دانشگاه صنعتی شاهرود
دانشکده مهندسی کامپیوتر و فناوری اطلاعات

سیستم‌های عامل

فرآیندها و نخ‌ها

استاد: علیرضا تجری

فرآیندها

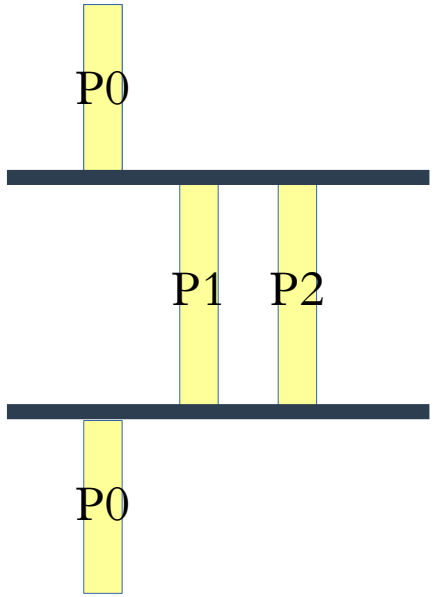
- مزایای فرآیند
  - برنامه نویس بدون توجه به فرآیندها دیگر در حال اجرا برنامه خود را می‌نویسد.
  - هر فرآیند فقط به حافظه و منابع خود دسترسی دارد.
    - فرآیندها در کار هم تداخلی ایجاد نمی‌کنند.
      - مگر اینکه کسی بخواهد عمداً این کار را انجام دهد.
  - امکان همکاری فرآیندها وجود دارد.
  - از منابع سیستم کامپیوتری به صورت عادلانه و بهینه استفاده می‌شود.

فرآیندها

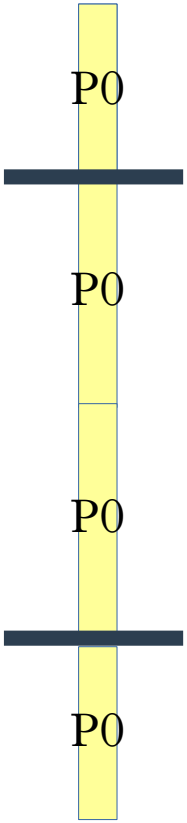
- برخی از کاربردهای فرآیندها
  - پردازش موازی
    - کاربرد محاسبات با کارآیی بالا: HPC High Performance Computing
    - الگوریتم‌های موازی: بخشی از الگوریتم می‌تواند به صورت موازی اجرا شود.
      - مثلاً Merge Sort: اگر n داده داشته باشید، می‌توانید آن‌ها را نصف کنید، هر بخش از این داده به صورت جدا گانه مرتب می‌شود و پس از آن، بخش‌های مرتب شده، در هم ادغام می‌شوند. مرتب کردن دو بخش داده، می‌تواند به صورت موازی انجام شود (برای هر بخش موازی، یک فرآیند جدید ایجاد می‌شود).
    - استفاده از الگوریتم‌های موازی در کامپیوترهایی که یک پردازنده تک هسته‌ی دارند، تا چه اندازه باعث کارآیی بیشتر می‌شود؟
      - تقریباً هیچ!، چون فرآیندهای موازی می‌توانند به صورت موازی اجرا شوند، اما از آنجا که فقط یک پردازنده وجود دارد، در عمل، کدهای آن‌ها به صورت سری اجرا می‌شود.

