@数据链路层的设计问题

基本概念

结点：主机、路由器

链路：网络中两个结点之间的物理通道，链路的传输介质主要有双绞线、光纤和微波。分为有线链路、无线链路

数据链路：网络中两个结点之间的逻辑通道，把实现控制数据传输协议的硬件和软件加到链路上就构成数据链路

帧：链路层的协议数据单元，封装网络层数据报

也有人采用另外的术语。这就是把链路分为物理链路和逻辑链路。
物理链路就是上面所说的链路。
逻辑链路就是上面的数据链路，是物理链路加上必要的通信协议

主要任务：负责通过一条链路从一个结点向另一个物理链路直接相连的相邻结点传送数据报

代表设备： 网卡（适配器）

主要功能

数据链路层在物理层提供服务的基础上**向网络层提供服务**，其最基本的服务是**将源自网络层来的数据可靠地传输到相邻节点的目标机网络层。**其主要作用是**加强物理层传输原始比特流的功能**，将物理层提供的可能出错的物理连接改造成为**逻辑上无差错的数据链路**，使之对网络层表现为一条无差错的链路

- 为网络层提供服务。无确认无连接服务，有确认无连接服务，有确认面向连接服务。**有连接一定有确认！**
- 链路管理，即连接的建立、维持、释放（用于面向连接的服务）
- 组帧
- 流量控制。限制发送方
- 差错控制（帧错/位错）

通信质量好有线传输链路

通信质量差的无线传输链路

使用信道

- 点对点信道 一对一
- 广播信道 一对多 必须使用专用的共享信道协议来协调这些主机的数据发送

封装成帧与透明传输

封装成帧

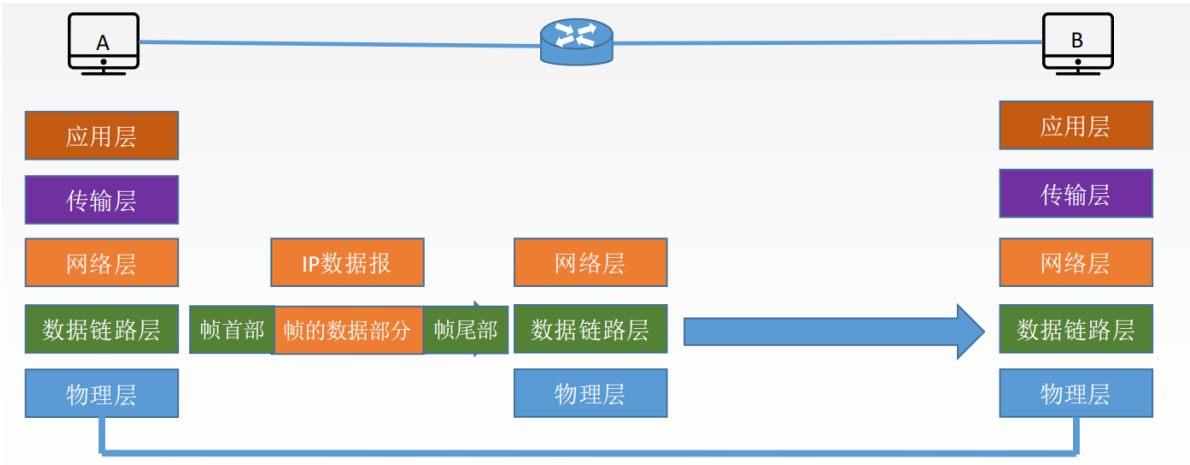
发送端将网络层的**IP数据报**进行包装，封装成帧，添加帧首部与帧尾部

接收端在收到物理层上交的比特流后，就能根据首部和尾部的标记，从收到的比特流中识别帧的开始和结束

首部和尾部包含许多的控制信息，他们的一个重要作用：**帧定界**（确定帧的界限）

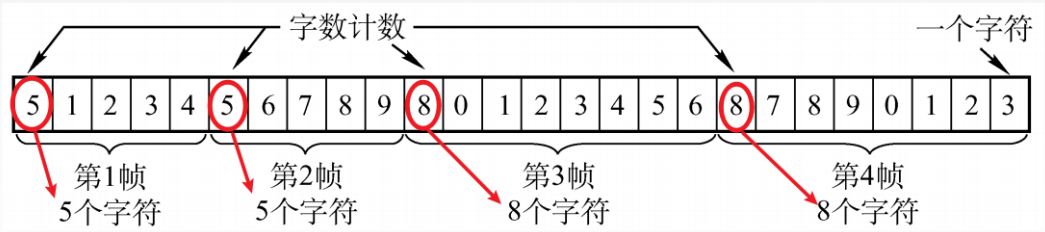
帧同步：接收方应当能从接收到的二进制比特流中区分出帧的起始和终止

组帧的四种方法：1.字符计数法，2.字符（节）填充法，3.零比特填充法，4.违规编码法。



1.字符计数法

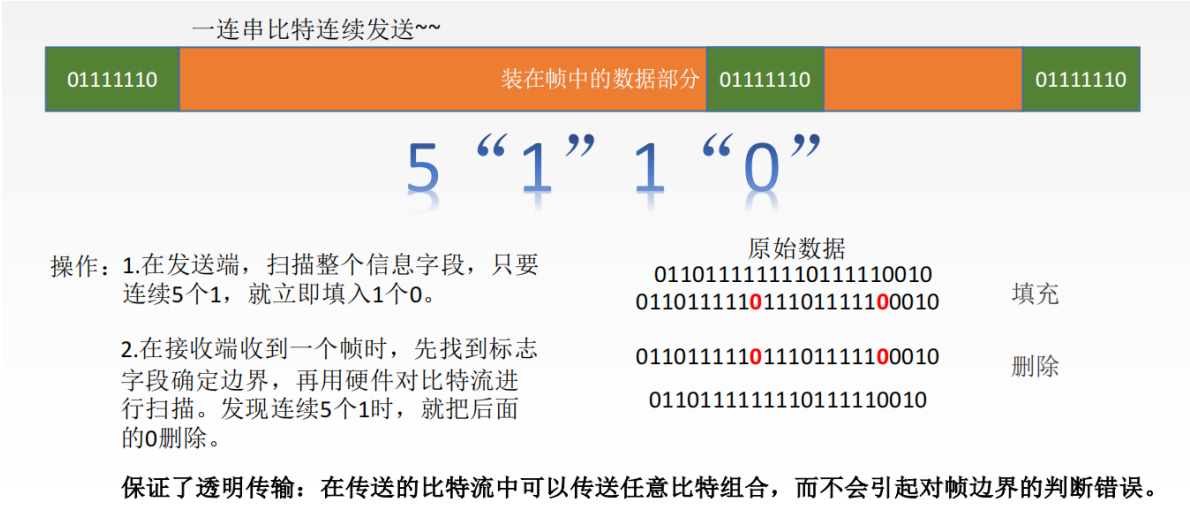
帧首部使用一个计数字段（第一个**字节**，八位）来标明帧内字符数。



2.字符填充法

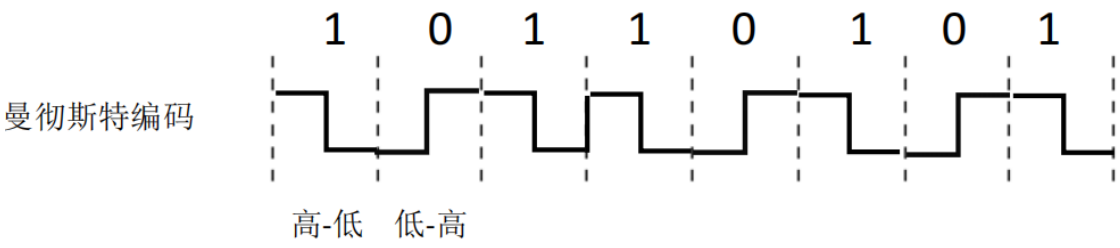
含有特殊字符的处理方法

3.零比特填充法



保证了透明传输：在传送的比特流中可以传送任意比特组合，而不会引起对帧边界的判断错误。

4.违规编码法



可以用“**高-高**”，“**低-低**”来定界帧的起始和终止。

由于字节计数法中Count字段的脆弱性（其值若有差错将导致灾难性后果）及字符填充实现上的复杂性和不兼容性，**目前较普遍使用的帧同步法是 两个比特填充法和违规编码法**

