计算机网络

Kajih Du

计算机网络

1 绪论
- 1.1 基本组成
- 1.2 协议和服务
- 1.3 OSI参考模型
- 1.4 TCP/IP参考模型
- 1.5 网络设备
- 1.6 度量单位

2 物理层
- 2.1 功能
- 2.2 特性
- 2.3 数据通信基本概念
- 2.4 传输方式
  - 2.4.1 基带传输和频带传输
  - 2.4.2 串行传输和并行传输
  - 2.4.3 传输模式
- 2.5 性能度量
- 2.6 复用技术
  - 2.6.1 时分复用（Time Division Multiplexing）
  - 2.6.2 统计时分复用
  - 2.6.3 频分复用
  - 2.6.4 波分复用
  - 2.6.5 码分复用
  - 2.6.6 正交频分复用
  - 2.6.7 空分复用

3 数据链路层
- 3.1 位置
- 3.2 服务
- 3.3 成帧
  - 3.3.1 字节计数法
  - 3.3.2 带字节填充定界符法
  - 3.3.3 带比特填充定界符法

3.4 差错控制
3.4.1 基本术语
3.4.2 检错码
3.4.2.1 奇偶校验
3.4.2.2 校验和
3.4.2.3 循环冗余校验（CRC）
3.4.3 纠错码
3.4.3.1 Hamming码
3.5 基本协议
3.5.1 乌托邦式单工协议
3.5.2 无错信道的停等式协议
3.5.3 有错信道的单工停等式协议
3.5.4 滑动窗口协议
3.5.5 回退N协议
3.5.6 选择性重传协议

1 绪论

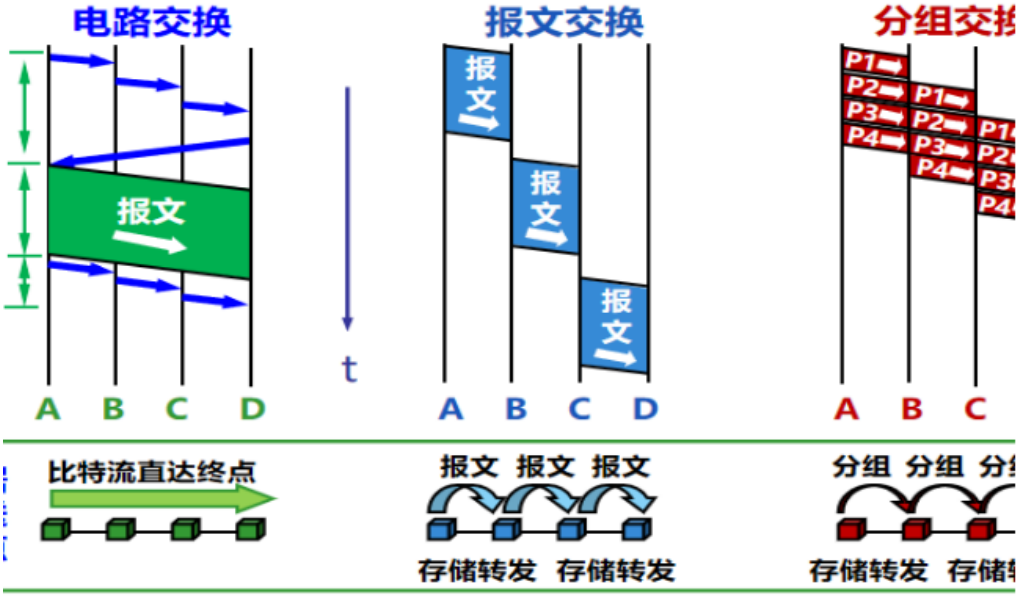
1.1 基本组成

ISP(Internet Service Provider) 网络服务提供商

网络的组成：边缘设备（接收和发送数据）和网络核心（路由和转发）。

传输单元：位（bit），在物理介质上传播的数据最小单元。数据传输使用Bit表示，K/M/G之间为10^3进制。

交换方式：



- 电路交换：需要建立连接并预留资源
- 报文交换：不常用
- 分组交换：适合有大量突发数据传输需求的互联网

1.2 协议和服务

协议：同一层对等实体的进行通讯的规定。包含**语法、语义、时序**三个要素。（“水平”的）
服务：上层实体通过接口调用下层实体的服务。（“垂直”的）分为**面向连接传输服务**（按电话系统模型）和**无连接传输服务**（按邮政系统模型）。