سیستم‌های تک پردازنده‌ای

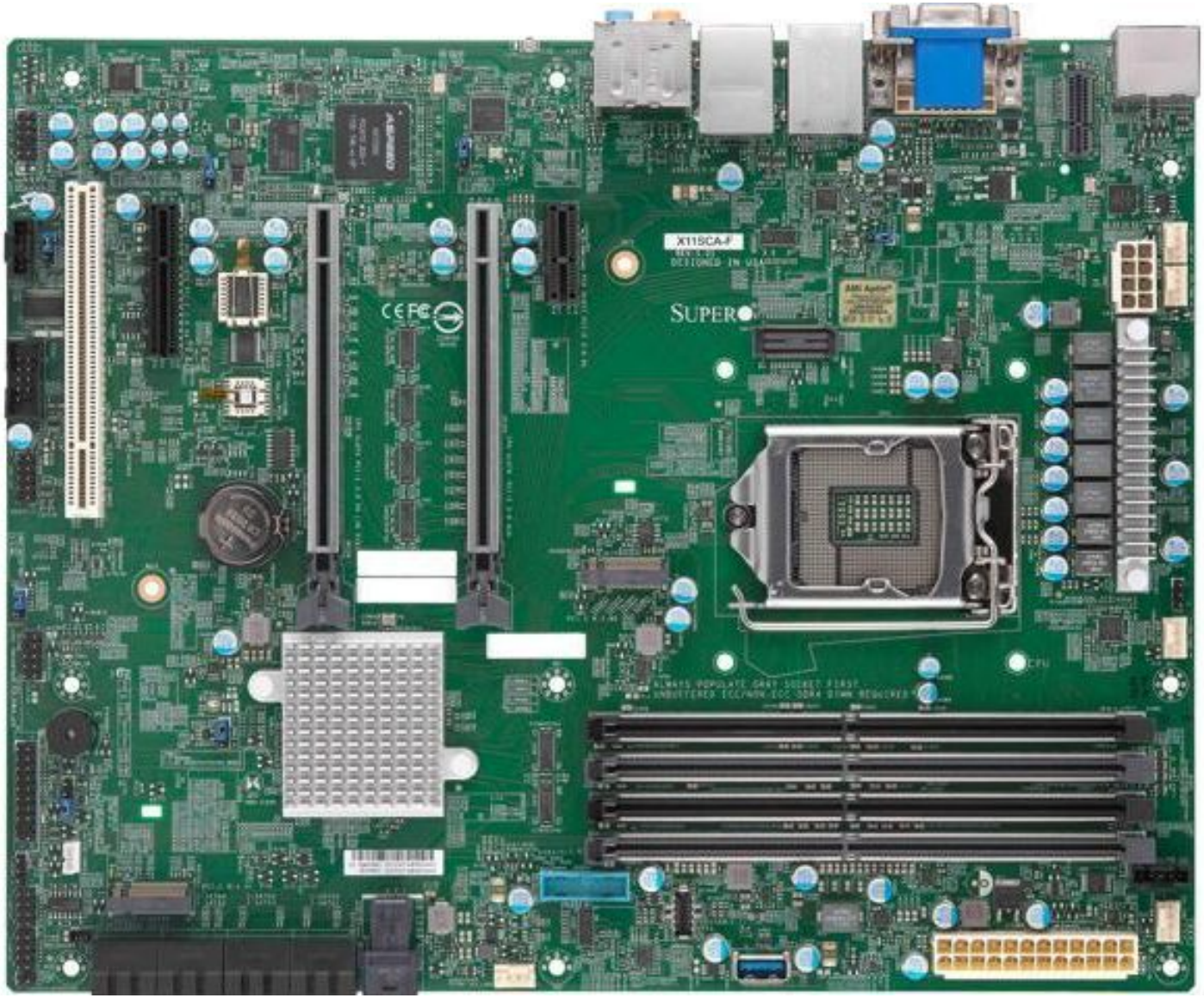
الگوریتم Merge Sort به صورت موازی

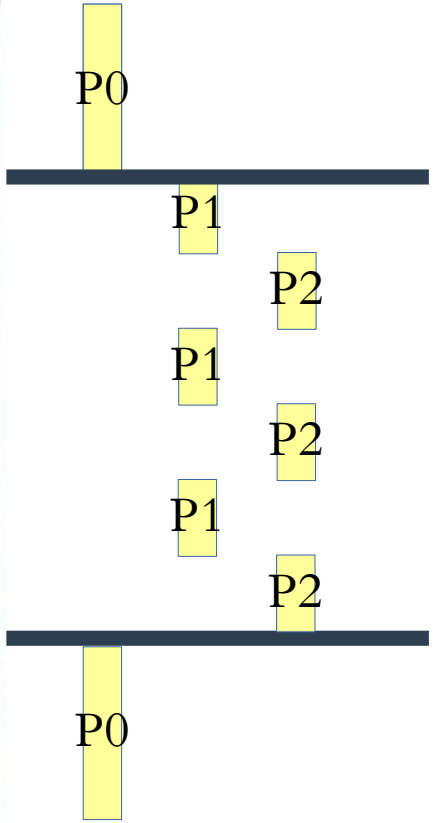


الگوریتم Merge Sort به صورت سری



سیستم‌های تک پردازنده‌ای

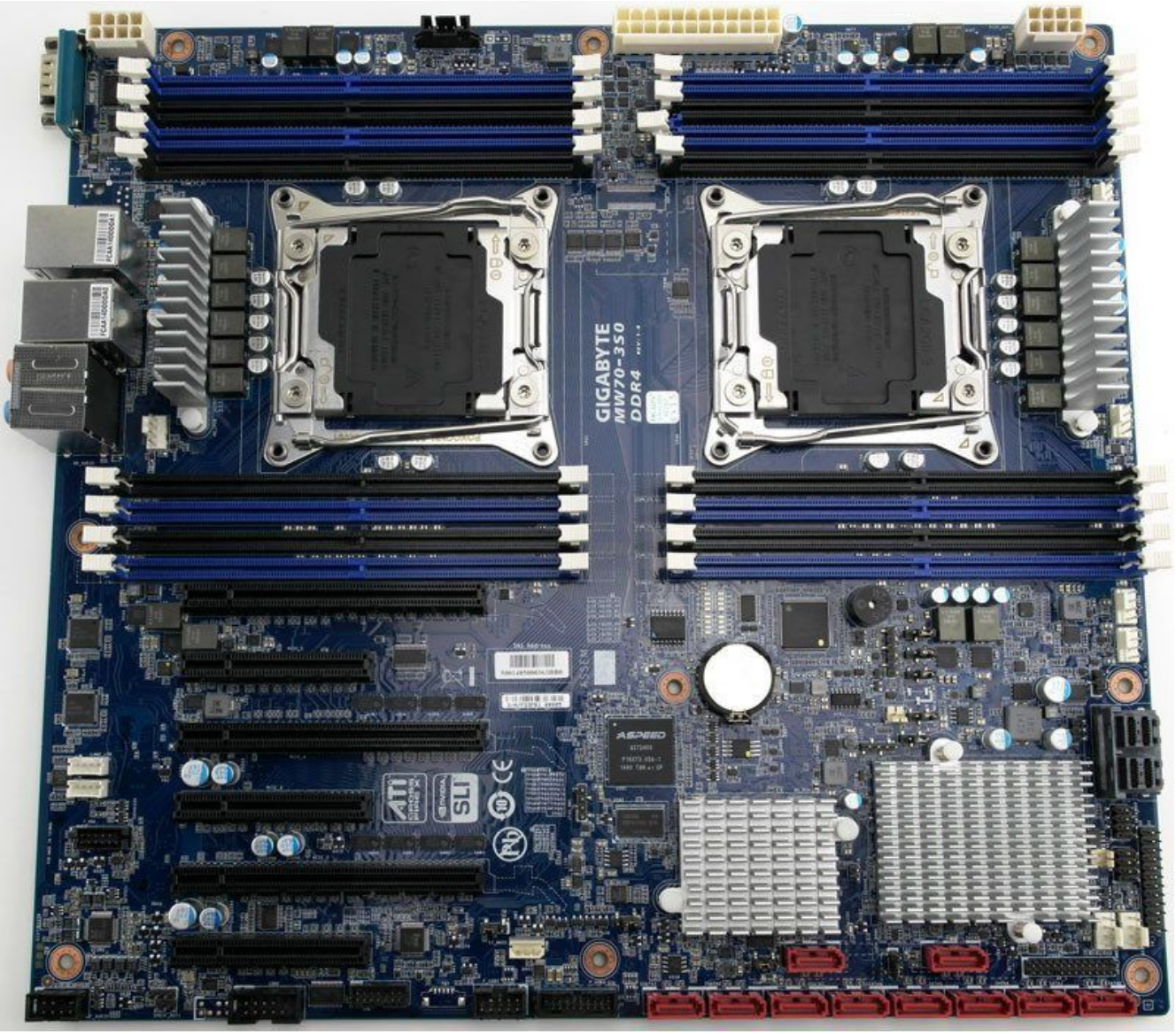


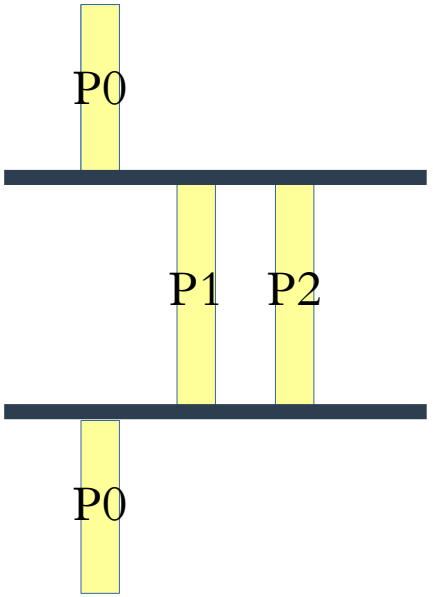


فرآیندها

- برخی از کاربردهای فرآیندها
  - پردازش موازی
    - نیاز به منابع موازی دارد
      - مهمترین منبع: پردازنده
    - آیا امکان دارد دو پردازنده داشته باشیم؟
      - بله، در محاسبات HPC و کاربردهای سرور، بیش از یک پردازنده وجود دارد.
    - زمانی که چند پردازنده داریم، الگوریتم‌های موازی، سریعتر از الگوریتم‌های سری اجرا می‌شوند.
    - بخش‌های موازی به صورت فرآیند ایجاد می‌شوند.
      - با این کار، می‌توانند به صورت موازی اجرا شوند.

