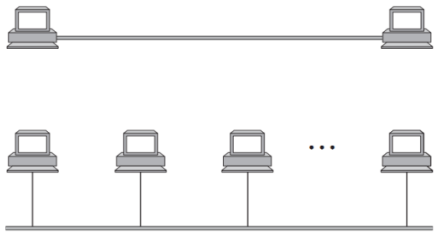
计算机网络

Computer Networks

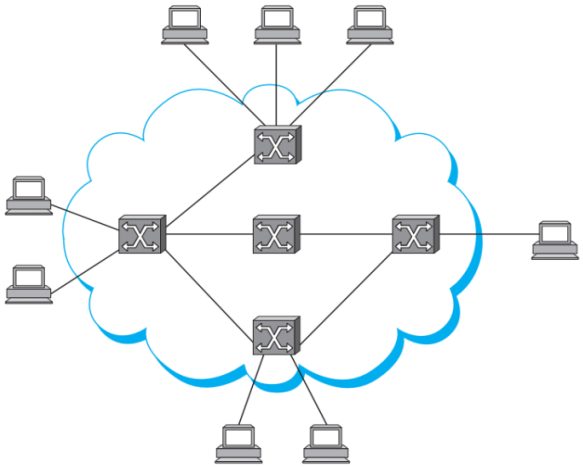
直连网络：点到点链路

回顾: 网络连通性

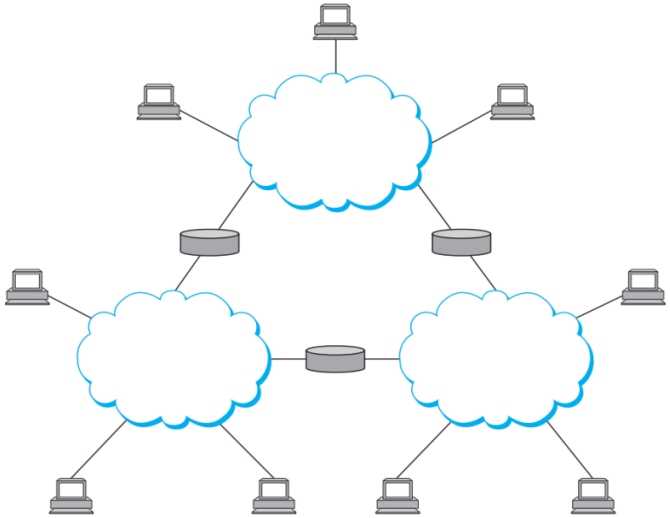
- 什么是直连网络?
  - 所有的主机通过某种物理介质直接连接



直连网络



交换网络



网络互联

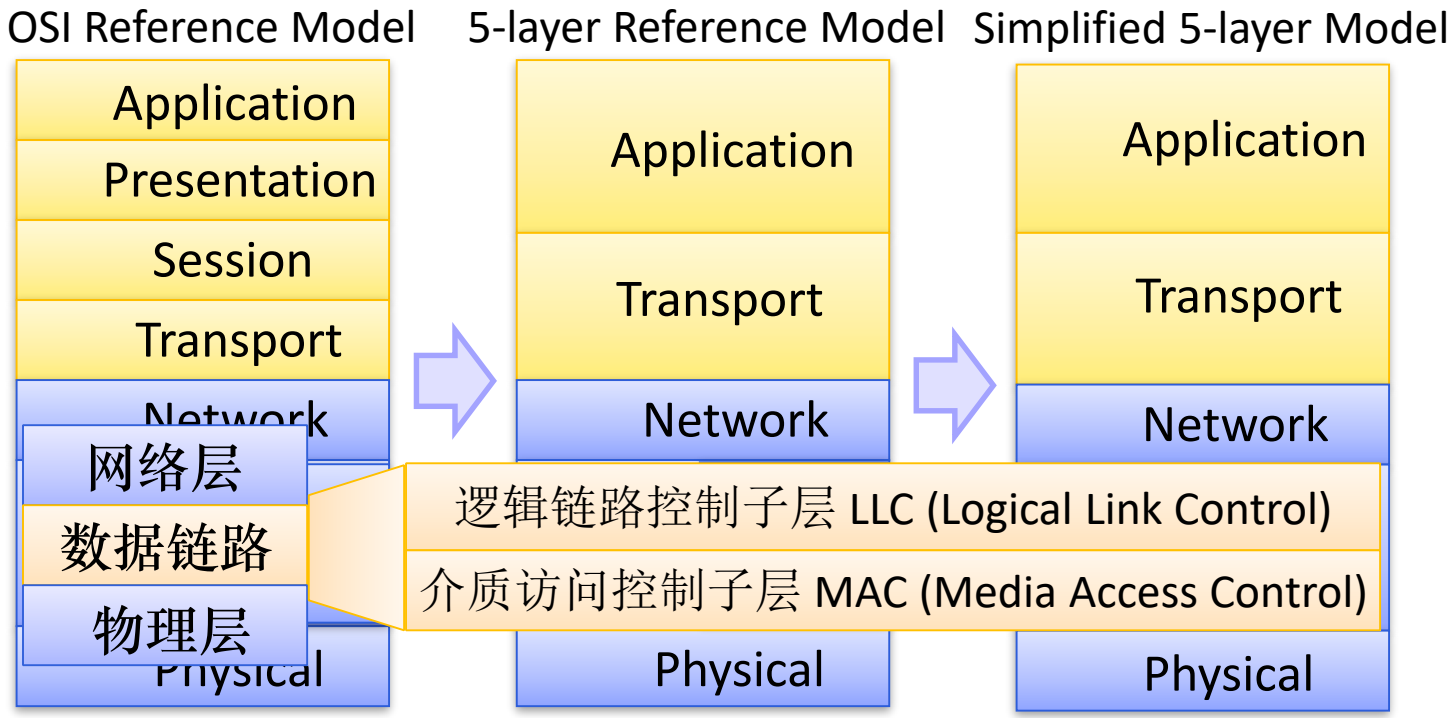
直连网络

什么是直连网络?

- 所有的主机通过某种物理介质直接连接
  - 物理传输介质: 电缆, 光纤, 无线电波, …
  - 与距离无关
    - 小的区域(例如, 一栋办公大楼)
    - 一个大的区域(例如, 横贯大陆)
  - 传播技术(通信方式)
    - 点到点的链路
    - 广播链路

回顾: 网络体系结构

- 直连网络问题在体系结构中所处的层次?
  - L1(物理层) 处理物理传输介质上的数字通信问题, 特别是比特如何表示为信号
  - 直连网络的研究主要集中于L2(数据链路层)



提纲

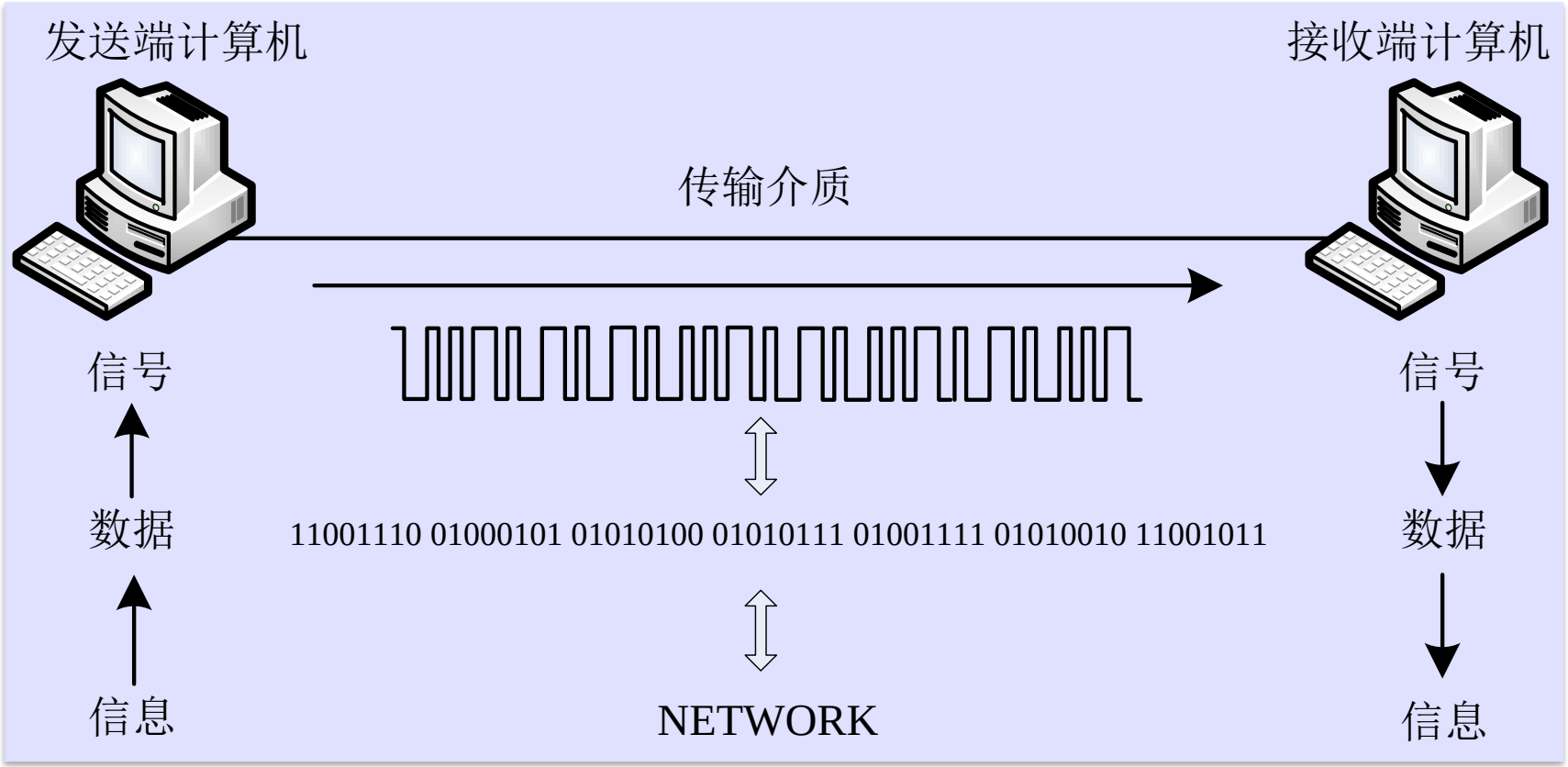
- 问题: 物理上相连的主机
  - 物理层
    - 数据编码
  - 数据链路层
    - 帧定界
    - 差错检测
    - 可靠传输
  - 介质访问控制子层
    - 多路访问
    - 局域网技术
    - 以太网 (802.3)
    - 无线局域网

点到点链路的四个基本问题

共享链路的问题

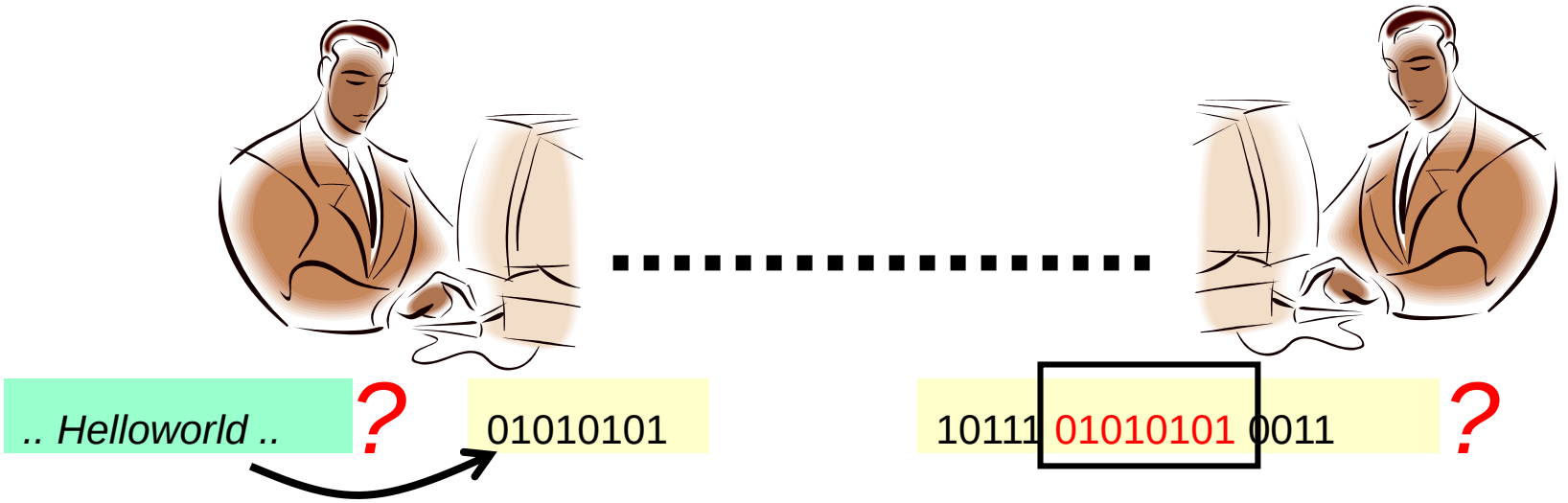
物理层的研究问题

- 信息、数据与信号



物理层的研究问题

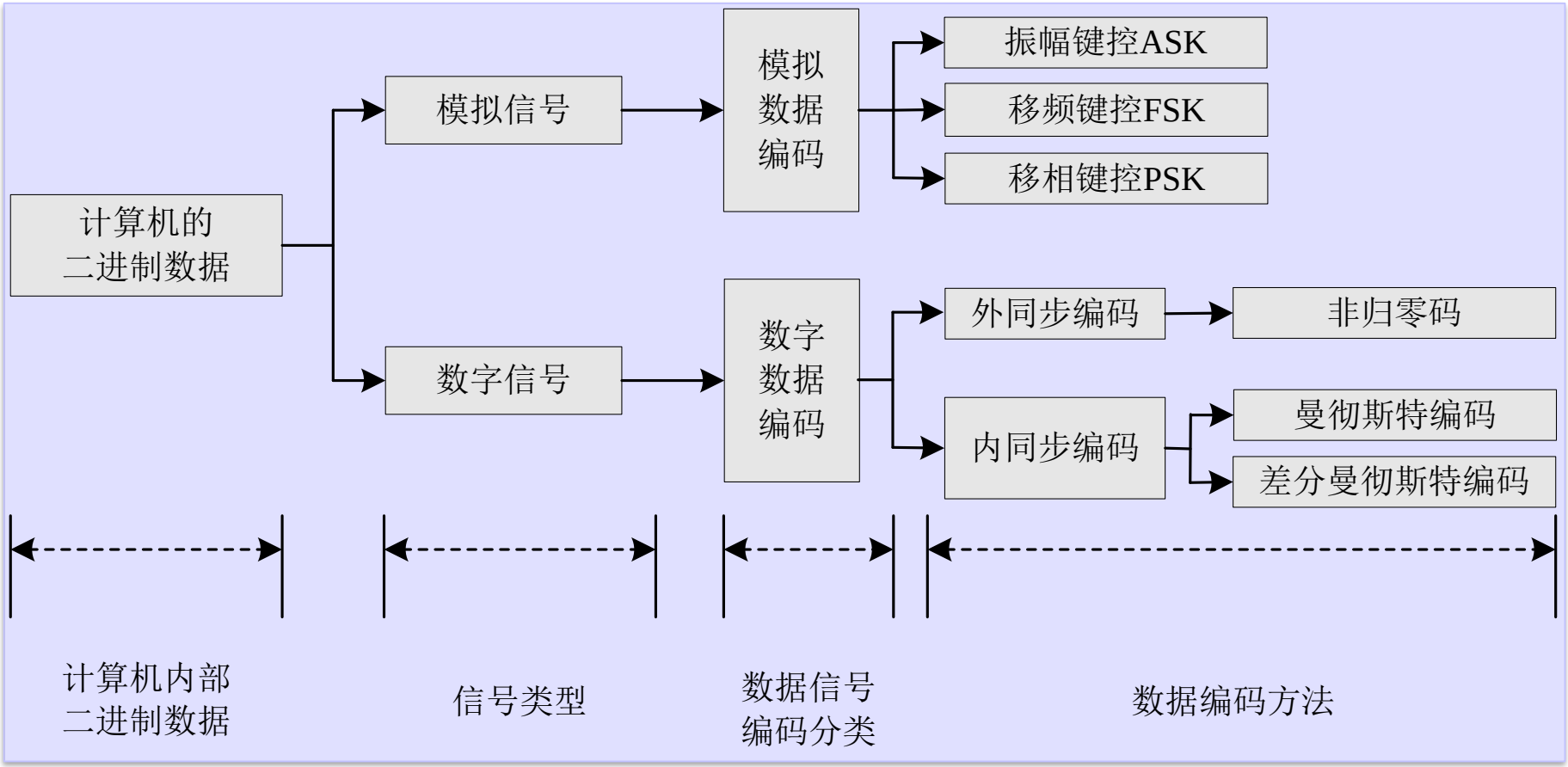
- 数据通信（编码）
  - 对发送到电缆或光纤上的比特进行编码，使其能被接收主机所理解



