计算机网络基础

第五章 数据链路层及协议

任课教师：马婷婷

第一部分 ▶

使用点对点信道的数据链路层

5.1.1 数据链路层的地位

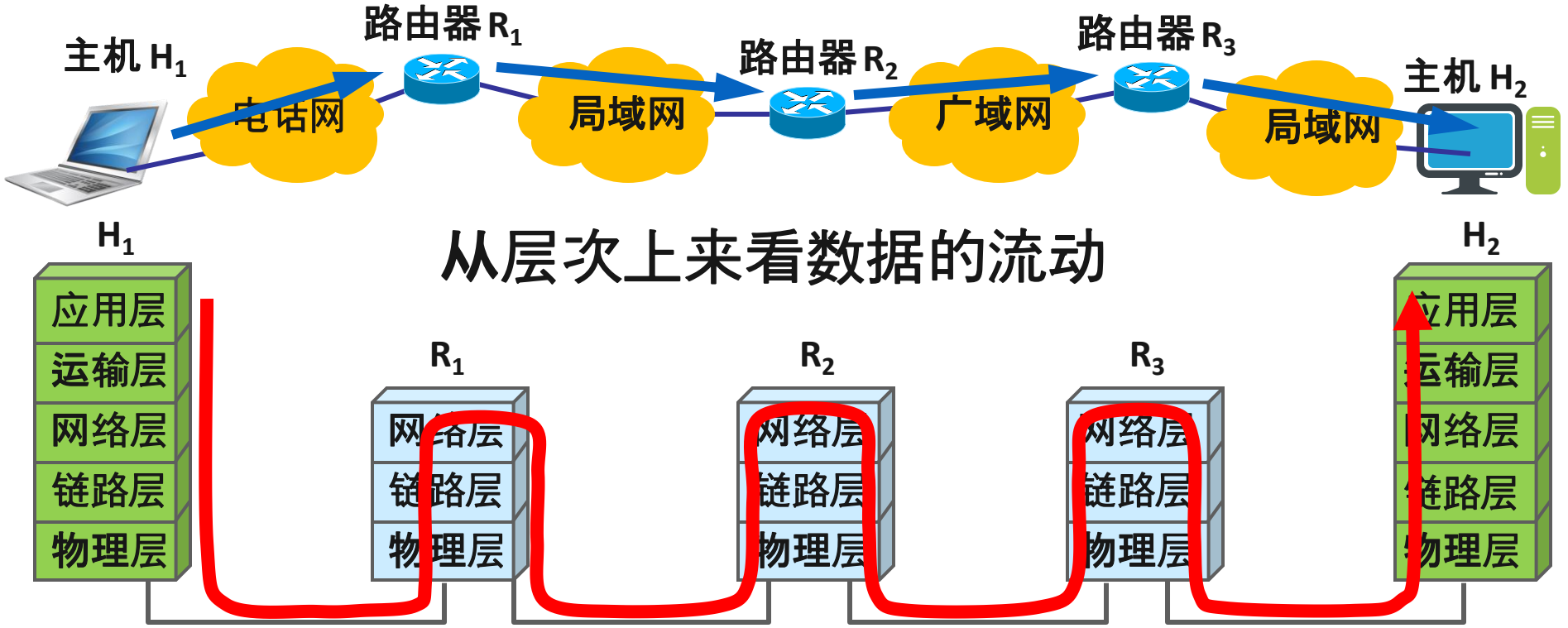
数据链路层使用的信道主要有以下两种类型：

点对点信道:这种信道使用一对一的点对点通信方式。

广播信道:这种信道使用一对多的广播通信方式，因此过程比较复杂。广播信道上连接的主机很多，因此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。

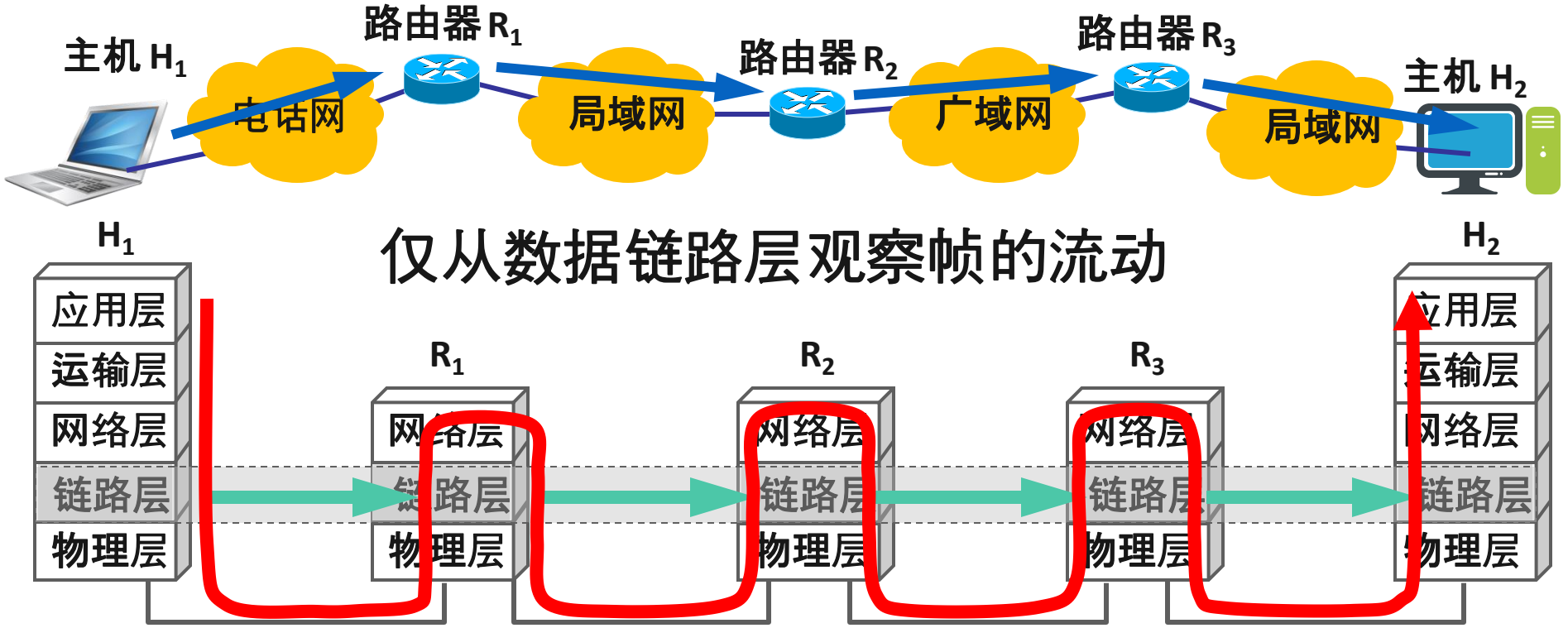
5.1.1 数据链路层的地位

主机 H_1 向 H_2 发送数据



5.1.1 数据链路层的地位

主机 H_1 向 H_2 发送数据

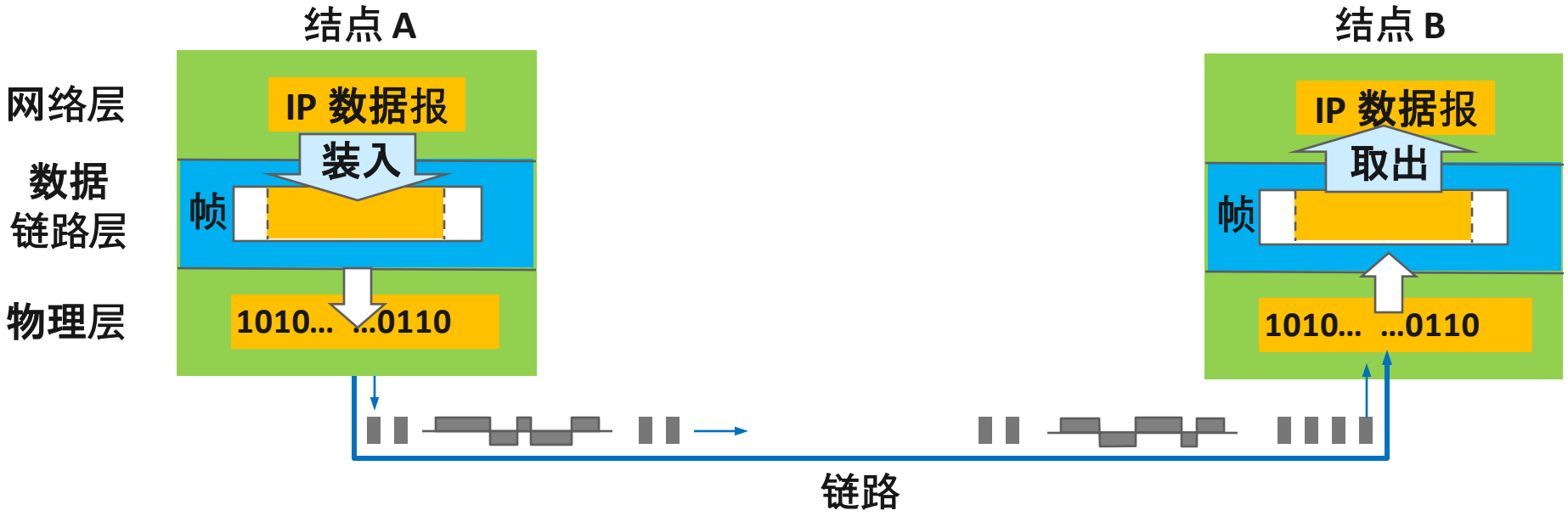


5.1.2 数据链路和帧

- 链路(link)是一条无源的点到点的物理线路段，中间没有任何其他的交换结点。
  - 一条链路只是一条通路的一个组成部分。
- 数据链路(data link) 除了物理线路外，还必须有通信协议来控制这些数据的传输。若把实现这些协议的硬件和软件加到链路上，就构成了数据链路。
  - 现在最常用的方法是使用适配器（即网卡）来实现这些协议的硬件和软件。
  - 一般的适配器都包括了数据链路层和物理层这两层的功能。