透明传输

透明传输是指不管所传数据是什么样的比特组合，都应当能够在链路上传送。因此，链路层就“看不见”有什么妨碍数据传输的东西

当所传数据中的比特组合恰巧与某一个控制信息完全一样时，就必须采取适当的措施，使收方不会将这样的数据误认为是某种控制信息。这样才能保证数据链路层的传输是透明的

@差错检测和纠正

差错控制（检错编码）

> 传输中的差错都是由于噪声引起的

全局性：由于线路本身电气特性所产生的随机噪声（热噪声），**是信道固有的**，随机存在的。

解决办法：提高**信噪比**来减少或避免干扰。（对传感器下手）

局部性：外界特定的短暂原因所造成的冲击噪声，是产生差错的主要原因

解决办法：通常利用**编码技术**来解决

需要注意的是

这里的编码与物理层的编码有很大不同

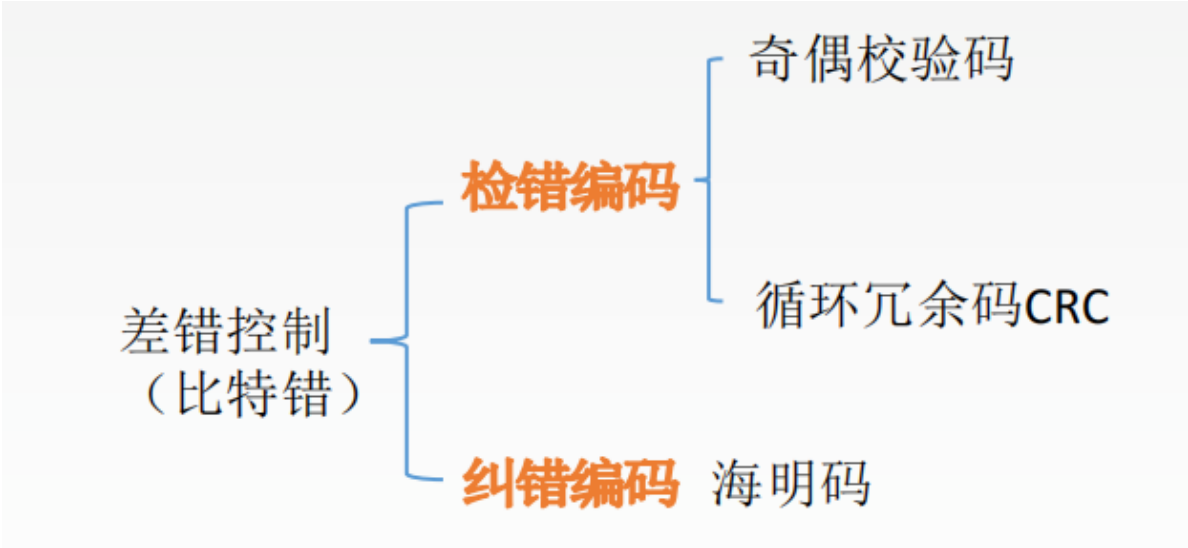
> 数据链路层编码和物理层的数据编码与调制不同。**物理层编码针对的是单个比特**，解决传输过程中比特的同步等问题，如曼彻斯特编码。而**数据链路层的编码针对的是一组比特**，它通过冗余码的技术实现一组二进制比特串在传输过程是否出现了差错

差错类型

位错

比特位出错，1变成0，0变成1

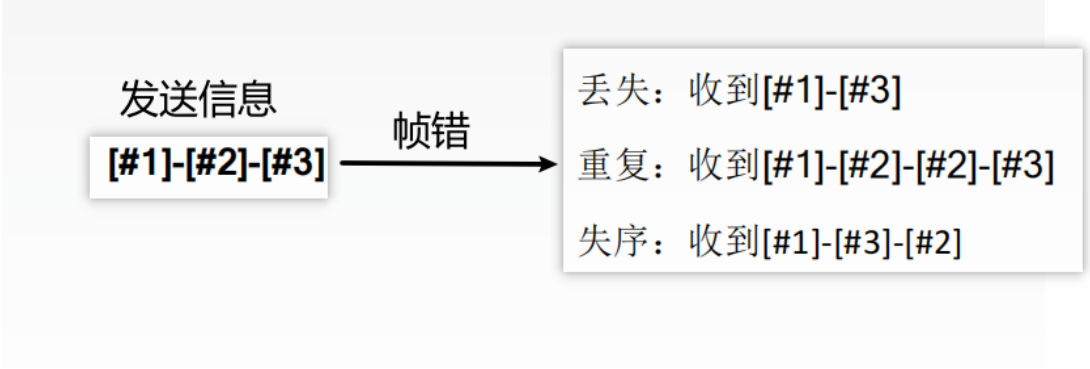
解决方法：



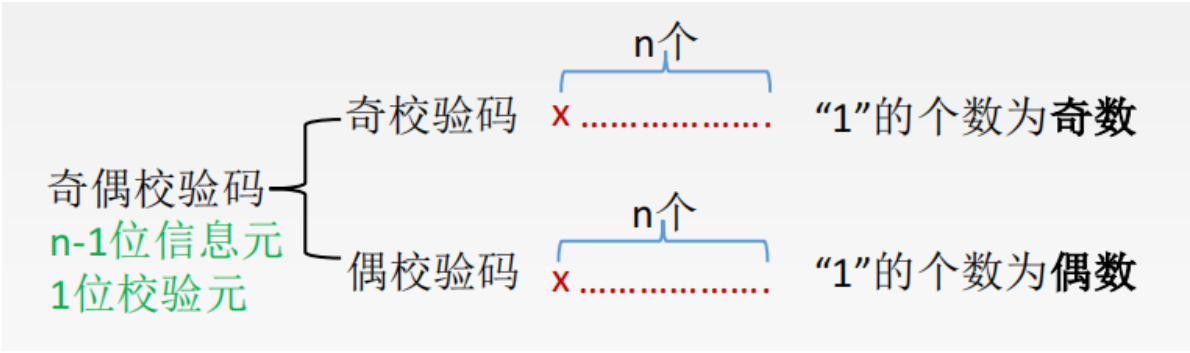
其中的循环冗余码

> 在数据发送之前，先按某种关系附加上一定的冗余位，构成一个符合某一规则的码字后再发送。当要发送的有效数据变化时，相应的冗余位也随之变化，使码字遵从不变的规则。接收端根据收到码字是否仍符合原规则，从而判断是否出错

帧错



奇偶校验码

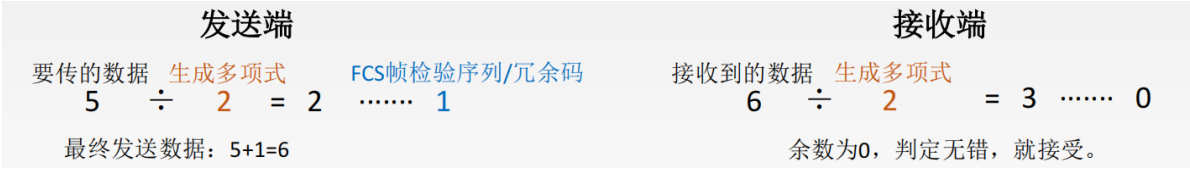


若发生偶数个错误时，会无法检测出

只能检查出奇数个比特错误，检错能力为50%

CRC循环冗余码

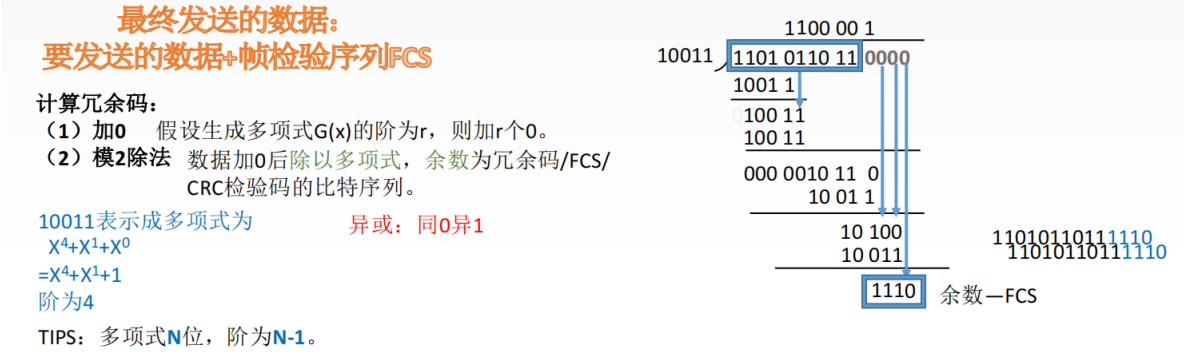
检错方式



例题

要发送的数据是`1101011011`，采用CRC校验，生成多项式是`10011`，那么最终发送的数据应该是?

因为生成多项式的阶数为 4　则在源数据后加4个零



先将 源数据 / 生成多项式　　用余数与源数据相加　得到最终发送的数据为 `11010110111110`

接收方检错过程

把收到的每一个帧都除以同样的除数，然后检查得到的余数R

1.余数为0，判定这个帧没有差错，接受
2.余数为不为0，判定这个帧有差错（无法确定到位），丢弃

FCS的生成以及接收端CRC检验都是由**硬件**实现，处理很迅速，因此不会延误数据的传输

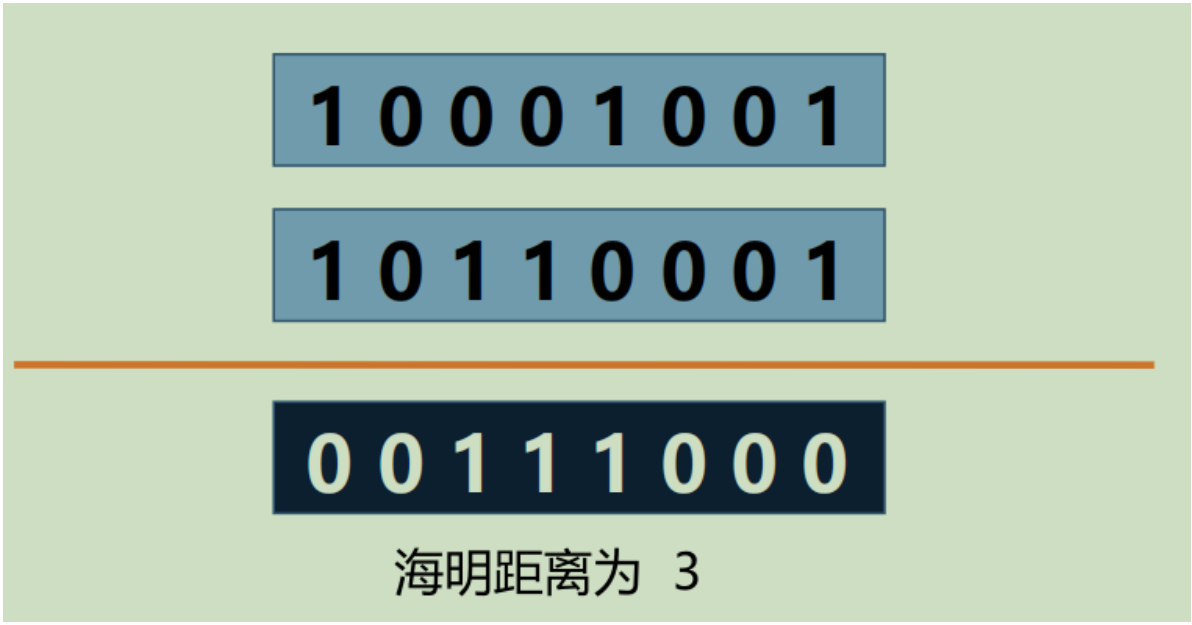
纠错编码---海明码

海明码：发现双比特错，纠正单比特错

动一发而牵全身

过程：　确定校验码位数r　--->　确定校验码和数据的位置 --->　求出校验码的值 ----->　检错并纠错

给定两个码字 10001001 和 10110001，如果要分析两者之间多少位不 同，通过异或XOR计算两个数据，得到1的个数即为不同的位数。两个码字 中不同位的个数称为**海明距离**



