计算机通信与网络

Computer Communications & Networks

第6章 传输层

南京邮电大学通信与信息工程学院

“计算机通信与网络” 国家精品课程组

大纲要求

Requirements

01 掌握传输层协议的原理和功能，理解和掌握传输层端口的意义和作用

02 了解**UDP**的工作机制及特点

03 掌握**TCP**的三次握手机制、流量控制和拥塞控制等传输控制过程，理解**TCP**报文段格式，确认往返时间测量等实现方法

内容纲要

Contents Page

传输层作用

传输层必要性

6.1.1 传输层作用

传输层又称为**运输层**，位于应用层和网络层之间，是分层网络体系结构的核心部分。

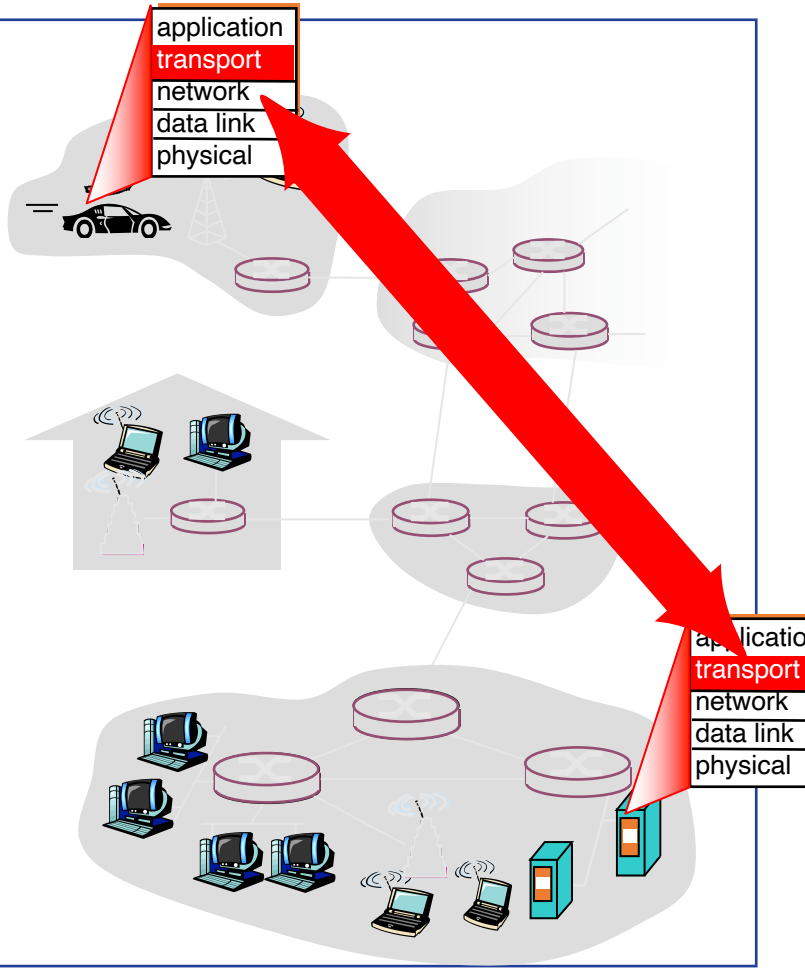
传输层的作用是在**通信子网**提供的服务的基础上，为应用层提供有效的、合理的传输服务。使高层用户在相互通信时**不必**关心通信子网实现细节和具体服务质量。

6.1.1 传输层作用

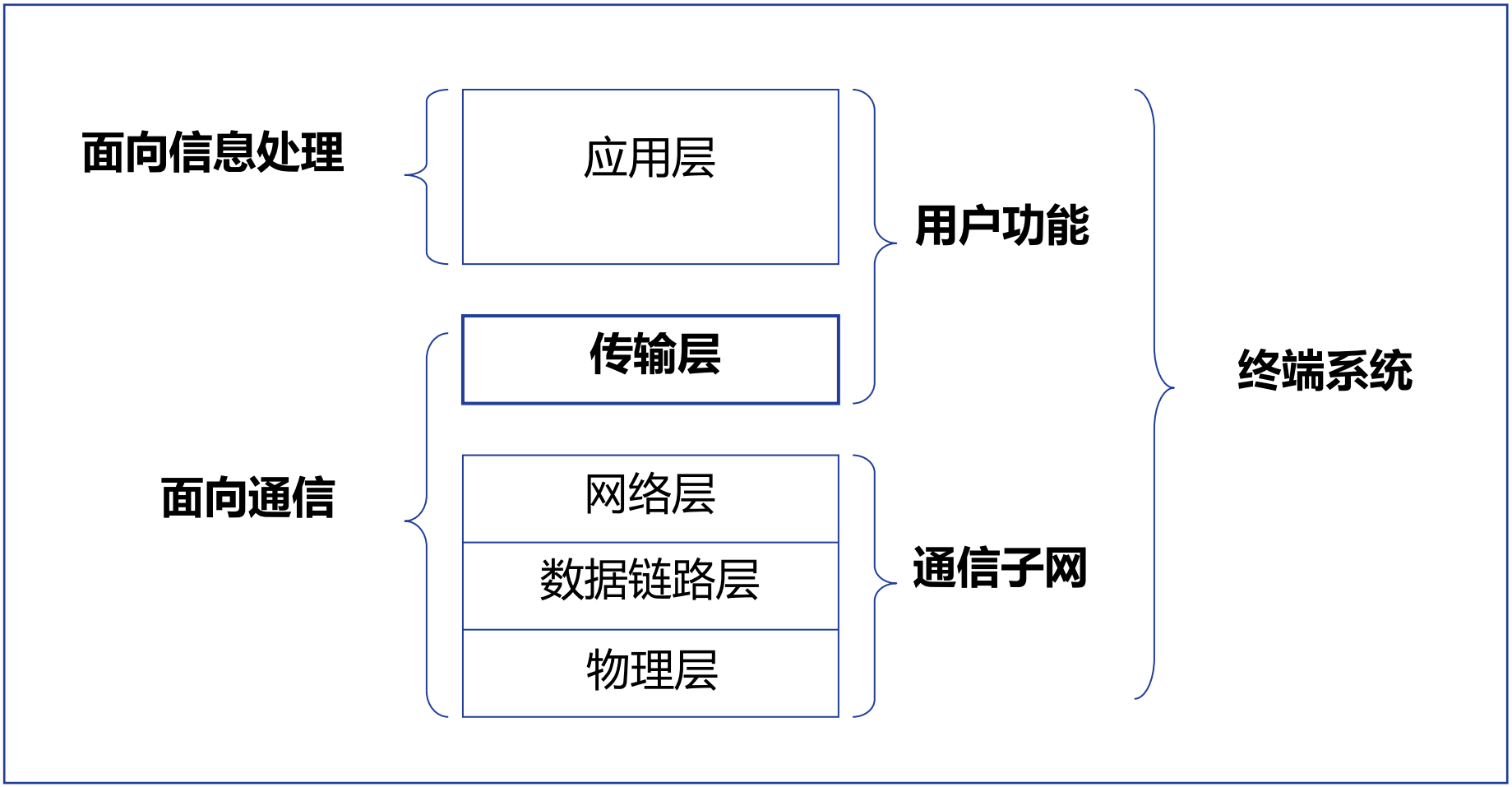
传输层提供应用程序进程之间的通信抽象。

传输协议在端点上运行：

- 发送端：传输协议将应用程序消息分割成段，并传递给网络层。
- 接收端：将分割的段重新组装成消息，并传递给应用层。



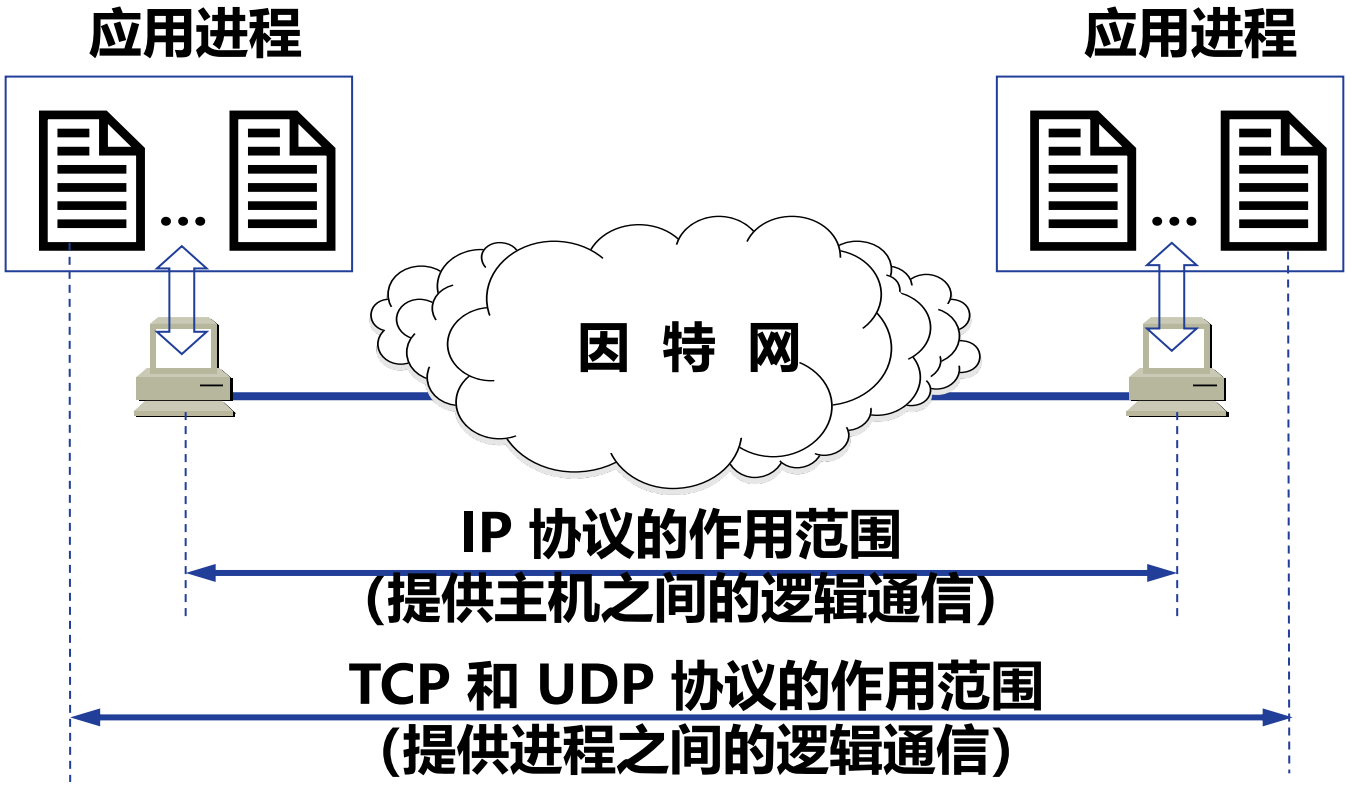
6.1.1 传输层作用



应用进程通信

6.1.2 传输层必要性

（1）两台主机进行通信实际上是两台主机中的**应用进程**互相通信



传输层 vs. 网络层

- **网络层**:用于在主机之间进行通信的抽象。网络层提供了最大努力的数据包传递到远程端点。
- **传输层**:进程之间的通信抽象。将数据包传递给进程。

家庭类比:

三个孩子给另外三个孩子写信。

- 节点 = 房子
- 进程 = 小孩
- 应用程序消息 = 信封中信
- 传输层协议 = Alice 和 Bob 分别连接到室内的兄弟姐妹进行复用（mux）和解复用（demux)
- 网络层协议=邮政服务

Alice

Bob

传输层 vs. 网络层

- 应用程序连接由四元组标识：

- 源IP地址
- 源端口
- 目的地IP地址
- 目的地端口

- 在这个类比中，

- **源地址**:第一座房子的地址
- **源端口**:第一座房子的一个孩子的名字
- **目的地地址**:第二座房子的地址
- **目的地端口**:第二座房子的二个孩子的名字

差错校验

6.1.2 传输层必要性

（2）传输层对**整个报文段**进行差错校验和检测

为了提高传输效率，**IP首部中的首部校验和字段**只检验IP数据报首部是否出现差错而**不检查数据部分**

传输层**TCP和UDP的校验和**既要**校验首部也要校验数据部分**，并且只在发送端进行一次校验和计算，在接收端进行一次检测

传输层协议

6.1.2 传输层必要性

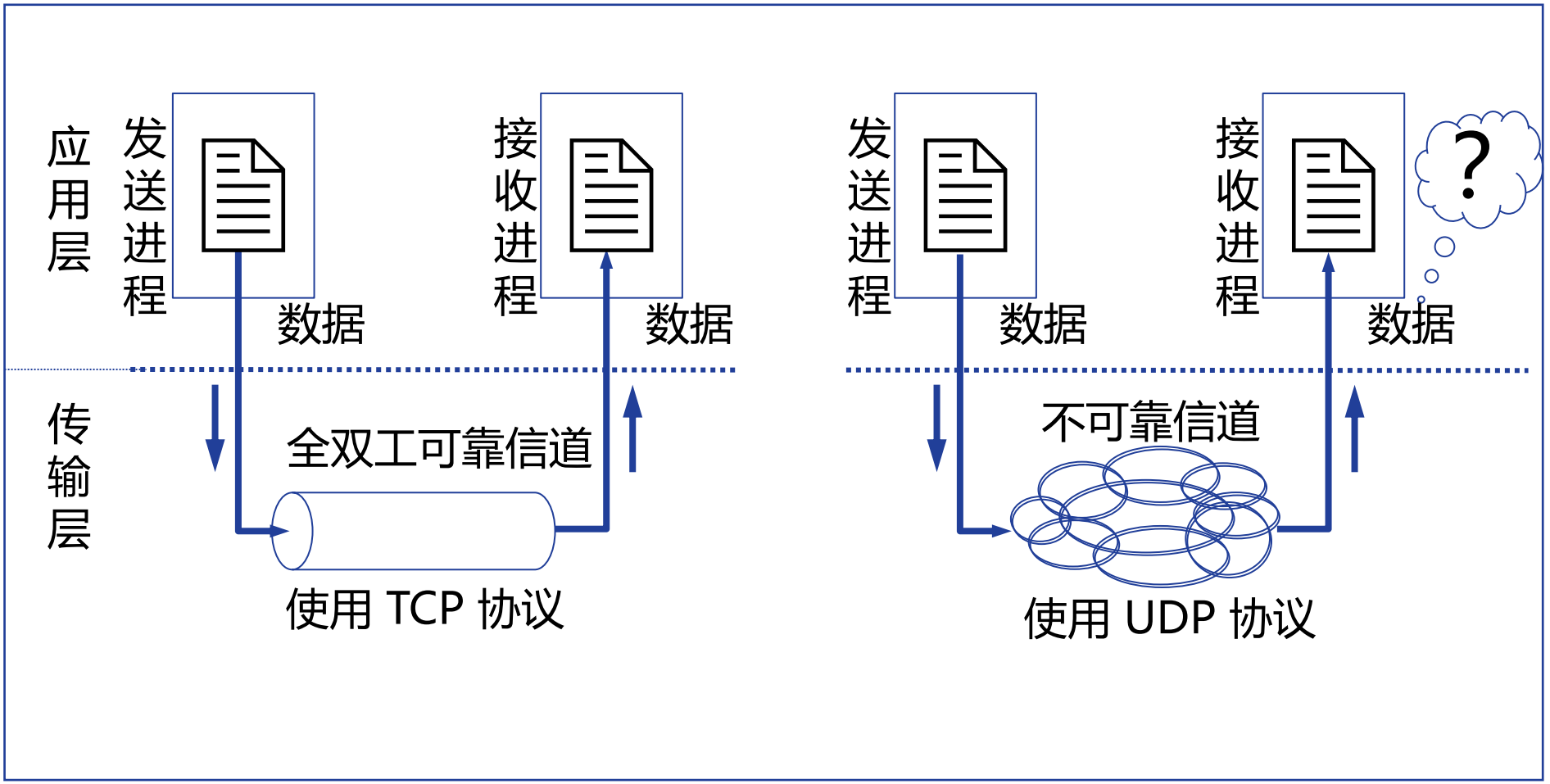
（3）根据应用的不同，传输层需要执行不同的传输协议来提供合理的传输服务

当传输层采用面向连接的协议（如TCP）时，它为应用进程在传输实体间建立一条全双工的**可靠逻辑信道**，尽管下面的网络可能是不可靠的（如IP网络）。

当传输层采用如UDP无连接协议时，这种**逻辑信道是不可靠的。**

传输层协议

6.1.2 传输层必要性



合理传输服务

6.1.2 传输层必要性

（4）传输层的存在使得传输服务比网络服务更加合理有效

用户不能对通信子网加以控制，**无法解决**网络层的服务质量不佳问题。

应用层协议如果强调数据传输的可靠性，那么选择TCP较好。如果应用层协议强调实时应用要求，那么选择UDP为宜。

服务原语

6.1.2 传输层必要性

（5）传输层采用一个标准的**原语集**提供传输服务

由于传输服务独立于网络服务，故可以采用一个标准的原语集提供传输服务。

为网络向高层提供了一个统一的服务界面，所以用传输服务原语编写的应用程序就可以广泛适用于各种网络。

UDP协议特点

UDP格式

传输层校验和计算

UDP应用

基本概念

6.3.1 UDP协议特点

UDP是一个简单的面向用户数据报的传输层协议。

应用进程的输出**正好**产生一个UDP数据报，并组装成一个待发送的IP数据报。

UDP 只在 IP 的数据报服务之上增加了很少一点的功能，即**端口**的功能和有限的**差错检测**功能。

基本概念

6.3.1 UDP协议特点

UDP是**无连接**的。

UDP提供**不可靠**的服务。

UDP同时支持**点到点**和**多点之间**的通信。

UDP是**面向报文**的。

Wireshark采集UDP数据报

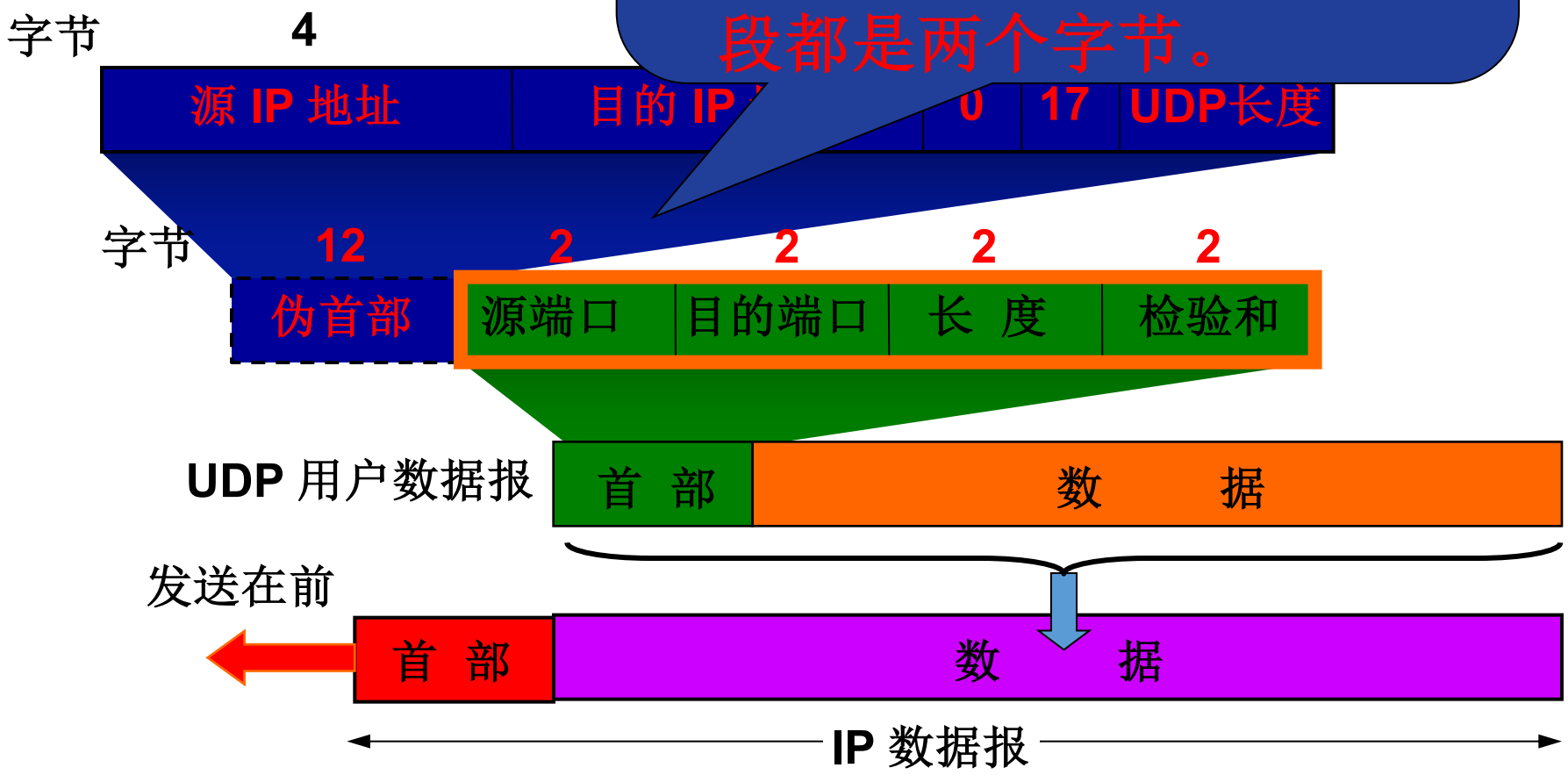
6.3.2 UDP格式

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.10.214.179	222.202.96.108	UDP	150	Source port: 56327 Destination port: ir
2	0.497020	10.10.214.179	222.202.96.108	UDP	150	Source port: 56327 Destination port: ir
16	4.022032	10.10.214.179	202.119.230.8	DNS	75	Standard query 0x7674 A www.xuruibin.cn
71	4.255989	202.119.230.8	10.10.214.179	DNS	232	Standard query response 0x7674 A 122.70
507	10.927384	10.10.214.79	239.255.255.250	SSDP	175	M-SEARCH * HTTP/1.1
508	10.927386	10.10.214.79	239.255.255.250	SSDP	174	M-SEARCH * HTTP/1.1
509	10.927388	10.10.214.79	239.255.255.250	SSDP	179	M-SEARCH * HTTP/1.1
510	10.927389	10.10.214.79	239.255.255.250	SSDP	143	M-SEARCH * HTTP/1.1
511	10.927391	10.10.214.79	239.255.255.250	SSDP	143	M-SEARCH * HTTP/1.1

```
⊞ Frame 1: 150 bytes on wire (1200 bits), 150 bytes captured (1200 bits)
⊞ Ethernet II, Src: Dell_2f:ec:19 (d0:67:e5:2f:ec:19), Dst: HuaweiTe_fb:ce:c2 (00:25:9e:fb:ce:c2)
⊞ Internet Protocol Version 4, Src: 10.10.214.179 (10.10.214.179), Dst: 222.202.96.108 (222.202.96.108)
⊟ User Datagram Protocol, Src Port: 56327 (56327), Dst Port: irdmi (8000)
    Source port: 56327 (56327)
    Destination port: irdmi (8000)
    Length: 116
  ⊞ Checksum: 0x207a [validation disabled]
⊞ Data (108 bytes)
```