实体可以通过协议来实现其定义的服务

面向链接服务的六个核心服务原语：连接请求、接受响应、请求数据、应答、请求断开、断开连接。

1.3 OSI参考模型

- 物理层：定义在信道上如何传输0和1
- 数据链路层：实现相邻网络实体间的数据传输；成帧：从物理层比特流提取出完整的帧；错误检测和纠正；流量控制；物理地址（MAC Address）；共享信道的访问控制
- 网络层：将数据包跨越网络从源设备发送到目的设备（host to host）；进行路由和转发；IP地址
- 传输层：将数据从源端口发送到目的端口（进程到进程）
- 会话层：在应用程序之间建立和维持会话，并能使会话获得同步
- 表示层：关注所传递信息的语法和语义，管理数据的表示方法，传输的数据结构
- 应用层：通过应用层协议，提供应用程序便捷的网络服务调用

1.4 TCP/IP参考模型

- 应用层
- 传输层
- 互联网层
- 网络接口层

采用IP分组交换：可在各种底层物理网络上运行；可支持各类上层应用；每个IP分组携带各自的目的地址，网络核心功能简单（通过路由表转发分组），适应爆炸性增长

1.5 网络设备

交换机：物理层、链路层 （两层，无法实现路由）
路由器：物理层、链路层、网络层 （三层）

只有物理层的网络设备：集线器、放大器

1.6 度量单位

比特率：数字信道上传送数据的速率，单位b/s或bps

带宽：网络内某通道传输数据能力，单位时间内网络中某信道能通过的“最高数据率”，单位bit/s

包转发率（PPS）：Packet Per Second(包/秒)，表示交换机或路由器等网络设备以包为单位的转发速率

时延：数据（一个报文或分组）从网络（或链路）的一端传送到另一端所需的时间。包含**传输时延**（数据从结点进入到传输媒体介质所需要的时间）、**传播时延**（电磁波在信道中需要传播一定距离而花费的时间）、**处理时延**（主机或路由器在收到分组时，为处理分组所花费的时间，与路由器有关）、**排队时延**（分组在路由器输入输出队列中排队等待处理所经历的时延，和路由算法有关）

往返时延：从发送方发送数据开始，到发送方收到来自接收方的确认，经历的总时间

时延带宽积：传播时延和带宽的乘积，即按比特计数的链路长度

吞吐量：单位时间内通过某个网络(或信道、接口)的数据量，单位是 b/s

有效吞吐量：单位时间内，目的地正确接收到的**有用信息的数目**（以 bit 为单位）

2 物理层

2.1 功能

位置：位于网络体系最底层（不是连接计算机或传输信号的具体的物理设备）

功能：在连接各计算机的传输媒体上**传输数据比特流**

作用：尽可能屏蔽掉不同传输媒体和通信手段的差异

2.2 特性

接口特性：物理层协议是DTE和DCE间的约定，规定了两者之间的接口特性
机械特性：定义接线器的形状和尺寸、引线数目和排列、固定和锁定装置等
电气特性：规定了DTE/DCE之间多条信号线的电气连接及有关电路特性
功能特性：描述接口执行的功能，定义接线器的每一引脚(针，Pin)的作用
过程特性：指明对于不同功能的各种可能事件的出现顺序

