



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



计算机网络之探赜索隐

主讲人：聂兰顺

本讲主题

传输层



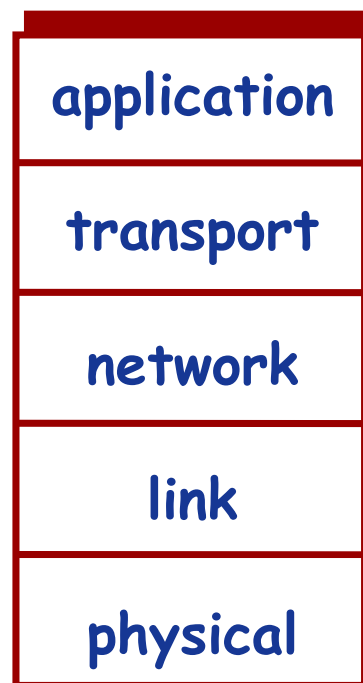
传输层？

❖ 理解传输层服务的基本理论和基本机制

- 复用/分用
- 可靠数据传输机制
- 流量控制机制
- 拥塞控制机制

❖ 掌握Internet的传输层协议

- UDP：无连接传输服务
- TCP：面向连接的传输服务
- TCP拥塞控制





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



谢谢!



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



计算机网络之探赜索隐

主讲人：聂兰顺

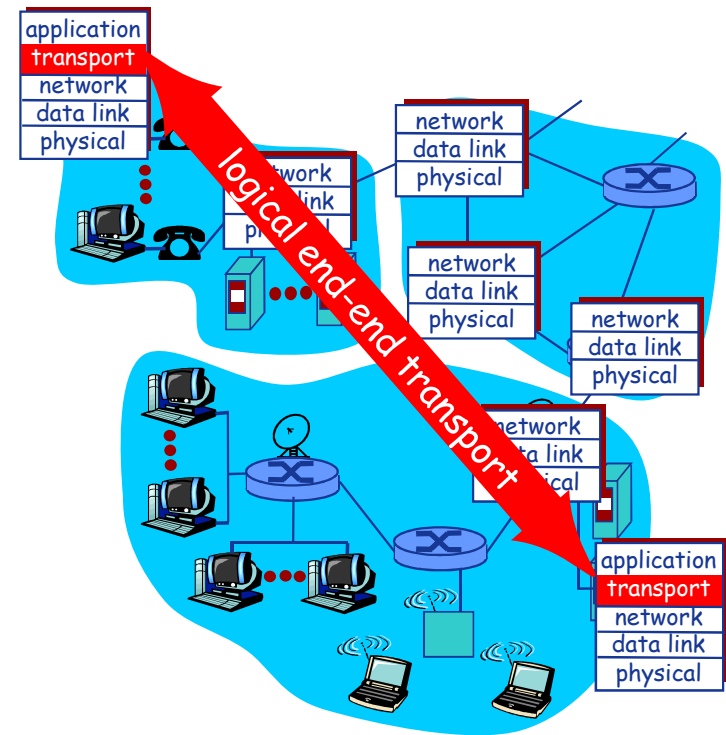
本讲主题

传输层服务概述



传输层服务和协议

- ❖ 传输层协议为运行在不同Host上的进程提供了一种逻辑通信机制
- ❖ 端系统运行传输层协议
 - **发送方**：将应用递交的消息分成一个或多个的Segment，并向下传给网络层。
 - **接收方**：将接收到的segment组装成消息，并向上交给应用层。
- ❖ 传输层可以为应用提供多种协议
 - Internet上的TCP
 - Internet上的UDP



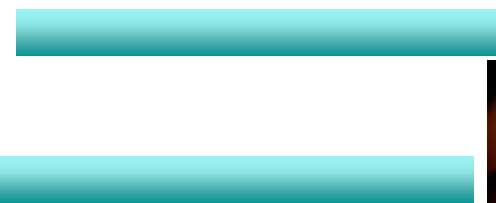
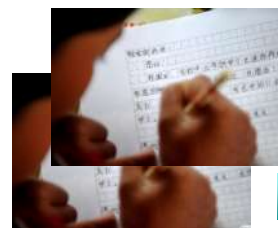
传输层 vs. 网络层

- ❖ 网络层：提供主机之间的逻辑通信机制
- ❖ 传输层：提供应用进程之间的逻辑通信机制
 - 位于网络层之上
 - 依赖于网络层服务
 - 对网络层服务进行（可能的）增强

家庭类比:

12个孩子给12个孩子发信

- ❖ 应用进程 = 孩子
- ❖ 应用消息 = 信封里的信
- ❖ 主机 = 房子
- ❖ 传输层协议 = 李雷和韩梅梅
- ❖ 网络层协议 = 邮政服务



Internet传输层协议

❖ 可靠、按序的交付服务(TCP)

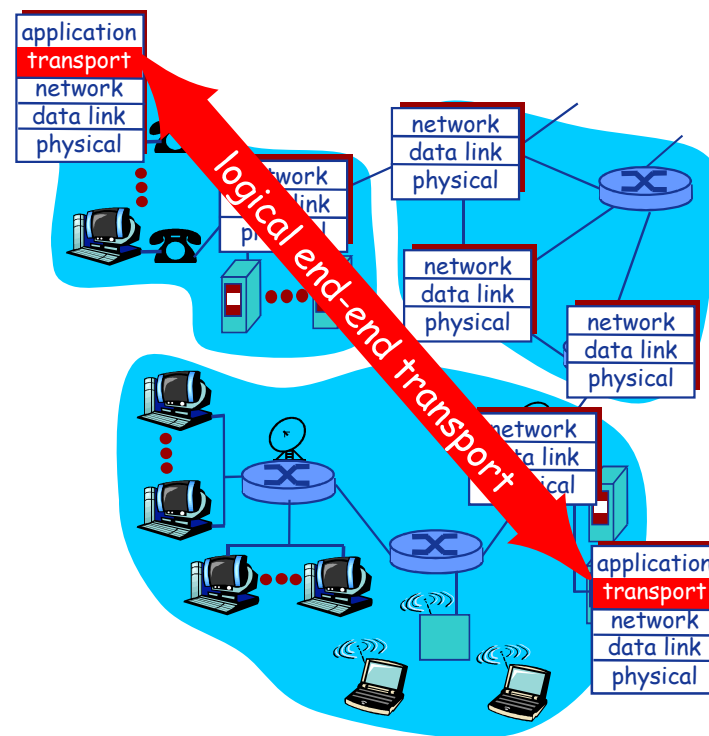
- 拥塞控制
- 流量控制
- 连接建立

❖ 不可靠的交付服务(UDP)

- 基于“**尽力而为(Best-effort)**”的网络层，没有做（可靠性方面的）扩展

❖ 两种服务均不保证

- 延迟
- 带宽





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



谢谢!



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



计算机网络之探赜索隐

主讲人：聂兰顺

本讲主题

多路复用和多路分用



多路复用/分用

❖ Why?