5.1.2 数据链路和帧

数据链路层传送的是帧

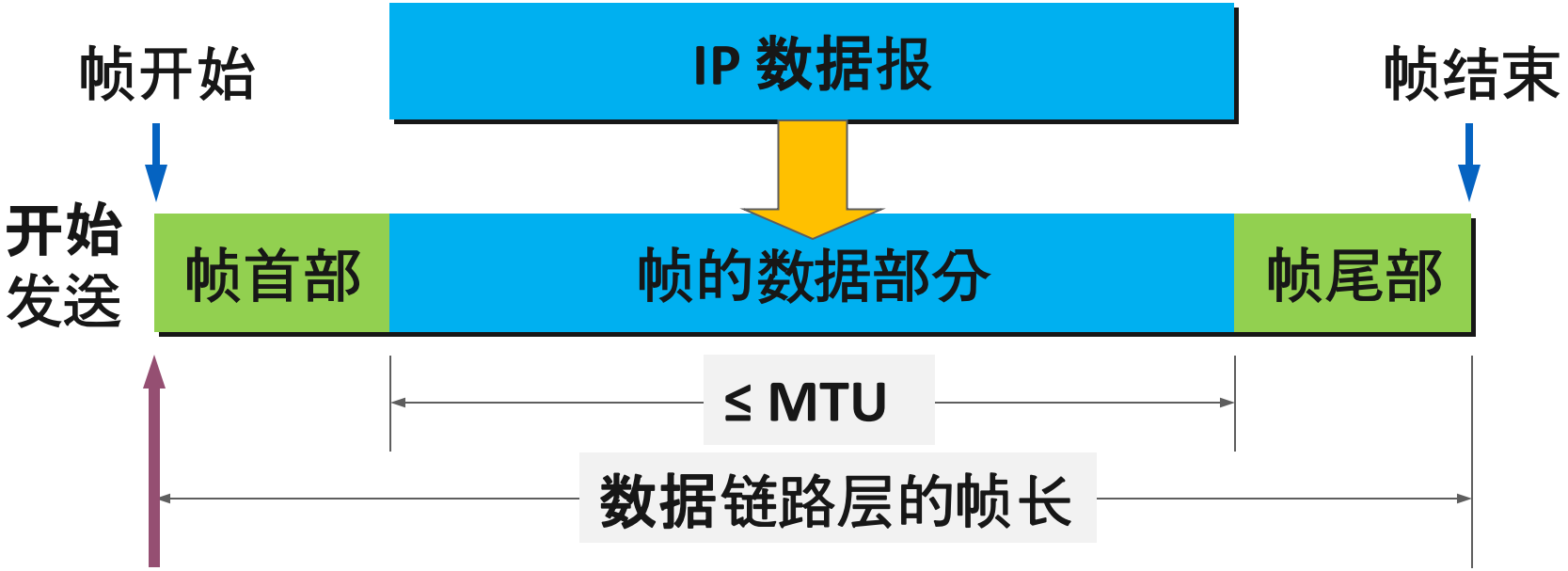


(a)

(b)