通常，将数字数据转换成数字信号的过程称为编码(coding)，而将数字数据转换成模拟信号的过程称为调制(modulation)。

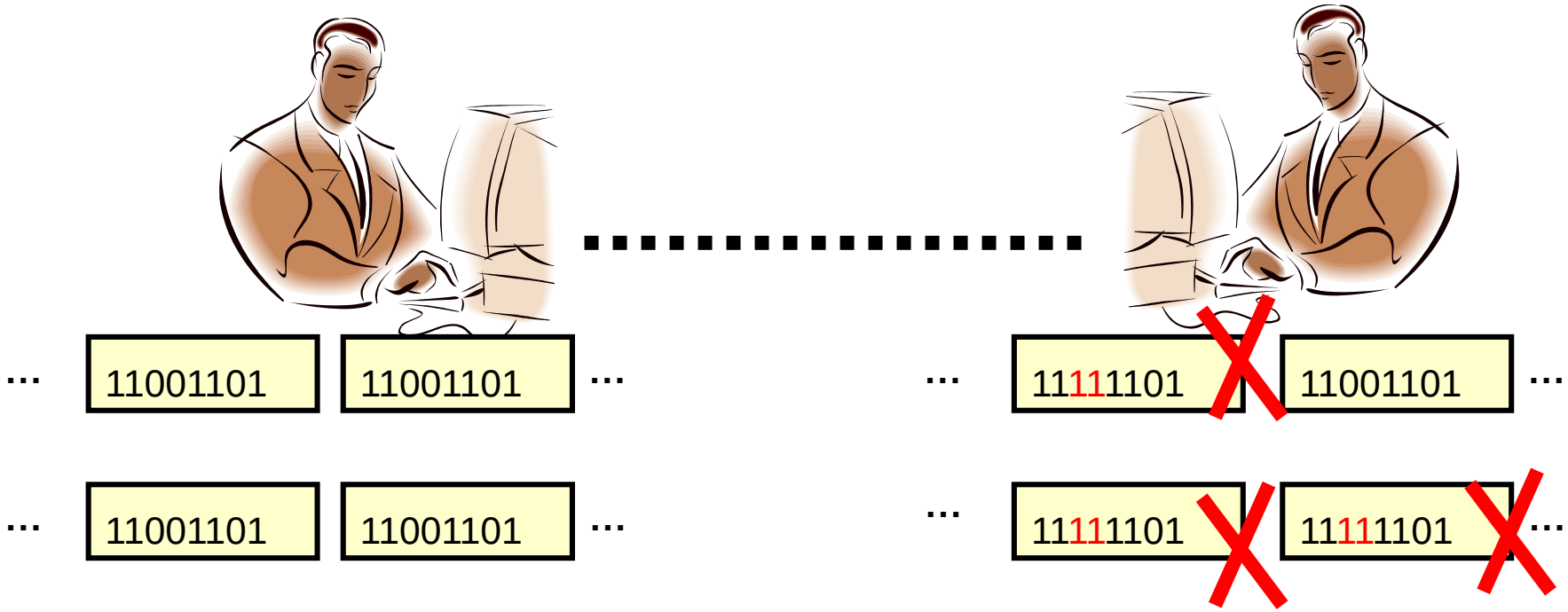
物理层的研究问题

- 数据编码分类



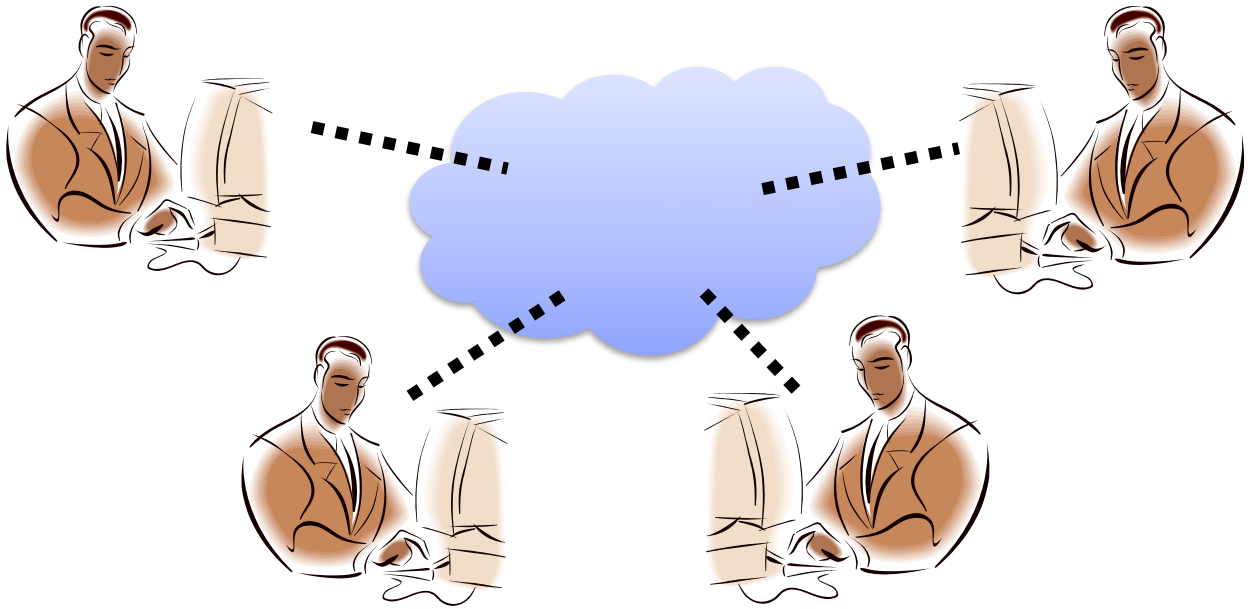
数据链路层的研究问题

- 帧定界（组帧）：把物理链路上传输的比特序列描述为完整的消息，以便传送到端结点
- 差错检测：检测帧传送过程种可能出现的错误，并采取相应的动作
- 可靠传输：保证链路在帧不时可能出现错误情况下的可靠性



介质访问控制子层的研究问题

- 通信链路共享 (介质访问控制)
  - 如果链路静态共享, 很容易处理
  - 如果链路动态共享, 如何控制多个主机的访问顺序?



提纲

- 问题: 物理上相连的主机
  - 物理层
    - 数据编码
  - 数据链路层
    - 帧定界
    - 差错检测
    - 可靠传输
  - 介质访问控制子层
    - 多路访问
    - 局域网技术
    - 以太网 (802.3)
    - 无线局域网

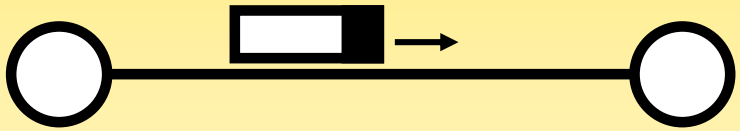
点到点链路的四个基本问题

共享链路的问题

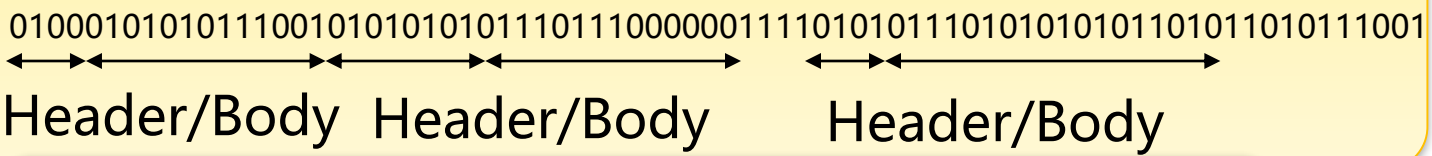
回顾: 物理层 vs. 数据链路层

数据链路层

传输数据帧



数据帧

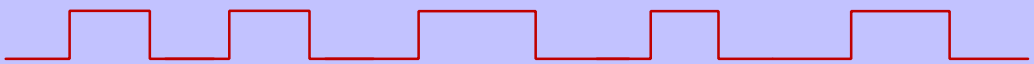


物理层

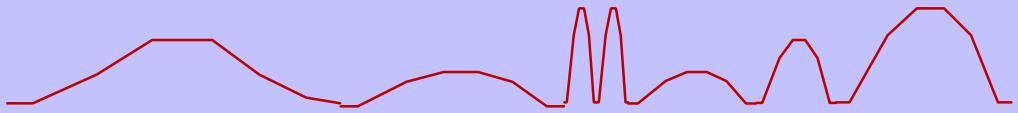
物理层为上层提供透明的比特流传输服务

比特流

数字信号

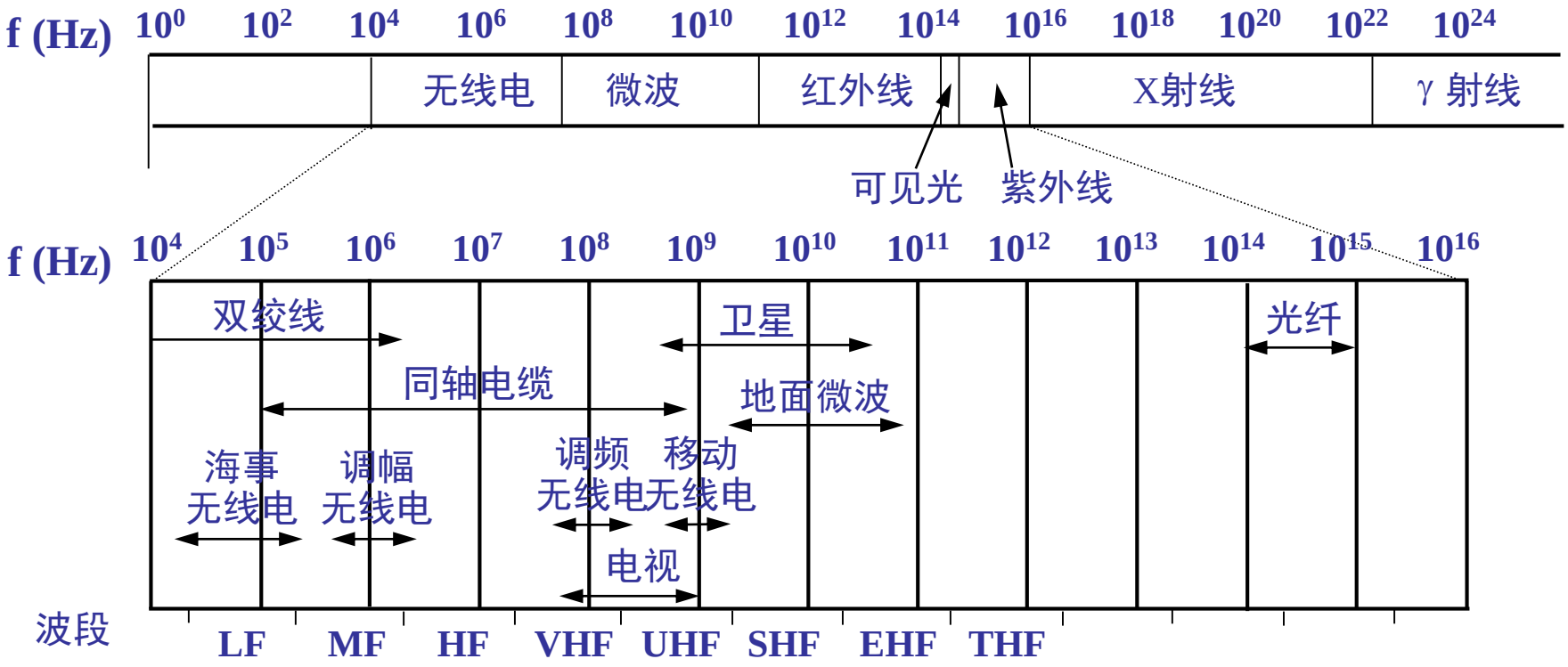


模拟信号



物理介质

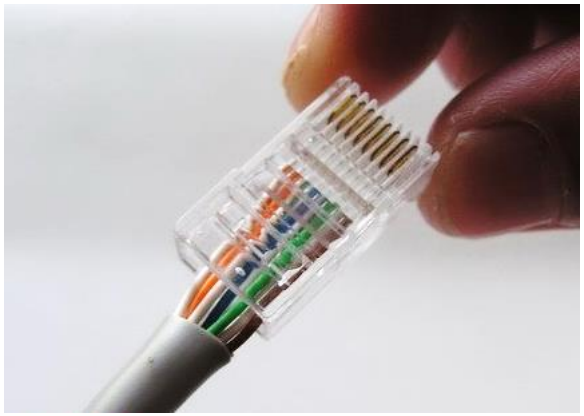
- 导向型介质: 信号在固态介质上传播, 例如同轴电缆, 光纤, 双绞线
- 非导向型介质: 信号自由传播, 例如电磁波



物理介质: 有线

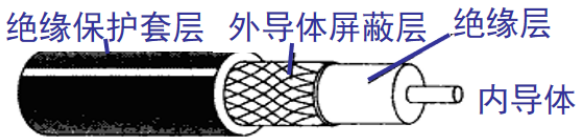
双绞线

- 两根绝缘的铜线逆时针绞合
  - 消除近端串扰(内部)
- 屏蔽双绞线
- 非屏蔽双绞线



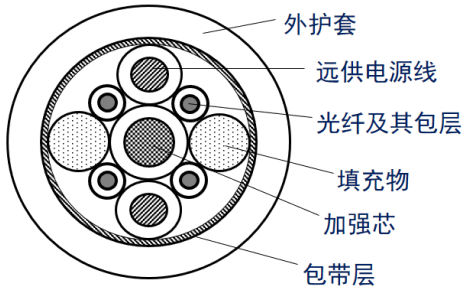
同轴电缆

- 全双工
- 基带
  - 单信道
  - 传统以太网
- 宽带
  - 多信道
  - 混合光纤同轴网



光纤

- 采用玻璃纤维传递光脉冲
  - 一个脉冲代表一个比特
- 原理: 光的全反射
  - 高速
  - 低误码率
  - 远距离中继转发



物理介质：无线

- 通过电磁波携带信号
- 使用的频段很广
- 不存在物理的“线路”
  - 双向
- 易受环境影响:
  - 反射
  - 障碍物
  - 干扰
  - 多径

无线链路类型

- 地面微波
  - 可达到45 Mbps
- 局域网(例如, WiFi)
  - 11Mbps, 54 Mbps
- 较大区域 (例如, 蜂窝网)
  - 4G 蜂窝网: ~ 10 Mbps
- 卫星网
  - 45Mbps
  - 270 ms 端到端时延

数据链路

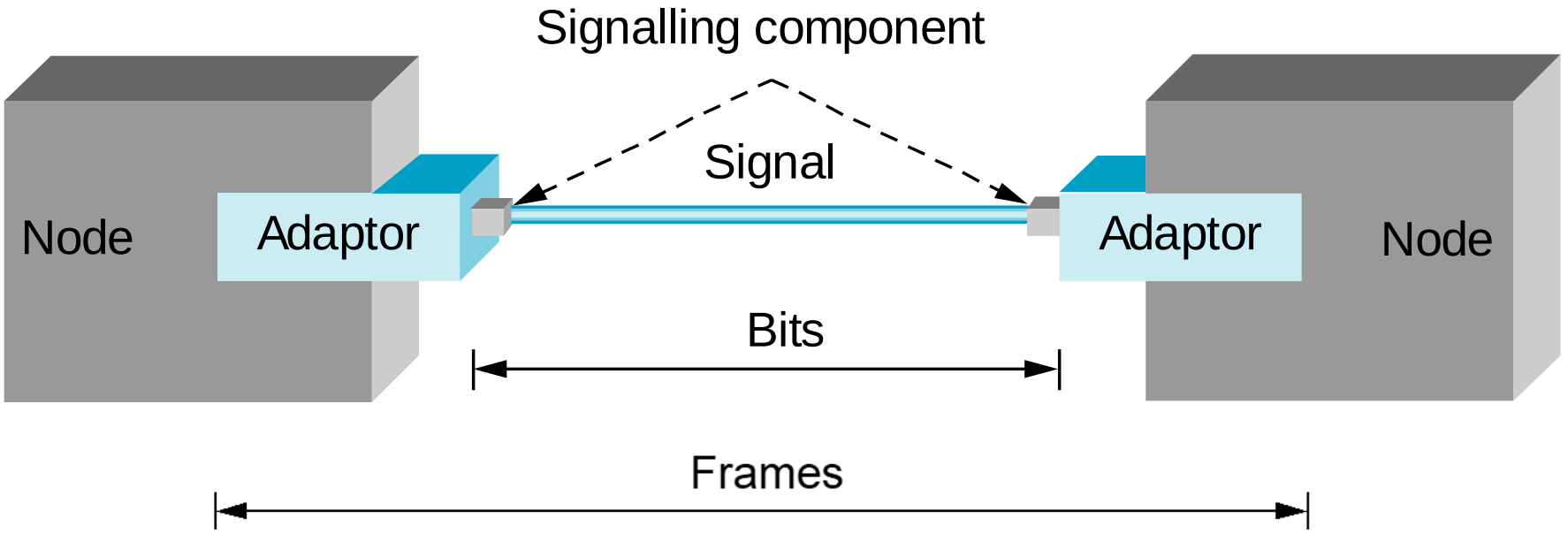
链路(link)