2、UDP首部格式

用户数据报 UDP 有两个字段：数据字段和首部字段。首部字段有 8 个字节，由 4 个字段组成，每个字段都是两个字节。

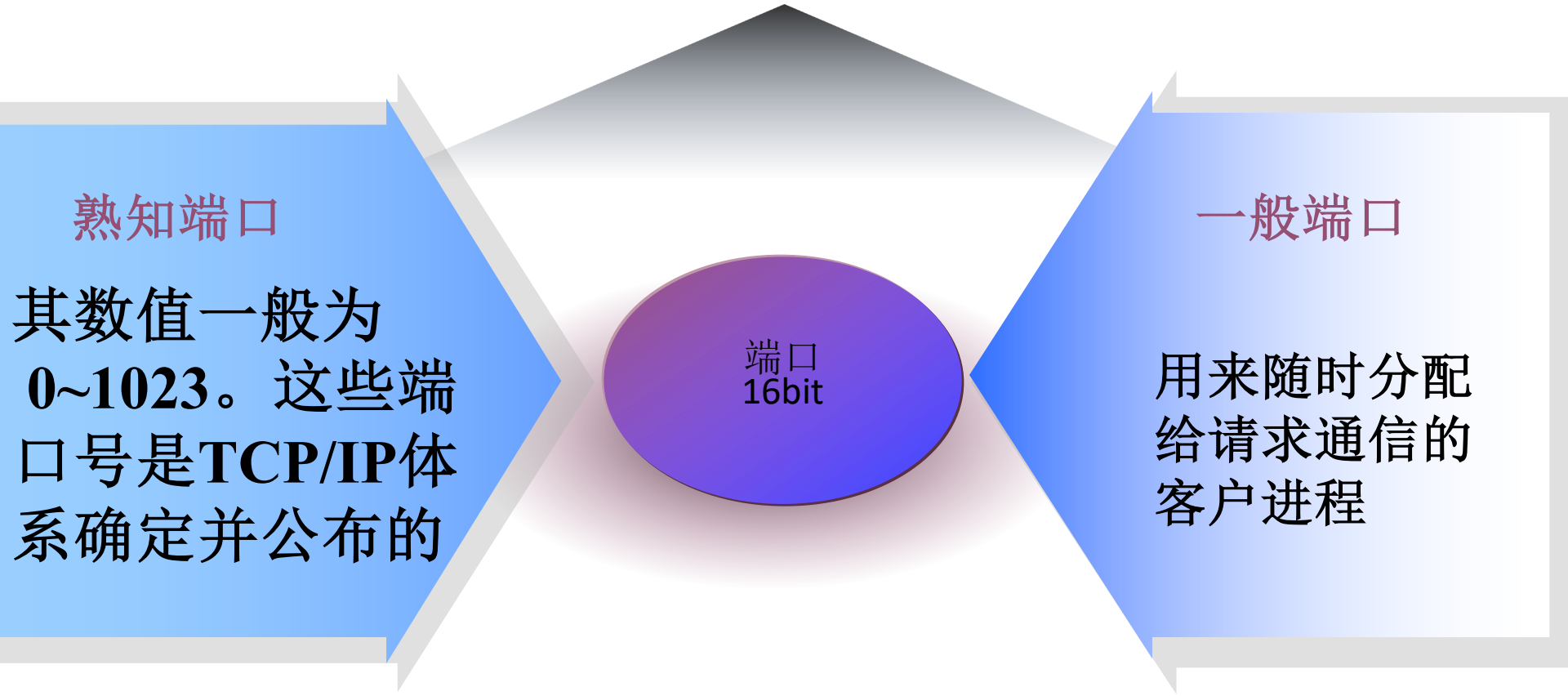


伪首部概念

6.3.3 传输层校验和计算

UDP 用户数据报的首部中检验和用来检验**整个用户数据报**（首部加数据部分）出现的差错。

在计算检验和时在 UDP 数据报之前要增加 12个字节的伪首部。所谓“**伪首部**”是因为这种首部只在计算UDP校验和的时候使用，既不向下层传送，也不向上层递交。


熟知端口
其数值一般为0~1023。这些端口号是TCP/IP体系确定并公布的
端口
16bit
一般端口
用来随时分配给请求通信的客户进程

6.3.4 UDP应用

协议名称	协议	默认端口	使用UDP协议原因说明
域名系统	DNS	53	为了减少协议的开销
动态主机配置协议	DHCP	67	需要进行报文广播
简单文件传输协议	TFTP	69	实现简单，文件需同时向许多机器下载
简单网络管理协议	SNMP	161	网络上传输SNMP报文的开销小
路由选择信息协议	RIP	520	实现简单，路由协议开销小
实时传输协议 实时传输控制协议	RTP RTCP	5004 5005	因特网的实时应用

TCP协议特点

TCP首部格式

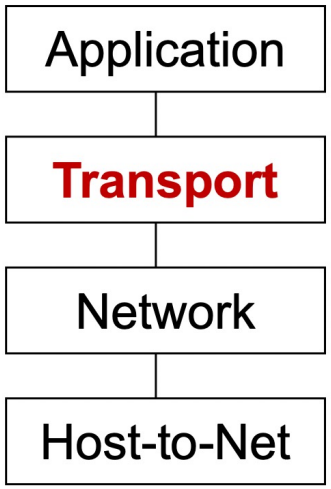
TCP连接管理

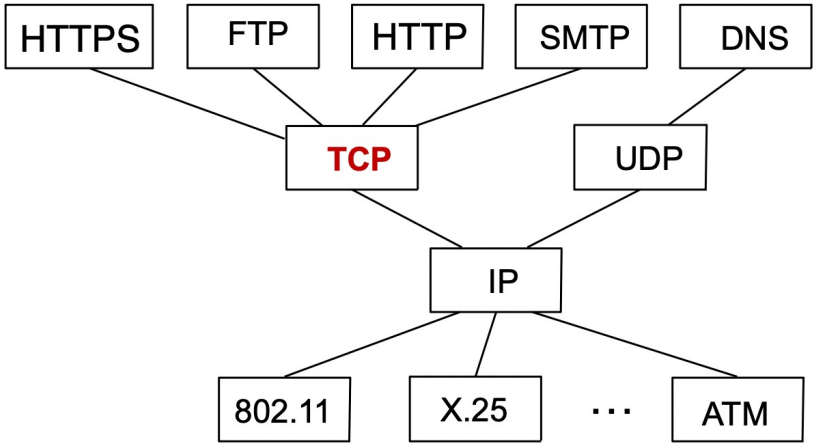
TCP可靠传输

TCP流量控制

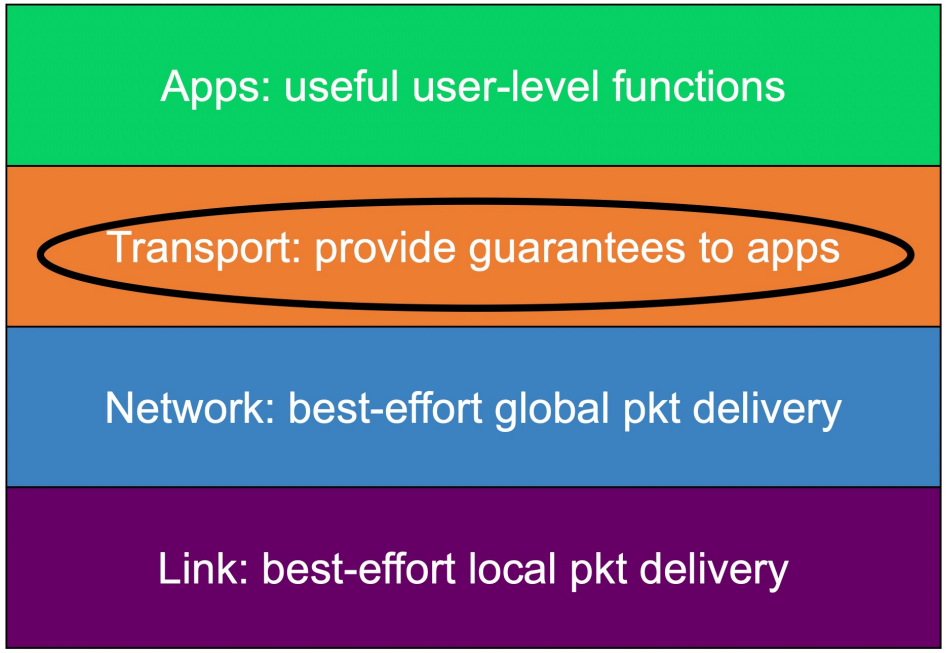
TCP拥塞控制

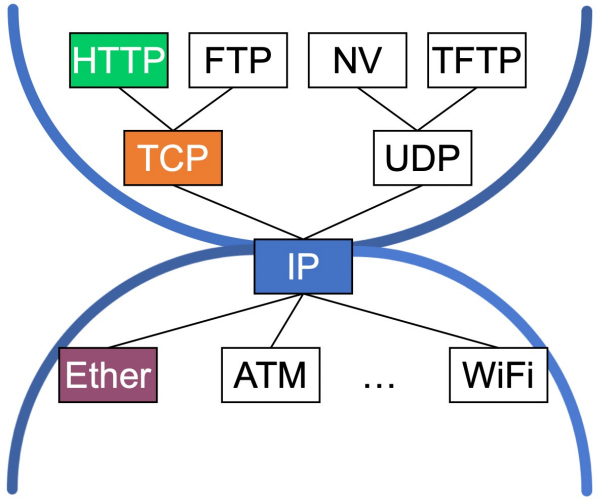
Transport


Application
Transport
Network
Host-to-Net


HTTPS
FTP
HTTP
SMTP
DNS
TCP
UDP
IP
802.11
X.25
···
ATM

Modularity through layering





TCP在best-effort的网络上提供可靠传输

基本概念

6.4.1 TCP协议特点

TCP是Internet的TCP/IP协议家族中的最重要协议之一，满足互联网**可靠传输**的要求。

可靠传输包括：

1. 可靠性（Reliability）
2. 按顺序传输（Ordered delivery）
3. 在核心网资源共享（Resource sharing in the network core）