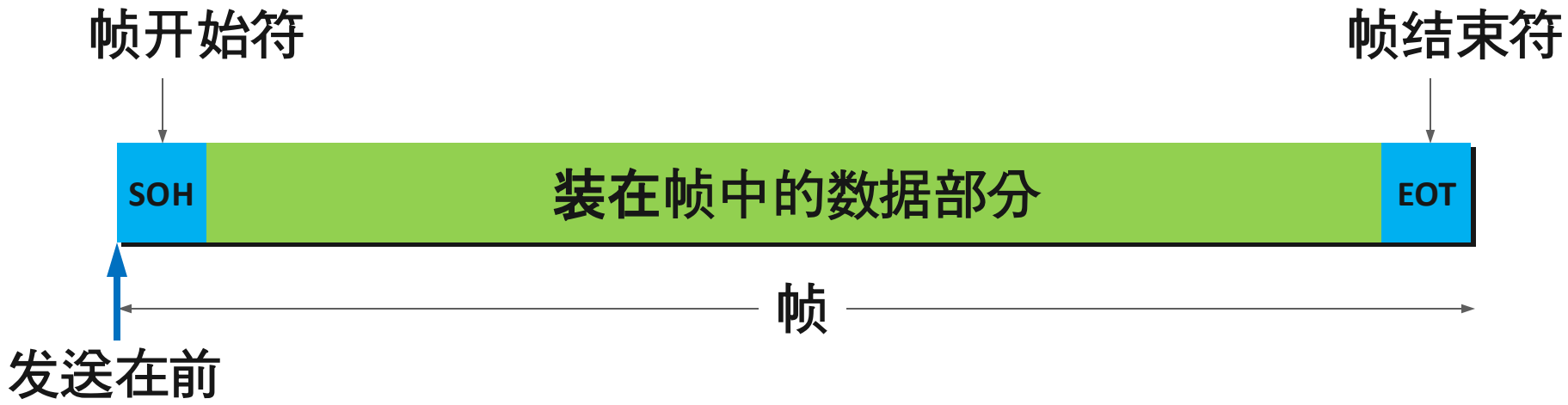
5.1.3 封装成帧

- 封装成帧(framing)就是在一段数据的前后分别添加首部和尾部，然后就构成了一个帧。确定帧的界限。
- 首部和尾部的一个重要作用就是进行帧定界。



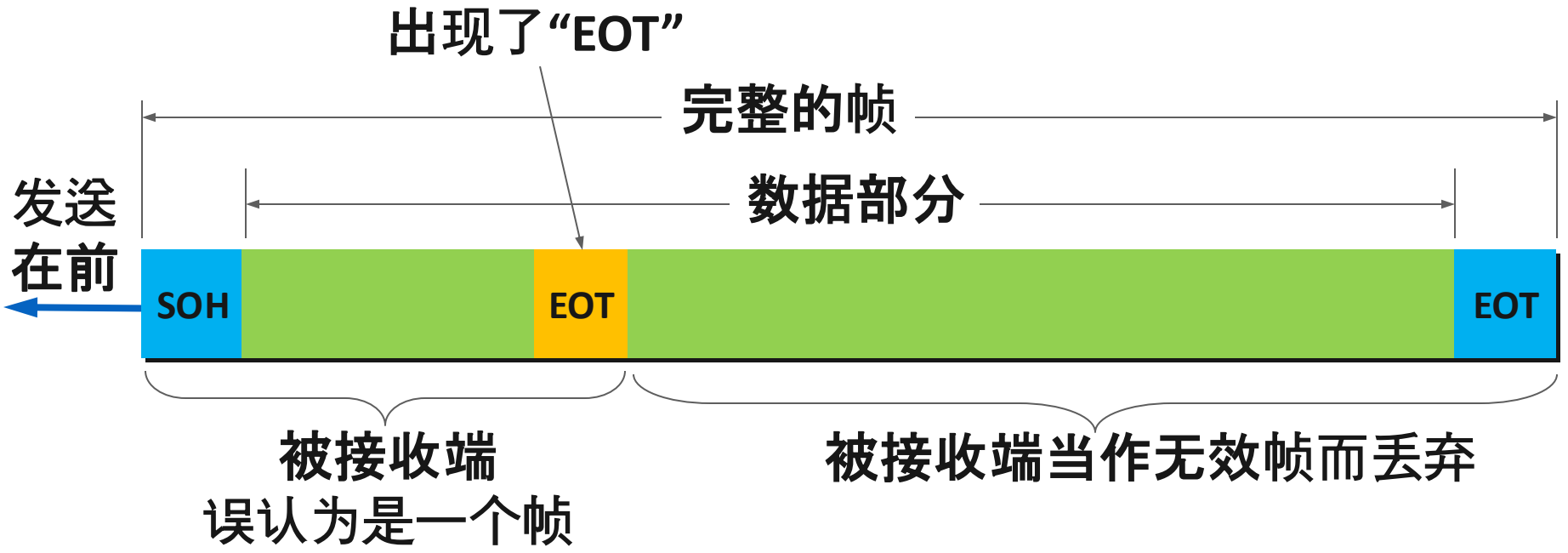
5.1.3　封装成帧

用**控制字符**进行帧定界的方法举例



5.1.3 封装成帧

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。



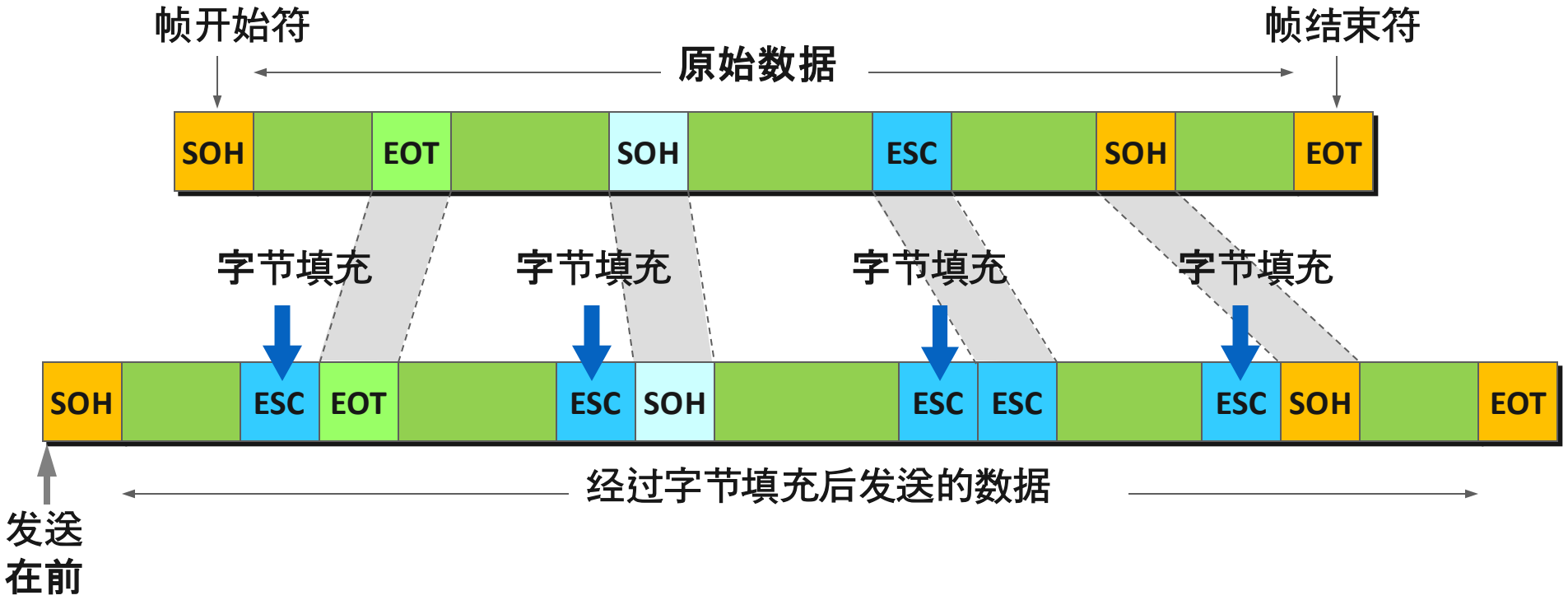
5.1.3 封装成帧

解决透明传输问题

- 发送端的数据链路层在数据中出现控制字符“SOH”或“EOT”的前面插入一个转义字符“ESC”(其十六进制编码是 1B)。
- 字节填充 (byte stuffing) （面向字符的物理链路）或字符填充(character stuffing)（面向比特的物理链路）——接收端的数据链路层在将数据送往网络层之前删除插入的转义字符。
- 如果转义字符也出现数据当中，那么应在转义字符前面插入一个转义字符。当接收端收到连续的两个转义字符时，就删除其中前面的一个。

5.1.3 封装成帧

用字节填充法解决透明传输的问题

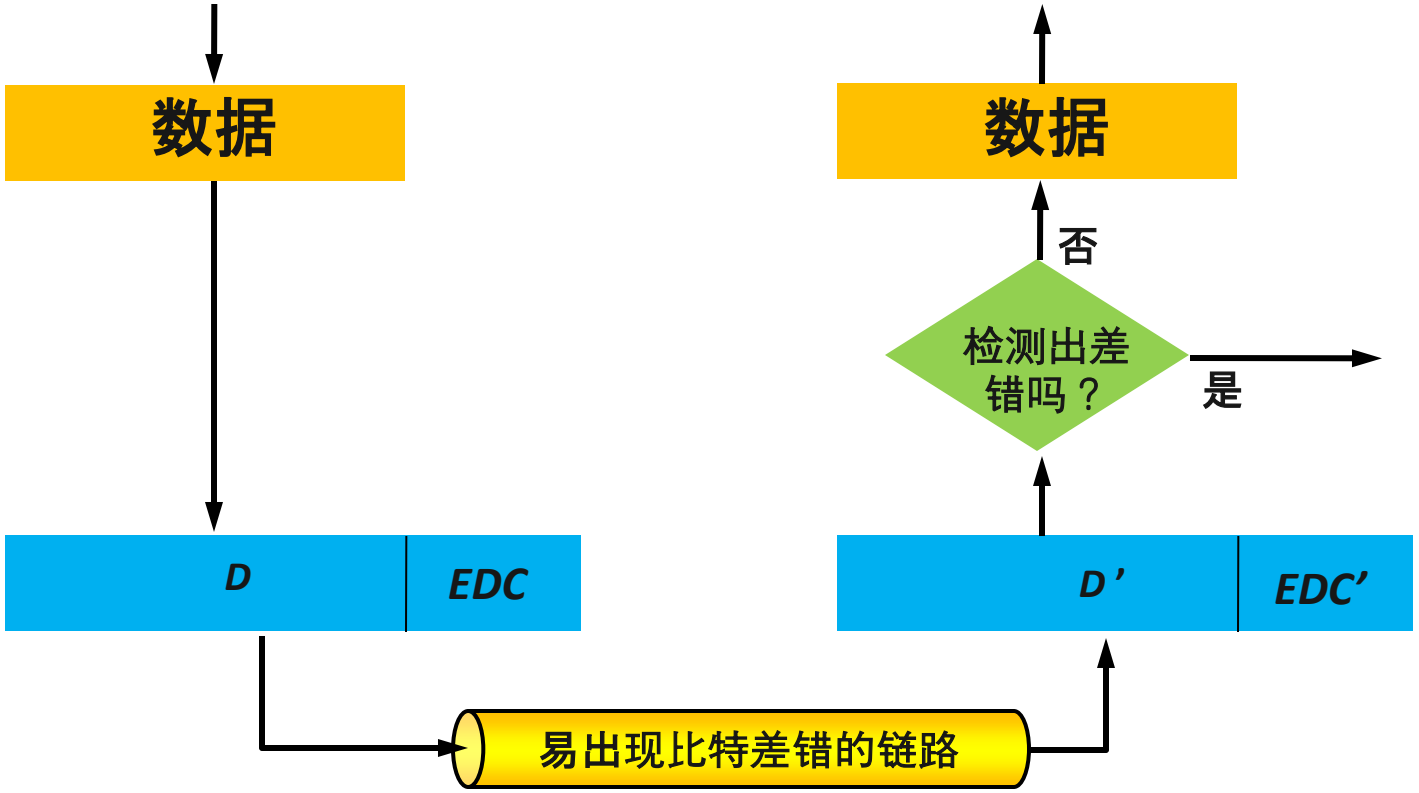


5.1.4 差错检测

- 在传输过程中可能会产生**比特差错**：1 可能会变成 0 而 0 也可能变成 1。
- 在一段时间内，传输错误的比特占所传输比特总数的比率称为**误码率** BER (Bit Error Rate)。
- 误码率与信噪比有很大的关系。
- 为了保证数据传输的可靠性，在计算机网络传输数据时，必须采用各种差错检测措施。

5.1.4 差错检测

差错检测的基本原理



1）发送方采用某种差错检测算法f,用发送的数据D计算出差错检测码EDC=f(D)；

2）将EDC随数据一起发送给接收方；

3）接收方通过同样的算法计算接收数据*D'*的差错检测码f（*D'*）

4）如果接收的差错检测码*EDC'*≠f（*D'*），则可判断传输的数据出现了差错。

5.1.4 差错检测

循环冗余检验的原理

- 在数据链路层传送的帧中，广泛使用了**循环冗余检验** CRC 的检错技术。
- 在发送端，先把数据划分为组。假定每组 k 个比特。
- 假设待传送的一组数据 $M = 101001$（现在 $k = 6$）。我们在 M 的后面再添加供差错检测用的 n 位**冗余码**一起发送。

5.1.4 差错检测

冗余码的计算

- 用二进制的模 2 运算进行 2^n 乘 M 的运算，这相当于在 M 后面添加 n 个 0。
- 得到的 $(k+n)$ 位的数除以事先选定好的长度为 $(n+1)$ 位的除数 P，得出商是 Q 而余数是 R，余数 R 比除数 P 少1 位，即 R 是 n 位。

