

0. 《并行计算与软件设计》 课堂要求

罗翔宇

中国人民大学统计与大数据研究院

任课教师

- ▶ 任课教师：罗翔宇
- ▶ 办公电话：82507567
- ▶ 电子邮件：xiangyuluo@ruc.edu.cn
- ▶ 办公地址：崇德西楼716
- ▶ 上课时间：周三 8:00 - 10:30

助教

- ▶ 助教：吴秋雨
- ▶ 电子邮件：w.qy@ruc.edu.cn
- ▶ 微信：课程群里可加

课堂要求

- ▶ 避免迟到、早退。

若有特殊情况发生迟到、早退，或上课期间需要短暂离开，不必报告，请小心翼翼进出教室，并尽量避免影响他人。

- ▶ 如需请假，务必请提前以你认为最便利的方式通知教师或助教。

如果已经被点名认定为迟到或缺席，无需提供任何事后解释或证明材料。即使提供这些材料，任课教师将不会接受。

- ▶ 鼓励上课期间主动提问。
尽量与教学内容相关。

课堂要求

- ▶ 禁止抄袭作业。

一经核实，将不予区分主动抄袭或被他人抄袭，本门课程**一律记0分**。如果抄袭作业已经被核实，可以选择主动退选本门课程。

- ▶ 上课期间，除饮用水之外，请不要携带饮料、咖啡或者餐食进入教室。

- ▶ 布置的作业，请在下周上课当天晚上9点之前递交作业。

递交作业无需使用纸质作业本，将电子版作业（代码文件、解答文档等）发送至

助教邮箱：w.qy@ruc.edu.cn

邮件标题：第x周作业+姓名+学号

(比如，第3周作业+罗翔宇+20180121)

课堂要求

- ▶ 平时成绩将包括出勤情况、平时作业等各种情况。最终成绩由平时成绩和期末成绩两部门组成，平时成绩占60%，期末成绩占40%。

我爱学习，
学习使我快乐



期末考试

- ▶ 期末成绩以考卷形式得出，闭卷考试



1. 并行计算和Linux简介

罗翔宇

中国人民大学统计与大数据研究院

此课件内容主要基于Blaise Barney的网络资料
[introduction to parallel computing](#) 以及张林
波等编著的《并行计算导论》（清华大学出版
社，2006.06）第一章

什么是并行计算?

► 串行计算(serial computing):

- 指令(instructions)序列地在单个处理器(processor)上进行执行
- 任一时间, 仅有一个指令正在执行

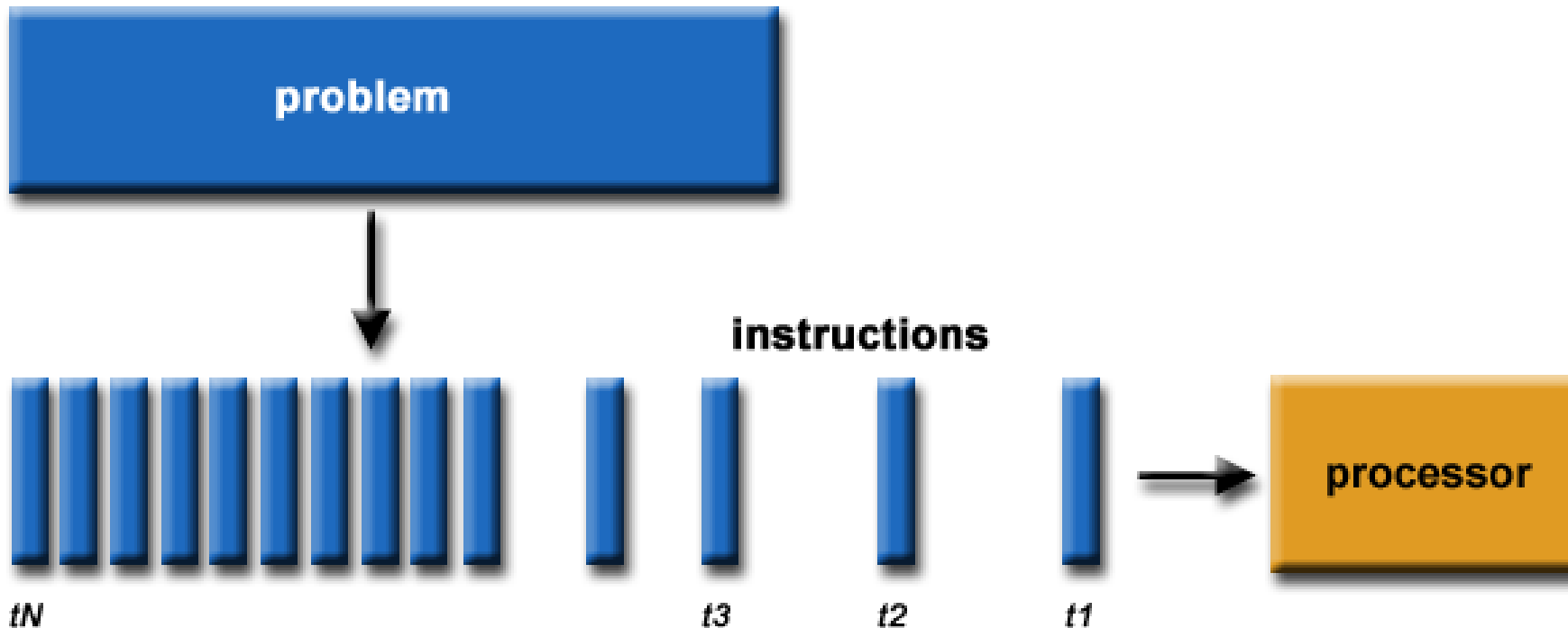


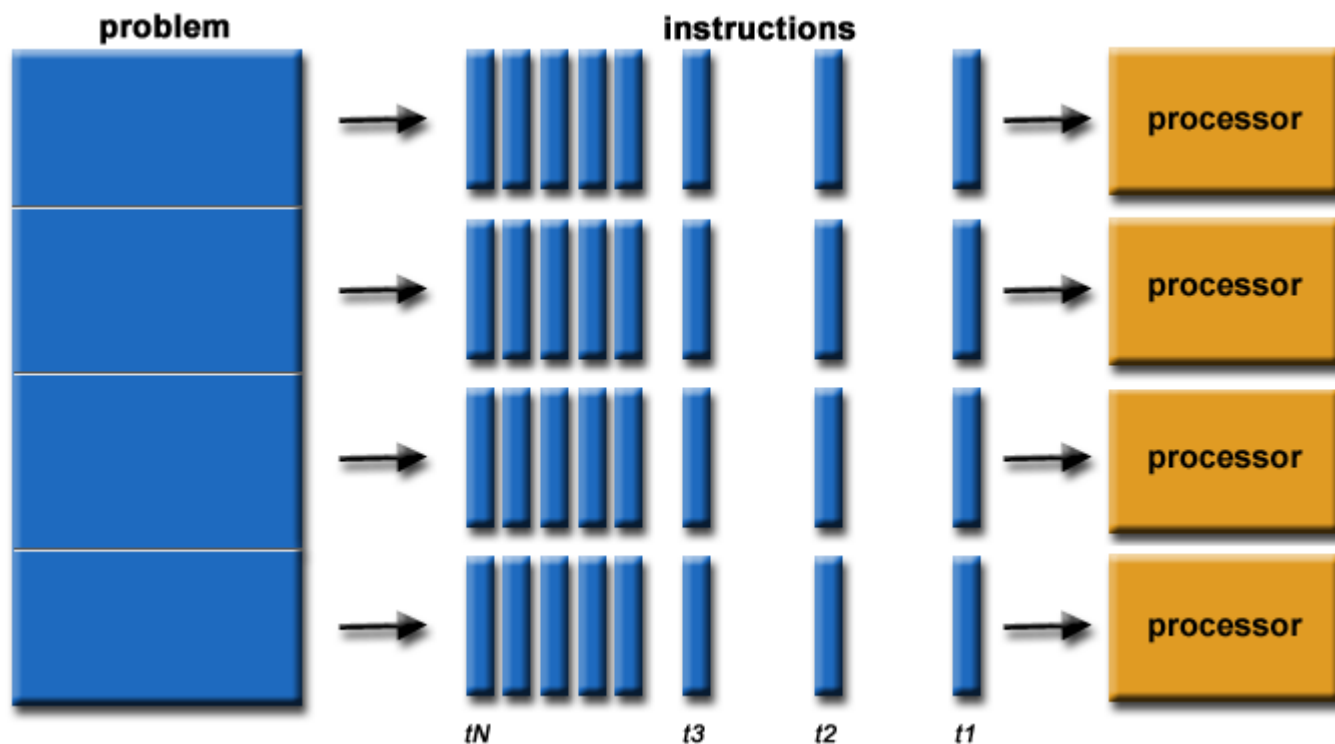
Figure credit: https://computing.llnl.gov/tutorials/parallel_comp/