- 一条无源的点到点的物理线路段（加上通信设备），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)

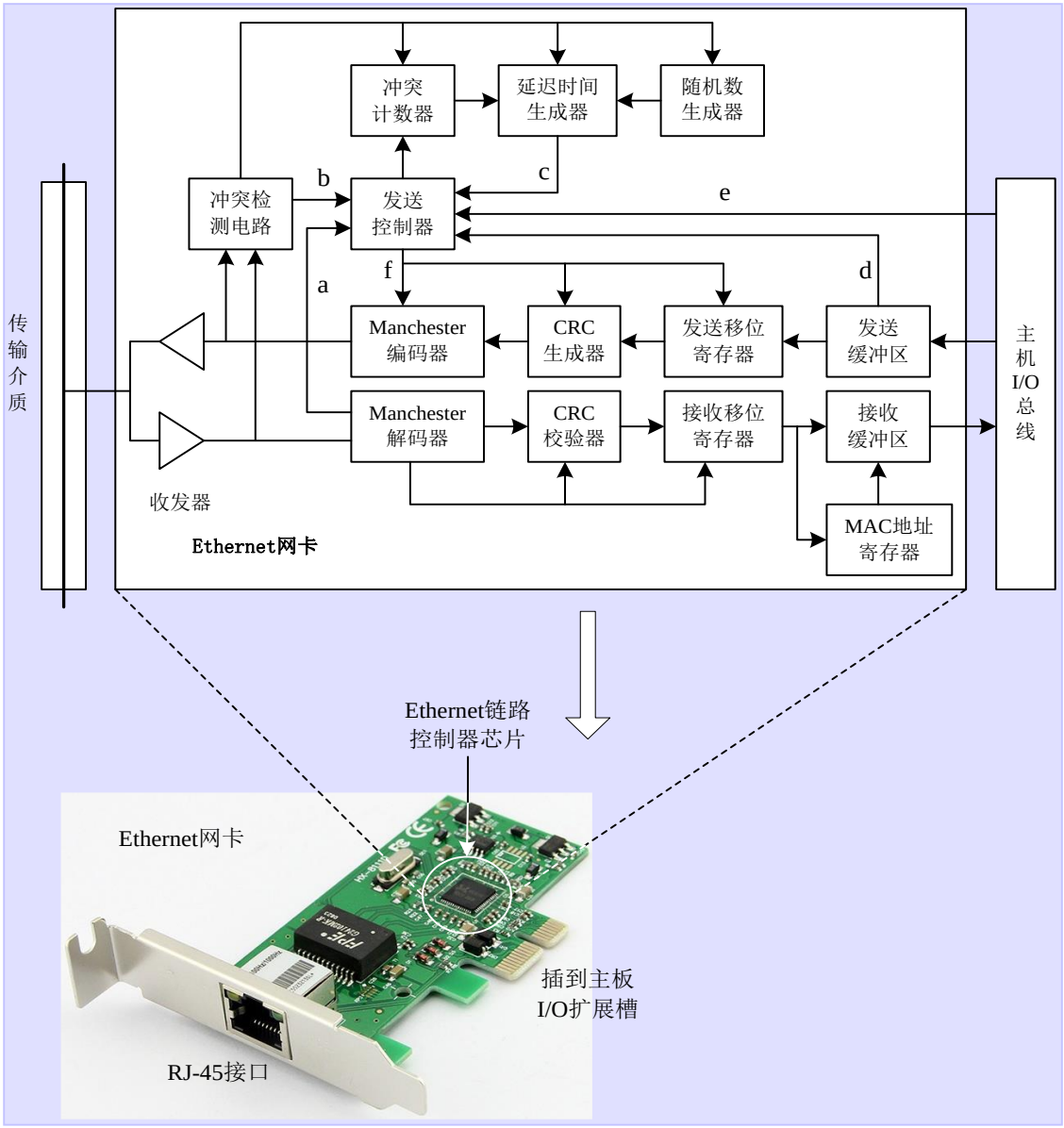
- 除了物理线路外，还必须有协议来控制这些数据的可靠传输。若把实现这些协议的硬件和软件加到链路上，就构成了数据链路。
- 现在最常用的方法是使用适配器（即网卡）来实现这些协议的硬件和软件。
- 一般的适配器包括数据链路层和物理层这两层的功能。

比特和信号



- 信号在信令构件之间传输（物理线路）
- 比特在适配器之间流动（链路/物理链路）
- 数据帧在结点之间流动（数据链路/逻辑链路）

网络适配器


冲突
计数器
延迟时间
生成器
随机数
生成器
冲突检
测电路
b
发送
控制器
c
e
a
f
d
传输介质
Manchester
编码器
CRC
生成器
发送移位
寄存器
发送
缓冲区
主机I/O总线
Manchester
解码器
CRC
校验器
接收移位
寄存器
接收
缓冲区
收发器
MAC地址
寄存器
Ethernet网卡
Ethernet链路
控制器芯片
Ethernet网卡
插到主板
I/O扩展槽
RJ-45接口

网络适配器

功能

- 进行串行/并行转换
- 对数据进行缓存
- 设备驱动程序（数据链路层协议）

接口特性

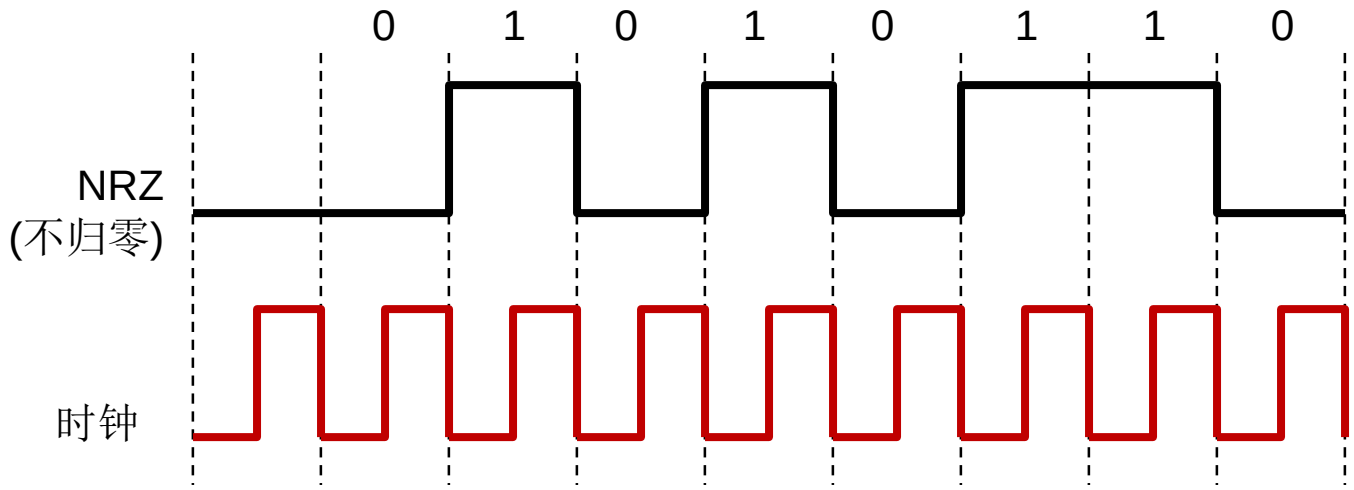
- 机械特性：接口所用接线器的形状和尺寸、引线数目和排列、固定和锁定装置等等
- 电气特性：在接口电缆的各条线上出现的电压的范围
- 功能特性：某条线上出现的某一电平的电压表示何种意义
- 规程特性：对于不同功能的各种可能事件的出现顺序

假设

- 假设处理离散信号(忽略调制细节), 高信号和低信号, 对应编码为1和0
- 收发双方同步, 即, 存在一个时钟进行信号采样
- 如果信号的幅值和持续时间足够大, 接收机可以识别出发送的信号.

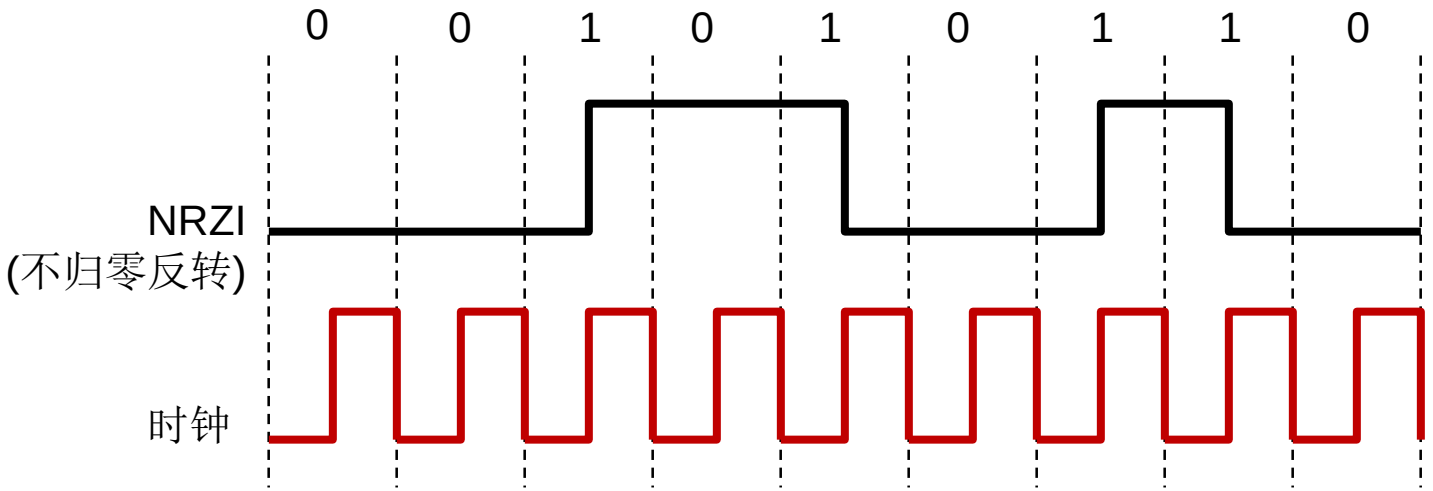
不归零 NRZ (Non-Return to Zero)

- 编码
  - 1 → 高信号; 0 → 低信号
- 问题: 连续的1 或 0
  - 连续的 0可能被误认为没有信号
  - 连续的 1可能导致基线漂移
  - 时钟恢复困难（同步问题）



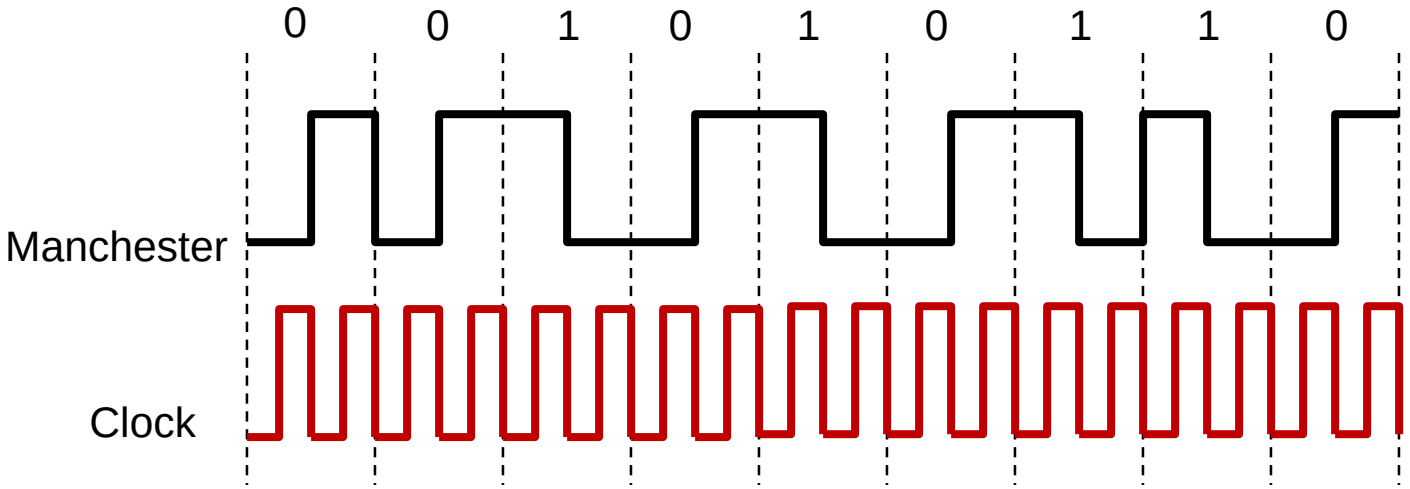
不归零反转 (NRZI)

- 编码
  - 1 → 信号跳变；0 → 信号保持
- 可以解决持续1的问题，未能解决连续 0的问题

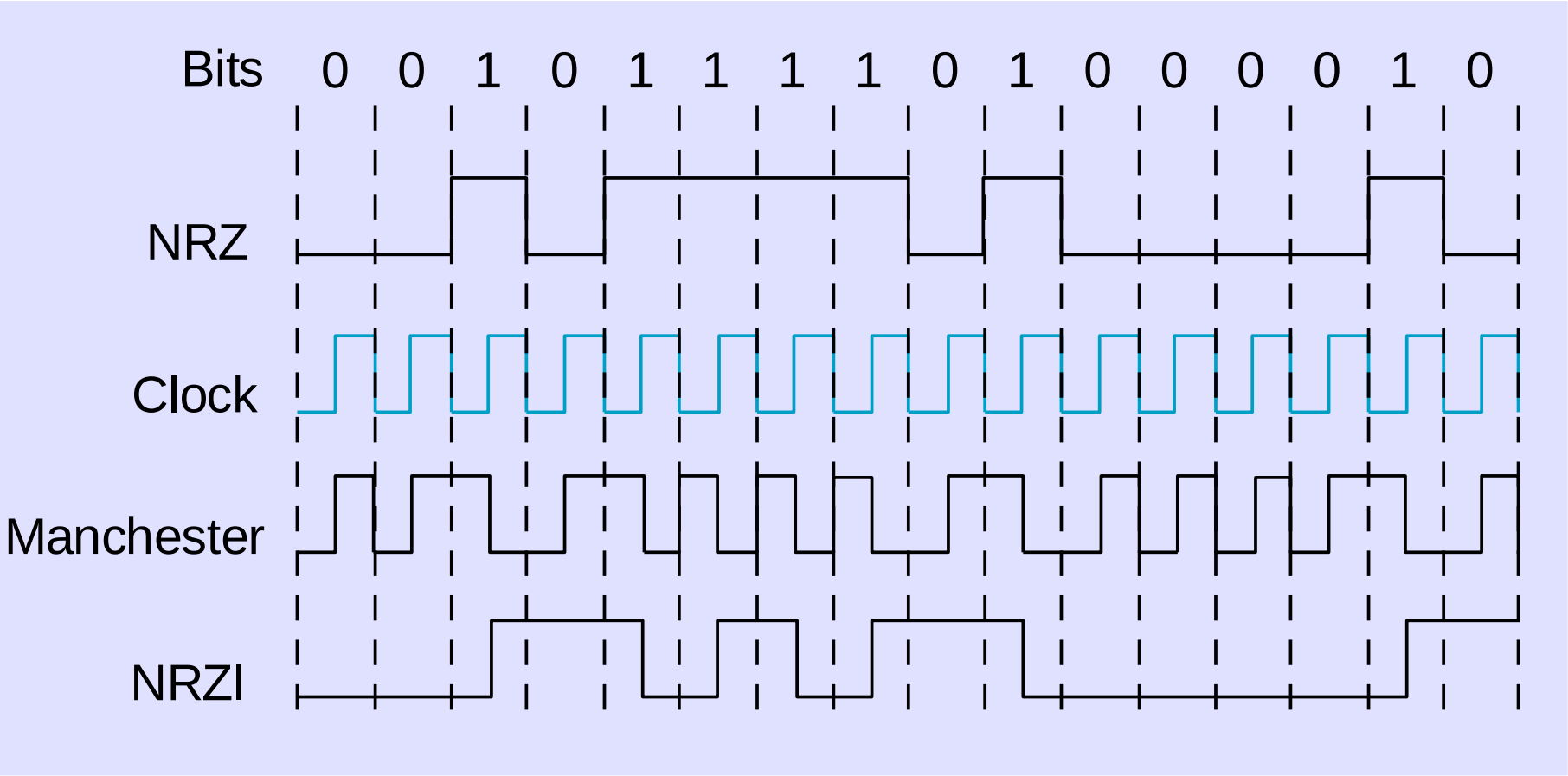


曼彻斯特编码

- 编码
  - 1 → 高到低跳变；0 → 低到高跳变
  - 能有效恢复时钟
- 缺点: 信号跳变速率翻倍
  - 比特率是信号跳变速率的一半
- 编码效率: 50%



图示说明



4-bit/5-bit

- 目标: 解决曼彻斯特编码的低效问题, 同时避免持续的低信号
- 解决方案:
  - 用5个比特对4个比特的数据进行编码, 其中每个代码(5个比特)中最多有1个前导0, 且末端最多有2个0
  - 采用NRZI对5比特的代码进行编码
  - 编码效率: 80%