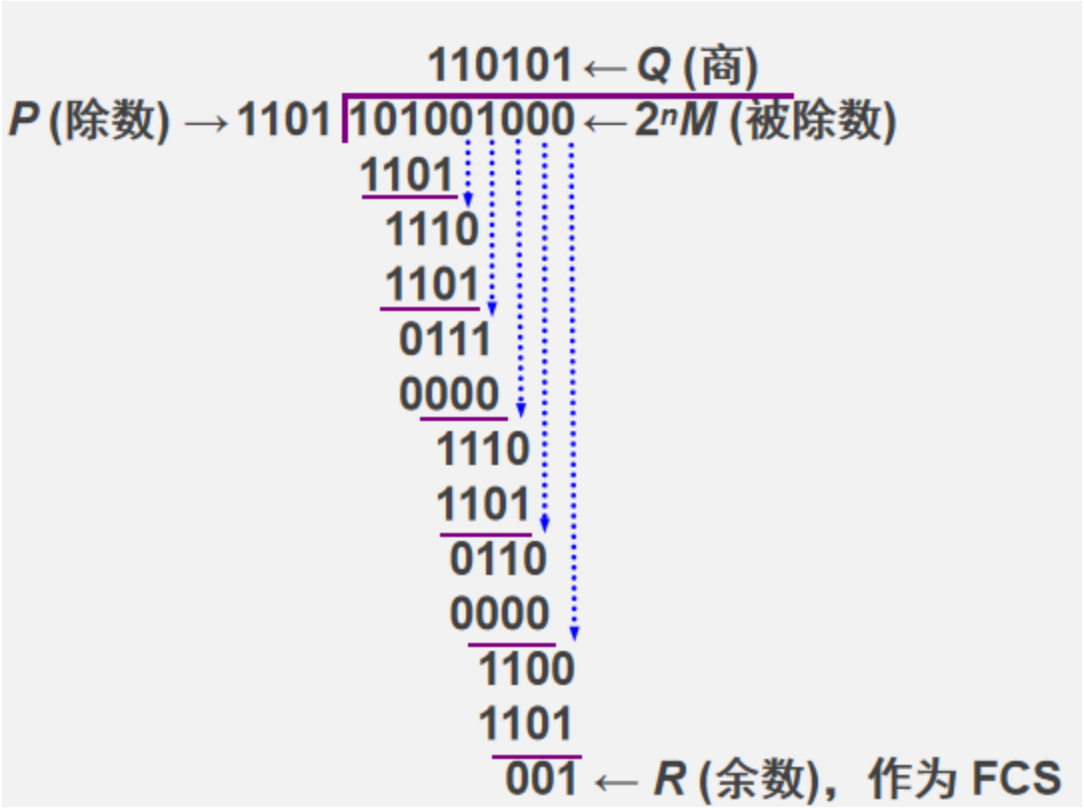
5.1.4 差错检测

冗余码的计算举例

- 现在 $k = 6$, $M = 101001$。
- 设 $n = 3$, 除数 $P = 1101$,
- 被除数是 $2^nM = 101001000$。
- 模 2 运算的结果是：商 $Q = 110101$,

  余数 $R = 001$。
- 把余数 R 作为冗余码添加在数据 M 的后面发送出去。发送的数据是：$2^nM + R$

  即：101001001，共 $(k + n)$ 位。



FCS是帧检验序列

5.1.4 差错检测

帧检验序列 FCS

- 在数据后面添加上的冗余码称为帧检验序列 FCS (Frame Check Sequence)。
- 循环冗余检验 CRC 和帧检验序列 FCS并不等同。
  - CRC 是一种常用的检错方法，而 FCS 是添加在数据后面的冗余码。
  - FCS 可以用 CRC 这种方法得出，但 CRC 并非用来获得 FCS 的唯一方法。

5.1.4 差错检测

接收端对收到的每一帧进行 CRC 检验

(1) 若得出的余数 $R = 0$，则判定这个帧没有差错，就接受(accept)。

(2) 若余数 $R \neq 0$，则判定这个帧有差错，就丢弃。

- 但这种检测方法并不能确定究竟是哪一个或哪几个比特出现了差错。
- 只要经过严格的挑选，并使用位数足够多的除数 P，那么出现检测不到的差错的概率就很小很小。

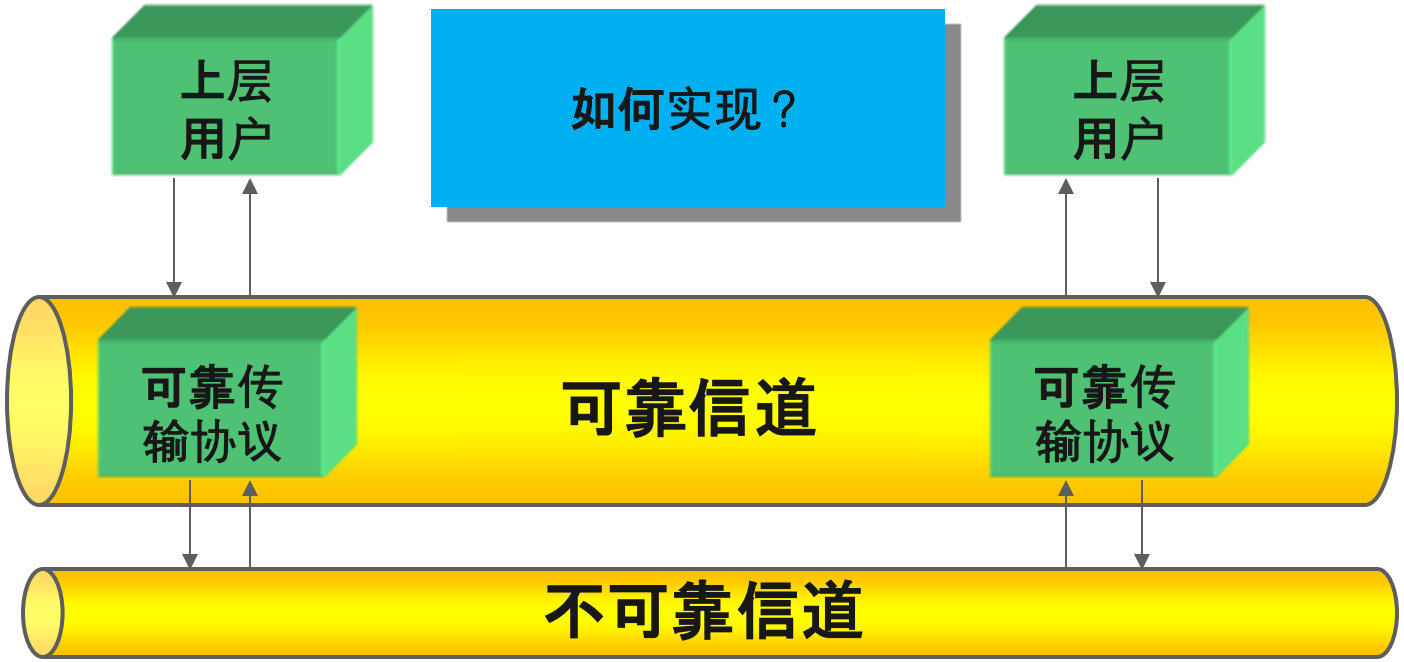
5.1.4 差错检测

应当注意

- 仅用循环冗余检验 CRC 差错检测技术只能做到无差错**接受**(accept)。
- “无差错接受”是指：“凡是接受的帧（即**不包括丢弃的帧**），我们都能以非常接近于 1 的概率认为这些帧在传输过程中没有产生差错”。
- 也就是说：“凡是接收端数据链路层接受的帧都没有传输差错”（有差错的帧就丢弃而不接受）。
- 要做到“**可靠传输**”（即发送什么就收到什么）就必须再加上下一节要讲的**确认**和**重传**机制。

