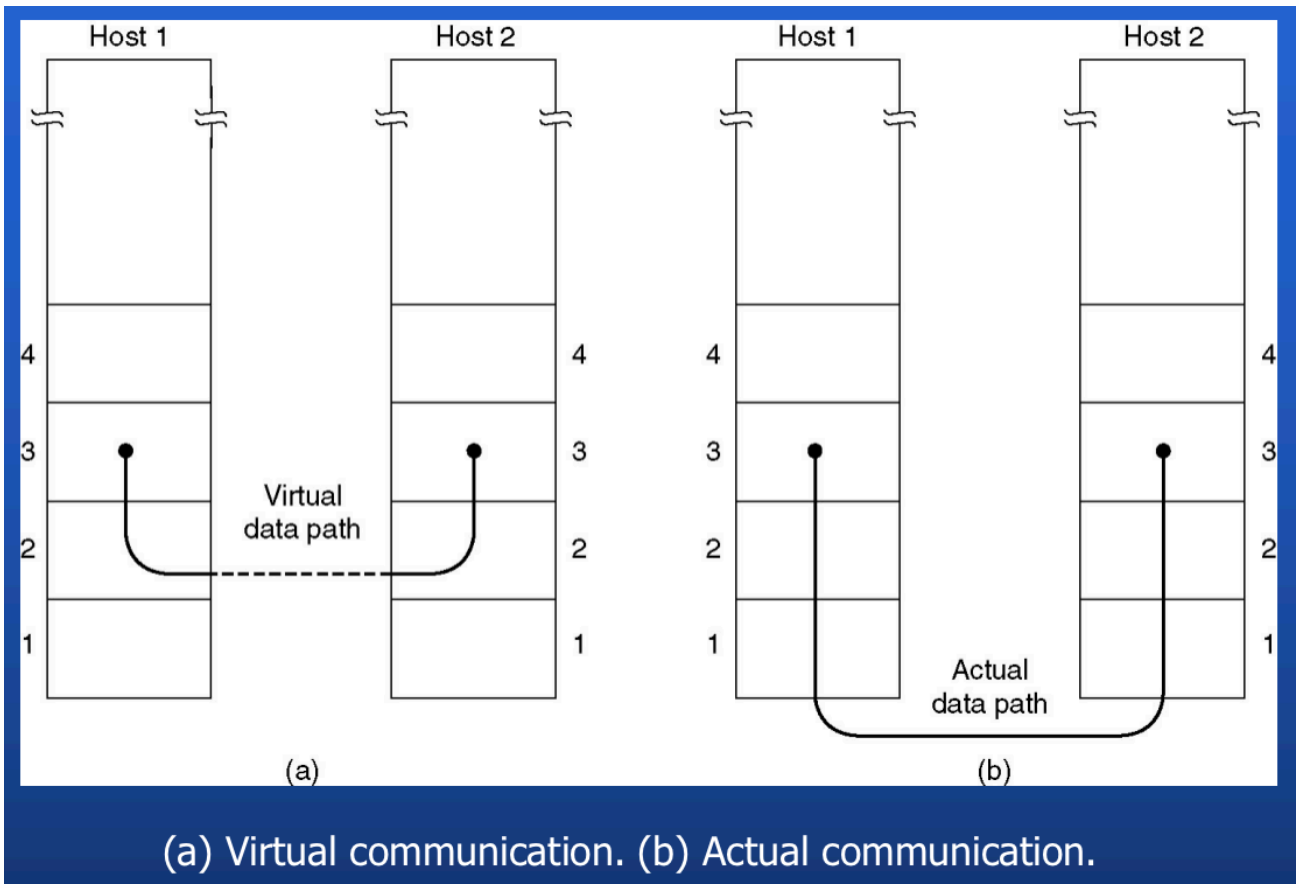
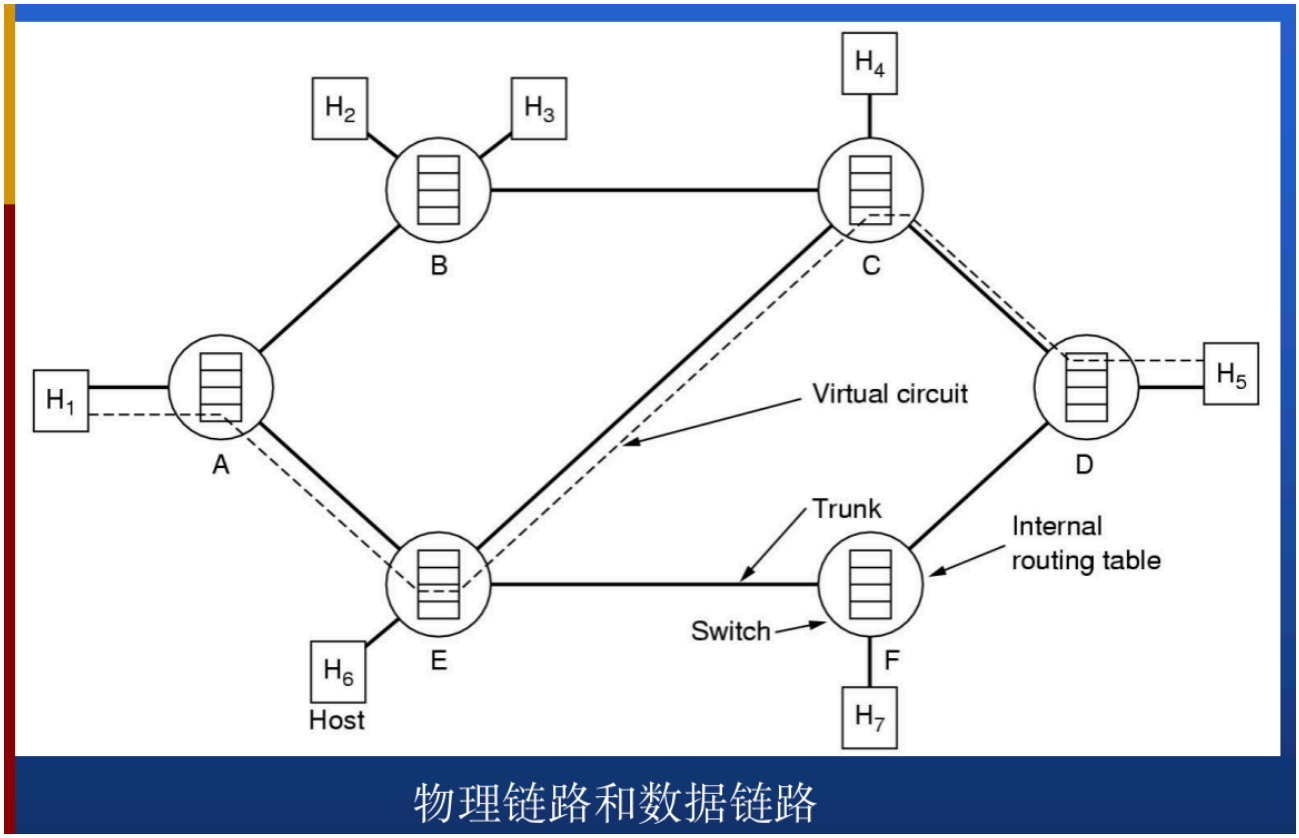
1 Data Link Layer Issues

- Services provided to the network layer
- framing,frame--- DATA LINK LAYER PDU(Protocol Data Unit)
- Error control
- Flow control



(a) Virtual communication. (b) Actual communication.



> 广播式链路（广播式信道)：WIFI、同位电缆
>
> 点对点链路

- 数据链路=物理链路+传输协议

1.1 Data Link Layer Design Issues

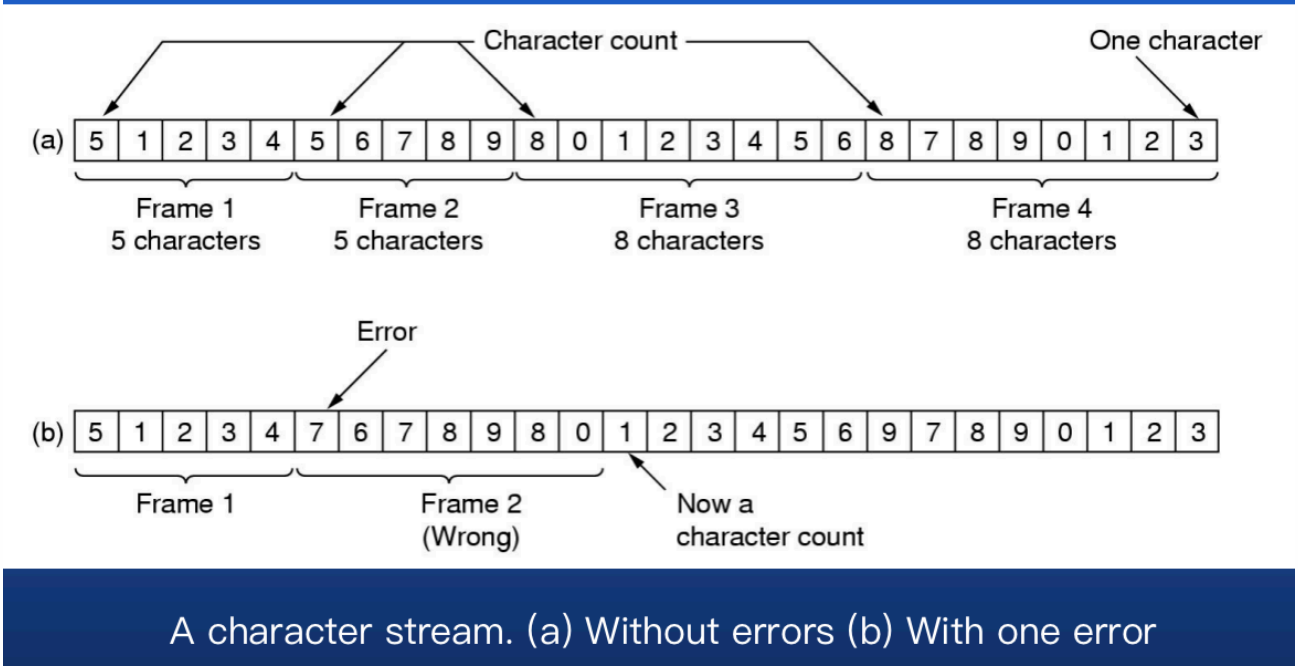
1.1.1 Services provided to the network layer

- Unacknowledged connectionless service 无应答的无连接服务
- Acknowledged connectionless service 有应答的无连接服务（发送简短的包，数据量少，发一次等一次应答）
- Acknowledged connection-oriented service 有应答的连接服务（大量数据传输，可以一次发很多，然后再等）

