基本概念、主要方式

进程间通信IPC

为什么需要通信机制？

- 信号量及管程的不足
- 不适用多处理器情况

适用于：分布式系统、基于共享内存的多处理机系统、单处理机系统可以解决进程间的同步问题、通信问题

进程通信机制

- 消息传递

  send & receive原语

所以呢，进程间通信机制是非常重要的啊一项内容

基本通信方式

- 消息传递
- 共享内存
- 管道
- 套接字
- 远程过程调用

那么，像套接字、 远程过程调用都适合于用于网络或者是分布式系统

消息传递

消息缓冲区结构：
消息头（消息类型、接收进程ID、发送进程ID、消息长度、控制信息）
消息体（消息内容）

发送进程S

send(destination, message)

message:
text, size

①陷入内核

②复制消息

一组消息缓冲区

③消息入队

④复制消息

接收进程R

receive(sour
mess

message:
text, siz

接收进程的PCB
中消息队列指针

操作系统空间

用P、V操作实现SEND原语

```
Send（destination, message）
{
  根据destination找接收进程;
  如果未找到，出错返回;

  申请空缓冲区P(buf-empty);
    P(mutex1);
      取空缓冲区;
    V(mutex1);

把消息从message处复制到空缓
冲区;
```

```
    P(mutex2);
      把消息缓冲区挂到接收进程
      的消息队列;
    V(mutex2);

    V(buf-full);
}
```

