进程

1. **进程描述**
   1. 进程**定义**

一个具有一定独立功能的程序在一个数据集合上的一次动态执行过程。

* 1. 进程的特点和组成

**组成**：程序的代码、程序处理的数据等。

**特点**：动态性、并发性、独立性、制约性。

程序与进程的**区别**：

一静一动；

程序是有序代码的集合，进程是程序的执行，且有用户态/核心态；

一个暂时一个永久；

进程包括程序、数据和PCB。

**Ex：**

人照着食谱做菜---食谱=程序，人=CPU，原料=数据，做菜=进程。

* 1. 进程**控制结构**

**PCB：**OS管理控制进程运行所用的信息集合（进程存在唯一标志）。

含有的信息：进程标识信息、处理机状态信息保存区、进程控制信息。

组织方式：链表

1. **进程状态**
   1. 进程的生命周期管理

进程创建、运行、等待、唤醒、结束。

When创建：系统初始化、用户请求、创建子进程

When 运行：OS选择一个就绪进程并使其占用处理机

When 等待：请求服务、启动操作无法马上完成，数据未到

When 唤醒：阻塞进程等待的资源或事件到达，将PCB插入就绪队列

（注：进程只能被别的进程或OS唤醒）

When 结束：Exit and abort.

* 1. 进程状态变化模型

**基本状态**：运行、就绪、等待。

* 1. 进程挂起模型

**挂起**状态：（进程从内存到外存）

**阻塞**挂起：进程在外存并等待某事件的出现；

**就绪**挂起：进程在外存，但只要进入内存就可以运行。

**状态队列：**

不同的状态对应着不同的队列，队列由OS维护，进程的PCB根据它的状态进入队列。

1. **线程**
   1. **什么**是线程

进程构成的资源平台上的一条执行流程。

线程=进程-共享资源

**优点**：

进程中可以存在多个线程，它们可并发执行、可以共享地址空间和文件等资源；

**缺点**：

一个线程崩溃，则其所属进程的所有线程崩溃。

进程与线程的**比较**：

资源分配单位VS CPU调度单位；

拥有一个完整的资源平台 VS 只独享必不可少的资源；

同样有三种基本状态及状态间的转移关系；

线程能减少并发执行的时空开销。

* 1. 线程的**实现**

由TCB管理。

**用户线程与内核线程**的对应关系：

多对一、一对一、多对多。

**用户线程**：

在用户空间实现，不依赖内核；

由一组用户级的线程库函数来完成线程的管理。

**缺点**：

一个线程阻塞，则整个进程等待；

一个线程开始运行后，除非它主动交出CPU的使用权，否则它所在的进程当中的其他线程将无法运行；（**因为线程没有中断，不能和线程一样上下文切换**）

多线程的进程，其线程执行较慢。

**内核线程**：

在OS内核中实现，由OS内核对其管理。

轻量级进程：

OS支持的用户线程。一个进程可以有多个轻量级进程，每个轻量级进程由一个单独的内核线程来支持。

1. **进程同步**
   1. **背景**

进程同步的**优点：**共享资源、加速、模块化**。**

* 1. **临界区**的互斥实现

1. 硬件**中断**：系统会停下来，可能导致其他进程处于饥饿状态；

无法限制响应中断所需的时间。

1. 基于**软件**：比较复杂，能解决，课本。
2. 更高级的**抽象**：将临界区操作抽象为锁。主要有两种操作，一种是Test-and-set指令，一种是Exchange。
   1. **信用量**

**概念：**

一个整形（sem），两个原子操作：

P()：sem减1，如果sem<0，等待，否则继续；（能够阻塞）

V()：sem加1，如果sem<=0，唤醒一个等待的P。（不能阻塞）

* 1. **管程**

**概念：**

一个锁**：**指定临界区；

0或者多个条件变量**：**等待/通知信号量用于管理并发访问共享数据。

**一般方法：**

收集在对象/模块中的相关共享数据；

定义方法来访问共享数据。

**经典问题：**

哲学家进餐、读者-写者问题。

1. **死锁**
   1. **系统模型**

资源分配图：

如果图中没有循环—》没有死锁；

如果图中有循环—》只有一个实例：死锁；

有几个实例：可能死锁。

* 1. **死锁特征**

死锁可能出现如果四个条件同时成立（**必要条件**）：

互斥、持有并等待、无抢占、循环等待。

* 1. **处理方法**

死锁**预防**：（破坏四个条件）：

互斥—占用非共享资源；

占用并等待—进程请求资源时不持有其他任何资源（资源利用率低，可能发生饥饿）；

无抢占—请求资源时需释放已有资源；

循环等待—对资源进行排序，申请资源需要按序。

死锁**避免：**

银行家算法。

死锁**检测**：

类似银行家算法，时间开销较大。

死锁**恢复**：

终止所有的死锁进程；

在一个时间内终止一个进程直到死锁消除。

1. **进程通信**
   1. **IPC**

2个操作：send()、receive()

进程P和Q通信需要：

1.在它们之间建立通信链路；

2.通过send和receive交换信息。

**直接**通信：

进程必须正确的命名对方；

通信链路：一条对应一对通信进程。

**间接**通信：

定向从消息队列接受消息；

通信链路：进程共享一个共同的消息队列才建立链路；进程可以共享多个通信链路。

消息**传递：**

阻塞：同步

非阻塞：异步。

* 1. **信号**

软件中断通知信息处理。

接受信号时：Catch、Ignore和Mask。

不足：不能传输要交换的任何数据。

* 1. **管道**

进程接受输入的来源及输出的地方。

* 1. **消息队列**

按FIFO的方式处理信息。

* 1. **共享内存（最快的方法）**

每个进程都有私有地址空间。在每个地址空间内，明确地设置了共享内存段。

优点：快速、方便地共享数据；

缺点：必须同步数据访问。