确定校验码位数r

海明不等式

$2^r \geq k+r+1$

r为冗余信息位，k为信息位

要发送的数据：D=101101

数据的位数k=6，
满足不等式的最小r为4，
也就是D=101101的海明码应该有6+4=10位，
其中原数据6位，效验码4位。

确定校验码和数据的位置

校验码必须是在2^n次方位置　　信息码的分布是非2n次方位置

D=101101

假设这4位校验码分别为P_1、P_2、P_3、P_4；数据从左到右为D_1、D_2、……、D_6。

放在2的几次方的位置　　按序把空填满

数据位	1	2	3	4	5	6	7	8	9	10
代码	P_1	P_2	D_1	P_3	D_2	D_3	D_4	P_4	D_5	D_6
实际值			1		0	1	1		0	1

求出校验码的值

校验位的值代表了代码字中部分数据位的奇偶性（最终要根据是采用奇校验，还是偶校验来确定），其所在位置决定了要校验的比特位序列

第i位校验码从当前位开始，每次连续校验i（这里是数值 i，不是第 i 位，下同）位后再跳过i位，然后再连续校验i位，再跳过i位，以此类推

p1（第1个校验位，也是整个码字的第1位）的校验规则是：**从当前位数起，校验1位，然后跳过1位，再校验1位，再跳过1位**，……。这样就可得出p1校验码位可以校验的码字位包括：第1位（也就是p1本身）、第3位、第5位、第7位、第9位、第11位、第13位、第15位，……。然后根据所采用的是奇校验，还是偶

校验，最终可以确定该校验位的值

p2（第2个校验位，也是整个码字的第2位）的校验规则是：**从当前位数起，连续校验2位，然后跳过2位，再连续校验2位，再跳过2位**，……。这样就可得出p2校验码位可以校验的码字位包括：第2位（也就是p2本身）、第3位，第6位、第7位，第10位、第11位，第14位、第15位，……。同样根据所采用的是奇校验，还是偶校验，最终可以确定该校验位的值

p3（第3个校验位，也是整个码字的第4位）的校验规则是：**从当前位数起，连续校验4位，然后跳过4位，再连续校验4位，再跳过4位**，……。这样就可得出p4校验码位可以校验的码字位包括：第4位（也就是p4本身）、第5位、第6位、第7位，第12位、第13位、第14位、第15位，第20位、第21位、第22位、第23位，……。同样根据所采用的是奇校验，还是偶校验，最终可以确定该校验位的值

p4（第4个校验位，也是整个码字的第8位）的校验规则是：**从当前位数起，连续校验8位，然后跳过8位，再连续校验8位，再跳过8位**，……。这样就可得出p4校验码位可以校验的码字位包括：第8位（也就是p4本身）、第9位、第10位、第11位、第12位、第13位、第14位、第15位，第24位、第25位、第26位、第27位、第28位、第29位、第30位、第31位，……。同样根据所采用的是奇校验，还是偶校验，最终可以确定该校验位的值

D=101101

二进制	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010
数据位	1	2	3	4	5	6	7	8	9	10
代码	P_1	P_2	D_1	P_3	D_2	D_3	D_4	P_4	D_5	D_6
实际值	0	0	1	0	0	1	1	1	0	1

令所有要校验的位异或=0。

> **异或：同为0　异为1**

结果为

故　101101　的海明码为　0010011101

检错并纠错

D=101101

数据位	1	2	3	4	5	6	7	8	9	10
代码	P_1	P_2	D_1	P_3	D_2	D_3	D_4	P_4	D_5	D_6
实际值	0	0	1	0	1	1	1	1	0	1

故101101的海明码为0010011101。

假设第五位出错，因此接收到的数据位0010111101。

令所有要校验的位异或运算。

对不同位数上为1 的位数的二进制数做异或运算

```
1 x x x : 1 0 1 = 0
x 1 x x : 0 1 1 1 = 1
x x 1 x : 0 1 1 1 1 = 0
x x x 1 : 0 1 1 1 0 = 1
```

得到的序列为 0101

二进制序列为0101，恰好对应十进制5，这样就找到了出错的位置，即出错位是第5位

奇校验，要保证插入监督码后”1“的个数为奇数，其中数据位是不变的，所以监督位置1或0；也就是说，若数据位中”1“的个数是偶数，则监督位为1；若数据位中”1“的个数是奇数，则监督位为0。

偶校验，要保证插入监督码后”1“的个数为偶数，其中数据位是不变的，所以监督位置1或0；也就是说，若数据位中”1“的个数是偶数的话，则监督位为0；若数据位中”1“的个数是奇数，则监督位为1

@流量控制

> 较高的发送速度和较低的接收能力的不匹配，会造成传输出错，因此流量控制也是数据链路层的一项重要工作

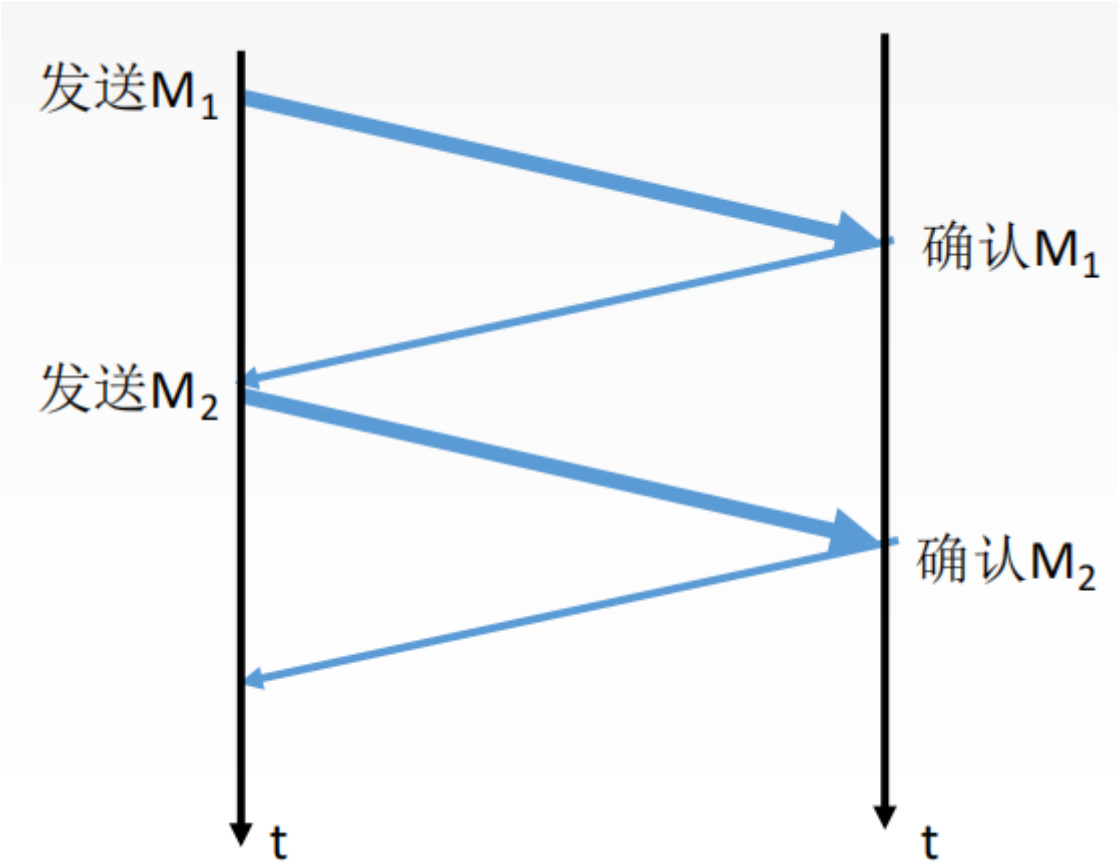
数据链路层的流量控制是点对点的，而传输层的流量控制是端到端的

数据链路层流量控制手段：接收方收不下就不回复确认。传输层流量控制手段：接收端给发送端一个窗口公告

流量控制的方法

停止-等待协议 相对低效

每发送完一个帧就停止发送，等待对方的确认，在收到确认后再发送下一个帧



滑动窗口协议 相对高效

- 后退N帧协议 GBN
- 选择重传协议 SR

发送窗口与接收窗口不断移动

- 停止-等待协议　　发送窗口大小=1，接收窗口大小=1
- 后退N帧协议（GBN）发送窗口大小1，接收窗口大小=1
- 选择重传协议（SB）发送窗口大小1，接收窗口大小1

