一、什么是进程

1.1 进程的概念

进程的概念起源于操作系统，是操作系统最核心的概念。进程是对正在运行的程序的一个抽象。

进程就是正在进行的一个过程或者说是一个正在进行的任务。

1.2 并发与并行

在用户看来都是"同时"运行的

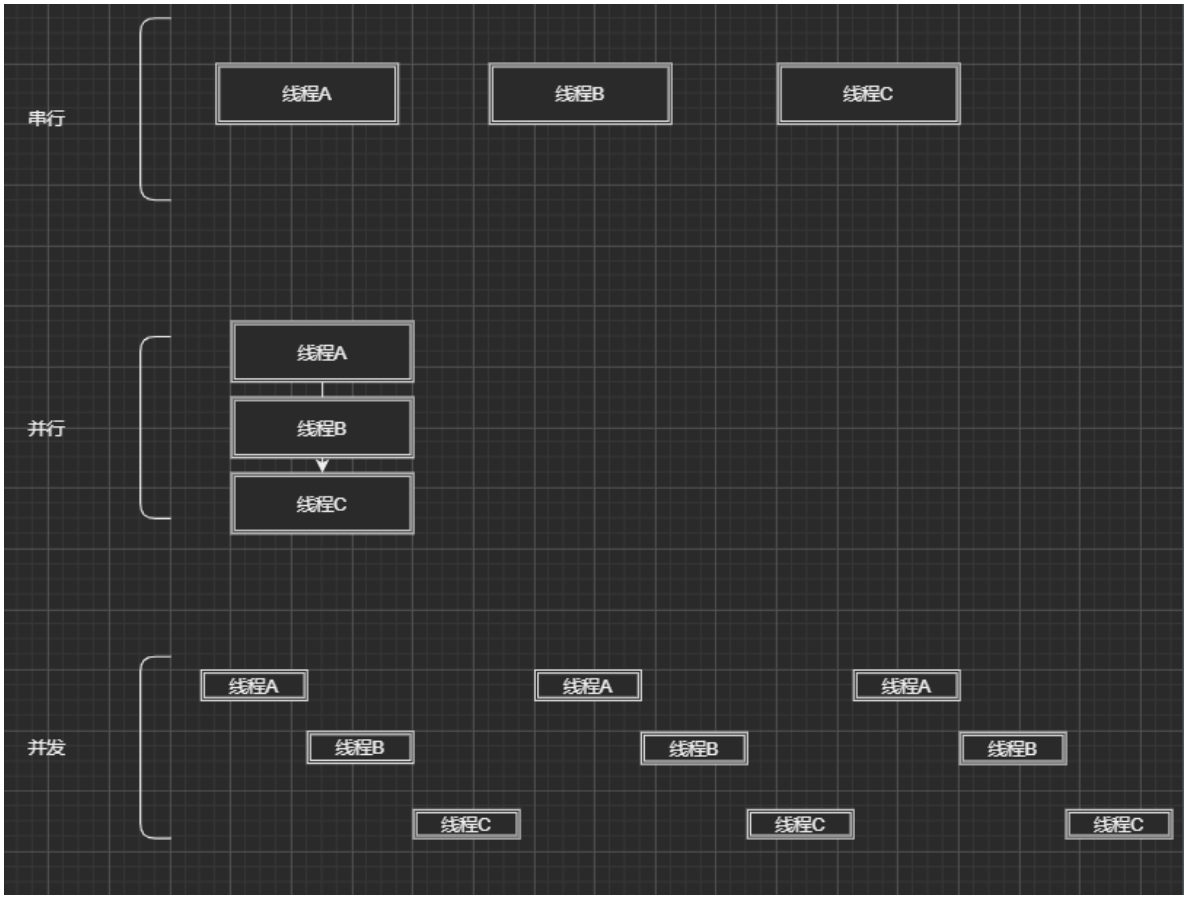
真正干活的是CPU，CPU同一时刻只能执行一个任务

并行：同时运行，只有具备多个CPU才能实现并行

并发：就是假并行，看起来是同时运行；单个CPU+多道技术就可以实现并发（并行也属于并发）

串行： 按顺序执行

多道技术：内存中同时存入多道（多个）程序，CPU从一个进程快速切换到另一个，使每个进程各自运行几十或几百毫秒，这样，虽然在某一个瞬间，一个cpu只能执行一个任务，但是在1s内，cpu却可以运行多个进程，这就给人产生了并行的错觉，也就是伪并发，以此来区分多处理器操作系统的真正硬件并行（多个CPU共享一个物理内存）