1.1.2 Framing 确定数据分割

1.1.2.1 character count

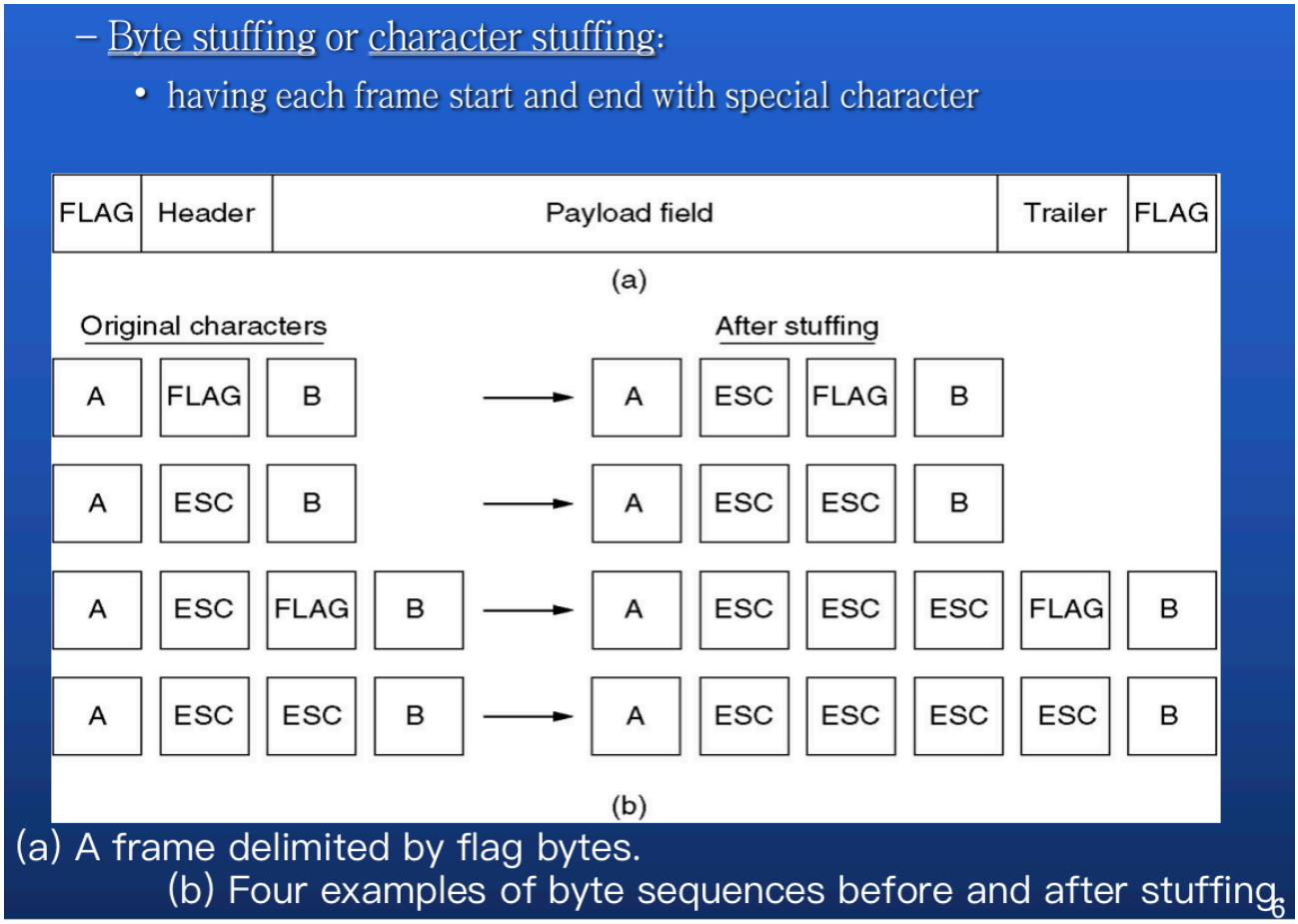
- framing
  - character count -[see fig 3-4]



A character stream. (a) Without errors (b) With one error

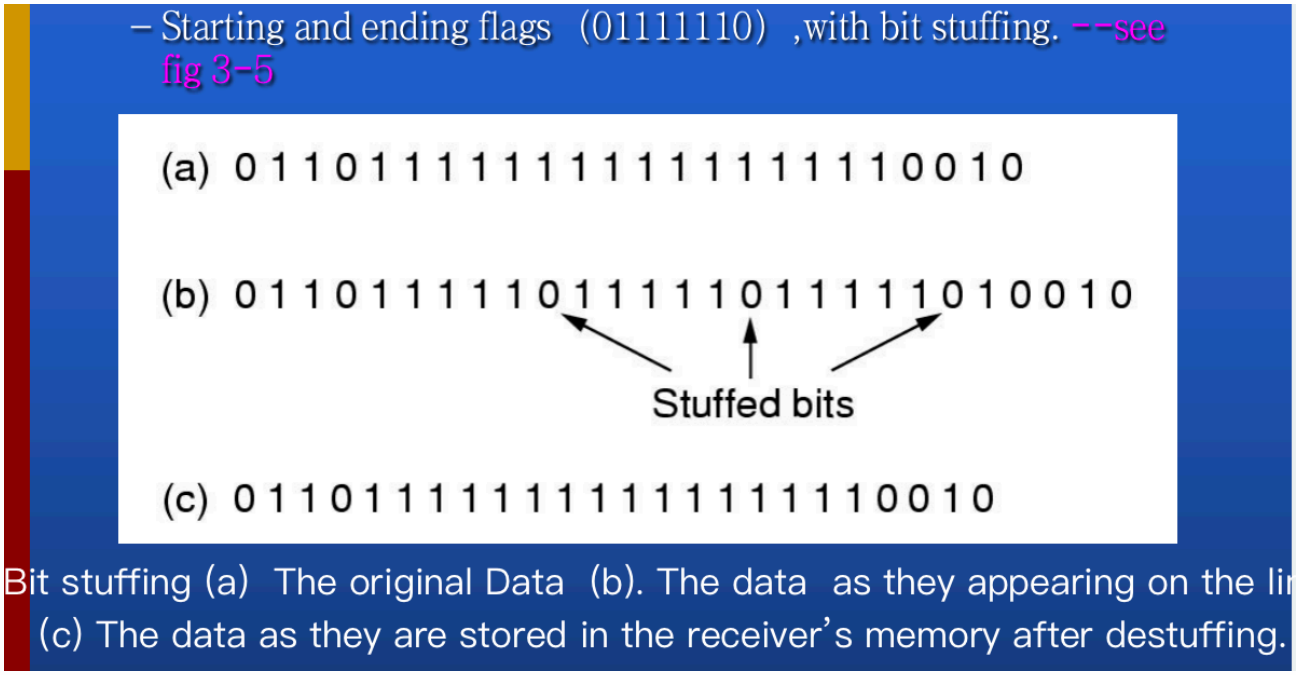
- a中，就是利用长度来确定边界，但是可能会存在b中这样丢包的现象，导致出错

1.1.2.2 Character Stuffing 字符填充法



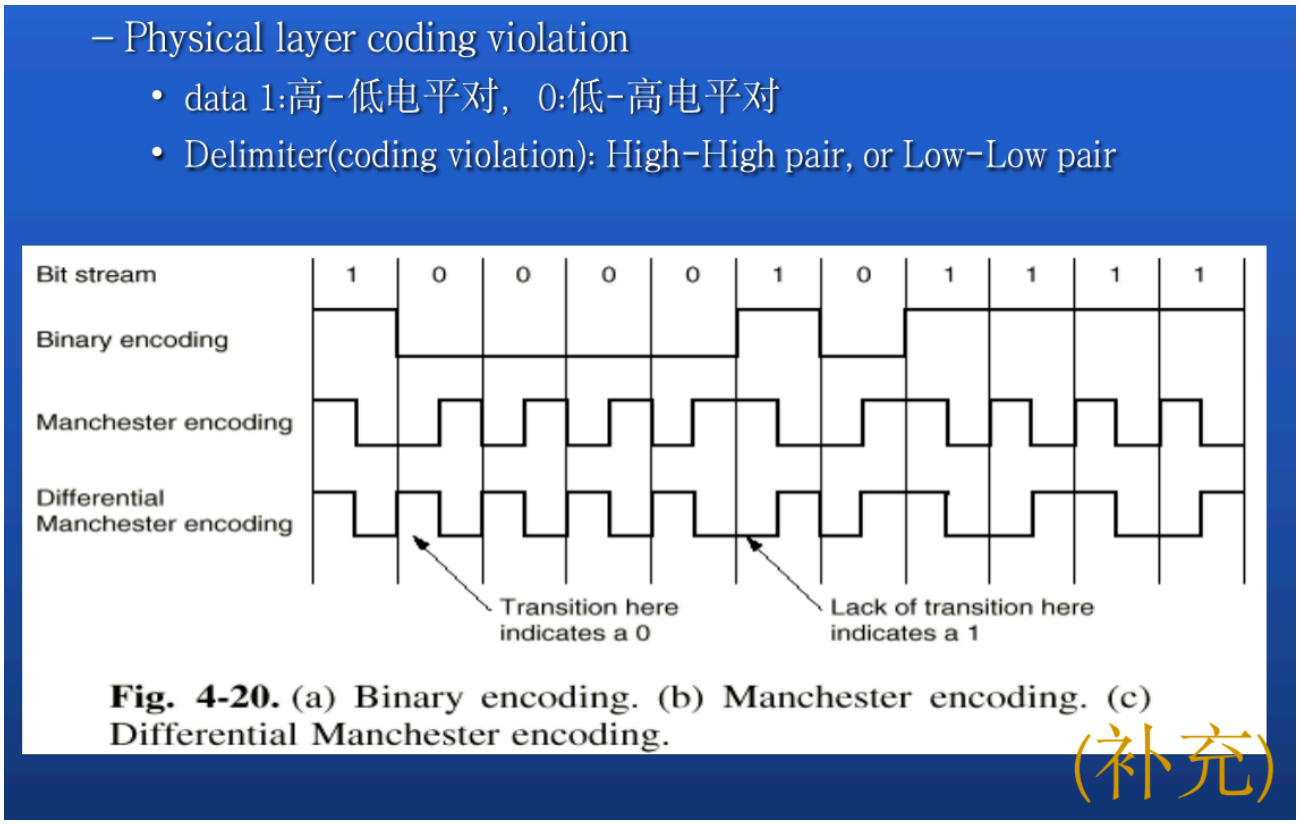
- 设定一个flag来作为开始和结束的标记
- 如果中间想发送flag，那就发送 “ESC + FLAG”，相当于转义符。但也有可能存在Trailer里存在ESC，导致FLAG被当成数据。

1.1.2.3 Bit Stuffing



- 比如FLAG是0x71（01111110）
- 那么五个1插入一个0，防止当成FLAG
- 如果五个1后面已经有0了，也要插，反正就是要满足上述的规定

1.1.2.4 编码传输



- 曼彻斯特编码，每个bit必须有跳变。如果有一段一直空着不变，意思是到尾了。

1.1.3 Error Control

- 丢帧
- 重复帧
- 错误帧

1.1.4 Flow Control

- to solve the problem that a sender wants to transmit frames faster than the receiver can accept them.
  - 发送方发的太快，快于接收方
- Two approaches
  - feedback_based flow control: receiver gives sender permission
    - 发送rr，说明可以接收，继续发；发送rn，说明缓存已满，不能发。
  - rate_based flow control: limit the data rate at witch the sender may transmit data.
    - 控制发送方的发送速率