基本概念

6.4.1 TCP协议特点

TCP是**面向连接**的

TCP提供**可靠**的服务

TCP只能进行**点到点**的通信

TCP是**面向字节流**的

基本概念

6.4.4 TCP可靠传输

TCP是可靠的传输层协议，主要通过以下三个方面来保证可靠传输：

（1）序号确认机制

（2）超时重传机制

（3）定时器设置

序号确认机制

6.4.4 TCP可靠传输

TCP 将所要传送的整个应用层报文看成是一个个字节组成的数据流，然后对每一个字节编一个序号。

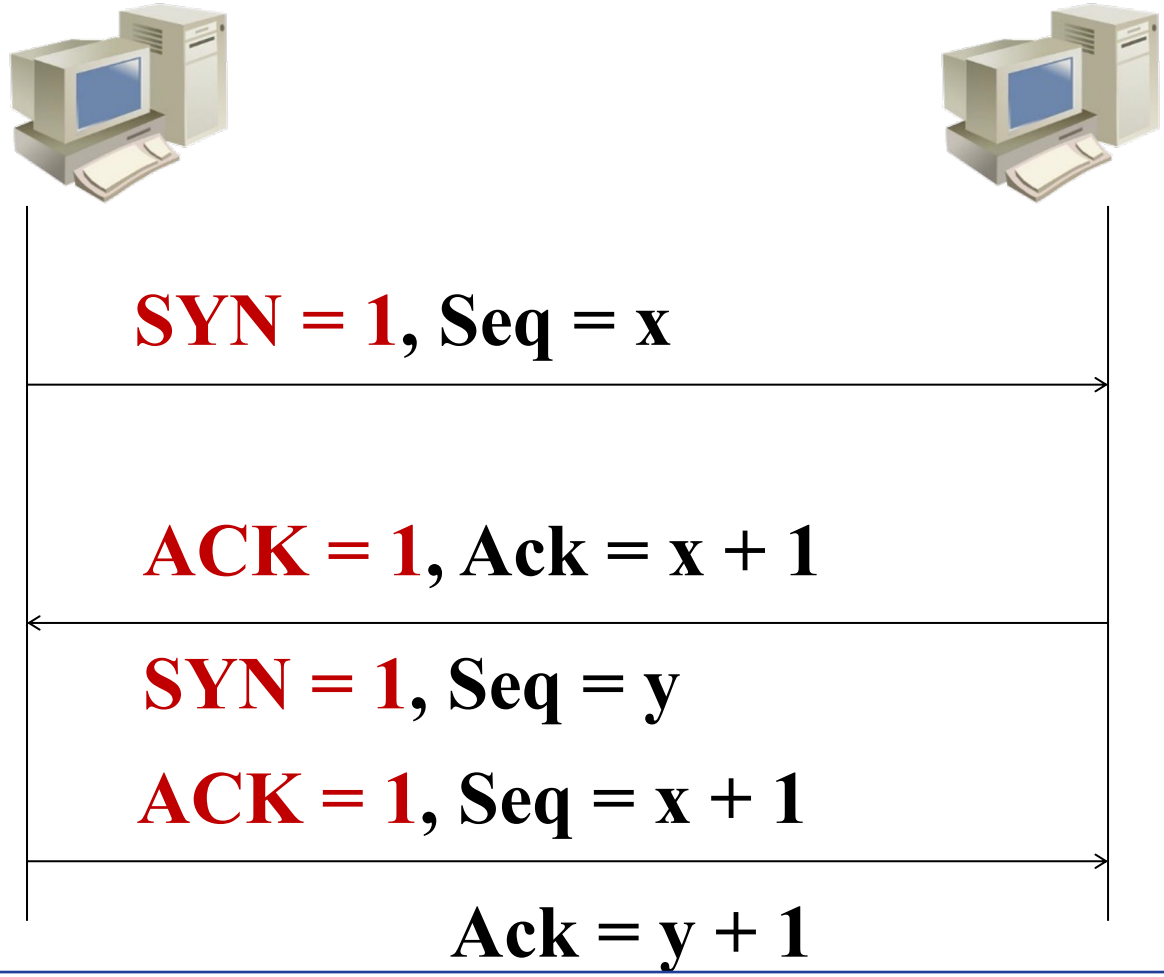
在连接建立时，双方要商定**初始序号**。

TCP 就将每一次所传送的报文段中的第一个数据字节的序号，放在TCP首部的**序号**字段中。

三次握手

6.4.3 TCP连接管理

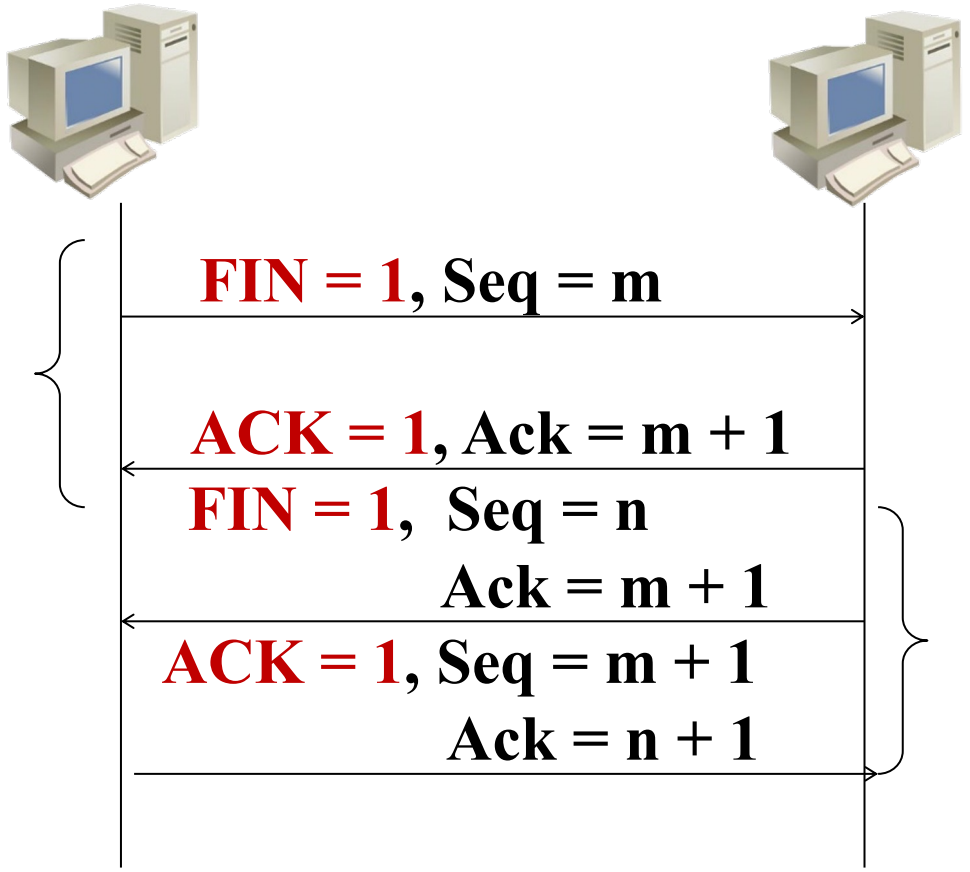
玫红色代表是标志位的置位情况




客户端
服务器
CLOSED
CLOSED

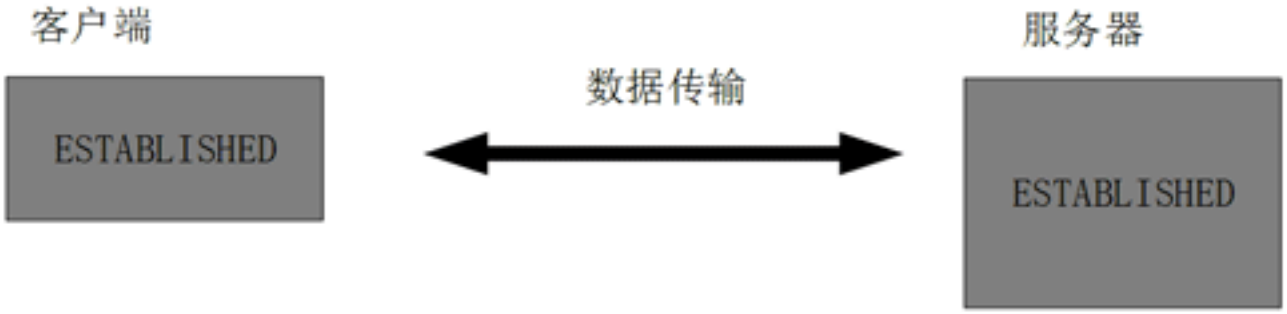
握手释放(四次挥手)

6.4.3 TCP连接管理



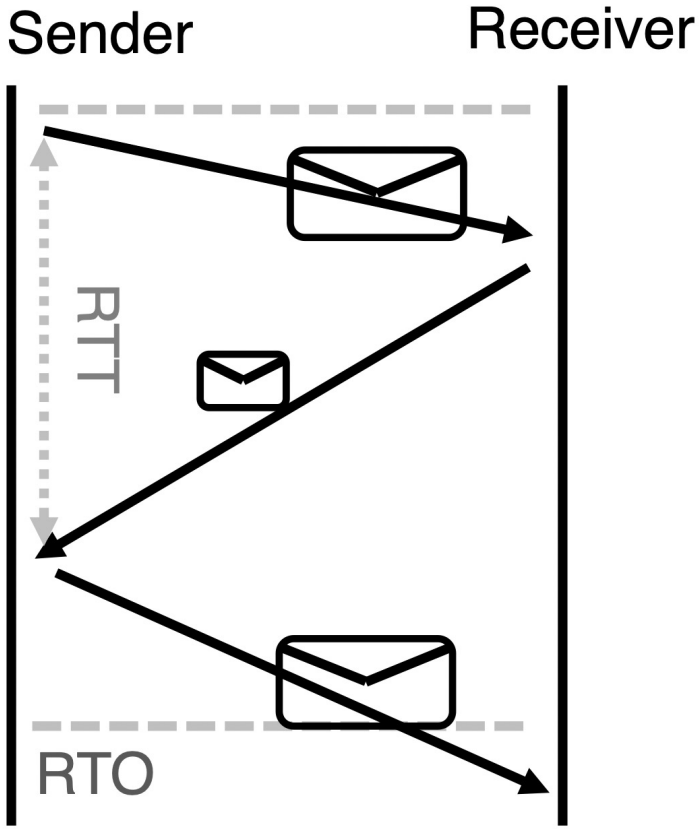
四次挥手

6.4.3 TCP连接管理



stop-and-wait

1. 假设发端需要等待ACK或者RTO才能发送下一个数据包
2. 假设没有丢包，其中：
   - RTT=RTO=100 ms
   - 数据包大小：12 Kbit
   - 链路速率：12 Mbit/s
3. 则数据传输率：
   - 12Kbits / 100 ms = 120 Kbit/s