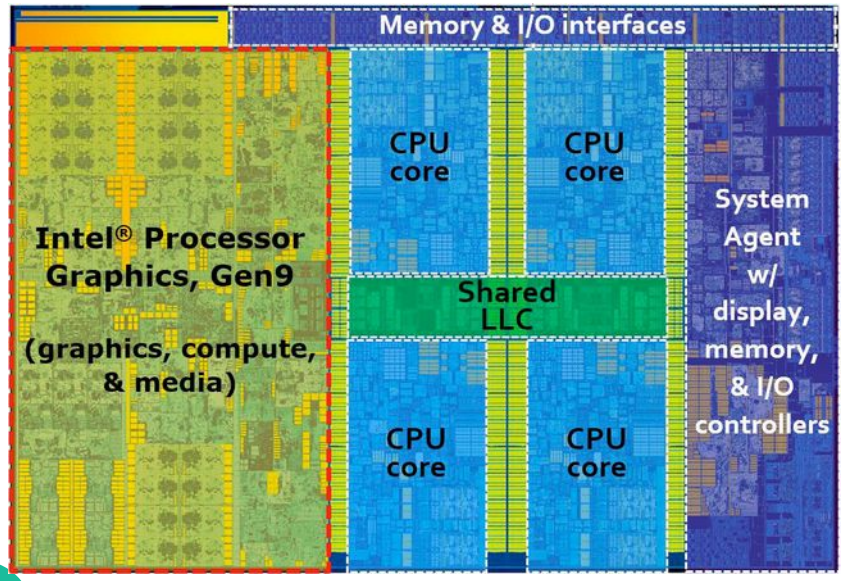
سیستم‌های چند پردازنده‌ای

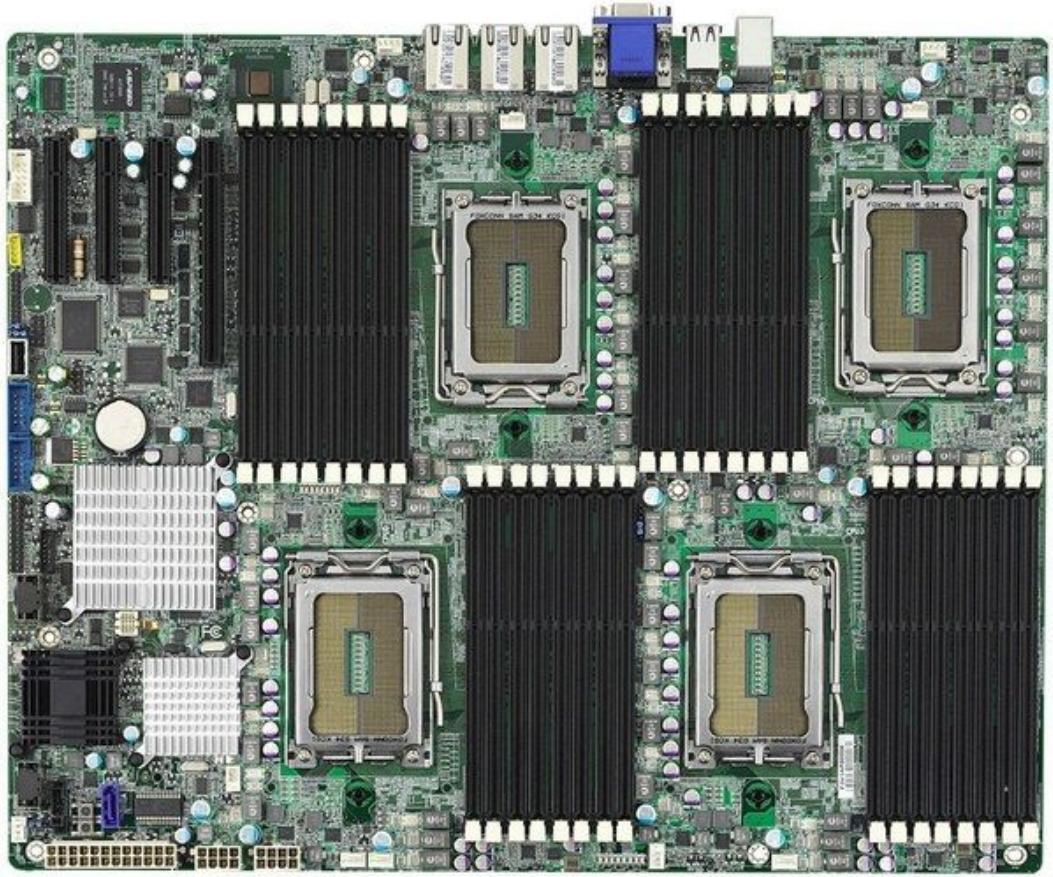




سیستم‌های چند پردازنده‌ای

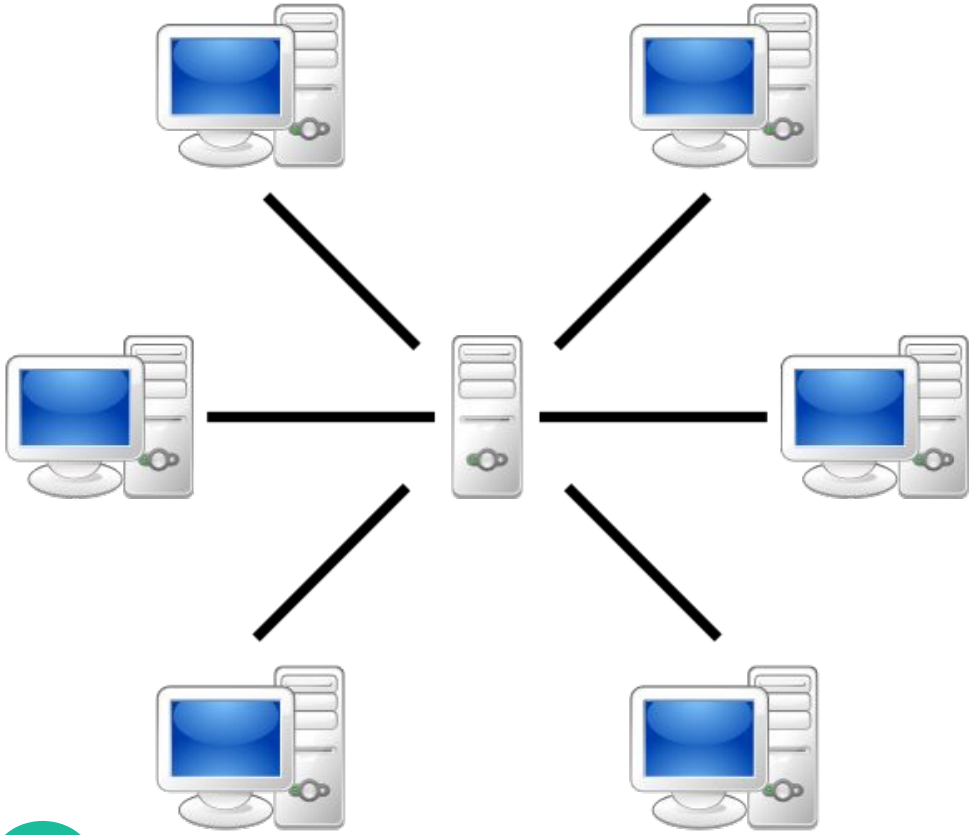
معماری‌های چند پردازنده چند هسته‌ای





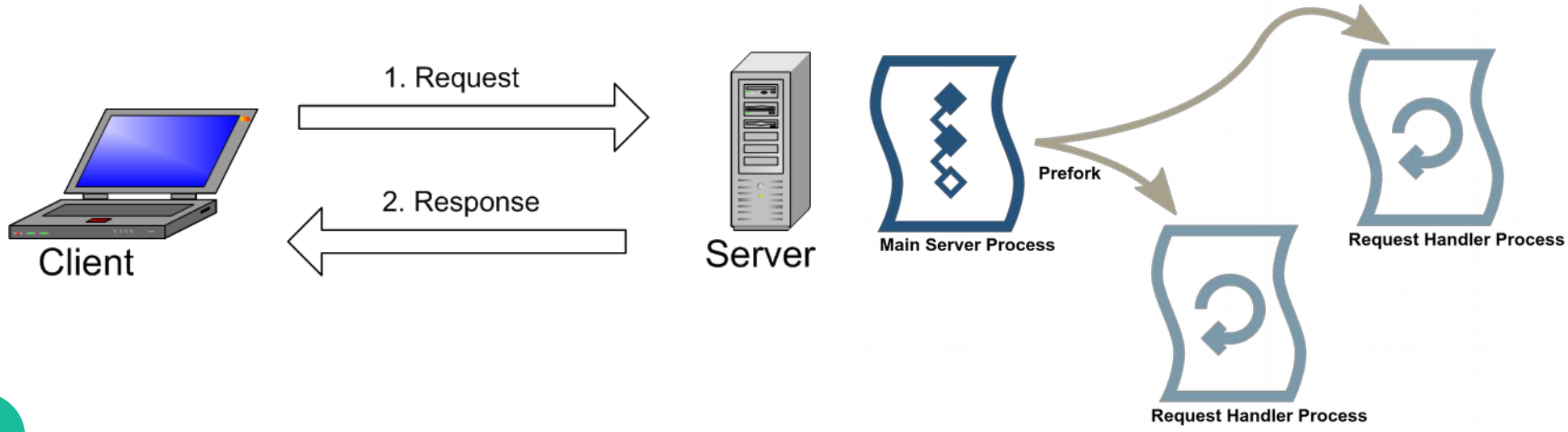
فرآیندها

- برخی از کاربردهای فرآیندها
  - نرم‌افزارهای سرور
    - معماری کلاینت سرور
      - سرور وب
      - سرور ایمیل
      - سرور تلگرام!
      - ...