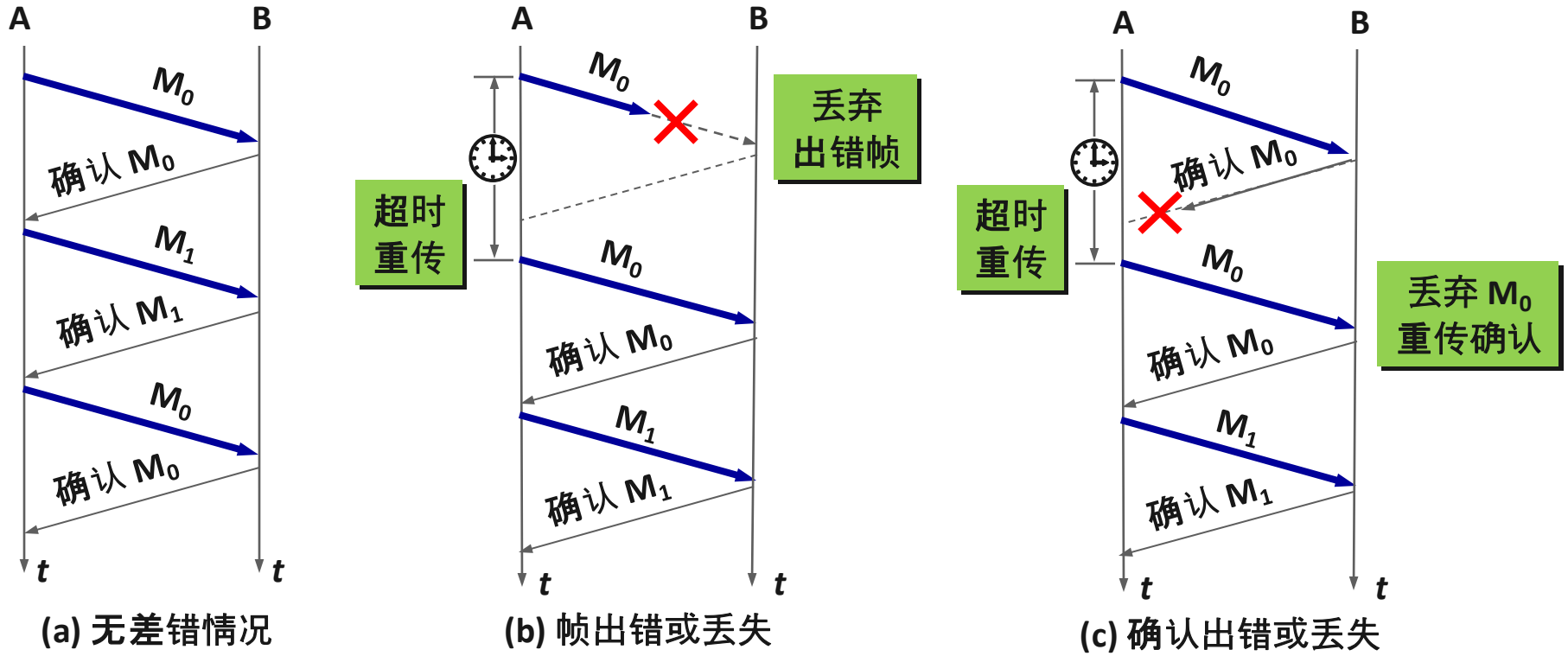
5.1.5 可靠传输

在不可靠的信道上实现可靠的数据传输，为上层提供一条可靠的逻辑通道。



5.1.5 可靠传输

1）停止等待协议



(a) 无差错情况　(b) 帧出错或丢失　(c) 确认出错或丢失

5.1.5 可靠传输

- 在发送完一个帧后，必须暂时保留已发送的帧的副本。
- 数据帧和确认帧都必须进行编号。
- 只要超过了一段时间还没有收到确认，就认为已发送的帧出错或丢失了，因而重传已发送过的帧。这就叫做**超时重传**。
- 超时计时器的重传时间应当比数据在分组传输的平均往返时间更长一些。

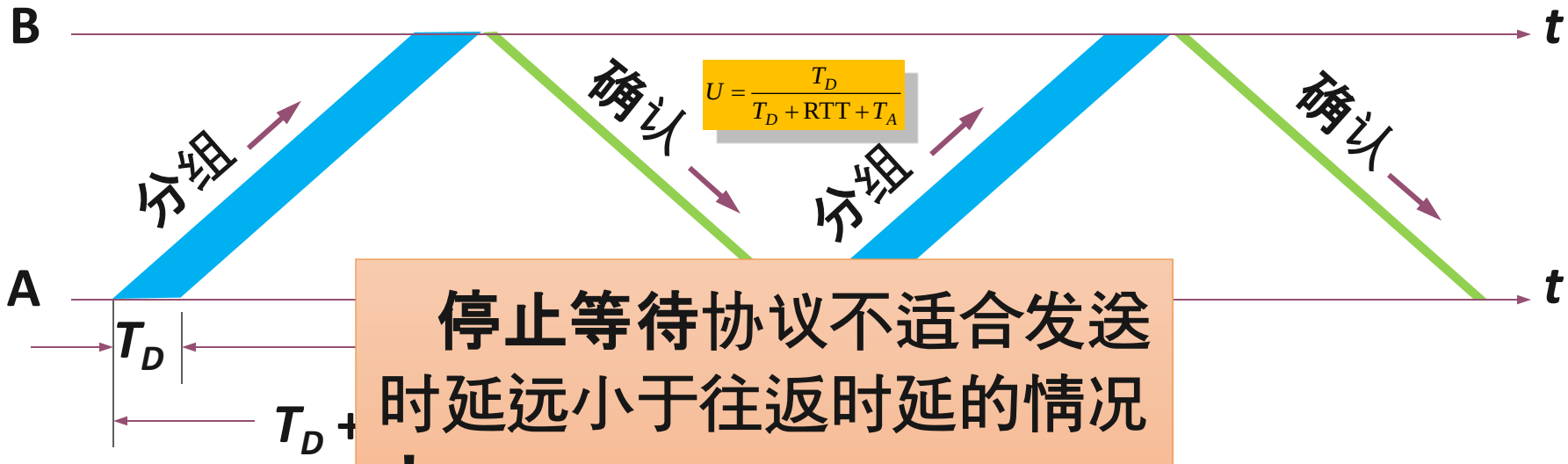
5.1.5 可靠传输

2） 自动重传请求ARQ

- 使用上述的确认和重传机制，我们就可以在不可靠的传输网络上实现可靠的通信。
- 这种可靠传输协议常称为自动重传请求ARQ (Automatic Repeat reQuest)。
- ARQ 表明重传的请求是自动进行的。接收方不需要请求发送方重传某个出错的分组 。

5.1.5 可靠传输

3）信道利用率



RTT(Round-Trip Time): 往返时延

T_D：A发送分组需要的时间

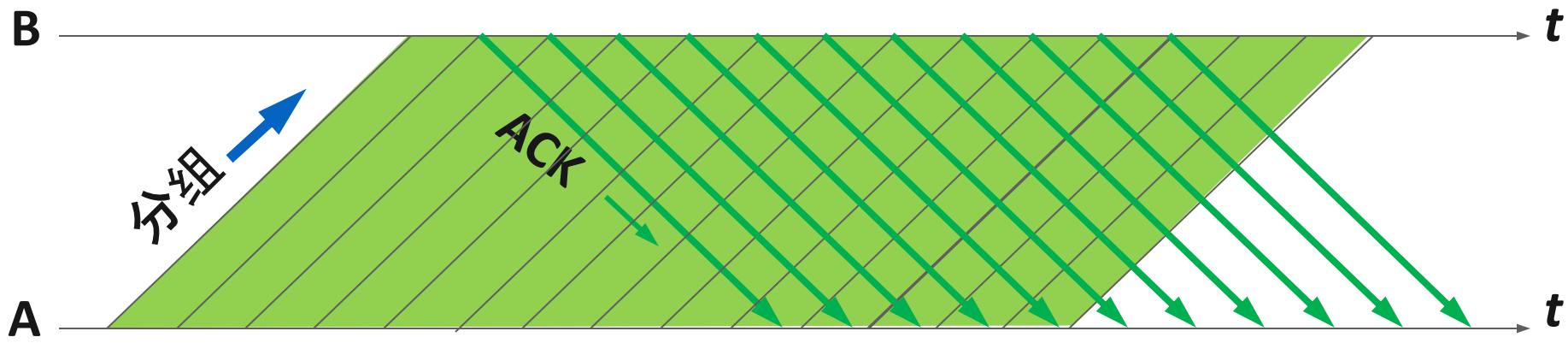
T_A：B发送确认分组需要的时间

停止等待协议的优点是简单，但缺点是信道利用率低。

5.1.5 可靠传输

连续ARQ: 流水线传输

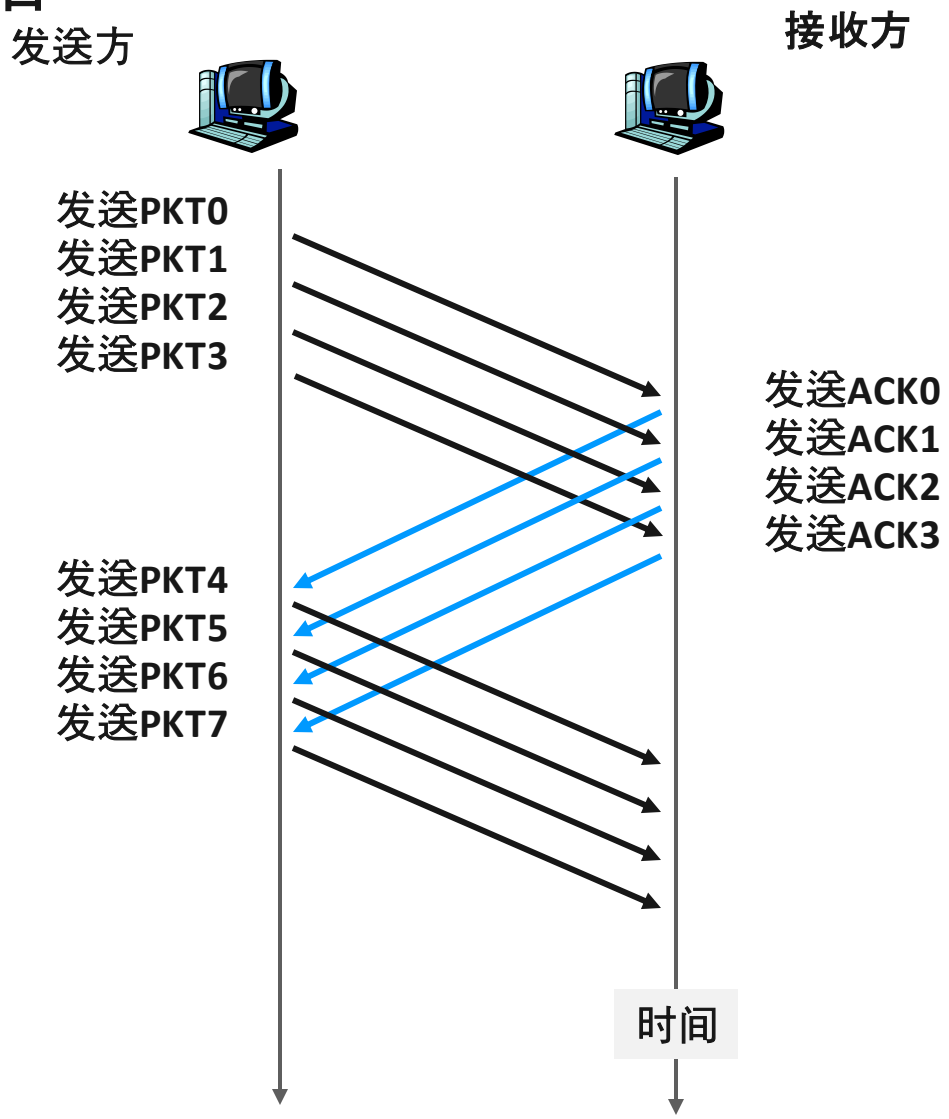
- 发送方可连续发送多个分组，不必每发完一个分组就停顿下来等待对方的确认。
- 由于信道上一直有数据不间断地传送，这种传输方式可获得很高的信道利用率。