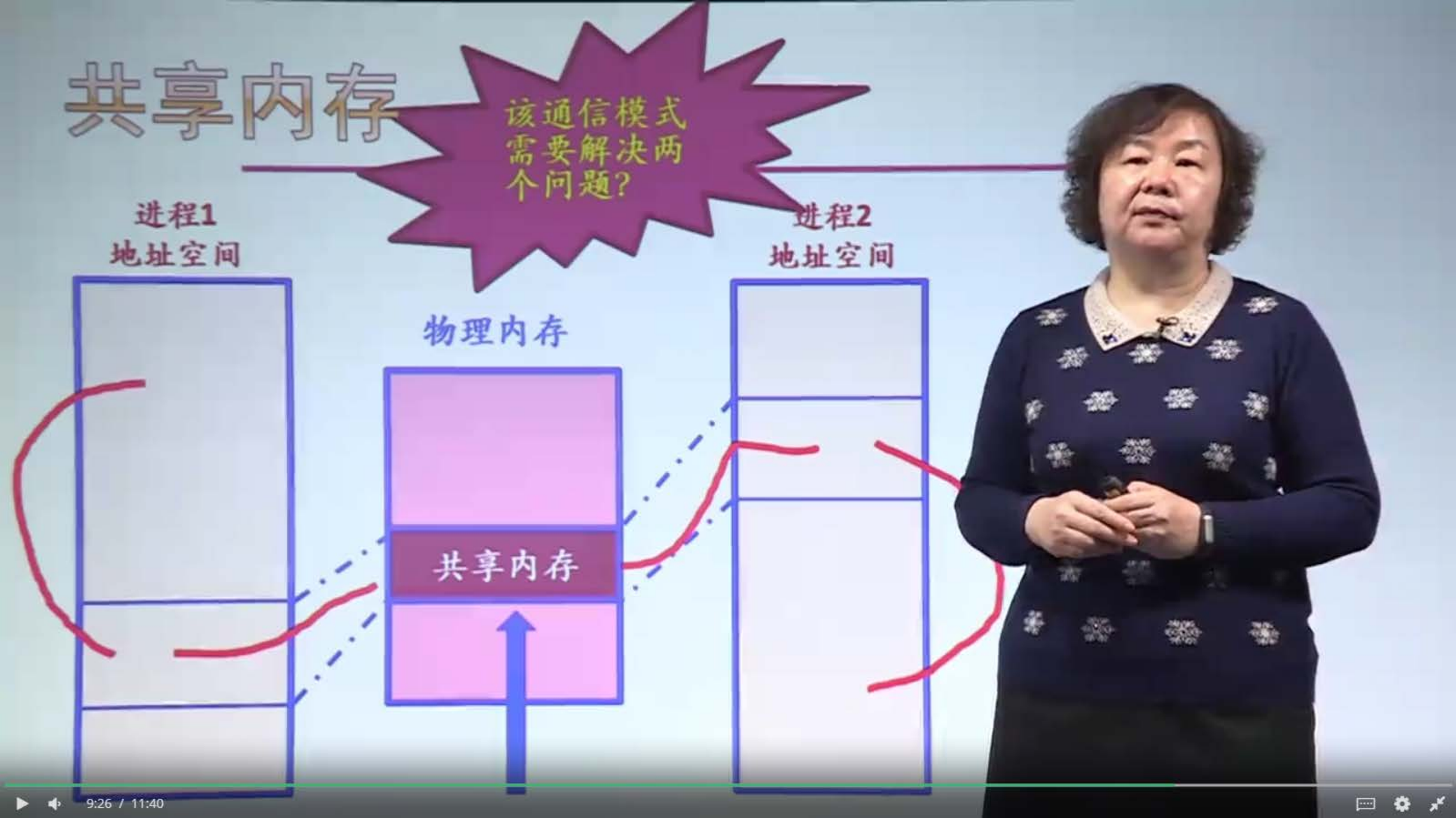
信号量：

buf-empty初值为N

buf-full初值为0

mutex1初值为1

mutex2初值为1


共享内存
该通信模式需要解决两个问题?
进程1
地址空间
物理内存
进程2
地址空间
共享内存

管道通信方式 PIPE

- 利用一个缓冲传输介质——内存或文件连接两个相互通信的进程

- 字符流方式写入读出
- 先进先出顺序
- 管道通信机制必须提供的协调能力
  - 互斥、同步、判断对方进程是否存在

这就是管道通信方式 那么我们这里头只介绍三种啊进程间通信方式

0:00 下面我们介绍进程间通信机制。 有了信号量 有了管程之后，为什么进程之间还需要新的通信机制？ 这主要原因是，信号量和管程只能传递 很简单的信息，不能传递大量的信息 比如说，我要把一个大的数组传送给另外一个 进程，那么信号量和管程在这一方面是做不到的。 另外呢 管程不适合于用于多处理器的情况 因此呢，我们需要在传递大量信息的时候呢，引入新的 通信机制，那么这个通信机制呢我们称之为，进程间通信机制 其中一个非常典型的形式呢，就是消息传递 消息传递呢，实际上就是由一个 send 和 receive 这样的，提供这样的原语操作 那么，当一个进程要把息，消息发送给另外一个进程的时候呢，就去调用 send 当另外一个进程说，我要想接收消息的时候呢，就去调用 receive 操作 那么，进程通信机制呢，它适合的场景很多 比如说，基于网络的分布式系统 比如说，基于共享内存的多处理机系统。 当然啦，它也适合于用于单处理机系统 所以，它适合的场景非常多。 那么它的用途也很广，可以解决进程之间的同步问题、 互斥问题 也可以解决啊进程的通信问题，传递大量信息的问题，所以它的用途也很广 所以呢，进程间通信机制是非常重要的啊一项内容 通常呢，我们有这样几个典型的通信方式 一个是消息传递，一个是共享内存 一个是管道，还有套接字和远程过程调用 那么，像套接字、 远程过程调用都适合于用于网络或者是分布式系统 那我们首先介绍一下消息传递 消息传递，我们先看一下这个场景 有一个进程它要发送消息给另外一个进程 所以，这个进程我们称之为发送进程，这个叫接收进程 发送进程把消息准备好啦，我们可以看到消息准备好啦，消息的内容和消息的大小 但是由于发送进程它所在自己的地址空间 它不能够到接收进程地址空间进行任何操作 因此，这个事情，发送消息这个事情，必须由操作系统帮助它来完成 那么操作系统呢，在它的区域里头设置了一个一组啊，消息缓冲区 那么，每一个消息都由一个 buffer 来 接收这个消息啊，存放这个消息。 而 buffer 的数据 结构呢，是这样一个结构。 它包括了消息的 头儿，啊这样一些个数据结构，描述了消息的类型，接收进程的 ID 还有发送进程的 ID ，以及消息的长度和一些控制信息 然后就是消息的内容啊，把所有消息内容都可以 放在一个 buffer 里头，这是一个等长的啊消息 那么，当发送进程 想要把消息发送到 这个这个另外一个进程的时候，那么它要调用这个发送原语 send 而发送原语呢实际上呢，它是一个陷入了操作系统，由操作系统完成的一个发送的过程 所以呢，陷入内核，然后操作系统呢接收过来 那么，操作系统主要做的工作就是把发送进程准备好的消息 拷贝到某一个缓冲区里头 找一个空的缓冲区，把消息内容拷贝过去 然后，把这个消息挂接到接收进