2.3 数据通信基本概念

通信：在源点和终点之间传递信息或消息
消息：通信过程中表达客观物质运动和主观思维活动的文字、符号、数据等
信息：消息中对通信者有意义的部分内容
数据：对某一事实的不经解释并赋予一定含义的数字、字母、文字等符号及其组合的原始表达
信号：消息的载体

信息量：一条消息所含信息的多少，与描述事件出现的概率有关，单位为比特

$$I = -\log_a(p)$$

平均信息量：每个符号包含信息量的统计平均，单位为比特/符号

$$H = \sum_i -p_i \log p_i$$

模拟数据：一段时间内具有连续的值
离散数据：一段时间内具有分散的值

2.4 传输方式

2.4.1 基带传输和频带传输

基带传输：不搬移信号频谱的传输机制
频带传输：通过调制调节器搬移信号频谱的传输机制（为了适应信道的频率特性）

2.4.2 串行传输和并行传输

串行传输：数据在一个信道上按位依次传输的方式
并行传输：数据在多个信道上同时传输的方式

2.4.3 传输模式

单工：指两个站之间只能沿一个指定的方向传送数据信号
半双工：指两个站之间可以在两个方向上传送数据信号，但不能同时进行
全双工：指两个站之间可以在两个方向上同时传送数据信号

2.5 性能度量

传输速率：单位时间内传送信息量
调制速率（波特率）：单位时间内调制信号波形变换次数，单位为波特（baud）

$$R_B = 1/T$$

数据信号速率（比特率）：单位时间内通过信道的信息量，单位bit/s

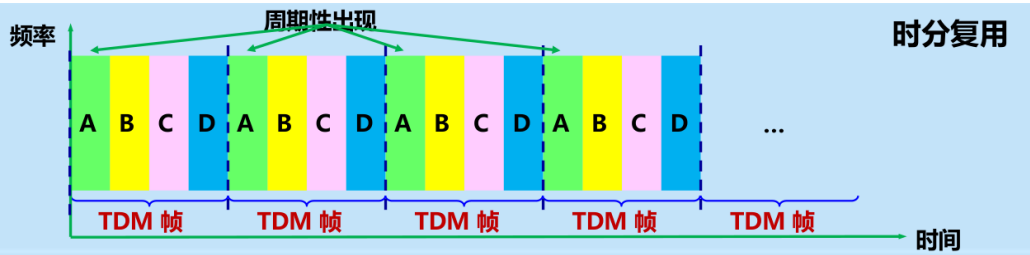
$$R_b = \sum_i \frac{1}{T_i} \log_2 M_i = R_B \log_2 M_i$$