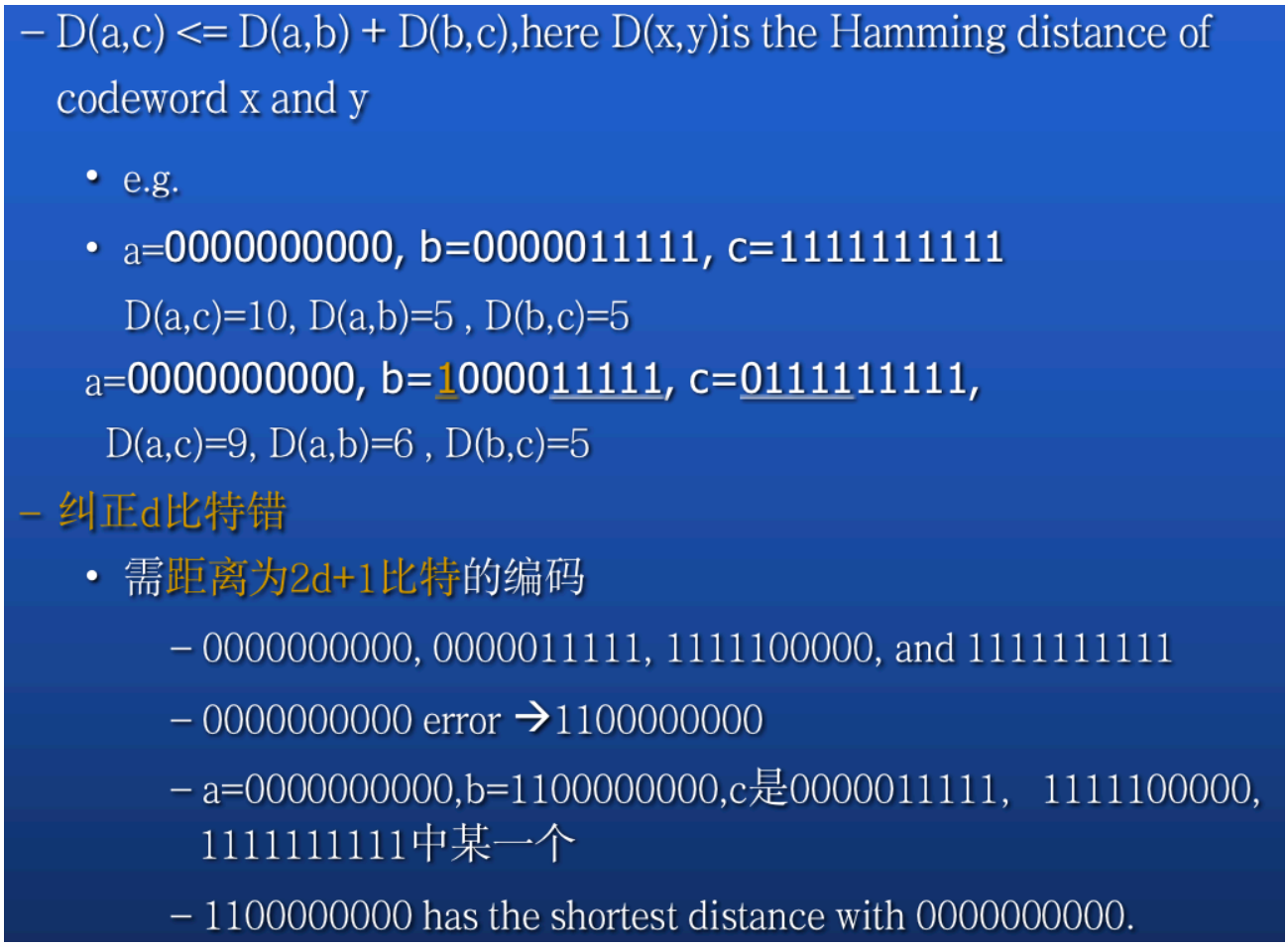
2 Error Detection And Correction

2.1 Error- correcting codes（不考！！）

- codeword码字：以2bit为单位，编码成码字发送
  - An n-bit unit containing data (m bit) and check bits (r bit). n=m+r
  - the list of four codewords : 0000000000（原始00）, 0000011111（原始01）, 1111100000, and 1111111111

2.1.1 Hamming 码

- The number of different bits in two codewords.
  - 10001001 xor异或 10110001=>00111000, Hamming distance=3
  - Hamming distance of a codewords list: Given a codewords list, look for the two codewords whose Hamming distance is minimum in the list. This distance is Hamming distance.

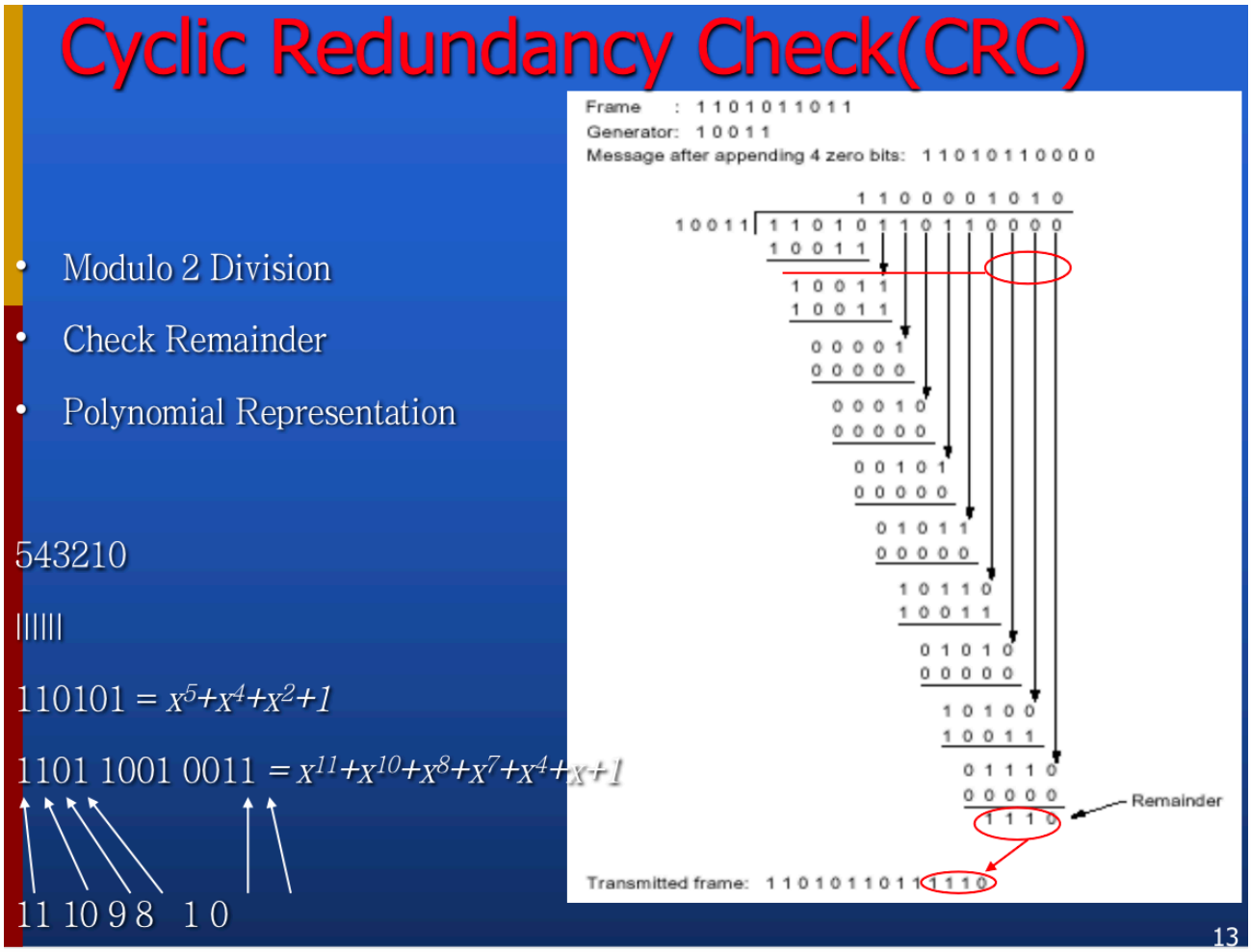


- 有点类似三角形任意一边小于任意两边之和
- 这样可以纠错，但是消耗很大

2.2 Error-detecting codes CRC

- polynomial(多项式) code（cyclic redundancy code循环冗余码，CRC code）
  - the algorithm for computing the checksum：
    - CRC-32
    - CRC-12 = x12+x11+x3+x2+x1+1
    - CRC-16 = x16+x15+x2+1

- CRC-CCITT = x16+x12+x5+1
- conclusion:
  - r阶的多项式能够检测所有长度<=r的突发错误（连穿错：最远的两个错了的bit，相差m，成为m大小的连穿错）
  - 长度大于r+1的错误漏检的 概率为$\frac{1}{2}^r$



- 根据除数的位数-1
- 模2加法无进位，模2减法无借位
- 先给原始数据加xbit位（根据采用的规则，CRC-12，就有13位的除数），上面那个appending的写错了
- 将原始数据+校验码发出去
- 接收方再对接收到的进行模2除法，一定为0，非0则出错
- 应该能证明，这个和普通的除法不太一样，模二除法/模二加法都是异或操作

2.3 停止等待协议

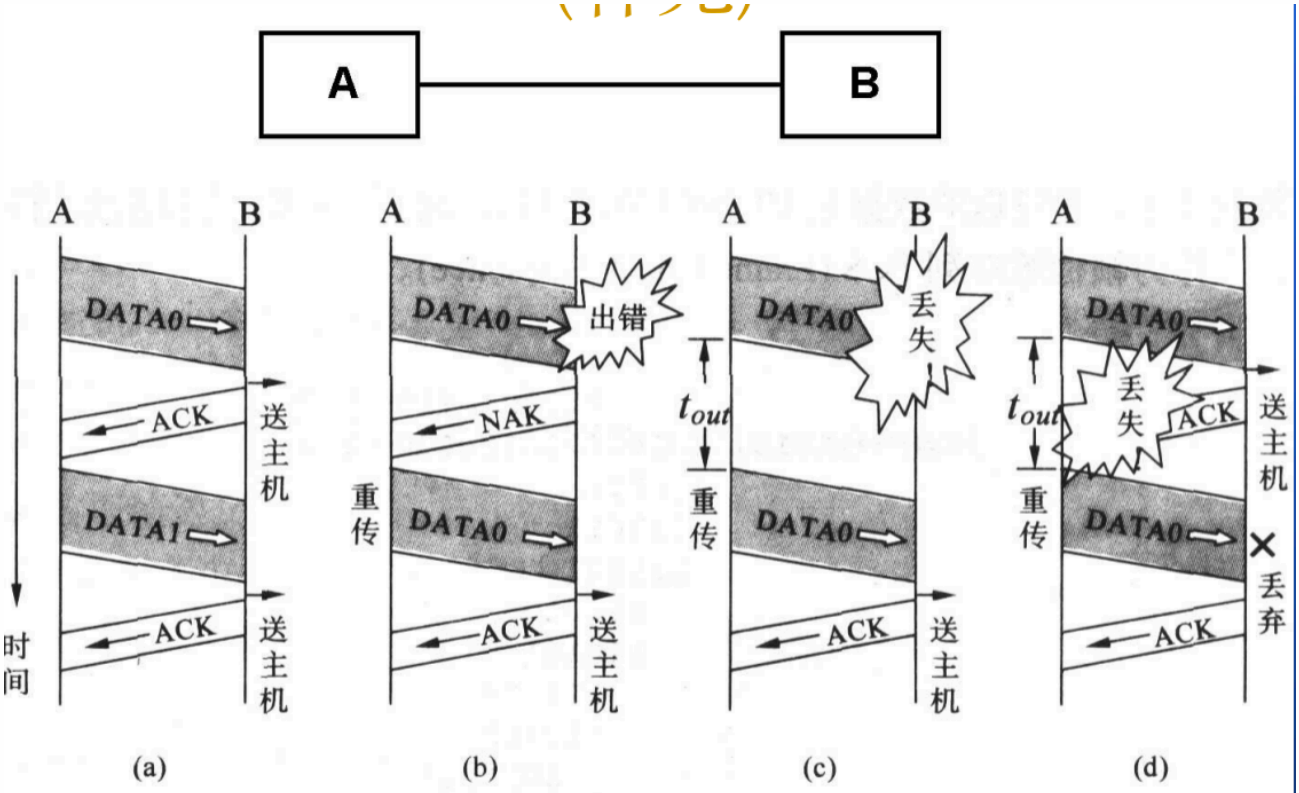


图 3-3　数据帧在链路上传输的几种情况：

(a) 正常情况；(b) 数据帧出错；(c) 数据帧丢失；(d) 确认帧丢失

- A、B就是电脑/路由器…
- 等待协议就是每发送一次都会等待应答，是正确应答还是错误应答还是丢失，根据不同的来处理
- 如何判断如何把重复帧扔掉：判断看序号，给帧编号（0、1、0、1、0、1…）
- 应答帧出错：A如果收到了出错的应答帧，那就和丢失的4情况一样处理，都会扔掉
- 应答帧必须发

3 Elementary Data Link Protocol

3.1 protocol 1

unidirectional data transmission, network layer of sender has unlimited data to transmit, network layer of receiver can receive unlimited data, channel never damages or loses frame.