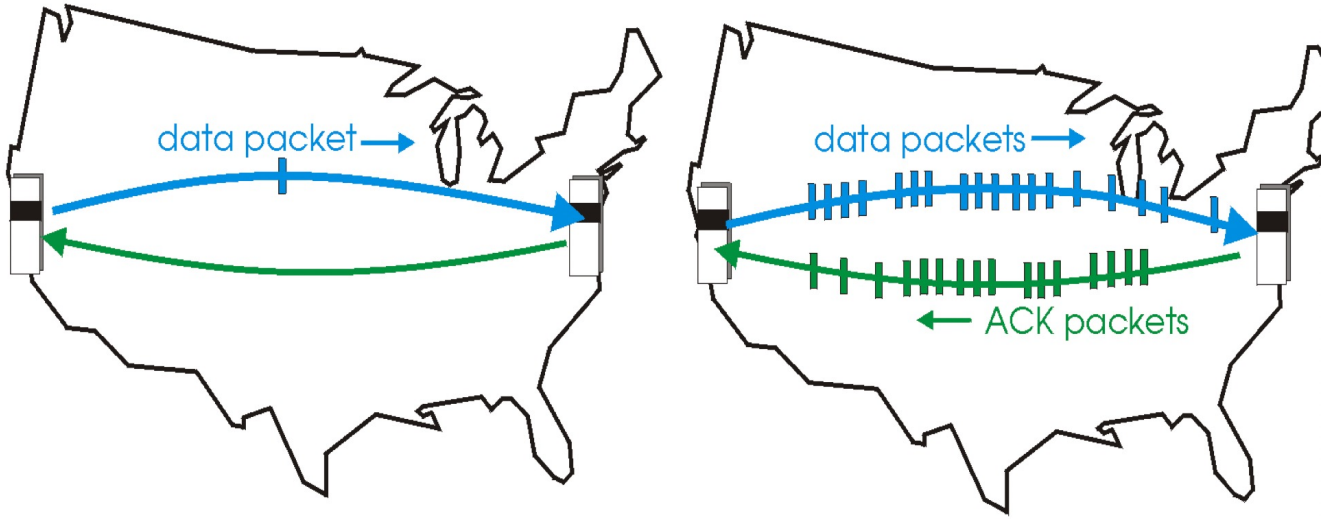


120 Kilobit/s == 1% of link rate

Pipelined：在途字节(In-flight)　6.4.3 TCP连接管理

假设发端有多个数据在途（in flight）

- 何为在途？已发送，但还未被确认（yet-to-be-acknowledged）
- 数据包与ACK（之前数据包的ACK）同时在传输

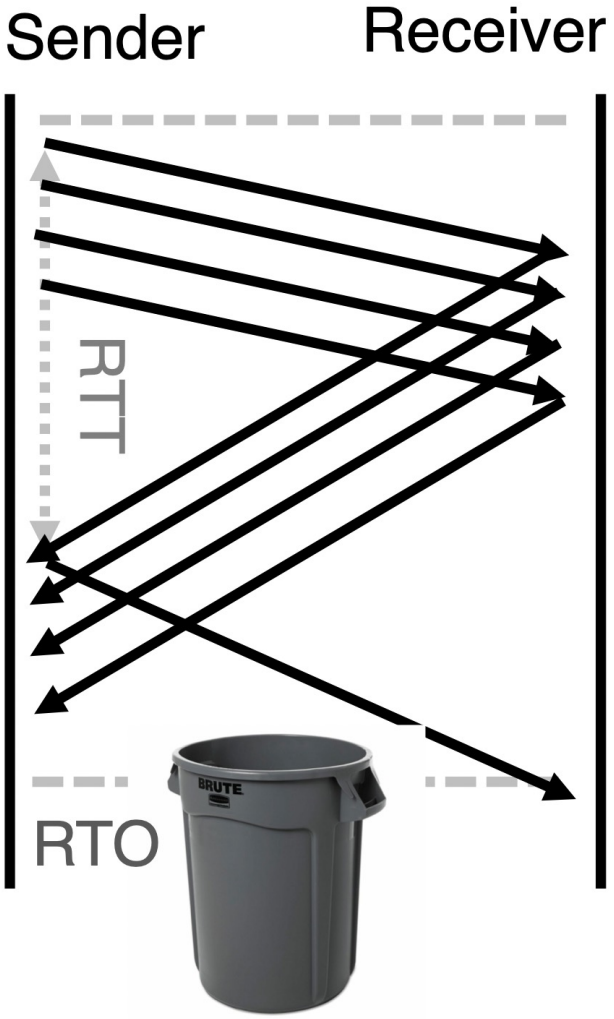


(a) a stop-and-wait protocol in operation　(b) a pipelined protocol in operation

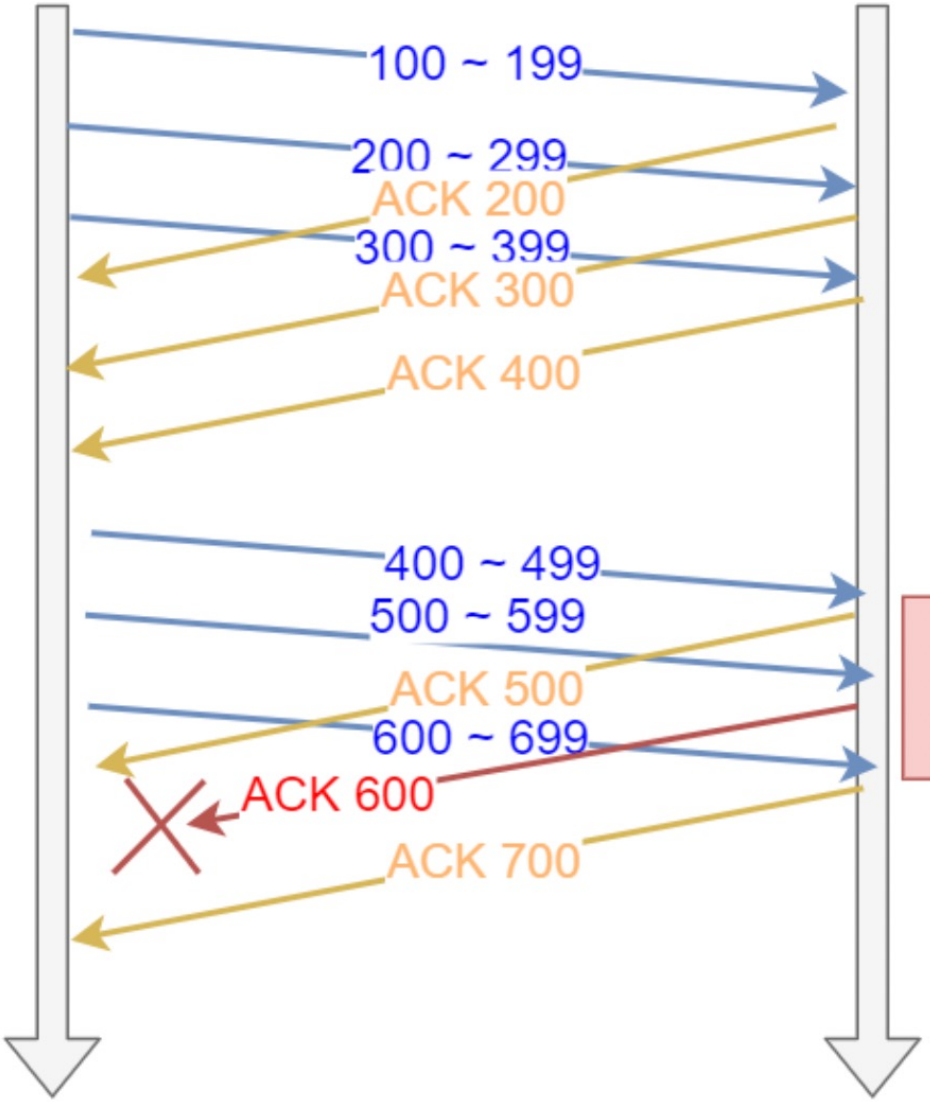
Pipelined

6.4.3 TCP连接管理

1. 如果有N个数据包在途，那么吞吐量(throughput)可以提高N倍。
2. 我们将在途(in flight)数据成为窗口(window)
3. 则在途数据量称为窗口大小(window size)



发送方 接收方



ACK 600 即使丢失了，
也不会进行数据重发，
可以通过下一个确认应答进行确认

用滑动窗口方式并行处理

有限状态机

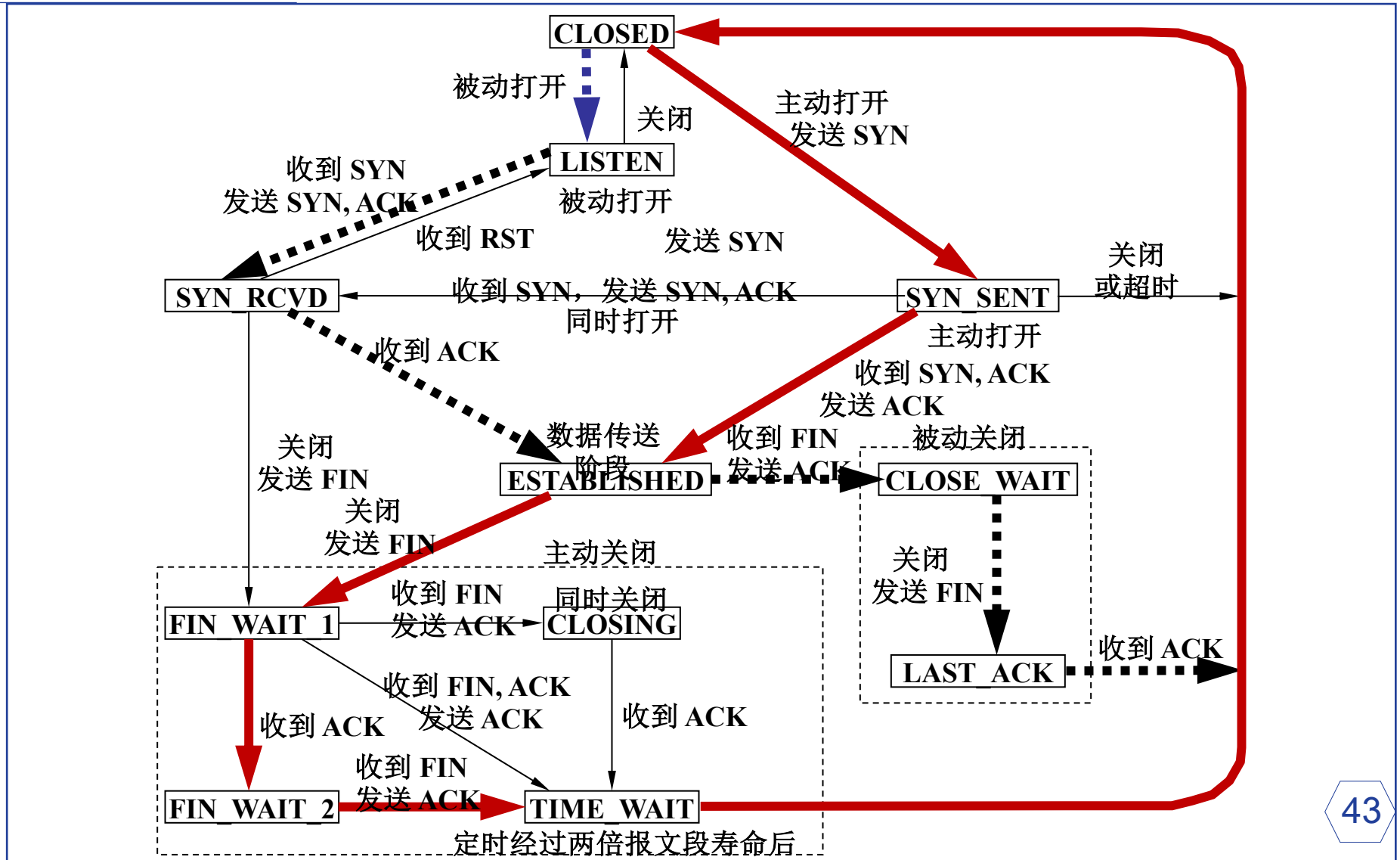
6.4.3 TCP连接管理

TCP 将连接可能处在的状态及各状态可能发生的变迁，画成状态转移图的形式，称为**有限状态机**。

图中每一个方框即TCP可能具有的状态，方框中的字是TCP标准使用的状态名；状态之间的箭头表示可能发生的状态变迁，箭头旁边的字表示变迁的原因，或状态变迁后又出现的动作。