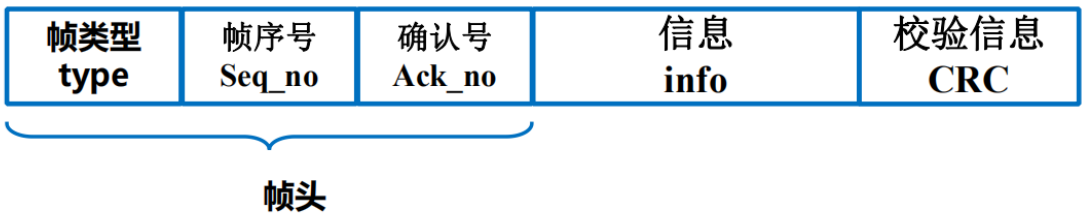
可靠传输：发送端发啥，接收端收啥。

流量控制：控制发送速率，使接收方有足够的缓冲空间来接收每一个帧。

滑动窗口解决
- 流量控制（收不下就不给确认，想发也发不了）
- 可靠传输（发送方自动重传）

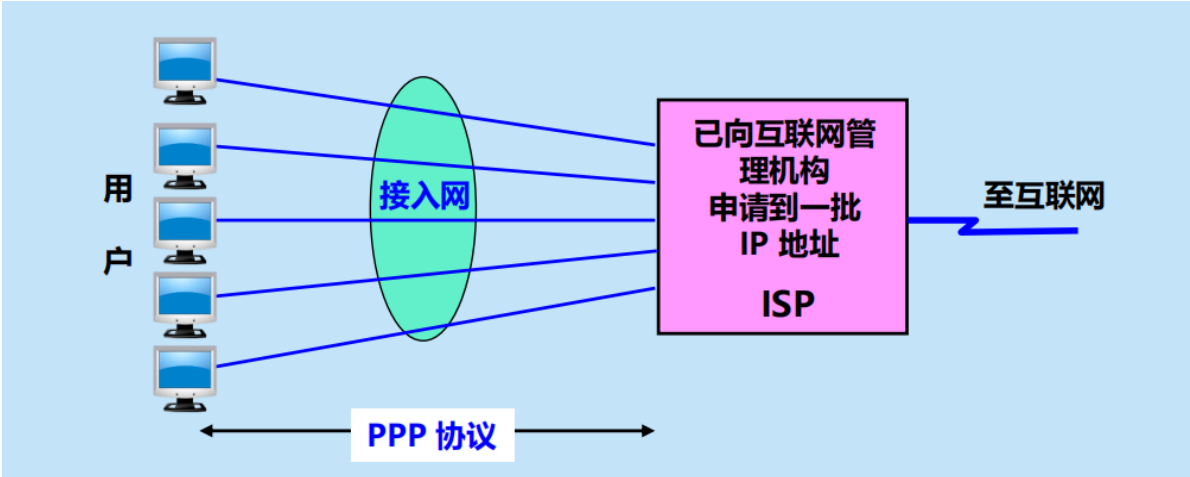
@基本数据链路层协议

基于上述讨论，一个数据链路层的帧至少应该包括下列内容：



PPP协议

- **对于点对点的链路，目前使用得最广泛的数据链路层协议是点对点协议 PPP (Point-to-Point Protocol)。**
- **PPP 协议在 1994 年就已成为互联网的正式标准。**



PPP协议的特点

- **简单**——这是首要的要求
- **封装成帧**——必须规定特殊的字符作为帧定界符
- **透明性**——必须保证数据传输的透明性
- **多种网络层协议**——能够在同一条物理链路上同时支持多种网络层协议（网络层协议可以是IPIPX，IPv6等，PPP支持对这些协议的标识）
- **多种类型链路**——能够在多种类型的链路上运行（多种类型链路是指：光纤，同轴电缆，双绞线，电话线，可在多种媒体介质上传输）
- **差错检测**——能够对接收端收到的帧进行检测，并立即丢弃有差错的帧
- **检测连接状态**——能够及时自动检测出链路是否处于正常工作状态。
  （因不同的情形显示不同的状态，欠费、登录账号密码错误、网线没有插好等，提供各种状态报告）
- **最大传送单元**——必须对每一种类型的点对点链路设置最大传送单元MTU的标准默认值，促进各种实现之间的互操作性
- **网络层地址协商**——必须提供一种机制使通信的两个网络层实体能够通过协商知道或能够配置彼此的网络层地址
- **数据压缩协商**——必须提供一种方法来协商使用数据压缩算法。
  （在数据传输前，进行压缩，节省带宽）

PPP协议的组成

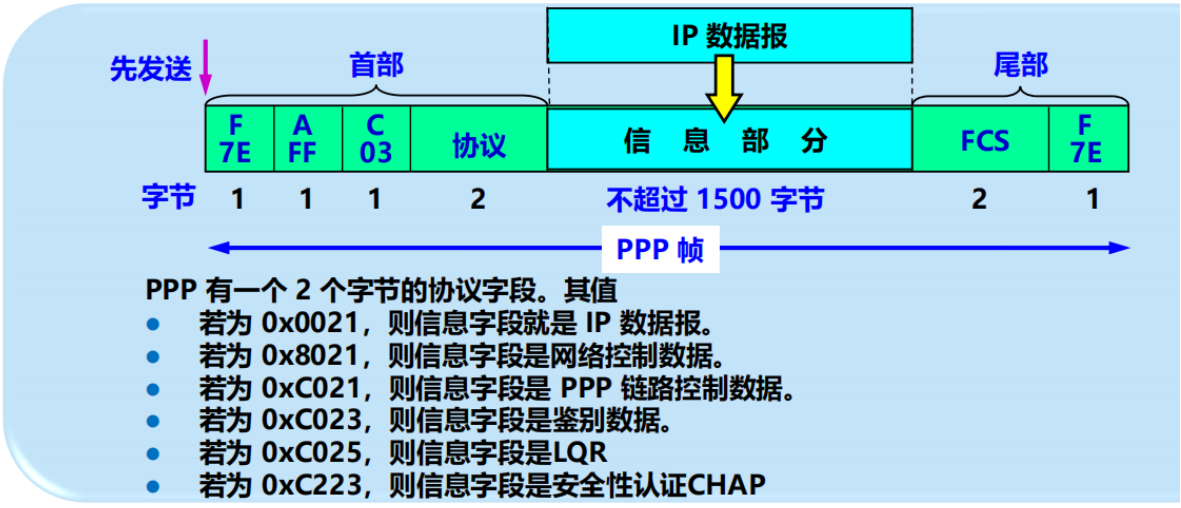
- 一个将IP数据报封装到串行链路的方法。（支持异步/同步串行）
- 链路控制协议LCP（Link Control Protocol）建立并维护数据链路连接，负责身份验证
- 网络控制协议 NCP（Network Control Protocol）
  允许点到点连接上使用多种网络层协议
  （只有当LCP工作正常后，NCP才能通信）
  （例如拨号上网，只有不欠费，账号密码正确之后，才能从ISP获取到IP地址，然后你的网络才能通信）

PPP协议的帧格式

3.2.2 PPP 协议的帧格式

- PPP 帧的首部和尾部分别为 4 个字段和 2 个字段。
- 标志字段 F = 0x7E （符号“0x”表示后面的字符是用十六进制表示。十六进制的 7E 的二进制表示是 01111110）。
- 地址字段 A 只置为 0xFF。地址字段实际上并不起作用。
- 控制字段 C 通常置为 0x03。
- PPP 是面向字节的，所有的 PPP 帧的长度都是整数字节。

在PPP协议中：第一个字节和最后的字节是相同的，都是0x7E，标志字段
A： 目标地址，但在PPP协议中不起作用，因为是点到点通信，
中间没有其他节点，发送方发送，接收端就能接收，无需特意指定地址

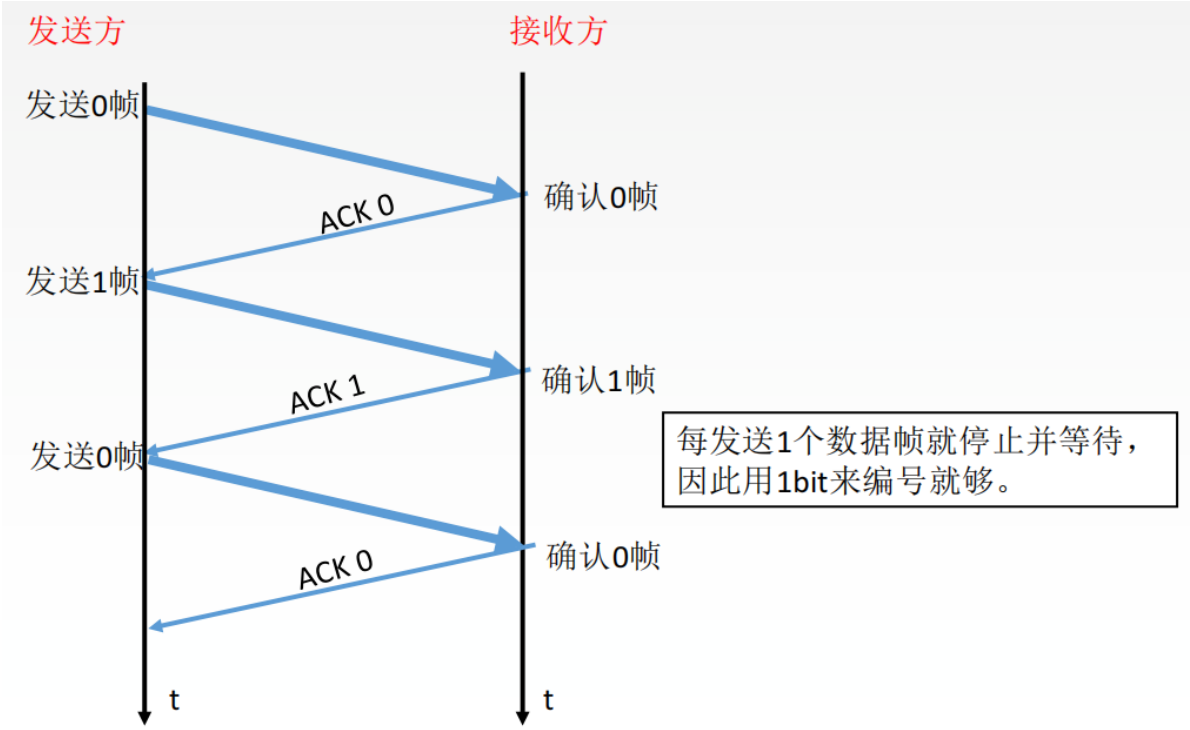


PPP 有一个 2 个字节的协议字段。其值
- 若为 0x0021，则信息字段就是 IP 数据报。
- 若为 0x8021，则信息字段是网络控制数据。
- 若为 0xC021，则信息字段是 PPP 链路控制数据。
- 若为 0xC023，则信息字段是鉴别数据。
- 若为 0xC025，则信息字段是LQR
- 若为 0xC223，则信息字段是安全性认证CHAP