3.2 protocol 2

stop-and-wait,simplex traffic, network layer of sender has unlimited data to transmit,channel never damages or loses frame.

Discard restricted condition: network layer of receiver can receive unlimited data.

3.3 protocol 3

- a protocol of PAR
- timer, sequence
- PAR(Positive Acknowledgement Retransmission) / ARQ(Automatic Repeat reQuest)
- network layer of sender has unlimited data to transmit.
- Discard restricted condition: channel never damages or loses frame.

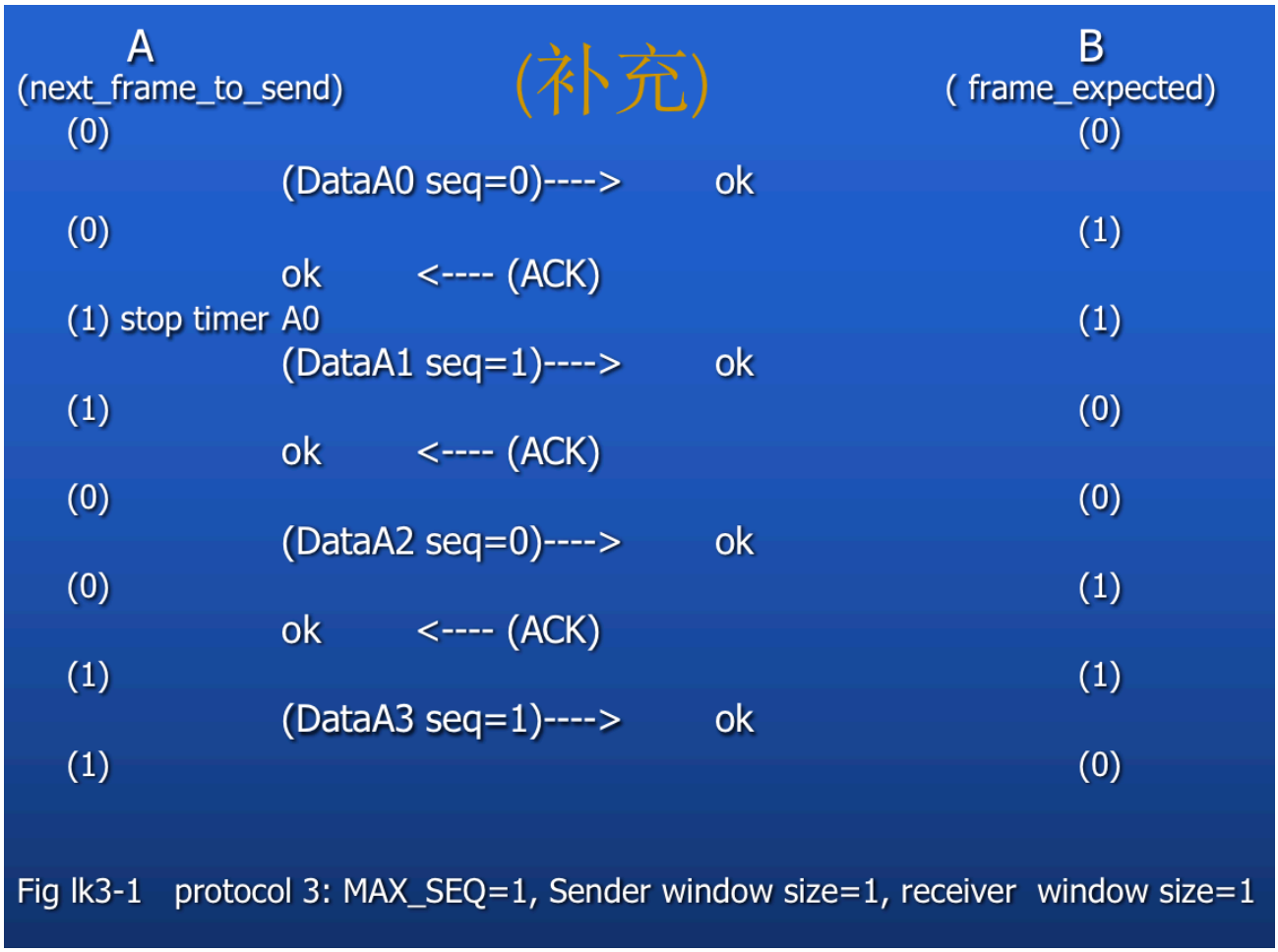
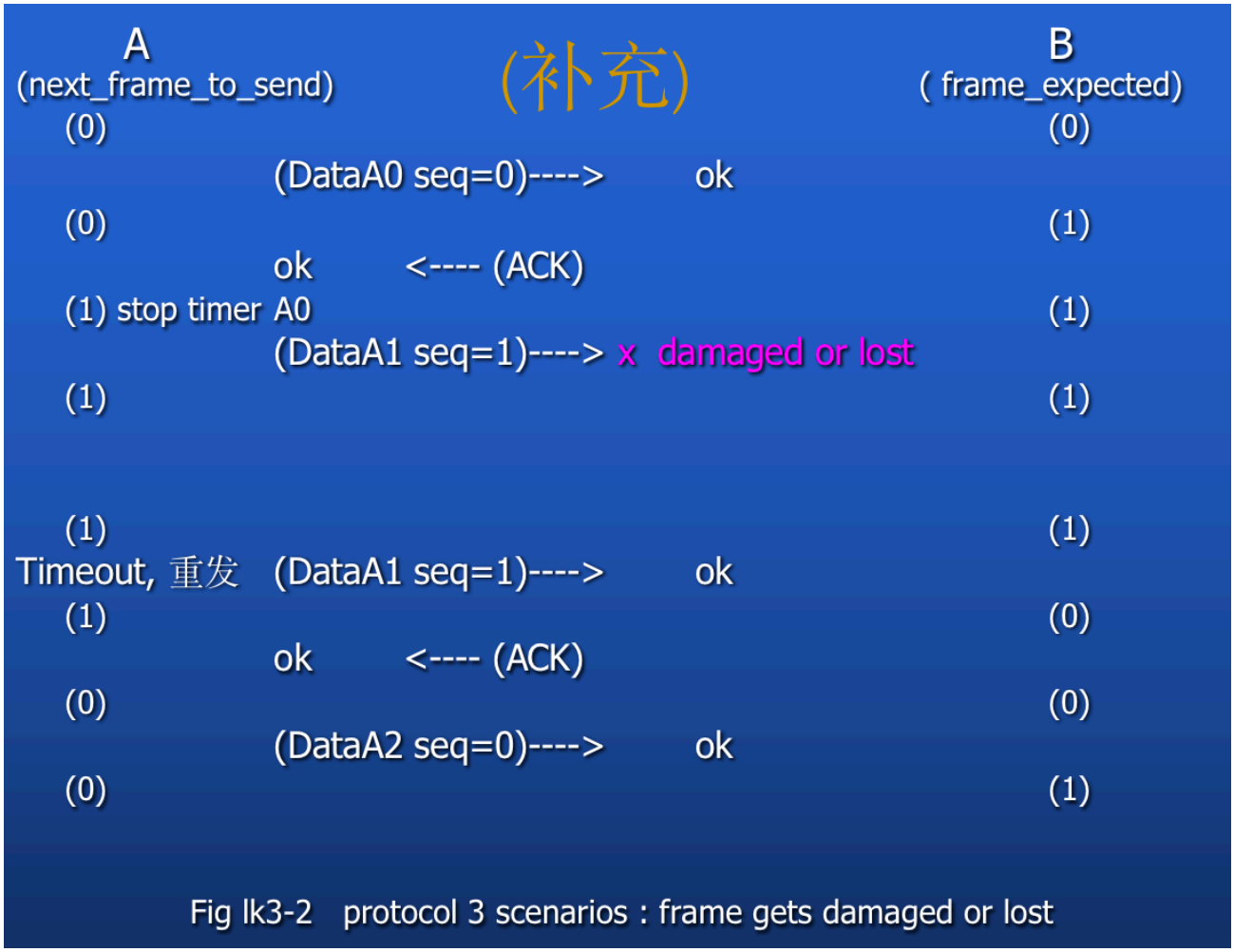