有限状态机

6.4.3 TCP连接管理



Wireshark采集TCP报文段

6.4.2 TCP首部格式

No.	Time	Source	Destination	Protocol	Length	Info
3	0.639046	10.10.214.179	124.115.1.33	TCP	62	49796 > https [SYN] Se
4	0.950762	10.10.214.179	113.142.24.30	TCP	62	49797 > https [SYN] Se
5	1.013696	10.10.214.179	222.202.96.108	TCP	62	49799 > http [SYN] Seq
6	1.014243	222.202.96.108	10.10.214.179	TCP	60	http > 49799 [SYN, ACK
7	1.014349	10.10.214.179	222.202.96.108	TCP	54	49799 > http [ACK] Seq
8	1.014658	10.10.214.179	222.202.96.108	HTTP	252	POST / HTTP/1.1

```
⊞ Frame 6: 60 bytes on wire (480 bits), 60 bytes captured (480 bits)
⊞ Ethernet II, Src: HuaweiTe_fb:ce:c2 (00:25:9e:fb:ce:c2), Dst: Dell_2f:ec:19 (d0:67:e5:2f:ec
⊞ Internet Protocol Version 4, Src: 222.202.96.108 (222.202.96.108), Dst: 10.10.214.179 (10.1
⊟ Transmission Control Protocol, Src Port: http (80), Dst Port: 49799 (49799), Seq: 0, Ack: 1
    Source port: http (80)
    Destination port: 49799 (49799)
    [Stream index: 2]
    Sequence number: 0    (relative sequence number)
    Acknowledgment number: 1    (relative ack number)
    Header length: 24 bytes
  ⊞ Flags: 0x012 (SYN, ACK)
    Window size value: 8192
    [Calculated window size: 8192]
  ⊞ Checksum: 0xa1da [validation disabled]
  ⊞ Options: (4 bytes), Maximum segment size
  ⊞ [SEQ/ACK analysis]
```

6.4.2 TCP首部格式

源端口和目的端口字段（2 字节）。

传输层的**复用和分用**功能都要通过端口才能实现。

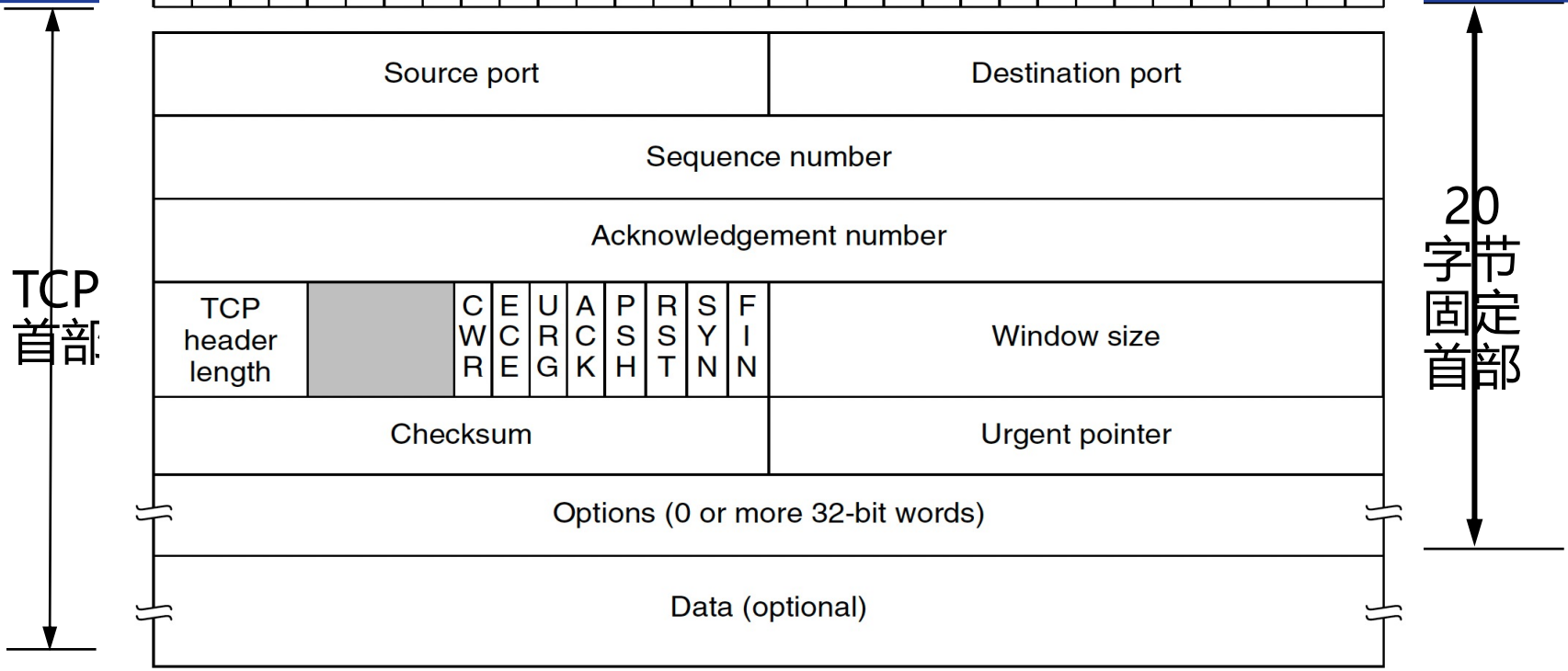
6.4.2 TCP首部格式

序号字段（4字节）。TCP 连接中传送的数据流中的**每一个字节**都编上一个序号。序号字段的值则指的是**本报文段所发送的数据的第一个字节**的序号。

6.4.2 TCP首部格式

确认号字段（4字节）。期望收到对方的**下一个报文段的数据的第一个字节**的序号。

格式



数据偏移（4bit）。指出 TCP 报文段的数据起始处距离 TCP 报文段的起始处有多远。**4字节为计算单位，e.g.，5*4=20字节，其中5为数据偏移的值。**

6.4.2 TCP首部格式

标志位	含义
URG	表明此报文段中包含紧急数据。
ACK	表明确认号字段有效。
PSH	表明应尽快将此报文段交付给接收应用程序。
RST	表明TCP连接出现严重差错，须释放连接，然后再重新建立连接。
SYN	在连接建立是用来同步序号。
FIN	用来释放一个连接。

6.4.2 TCP首部格式

紧急指针字段（2字节）。紧急指针指出在本报文段中的紧急数据的**最后一个字节**的序号。

6.4.2 TCP首部格式

窗口字段（2字节）。窗口字段用来控制对方发送的数据量，**单位为字节**。TCP 连接的一端根据设置的缓存空间大小确定自己的接收窗口大小，然后通知对方以确定对方的发送窗口的上限。

6.4.2 TCP首部格式

检验和（2字节）。检验和字段检验的范围包括**首部和数据**这两部分。在计算检验和时，要在 TCP 报文段的前面加上 12 字节的**伪首部**。

6.4.2 TCP首部格式

选项字段（长度可变）。TCP只规定了一种选项，即最大报文段长度 MSS。

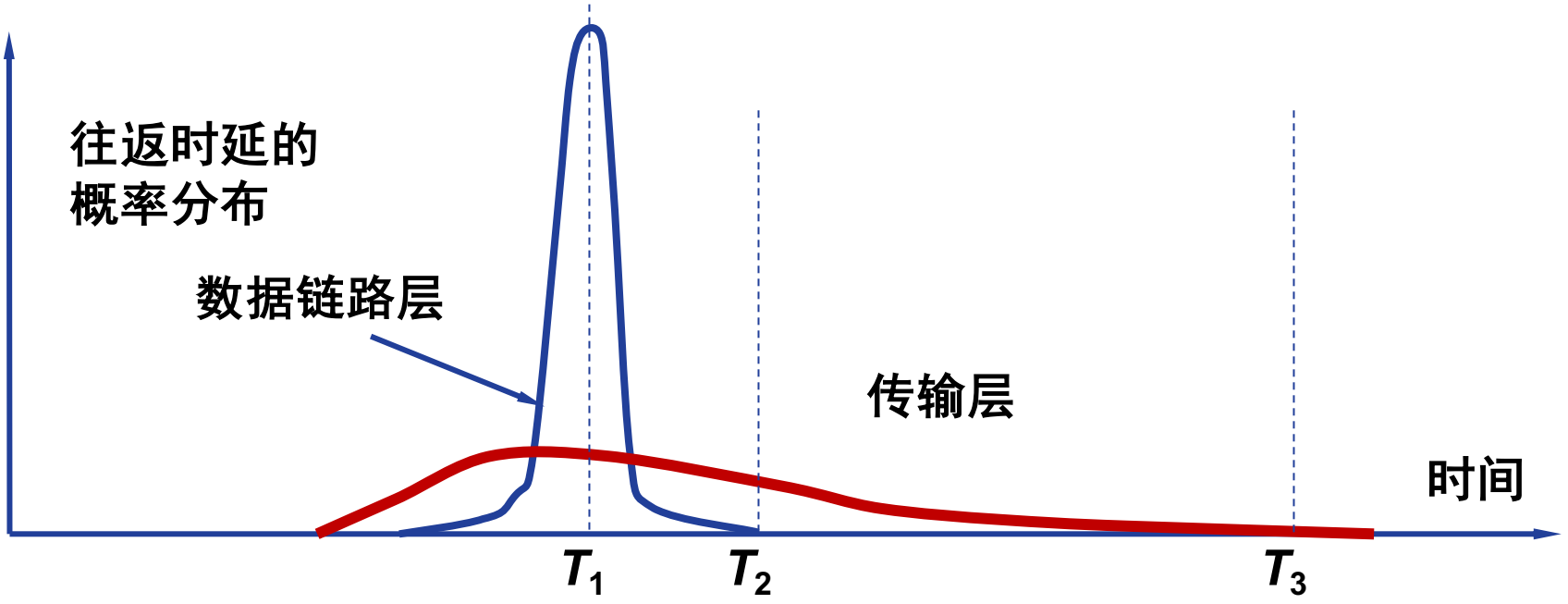
MSS = TCP报文长度 - TCP首部长度

填充字段，为了使整个首部长度是 4 字节的整数倍。

超时重传机制

6.4.4 TCP可靠传输

由于 TCP 的下层是一个互连网环境，IP 数据报所选择的路由变化很大。传输层**往返时延**的方差也很大。



RTT自适应算法

计算**平均往返时延**:

RTTnew = RTTsample （第一次测量）

RTTnew = α×RTTnew + (1 - α) ×RTTsample（第二次以后的测量）

在上式中对$0 \leq \alpha < 1$。典型的α值为7/8。

RTT自适应算法举例

已知TCP的往返时延的当前值是30 ms。现在收到了三个接连的确认报文段，它们比相应的数据报文段的发送时间分别滞后的时间是：26ms，32ms和24ms。设α=0.9。试计算新的估计的往返时延值RTTnew。

RTT自适应算法举例

$$RTTnew = 30 \times \alpha + 26 \times (1-\alpha) = 29.6\ ms$$

$$RTTnew = 29.6 \times \alpha + 32 \times (1-\alpha) = 29.84\ ms$$

$$RTTnew = 29.84 \times \alpha + 24 \times (1-\alpha) = 29.256\ ms$$

RTTD算法

引入RTT的偏差的加权平均值RTTD，计算方法如下：

RTTDnew = RTTsample / 2 （第一次测量）

RTTDnew = β×RTTDnew + (1 - β) ×

| RTTnew - RTTsample | （第二次以后的测量）

在上式中对 $0 \le \beta < 1$。典型的β值为 3/4。

RTO算法

超时重传时间RTO采用以下公式计算出来：

$$RTO = RTTnew + 4\times RTTDnew$$

Karn 提出了一个算法：在计算平均往返时延时，只要报文段重发了，就不采用其往返时延样本。这样得出的平均往返时延和重发时间较准确。

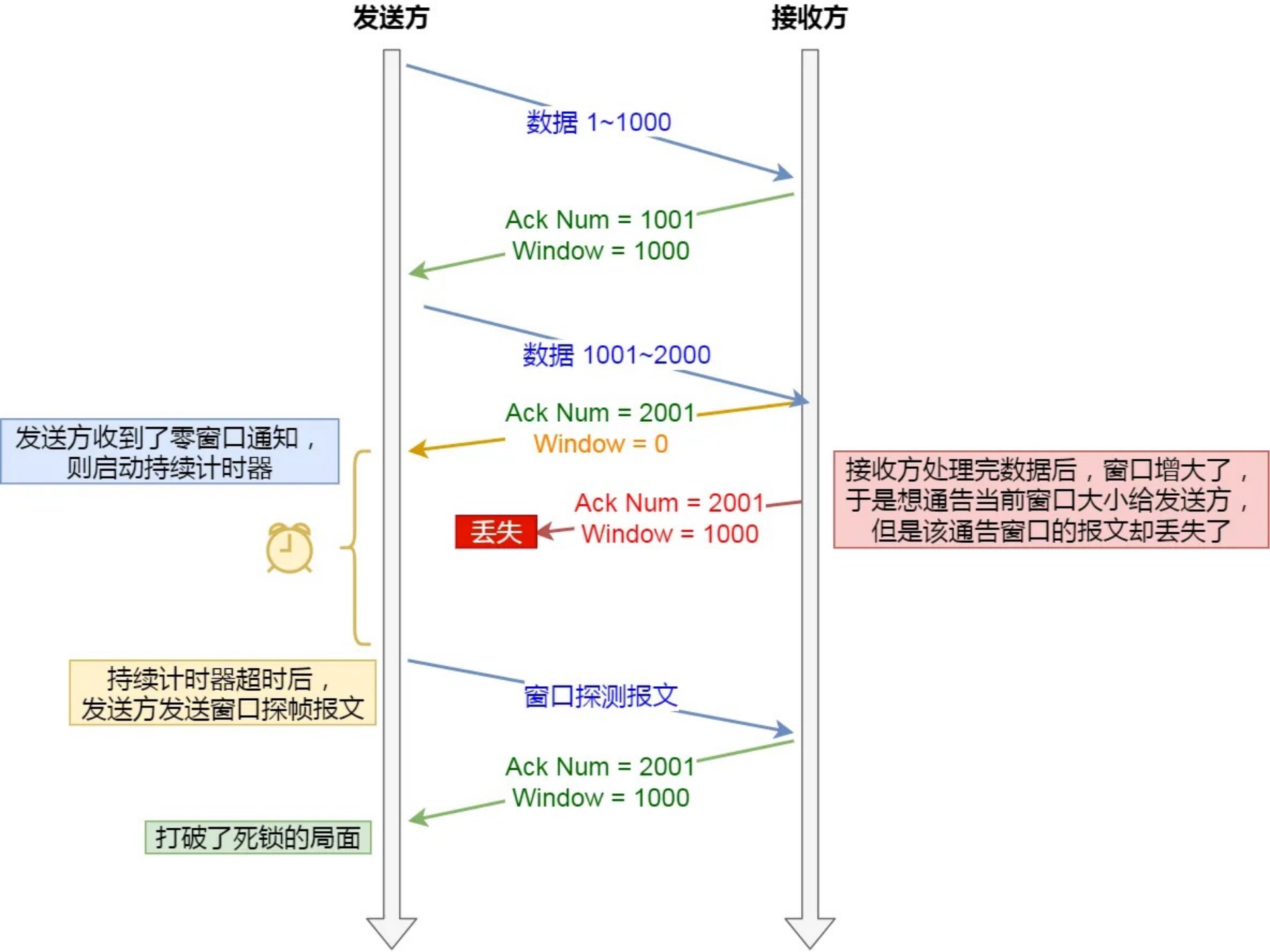
定时器设置

6.4.4 TCP可靠传输

重传定时器

持续定时器

保活定时器


发送方
接收方
数据 1~1000
Ack Num = 1001
Window = 1000
数据 1001~2000
Ack Num = 2001
Window = 0
发送方收到了零窗口通知，
则启动持续计时器
接收方处理完数据后，窗口增大了，
于是想通告当前窗口大小给发送方，
但是该通告窗口的报文却丢失了
Ack Num = 2001
丢失
Window = 1000
持续计时器超时后，
发送方发送窗口探帧报文
窗口探测报文
Ack Num = 2001
Window = 1000
打破了死锁的局面

基本概念

6.4.5 TCP流量控制

TCP 采用**大小可变滑动窗口**的方式进行流量控制。窗口大小的单位是字节。

根据接收方接收能力，通过**接收窗口rwnd**（receive window）可以实现端到端的流量控制，接收端将接收窗口rwnd的值放在 TCP 报文的首部中的“窗口”字段，传送给发送端。

基本概念

6.4.5 TCP流量控制

发送窗口在连接建立时由双方商定初始值。

在通信的过程中，接收端可根据自己的资源情况，随时动态地调整自己的接收窗口，然后告诉发送方，使发送方的发送窗口和自己的接收窗口一致。

这种由接收端控制发送端的做法，在计算机网络中经常使用。

举例

6.4.5 TCP流量控制

TCP采用大小可变滑动窗口的方式（rwnd）进行流量控制。根据题图的通信情况，设主机A向主机B发送数据。双方商定的窗口值是500。设每一个报文段为100字节长，序号的初始值为1。请问接收方对发送方进行了几次的流量控制?

6.4.5 TCP流量控制

举例

基本概念

6.4.6 TCP拥塞控制

拥塞控制的基本功能是避免网络发生拥塞，或者缓解已经发生的拥塞。TCP/IP拥塞控制机制主要集中在传输层实现。

流量控制

- 避免收端过载
- 发端需要根据收端buffer来调整速率

拥塞控制

- 避免链路瓶颈
- 发端根据瓶颈链路的容量或者瓶颈路由器的buffer来调整速率

基本概念

6.4.6 TCP拥塞控制

TCP为了进行有效的拥塞控制，需要通过拥塞窗口cwnd（congestion window）来进行衡量网络的拥塞程度。发送窗口的取值依据拥塞窗口和接收窗口中的较小的值，即

Min［rwnd，cwnd］

rwnd在流量控制中已阐述，在下文中将只关注cwnd。

基本概念

6.4.6 TCP拥塞控制

（1）**慢启动**：指在TCP刚建立连接或者当网络发生拥塞超时的时候，将拥塞窗口cwnd设置成一个报文段大小，并且当**cwnd≤ssthresh**时，**指数方式**增大cwnd（即每经过一个传输轮次，cwnd加倍）。

基本概念

6.4.6 TCP拥塞控制

（2）**拥塞避免**：当**cwnd≥ssthresh**时，为避免网络发生拥塞，进入拥塞避免算法，这时候以**线性方式**增大cwnd（即每经过一个传输轮次，cwnd只增大一个报文段）。

基本概念

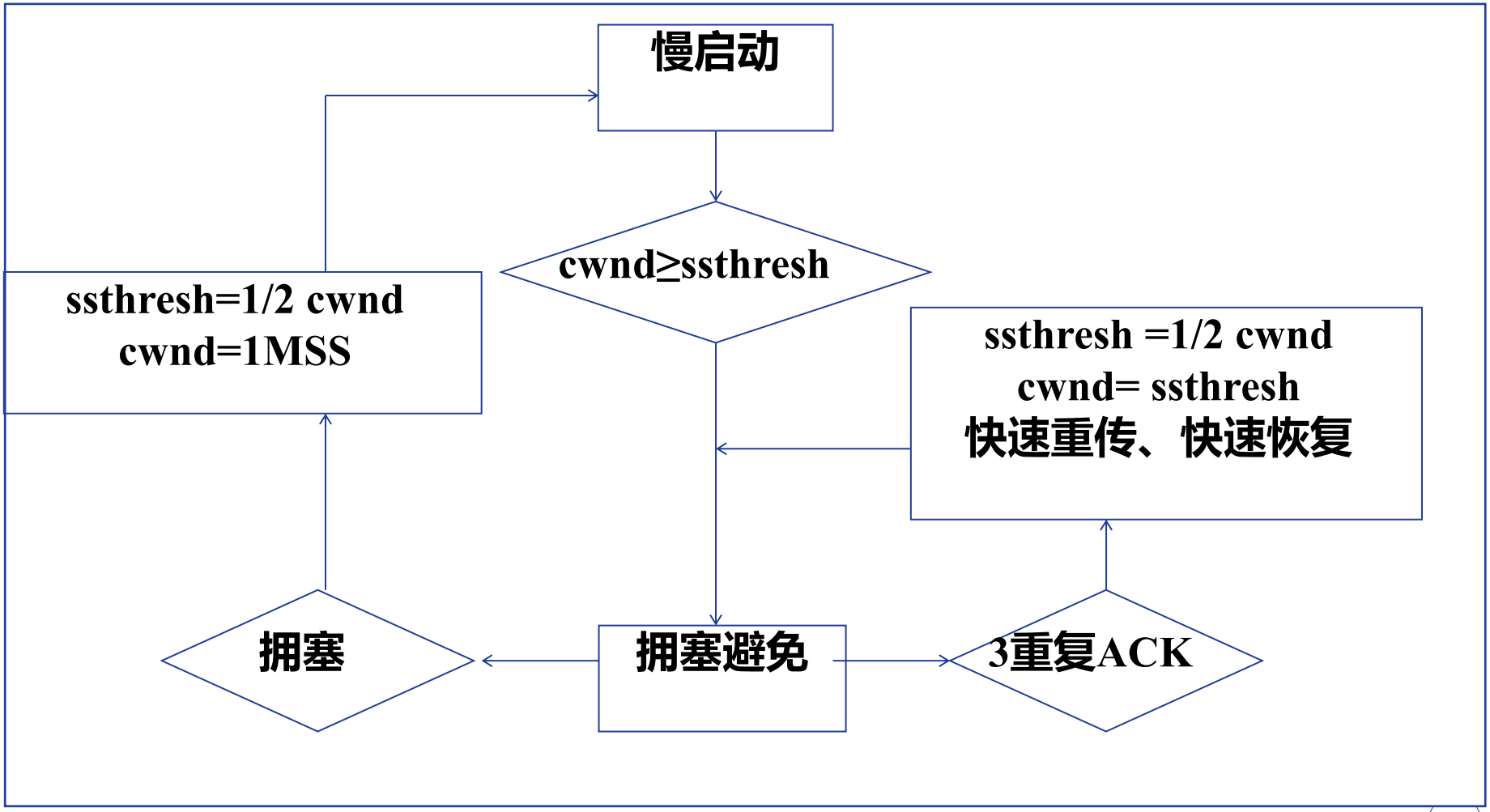
6.4.6 TCP拥塞控制

（3）**快速重传**：快速重传算法是指发送方如果连续收到**三个重复确认的ACK**，则立即重传该报文段，而不必等待重传定时器超时后再重传。 **ssthresh=1/2 cwnd（重传前），cwnd（新）=ssthresh。**