فرآیندها

- برخی از کاربردهای فرآیندها
  - دو نوع فرآیند در نرم‌افزارهای سرور:
    - فرآیند اصلی سرور
    - فرآیند پاسخ به کلاینت
- با هر درخواست کلاینت، یک فرآیند برای پاسخ به کلاینت ایجاد می‌شود.



فرآیندها

- برخی از کاربردهای فرآیندها
  - نرم‌افزارهای سرور
    - با هر درخواست کلاینت، یک فرآیند برای پاسخ به کلاینت ایجاد می‌شود.
    - کد (بخش text فرآیند) فرآیندهای پاسخ‌دهی به کلاینت یکسان است.
    - پس از مدتی، اگر به حافظه اصلی نگاه کنیم، تعداد زیادی فرآیند می‌بینیم (به تعداد درخواست‌های فعال) که همه دارای کد یکسان هستند.
      - کاش می‌شد فقط یک نسخه از کد در حافظه قرار می‌گرفت!
      - ایجاد کردن فرآیند یک کار زمانبر است.

آیا راهی وجود دارد که بدون ایجاد فرآیند جدید، بتوانیم بخشی از کد را به صورت موازی اجرا کنیم؟

نخ‌ها

آیا راهی وجود دارد که بدون ایجاد فرآیند جدید، بتوانیم بخشی از کد را به صورت موازی اجرا کنیم؟

بله!

با استفاده از Thread ها

(نخ، ریسمان)

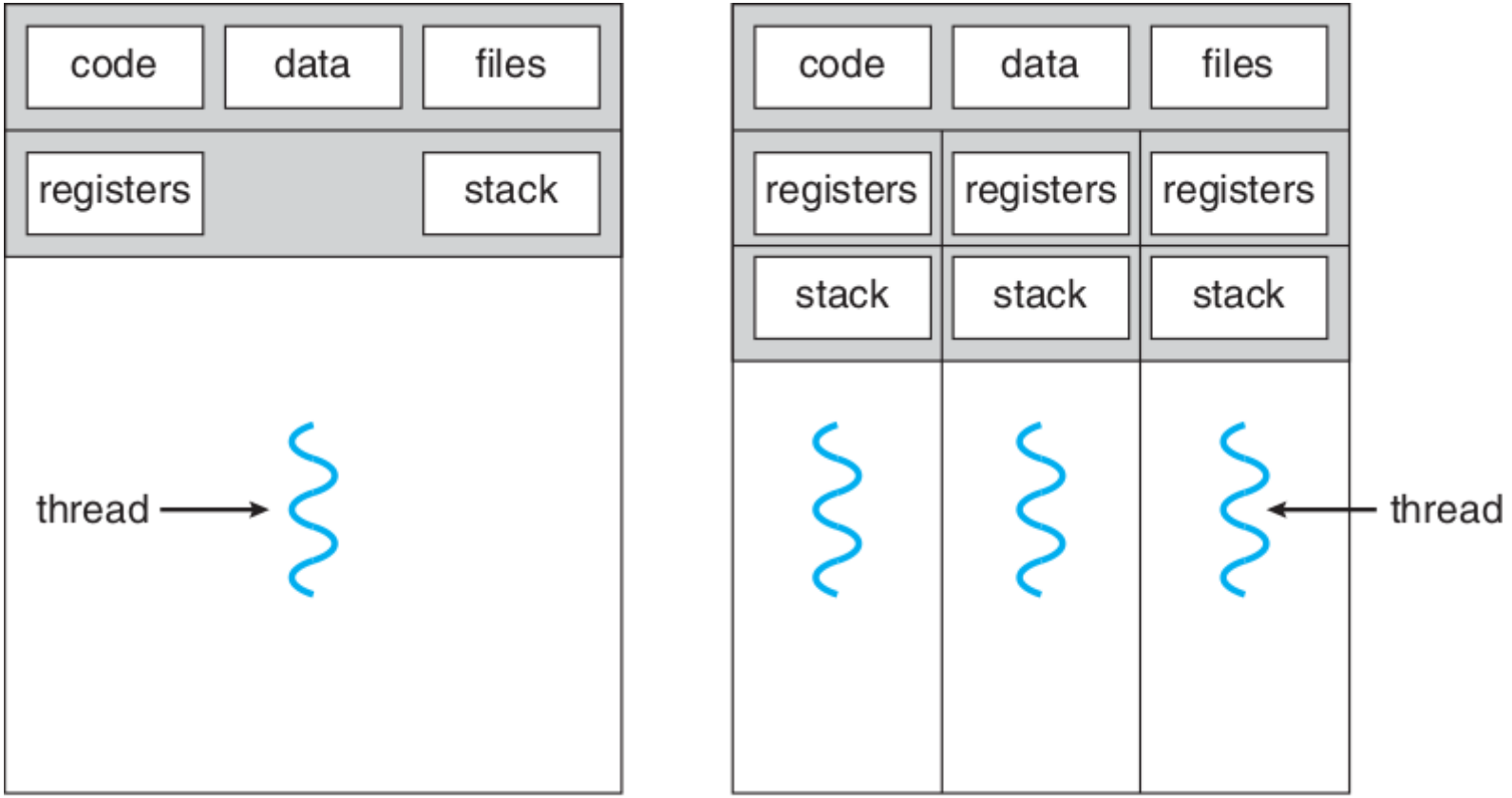
با استفاده از Thread،
می‌توان بخشی از کد (که معمولا یک تابع است) را
به صورت موازی اجرا کنیم.
(البته در یک فرآیند)

نخ‌ها

- فرآیندها
  - تک نخی Single Threaded
    - فرآیندهای معمولی
    - فقط یک دنباله اجرای دستورات وجود دارد.
    - اکثر برنامه‌هایی که شما نوشتید به این صورت است.
  - چند نخی Multi-Threaded
    - چند دنباله اجرای دستورات دارد!
    - به هر دنباله اجرا یک نخ گفته می‌شود.
    - هر دنباله اجرا دارای رجیسترهای پردازنده و stack مربوط به خود است.

نخ‌ها

کد و داده فرآیند بین همه نخ‌های فرآیند مشترک است.