什么是并行计算?

► 并行计算(parallel computing):

► 一个问题可以分解为若干子问题; 多个子问题在多个处理器上并行解决。

► 任一时间, 有多个指令正在同时执行。



什么是并行计算？

- ▶ 定义：把一个计算问题分解为多个子问题，并将其分配到并行机的多个处理器上，处理器之间相互协调、并行地完成子问题，以实现加快问题求解速度或增大问题应用规模。

并行计算的基本条件

▶ 三个基本条件：

▶ 1. 计算问题能够分解为若干可以并行解决的子问题

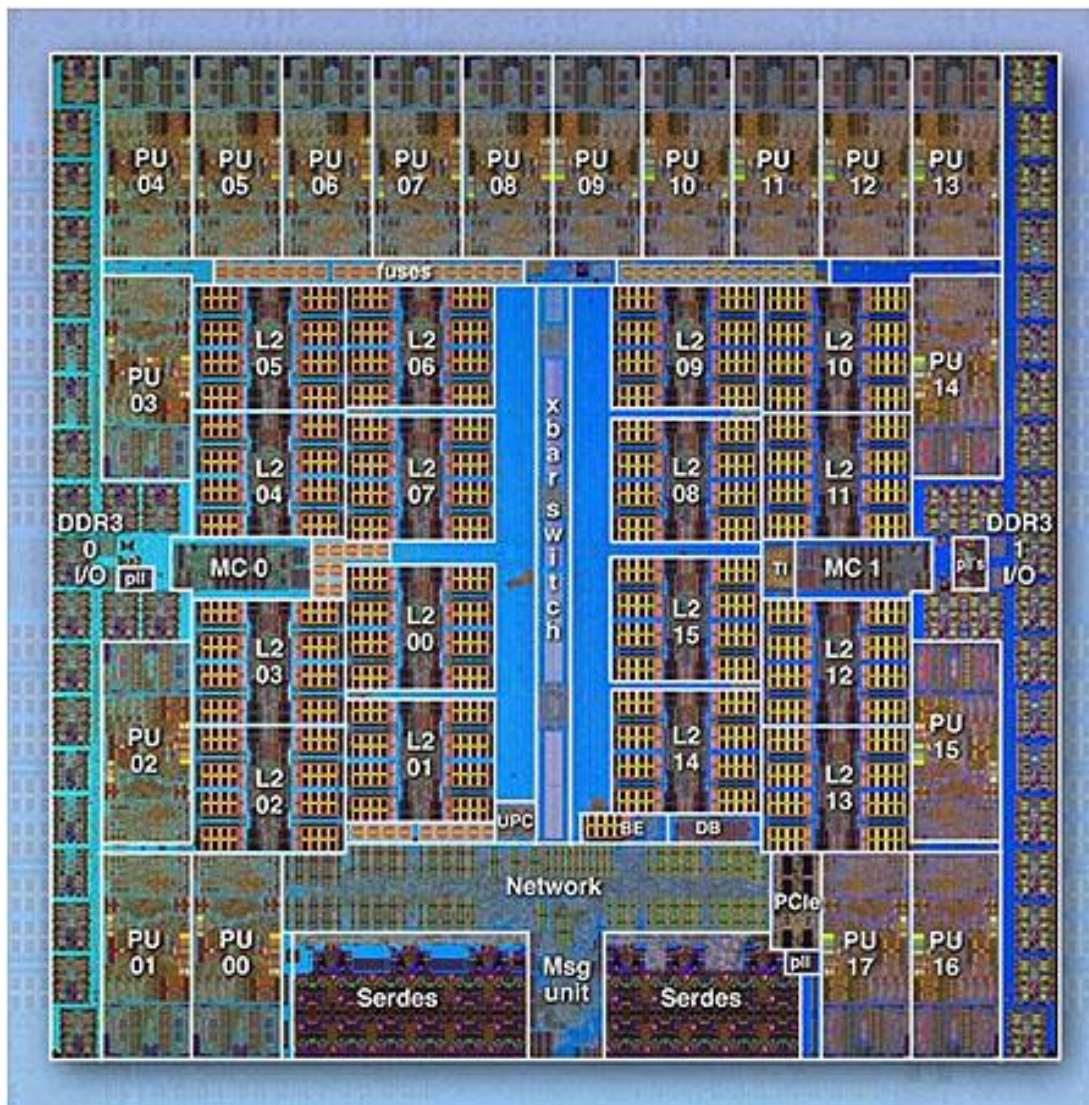
▶ 2. 并行机：

▶ 一个具有多个处理器的计算机

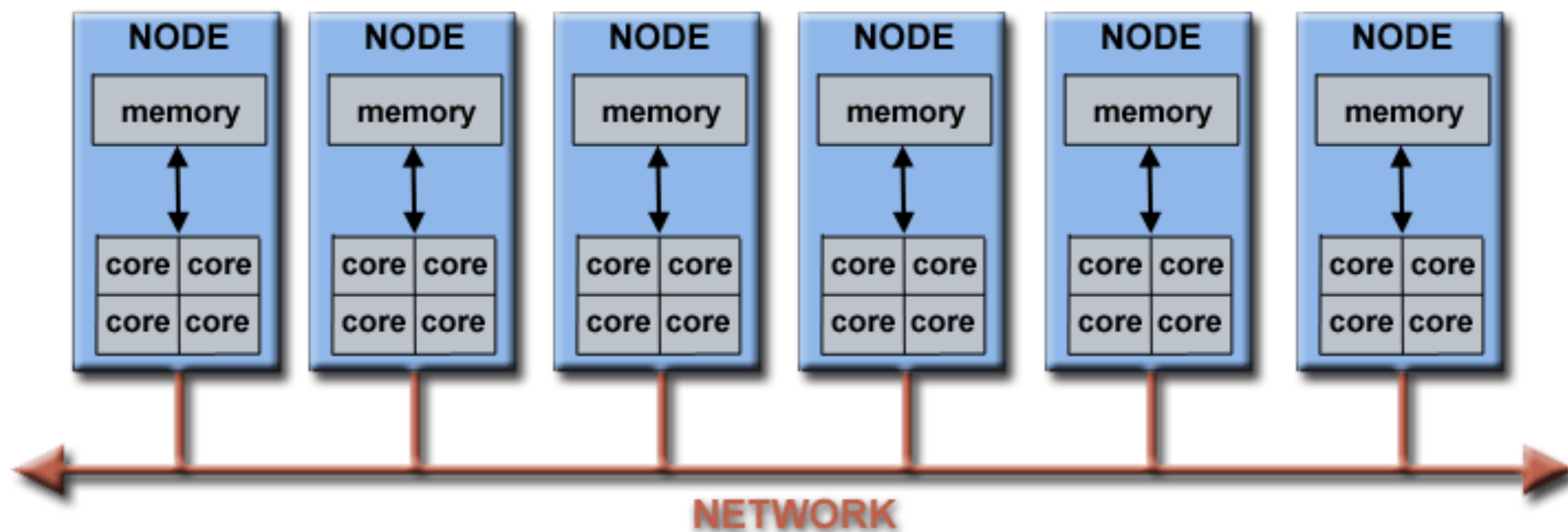
▶ 由网络连接的多个计算机

▶ 3. 并行编程：编程实现并行算法

一个具有18个处理单元的芯片



由网络连接的6个计算机，每个计算机具有4个处理单元



中国人民大学高性能计算集群具有类似结构

并行计算的主要目的

- ▶ 加快求解速度：在1台处理器上串行需14天。若利用100台处理器加速了50倍，时间缩减到6.72小时。
- ▶ 增大问题应用规模：1台2GB内存处理器计算10万个网格，100台同样的处理器能计算一千万个网格。

为什么要使用并行计算?

► 这个世界是大规模并行的