停止---等待协议

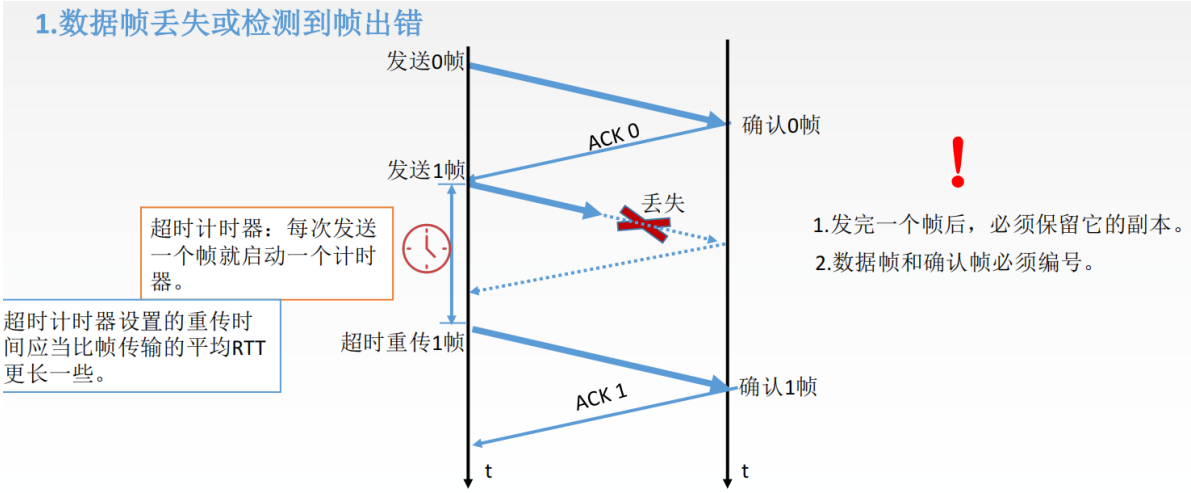
> 除了比特出差错，底层信道还会出现丢包问题
> 为了实现流量控制

虽然现在常用全双工通信方式，但为了讨论问题方便，仅考虑一方发送数据（发送方），一方接收数据（接收方）
因为是在讨论可靠传输的原理，所以并不考虑数据是在哪一个层次上传送的
“停止-等待”就是每发送完一个分组就停止发送，等待对方确认，在收到确认后再发送下一个分组