连续不间断发送数据可能导致接收方或网络来不及处理

5.1.5 可靠传输

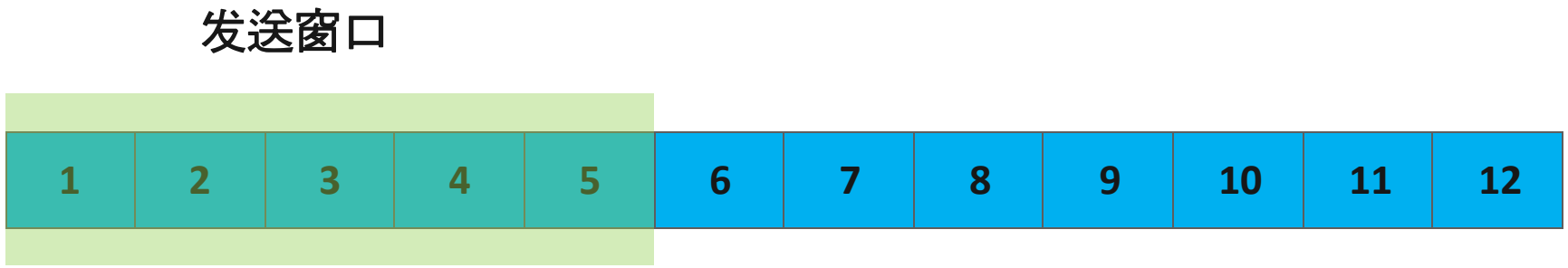
限制连续发送分组的数目



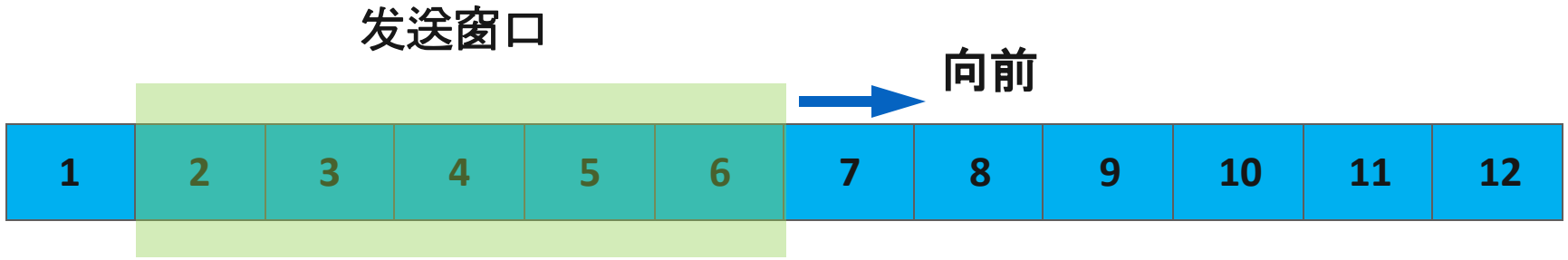
5.1.5 可靠传输

滑动窗口

通过设置发送窗口来限制发送方的发送速率



(a) 发送方维持发送窗口（发送窗口是 5）

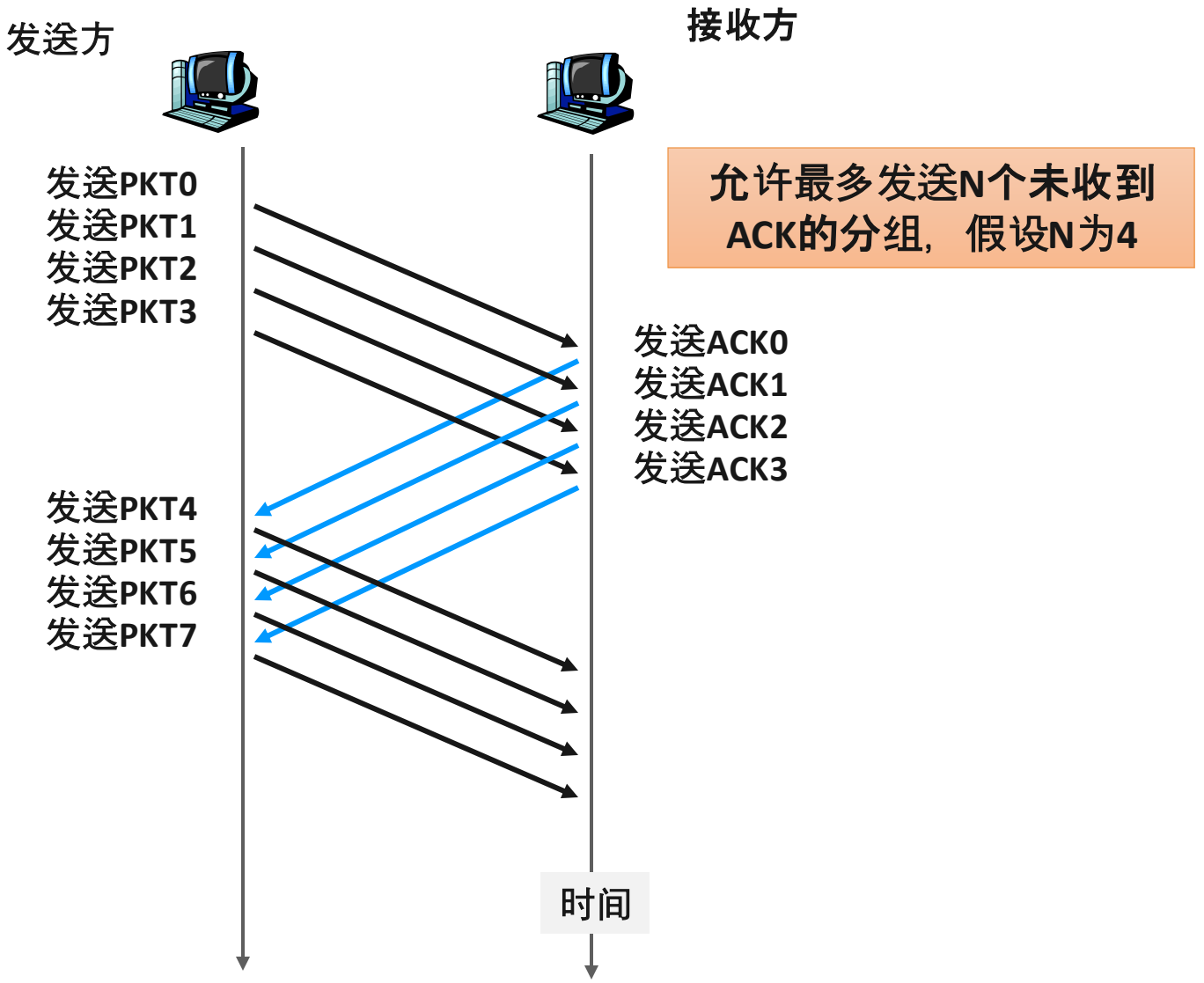


(b) 收到一个确认后发送窗口向前滑动

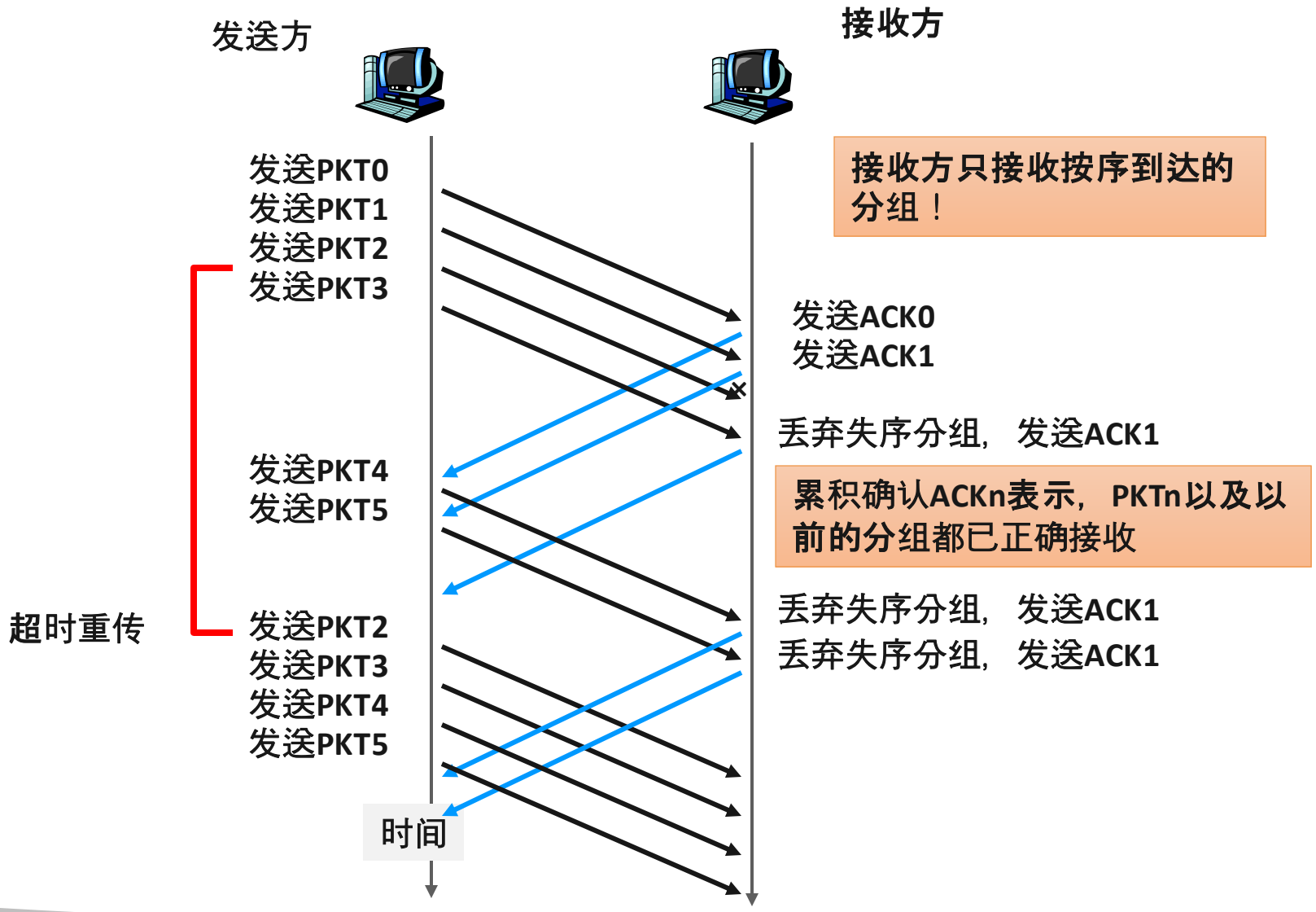
发送窗口大小是已发送但还没有收到确认的最大分组数

5.1.5 可靠传输

滑动窗口的作用



5.1.5 可靠传输


发送方
接收方
发送PKT0
发送PKT1
发送PKT2
发送PKT3
接收方只接收按序到达的分组！
发送ACK0
发送ACK1
丢弃失序分组，发送ACK1
发送PKT4
发送PKT5
累积确认ACKn表示，PKTn以及以前的分组都已正确接收
超时重传
发送PKT2
发送PKT3
发送PKT4
发送PKT5
丢弃失序分组，发送ACK1
丢弃失序分组，发送ACK1
时间

5.1.5 可靠传输

4) Go-back-N（回退 N）协议

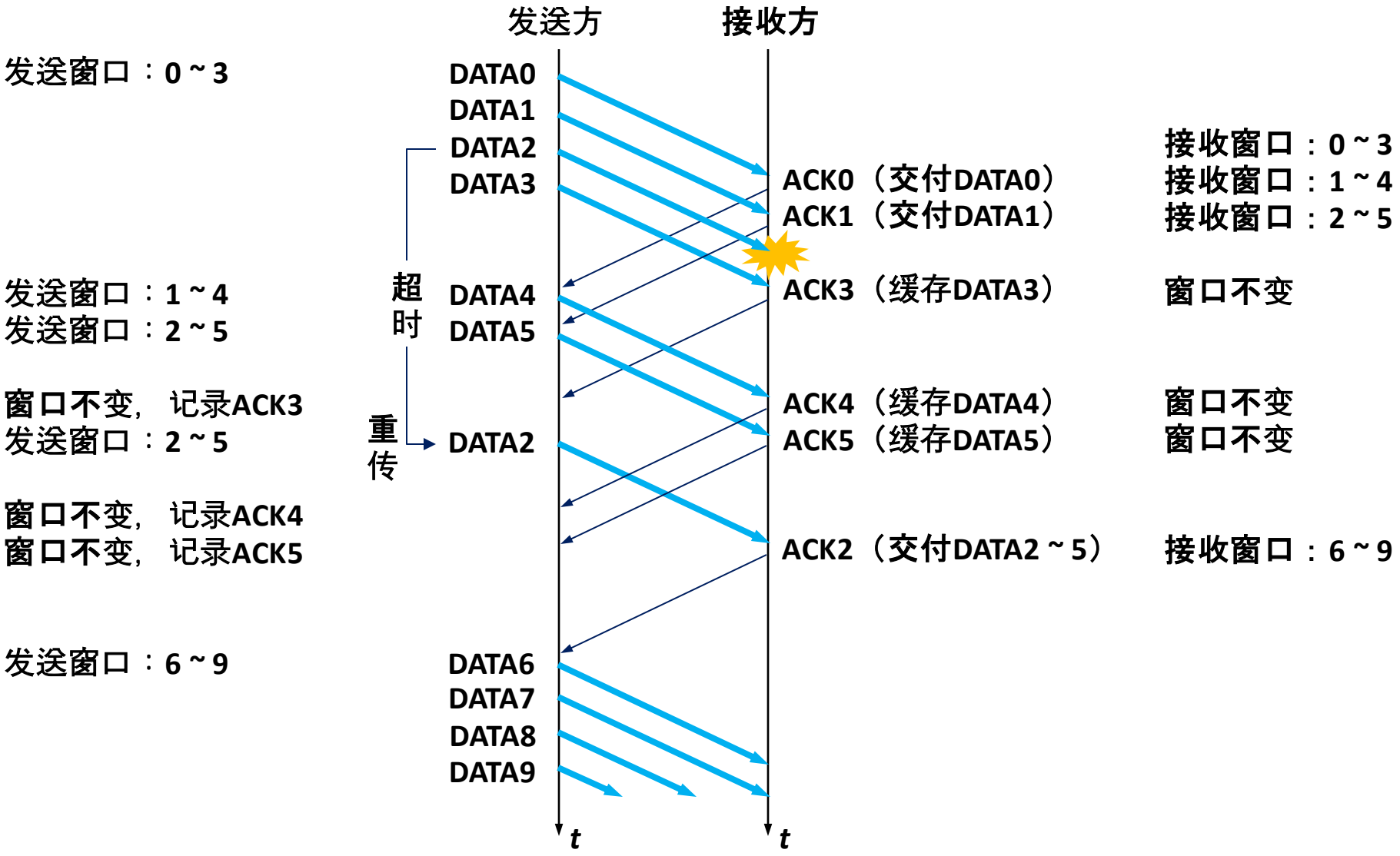