Galaxy Formation



Planetary Movements



Climate Change



Rush Hour Traffic



Plate Tectonics



Weather



Auto Assembly



Jet Construction



Drive-thru Lunch

Figure credit: https://computing.llnl.gov/tutorials/parallel_comp/

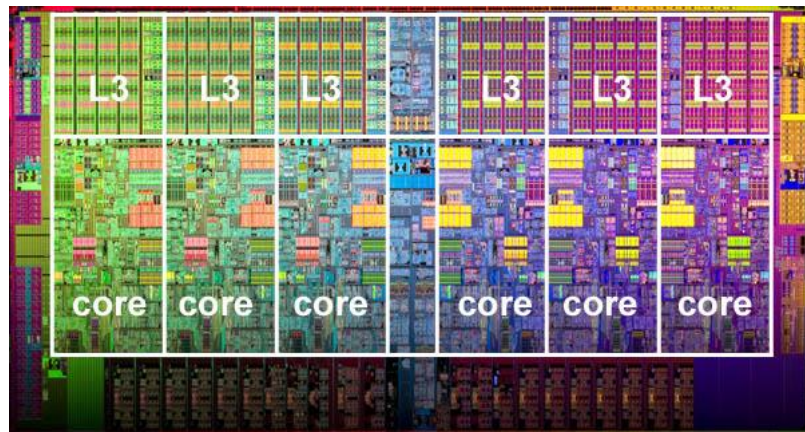
为什么要使用并行计算?

► 节约时间、金钱



为什么要使用并行计算?

- ▶ 解决更大、更复杂的问题
 - ▶ 很多问题不能在单个电脑上处理，比如“Grand challenge problems”需要Peta级别的FLOPS(每秒浮点运算次数)和Bytes
 - ▶ 例如，网络搜索引擎，处理每秒百万级别的交易
- ▶ 更好地利用潜在的并行硬件
 - ▶ 现代的手提或者台式电脑都具有并行硬件设备，从而串行程序会浪费潜在的计算资源



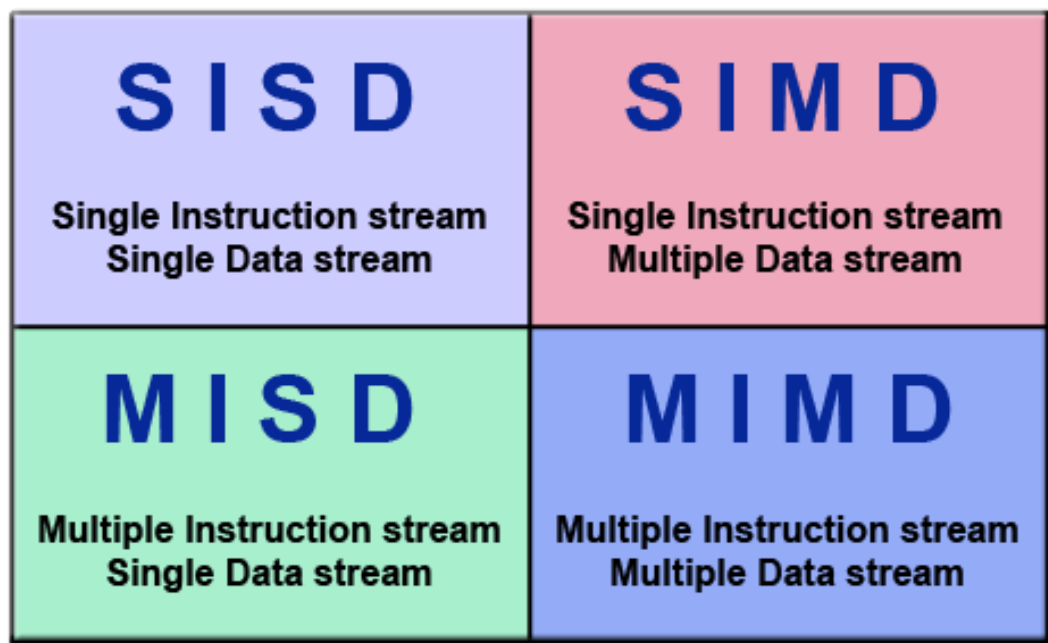
Intel Xeon processor with 6 cores and 6 L3 cache units

