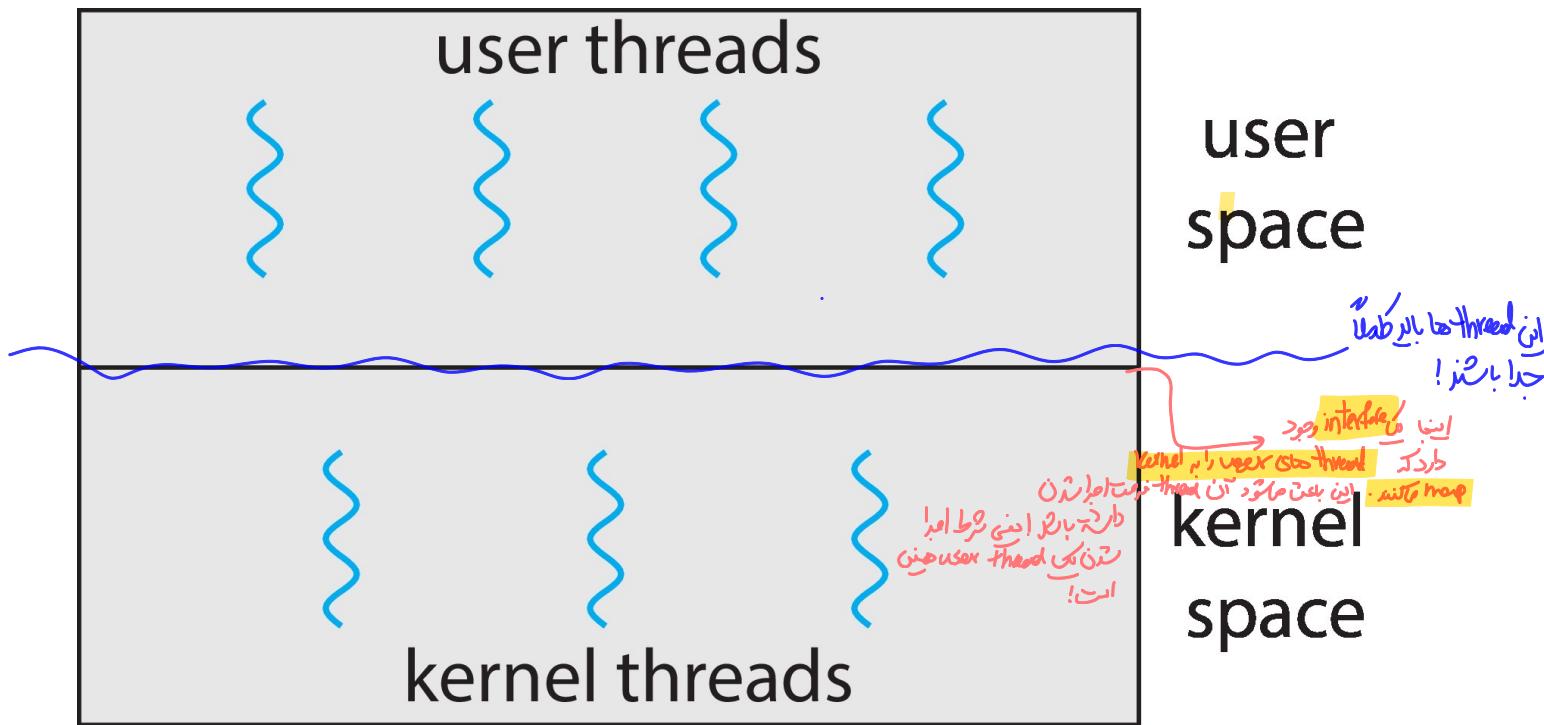
- Three primary thread libraries:

- POSIX Pthreads
- Windows threads
- Java threads

Thread → User copy
pc file open
... Registers

- **Kernel threads:** supported by the Kernel → ناچاری داشت
- Examples – virtually all general-purpose operating systems, including: Windows, Linux, Mac OS X, iOS, Android

User and Kernel Threads



Additional review: <https://www.geeksforgeeks.org/difference-between-user-level-thread-and-kernel-level-thread/>

Multithreading Models

- How to map user threads to kernel threads?
- Many-to-One
- One-to-One
- Many-to-Many

Additional review:

<https://stackoverflow.com/questions/14791278/threads-why-must-all-user-threads-be-mapped-to-a-kernel-thread>

