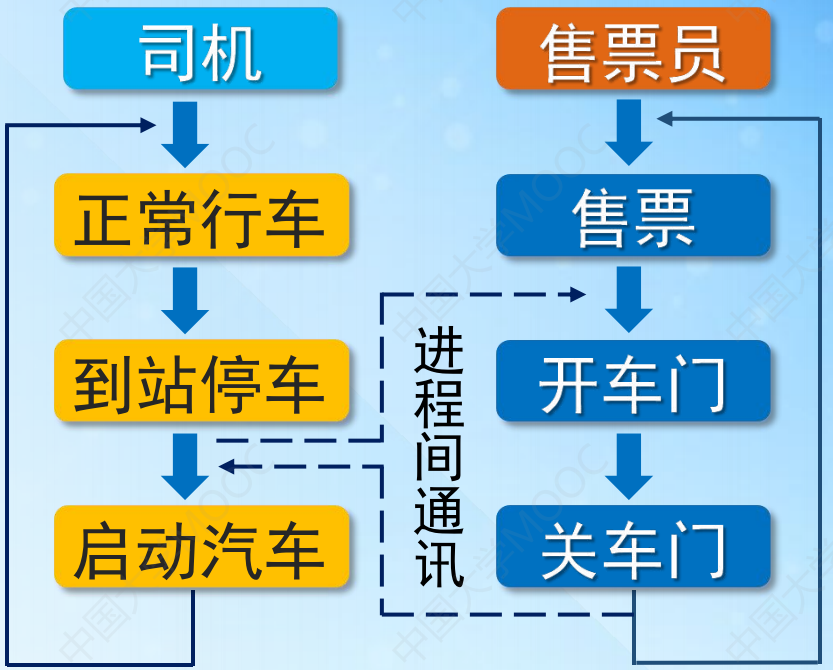
操作系统及Linux内核

西安邮电大学

进程的同步与互斥关系

同步(Synchronization)

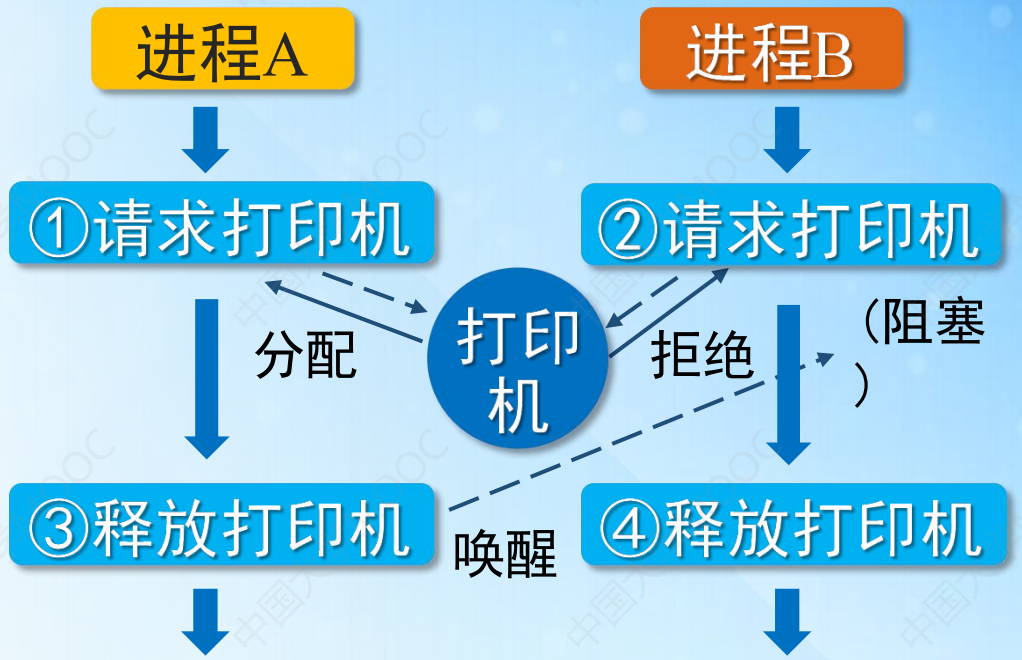
多个并发进程(线程)协同完成一项任务时，由于数据交换需要而在进程(线程)执行次序上的约束关系，叫同步关系，亦称功能合作关系。