Fig lk3-1 protocol 3: MAX_SEQ=1, Sender window size=1, receiver window size=1



- 发送帧出错：定时器超时，A重发

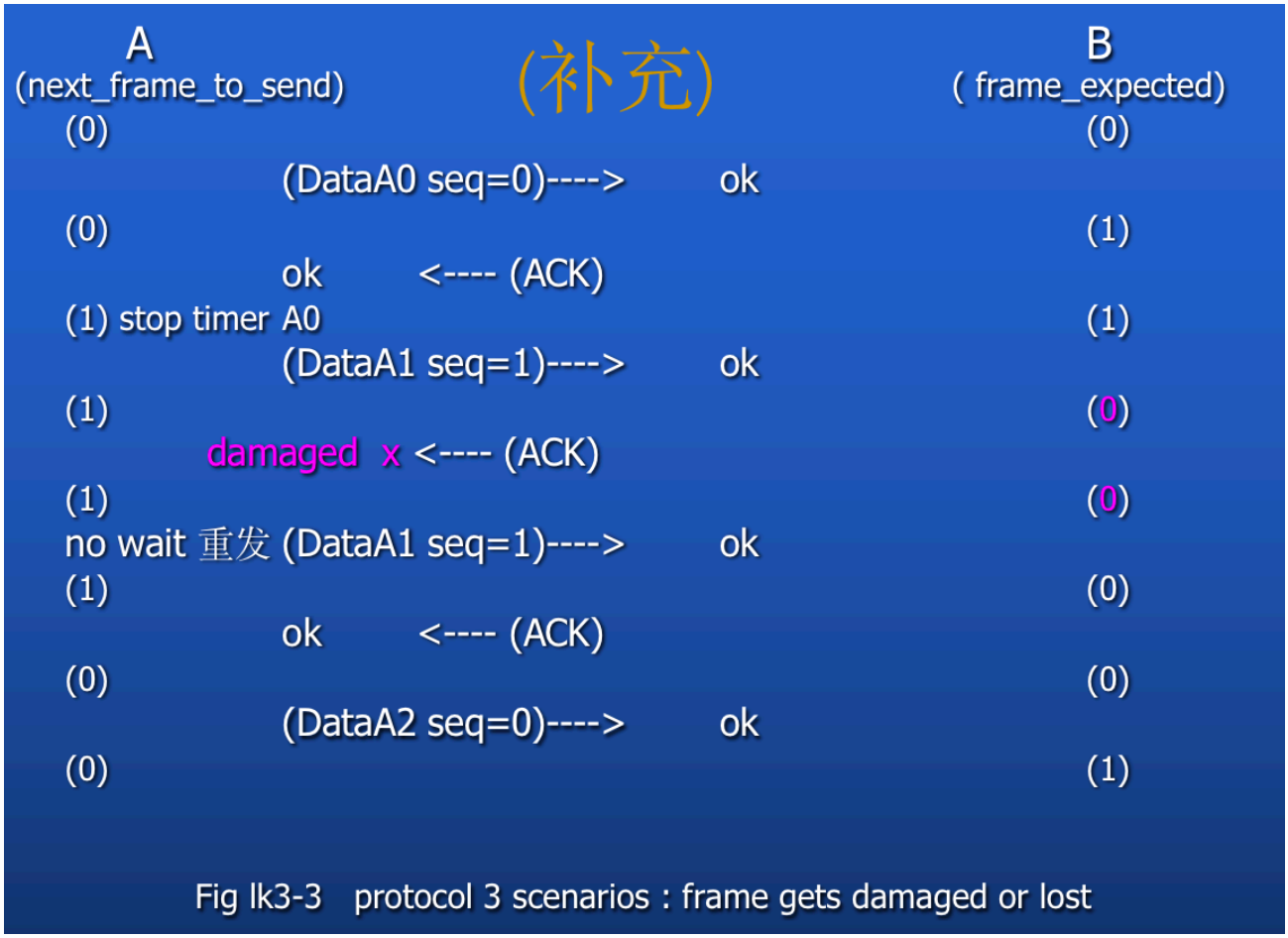
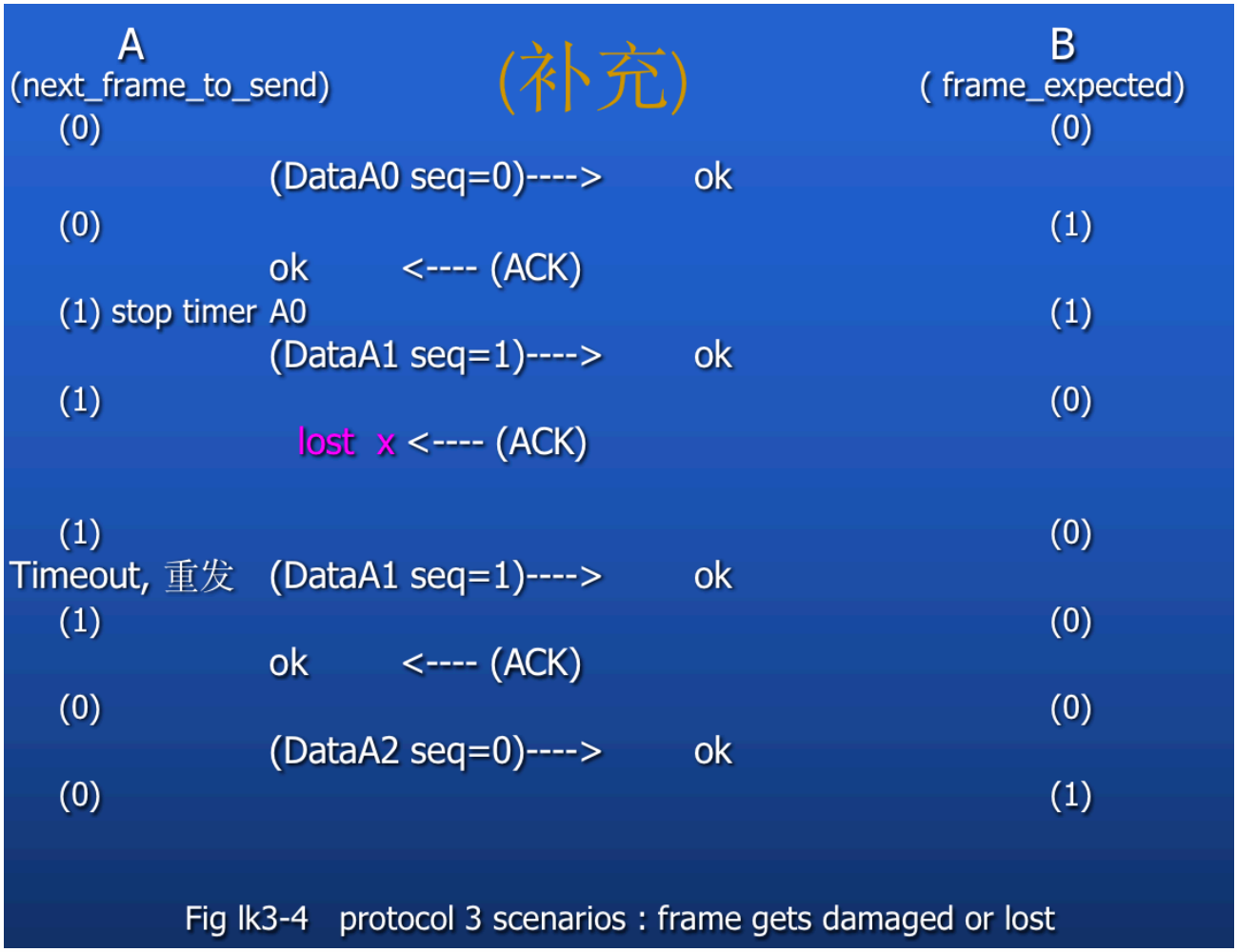


Fig lk3-3 protocol 3 scenarios : frame gets damaged or lost



- 应答帧出错/丢失：不会接收重复的发送帧，但会发一个应答帧

3.4 Sliding Window Protocol

- sending window: The sequence numbers within the sender's window represent frames that have been sent or can be sent but are as yet not acknowledged. 在等待应答协议中，sending window的大小=1。
- receiving window: corresponding to the set of frames which is permitted to accept for receiver. 应答窗口大小=1，always。

别的书上说：sending window is a set of sequence numbers corresponding to frame it is permitted to send for sender.下面是我根据别的书的作者的观点修改后的图片（红色是修改部分）。

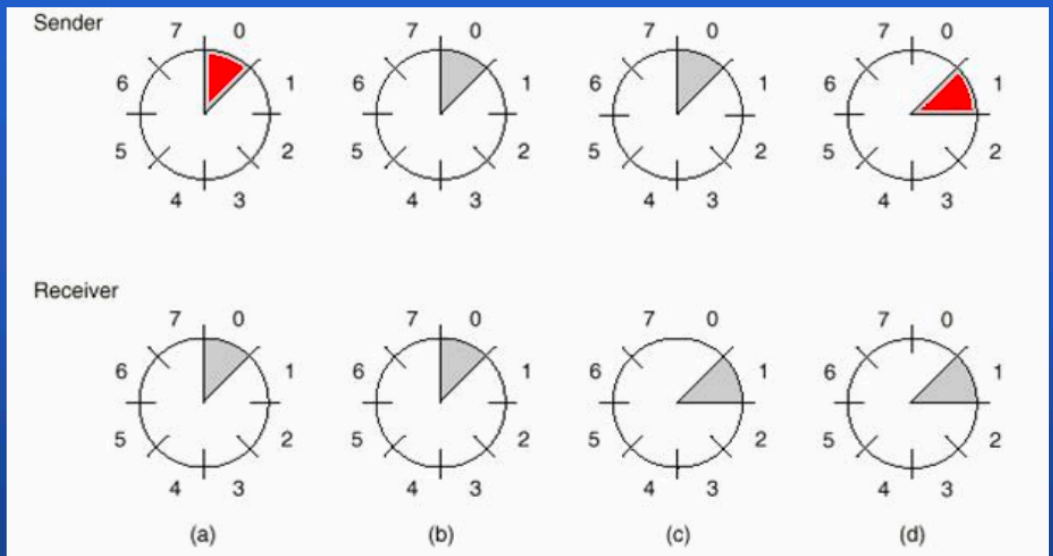
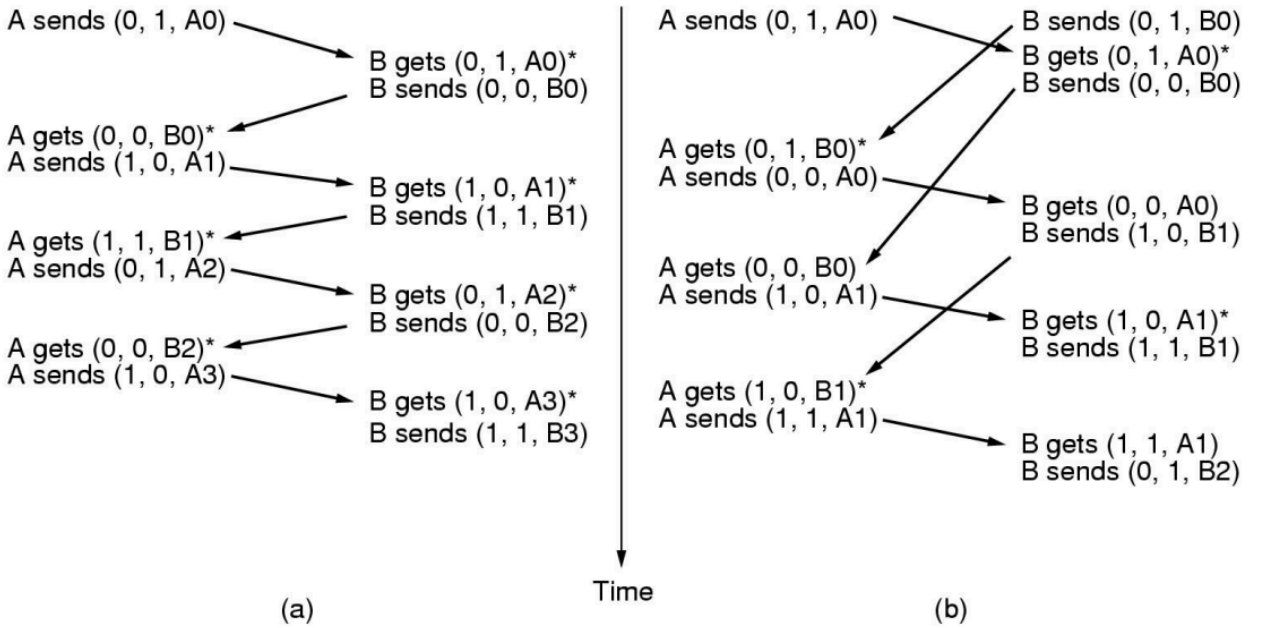


Fig. 3-12. A sliding window of size 1, with a 3-bit sequence number. (a) Initially. (b) After the first frame has been sent. (c) After the first frame has been received. (d) After the first acknowledgement has been received.

- 如果发送要发的话，那个集合里就会有那个序号，
- 接收方也是，就是集合里有哪些已经接收了的序号/等待接收的信号
- 上下俩是一对，如果一方不工作，那么另一方也做不了



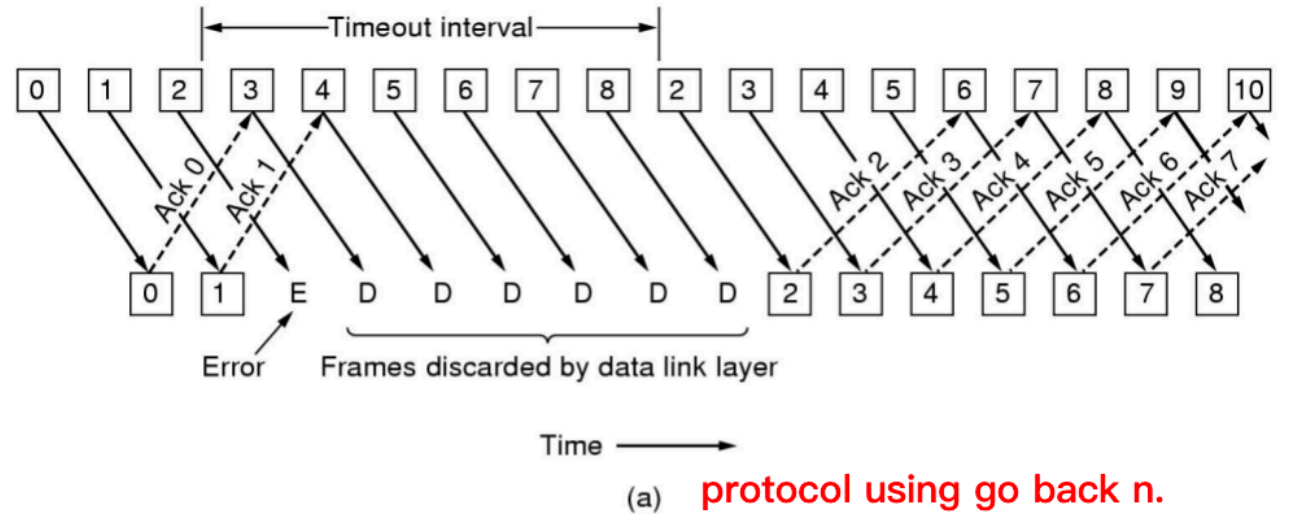
Two scenarios for protocol 4. The notation is (seq,ack,packet number). An asterisk indicates where a network layer accept a packet.