相同条件下，进制越大，比特速率越大，但差错率也大。

2.6 复用技术

复用：允许用户使用一个共享信道进行通信，避免相互干扰，降低成本，提高利用率。

2.6.1 时分复用（Time Division Multiplexing）



- 每一个时分复用的用户在每一个时分复用（TDM）帧占用固定序号的时隙

- 每一个用户占用时隙周期性出现（周期为TDM帧长度）
- TDM信号为等时信号
- 所有用户在不同时间占据同样频带宽度

缺点：对信道利用率低（某用户暂时无数据发送时，在时分复用帧中分配给该用户的时隙只能处于空闲状态），不同用户之前传输时需要进行同步

2.6.2 统计时分复用

统计时分复用：动态地按需分配共用信道的时隙，只将需要传送数据的终端接入共用信道，以提高信道利用率

在传输数据前进行统计，按需**动态分配时隙**

2.6.3 频分复用

频分复用：将多路基带信号调制到不同频率载波上，再进行叠加形成一个复合信号的多路复用技术

频分复用的所有用户在同样的时间**占用不同的频率带宽资源**

2.6.4 波分复用

波分复用：利用多个激光器在单条光纤上同时发送多束不同波长激光的技术

2.6.5 码分复用

码分复用：利用码序列相关性实现的多址通信 , 基本思想是靠不同的地址码来区分的地址

每个站的码片序列各不相同，并且需要正交

缺点：容易受到噪声影响；传输过程中传输的数据量增加，需要的频率带宽增加

2.6.6 正交频分复用

将信道分为若干正交子通道，将高速数据信号转换成并行的低速子数据流，调制到在每个子信道上进行传输

相比于频分复用，其允许不同使用者的频带可以重叠，提高了频带的利用率

2.6.7 空分复用

空分复用：让同一个频段在不同的空间内得到重复利用

3 数据链路层

3.1 位置

向下：利用物理层提供的位流服务
向上：向网络层提供明确接口

实现功能：成帧、差错控制、流量控制

3.2 服务

根据确认（ACK）有无和连接（connection）有无分类

- 无确认无连接服务

  接收方不会确认是否收到，不会记录网络传输状态

  优点：传输速度快；缺点：不能确认数据是否有错误

  适用：误码率低的可靠信道；实时通信
- 有确认无连接服务

接收方会确认是否收到，不会记录网络传输状态