- 如果发送方发送了前 5 个分组，而中间的第 3 个分组丢失了。这时接收方只能对前两个分组发出确认。发送方无法知道后面三个分组的下落，而只好把后面的三个分组都再重传一次。
- 这就叫做 Go-back-N（回退 N），表示需要再退回来重传已发送过的 *N* 个分组。

5.1.5 可靠传输

5）选择重传

- GBN协议存在一个缺点：一个分组的差错可能引起大量分组的重传，这些分组可能已经被接收方正确接收了，但由于未按序到达而被丢弃。
- 可设法只重传出现差错的分组。但必须加大接收窗口，以便先收下失序到达但仍然处在接收窗口中的哪些分组，等到所缺分组收齐后再一并送交上层。这就是选择重传SR(Selective Repeat)协议。

5.1.5 可靠传输


发送方
接收方
发送窗口：0 ~ 3
DATA0
DATA1
DATA2
DATA3
ACK0（交付DATA0）
ACK1（交付DATA1）
接收窗口：0 ~ 3
接收窗口：1 ~ 4
接收窗口：2 ~ 5
超时
发送窗口：1 ~ 4
发送窗口：2 ~ 5
DATA4
DATA5
ACK3（缓存DATA3）
窗口不变
窗口不变，记录ACK3
发送窗口：2 ~ 5
重传
DATA2
ACK4（缓存DATA4）
ACK5（缓存DATA5）
窗口不变
窗口不变
窗口不变，记录ACK4
窗口不变，记录ACK5
ACK2（交付DATA2 ~ 5）
接收窗口：6 ~ 9
发送窗口：6 ~ 9
DATA6
DATA7
DATA8
DATA9
t
t