- 顺带应答，发送信息帧的时候顺便把应答帧给发了
- ACK取决于发送方发的数据帧的序号
- 图右是处理不当的情况（很乱，我也没看懂）

3.5 protocol 5

- Stop-and-wait is not fitful for channel which has long round-trip delay, e.g. Satellite channel,270 ms
- 发送等待的信道浪费严重
- Bidirectional transmission,noisy channel,
- Discard restricted condition: network layer of sender has unlimited data to transmit
- size of receiving window = 1, receiver discards all subsequent frames after a damaged or lost data frame.

3.5.1 Go back N -后退N协议



- 可以一次发7个，但是超过7个包还没有应答，那么就要等待了
- 如果B给一个应答回来，那么窗口就滚动一个，发送方扔掉一个集合内的元素，可以发一个新的包
- 图上的2号帧出错了，那么接收方后面得到的都扔掉，一定要等2号帧，一直不发ACK。然后超时发送方重新发送。
- 链路层用该协议多。

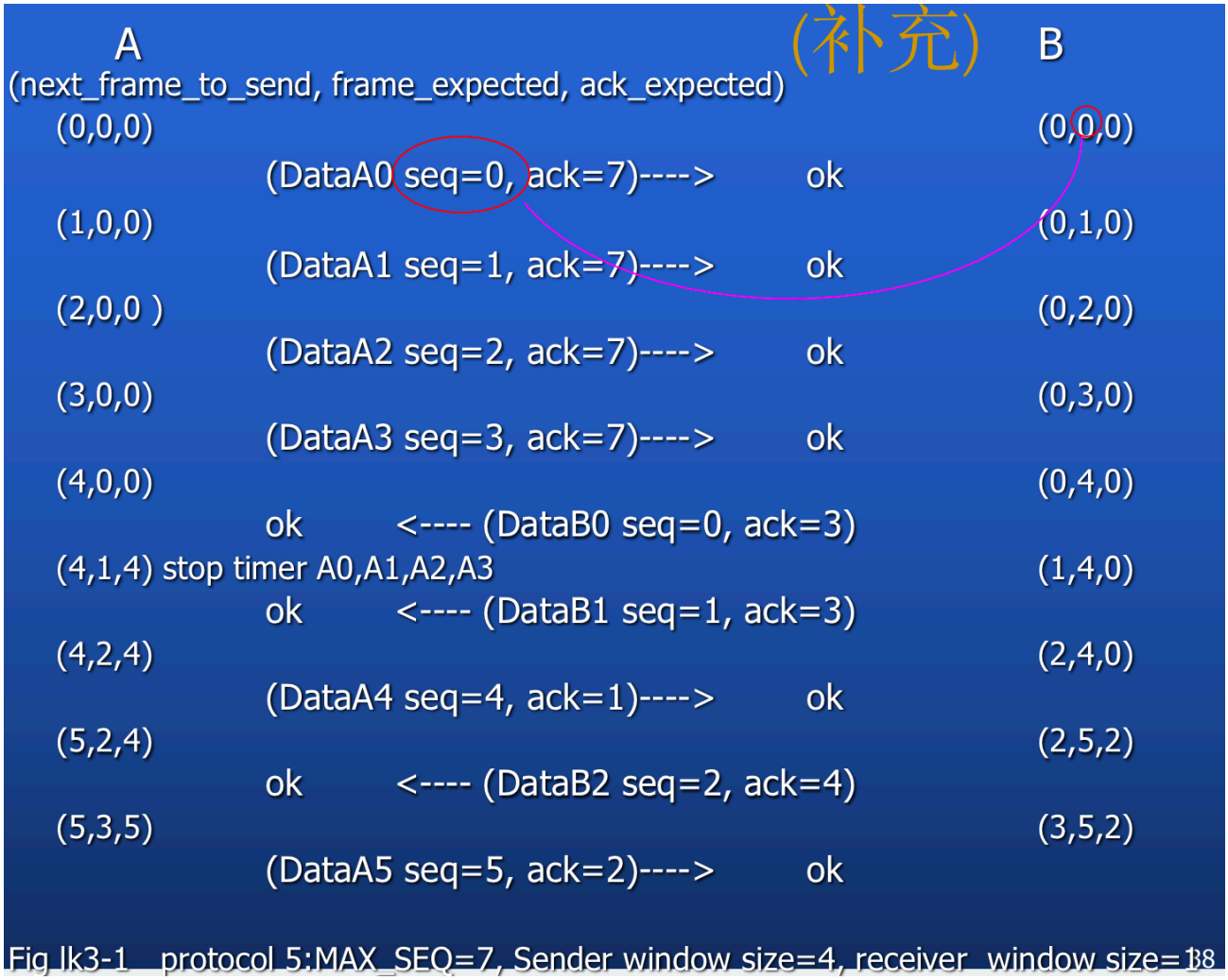


Fig lk3-1 protocol 5:MAX_SEQ=7, Sender window size=4, receiver window size=1 38

- seq：帧号，从 `next_frame_to_send` 拷贝过来，就是即将要发的帧号
- `frame_expected` 是期望收到的帧，比如第一次B就在等A发0号帧，收到之后就加一变成了1，但是ack发送给A的要减1，即0。
- `ack_expected` 是最早发送的但还未被应答那个帧。如果对方应答了3号帧已经收到了，那么`ack_expected` 就加1，就是说我接下来要发4号帧，我也在等4号帧的回复。之后发送窗口就往后移xx位。
- ack是说对面发送的x即以及x之前的帧已经接收到了。比如ack=7，说明之前那个窗口已经接收完毕了，其实是“上辈子”B发的帧已经全部被接收了（相当于初始化）。如果ack=3，那么说明A之前发的3、2、1、0 四个帧都已经收到了。
- `ack = frame_expected - 1; seq = next_frame_to_send; ack_expected =` 发送过的帧号

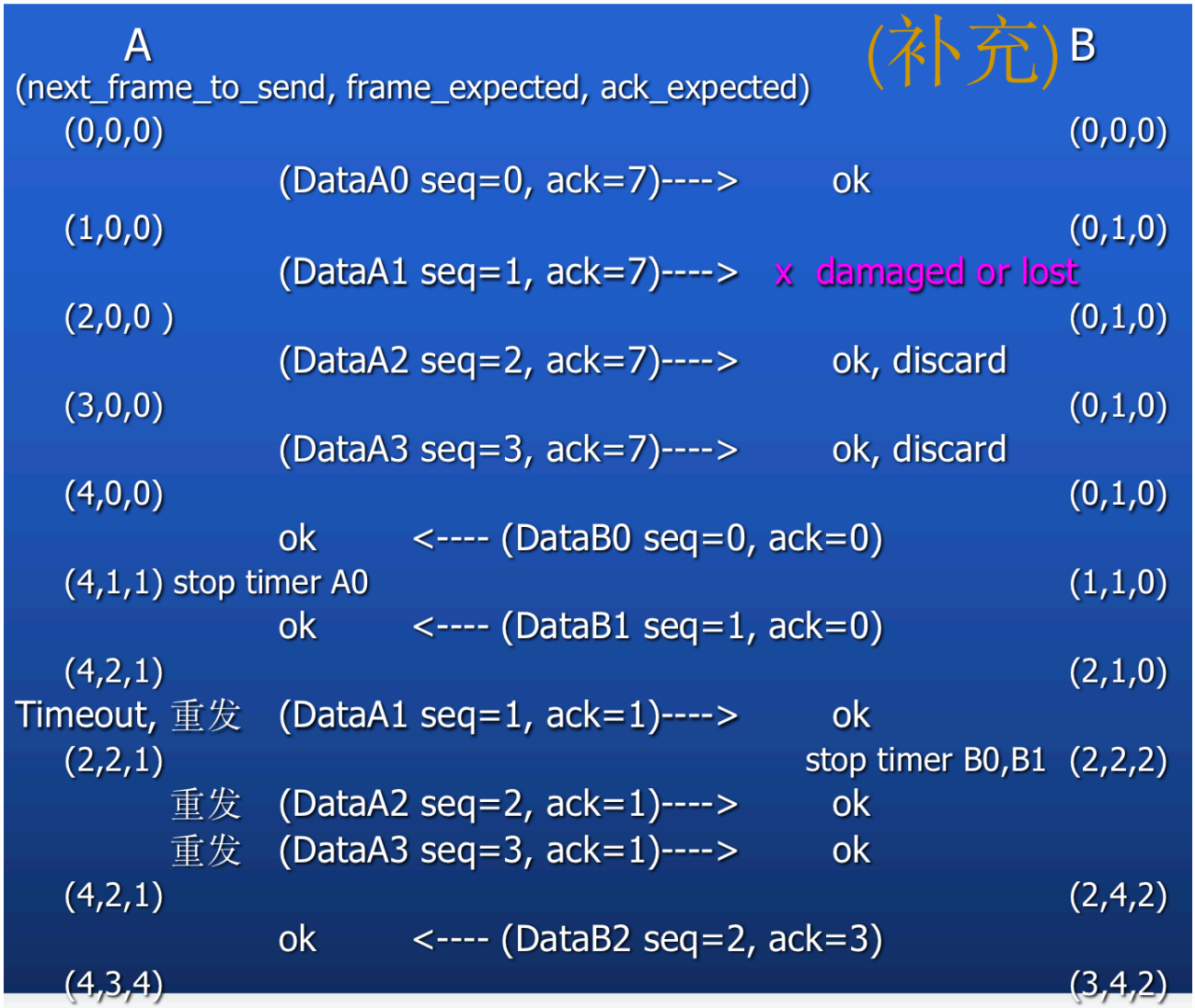
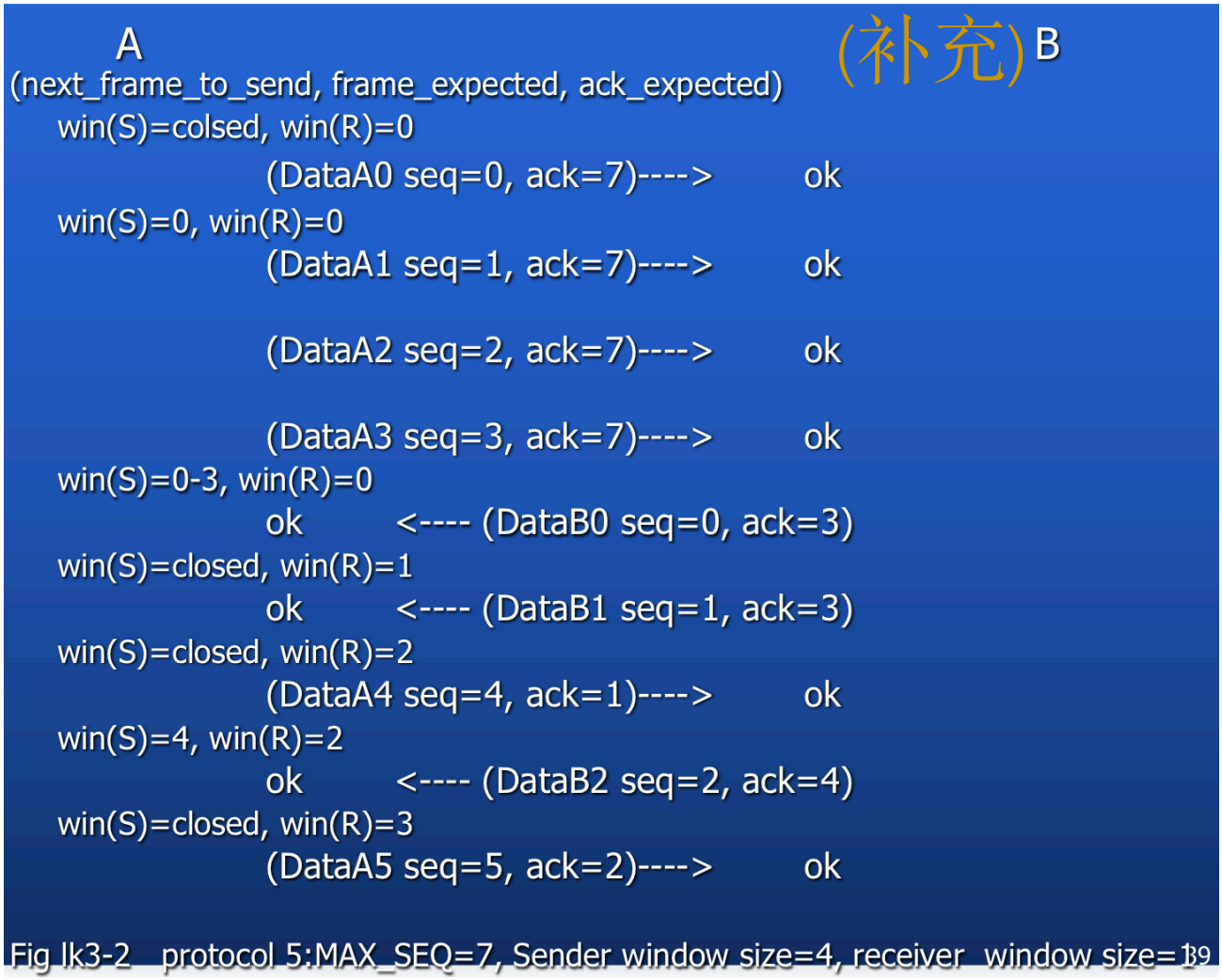