并行计算的主要研究内容

- ▶ 并行机的高性能特征抽取
- ▶ 并行算法设计与分析
- ▶ 并行实现技术
- ▶ 并行应用

Flynn分类法

- ▶ 根据指令流(instruction stream)和数据流(data stream)来对计算机分类

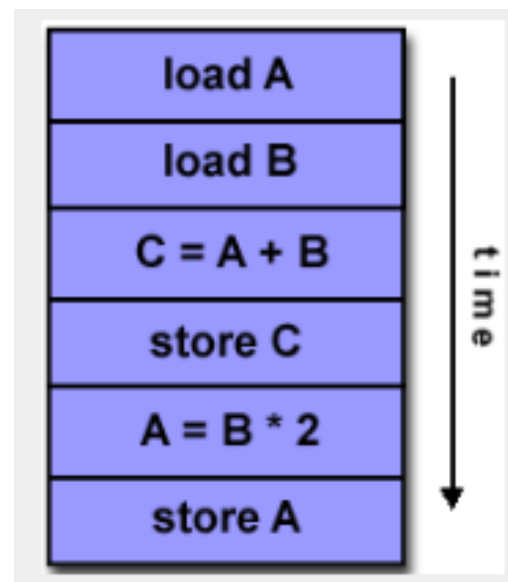
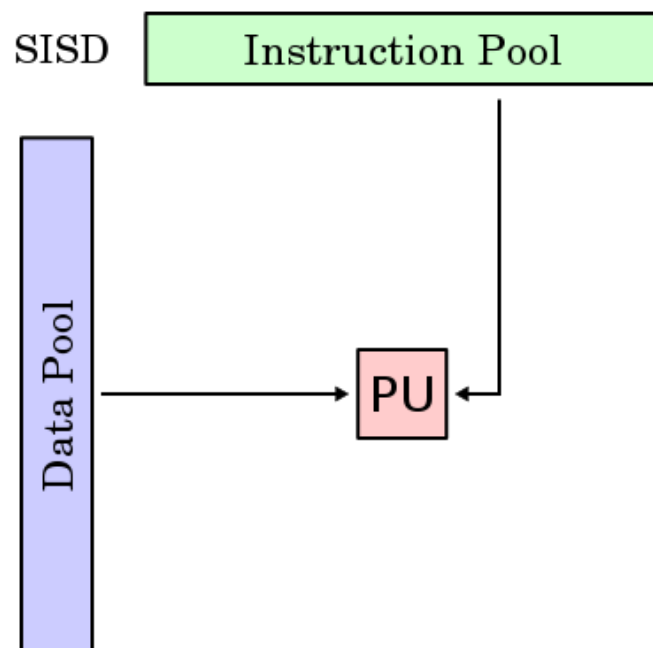


Flynn分类法

► 单指令流单数据流机器(SISD)

► 传统的串行计算机，不支持任何并行操作

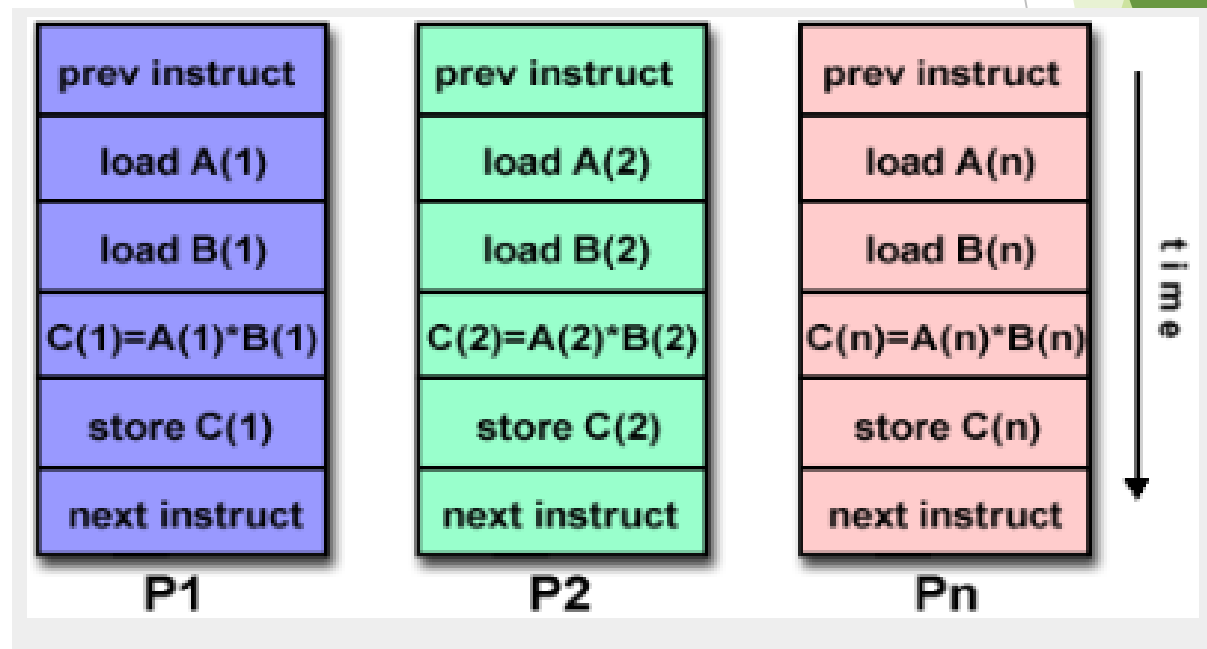
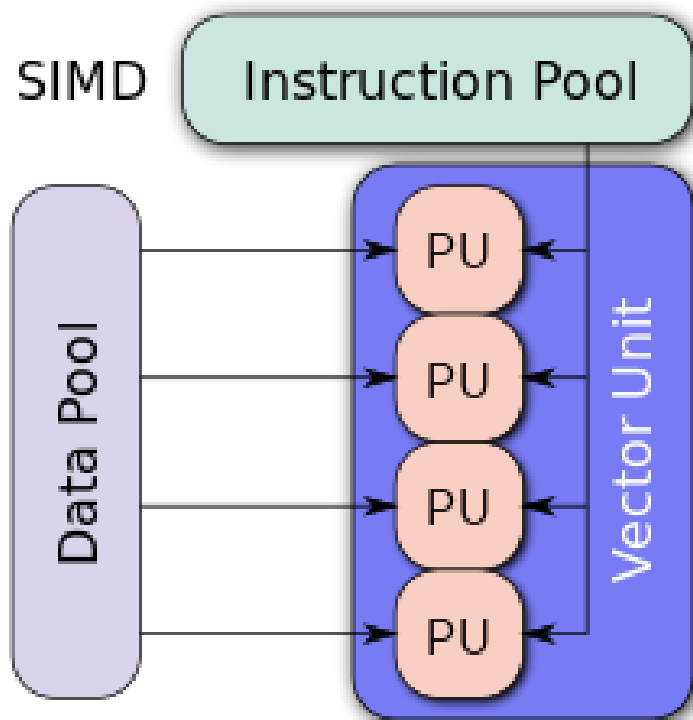
► 比如IBM 360



Flynn分类法

► 单指令流多数据流机器(SIMD)