مثال از نخ

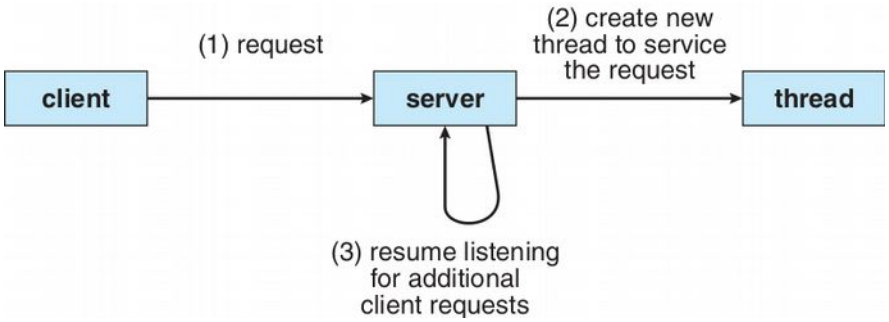
```
class MultithreadingDemo extends Thread{
    public void run(){
        try{
            // Displaying the thread that is running
            System.out.println ("Thread " +
                    Thread.currentThread().getId() +
                    " is running");

        }catch (Exception e){
            // Throwing an exception
            System.out.println ("Exception is caught");
        }
    }
}
```

مثال از نخ

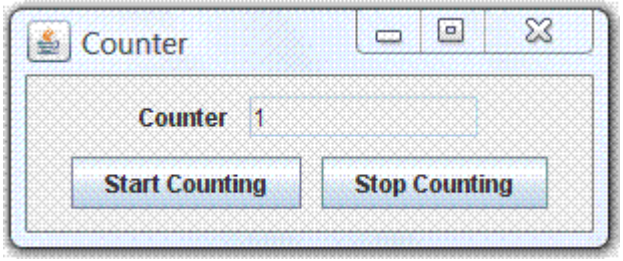
```
// Main Class
public class Multithread{
    public static void main(String[] args){
        int n = 8; // Number of threads
        for (int i=0; i<n; i++){
            MultithreadingDemo object = new
                MultithreadingDemo();
            object.start();
        }
    }
}
```