- ❖ 如果某层的一个协议对应直接上层的多个协议/实体，则需要复用/分用

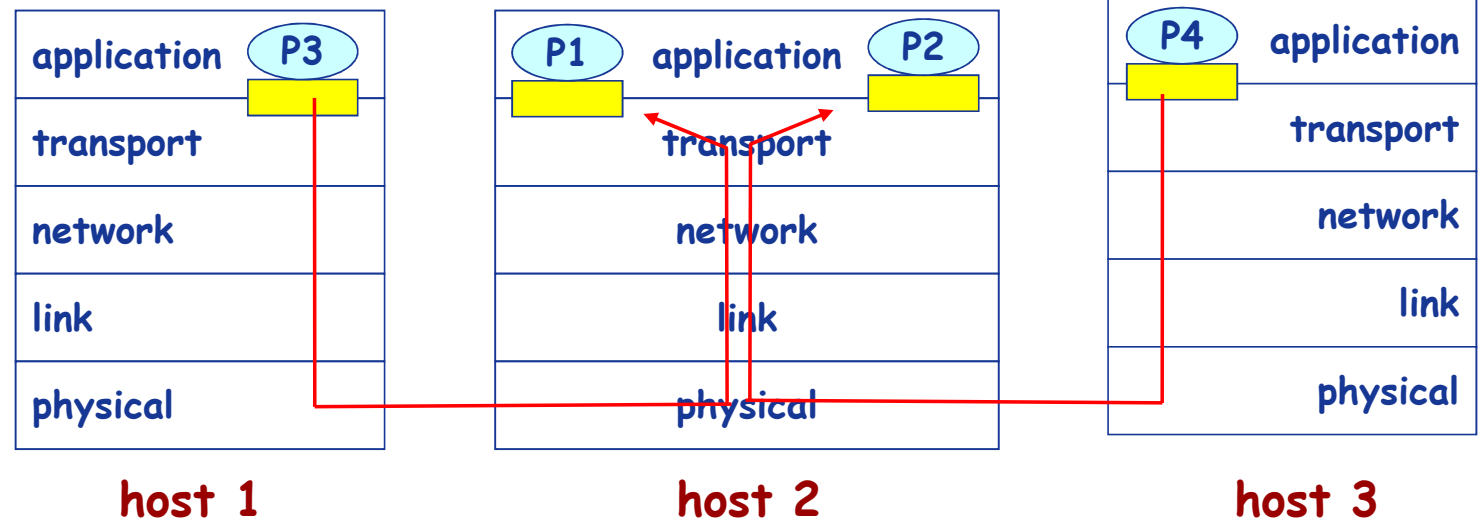
接收端进行多路分用：

传输层依据头部信息将收到的 **Segment** 交给正确的 **Socket**，即不同的进程

发送端进行多路复用：

从多个 **Socket** 接收数据，为每块数据封装上头部信息，生成 **Segment**，交给网络层

■ = socket ○ = process



分用如何工作？

❖ 主机接收到IP数据报(datagram)

- 每个数据报携带源IP地址、目的IP地址。
- 每个数据报携带一个传输层的段(Segment)。
- 每个段携带源端口号和目的端口号

❖ 主机收到Segment之后，传输层协议提取IP地址和端口号信息，将Segment导向相应的Socket

- TCP做更多处理



TCP/UDP 段格式



无连接分用

❖ 利用端口号创建Socket

```
DatagramSocket mySocket1 = new  
    DatagramSocket(99111);  
DatagramSocket mySocket2 = new  
    DatagramSocket(99222);
```

❖ UDP的Socket用二元组标识

- (目的IP地址, 目的端口号)

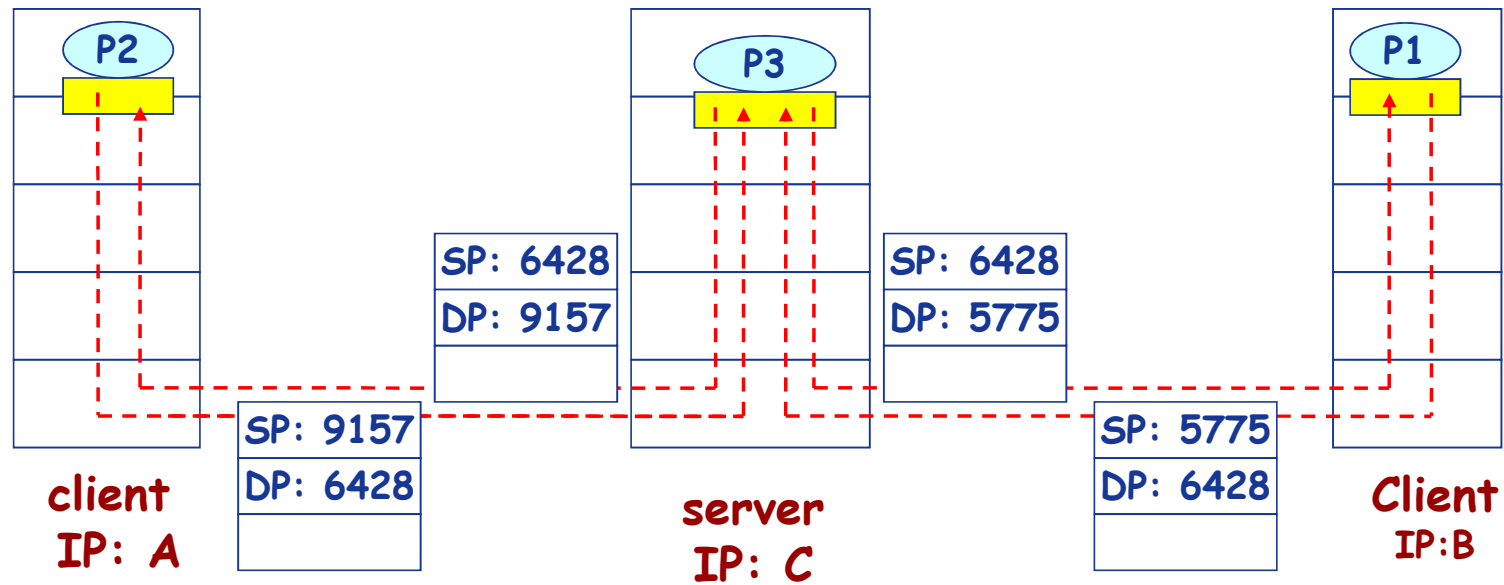
❖ 主机收到UDP段后

- 检查段中的目的端口号
 - 将UDP段导向绑定在该端口号的Socket
- ## ❖ 来自不同源IP地址和/或源端口号的IP数据包被导向同一个Socket



无连接分用

```
DatagramSocket serverSocket = new DatagramSocket(6428);
```



SP 提供“返回地址”