优点：网络通信可靠

适用：不可靠的信道（无线信道）

- 有确认有连接服务

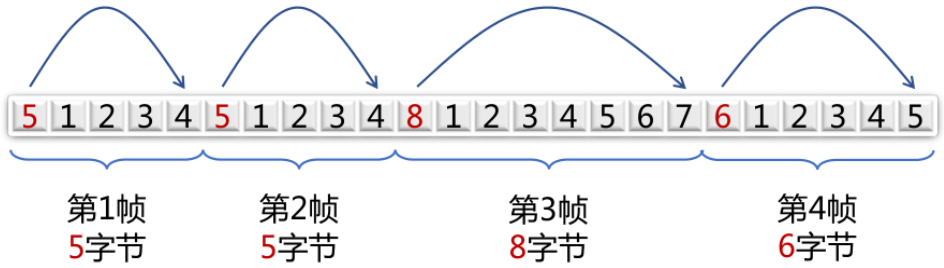
  接收方会确认是否收到，并记录网络传输状态

  适用：长延迟的不可靠的信道

3.3 成帧

目的：为了差错控制和流量控制

3.3.1 字节计数法



每个帧的头标表示帧包含的字节数。

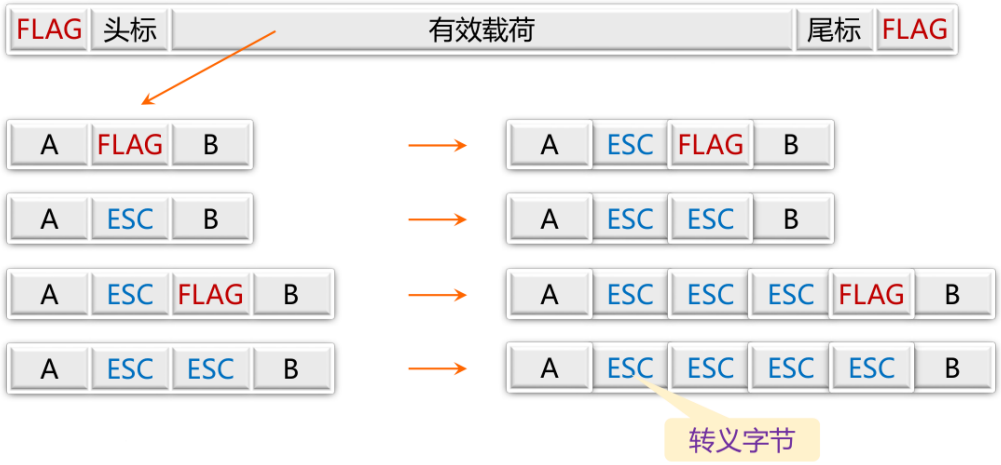
缺点：只能用于无差错传输的情形，方法抗噪性低。

3.3.2 带字节填充定界符法

用定界符区分前后两个帧

缺点：有效载荷部分不能有与定界符相同的字节

解决方法：定义转义字符，接收到转义字符，后一字节无条件成为有效载荷

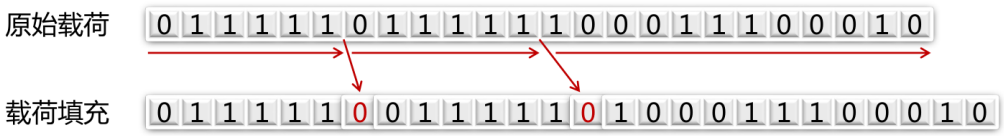


不足：转义字符和flag以字节为单位
改进方式：使用带比特填充定界符法

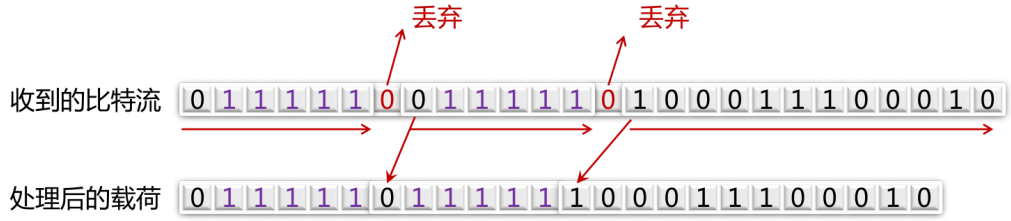
3.3.3 带比特填充定界符法

采用一特定的比特组合01111110（0x7E）来标志帧的边界，也就是一个标志字节。

发送方检查有效载荷：若在有效载荷中出现连续5个1比特，则直接插入1个0比特



接收方处理：若出现连续5个1比特，若下一比特为0，则为有效载荷，直接丢弃0比特；若下一比特为1，则连同后一比特的0，构成定界符，一帧结束



带字节填充定界符法和带比特填充定界符法属于透明传输

3.4 差错控制

主要策略：检错码、纠错码

检错码：在被发送的数据块中，包含一些冗余信息，但这些信息**只能使接收方推断是否发生错误**，但不能推断哪位发生错误，接收方可以请求发送方重传数据

检错码用在高可靠、误码率较低的信道上，差错可以用重传的方法解决

纠错码：发送方在每个数据块中加入足够的冗余信息，使得接收方能够判断接收到的数据是否有错，并能纠正错误

纠错码用在错误发生比较频繁的信道上

3.4.1 基本术语

码字：一个包含m个数据位和r个校验位的n位单元，(n, m)码，$n = m + r$
码率：码字中不含冗余部分所占的比例，可以用m/n表示
Hamming距离：两个码字之间不同对应比特的数目