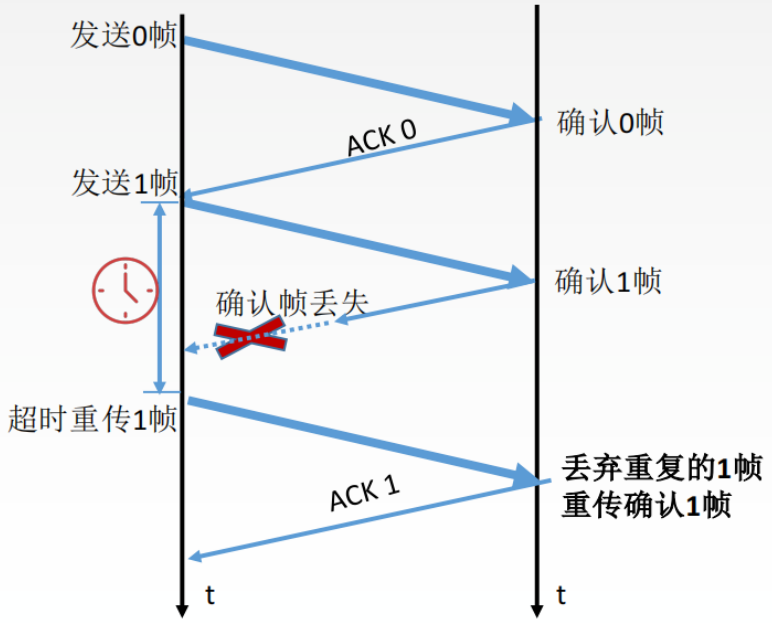


无差错情况&有差错情况

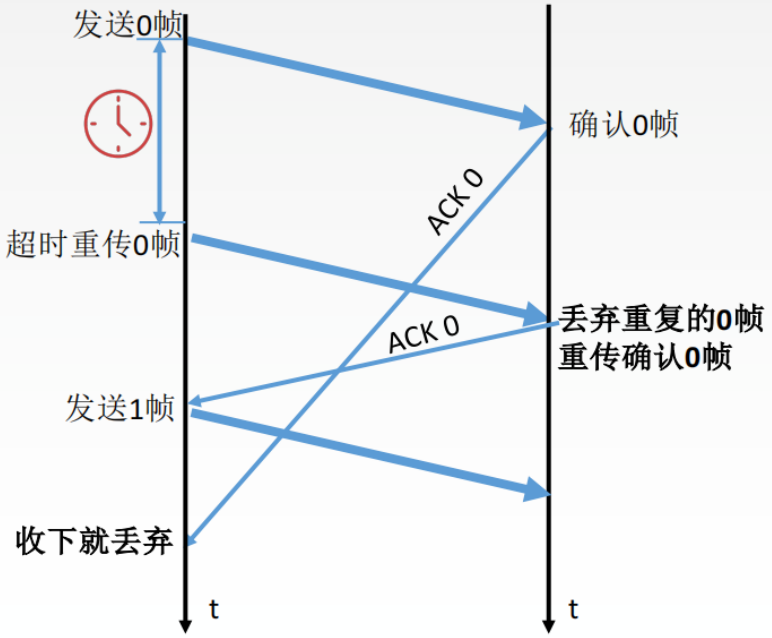
有差错情况



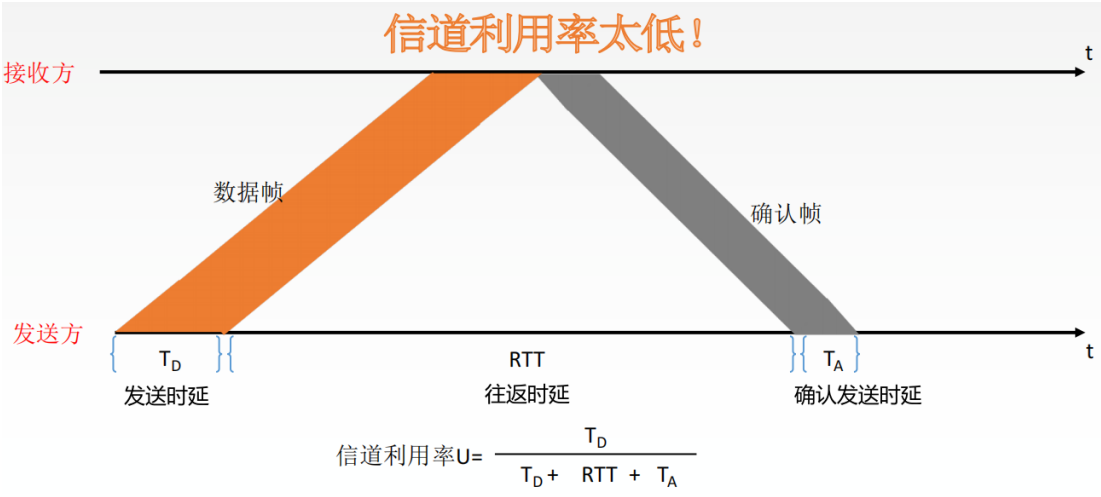
2.ACK丢失

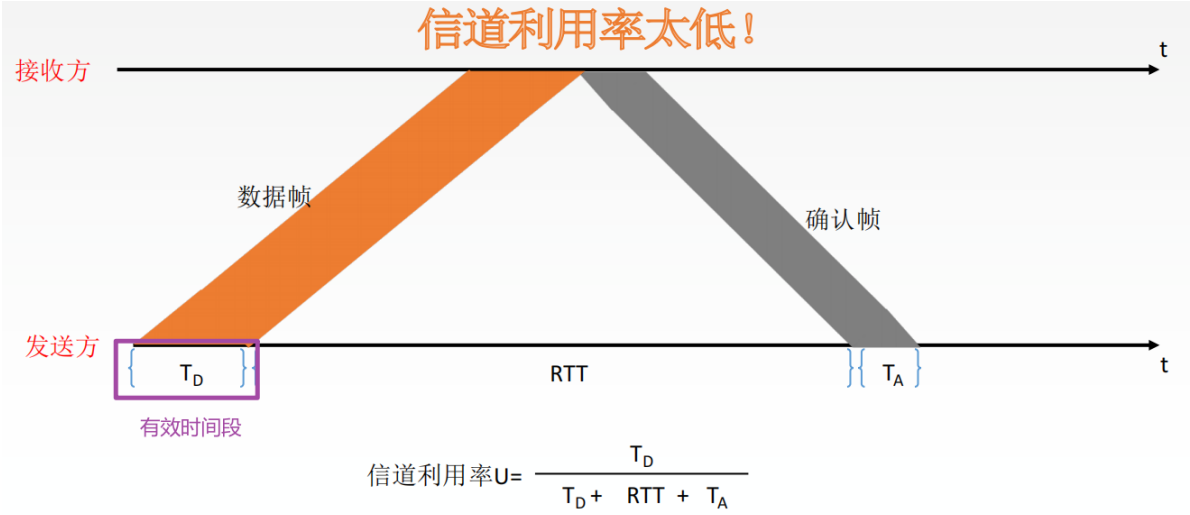


3.ACK迟到



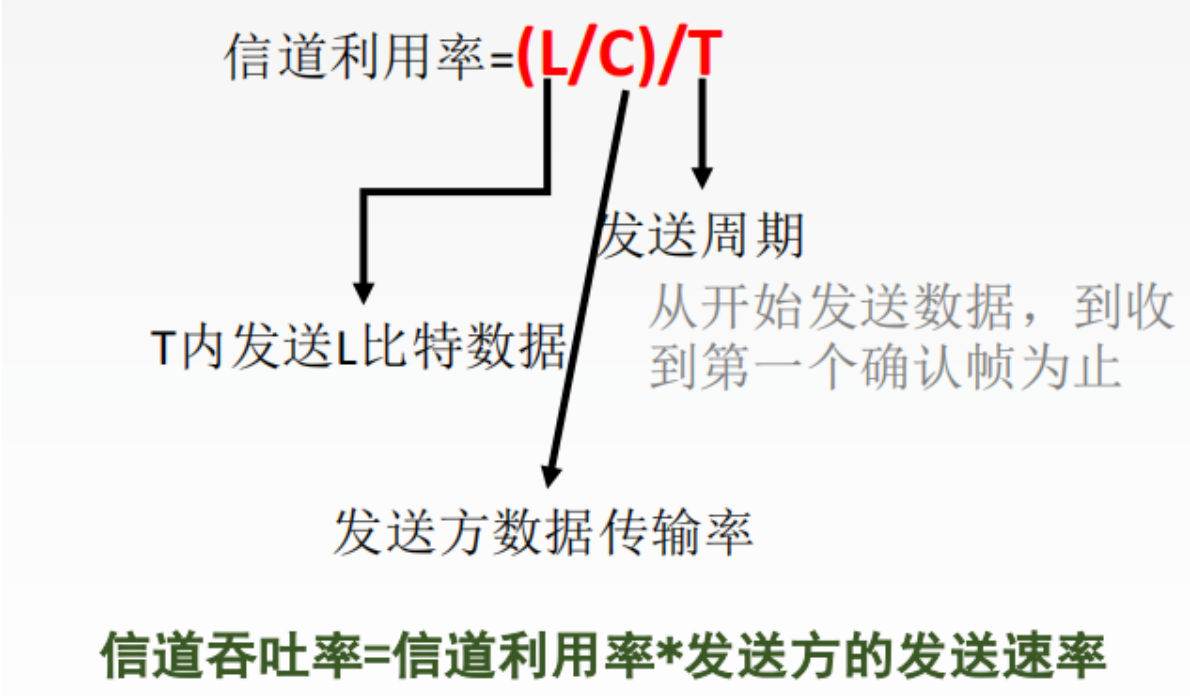
性能分析





信道利用率

> 发送方在一个发送周期内，有效地发送数据所需要的时间占整个发送周期的比率



例题：

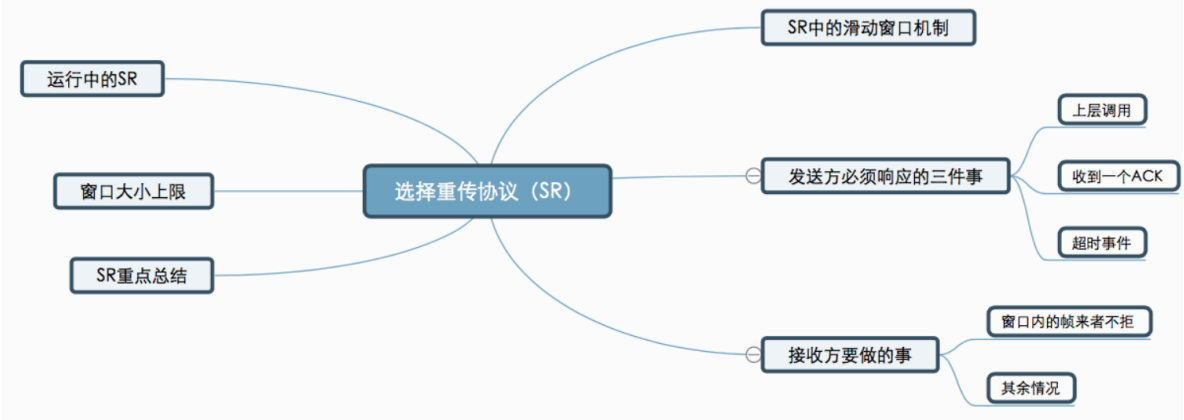
一个信道的数据传输率为4kb/s，单向传播时延为30ms，如果使停止-等待协议的信道最大利用率达到80%，要求的数据帧长度至少为多少？

$$80\% = \frac{L/4}{L/4+2 \times 30ms}$$

$$= \frac{L}{L+2 \times 30ms \times 4kb/s}$$

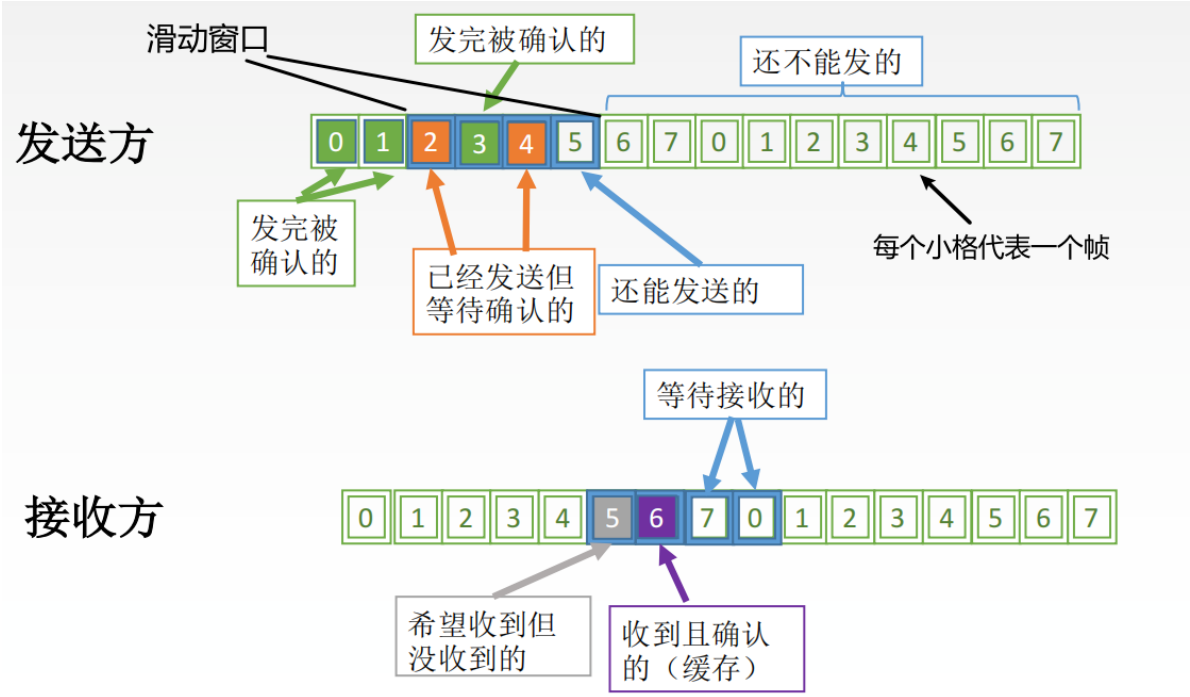
$$L=960bit$$

选择重传协议



停等协议存在一定弊端：**通过累计确认导致了批量重传**

而选择重传协议解决了这个问题：设置单个确认，同时加大接收窗口，设置接收缓存，缓存乱序到达的帧



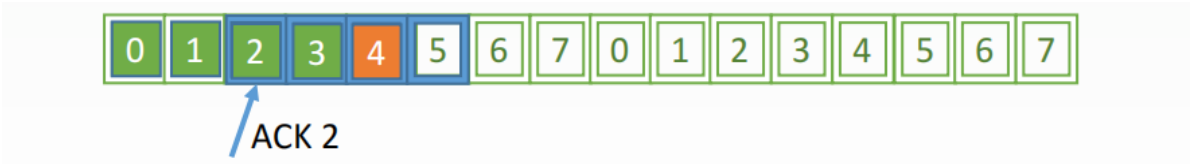
作为发送方需要做的三件事

1.上层的调用

从上层收到数据后，SR发送方检查下一个可用于该帧的序号，如果序号位于发送窗口内，则发送数据帧；否则就像GBN一样，要么将数据缓存，要么返回给上层之后再传输

2.收到了一个ACK（确认帧）

如果收到ACK，加入该帧序号在窗口内，则SR发送方将那个被确认的帧标记为已接收。如果该帧序号是窗口的下界（最左边第一个窗口对应的序号），则窗口向前移动到具有最小序号的未确认帧处。如果窗口移动了并且有序号在窗口内的未发送帧，则发送这些帧