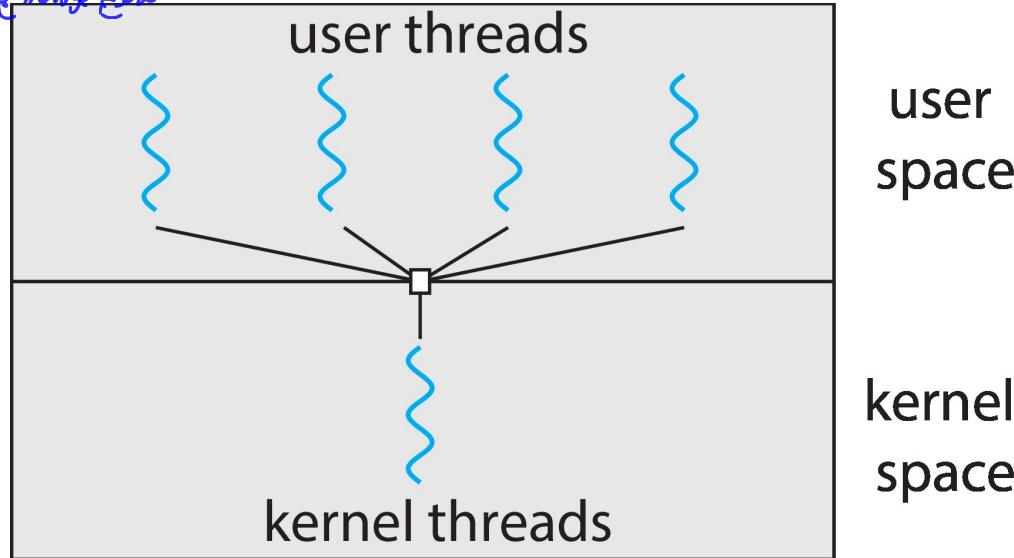
→ ↗

Many-to-One

- Many user-level threads mapped to single kernel thread
متعدد پردازنده (پردازشی) از اندیšا (پردازشی) که در نظر نمی‌گیرد.
- Thread management is done by the thread library in user space
اداره پردازشی در فضای کاربر از اندیšا (پردازشی) اینکه که در نظر نمی‌گیرد!

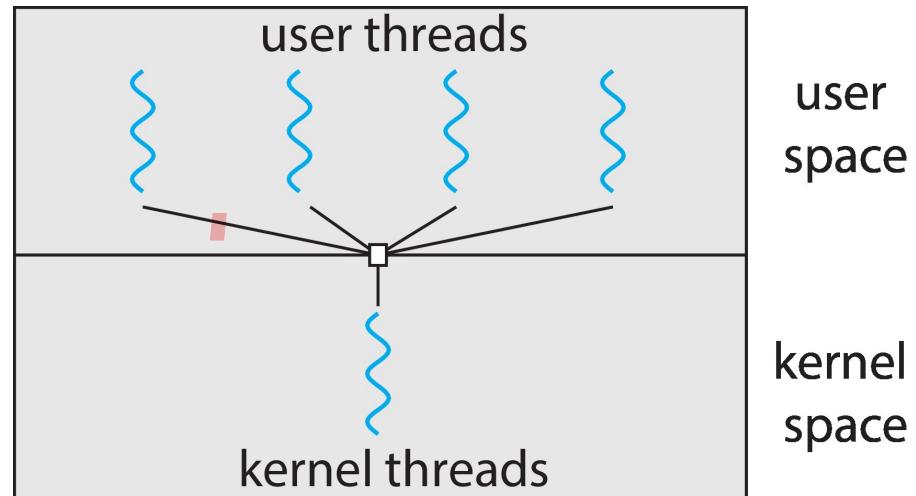
- So it is efficient

کتابخانه پردازشی کاربری برای اینکه که در نظر نمی‌گیرد! حرفی! اینکه که در نظر نمی‌گیرد! اینکه که در نظر نمی‌گیرد! اینکه که در نظر نمی‌گیرد! اینکه که در نظر نمی‌گیرد!



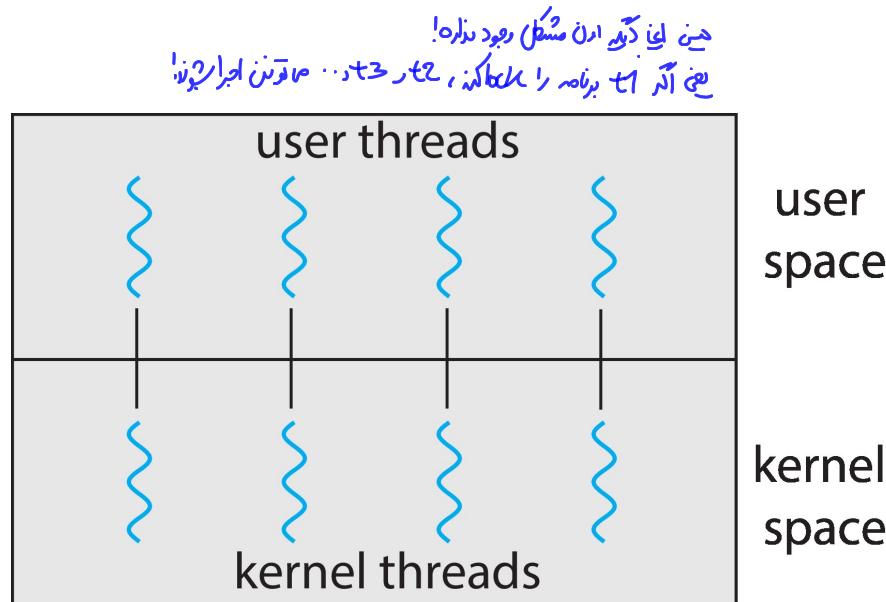
Many-to-One (cont.)