检查出d个错，可用Hamming距离$d + 1$的编码
纠正d个错，可用Hamming距离$2d + 1$的编码（找Hamming距离最小的有效码字）

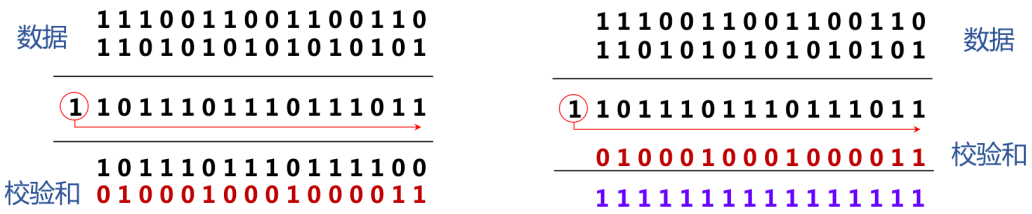
3.4.2 检错码

3.4.2.1 奇偶校验

奇偶校验：增加1位校验位，可以**检查奇数位错误**。

- 偶校验：保证1的个数为偶数个
- 奇校验：保证1的个数为奇数个

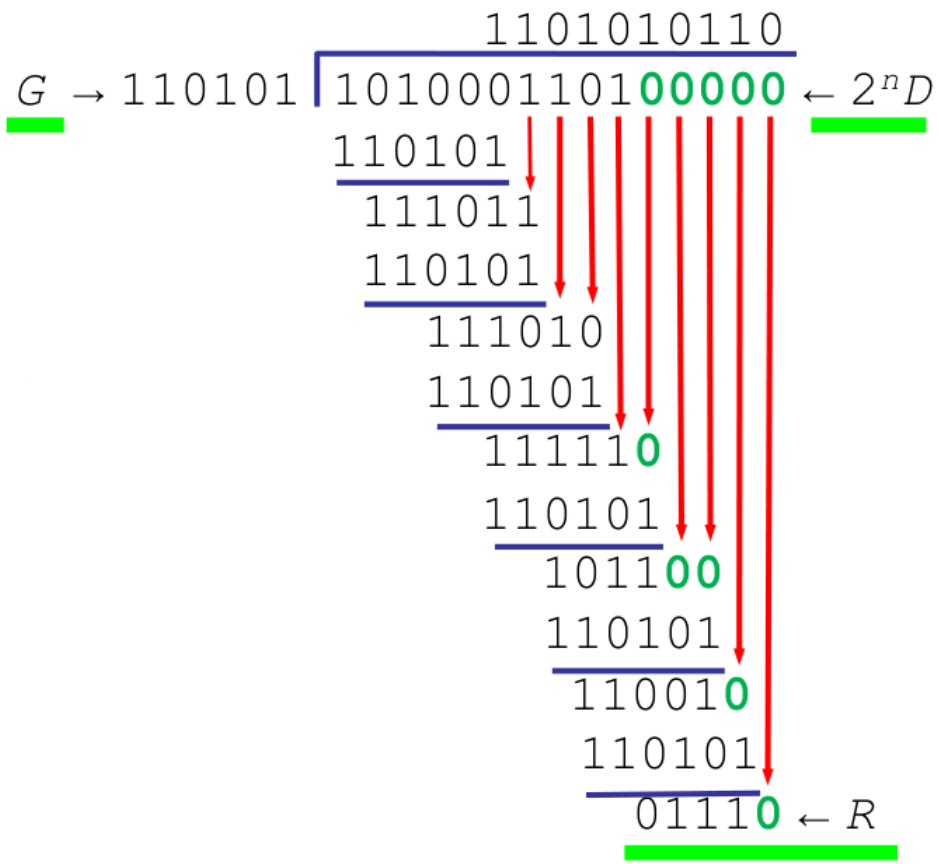
3.4.2.2 校验和



发送方：进行 16 位二进制补码求和运算，计算结果取反，随数据一同发送
接收方：进行 16 位二进制补码求和运算（包含校验和），结果非全1，则检测到错误

3.4.2.3 循环冗余校验（CRC）

- 设原始数据D为k位二进制位模式
- 如果要产生n位CRC校验码，事先选定一个$n+1$位二进制位模式G (称为生成多项式，收发双方提前商定)，G的最高位为1
- 将原始数据D乘以2^n（相当于在D后面添加 n 个0），产生$k+n$位二进制位模式，用G对该位模式做模2除，得到余数R（n位，不足n位前面用0补齐）即为CRC校验码



3.4.3 纠错码

单比特纠错：
m个信息位，r个校验位，对2^m个有效信息中任何一个，有n个与其距离为1的无效码字，于是$(n+1)2^m \leq 2^n$，进而

$$m + r + 1 \leq 2^r$$

3.4.3.1 Hamming码

校验位：2的幂次方位

...