3.超时事件

每个帧都有自己的定时器，一个超时事件发生后只重传一个帧

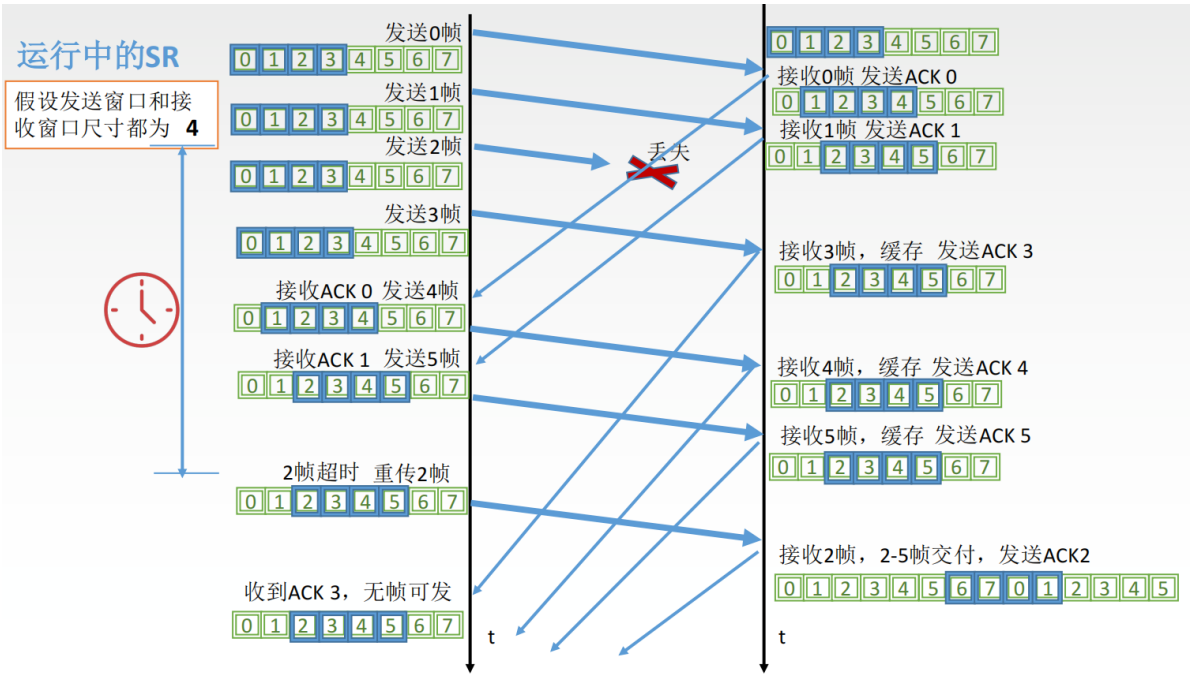
作为接收方需要做的三件事

来者不拒（窗口内的帧）

接收方将确认一个正确接收的帧而**不管其是否按序。**失序的帧将被**缓存**，并返回给发送方一个该帧的确认帧【收谁确认谁】，直到所有帧（即序号更小的帧）皆被收到为止，这时才可以将一批帧按序交付给上层，然后**向前移动滑动窗口**

如果收到了窗口序号外（小于窗口下界）的帧，就返回一个ACK
其他情况，就忽略该帧

运行中的SR

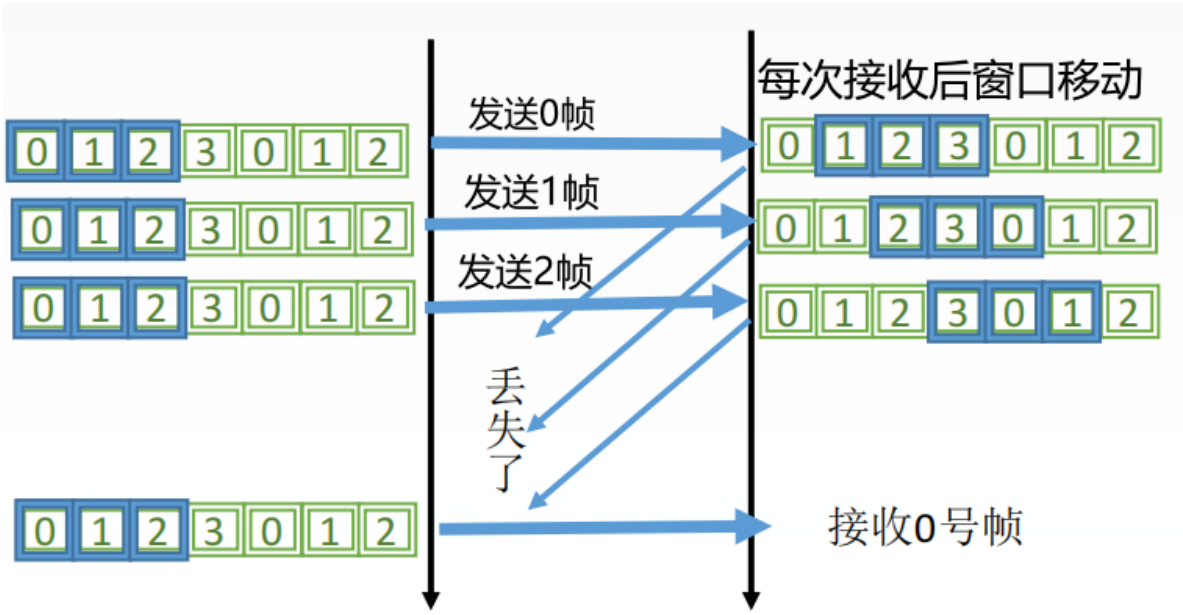


滑动窗口长度

发送窗口最好等于接收窗口。（大了会溢出，小了没意义）

$$W_{Tmax}=W_{Rmax}=2^{(n-1)}$$

例子



协议重点

- 对数据帧逐一确认重传
- 只重传错帧

- 接收方有缓存
- `发送窗口 = 接收窗口 = 2^(n-1)`

习题

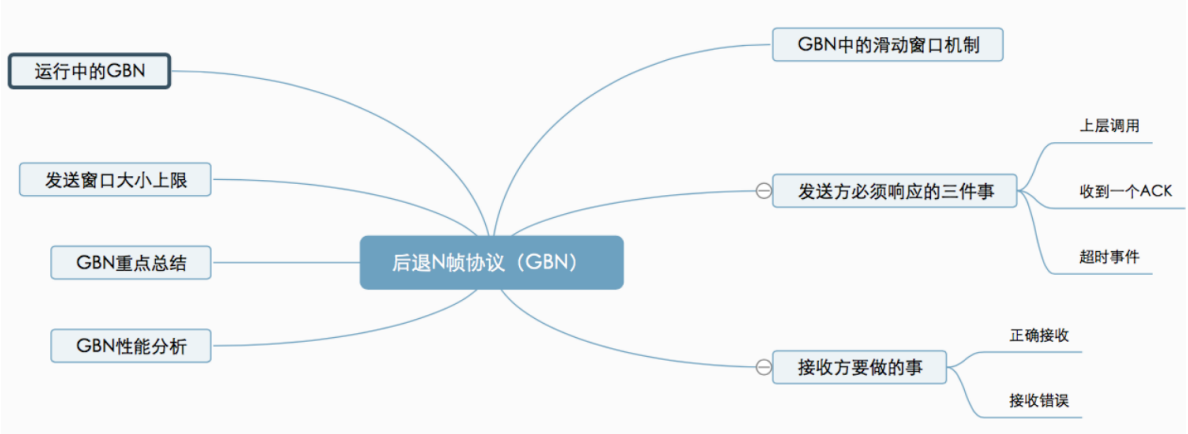
数据链路层采用了选择重传（SR）协议，发送方已经发送了编号为0~3的帧。现已收到1号帧的确认，而0、2号帧依次超时，则发送方需要重传的帧数是（ ）

解析：

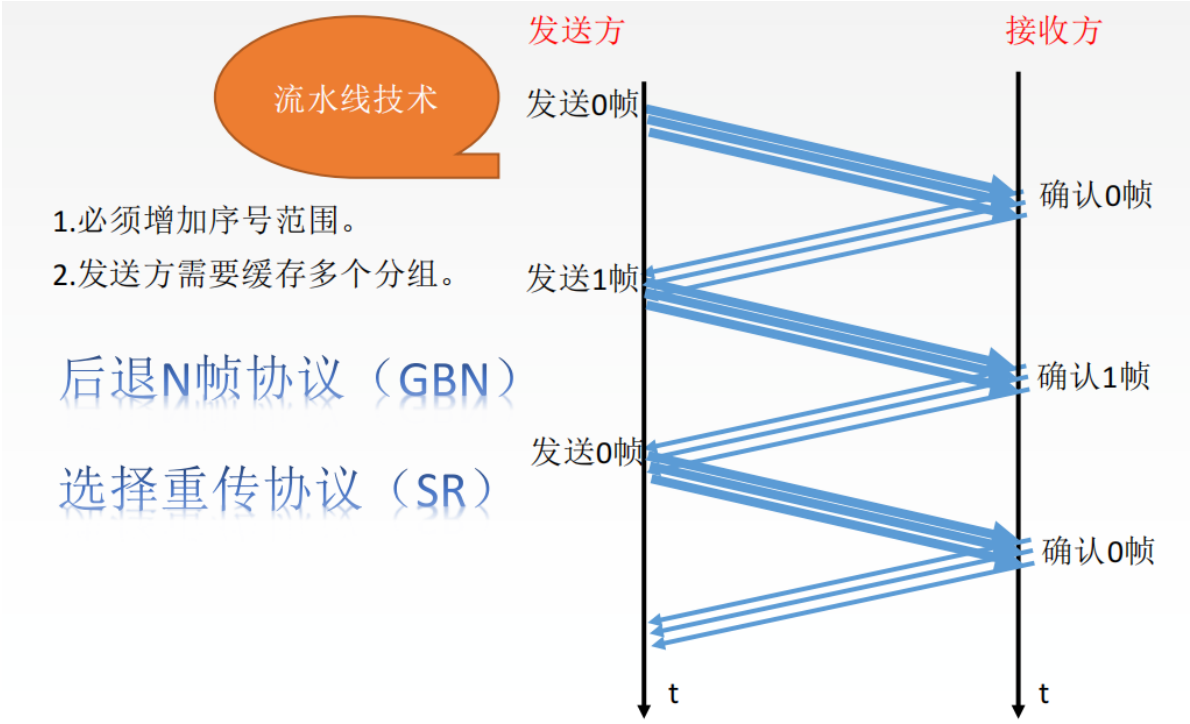
> 1号帧已经确认，无需重传
>
> 3号帧未知，可能在路上，或者是已经确认
>
> 只有0，2号帧超时
>
> 所以答案为2　需要重新发送0，2号帧