- One thread blocking causes all to block
- *Multiple threads may not run in parallel on multicore system*
 - Because only one may be in kernel at a time
- Few systems currently use this model
- Examples:
 - Solaris Green Threads
 - GNU Portable Threads



One-to-One

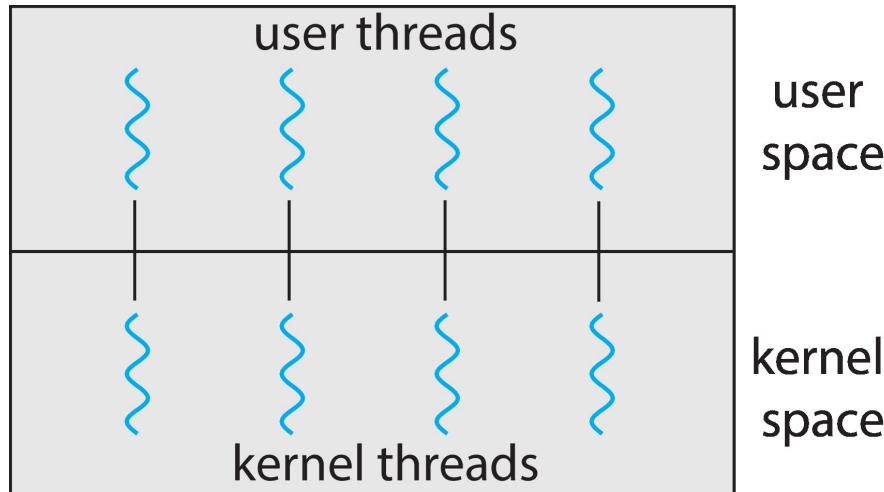
- Each user-level thread maps to kernel thread
- Creating a user-level thread creates a kernel thread



One-to-One (cont.)

- More concurrency than many-to-one
- Number of threads per process may be restricted due to overhead
- Examples

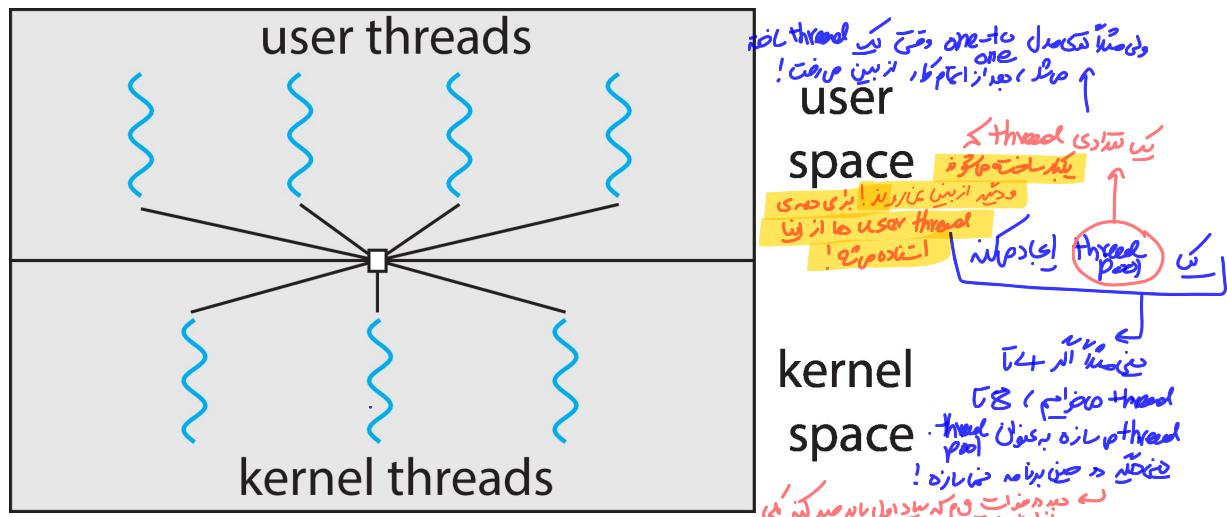
- Windows
- Linux



It also allows another thread to run when a thread makes a blocking system call. It supports multiple threads to execute in parallel on microprocessors.

Many-to-Many Model

- Many user level threads to be mapped to many kernel threads
- Operating system can create a sufficient number of kernel threads



- Examples
 - Windows with the *ThreadFiber* package
 - Otherwise not very common

Two-level Model

- Similar to M:M, except that it allows a user thread to be **bound** to kernel thread