（4）**快速恢复**：快速恢复算法是指当采用快速重传算法的时候，直接执行**拥塞避免**算法。这样可以提高传输效率。

基本概念

6.4.6 TCP拥塞控制



举例

6.4.6 TCP拥塞控制

TCP的拥塞窗口cwnd大小（以报文段个数为单位）与传输轮次n的关系如图所示：

（1）请画出拥塞窗口和传输轮次的关系曲线图。

（2）请问各个传输轮次使用的是什么拥塞控制算法？

（3）各个阶段的门限值ssthresh各是多大？

（4）第40个报文段在第几个传输轮次发送？

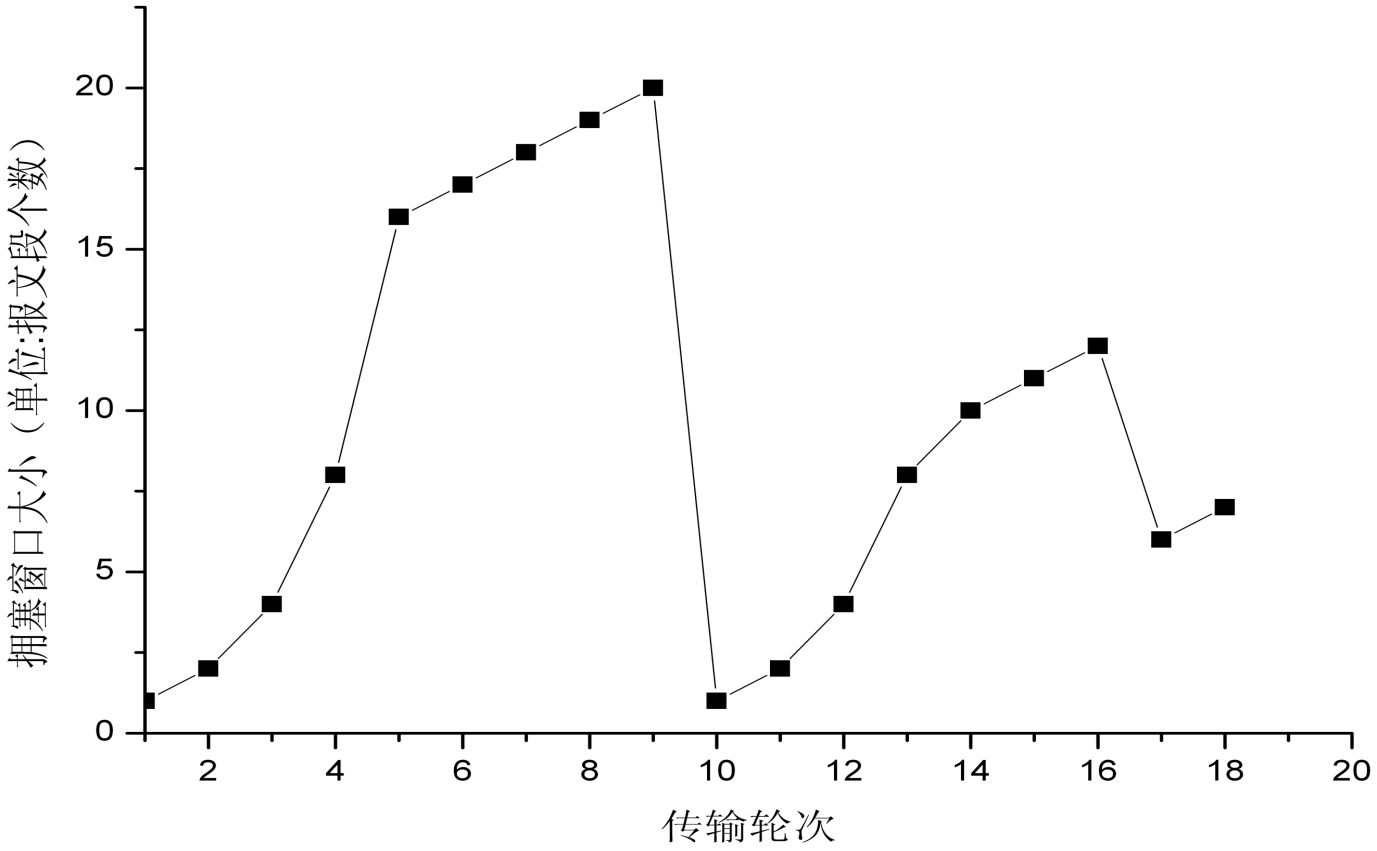
举例

6.4.6 TCP拥塞控制

CWNd	1	2	4	8	16	17	18	19	20
n	1	2	3	4	5	6	7	8	9
CWNd	1	2	4	8	10	11	12	6	7
n	10	11	12	13	14	15	16	17	18

6.4.6 TCP拥塞控制

举例



举例

6.4.6 TCP拥塞控制

（1）拥塞窗口和传输轮次的关系曲线图如图。

（2）慢开始算法的时间间隔[1，5]和[10，14]；

拥塞避免算法的时间间隔[5，9]、[14，16]和[17，18]。

（3）时间间隔[1，9]的门限值ssthresh = 16；

时间间隔[10，16]的门限值ssthresh = 10，因为在cwnd = 20的时候发生了网络的超时（其根据就是发送端没有按时收到确认），所以ssthresh =1/2 cwnd = 20 / 2 =10；

时间间隔[17，18]的门限值ssthresh = 6，因为这是收到3个重复的ACK，所以进入快速重传算法，这时候ssthresh =1/2 cwnd = 12 / 2 =6。

（4）表中传输轮次可发送的报文段个数为依据，所以第40个报文段在第6传输轮次。

$1 + 2 + 4 + 8 + 16 < 40 < 1 + 2 + 4 + 8 + 16 + 17$

6.4.6 TCP拥塞控制

举例

TCP最大长度为1000字节，若主机甲的当前拥塞窗口为5000字节（初始的接收窗口足够大），假定报文段初始的序号是1，在主机甲向乙连续发送2个最大段后，成功收到主机乙发送的第一段的确认段，确认段中通告的接收窗口大小为3000字节，此时主机甲还可以向主机乙发送的报文段序号分别是多少？

6.4.6 TCP拥塞控制

举例

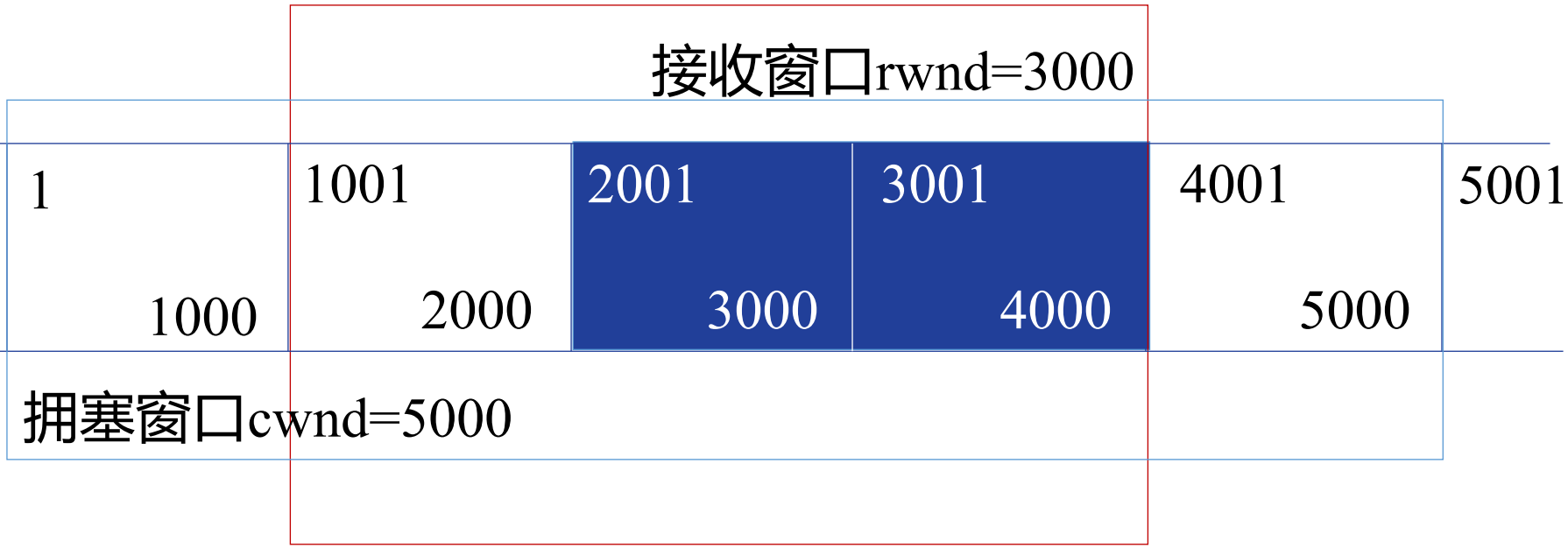
发送方的发送窗口的上限值应该取接收方窗口和拥塞窗口这两个值中较小的一个，于是此时发送方的发送窗口为**MIN{5000B, 3000B}=3000B**。

由于发送方还没有收到第二个最大段的确认，所以此时主机甲还可以向主机乙发送的最大字节数为**3000B-1000B=2000B**。

第一个报文段的序号是1，**第二个报文段的序号是1001（尚未收到确认）**。所以，还可以发送的两个TCP报文段的序号分别是：**2001和3001**。

6.4.6 TCP拥塞控制

举例



6.4.7 TCP应用

协议名称	协议	默认端口	使用TCP协议原因说明
文件传输	FTP	20 21	要求保证数据传输的可靠性
远程终端接入	TELNET	23	要求保证字符正确传输
邮件传输	SMTP POP3	25 110	要求保证邮件从发送方正确到达接收方
万维网	HTTP	80	要求可靠的交换超媒体信息

Thank You

Have A Nice Day

南京邮电大学通信与信息工程学院

“计算机通信与网络” 国家精品课程组