1.3 同步和异步

同步：就是指一个进程在执行某个请求的时候，若该请求需要一段时间才能返回信息，那么这个进程将会一直等待下去，知道收到返回信息才继续执行。

异步：指的是进程不需要一直等待下去，而非是继续执行下面的操作，不管其他进程的状态。当有消息返回时系统会通知进程进行处理，这样可以提高执行效率。

1.4 进程的创建

新进程的创建都是由一个已经存在的进程执行了一个用于创建进程的系统调用而创建的。

1. 在UNIX中该系统调用fork，fork会创建一个与父进程一模一样的副本，二者有相同的存储映射，同样的环境字符串和同样的打开文本（在shell解析器进程中，执行一个命令就会创建一个子进程）
2. 在windows中该系统调用的是CreateProcess，createProcess既处理进程的创建，也负责吧正确的程序装入新进程。

相同点：进程创建之后，父进程和子进程有各自不同的地址空间（多道技术要求物理层面实现进程之间内存的隔离），任何一个进程在其地址空间中的修改都不会影响到另一个进程。

不同点：在UNIX中，子进程的初始地址空间是父进程的一个副本（子进程和父进程是可以有只读的共享内存区的），但是对于windows系统来说，从一开始父进程和子进程的地址空间就是不同的。

1.5 进程的终止

```
1. 正常退出（自愿，用户点击X号，程序执行完毕发起系统调用正常退出，在linux中的exit，在windows中的ExitProcess）
```

2. 出错退出（自愿，python 执行a.py文件，但是a.py不存在）
3. 严重错误（非自愿，执行非法指令，如引用不存在的内容，I/O操作）
4. 被其他进程杀死（kill all -9 uwsgi）

1.6 进程的状态

tail -f access.log | grep '404' 查看在access.log 日志文件中，出现404的位置

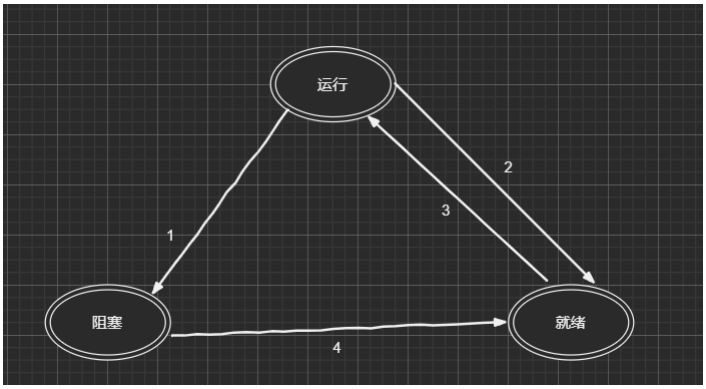
执行tail，开启一个子进程，执行程序grep，开启另一个子进程，两个子进程之间基于管道'|' 通讯，将tail的结果作为grep的输入

进程grep在等待输入时的状态称为阻塞，此时grep命令都无法运行

两种导致进程在逻辑上不能运行的情况：

1. 进程挂起是自身原因，遇到I/O阻塞，便要让出CPU让其他进程去执行，这样保证CPU一直在工作
2. 与进程无关，是操作系统层面，可能会因为一个进程占用时间过多，或者优先级等原因，而调用其他的进程去使用CPU。

所有，一个进程由三种状态组成：阻塞，就绪，运行



1. 进程为等待输入而阻塞
2. 高度程序选择另一个进程
3. 高度程序选择这个进程
4. 出现有效输入

二、线程

三、Python并发编程—多进程

四、paramiko模块