کاربردهای نخ

- معماری کلاینت سرور



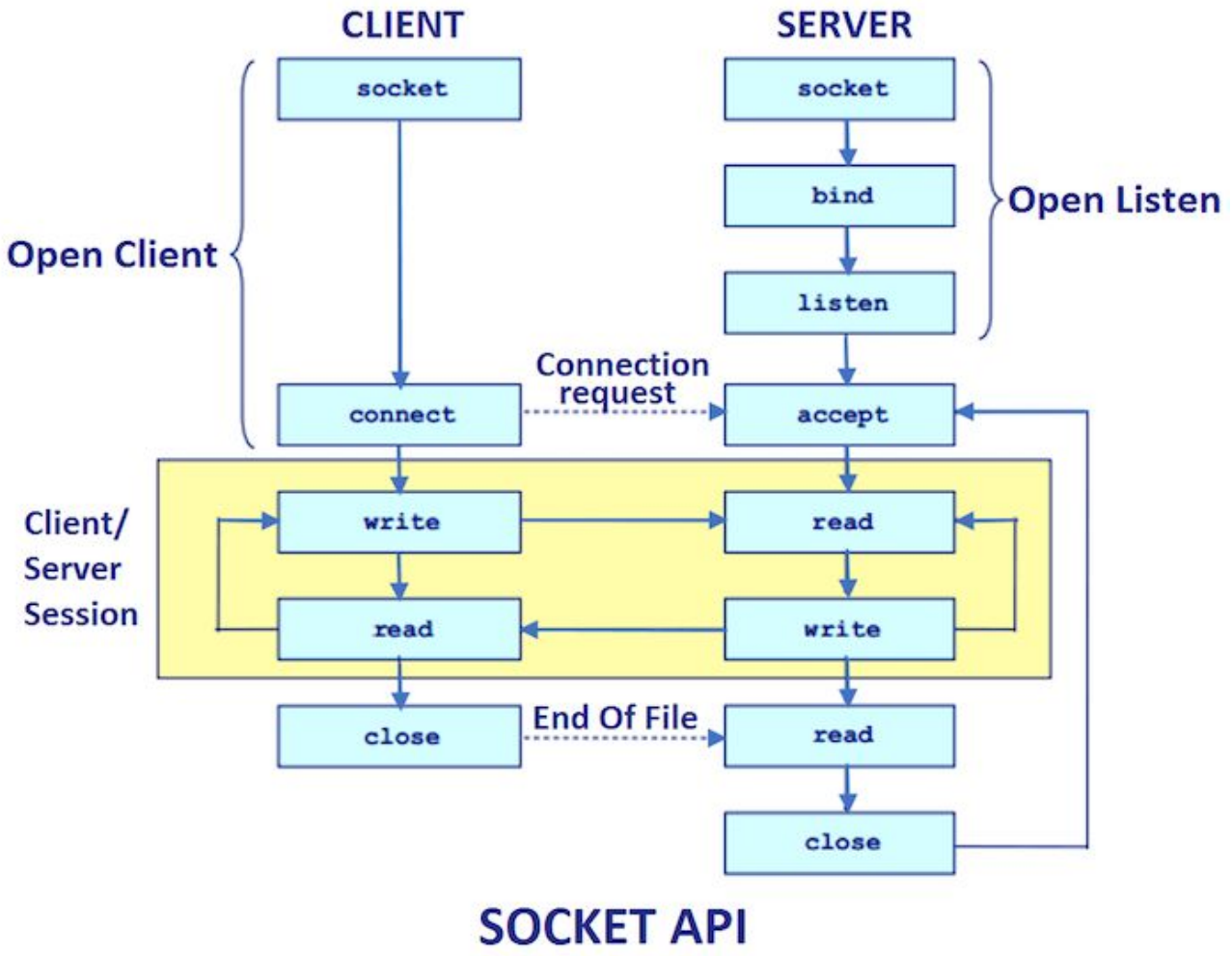
- واسط‌های کاربری گرافیکی **GUI**



- محاسبات با کارآیی بالا **HPC**

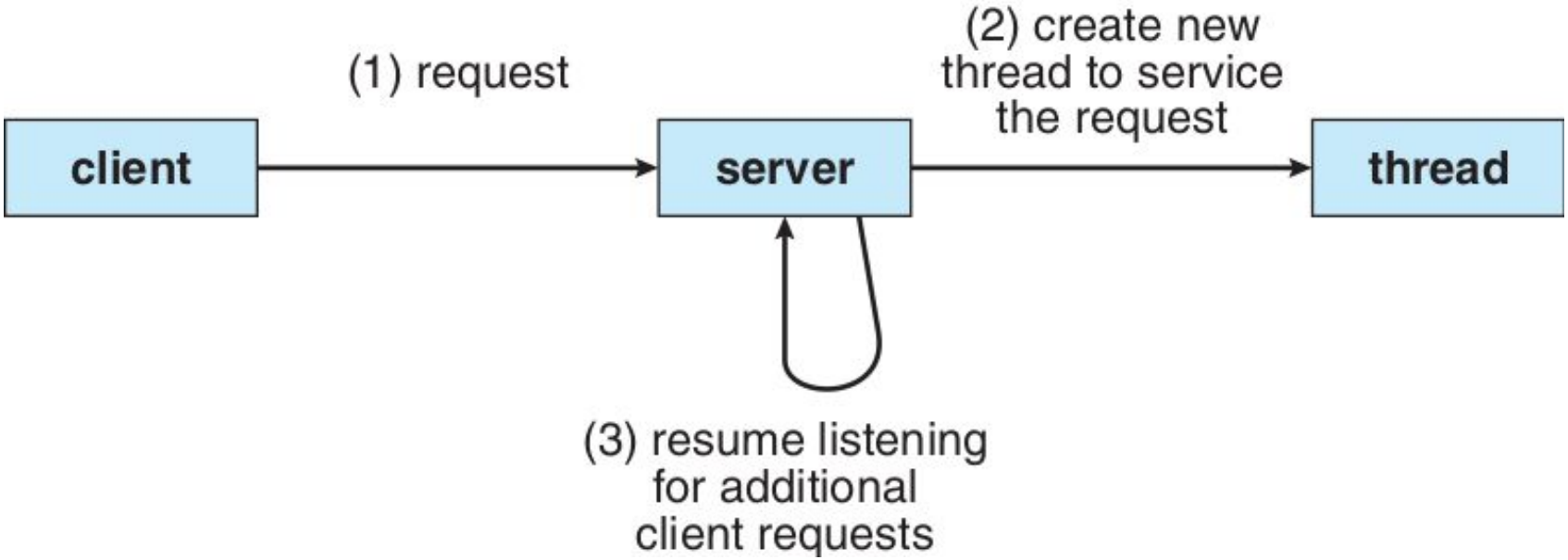
کاربردهای نخ

- معماری کلاینت سرور



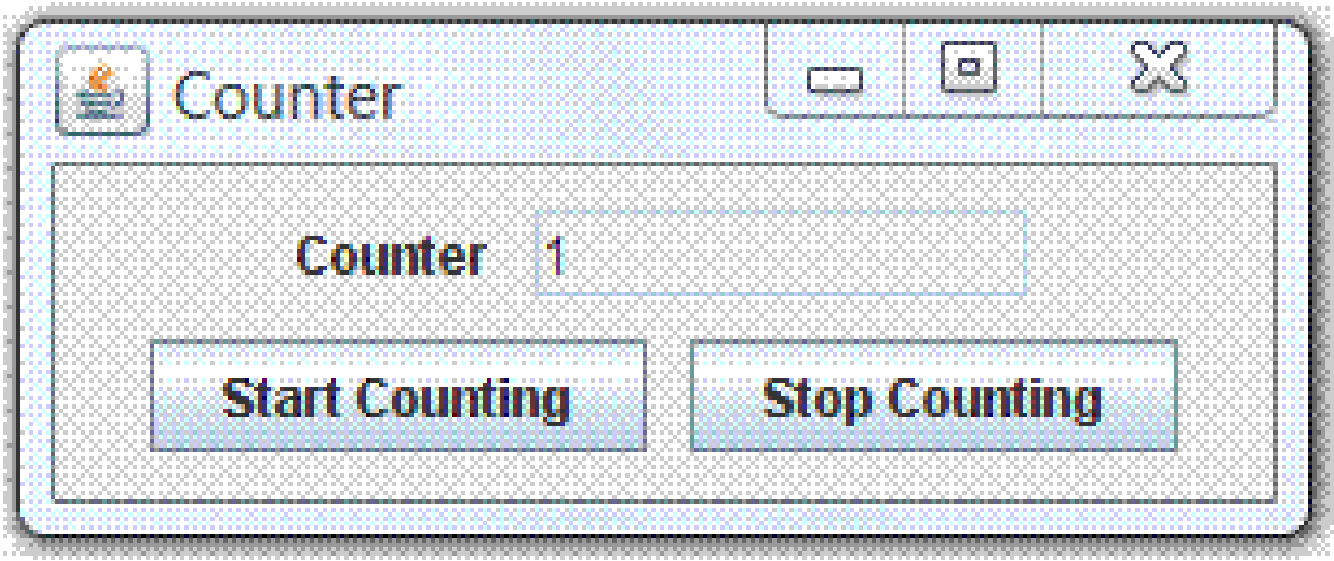
کاربردهای نخ

- معماری کلاینت سرور



کاربردهای نخ

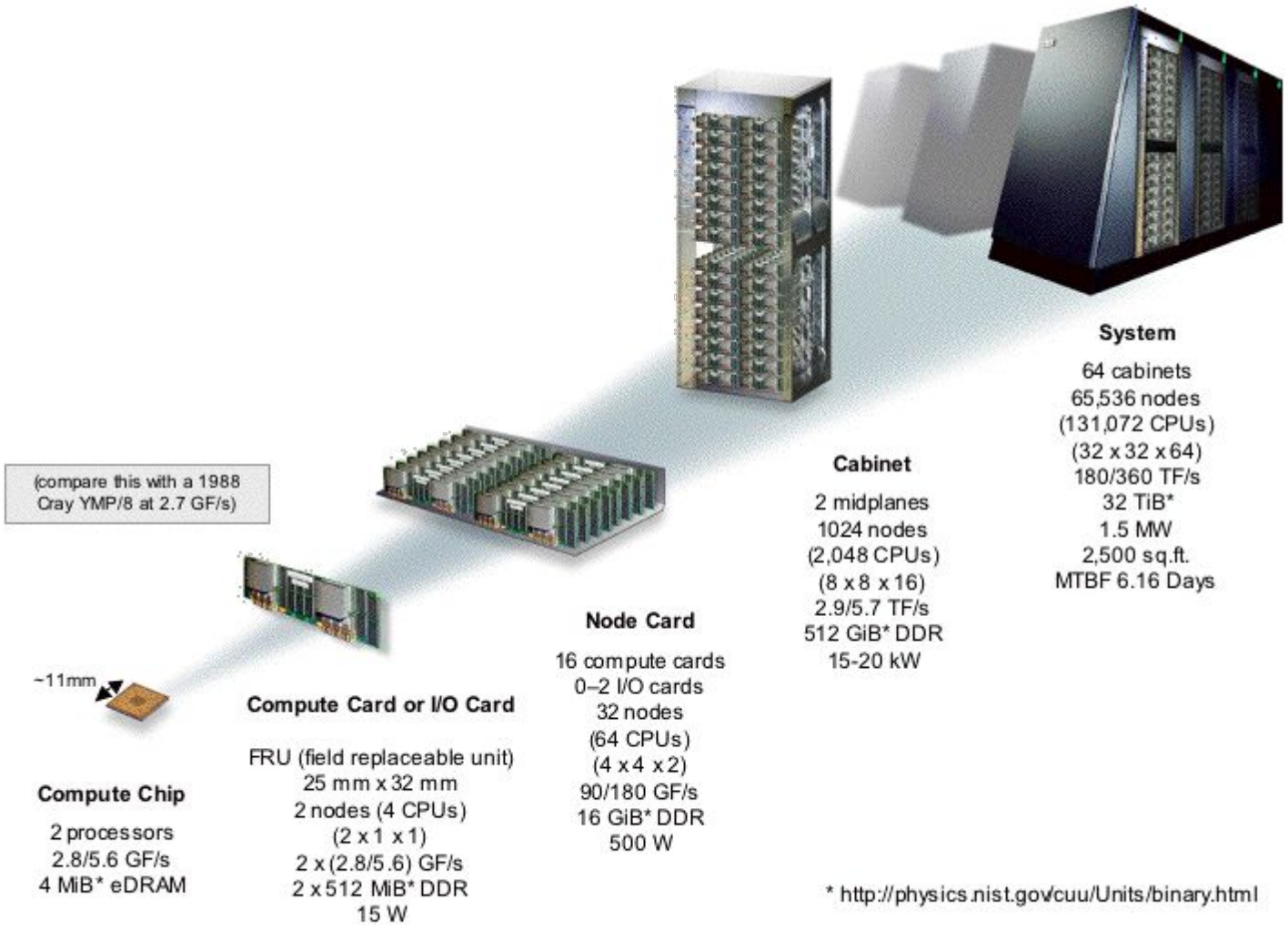
- واسط‌های کاربری گرافیکی GUI



مثال همراه با کد

کاربردهای نخ

- محاسبات با کارآیی بالا **HPC**



کاربردهای نخ

- محاسبات با کارآیی بالا HPC

مزایای استفاده از Thread

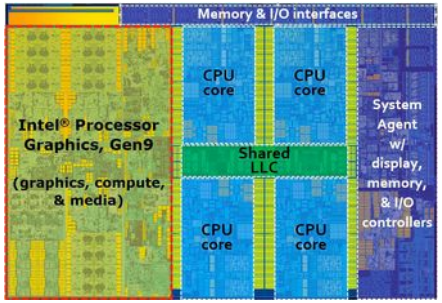
- پاسخ‌گویی مناسب **Responsiveness**
  - در GUI ها
- به اشتراک گذاری منابع (**Resource Sharing**)، به صورت راحت‌تر
  - منابع و داده‌ها به صورت پیش‌فرض بین نخ‌ها مشترک است.
- طراحی اقتصادی‌تر
  - منابع مورد نیاز برای ایجاد یک نخ، از منابع مورد نیاز برای ایجاد یک فرآیند بسیار کمتر است.
  - مدت زمان ایجاد یک فرآیند، ۳۰ برابر مدت زمان ایجاد یک نخ بوده است.
- مقیاس‌پذیری
  - نخ‌ها می‌توانند بر روی هسته‌های متفاوت اجرا شوند.
  - با اضافه کردن هسته‌ها، می‌توان تعداد نخ‌ها را اضافه کرد.