面向连接的分用

❖ TCP的Socket用四元组标识

- 源IP地址
- 源端口号
- 目的IP地址
- 目的端口号

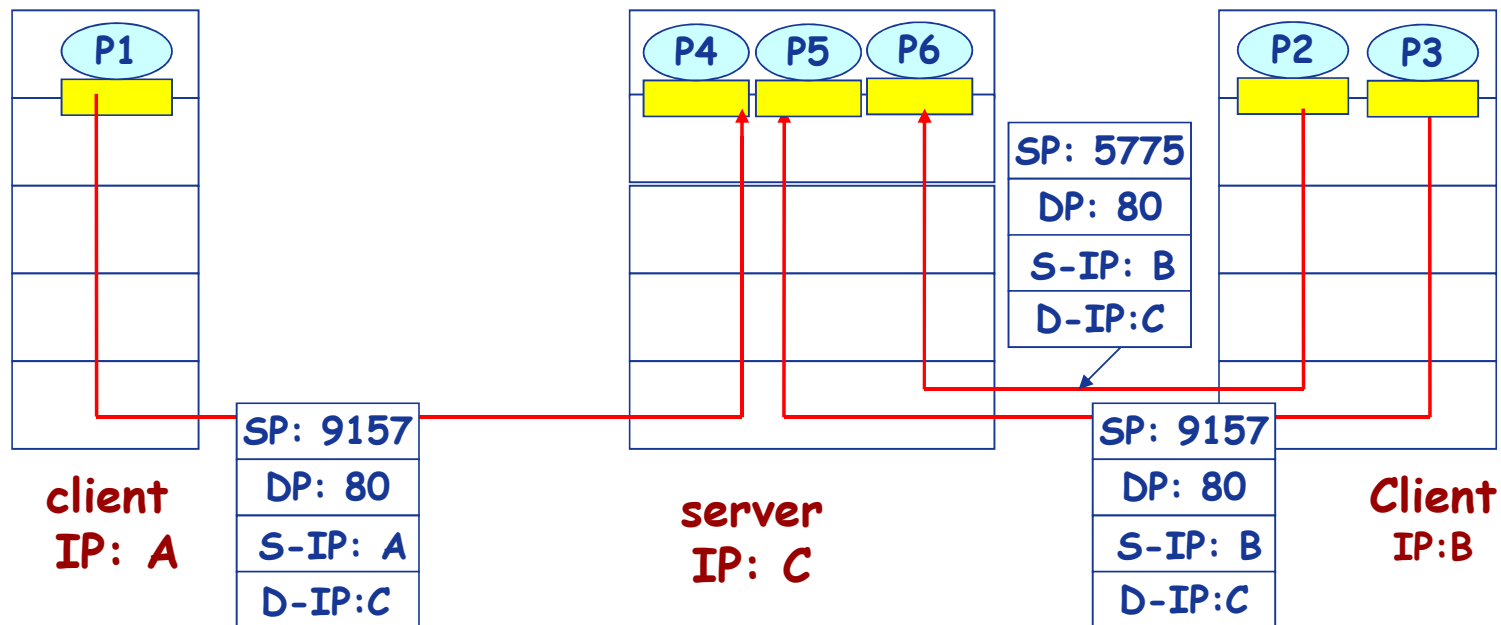
❖ 接收端利用所有的四个值将Segment导向合适的Socket

❖ 服务器可能同时支持多个TCP Socket

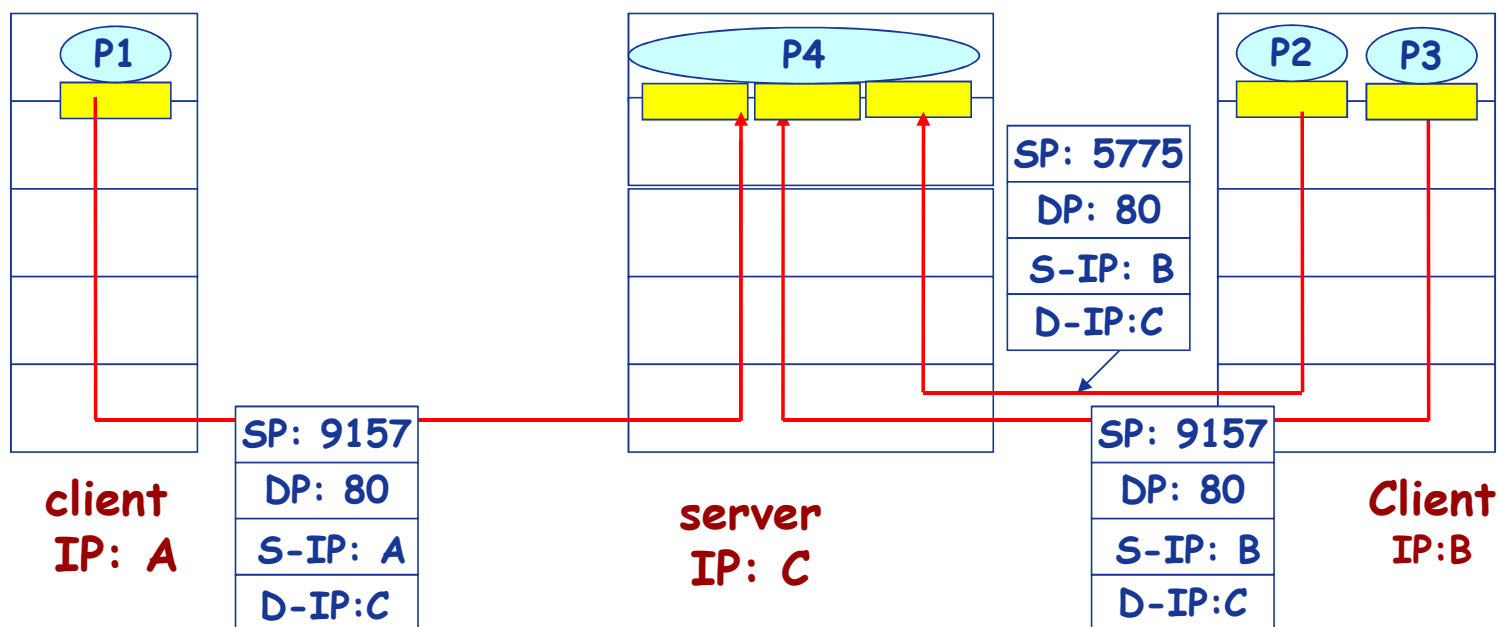
- 每个Socket用自己的四元组标识
- ## ❖ Web服务器为每个客户端开不同的Socket



面向连接的分用



面向连接的分用：多线程Web服务器





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



谢谢!



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



计算机网络之探赜索隐

主讲人：聂兰顺

本讲主题

UDP



UDP: User Datagram Protocol [RFC 768]

❖ 基于Internet IP协议

- 复用/分用
- 简单的错误校验

❖ “Best effort” 服务，UDP段可能

- 丢失
- 非按序到达

❖ 无连接

- UDP发送方和接收方之间不需要握手
- 每个UDP段的处理独立于其他段

UDP为什么存在？

- ❖ 无需建立连接 (减少延迟)
- ❖ 实现简单：无需维护连接状态
- ❖ 头部开销少
- ❖ 没有拥塞控制：应用可更好地控制发送时间和速率



UDP: User Datagram Protocol [RFC 768]

❖ 常用于流媒体应用

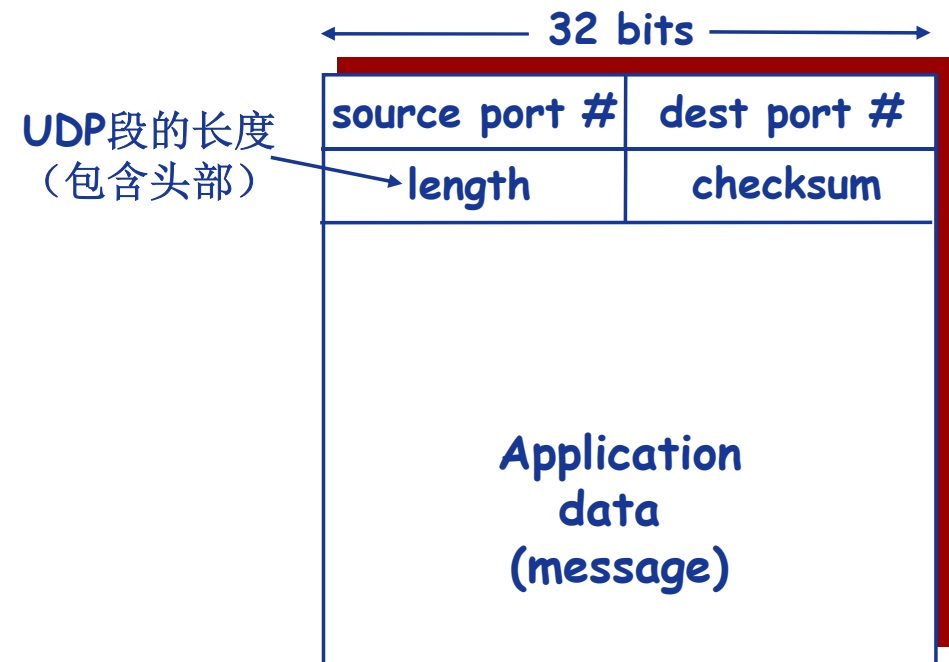
- 容忍丢失
- 速率敏感

❖ UDP还用于

- DNS
- SNMP

❖ 在UDP上实现可靠数据传输？

- 在应用层增加可靠性机制
- 应用特定的错误恢复机制



UDP segment format



UDP校验和(checksum)

目的：检测UDP段在传输中是否发生错误（如位翻转）

❖ 发送方

- 将段的内容视为**16-bit**整数
- 校验和计算：计算所有整数的和，进位加在和的后面，将得到的值按位求反，得到校验和
- 发送方将校验和放入校验和字段

❖ 接收方

- 计算所收到段的校验和
- 将其与校验和字段进行对比
 - 不相等：检测出错误
 - 相等：没有检测出错误（但可能有错误）



校验和计算示例

❖ 注意：

- 最高位进位必须被加进去

❖ 示例：

	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	
	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	
<hr/>																	
wraparound	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1
<hr/>																	
sum	1	0	1	1	1	0	1	1	1	0	1	1	1	1	0	0	
checksum	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	1	1





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



谢谢!