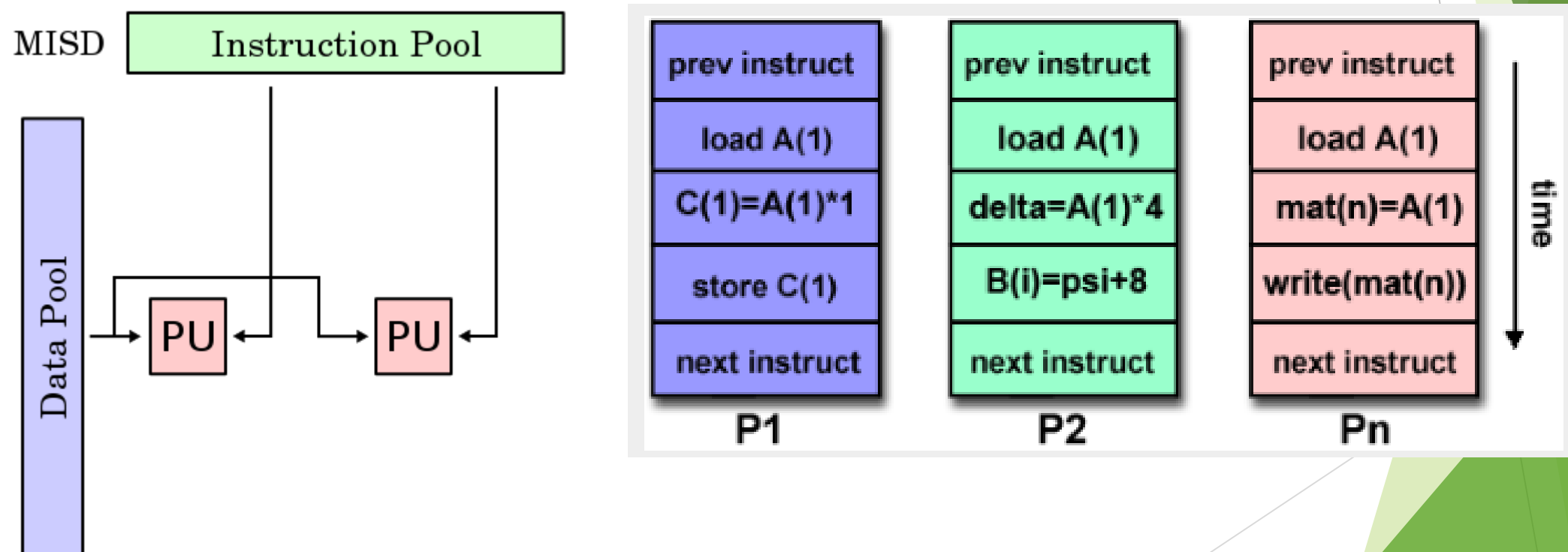
- 一个指令流处理多个数据流(一种并行机)
- 在信号处理、图像处理等领域起着很大作用



Flynn分类法

► 多指令流单数据流机器(MISD)

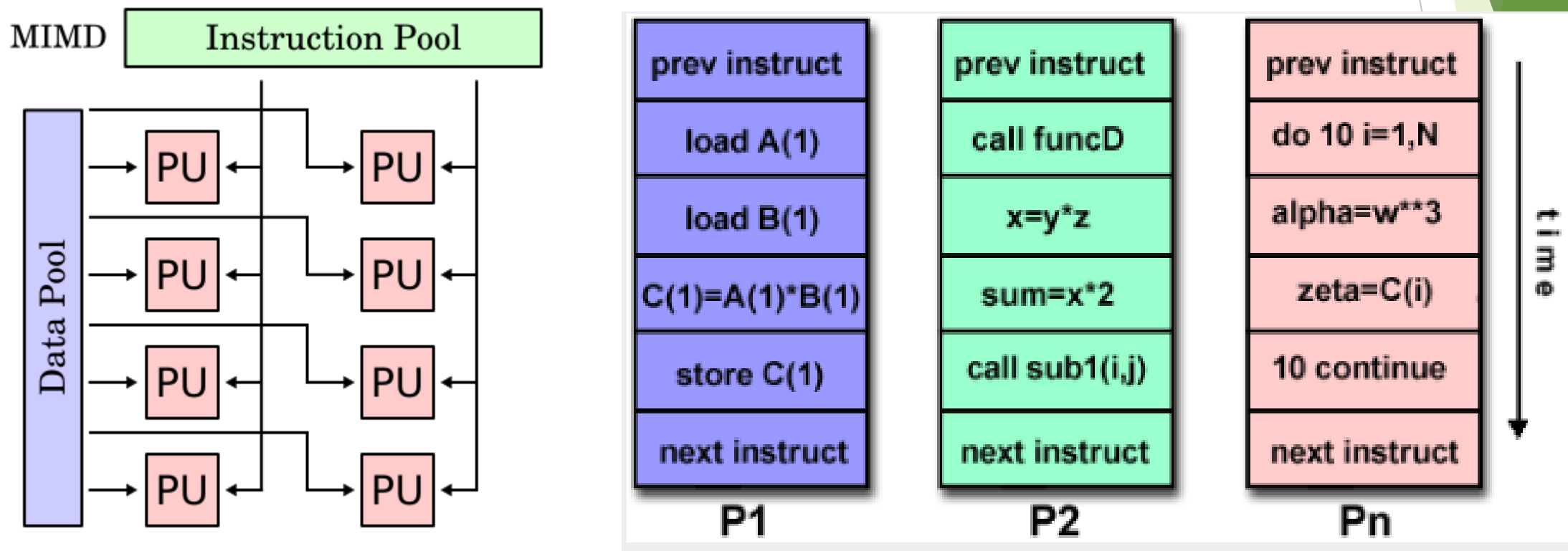
- 多个指令流仅处理一个数据流(一种并行机)
- 没有投入到实际应用中, 仅出现在理论模型中



Flynn分类法

► 多指令流多数据流机器(MIMD)

- 可以执行多个指令流(多指令流)
- 每个指令流可以作用在不同的数据流(多数据流)
- 最常用的并行计算机、大多数现代的超级计算机

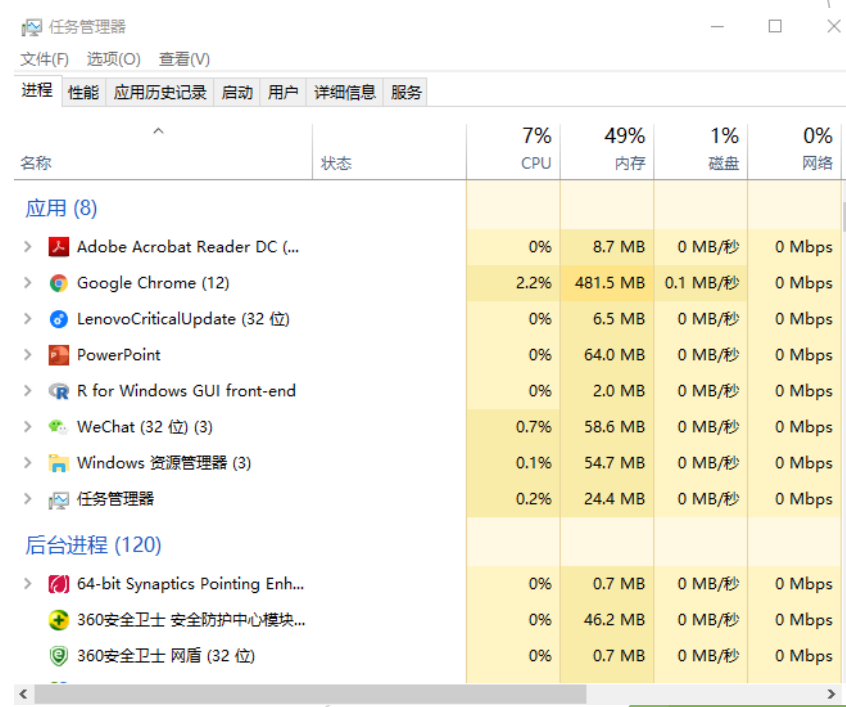


并行计算中的常用术语

- ▶ 超级计算 (Supercomputing)
- ▶ 高性能计算 (High Performance Computing)
- ▶ 计算节点(Node), 单独的计算机, 通常包含多个CPU处理器。节点之间通过网络连接形成计算集群。
- ▶ 中央处理器(CPU), 核(cores): 在集群中可能的情况是, 一个节点具有多个CPU, 一个CPU具有多个核。
- ▶ 共享内存(Shared Memory): 从硬件上来说, 所有处理器能够直接接触共同的物理内存。从编程上来说, 所有并行任务能处理和接触相同的逻辑内存地址。
- ▶ 分布式内存(Distributed Memory): 从硬件上来说, 物理内存通过网络连接。从编程模型上说, 一个任务只能看到本地的内存, 为了接触其他任务的内存, 只能通过通信(communication)。
- ▶ 高度并行(Embarrassingly Parallel): 一个问题能非常容易化为若干相互独立的子问题, 其并行程序几乎不需要通信。
- ▶ 可扩展性(Scalability): 指并行系统展示出随着资源增加并行速度成比例增加的能力。