روند معماری پردازنده‌ها در سیستم‌های کامپیوتری

- یک پردازنده تک هسته‌ای

- چند پردازنده تک هسته‌ای

- یک پردازنده چند هسته‌ای

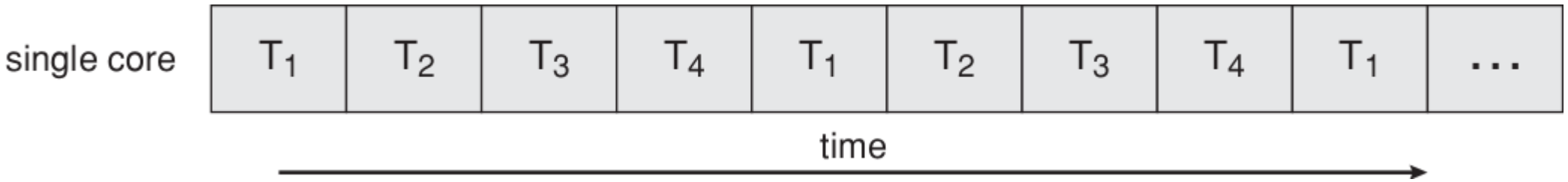


- چند پردازنده چند هسته‌ای

اجرای نخ‌ها در پردازنده تک هسته‌ای

- فرض کنید یک فرآیند دارای ۴ نخ است.

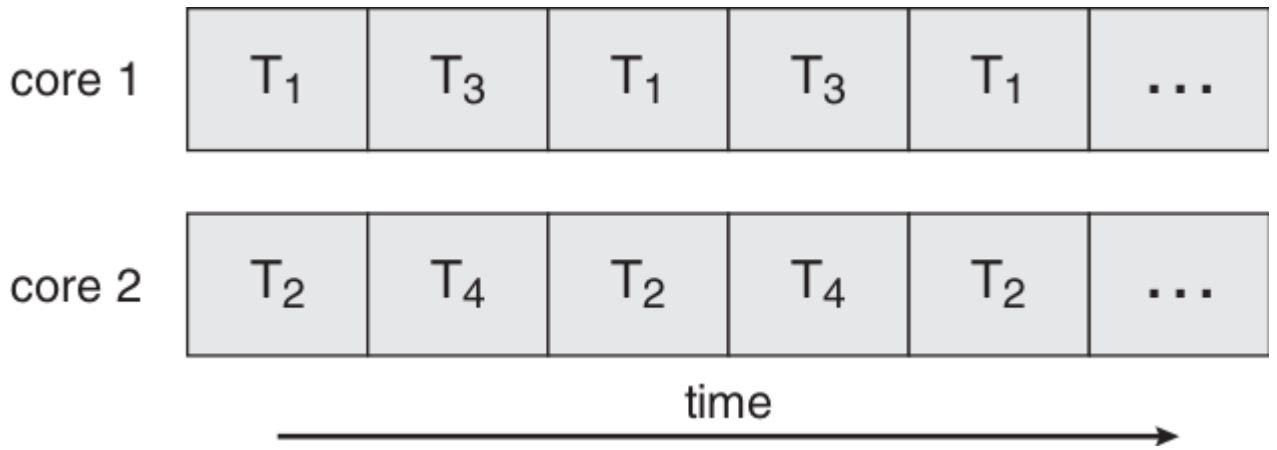
نحوه اجرا بر روی پردازنده تک هسته‌ای



اجرای نخ‌ها در پردازنده چند هسته‌ای

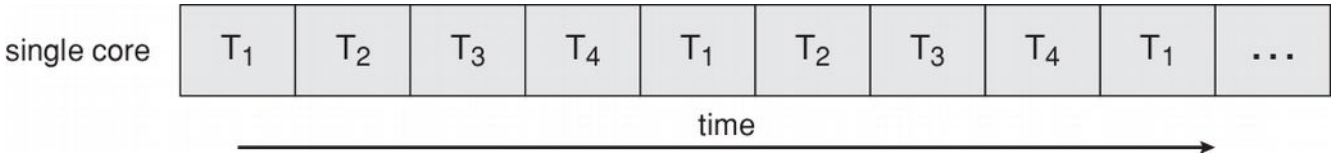
- فرض کنید یک فرآیند دارای ۴ نخ است.

نحوه اجرا بر روی پردازنده دو هسته‌ای



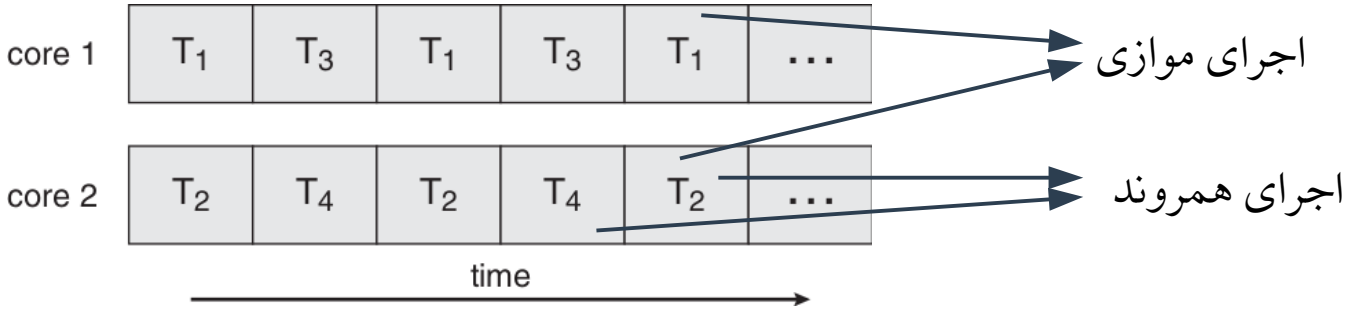
اجرای همروند و اجرای موازی

- اجرای دو نخ، نسبت به هم، می‌تواند همروند یا موازی باشد.
- اجرای همروند: دو نخ بر روی یک هسته، اجرا می‌شوند.



اجرای موازی: دو نخ بر روی دو هسته متفاوت، اجرا می‌شوند.

- در هر لحظه، دستورات هر دو نخ اجرا می‌شود.