3.5 基本协议

- 分层进程独立假设：网络层、数据链路层、物理层为独立进程，进程间通过传递信息通信
- 提供可靠服务假设：提供可靠、面向连接的服务
- 只处理通信错误假设：仅处理通信错误

声明：

- `void to_physical_layer(frame *s);`
- `void from_network_layer(packet *p);`
- `void from_physical_layer(frame *r);`
- `void to_network_layer(packet *p);`
- 事件：帧到达`frame_arrival`、帧出错`cksum_err`、超时`timeout`
- 帧

```
typedef struct {
    frame_kind kind; // 数据帧或确认帧
    seq_nr seq; // 序列号
    seq_nr ack; // 确认号
    packet info;
} frame;
```

3.5.1 乌托邦式单工协议

假设：

- 完美信道：传输过程中无任何差错
- 始终就绪：发送方/接收方的网络层始终处于就绪状态
- 瞬间完成：发送方/接收方能生成/处理无限多数据

特点：不需要进行流量控制和纠错

实现：

- 发送方：循环中不断发送，从网络层获得数据封装成帧，交给物理层完成一次发送

```
frame s;
packet buffer;
while (true) {
    from_network_layer(&buffer); // 从网络层获得数据
    s.info = buffer; // 封装成帧
    to_physical_layer(&s); // 传送到物理层
}
```

- 接收方：循环中持续接受，等待帧到达`frame_arrival`，从物理层获得帧，解封装将帧的数据传到网络层，完成一次接收

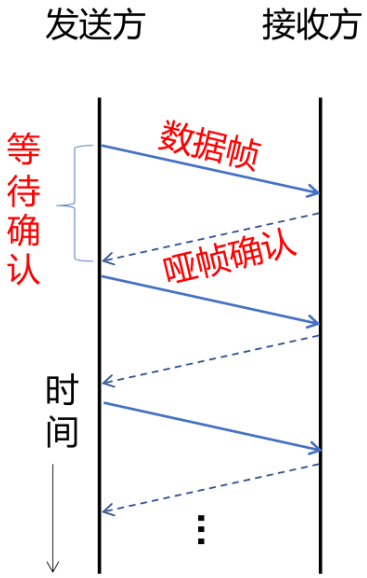
```
frame r;
event_type event;
while (true) {
    wait_for_event(&event);
    from_physical_layer(&r); // 从物理层获得帧
    to_network_layer(&r.info); // 解封装发送到网络层
}
```

3.5.2 无错信道的停等式协议

假设：

- 完美信道：传输过程中无任何差错
- 始终就绪：发送方/接收方的网络层始终处于就绪状态

特点：需要考虑流量控制



实现：

- 发送方：发送一帧后，等待确认到达，确认到达后发送下一帧

```
frame s;
packet buffer;
event_type event;
while (true) {
    from_network_layer(&buffer); // 从网络层获得数据
    s.info = buffer; // 封装成帧
    to_physical_layer(&s); // 传送到物理层
    wait_for_event(&event); // 等待确认到达
}
```

- 接收方：完成一帧接收后交给物理层一个哑帧（帧的内容无意义），作为确认帧

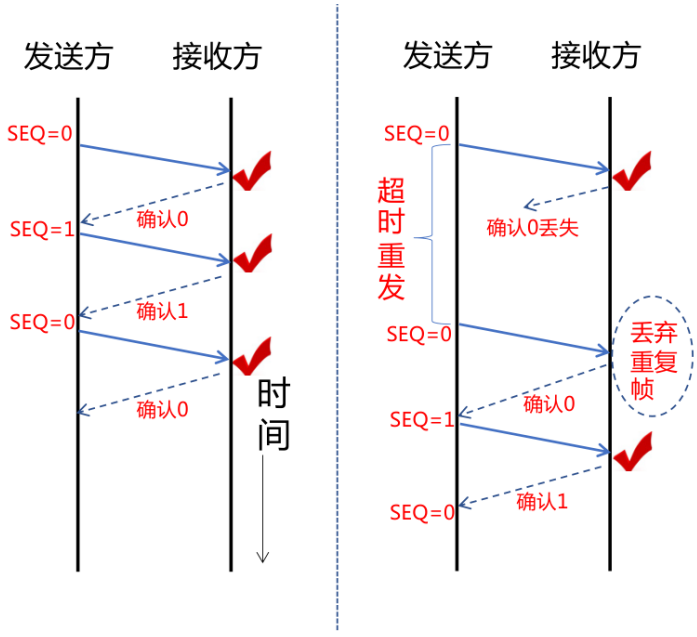
```
frame r;
frame s; // 哑帧
event_type event;
while (true) {
    wait_for_event(&event);
    from_physical_layer(&r); // 从物理层获得帧
    to_network_layer(&r.info); // 解封装发送到网络层
    to_physical_layer(&s);
}
```