并行计算中的常用术语 进程

- ▶ 进程(process)是资源分配的基本单位，也是操作系统调度的基本单位
 - ▶ 可表示四元组(P,C,D,S), P是程序代码、C是进程的控制状态、D是进程的数据、S是进程的执行状态
 - ▶ 程序运行时系统就会创建一个进程并为其分配资源
 - ▶ 进程之间可以相互通信。可以不用了解实现的具体细节，只需要掌握适用的应用程序接口(API)，比如MPI

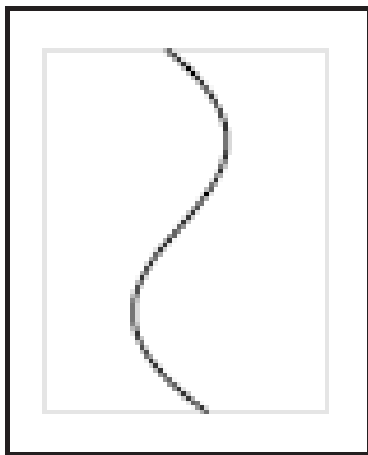


The screenshot shows the Windows Task Manager window with the 'Processes' tab selected. It displays a list of running applications and background processes, along with their resource usage (CPU, Memory, Disk, Network).

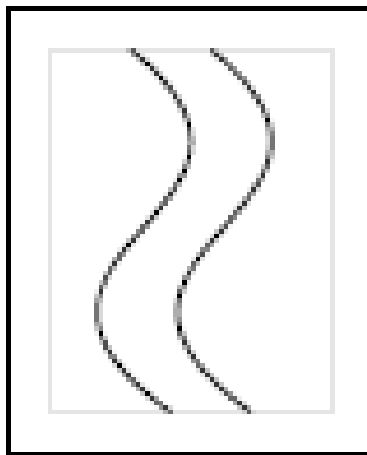
名称	状态	7% CPU	49% 内存	1% 磁盘	0% 网络
应用 (8)					
> Adobe Acrobat Reader DC (...)		0%	8.7 MB	0 MB/秒	0 Mbps
> Google Chrome (12)		2.2%	481.5 MB	0.1 MB/秒	0 Mbps
> LenovoCriticalUpdate (32 位)		0%	6.5 MB	0 MB/秒	0 Mbps
> PowerPoint		0%	64.0 MB	0 MB/秒	0 Mbps
> R for Windows GUI front-end		0%	2.0 MB	0 MB/秒	0 Mbps
> WeChat (32 位) (3)		0.7%	58.6 MB	0 MB/秒	0 Mbps
> Windows 资源管理器 (3)		0.1%	54.7 MB	0 MB/秒	0 Mbps
> 任务管理器		0.2%	24.4 MB	0 MB/秒	0 Mbps
后台进程 (120)					
> 64-bit Synaptics Pointing Enh...		0%	0.7 MB	0 MB/秒	0 Mbps
> 360安全卫士 安全防护中心模块...		0%	46.2 MB	0 MB/秒	0 Mbps
> 360安全卫士 网盾 (32 位)		0%	0.7 MB	0 MB/秒	0 Mbps

并行计算中的常用术语 线程

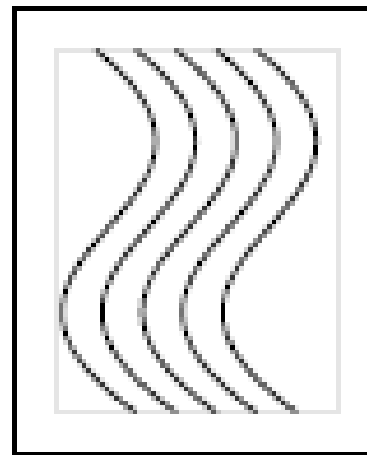
- ▶ 线程(thread)是程序执行时的最小单位，是进程的一个执行流
- ▶ 为了在共享存储环境下有效地开发应用程序的细粒度并行度，将一个进程分解两个部分，其中一部分由其资源特征构成，仍称之为进程；另一部分由其执行特征构成，称之为线程
- ▶ 多个线程将共享该进程的所有资源特征. API: OpenMP, POSIX Threads



单进程单线程执行



单进程双线程执行

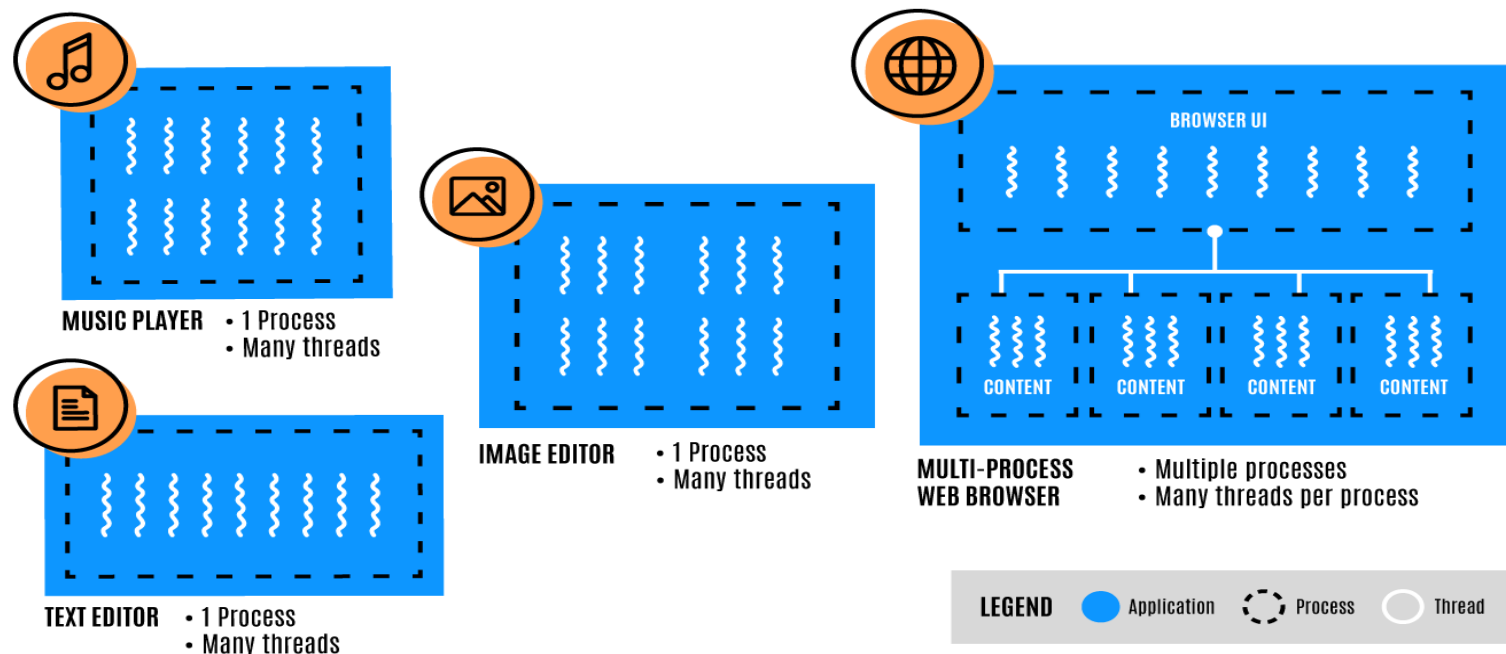


单进程多线程执行

并行计算中的常用术语 进程与线程

- 1、一个进程可以有多个线程，但至少有一个线程；而一个线程只能在一个进程的地址空间内活动
- 2、资源分配给进程，同一个进程的所有线程共享该进程所有资源
- 3、线程在执行过程中需要协作同步，不同进程的线程间要利用消息通信的办法实现同步