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



计算机网络之探赜索隐

主讲人：聂兰顺

本讲主题

可靠数据传输原理



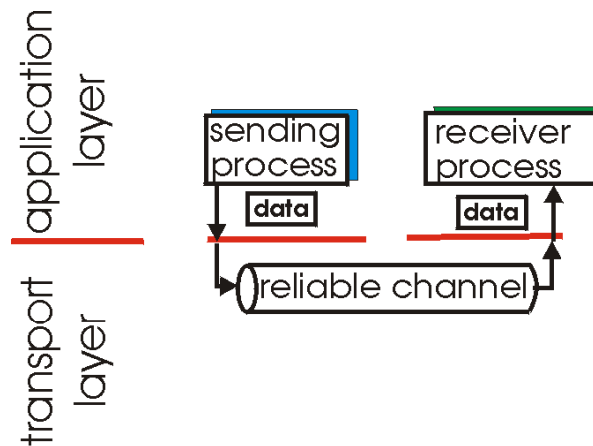
可靠数据传输原理

❖ 什么是可靠？

- 不错、不丢、不乱

❖ 可靠数据传输协议

- 可靠数据传输对应用层、传输层、链路层都很重要
- 网络Top-10问题
- 信道的不可靠特性决定了可靠数据传输协议(rdt)的复杂性



(a) provided service



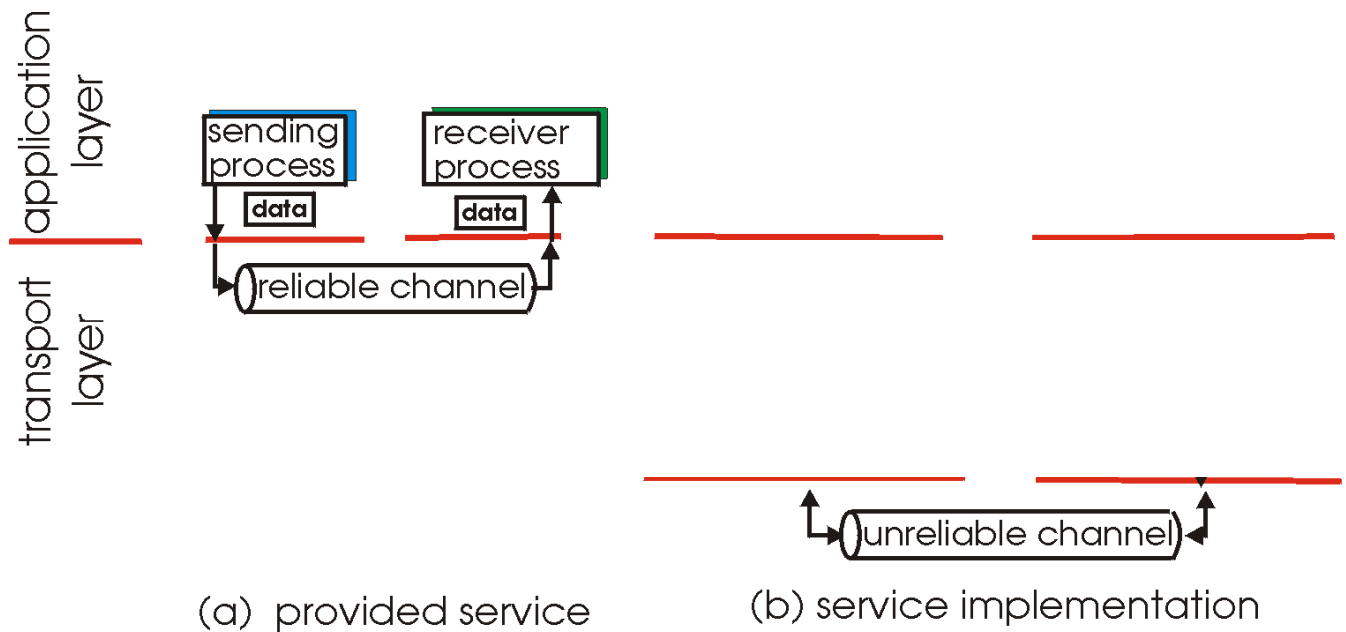
可靠数据传输原理

❖ 什么是可靠？

- 不错、不丢、不乱

❖ 可靠数据传输协议

- 可靠数据传输对应用层、传输层、链路层都很重要
- 网络Top-10问题
- 信道的不可靠特性决定了可靠数据传输协议(rdt)的复杂性



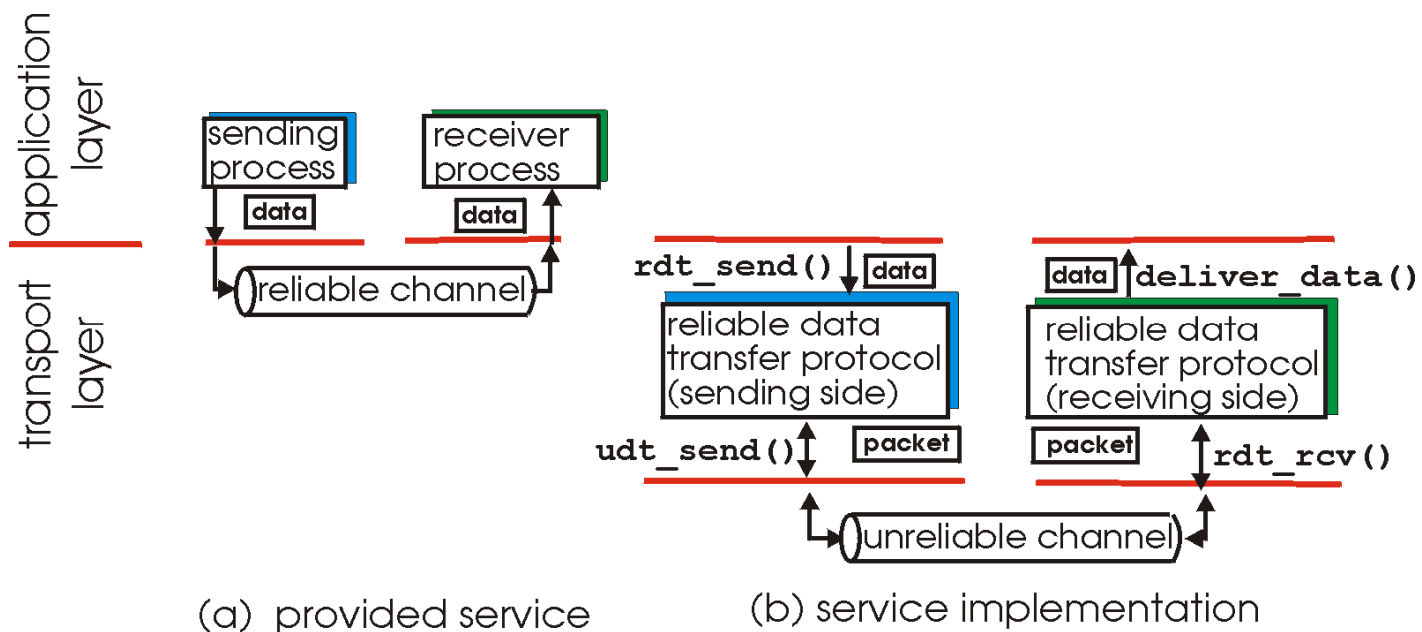
可靠数据传输原理

❖ 什么是可靠？

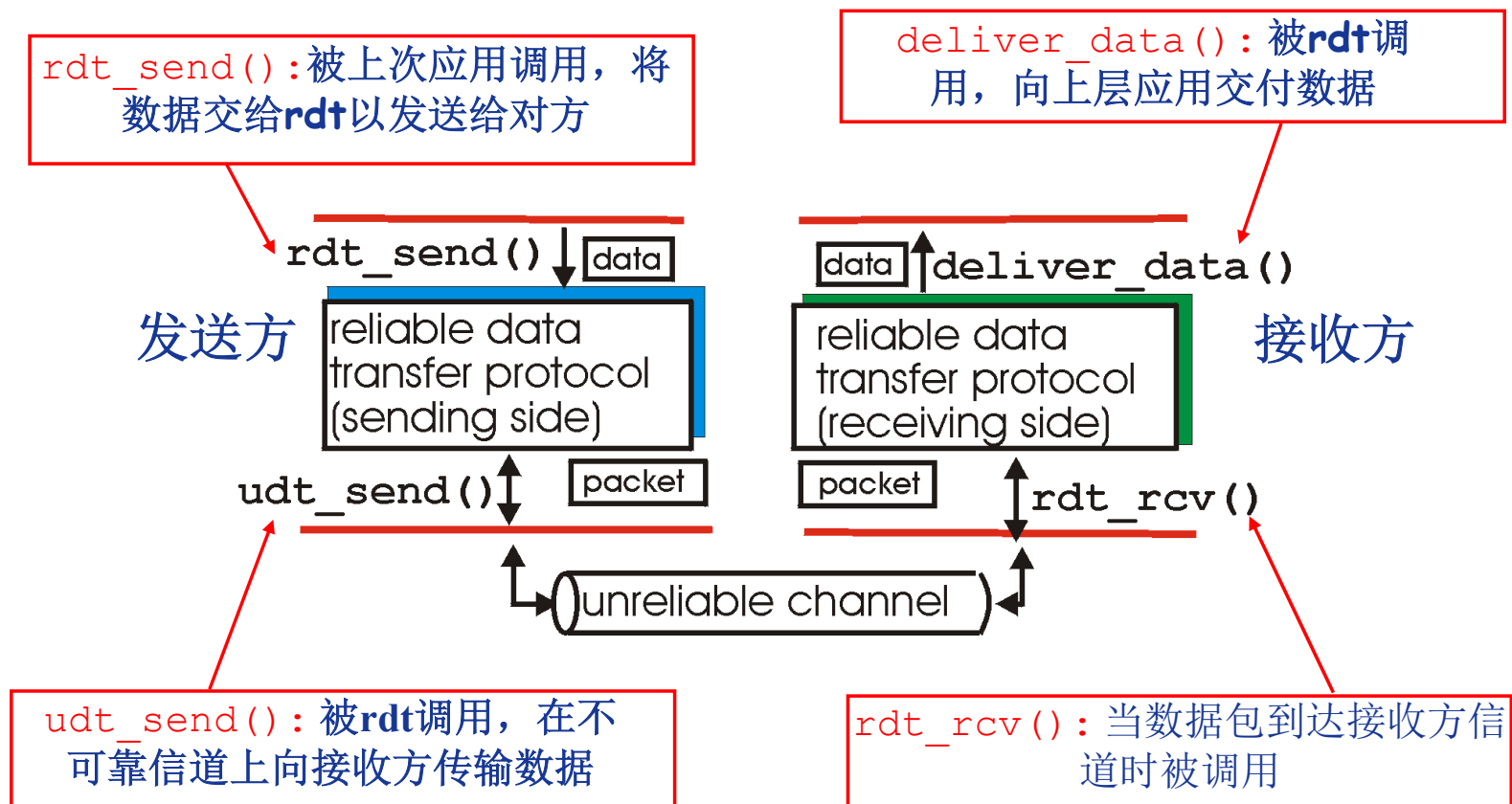
- 不错、不丢、不乱

❖ 可靠数据传输协议

- 可靠数据传输对应用层、传输层、链路层都很重要
- 网络Top-10问题
- 信道的不可靠特性决定了可靠数据传输协议(rdt)的复杂性

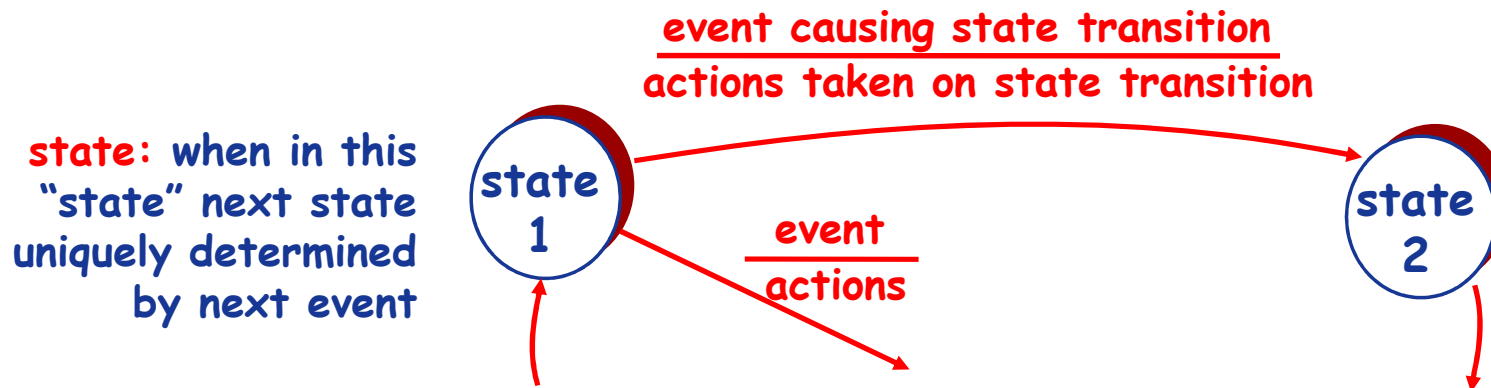