Fig lk3-3 protocol 5 scenarios: frame gets damaged or lost

- 这是出错的情况
- 出错未卡死
- 但是还是有问题，程序没有单独的应答帧，如果B不给A发数据，那么A永远接受不到应答帧，那么A就会死循环一直发送消息



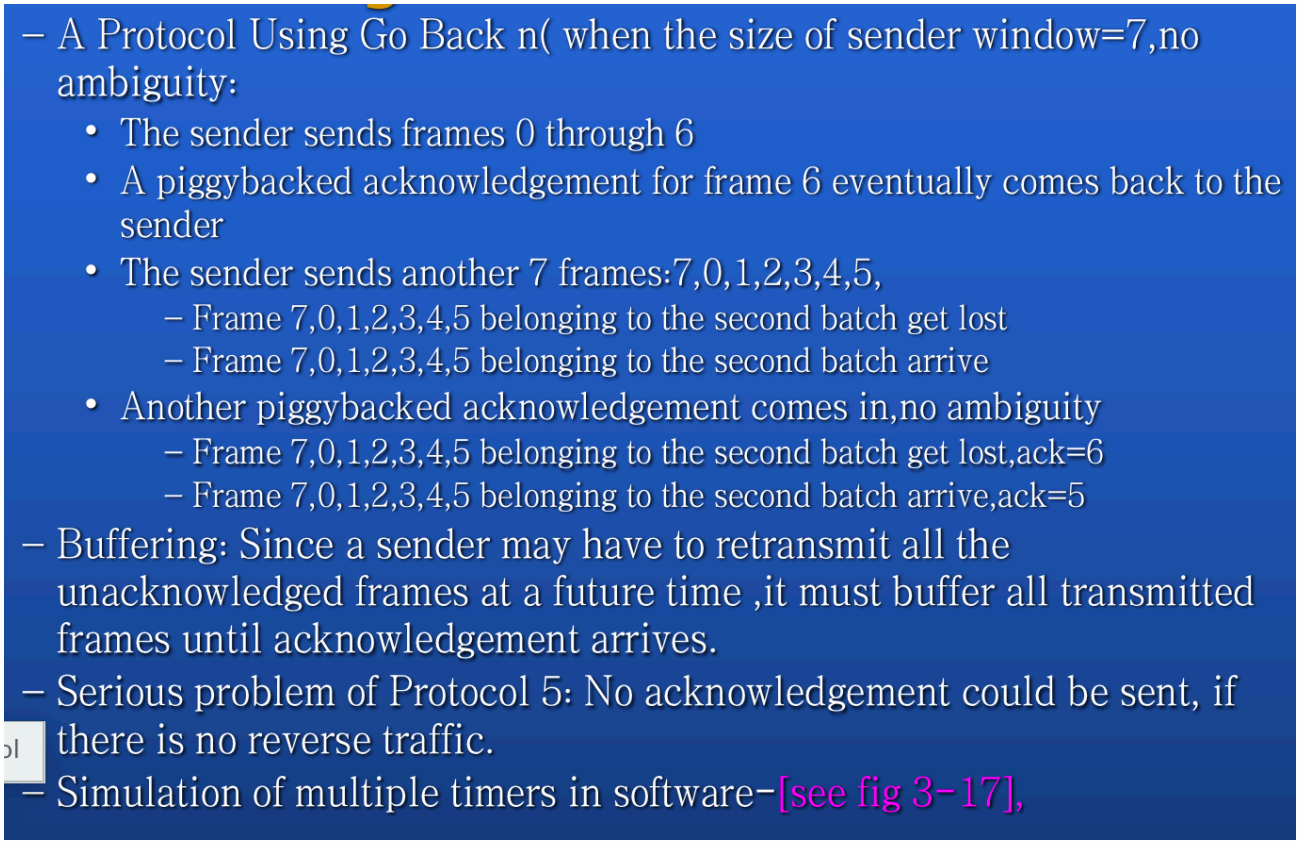
- 这是发送窗口和接收窗口变化的情况

3.5.2 discussion

- A Protocol Using Go Back n
  - MAX_SEQ+1 distinct sequence numbers(0,1,2,... MAX_SEQ),no more than MAX_SEQ unacknowledged frames,the sender window<=MAX_SEQ.Why? Consider the following scenario with MAX_SEQ=7(a sender window of size eight):
    - The sender sends frames 0 through 7.
    - A piggybacked acknowledgement for frame 7 eventually comes back to the sender.
    - The sender sends another eight frames,again with sequence numbers 0 through 7.
      - All eight frames belonging to the second batch get lost
      - All eight frames belonging to the second batch arrive successfully.
    - Another piggybacked acknowledgement for frame 7 come in, ambiguity will arise to sender:
      - All eight frames belonging to the second batch get lost,ack=7
      - All eight frames belonging to the second batch arrive successfully,ack=7

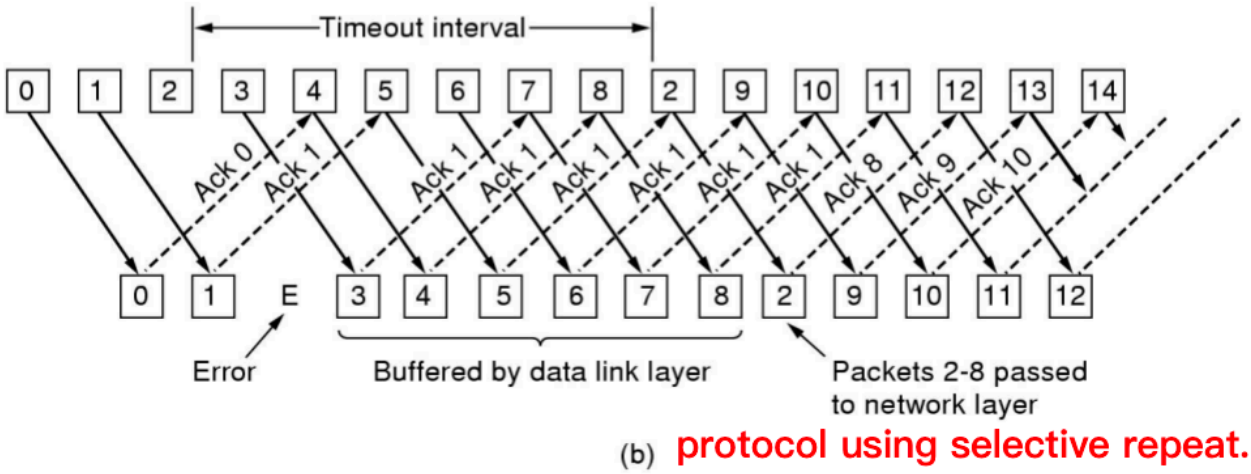
$$sender\ window \leq MAX\_SEQ \tag{1}$$

- 比如我的MAX_SEQ是7，即帧号可以到7，那么我窗口只能有7个（0-6），不能有8个（0-7），why?
- 如果发0-7 八种帧，同时有8个窗口
- 存在一种问题：如果包全部丢了，此时也是发送7（上次最后接收到了7），如果包全部收到，也是发送7，所以会出现两头难ambiguity。



- 开窗口等于7，那么不会出现ambiguity
- 如果全丢，那么我要等待收到帧号7，我应该给ack=6；
- 如果全到，那么我要等待收到6，那么应该给ack=5。
- 本质上就是牺牲一个窗口，来区别收到的情况
- 发送窗口数是最大帧号减1。

3.6 Selective repeat 选择重传协议（Protocol 6）



(b) protocol using selective repeat.

(a). Effect of an error when the receiver window size is 1.
(b) the receiver window size is larger than 1.