4-bit	5-bit
0000	11110
0001	01001
0010	10100
0011	10101
0100	01010
0101	01011
0110	01110
0111	01111

4-bit	5-bit
1000	10010
1001	10011
1010	10110
1011	10111
1100	11010
1101	11011
1110	11100
1111	11101

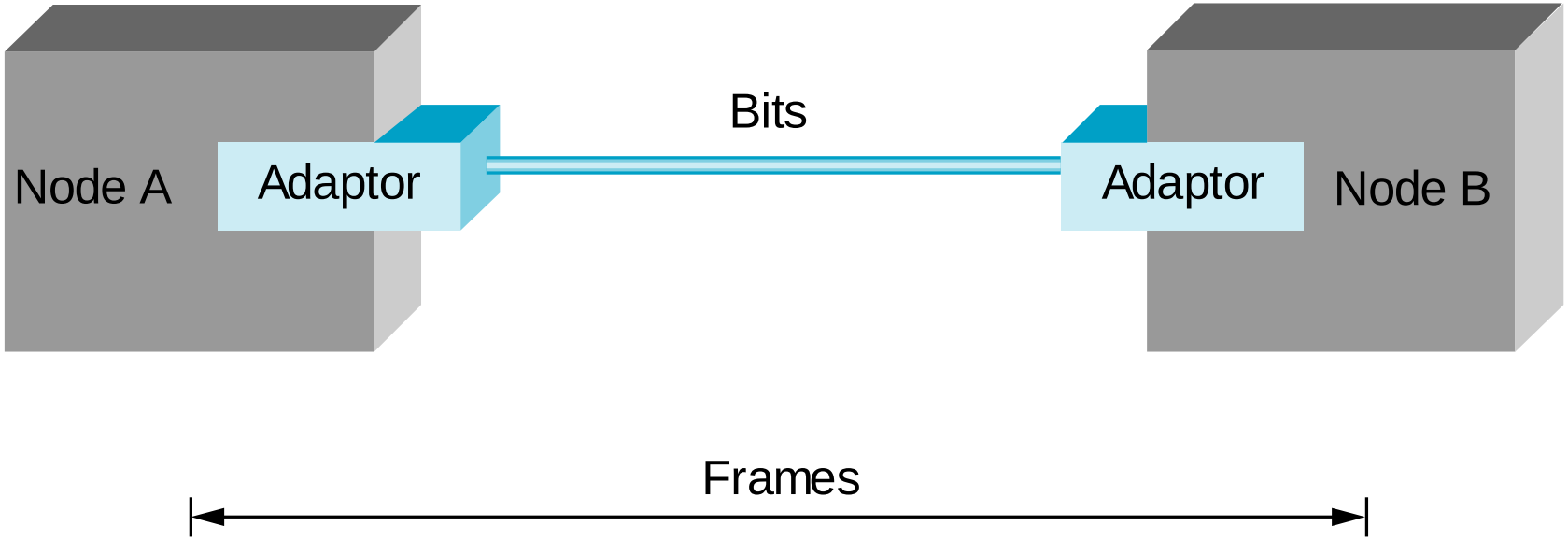
提纲

- 问题: 物理上相连的主机
  - 物理层
    - 数据编码
  - 数据链路层
    - 帧定界
    - 差错检测
    - 可靠传输
  - 介质访问控制子层
    - 多路访问
    - 局域网技术
    - 以太网 (802.3)
    - 无线局域网

点到点链路的四个基本问题

共享链路的问题

帧定界问题

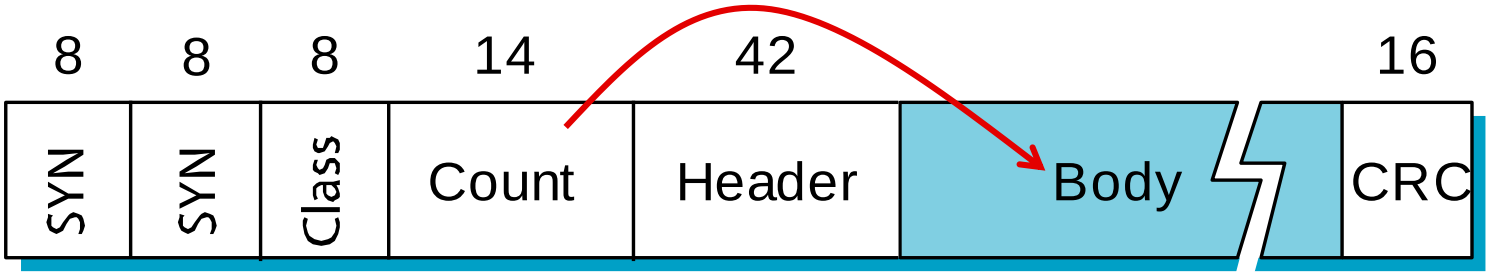


- 两个结点之间的数据传输以块为单位(帧)
- 挑战：准确识别数据帧的开始和结束→帧定界

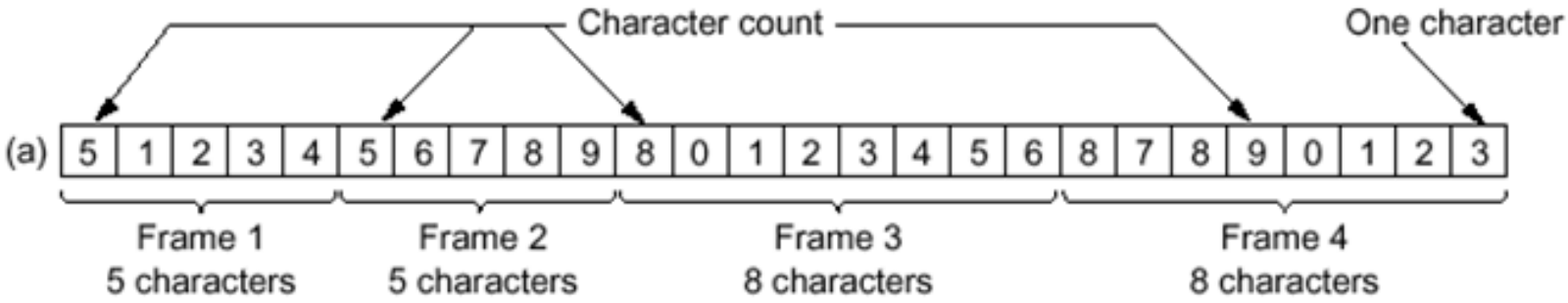
面向字节的协议

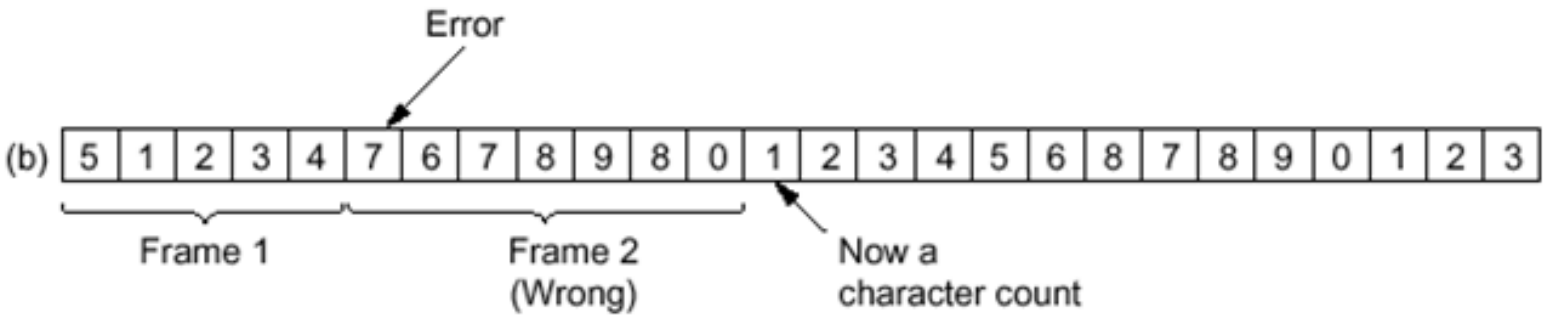
- 面向字节
  - 把每一帧看做一个字节(字符)集合
- 两种方法
  - 字符计数法
  - 起止标记法

字节计数法

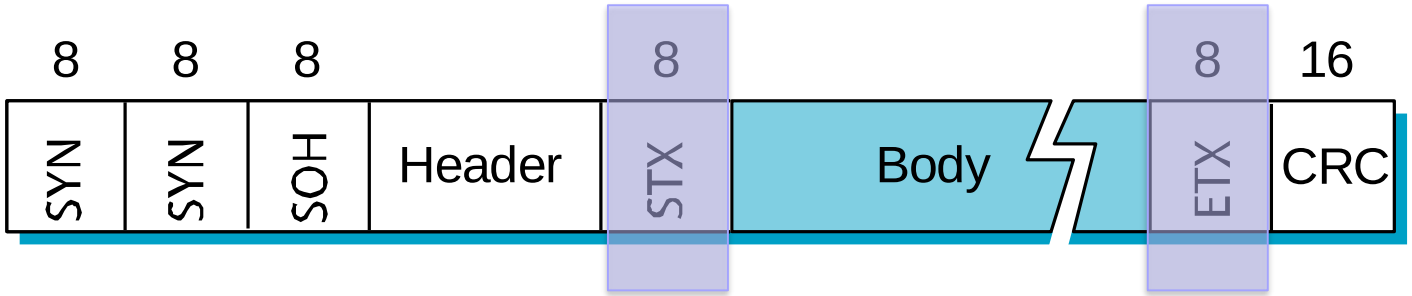


Count: Specifies # of bytes in the body





起止标记法

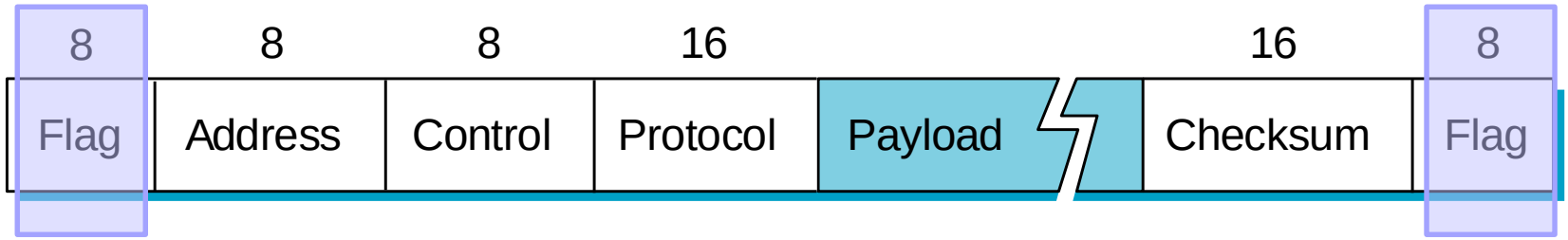


SYN: Synchronization character
SOH: Start of header
STX, ETX: Start of text, End of text
CRC: Cyclic redundancy check

- BISYNC (二进制同步通信)
  - IBM在1960s末期开发
- 特点
  - 起止字符: 开始和结束字符

    STX (0x02, 正文开始符), ETX (0x03, 正文结束符)
- 问题
  - ETX字符可能出现在数据帧的数据部分
- 解决方案： 字符填充
  - 在数据部分的ETX前填充DLE (0x10, 数据链路转义)
  - 数据部分的DLE前也重复填充DLE

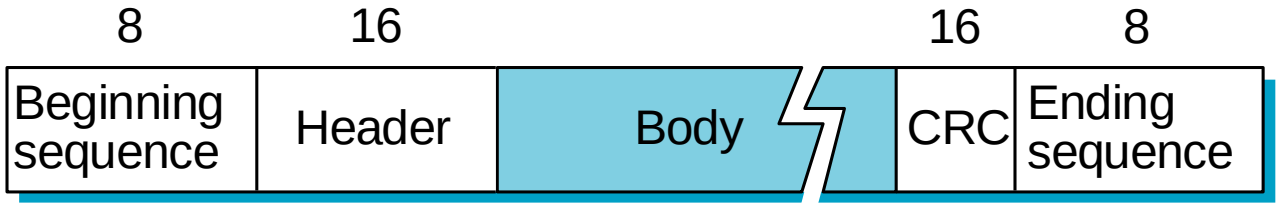
起止标记法



- PPP (Point-to-Point Protocol)
  - IETF, 首次提出在 1992
  - 正式版版本 RFC 1661, 1994
- 特点
  - Flag: 0x7E（对应字符为“~”波浪号）
  - 数据帧的有效载荷字段的长度可以协商, 缺省值为1500字节
  - 链路控制功能：LCP (链路控制协议)，用于启动路线、测试线路、协商参数以及关闭线路
  - 网络控制功能：NCP (网络控制协议)，协商网络参数，用于在连接时刻协商IP地址

面向比特的协议

- 把数据帧看做比特的集合



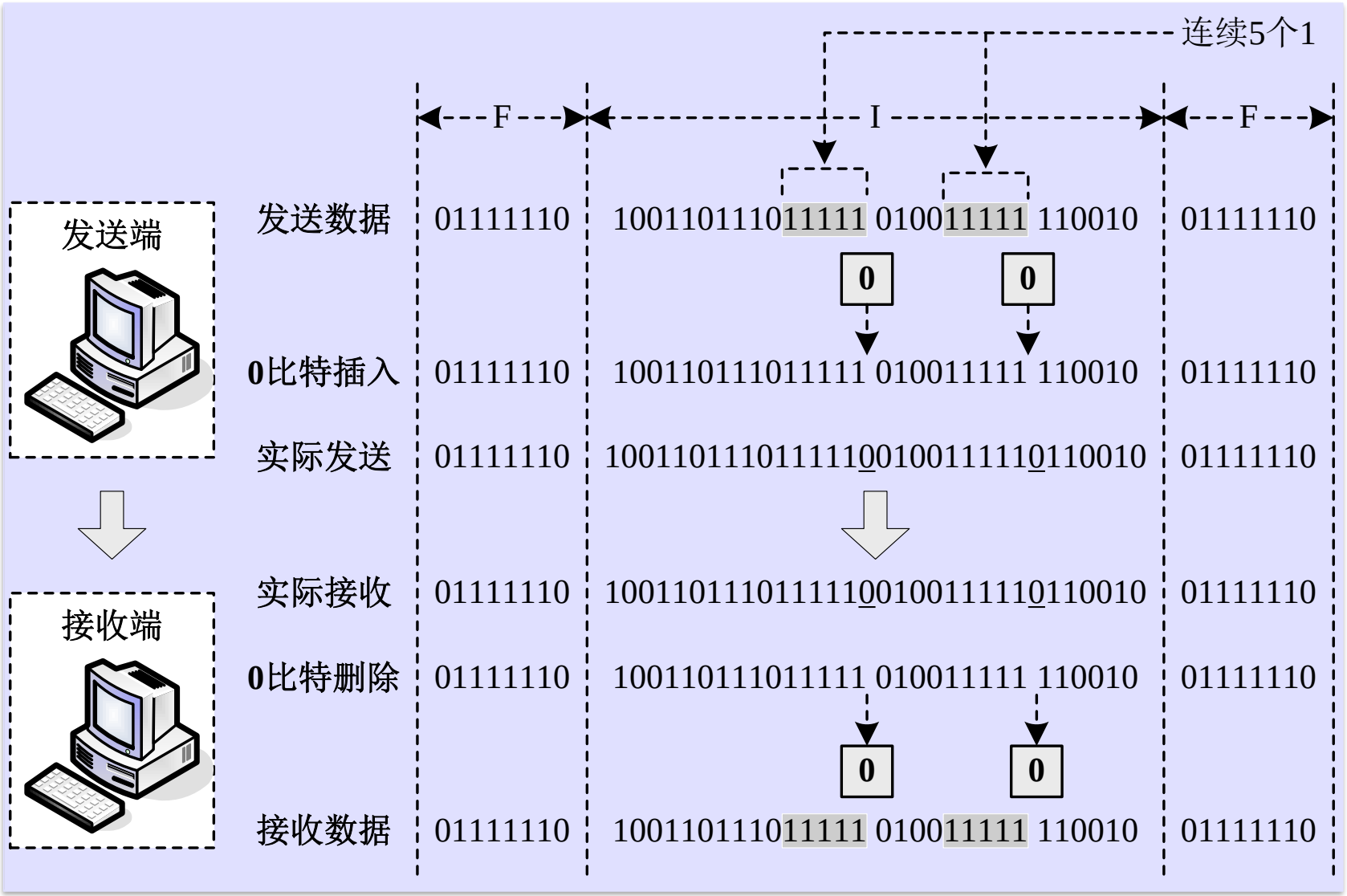
- HDLC (High-Level Data Link Control)
  - 高级链路控制规程
  - 起止比特位串:

    - 01111110
  - 问题: 数据字段可能出现01111110
  - 解决方法：零比特填充

HDLC (高级链路控制规程)

- 发送端，每连续发送5个1
  - 插入一个比特0
- 接收端，每连续收到5个1
  - 如果后续比特为0: 删除0比特
  - 如果后续比特为10: 帧结束
  - 如果后续比特为11: 出错
- 比特填充的特点
  - 可能连续两次接收失败
  - 帧的长度由帧中有效载荷中传送的数据决定