5.1.5 可靠传输

6) 数据链路层的可靠传输

- 实现可靠传输需要付出代价（例如会降低传输效率）。
- 因此，应当根据链路的具体情况来决定是否需要让链路层向上提供可靠传输服务。
- 当链路误码率非常低时，在数据链路层可不实现可靠传输，而是由上层协议（例如，运输层的TCP协议）来完成。
- 但是在使用无线信道传输数据时，由于信道质量较差，在数据链路层仍需要实现可靠传输（例如使用停止等待协议）。

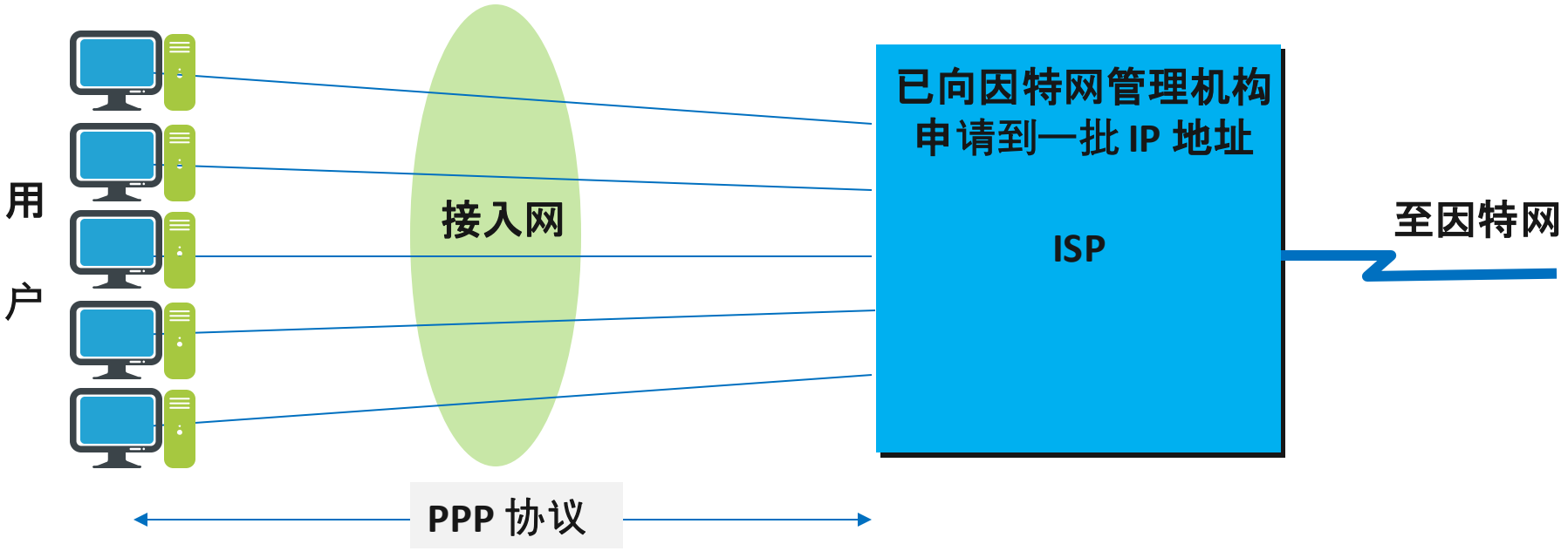
第二部分

数据链路层的协议

5.2.1 PPP协议的特点

- 现在全世界使用得最多的点对点数据链路层协议是**点对点协议** PPP (Point-to-Point Protocol)。
- 用户使用拨号电话线接入因特网时，一般都是使用 PPP 协议。

用户到 ISP 的链路使用 PPP 协议



5.2.1 PPP协议的特点

1)简单——这是首要要求

2）封装成帧

3）透明性

4）多种网络层协议

5）多种类型链路

6）差错检测

7）检测连接状态

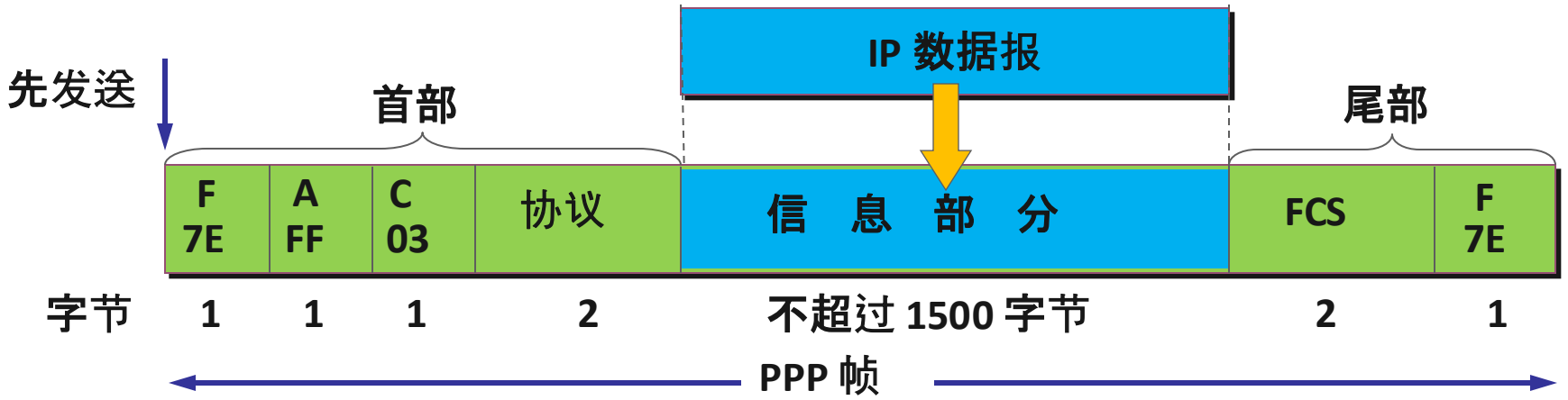
5.2.2 PPP协议的组成

- PPP 协议有三个组成部分
  - 一个将 IP 数据报封装到串行链路的方法。
  - 链路控制协议 LCP (Link Control Protocol)。
  - 网络控制协议 NCP (Network Control Protocol)。

5.2.3 PPP协议的帧格式

- 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。

5.2.3 PPP协议的帧格式



- PPP 有一个 2 个字节的协议字段。
  - 当协议字段为 0x0021 时， PPP 帧的信息字段就是IP 数据报。
  - 若为 0xC021, 则信息字段是 PPP 链路控制数据。
  - 若为 0x8021， 则表示这是网络控制数据。

5.2.3 PPP协议的帧格式

透明传输问题

- 当 PPP 用在同步传输链路时，协议规定采用硬件来完成比特填充（和 HDLC 的做法一样）。
- 当 PPP 用在异步传输时，就使用一种特殊的**字符填充法**。

5.2.3 PPP协议的帧格式

字符填充

- 将信息字段中出现的每一个 0x7E 字节转变成为 2 字节序列(0x7D, 0x5E)。
- 若信息字段中出现一个 0x7D 的字节, 则将其转变成为 2 字节序列(0x7D, 0x5D)。
- 若信息字段中出现 ASCII 码的控制字符（即数值小于 0x20 的字符）， 则在该字符前面要加入一个 0x7D 字节， 同时将该字符的编码加以改变。

5.2.3 PPP协议的帧格式

零比特填充

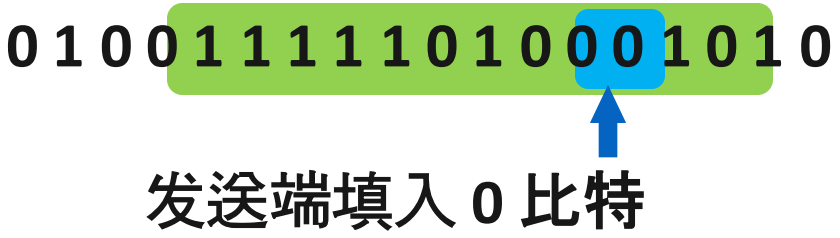
- PPP 协议用在 SONET/SDH 链路时，是使用同步传输（一连串的比特连续传送）。这时 PPP 协议采用零比特填充方法来实现透明传输。
- 在发送端，只要发现有 5 个连续 1，则立即填入一个 0。接收端对帧中的比特流进行扫描。每当发现 5 个连续1时，就把这 5 个连续 1 后的一个 0 删除，

5.2.3 PPP协议的帧格式

零比特填充

信息字段中出现了和标志字段 F 完全一样的 8 比特组合

发送端在 5 个连 1 之后填入 0 比特再发送出去



在接收端把 5 个连 1 之后的 0 比特删除

5.2.4 PPP的工作状态

- 当用户拨号接入 ISP 时，路由器的调制解调器对拨号做出确认，并建立一条物理连接。
- PC 机向路由器发送一系列的 LCP 分组（封装成多个 PPP 帧）。
- 这些分组及其响应选择一些 PPP 参数，和进行网络层配置，NCP 给新接入的 PC 机分配一个临时的 IP 地址，使 PC 机成为因特网上的一个主机。
- 通信完毕时，NCP 释放网络层连接，收回原来分配出去的 IP 地址。接着，LCP 释放数据链路层连接。最后释放的是物理层的连接。

5.2.4 PPP的工作状态