- 像后退N把很多扔掉了，有点浪费；选择重传协议尽管出错，但把收到的正确帧先存到缓存区，不会直接传到网络层，等到正确的帧也发送过来，再给网络层。
- 如果2出错了，那么就发送已经完成的序号（ACK=1，1前面的0、1已经发送完成），后面再重发2。
- TCP用该协议多。
- 不按顺序到达的帧，一定要选择Protocol 6
- 接收窗口大小一定要等于发送窗口大小：等在那里
- 对于3 bit序号值，序号范围为 0 - 7，即MAX_SEQ=7,发送窗口的大小为(MAX_SEQ+1)/2=4，接受窗口的大小=4，证明如下

$$(MAX\_SEQ + 1)/2 = \text{接收窗口大小} = \text{发送窗口大小} \tag{2}$$

- 发送窗口的大小不能大于(MAX_SEQ+1)/2，这里是4（设MAX_SEQ＝7），原因如下：
  - 设发送窗口和接受窗口的大小都是5，序号空间为0－7；
  - A的发送窗口＝[0,1,2,3,4], B的接收窗口＝[0,1,2,3,4]
  - A向B发送 0、1、2、3、4＃帧，两种不同的情况发生了：
    - 情况1：B收到全部5个帧，应答也全部被A收到；A新的发送窗口＝[5，6，7，0，1], B新的接收窗口＝[5，6，7，0，1], A新发5，6，7，0，1＃帧；
    - 情况2：5个帧全部收到，但应答全部丢失，A新的发送窗口＝[0,1,2,3,4], B新的接收窗口＝[5，6，7，0，1]，A重发0＃帧；
  - 问题：新来的0＃帧会被接收并交给高层，如果是重发的（情况2）本该丢弃，却被B接收并交给高层。
    - 原因：在情况2下A的发送窗口（[0,1,2,3,4]）与B的接收窗口（[5，6，7，0，1]）有重叠部分。

- 根本上就是不能有一个窗口内不能有重复部分，有重复部分就会重叠，导致BUG

4 Example Data Link Protocols 实际

4.1 HDLC---High-level Data Link Control

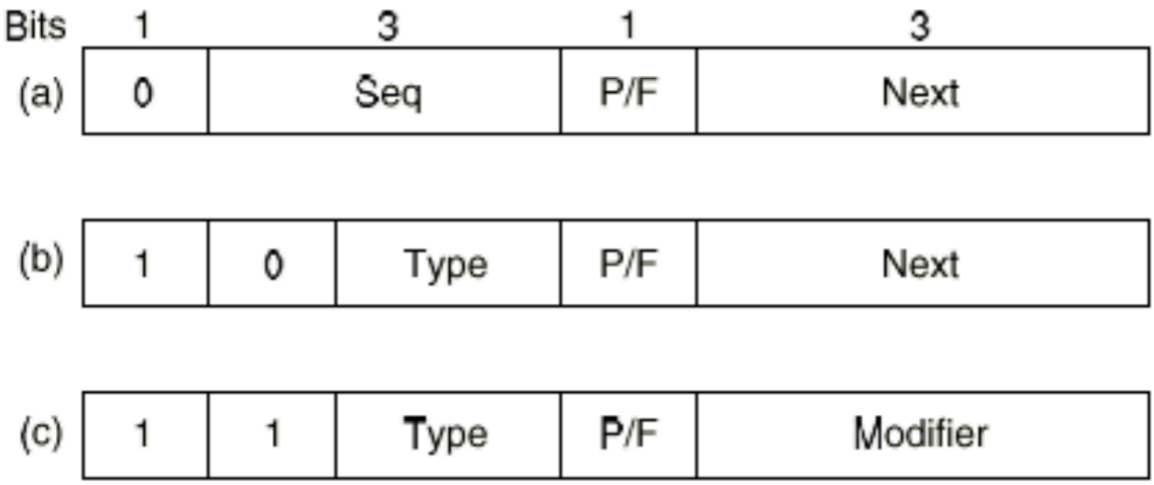


Fig. 3-25. Control field of (a) an information frame, (b) a supervisory frame, (c) an unnumbered frame.

- Seq --- sender seq(like former protocol 5)
- Next --- expected frame seq, different from former ack,it is (ack+1) mod 8

- next不等于ack！！！但是和ack作用一样，之前是ack回应收到了3号帧，next是说我想收到你的4号帧，言下之意也是1、2、3号帧我已经收到了

- Information frame
- Supervisory frame:
  - RR(Receive Ready),接收准备就绪
  - RNR(Receive Not Ready): to implement flow control
  - REJ(Reject): like NAK frame,从Next起的所有帧都被否认,但确认序号为Next-1及其以前的各帧
  - SREJ(Selective Reject),选择性拒绝
- Unnumbered frame
  - request to establish connection: SNRM(Set Normal Response Mode), SABM(set asynchronous balanced mode)
  - request to disconnect: DISC(Disconnect)
  - response frame: UA(unnumbered acknowledgement)
  - report severe error: FRMR(FRaMe Reject)

4.2 PPP协议

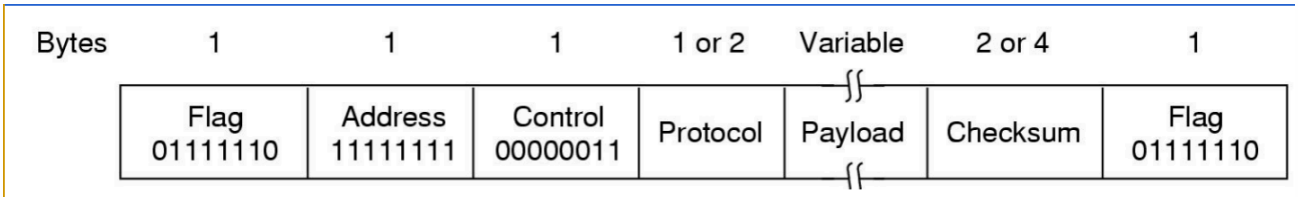


Fig. 3-27. The PPP full frame format for unnumbered mode operation.

- 信息是面向字节的，因此一定要 8bit位的整数倍
- 但HDLC不需要是字节的整数倍

– PPP—Point to Point Protocol

- support: error detection、multiple protocols(IP,IPX,Apple Talk,XNS,...),allows IP address to be negotiated、permits authentication...
- LCP(Link Control Protocol).
- NCP(Network Control Protocol).
- Frame format(as HDLC):face character-[see fig 3-27]
- A simplified phase diagram for bringing a line up and down-[see fig 3-28]

- 支持身份鉴别

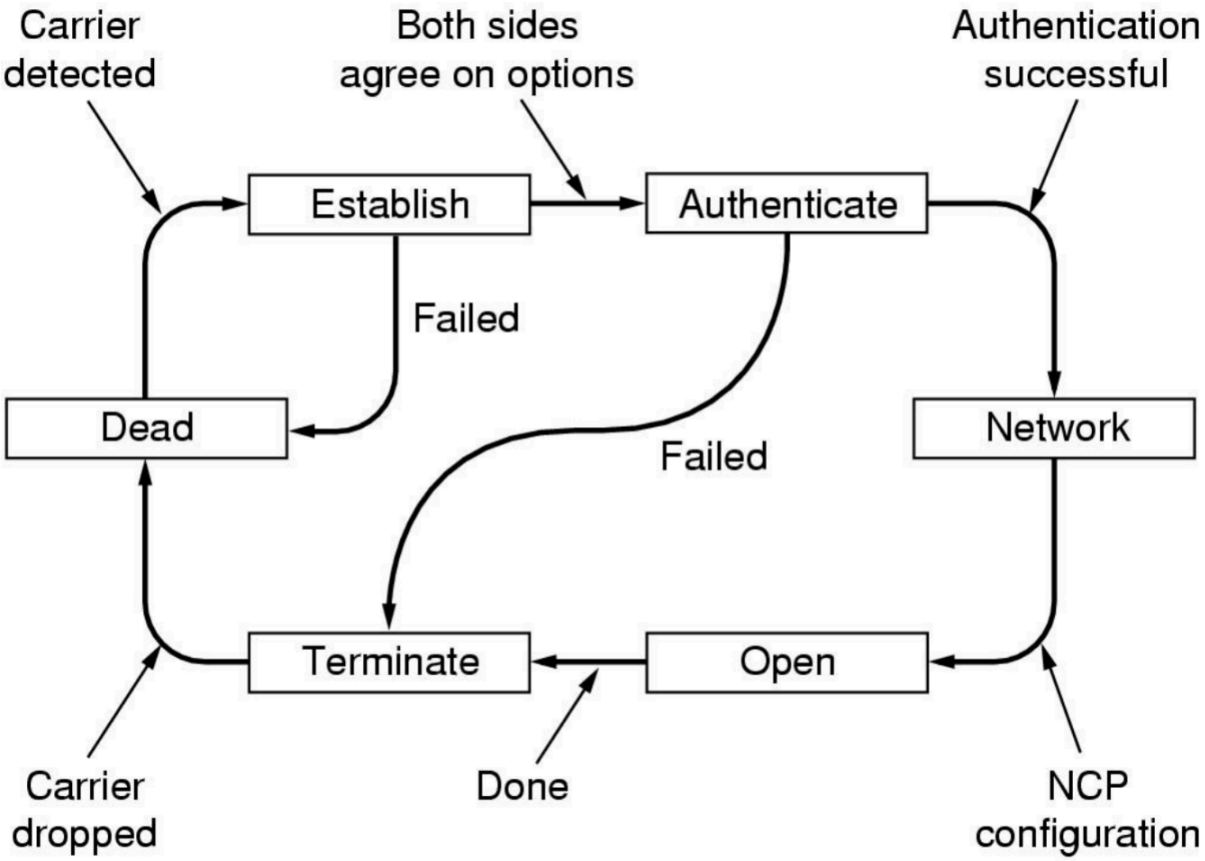
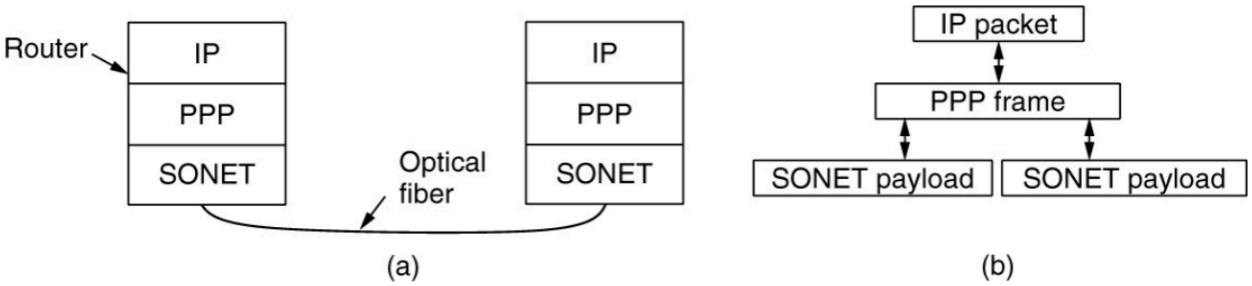


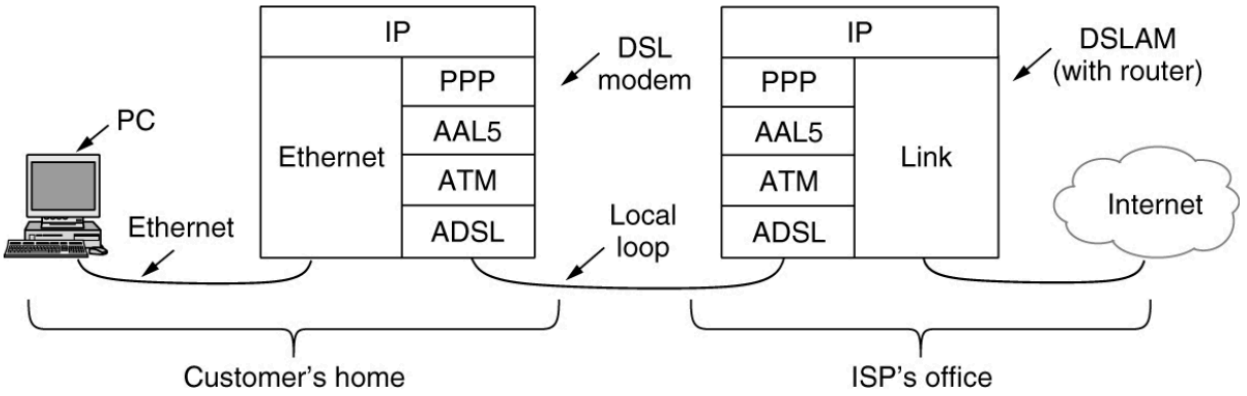
Fig. 3-28. A simplified phase diagram for bringing a line up and down.

4.3 SONET



Packet over SONET. (a) A protocol stack. (b) Frame relationships.

4.4 ADSL



ADSL protocol stacks