实例


连续5个1
F
I
F
发送端
接收端
发送数据
0比特插入
实际发送
实际接收
0比特删除
接收数据
01111110
100110111011111 010011111 110010
0
100110111011111 010011111 110010
100110111011111 0 10011111 0 110010

提纲

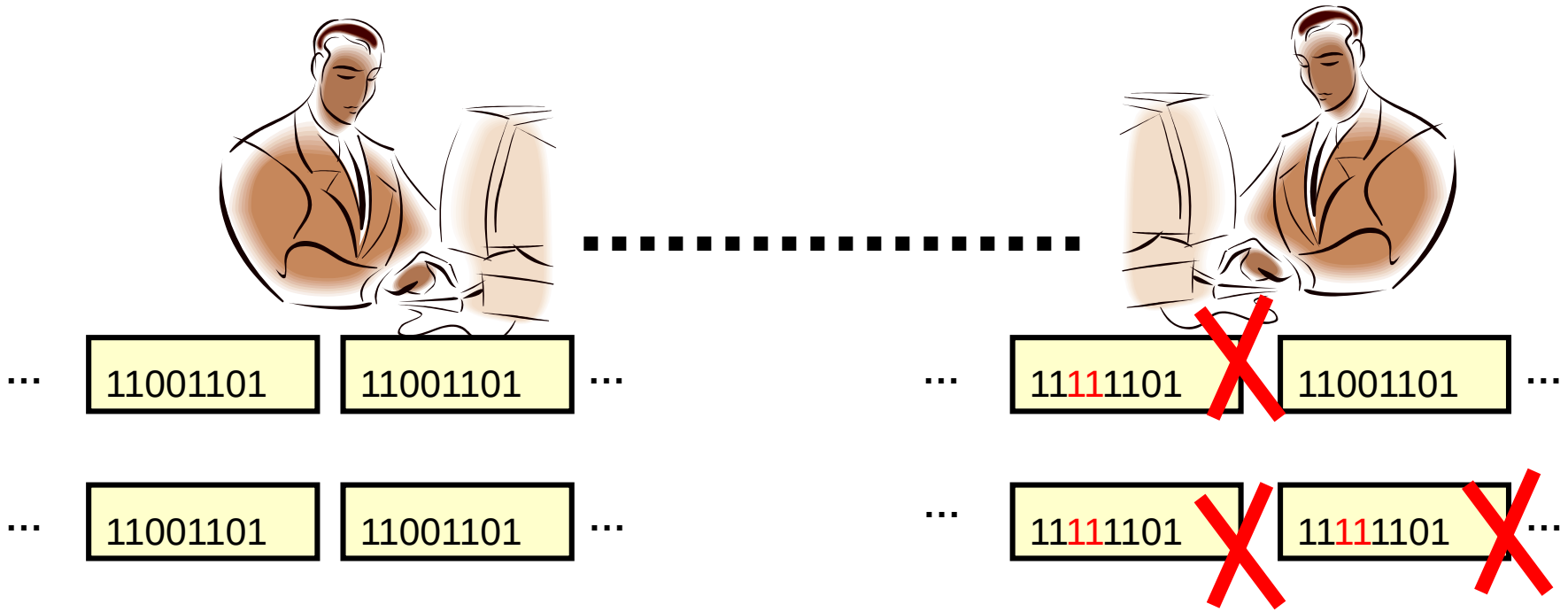
- 问题: 物理上相连的主机
  - 物理层
    - 数据编码
  - 数据链路层
    - 帧定界
    - 差错检测
    - 可靠传输
  - 介质访问控制子层
    - 多路访问
    - 局域网技术
    - 以太网 (802.3)
    - 无线局域网

点到点链路的四个基本问题

共享链路的问题

直连网络的研究问题

- 差错检测
  - 检测帧传送过程种可能出现的错误，并采取相应的动作
- 可靠传输
  - 保证链路在帧不时可能出现错误情况下的可靠性



比特错误

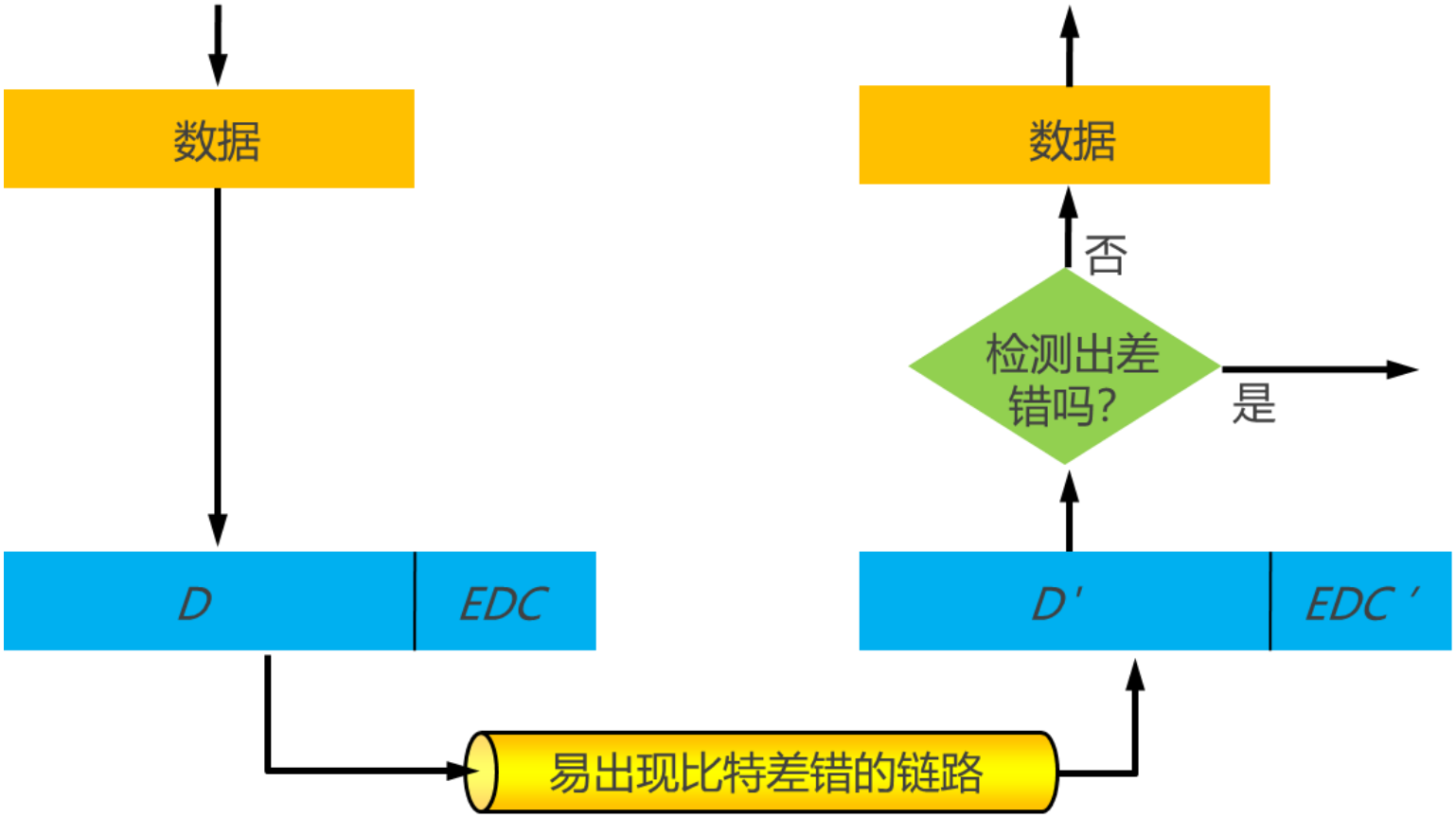
问题

- 热噪声和冲击噪声可能导致比特错误
- 热噪声(介质热运动)导致随机差错
- 冲击噪声(电磁干扰)导致突发差错

解决方案

- 差错检测
  - 接收方可以通过编码方式检测到差错→校验
- 差错纠正, 通常有两种策略:
  - 接收方通知发送方重发消息→请求重发
  - 接收方重新构造消息→纠错码

差错检测的基本原理



EDC/ECC = 检错和纠错的比特(冗余填充)
D = 受差错保护的数据部分，包括首部字段

差错检测的基本原理

- 主要目标
  - 冗余度
    - n = 数据长度, k = EDC长度
    - k 远小于 n
  - 检错概率
    - 概率最大化
- 差错检测并不一定100%可靠！
  - 协议可能出现错误(极少数)
  - k 越大检错效果越好

差错检测

❑ 差错检测通用方法（链路层）

- ➢ 二维奇偶校验(数据链路层协议)
- ➢ 循环冗余校验, CRC(数据链路层协议)

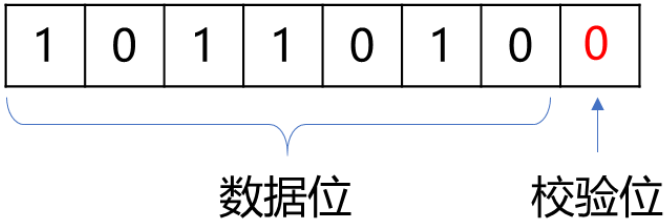
组帧原则	链路层协议	检错及纠错
面向字节	BISYNC	ASCII char: 二维奇偶校验 EBCDIC char: CRC
	PPP	CRC (名字称为校验和)
	DDCMP	CRC
面向比特	HDLC	CRC

二维奇偶校验

基本的单比特奇偶校验

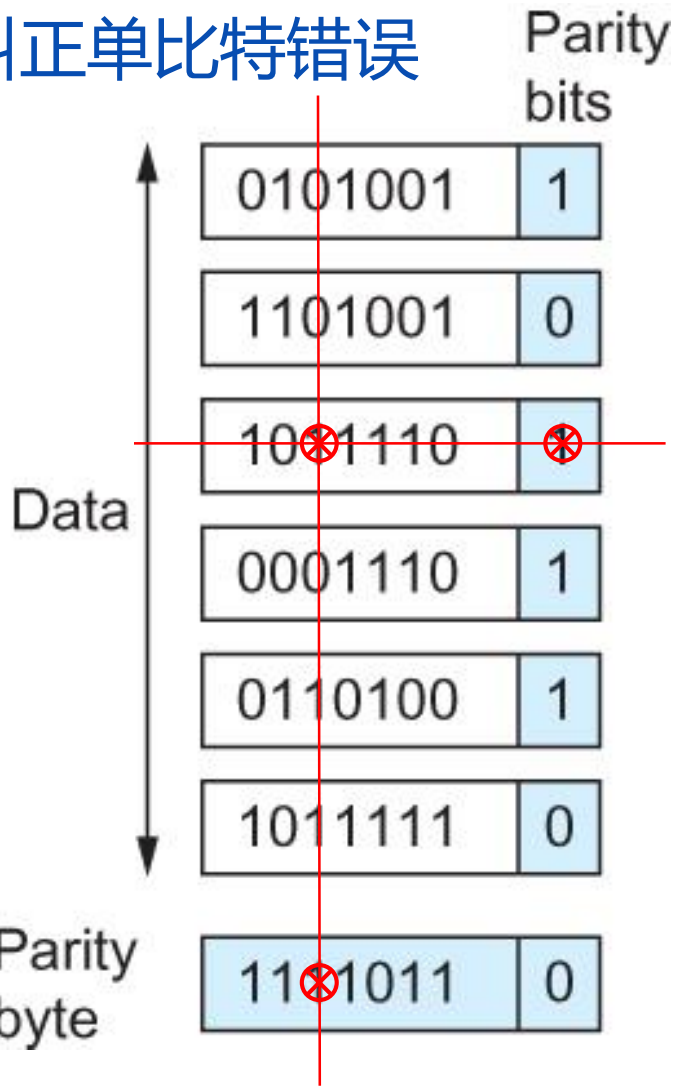
❑ 单比特校验

- ➢ 检测奇数个比特错误
- ➢ 奇数校验
  - ● 根据字节中1的个数置位第8比特使1的个数为奇数
- ➢ 偶数校验
  - ● 根据字节中1的个数置位第8比特使1的个数为偶数



二维奇偶校验

- 可以检测并纠正单比特错误



循环冗余校验

❑ 原理

- ➢ 以一个称为有限域的数学分支为依据

❑ 数学表述

- ➢ 待发送数据
  - ●n-比特的数据看做 n-1 次的多项式$M(x)$
    例如: 对于10011010, $M(x)= x^7+x^4+x^3+x^1$
- ➢ 检错码
  - ●表示为 $(n+k-1)$ 次的多项式 $P(x)$
  - ●基于k次的除数多项式 $C(x)$ 进行计算
    例如: $C(x)= x^3+x^2+1$, 其中$k=3$
  - ●除数多项式收发双方提前确定

循环冗余校验

- 错误条件
  - 接收方判断$P(x)$ 是否可以整除$C(x)$
- 如何计算 $P(x)$
  - $M(x)$乘以x^k 得到$T(x)$
    - 相当于在$M(x)$后面添加 k 个 0
  - $T(x)$除以$C(x)$，得到余数$R(x)$
    - $R(x)$为 k 位，不足 k 位前面用0补齐
    - $R(x)$即为CRC校验码
  - $T(x)$减去$R(x)$
    - 除法和减法均为模2运算，相当于XOR操作

模2运算

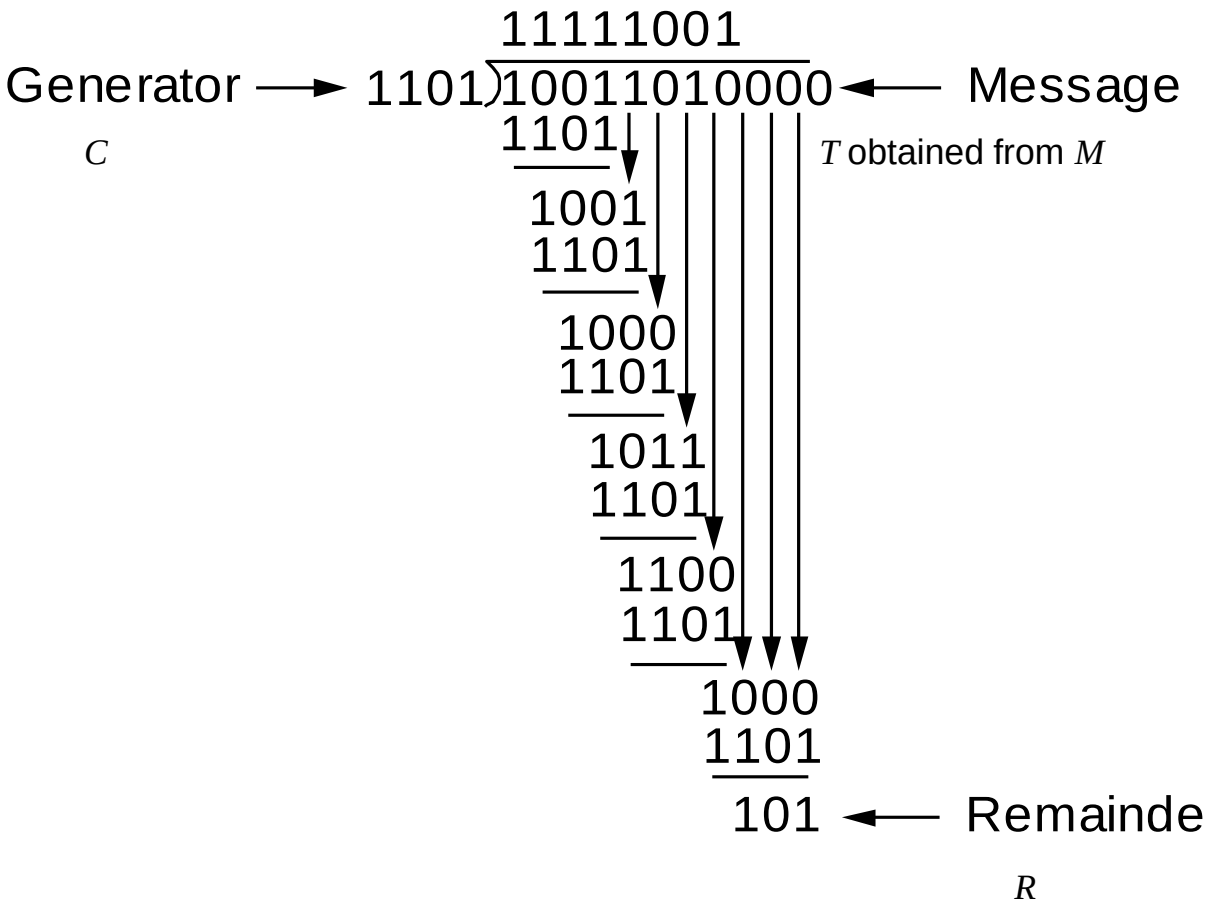
- 类似于二进制运算，但减法不借位、加法不进位
  - 示例:

a	b	a⊗ b
0	0	0
0	1	1
1	0	1
1	1	0

```
101 +      101 +      1011 +          101 -      101 -      1011 -
010        001        0111            010        001        0111
---        ---        ----            ---        ---        ----
111        100        1100            111        100        1100
```