进程的同步与互斥关系

互斥(Mutual Exclusion)

并发进程（线程）间为竞争同一个资源而间接发生的相互制约关系，叫互斥关系，亦称为资源竞争关系。



临界资源与临界区

临界资源(Critical Resource)

系统中一次只允许一个进程(线程)访问的资源称为临界资源。一旦分配给进程，不能强制剥夺。

物理资源：

如打印机、扫描仪等I/O设备。

逻辑资源：

如共享变量、共享文件、控制信号等。

临界资源与临界区

临界区(Critical Section)

并发执行的进程中，访问（读取或修改）临界资源必须互斥执行的程序段叫临界区。临界区分散在并发执行的进程中。

P_1: ……
C_1 a = a + 1;
……

P_2: ……
C_2 print (a);
……

P_3: ……
C_3
```
if (a < 0)
    a= a +1;
else
    a= a-1;
```
……

- 变量a是临界资源；
- C1、C2、C3是临界区；
- 对a互斥访问

进程同步机制

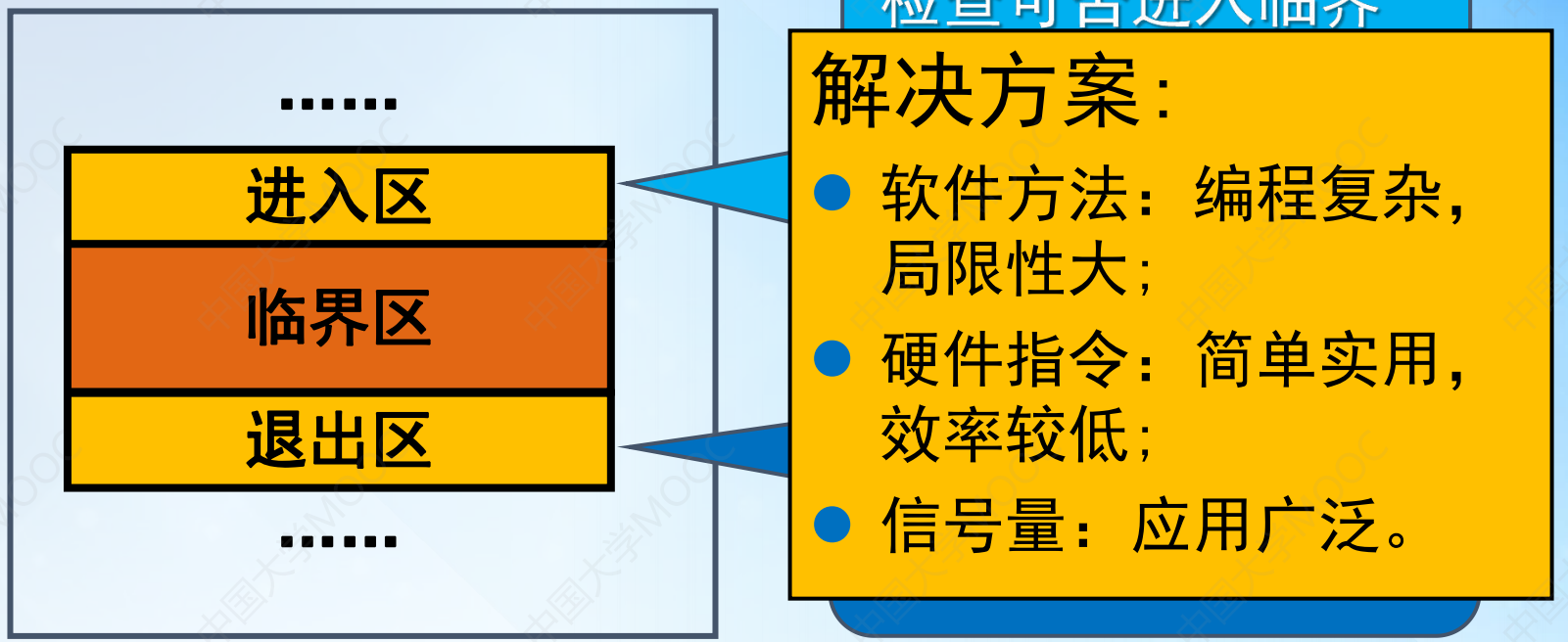
同步机制(Syn. Mechanism)