可靠数据传输协议基本结构:接口



可靠数据传输协议

- ❖ 渐进地设计可靠数据传输协议的发送方和接收方
- ❖ 只考虑单向数据传输
 - 但控制信息双向流动
- ❖ 利用状态机(Finite State Machine, FSM)刻画传输协议

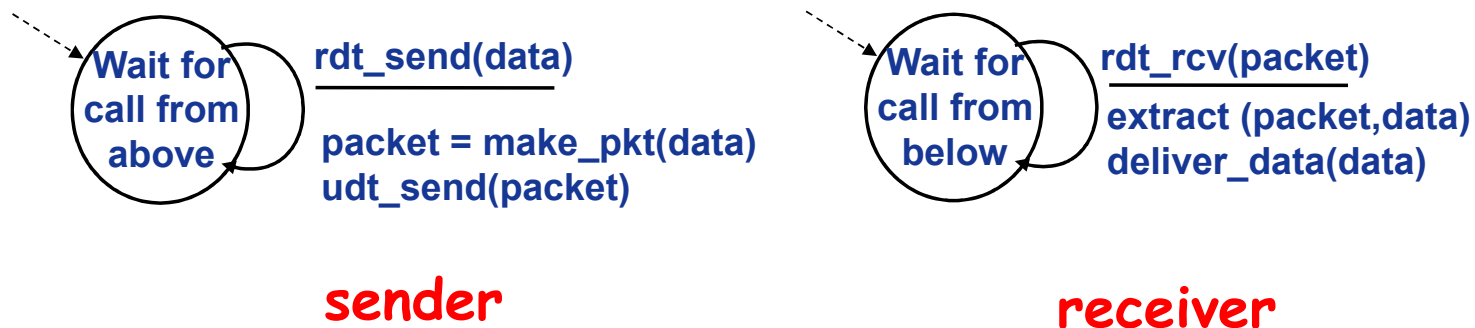


Rdt 1.0: 可靠信道上的可靠数据传输

❖ 底层信道完全可靠

- 不会发生错误(bit error)
- 不会丢弃分组

❖ 发送方和接收方的FSM独立





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



谢谢!



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



计算机网络之探赜索隐

主讲人：聂兰顺

本讲主题

Rdt 2.0



Rdt 2.0: 产生位错误的信道

❖ 底层信道可能翻转分组中的位(bit)

- 利用校验和检测位错误

❖ 如何从错误中恢复？

- 确认机制(Acknowledgements, ACK): 接收方显式地告知发送方分组已正确接收
- NAK:接收方显式地告知发送方分组有错误
- 发送方收到NAK后, 重传分组

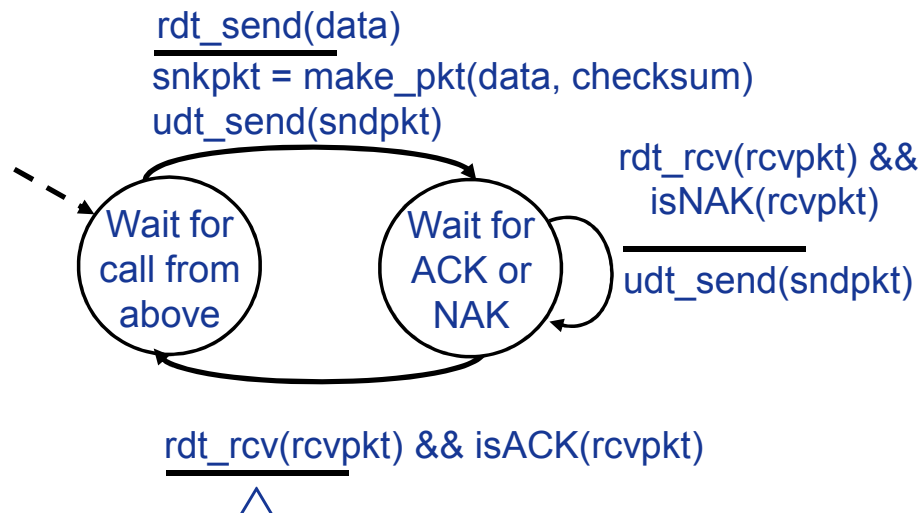
❖ 基于这种重传机制的rdt协议称为ARQ(Automatic Repeat reQuest)协议