- 模2运算中的加法和减法等同于XOR操作

循环冗余校验



$M(x)$ = 10011010

$C(x) = x^3 + x^2 + 1$

$T(x)$ = 10011010 | 000

$R(x)$ = 101

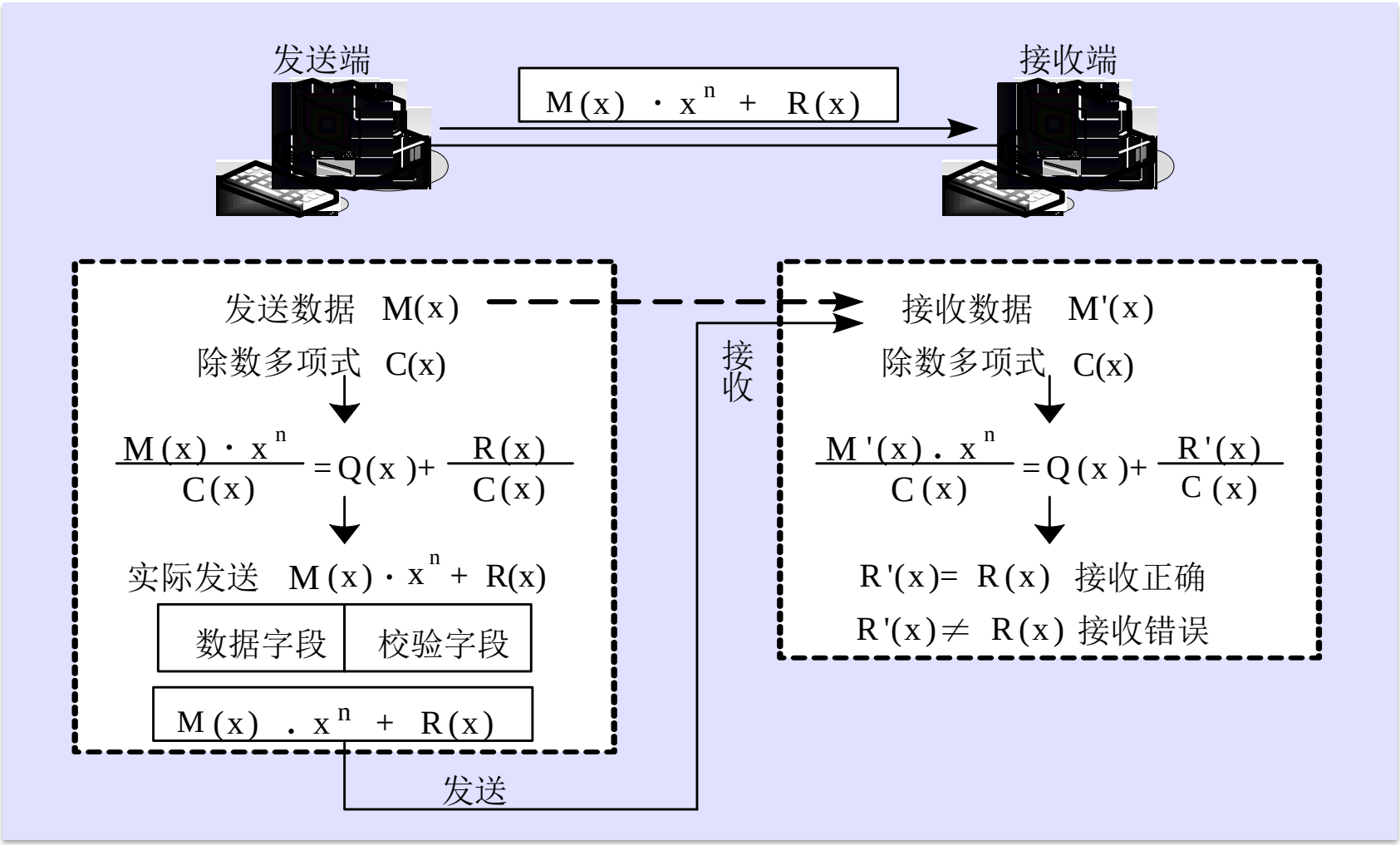
$P(x) = M(x)$ XOR $R(x)$

= 1001 1010 000 XOR 101

= 1001 1010 101

$P(x)$ = 10011010 | 101

循环冗余校验示意图


发送端
M(x) · x^n + R(x)
接收端
发送数据 M(x)
除数多项式 C(x)
M(x) · x^n / C(x) = Q(x) + R(x) / C(x)
实际发送 M(x) · x^n + R(x)
数据字段
校验字段
M(x) · x^n + R(x)
发送
接收
接收数据 M'(x)
除数多项式 C(x)
M'(x) · x^n / C(x) = Q(x) + R'(x) / C(x)
R'(x)= R(x) 接收正确
R'(x)≠ R(x) 接收错误

循环冗余校验

- 选择除数多项式
  - 不同的C(x)可以检测出特定的比特错误
  - 合理的选择除数多项式可以实现：
    - 检测出所有1 & 2-比特的错误
    - 任意奇数个错误
    - 任何小于 k 比特的连续比特错误序列
    - 部分大于 k 比特的连续比特错误序列
- 硬件实现
  - 在网卡实现
  - 使用一个k比特移位寄存器和若干个XOR门

循环冗余校验

❑ 常用的CRC除数多项式

CRC	C(x)	Link Protocol
CRC-8	$x^8 + x^2 + x^1 + 1$	ATM
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^1 + 1$	ATM
CRC-12	$x^{12} + x^{11} + x^3 + x^2 + 1$	
CRC-16	$x^{16} + x^{15} + x^2 + 1$	
CRC-CCITT	$x^{16} + x^{12} + x^5 + 1$	HDLC
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$	ATM, Ethernet, 802.5

提纲

- 问题: 物理上相连的主机
  - 物理层
    - 数据编码
  - 数据链路层
    - 帧定界
    - 差错检测
    - 可靠传输
  - 介质访问控制子层
    - 多路访问
    - 局域网技术
    - 以太网 (802.3)
    - 无线局域网

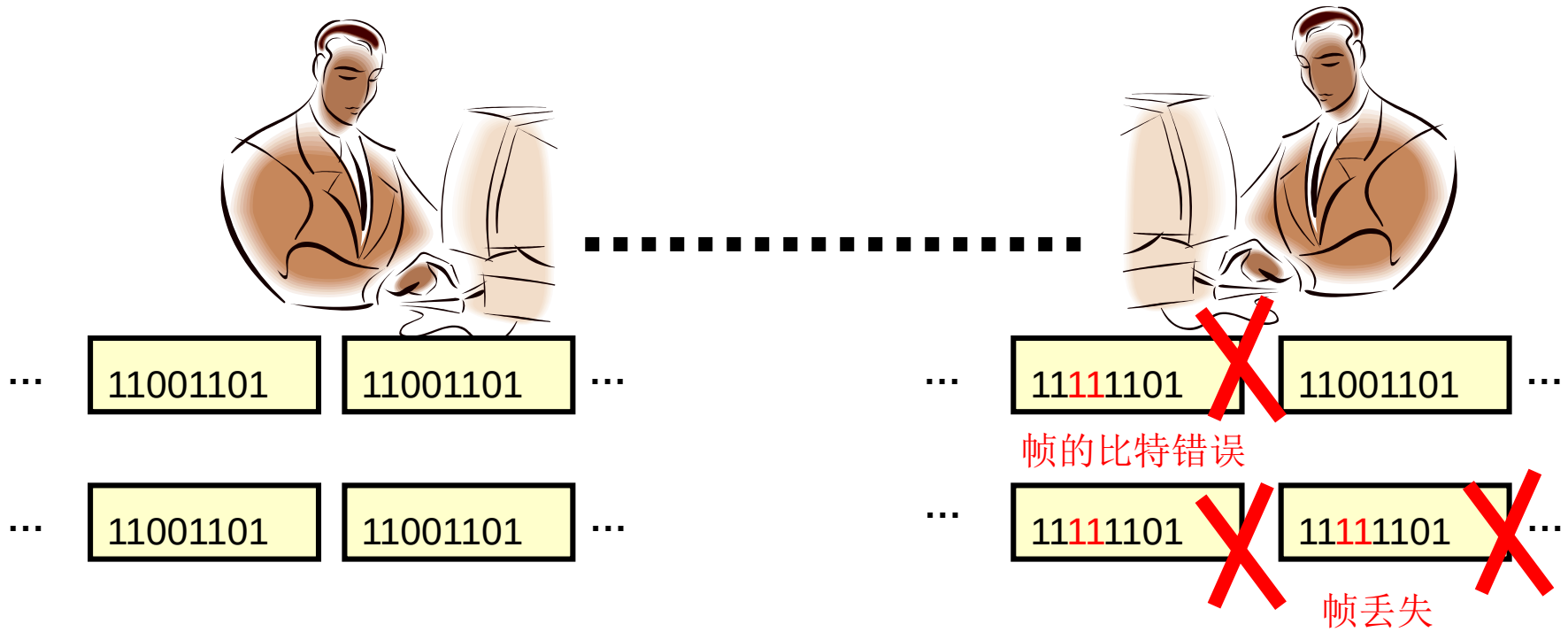
点到点链路的四个基本问题

共享链路的问题

直连网络中的研究问题

❑ 可靠传输

- ➢ 在时常可能发生错误的物理链路上建立一条可靠的数据链路



可靠传输

问题

- 数据帧可能出现比特错误

基本解决方案

- ACK (确认): 接收方根据差错检测的结果向发送方发送一个小的控制帧进行确认
  - ACK = 正确接收, NAK = 帧错误

可靠传输

问题

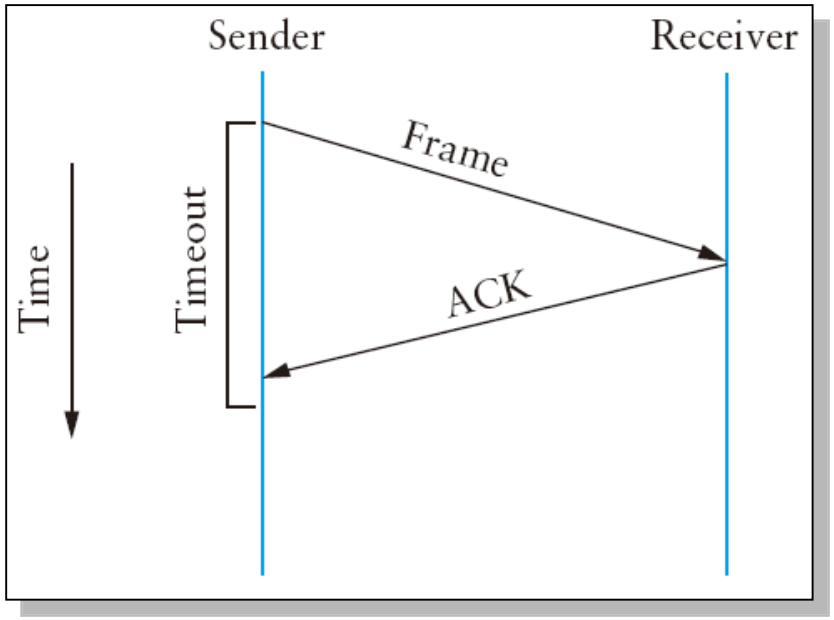
- 数据帧可能出现比特错误
- 数据帧整个被破坏(帧丢失)

基本解决方案

- ACK (确认): 接收方根据差错检测的结果向发送方发送一个小的控制帧进行确认
  - ACK = 正确接收, NAK = 帧错误
- 超时: 如果发送方在一定的时间范围内未收到来自接受方的确认, 则重传数据帧

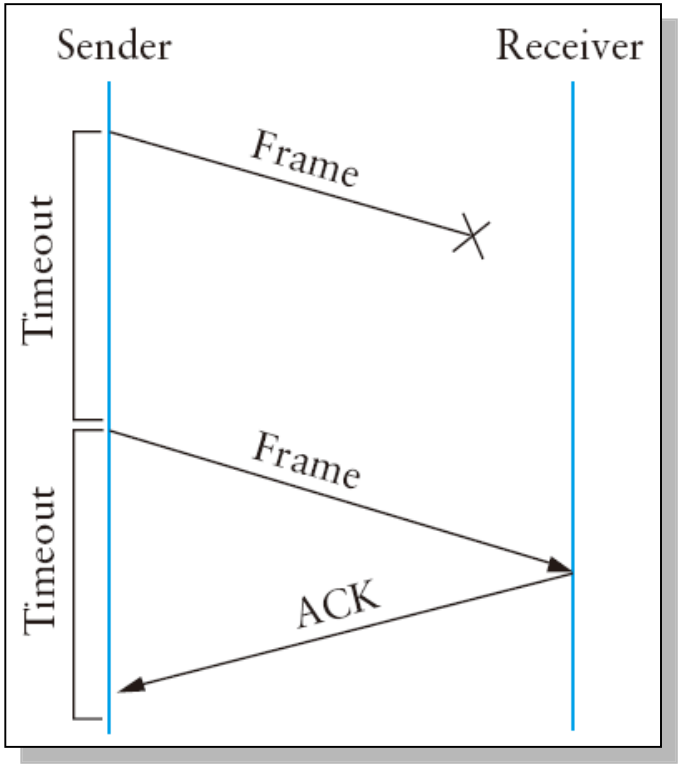
停止等待协议

- 最简单的ARQ(Automatic Repeat Request)机制
- 每发送完一个数据帧，发送方在继续发送下一个数据帧之前必须等待确认
- 如果在一定的时间范围内发送方确认未收到确认，则发送超时定时器过期，发送方重传原始数据帧.

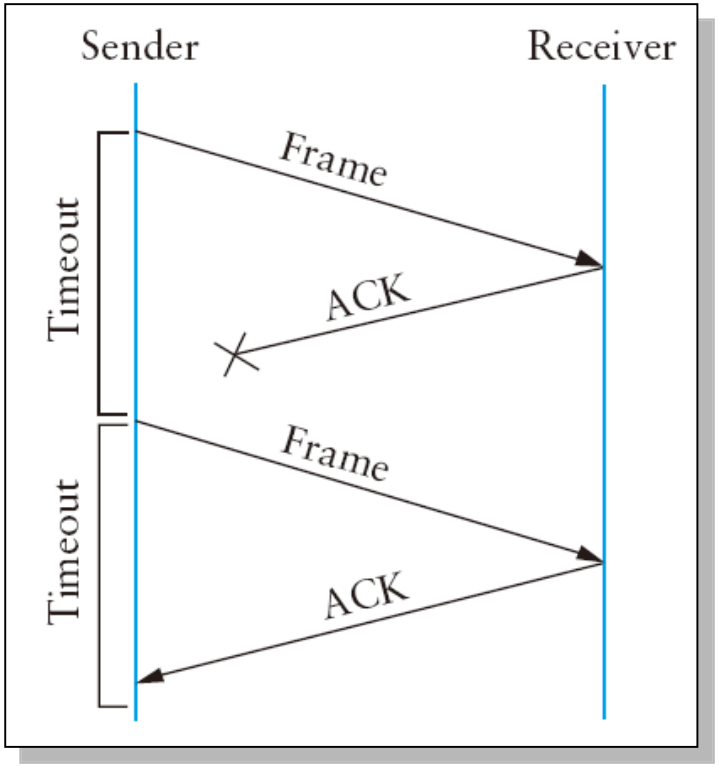


停止等待协议

- 两种不同的帧丢失情况

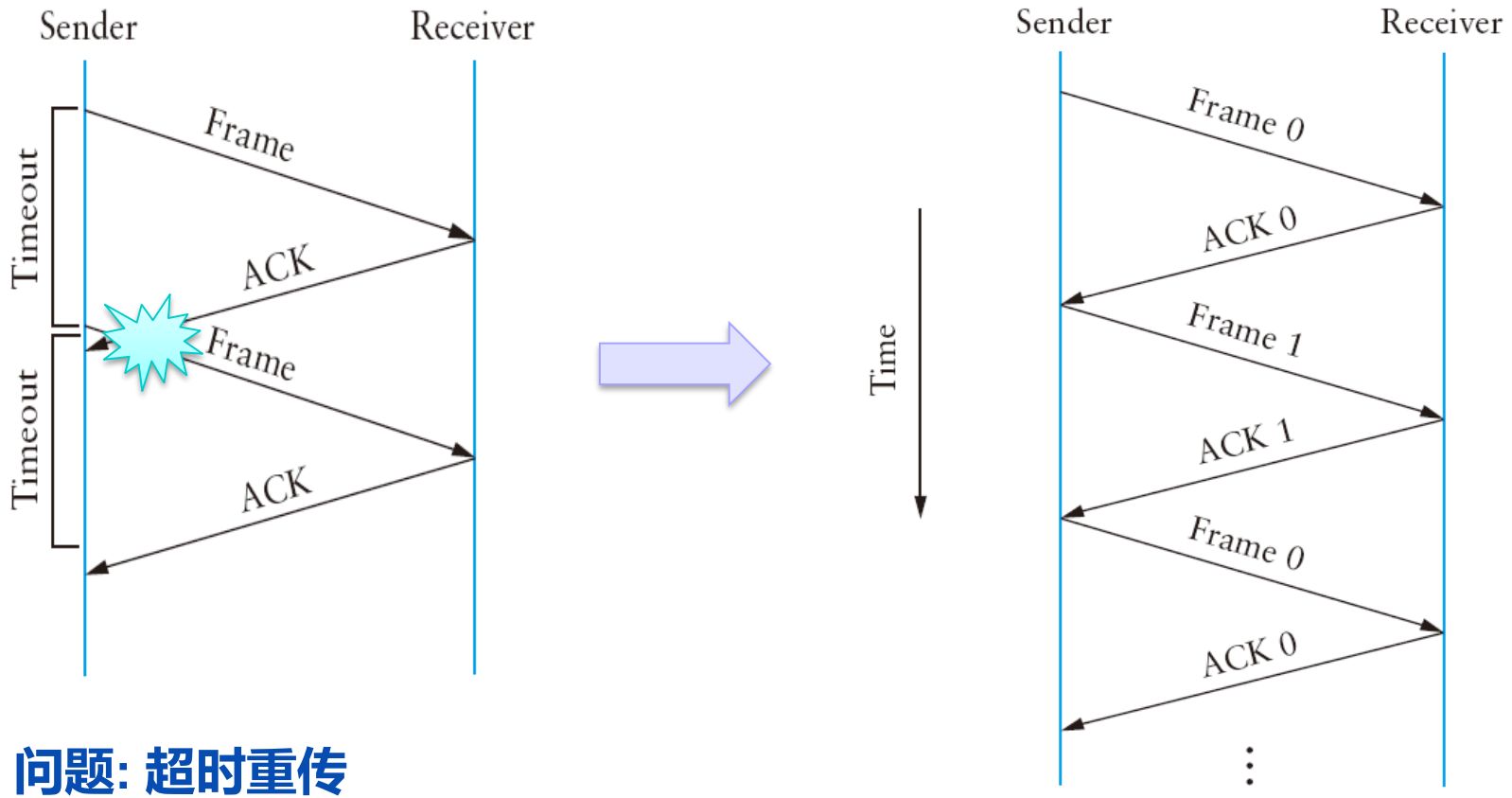


数据帧丢失



ACK丢失

停止等待协议



- **问题: 超时重传**
- **解决方案: 帧序号 (SeqNum)**
  - 避免确认丢失导致的重复数据帧
  - 在有线传输介质中, 只需1比特

可靠传输

问题

- 数据帧可能出现比特错误
- 数据帧整个被破坏(帧丢失)