APPS, PROCESSES, AND THREADS



(原文链接: https://blog.csdn.net/qq_40340448/article/details/81836065)

Amdahl's Law

- ▶ Amdahl's Law表明最大的程序速度提升由代码中可并行部分的比例(P)决定。
- ▶ $speedup = \frac{1}{1-P}$
- ▶ 若代码不能并行, $P=0$ 意味着速度提升为1(即相较于完全串行无速度提升)
- ▶ 若全部代码均可并行, $P=1$ 意味着理论上的速度提升为无穷大
- ▶ 若有一半的代码可并行, $P=0.5$ 意味着速度提升为2

Amdahl's Law

► 假设代码中不能并行的比例为 S ($S=1-P$), 用 N 个处理器, Amdahl's Law 为

► $speedup = \frac{1}{\frac{P}{N} + S}$

N	speedup			
	P = .50	P = .90	P = .95	P = .99
10	1.82	5.26	6.89	9.17
100	1.98	9.17	16.80	50.25
1,000	1.99	9.91	19.62	90.99
10,000	1.99	9.91	19.96	99.02
100,000	1.99	9.99	19.99	99.90

可以看出, 在考虑到处理器成本的情况下, 并不是处理器越多越好, 而依赖于代码并行比例

"Famous" quote: *You can spend a lifetime getting 95% of your code to be parallel, and never achieve better than 20x speedup no matter how many processors you throw at it!*

Amdahl's Law

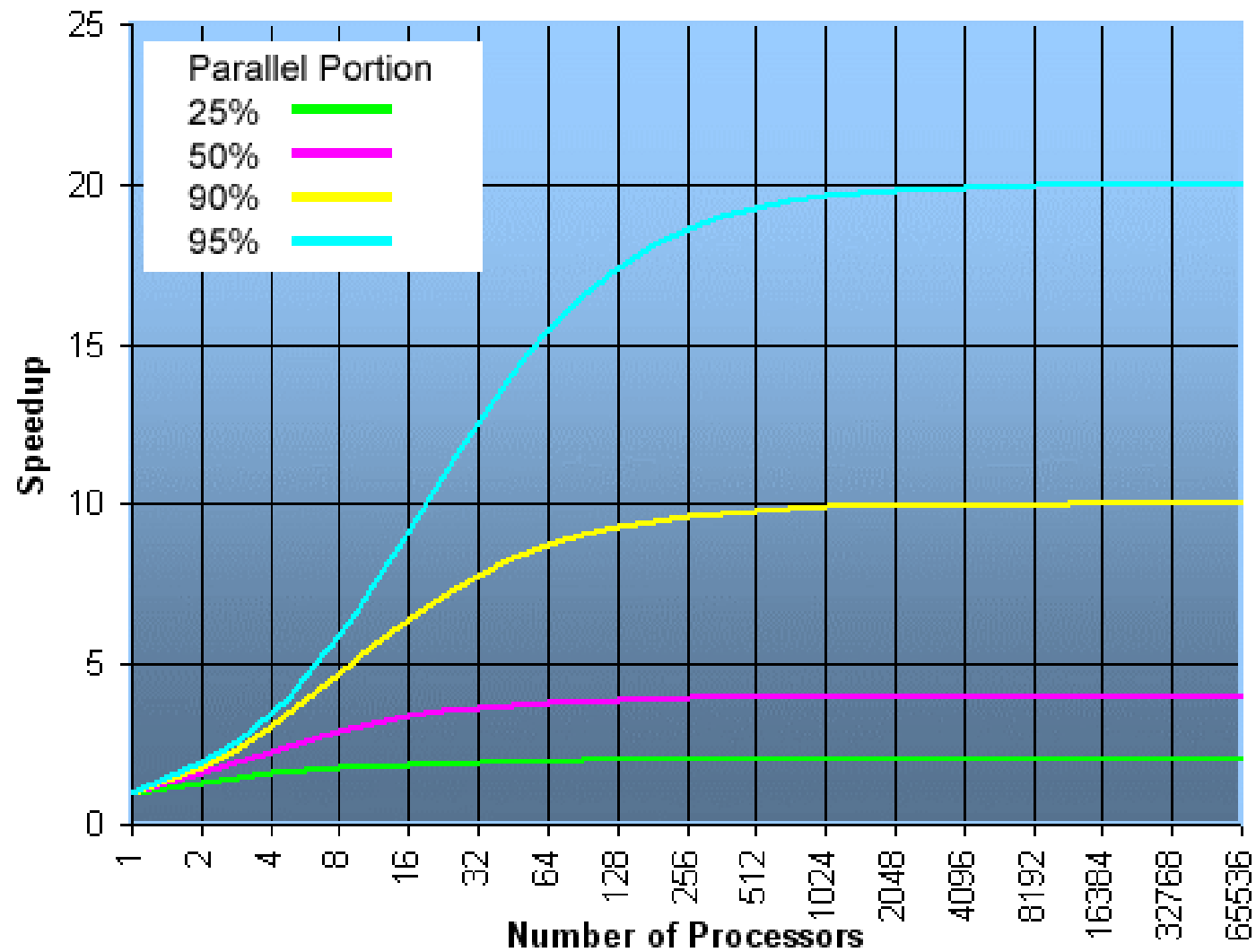


Figure credit: https://computing.llnl.gov/tutorials/parallel_comp/

并行程序的开发周期

- ▶ 设计 design
- ▶ 编程 coding
- ▶ 排错 debugging
- ▶ 调试 tuning
- ▶ 维护 maintenance

Linux操作系统

- ▶ 当前流行的计算机集群的节点上安装Linux操作系统
- ▶ 一个超级用户(管理员)和若干普通用户
- ▶ 子目录
 - ▶ /usr 主要的系统文件和软件
 - ▶ /home 普通用户的家目录
 - ▶ /root 超级用户的家目录
 - ▶ ...

```
xiangyu@xiangyu-VirtualBox:/$ ls
bin      dev      initrd.img      lib64      mnt      root      snap      sys      var
boot     etc      initrd.img.old  lost+found opt       run       srv       tmp      vmlinuz
cdrom    home     lib              media      proc     sbin      swapfile  usr
```

```
xiangyu@xiangyu-VirtualBox:/$
```