❖ Rdt 2.0中引入的新机制

- 差错检测
- 接收方反馈控制消息: ACK/NAK
- 重传



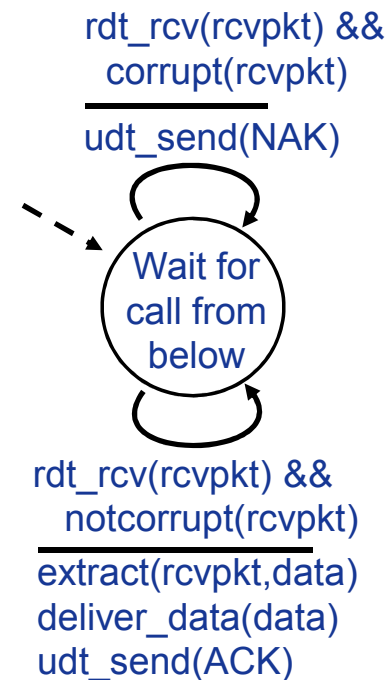
Rdt 2.0: FSM规约



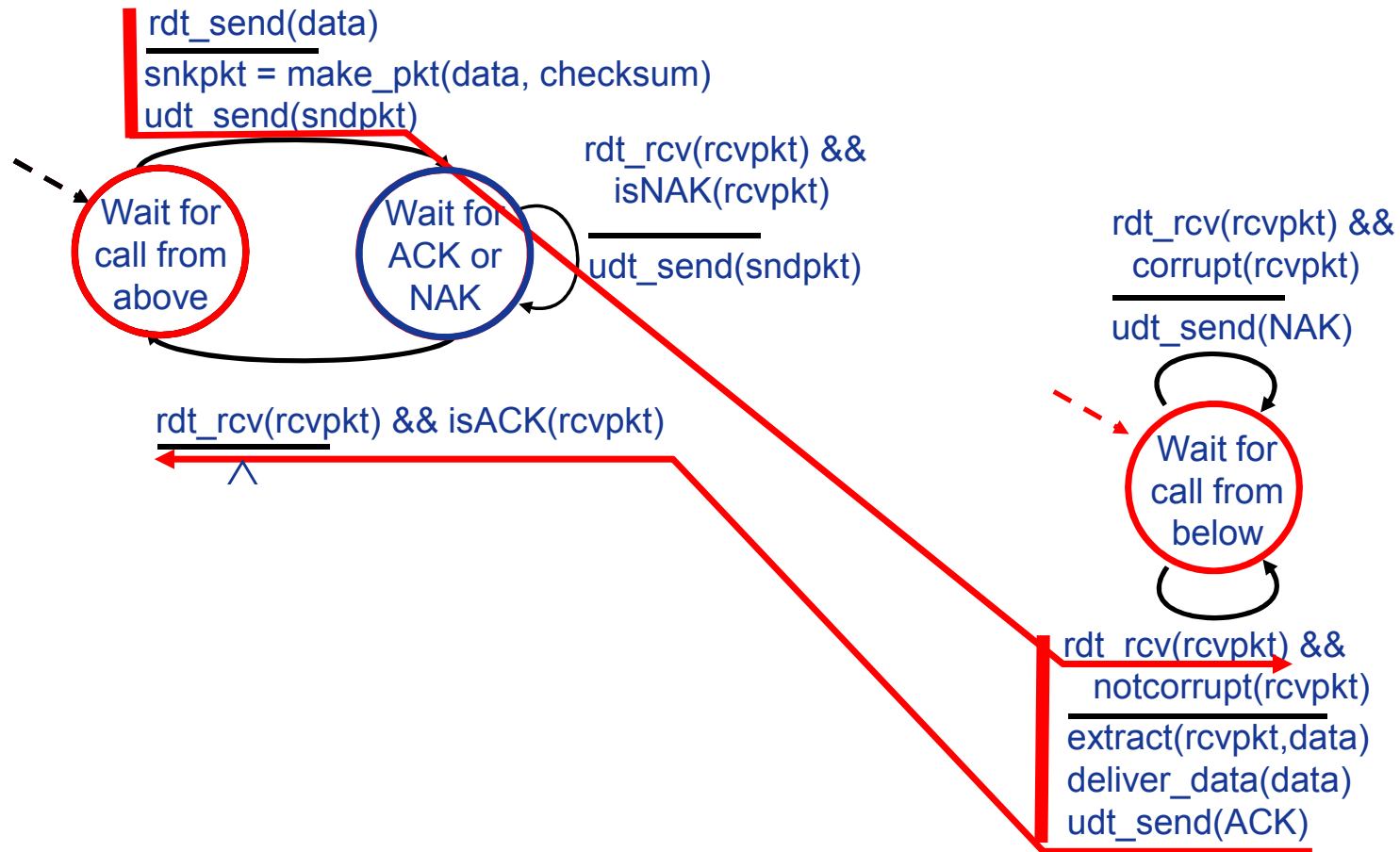
sender

停一等协议

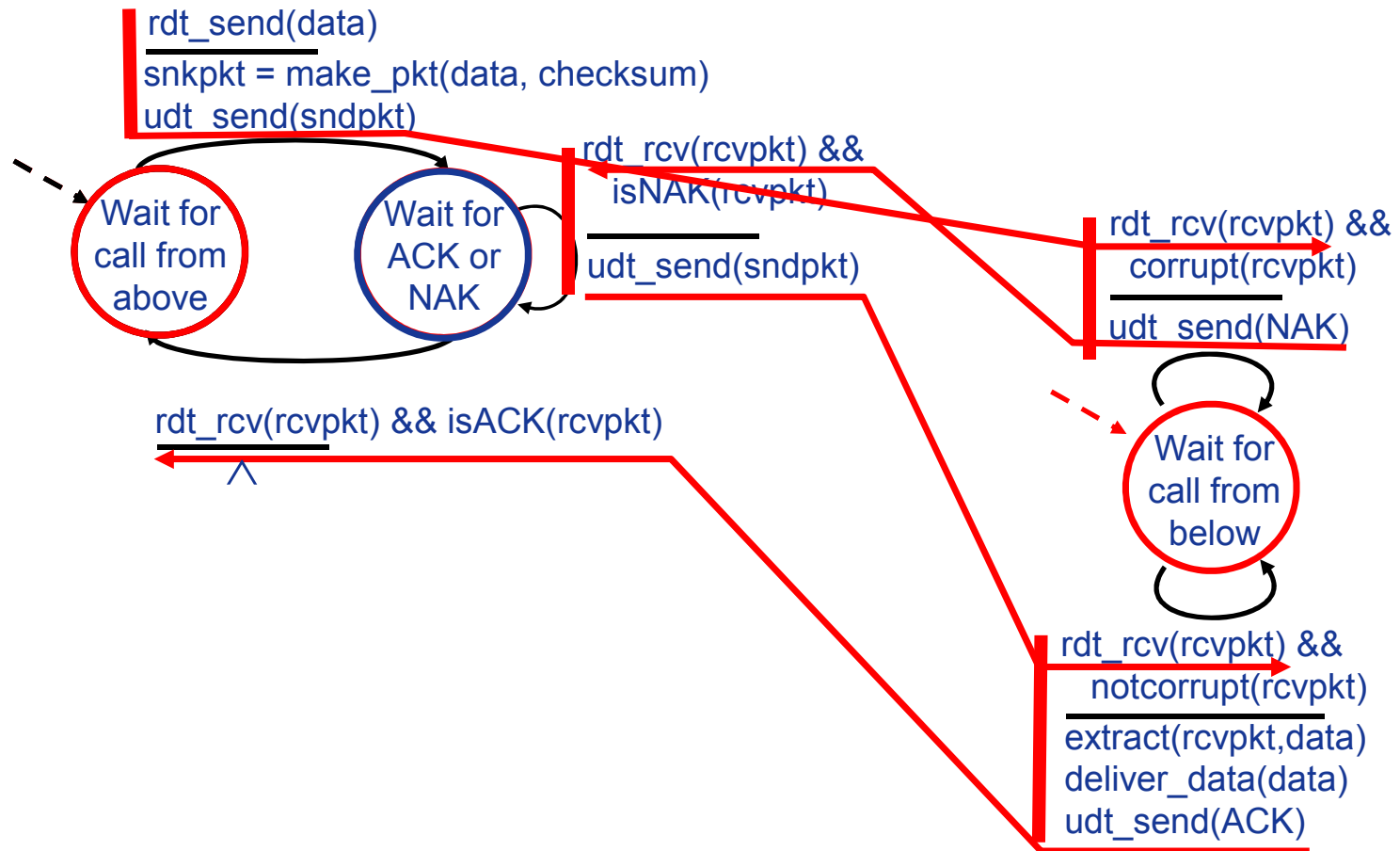
receiver



Rdt 2.0: 无错误场景



Rdt 2.0: 有错误场景





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



谢谢!