基本解决方案

- ACK (确认): 接收方根据差错检测的结果向发送方发送一个小的控制帧进行确认
  - ACK = 正确接收, NAK = 帧错误
- 超时: 如果发送方在一定的时间范围内未收到来自接受方的确认, 则重传数据帧

可靠传输

问题

- 数据帧可能出现比特错误
- 数据帧整个被破坏(帧丢失)

基本解决方案

- ACK (确认): 接收方根据差错检测的结果向发送方发送一个小的控制帧进行确认
  - ACK = 正确接收, NAK = 帧错误
- 超时: 如果发送方在一定的时间范围内未收到来自接受方的确认, 则重传数据帧
- 帧序号: 识别数据帧

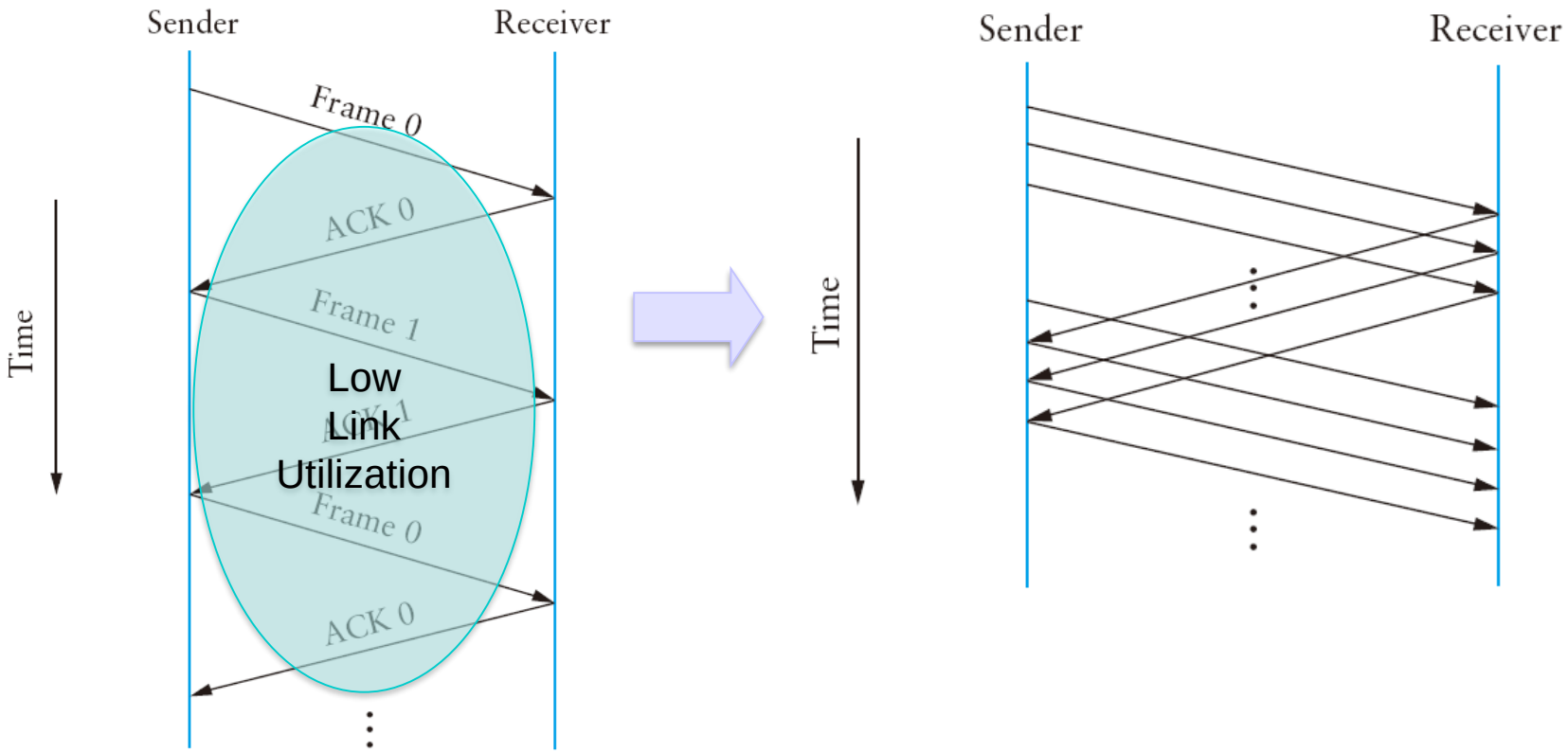
自动请求重传 (ARQ)

- ARQ
  - 采用确认和超时定时器的可靠传输机制
- 链路层假定
  - 串行通信信道 – 传输过程中不存在帧的乱序
  - 所有数据帧的传播时延相同
- 接收方
  - 对数据帧进行差错检测
  - 对正确帧进行确认，丢失错误帧
  - 丢弃禁止接收的数据帧
- 发送方
  - 发送原始数据帧，暂时保留已发送帧的副本
  - 对错误帧和丢失帧进行重传

停止等待协议

- 确定
  - 链路带宽利用率较低
  - 示例
    - 链路带宽为2Mbps, RTT为45ms, 数据帧大小为1.5KB
    - 每一个RTT内, 发送方仅能发送一个数据帧
    - 吞吐量为1.5x1024x8/0.045 = 273.07kbps
- 改进的动机
  - 充分利用管道传送!
  - 带宽时延积，keep the pipe full

连续ARQ协议

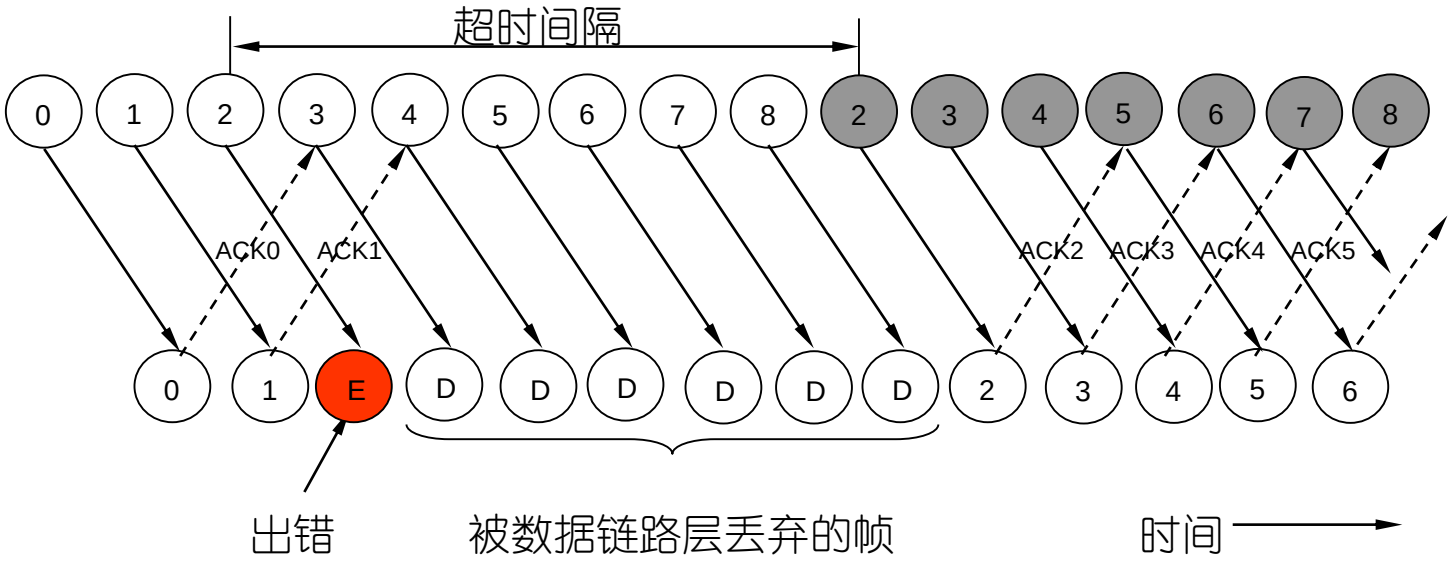


- 问题: 链路带宽利用率较低
- 解决方案: 采取流水线设计，允许未收到确认，连续发送数据帧

连续ARQ协议的两种策略

❑ Go-Back-N

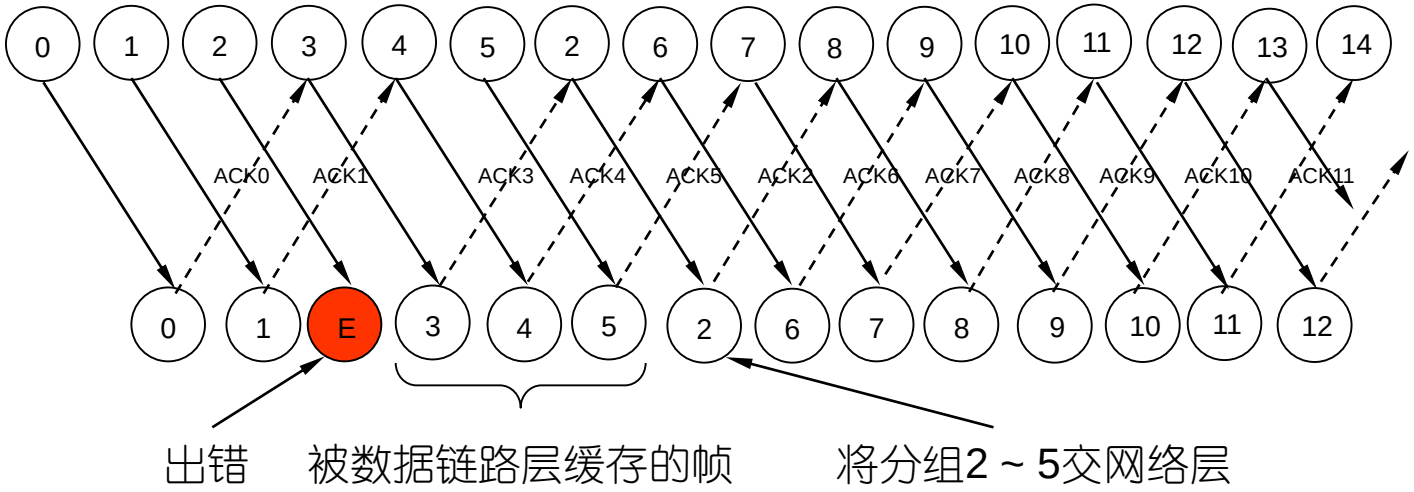
- 一次性发送N个数据帧
- 如果第k个帧丢失, 则对从k帧开始之后的所有帧重传
- 优点: 接收方不需要缓存待交付网络层的帧
- 缺点: 正确帧也可能被重传



连续ARQ协议的两种策略

❑ 选择重传

- 一次性发送N个数据帧
- 如果第k个帧丢失, 仅重传第k个帧
- 接收方对每一个帧进行确认
- 优点: 链路利用率较高
- 缺点: 接收方需要缓存待交付网络层的帧

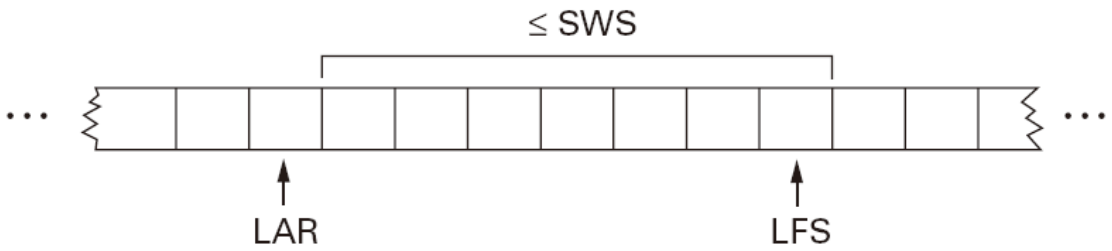


滑动窗口协议

- 引入滑动窗口对收发行为进行控制
- 发送方
  - 发送窗口大小: 发送方在未收到确认前能够连续发送的数据帧的最大个数
  - 发送方在未收到确认前最多可以发送多个数据帧 (受限于发送窗口大小)
  - 对未确认的数据帧缓存
- 接收方
  - 接收窗口大小: 接收方期望连续接收的数据帧的最大个数
  - 接收方通过ACK告知发送方下一次期望其传送的数据帧编号，避免每一次确认