后退N帧协议

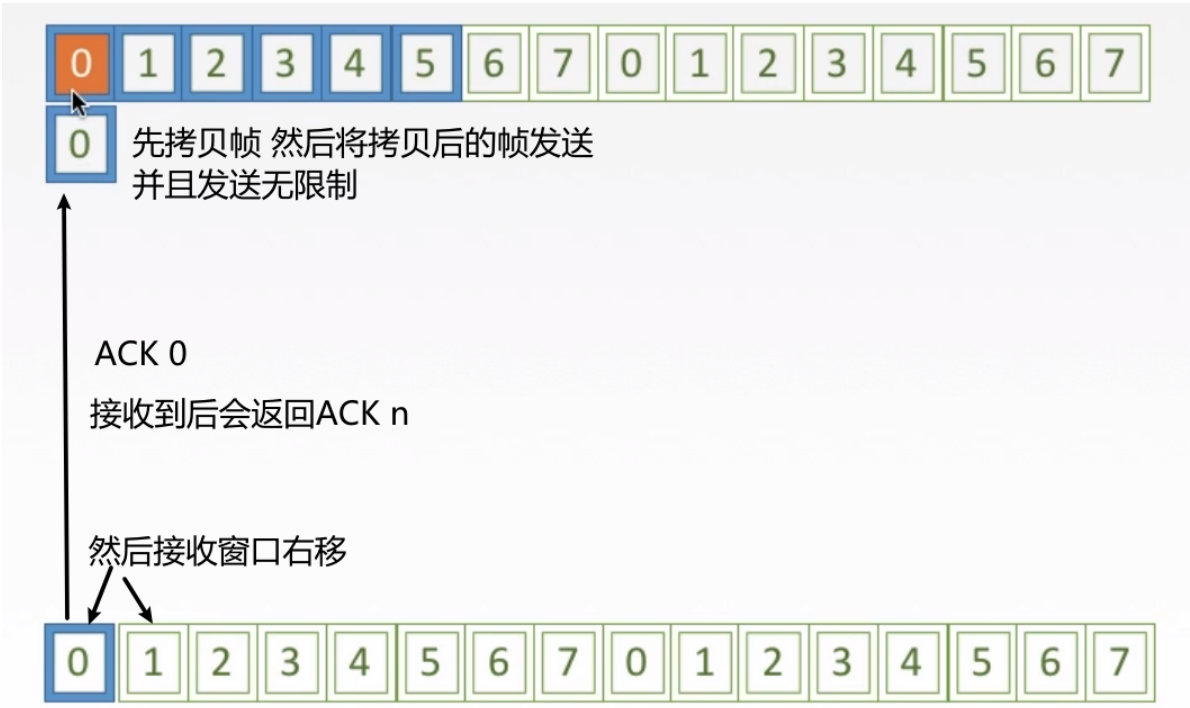
未收到确认会一直发送



前两个协议 的信道利用率都不是很高


流水线技术
1.必须增加序号范围。
2.发送方需要缓存多个分组。
后退N帧协议（GBN）
选择重传协议（SR）
发送方
接收方
发送0帧
确认0帧
发送1帧
确认1帧
发送0帧
确认0帧
t
t

滑动窗口


0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
0
先拷贝帧 然后将拷贝后的帧发送
并且发送无限制
ACK 0
接收到后会返回ACK n
然后接收窗口右移
0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7

GBN发送方必须响应的三件事

1.上层的调用

上层要发送数据时，发送方先检查发送窗口是否已满，如果未满，则产生一个帧并将其发送；如果窗口已满，发送方只需将数据返回给上层，暗示上层窗口已满。上层等一会再发送。（实际实现中，发送方可以缓存这些数据，窗口不满时再发送帧）

2.收到了一个ACK

GBN协议中，对n号帧的确认采用累积确认的方式，标明接收方已经收到n号帧和它之前的全部帧

3.超时事件

协议的名字为后退N帧/回退N帧，来源于出现丢失和时延过长帧时发送方的行为。就像在停等协议中一样，定时器将再次用于恢复数据帧或确认帧的丢失。如果出现超时，发送方重传所有已发送但未被确认的帧

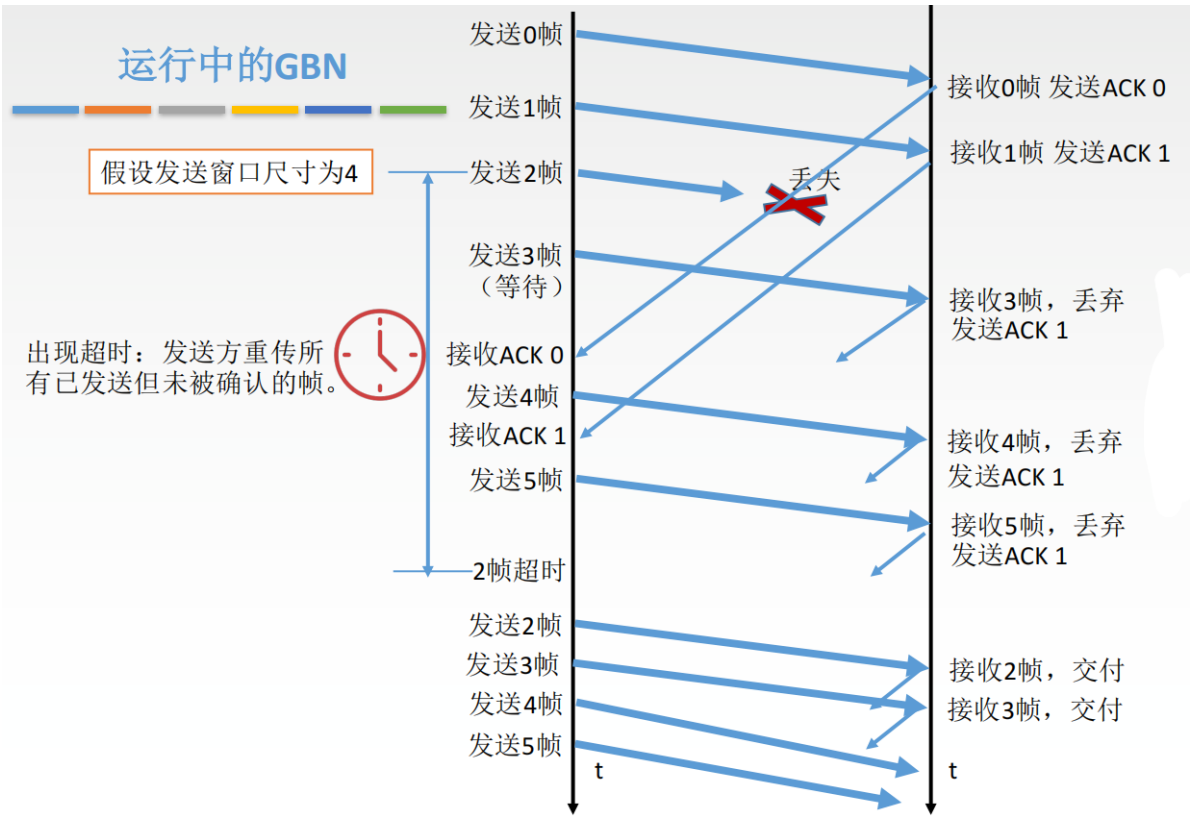
GBN接收方必须做的三件事

如果正确收到n号帧，并且按序，那么接收方为n帧发送一个ACK，并将该帧中的数据部分交付给上层

其余情况都丢弃帧，并为最近按序接收的帧重新发送ACK。接收方无需缓存任何失序帧，只需要维护一个信息：`expectedseqnum`（下一个按序接收的帧序号）

运行中的GBN

特点是：若有一个帧n未收到，即接收方不断发送ACK n-1 直到发送方发现帧n超时 发送帧n 恢复正常



滑动窗口长度

若采用n个比特对帧编号，那么发送窗口的尺寸`Wr`应满足：

$$1 \leq W_T \leq 2^n - 1$$

因为发送窗口尺寸过大，就会使得接收方无法区别新帧和旧帧

GBN协议重点总结

- 累积确认（偶尔捎带确认）
- 接收方只按顺序接收帧。不按序无情丢弃
- 确认序列号最大的、按序到达的帧
- 发送窗口最大为20-1，接收窗口大小为1

例题

数据链路层采用了后退N帧（GBN）协议，发送方已经发送了编号为0~7的帧。当计时器超时时，若发送方只收到0、2、3号帧的确认，则发送方需要重发的帧是（　　）

解析：

> 因为已经收到3号帧的确认，说明0，1，2，3帧的都已完成传输，所以只需要重新发送4，5，6，7帧

GBN协议优缺点分析

- 因连续发送数据帧而提高了信道利用率
- 在重传时必须把原来已经正确传送的数据帧重传，使传送效率降低。