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



计算机网络之探赜索隐

主讲人：聂兰顺

本讲主题

Rdt 2.1和2.2



Rdt 2.0有什么缺陷？



❖ 如果ACK/NAK消息发生错误/被破坏(corrupted)会怎么样？

- 为ACK/NAK增加校验和，检错并纠错
- 发送方收到被破坏ACK/NAK时不知道接收方发生了什么，添加额外的控制消息
- 如果ACK/NAK坏掉，发送方重传
- 不能简单的重传：产生重复分组

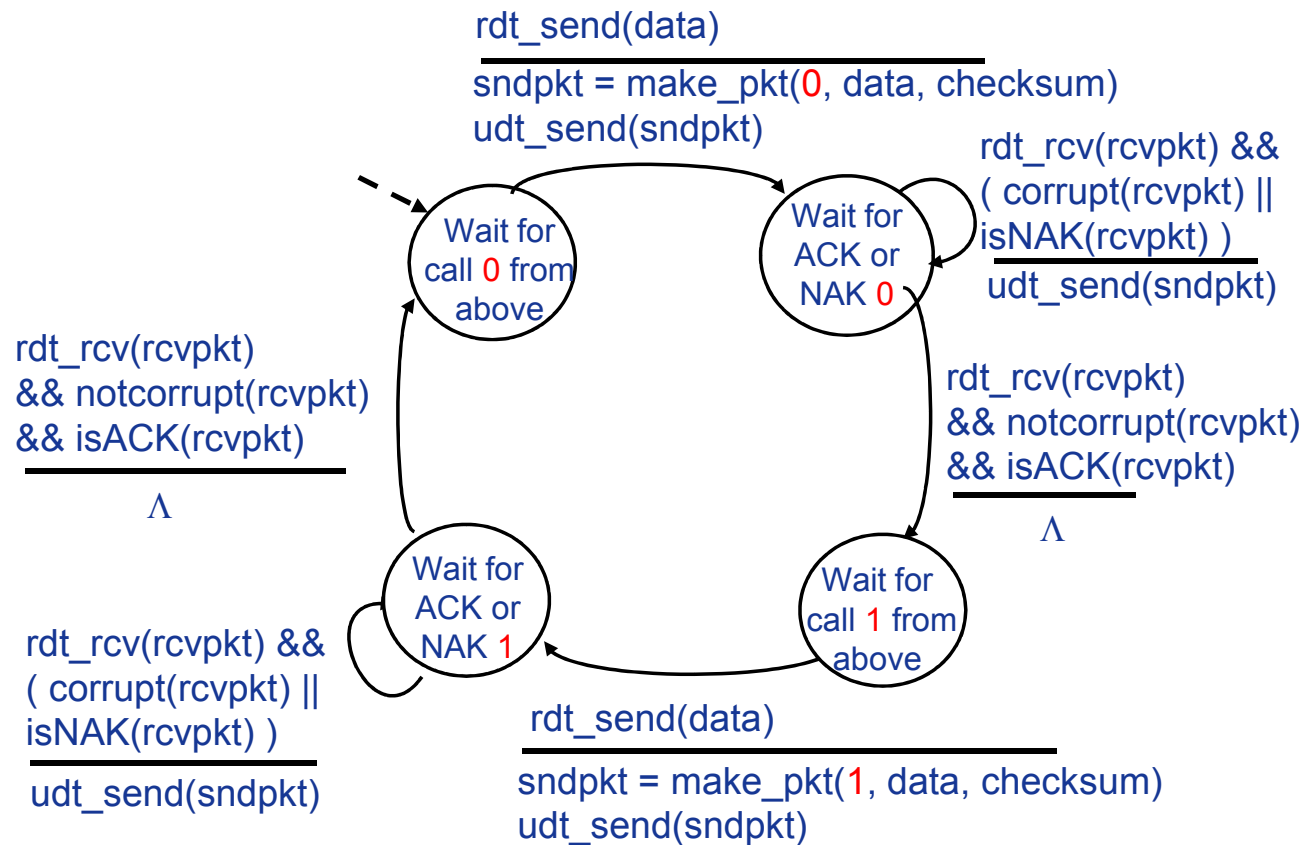
❖ 如何解决重复分组问题？

- 序列号(Sequence number): 发送方给每个分组增加序列号
- 接收方丢弃重复分组

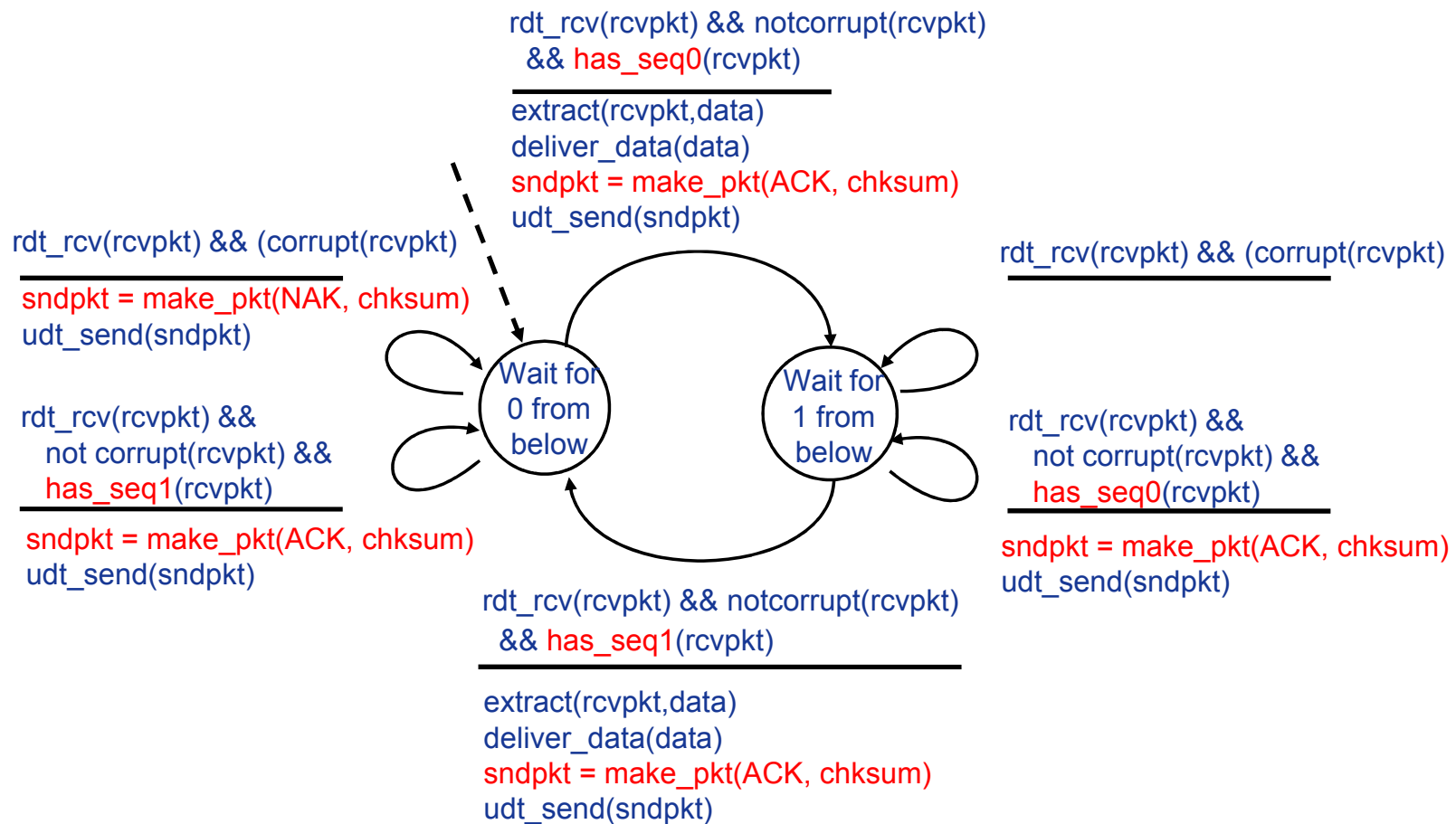
stop and wait
Sender sends one packet,
then waits for receiver
response



Rdt 2.1: 发送方, 应对ACK/NAK破坏



Rdt 2.1: 接收方, 应对ACK/NAK破坏



Rdt 2.1 vs. Rdt 2.0

❖ 发送方:

- 为每个分组增加了序列号
- 两个序列号(0, 1)就够用, 为什么?
- 需校验ACK/NAK消息是否发生错误
- 状态数量翻倍
 - 状态必须“记住”“当前”的分组序列号