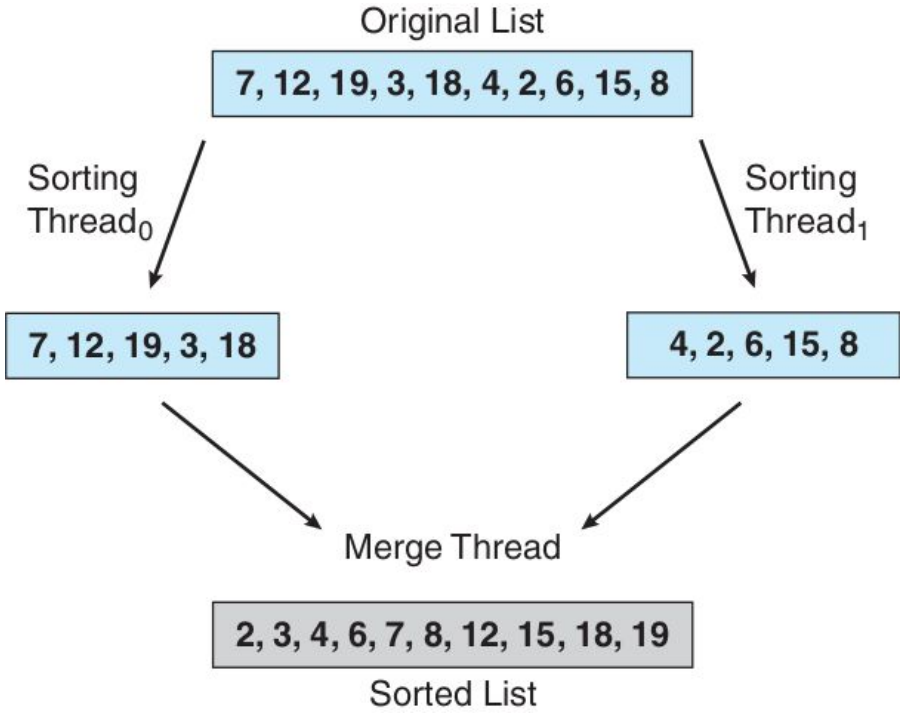
انواع موازی سازی

- موازی سازی داده
  - داده‌ها را به چند بخش تقسیم می‌کنیم و پردازش بخش‌های داده‌ها را به صورت موازی انجام می‌دهیم.
- موازی سازی کار
  - کار را به چند بخش مستقل تقسیم می‌کنیم، به گونه‌ای که این بخش‌ها بتوانند به صورت موازی اجرا شوند.

انواع موازی سازی

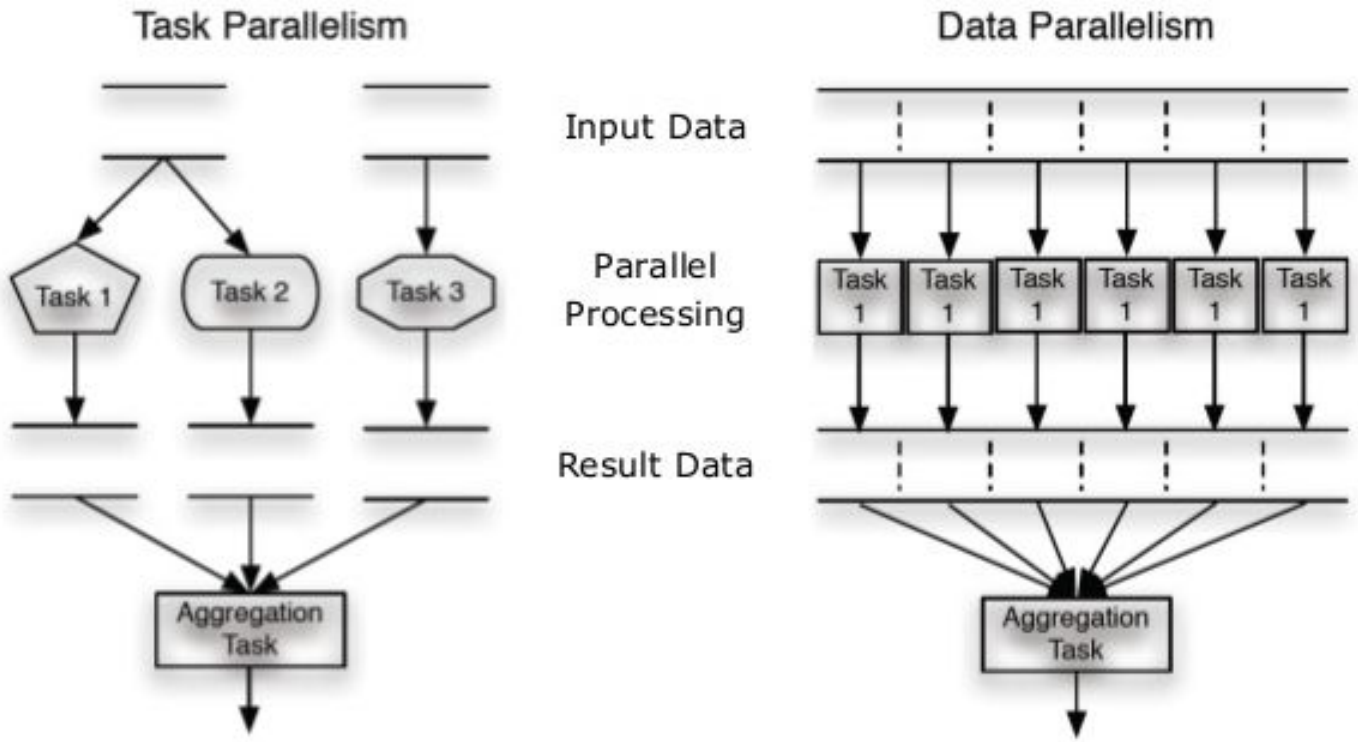
- موازی سازی داده

  - مثل مرتب کردن n عدد: آن‌ها را به دو بخش تقسیم می‌کنیم، این دو بخش را به صورت موازی مرتب می‌کنیم و نتایج مرتب شده را با هم ترکیب می‌کنیم.



انواع موازی سازی

- موازی سازی کار
  - مثل محاسبه چند پارامتر آماری بر روی تعدادی داده: محاسبه هر پارامتر، یک کار مستقل است.



کتابخانه‌های ایجاد نخ

- کتابخانه **Pthread**: کتابخانه استاندارد ایجاد نخ در لینوکس (**C**)
- کتابخانه **Windows Threads**: کتابخانه استاندارد ایجاد نخ در ویندوز (**C**)
- کتابخانه‌های زبان‌های برنامه‌نویسی
  - Java
  - C#
  - Python

مدیریت نخ‌ها

- کار با نخ‌ها، سهل ممتنع است.
- یک اشتباه موجود در یک نخ، ممکن است باعث شود که کل نرم‌افزار به درستی کار نکند.
- طراحی نرم‌افزار با نخ‌ها نیازمند داشتن موارد زیر است.
  - آشنایی دقیق و کامل نخ‌ها
  - تجربه زیاد در کار با نخ‌ها
  - یک معماری مناسب

چند نخی ضمنی

- کتابخانه‌ها کمکی برای کار با نخ‌ها
  - مدیریت نخ‌ها را آسانتر می‌کنند.
- **Thread Pool**
  - تعدادی نخ از پیش ساخته وجود دارد.
  - هر نخ جدید به یک نخ از پیش‌ساخته assign می‌شود و شروع به اجرا می‌کند.
  - پس از اتمام اجرای نخ، نخ پیش‌ساخته آزاد می‌شود.
  - تعداد نخ‌های از پیش‌ساخته محدود است.
    - مدیریت منابع به خوبی انجام می‌شود.

چند نخی ضمنی

- **OpenMP**
  - ساختاری برای تعریف قطعات کد موازی در زبان C
  - چیزی به نام نخ وجود ندارد. نخ‌ها، توسط این کتابخانه ایجاد و مدیریت می‌شوند.

```
int main(int argc, char *argv[])
{
   /* sequential code */

   #pragma omp parallel
   {
      printf("I am a parallel region.");
   }

   /* sequential code */

   return 0;
}
```