Shell

- ▶ Shell是一个应用程序，提供用户和Linux系统的交互。最常用的叫做Bash，全称是Bourne-Again Shell
- ▶ 对于管理员可以添加/删除用户
- ▶ 增加用户 `useradd [-d 家目录] [-g 组名] [-s shell] 用户名`
- ▶ 删除用户 `userdel [-r] 用户名`
- ▶ 修改口令 `passwd [用户名]`
- ▶ 修改属性 `usermod`

Linux基本命令和概念

- ▶ pwd显示当前目录
- ▶ cd改变当前目录
 - ▶ cd- 回到上一次所在目录 cd 回到用户的家目录。
 - ▶ .. 指上一级目录 .指当前目录

```
xiangyu@xiangyu-VirtualBox:~$ pwd
/home/xiangyu
xiangyu@xiangyu-VirtualBox:~$ cd ..
xiangyu@xiangyu-VirtualBox:/home$ pwd
/home
```

Linux基本命令和概念

- ls列出当前目录下的文件和子目录名

```
xiangyu@xiangyu-VirtualBox:~$ ls
breeze.linalg._  Downloads          parallel_computing_files  Templates
Desktop          examples.desktop  Pictures                  Videos
Documents        Music              Public
```

- mkdir建立一个文件夹

```
xiangyu@xiangyu-VirtualBox:~$ mkdir test
xiangyu@xiangyu-VirtualBox:~$ ls
breeze.linalg._  Downloads          parallel_computing_files  Templates
Desktop          examples.desktop  Pictures                  test
Documents        Music              Public                   Videos
```

Linux基本命令和概念

- ▶ man查询函数的使用方法，比如man mkdir
- ▶ 按q返回到命令行

File Edit View Search Terminal Help

MKDIR(1)

User Commands

MKDIR(1)

NAME

mkdir - make directories

SYNOPSIS

mkdir [OPTION]... DIRECTORY...

DESCRIPTION

Create the DIRECTORY(ies), if they do not already exist.

Mandatory arguments to long options are mandatory for short options too.

-m, --mode=MODE

set file mode (as in chmod), not a=rwx - umask

Linux基本命令和概念

- ▶ rm 删除文件或目录
 - ▶ rm newfile, rm -r newdir

```
xiangyu@xiangyu-VirtualBox:~$ rm -r test
xiangyu@xiangyu-VirtualBox:~$ ls
breeze.linalg._  Downloads          parallel_computing_files  Templates
Desktop          examples.desktop  Pictures                  Videos
Documents        Music             Public
```

- touch 建立一个空白文件

```
xiangyu@xiangyu-VirtualBox:~$ touch tmp.txt
xiangyu@xiangyu-VirtualBox:~$ ls
breeze.linalg._  Downloads          parallel_computing_files  Templates
Desktop          examples.desktop  Pictures                 tmp.txt
Documents        Music             Public                  Videos
xiangyu@xiangyu-VirtualBox:~$
```

Linux基本命令和概念

► cp用于复制一个文件

```
xiangyu@xiangyu-VirtualBox:~$ cp tmp.txt tmp2.txt
xiangyu@xiangyu-VirtualBox:~$ ls
breeze.linalg._  Downloads          parallel_computing_files  Templates  Videos
Desktop          examples.desktop  Pictures                  tmp2.txt
Documents        Music              Public                    tmp.txt
```

► mv用于移动一个文件

```
xiangyu@xiangyu-VirtualBox:~$ ls
breeze.linalg._  Downloads          parallel_computing_files  Templates  Videos
Desktop          examples.desktop  Pictures                  tmp2.txt
Documents        Music              Public                    tmp.txt
xiangyu@xiangyu-VirtualBox:~$ mv tmp2.txt Desktop/tmp3.txt
xiangyu@xiangyu-VirtualBox:~$ ls
breeze.linalg._  Downloads          parallel_computing_files  Templates
Desktop          examples.desktop  Pictures                  tmp.txt
Documents        Music              Public                    Videos
xiangyu@xiangyu-VirtualBox:~$ cd Desktop
xiangyu@xiangyu-VirtualBox:~/Desktop$ ls
tmp3.txt
xiangyu@xiangyu-VirtualBox:~/Desktop$
```


Linux基本命令和概念

- ▶ whoami用于查看“我是谁”
- ▶ who用于看看当前有哪些用户在线

```
xiangyu@xiangyu-VirtualBox:~/Desktop$ whoami
xiangyu
xiangyu@xiangyu-VirtualBox:~/Desktop$ who
xiangyu :0          2020-02-18 17:16 (:0)
xiangyu@xiangyu-VirtualBox:~/Desktop$
```

- ▶ Shell脚本：将多个shell命令放在同一个文件进行执行

程序开发环境：vi编辑器

- ▶ 新建c文件: vi hello_world.c
- ▶ 按i进入insert模式，输入相关C代码
- ▶ 按Esc键，回到命令模式，再输入:w即将内容存入到了文件(一定要注意冒号)

程序开发环境：vi编辑器

The screenshot shows a Linux desktop environment with a terminal window open. The terminal title bar reads "xiangyu@xiangyu-VirtualBox: ~/parallel_computing_files". The terminal contains the following C code:

```
File Edit View Search Terminal Help
#include<stdio.h>

int main(){
    printf("Add oil China! Add oil Wuhan!\n");
    return 0;
}
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --
```

程序的编译和运行

- ▶ 最常用的C编译器是GNU C编译器，在Linux中的命令为gcc
 - ▶ gcc -o hello_world hello_world.c
 - ▶ 完成了编译并链接得到了可执行文件hello_world
 - ▶ 运行可执行文件./hello_world
 - ▶ 注意这里需要加上./表示在当前目录下执行名字为hello_world的文件，否则shell会在PATH指定目录下去寻找hello_world的文件。若恰有相同名字的文件，则会造成不必要的麻烦
- ▶ GNU附带的编译器包括C, C++, Fortran, JAVA, Object C等
- ▶ 注意：非法内存使用
 - ▶ 内存没有分配就用
 - ▶ 内存分配了没有释放
 - ▶ 分配内存和释放内存的命令不配套

程序的编译和运行

```
xiangyu@xiangyu-VirtualBox:~/parallel_computing_files$ gcc -o hello_world hello_world.c
xiangyu@xiangyu-VirtualBox:~/parallel_computing_files$ ./hello_world
Add oil China! Add oil Wuhan!
```

软件开发

- ▶ 三类软件：商业软件、学术软件、公开软件
- ▶ 软件特点：规模比单个程序大得多，附带图形文件、清晰的帮助文件，能在不同机器硬件和软件环境之间移植，提供技术支持
- ▶ 文件组织
 - ▶ 能够分开的东西就尽量不要放在一起
- ▶ 实用工具make
 - ▶ make的目标是当一个文件被修改后，make会自动处理依赖于这个文件的其他文件，使得所有的衍生文件都处于最新状态
- ▶ 文档开发和维护
 - ▶ 程序中的注释面向开发，文档面向用户
- ▶ 版本管理和协同工作
 - ▶ 版本管理：一个可以记录开发过程中所有文件的所有变化的工具
 - ▶ 比如 GitHub

注意事项

- ▶ Mac OS, Ubuntu操作系统均可通过进入terminal（终端）通过shell与系统进行交互
- ▶ 对于Windows系统，为使用shell
 - ▶ 可以安装虚拟机并在上面安装Ubuntu操作系统
 - ▶ 或用网页版Webminal <https://webminal.org/> 进行Linux命令练习（注册后登陆，有点慢）。