多个相关进程（线程）在执行次序上的协调称为进程同步。用于保证多个进程在执行次序上的协调关系的机制称为进程同步机制。

- 空闲让进
- 忙则等待
- 有限等待
- 让权等待

临界区互斥访问

互斥访问策略



临界区互斥访问

软件方法

```
bool CBusy = false;  //标志
int main(void){
    pthread_t tid[10];
    for(int i=0; i<10; i++)
        pthread_create(&tid[i],
            NULL, fun, NULL);
    ……
    return  0;
}
```

```
void *fun(void *p){
    ……
    while(CBusy==true) ;
    CBusy = true;
    Critical Section;
    CBusy = false;
    ……
    return NULL;
}
```

存在问题：可能两个线程都进入临界区，违反忙则等待原则。

临界区互斥访问

硬件指令—“测试并设置原语”

```
//加锁原语
bool Lock(bool *target) {
    bool rtn = *target;
    *target = true;
    return rtn;
}
```

原语(atomic action) 是**若干条机器指令构成的完成某种特定功能的一段程序。**

原语具有不可分割性，执行过程中不允许被中断。

```
bool CBusy = false; //标志
int main(void){
    ……
}
```

存在问题：违反让权等待原则，效率低下。

小结

进程（线程）同步机制

类型
- 同步关系：相互协作
- 互斥关系：资源竞争

核心
- 临界资源：物理、逻辑资源
- 临界区：互斥访问

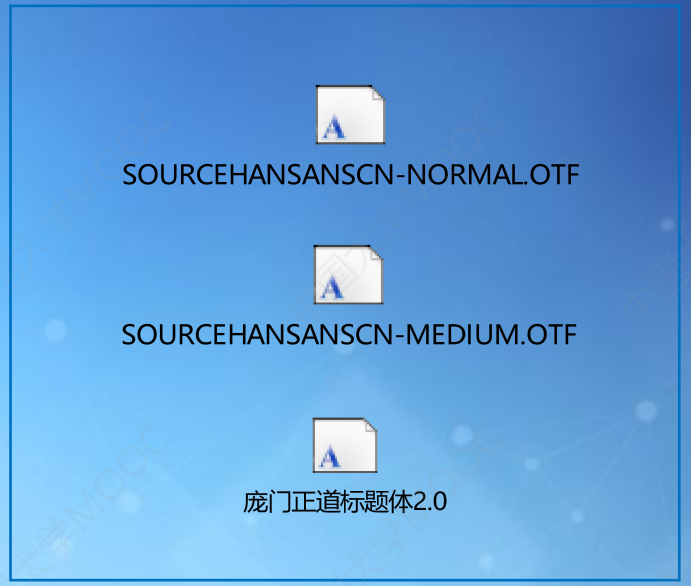
原则
- 空闲让进
- 忙则等待
- 让权等待
- 有限等待

方案
- 软件方法
- 硬件指令
- 信号量（下一节）

字体：中文：思源黑体≥24
英文：新罗马 ≥24

配色

层级  文件管理



特殊字体双击安装