3.5.3 有错信道的单工停等式协议

传输过程中出现的问题：差错、丢失、重复帧

解决方法：

- 否定确认帧（NACK）：解决差错
- 差错和丢失同时解决：设置超时（timeout）
- 重复帧出现的原因：1. 超时设置太短；2.确认帧丢失
  解决重复帧：在帧中添加序列号（`SEQ`），1位序号（0或1）即可



实现：

- 发送方：初始化帧序号0，发送帧；等待事件；如果正确确认，发送下一帧，否则重发

```
frame s;
packet buffer;
event_type event;
next_frame_to_send = 0;
while (true) {
    s.info = buffer; // 封装成帧
    to_physical_layer(&s); // 传送到物理层
    start_timer(s.seq);
    wait_for_event(&event); // 等待事件
    if (event == frame_arrival) {
        from_physical_layer(&s); // 从物理层获得帧
        if (s.ack == next_frame_to_send) { // 确认号确认
            from_network_layer(&buffer);// 从网络层获得数据
            inc(next_frame_to_send); // 修改序列号
        }
        // 如果确认号不一致，会直接回到循环体头部重传帧
    }
    // 如果不是帧到达（超时、帧出错），则回到循环体头部重传帧
}
```

- 接收方：初始化期待0号帧；等待帧到达；如果为正确帧，则交给网络层并发送帧确认，否则发送上一个成功接收帧的确认

```
frame r;
frame s; // 确认帧
event_type event;
frame_expected = 0;
while (true) {
    wait_for_event(&event);
    if (event == frame_arrival) { // 帧到达事件
        from_physical_layer(&r);
        if (r.seq == frame_expected) { // 判断是否为重复帧
            to_network_layer(&r.info);
```

```
            inc(frame_expected); // 修改期待的序列号
        }
        // 如果是重复帧，发送前一个帧的序列号，即1 - frame_expected
        // 如果不是重复帧，frame_expected被修改，原来帧的序列号为1 - frame_expected
        s.ack = 1 - frame_expected;
        to_physical_layer(&s);
    }
    // 帧出错等待下一个事件
}
```

效率评估：

- 帧的大小F，bits
- 信道带宽R
- 传播时延和处理时延I，总延迟$D = 2I$
- 信道利用率：

  用于发送的有效时间为F/R，消息发送使用的总时间为$F/R + D$，利用率为

  $$\frac{\frac{F}{R}}{\frac{F}{R} + D} = \frac{F}{F + RD}$$

  RD为时延带宽积

3.5.4 滑动窗口协议

长肥网络：时延带宽积很大（大于10^5 bits）的网络，如卫星通讯网络。

对于长肥网络，信道利用率很低。

解决方法：使用更大的帧。但帧越大，在传输中出错的概率越高，将导致更多的重传。参考分组交换的方法，允许发送方在没收到确认前发送多个帧

3.5.5 回退N协议

采用出错全部重发的方式：当接收端收到一个出错帧或乱序帧时，丢弃所有的后继帧，并且不为这些帧发送确认；发送端超时后，重传所有未被确认的帧

适用场景：该策略对应接收窗口为1的情况，即只能按顺序接收帧

优点：连续发送提高了信道利用率
缺点：按序接收，出错后即便有正确帧到达也丢弃重传

3.5.6 选择性重传协议