滑动窗口协议

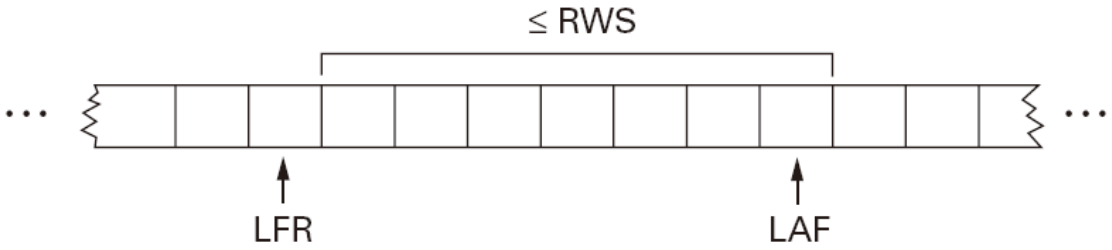
❑ 发送方



$$LFS - LAR \leq SWS$$

SWS = 发送窗口大小
LAR = 最后收到的ACK中的SeqNum
LFS = 最后发送的数据帧的SeqNum

❑ 接收方



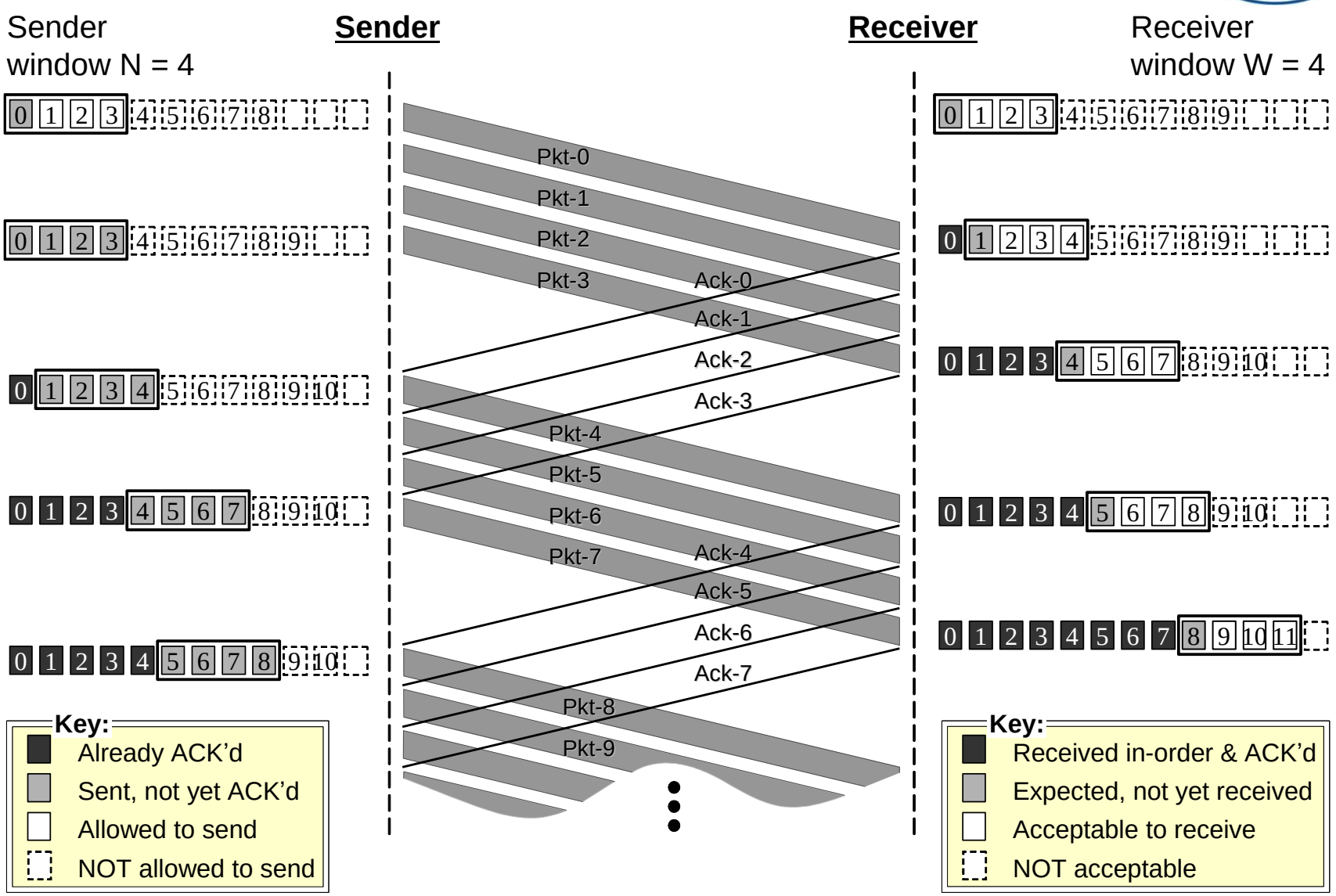
$$LAF - LFR \leq RWS$$

RWS = 接收窗口大小
LAF = 允许接收的数据帧的最大SeqNum
LFR = 最后接收的数据帧的SeqNum

滑动窗口协议

- 接收方: 接收数据帧
  - 如果一个数据帧到达(帧序号为SeqNum)
  - 如果SeqNum ≤ LFR 或SeqNum > LAF,
    - 数据帧落在接收窗口外, 则丢弃该数据帧.
  - 如果LFR < SeqNum ≤ LAF,
    - 数据帧落在接收窗口内, 则接收.
- 接收方: 回复ACK
  - SeqNumToAck = 未确认数据帧的最大SeqNum
  - 接收方确认SeqNumToAck之前的数据帧, 即更大序号的数据已接收.
  - 设置LFR = SeqNumToAck , 调整LAF = LFR + RWS

滑动窗口ARQ


Sender
window N = 4
Sender
Receiver
Receiver
window W = 4
0 1 2 3 4 5 6 7 8
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11
Pkt-0
Pkt-1
Pkt-2
Pkt-3
Ack-0
Ack-1
Ack-2
Ack-3
Pkt-4
Pkt-5
Pkt-6
Pkt-7
Ack-4
Ack-5
Ack-6
Ack-7
Pkt-8
Pkt-9
Key:
Already ACK'd
Sent, not yet ACK'd
Allowed to send
NOT allowed to send
Key:
Received in-order & ACK'd
Expected, not yet received
Acceptable to receive
NOT acceptable

滑动窗口算法的讨论

- 问题一：帧的序号空间是有限的
  - 帧序号需要由首部中的特定字段进行说明
    - 例如：用3比特字段来说明帧序号，则最多8个帧序号可用
  - 序号字段的可用长度限制了可支持的序号空间大小
- 解决方案：
  - 窗口的大小(w比特字段)应与帧序号空间保持一定关系
    - 发送窗口大小 + 接收窗口大小 ≤ 2^w
- 问题二：接收方如何确认ACK
  - 逐一回复每个数据帧，效率比较低
- 两种典型的情况：
  - Go-back-N 回退N帧
  - Selective ACK 选择性重传

Go-back-N

思路

- 发送方的发送窗口($\leq 2^w-1$)，一次性发送N个数据帧
- 接收方的接收窗口为1，按序接收，只对正确接收的回复ACK
- 发送方发现第k个帧丢失(超时)，则对[k, k-1+N]范围内的所有帧重传

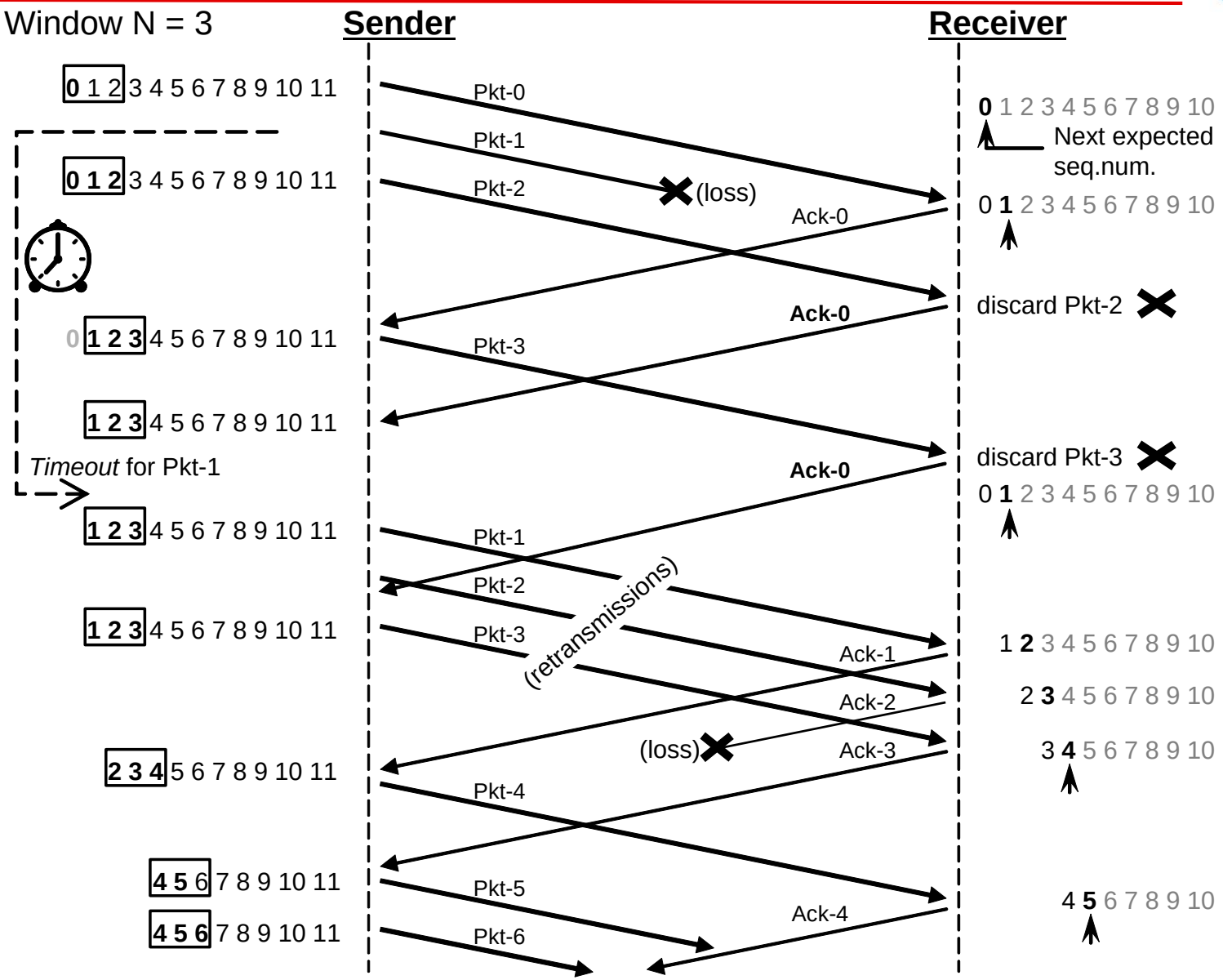
优点

- 接收方不需要缓存接收到的乱序帧，确认简单

缺点

- 按序接收，出错后即便有正确帧到达也丢弃

Go-back-N


Window N = 3
Sender
Receiver
0 1 2 3 4 5 6 7 8 9 10 11
Pkt-0
Pkt-1
Pkt-2
(loss)
Ack-0
0 1 2 3 4 5 6 7 8 9 10
Next expected seq.num.
0 1 2 3 4 5 6 7 8 9 10
0 1 2 3 4 5 6 7 8 9 10 11
discard Pkt-2
Ack-0
Pkt-3
1 2 3 4 5 6 7 8 9 10 11
discard Pkt-3
Ack-0
Timeout for Pkt-1
0 1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10 11
Pkt-1
Pkt-2
Pkt-3
(retransmissions)
1 2 3 4 5 6 7 8 9 10 11
Ack-1
Ack-2
Ack-3
(loss)
1 2 3 4 5 6 7 8 9 10
2 3 4 5 6 7 8 9 10
3 4 5 6 7 8 9 10
2 3 4 5 6 7 8 9 10 11
Pkt-4
Pkt-5
Ack-4
Pkt-6
4 5 6 7 8 9 10
4 5 6 7 8 9 10 11
4 5 6 7 8 9 10 11

选择重传

思路

- 发送方的发送窗口($\leq 2^{w-1}$)，可以一次性发送N个数据帧；
- 接收方的接收窗口($\leq 2^{w-1}$)，对每一个帧进行确认
- 发送方发现第k个帧丢失，仅重传第k个帧；
- 接收方需要缓存乱序抵达的数据帧

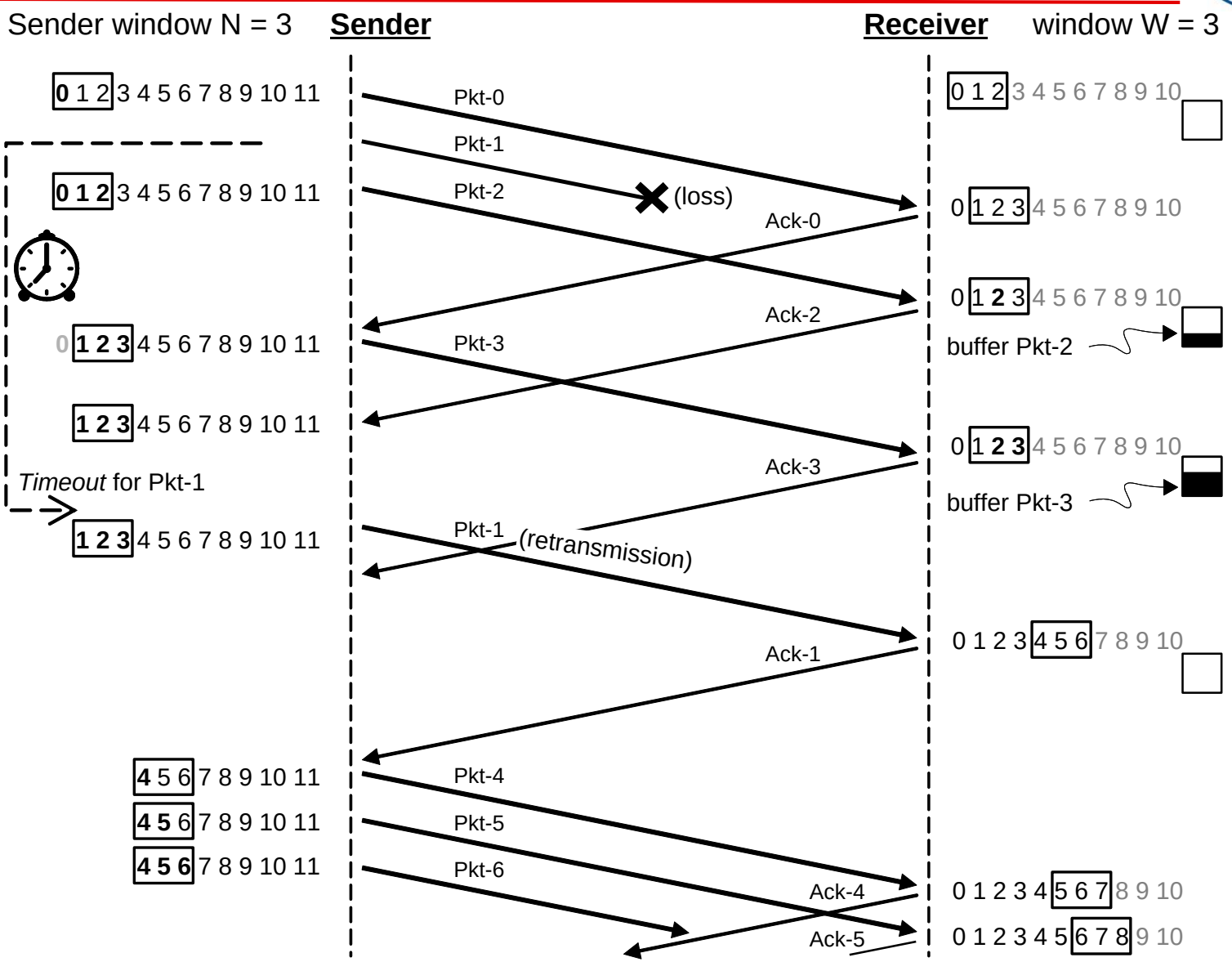
优点

- 避免重传已正确传送的帧，链路利用率较高

缺点

- 接收方更复杂，需要占用一定容量的缓存

选择重传ARQ协议


Sender window N = 3
Sender
Receiver
window W = 3
0 1 2 3 4 5 6 7 8 9 10 11
Pkt-0
Pkt-1
Pkt-2
(loss)
Ack-0
0 1 2 3 4 5 6 7 8 9 10 11
Ack-2
Pkt-3
0 1 2 3 4 5 6 7 8 9 10 11
buffer Pkt-2
1 2 3 4 5 6 7 8 9 10 11
Ack-3
Timeout for Pkt-1
0 1 2 3 4 5 6 7 8 9 10 11
buffer Pkt-3
Pkt-1
(retransmission)
0 1 2 3 4 5 6 7 8 9 10
Ack-1
4 5 6 7 8 9 10 11
Pkt-4
4 5 6 7 8 9 10 11
Pkt-5
4 5 6 7 8 9 10 11
Pkt-6
Ack-4
0 1 2 3 4 5 6 7 8 9 10
Ack-5
0 1 2 3 4 5 6 7 8 9 10

有限范围的帧序号

- 发送方
  - 对数据帧编号
  - 帧中包含 k 比特的帧序号字段
  - 帧序号循环使用
  - 帧序号的有效取值范围: $0 \sim 2^k-1$
- 可以通过帧序号的周期区别相同序号的数据帧
  - 如果接收窗口大小=1

    发送窗口大小≤ 2^k-1
  - 如果接收窗口=发送窗口大小

    发送窗口大小≤ $2^k/2$

滑动窗口协议小结

❑ 滑动窗口协议

不仅:

- 保证帧在物理链路上的可靠传输

而且:

- 保证帧传送的顺序
  - 通过帧序号和滑动窗口，数据链路层协议可以将数据帧按发送顺序地交给高层协议
- 支持流量控制
  - 接收方通过反馈机制可以压制发送方的速率
  - 同步发送方的发送(帧)速率和接收方的接收(帧)速率

小结: 可靠传输

- 可靠传输的核心机制

机制	用途
CRC	差错检测
ACK	成功接收数据帧的确认(携带ACK序号)
定时器	检测发送方超时时间(帧丢失)
帧序号	识别不同的帧(避免帧丢失导致的重复帧)
滑动窗口	控制数据帧的收发 流量控制(保证传输的顺序, 控制传输速率)

考纲要求

- 了解：直连网络的概念
- 了解：网络通信编码(NRZ、NRZI、Manchester、4B/5B)
- 了解：不同的组帧 (面向字节/比特的组帧)方法
- 理解：差错控制(二维奇偶校验、循环冗余校验等)的概念
- 掌握：可靠传输的概念和基本实现机制(停止等待、后退N帧、选择重传等ARQ算法)
- 掌握：相关滑动窗口参数的计算方法

要点回顾

组件	协议			
	BISYNC	DDCMP	PPP	HDLC
可靠传输	停止等待	连续ARQ（滑动窗口）		
差错检测	二维奇偶校验	CRC		
帧定界	面向字节			面向比特
物理链路特点	点到点			

THANKS

谢 谢 聆 听

Email: chenwang@hust.edu.cn

Website: http://www.chenwang.net.cn