程的 消息队列的末尾。 大家还记不记得在 这个讲进程控制块的时候 那么 PCB 里头，有很多的啊数据项，那么其中 包括了一个数据项就是，消息队列的指针 那么，通过这个指针，所有发给这个进程的消息，都挂接到这个队列里头 做完这些事情之后，那么 发送进程其实就是完成了发送工作，可以接着啊做下面的事情了 那么什么时候接收进程上 CPU 了 执行到了 receive 这样一个接收原语呢 那么，可能它被调度上 CPU 之后，它来执行这个原语 当这个时候，执行这个原语，也是陷入操作系统内核 那么由操作系统来完成相应的工作，因此 操作系统来帮助它把 这个消息复制到了，接收进程的地址空间 那么，这样就完成了一个发送的过程，一个接收的过程 所以，发送进程只是把消息准备好 调用 send 操作，那么操作系统 做相应的复制消息的内容，挂接的内容 那么，接收消息呢，接收进程呢接收消息的时候呢，也是把这个 请求提交给操作系统，操作系统完成把消息复制到接收进程空间的工作 所以呢，操作系统要提供这样一个通信机制，来完成进程之间的信息传送 这里头我们简单给出来，用 P、 V 操作来实现 SEND 原语 那么，应该首先先要得到一个 空的缓冲区，所以通过一个 P 操作，看看有没有空缓冲区了 因为，我们假设空缓冲区的个数是有限的 然后，就去摘取空缓冲区 下面是，把消息复制到缓冲区里头 接着，把消息啊挂接到接收进程的消息队列 然后，通知接收进程说，有消息了。 那么如果 接收进程刚才来取消息，没有取到。 那么进入等待，这个时候呢 这样一个操作就会把这个接收进程唤醒 重新让它就绪。 那么如果，接收进程还没有 执行到 receive 的话呢，那么这就是通知是吧，啊增加了一个新的消息 那么接收进程执行 receive 的时候，就有消息可接收了 关于 receive 的原语怎么实现？ 大家回去可以按照这样一个啊模式来试着 实现一下。 下面我们来介绍第二种的啊，进程间通信的方式叫做，共享内存 我们来看一下啊，这是啊两个进程 分别有自己的地址空间，那么它们两个进程之间如果需要 送一些数据给对方，或者是读一些数据，那怎么做呢？ 那么可以通过共享内存的方式。 那么 用这种方式来实现这个消息传递，啊消息的这个通信，那么需要解决两个问题 第一个问题呢，需要在物理内存里头建立 一个大家能够共享的一块内存空间 并且，通过相应的映射，把这个物理内存空间 映射到了两个进程的 相应的地址空间里头。 我们可以看到在进程 1 的这一块空间 和进程 2 的这一块空间，都映射到了同一块物理内存 通过这样一个映射，那么这两个进程其实都是在这块物理内存上做相应的操作 所以这是第一个问题。 第二个问题呢，

其实就是我们前面介绍的 读者、 写者问题，因为，这块区域不能同时去写 但是呢，可以同时去读。 所以呢，我们可以利用 控制读者、 写者问题的这个方法呢来解决 啊它们之间的这个互斥问题 那么，数据当进程 1 要想往这个物理内存里头读或者写数据的时候呢 那么，它实际上是往它的那块空间里头去写 而这个空间呢，因为映射到了这块物理内存，相当于往这里去写 那进程 2 也是一样，假设它是从这里去读数据 读呢，读到了自己的这个空间。 其实呢这个空间内容呢也就是读到了 这个进程的相应的，当做读操作的时候呢，它俩是关联起来的 这就是共享内存 第三种进程间通信方式呢，就是管道通信方式 管道通信方式呢，其实非常形象，是利用了一个传输的 介质，它是一个放了很多数据的一个缓冲的传输介质 然后，这个传输介质呢可以是内存的一块空间，也可以是某一个文件 用这样一个介质联接了两个不同的进程，这两个进程可以通过这个 中间这个传输介质来相互通信 假设这像一个管道一样，那么，发送进程实际上就是往管道里写 啊可以有很多发送进程都往管道里写 那么，接收进程呢，实际上就是从管道读，啊管道读 读完了，这些内容就没了。 所以写是 在尾巴上写，那么读呢是在头上读。 这就是一个 管道通信方式一个示意。 在实现管道通信的方式的过程中呢，我们可以看到 有几个问题。 第一个问题呢，是字符流的方式来写入读出。 这它是 没有格式的，是按字符流的方式写入读出。 而且是有 顺序的，先写的先读出。 另外呢，管道的本身要 提供这个，因为提供一个进程之间的一种协调能力啊，协调能力 因为，如果读进程发现根本就没有写进程了 那么读进程就不该再读了。 那么如果 写进程写的过程中，发现没有读进程了，那么 对方不存在了，所以写进程也没必要做，所以它有一个判断 啊对方是否存在这么一个，同时呢还有一个同步和互斥的问题，如果 管道里头没东西了，那么读的就得 停止，暂停啊暂停，直到有东西进来才能继续读 那么，读写还不能同时，对吧？所以这样的话呢，就要注意这个啊互斥问题 这就是管道通信方式 那么我们这里头只介绍三种啊进程间通信方式