❖ 接收方

- 需判断分组是否是重复
 - 当前所处状态提供了期望收到分组的序列号
- 注意: 接收方无法知道ACK/NAK是否被发送方正确收到

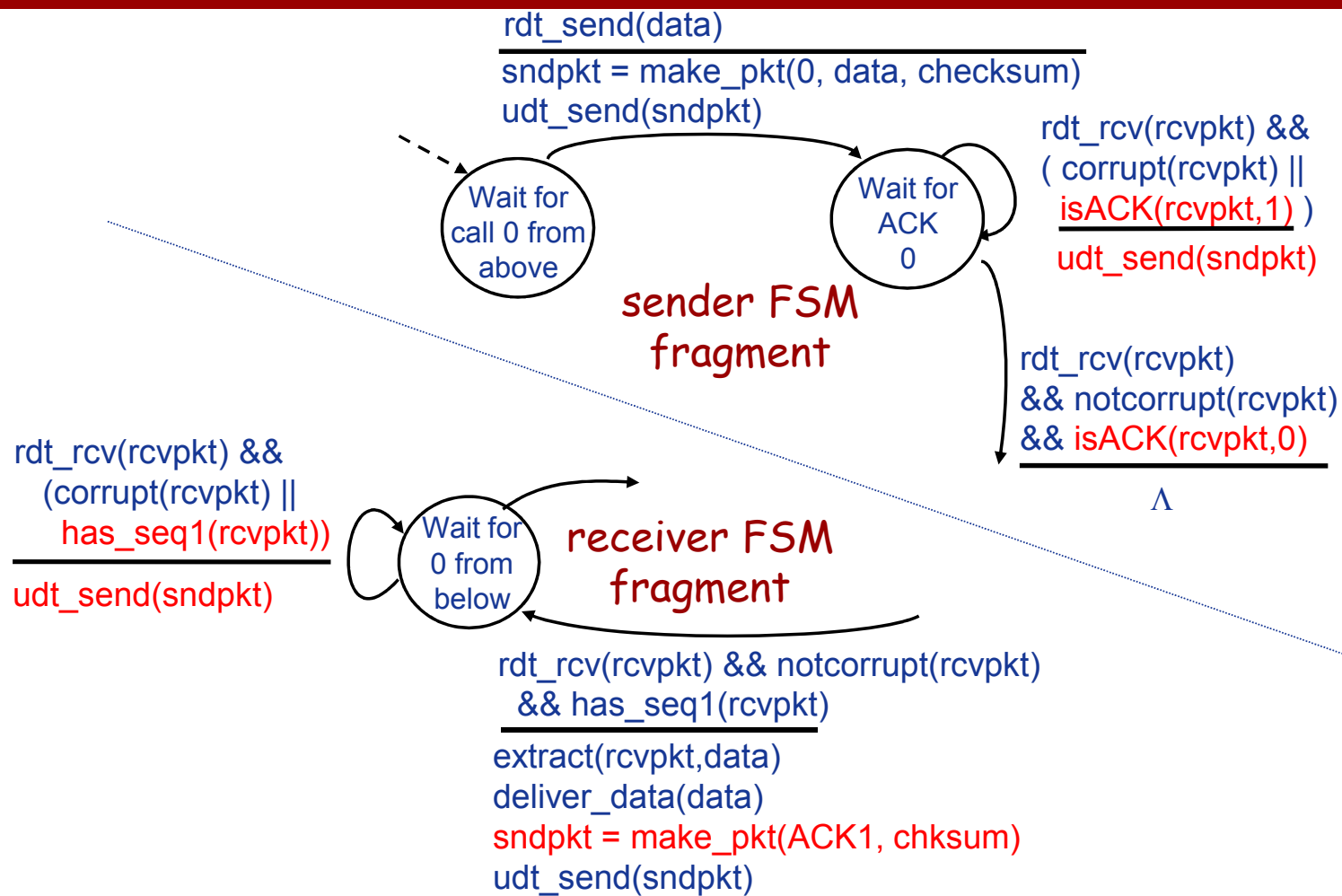


Rdt 2.2: 无NAK消息协议

- ❖ 我们真的需要两种确认消息(ACK + NAK)吗?
- ❖ 与rdt 2.1功能相同，但是只使用ACK
- ❖ 如何实现?
 - 接收方通过ACK告知最后一个被正确接收的分组
 - 在ACK消息中显式地加入被确认分组的序列号
- ❖ 发送方收到重复ACK之后，采取与收到NAK消息相同的动作
 - 重传当前分组



Rdt 2.2 FSM片段





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



谢谢!



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



计算机网络之探赜索隐

主讲人：聂兰顺

本讲主题

Rdt 3.0



Rdt 3.0

❖ 如果信道既可能发生错误，也可能丢失分组，怎么办？

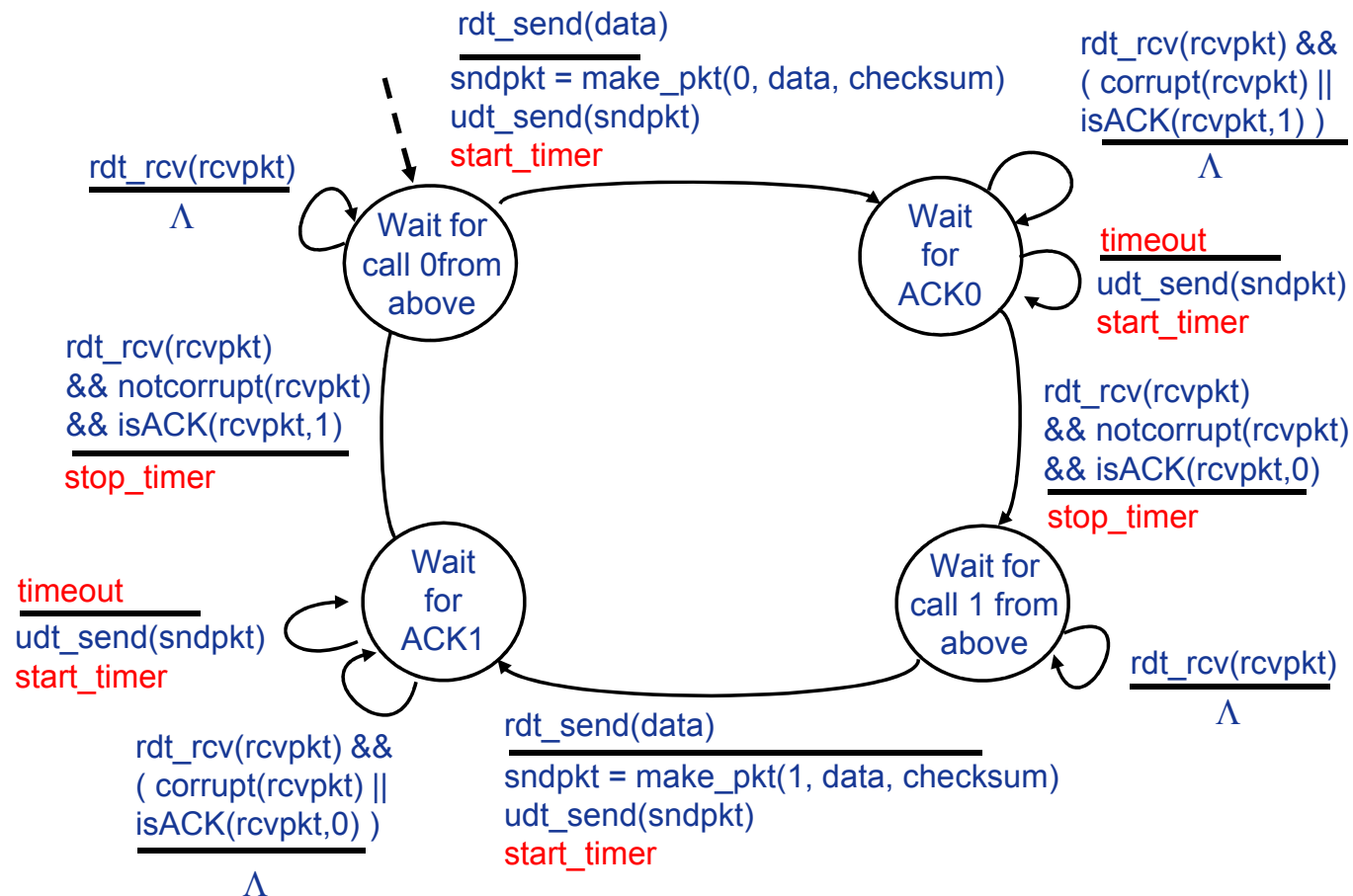
- “校验和 + 序列号 + ACK + 重传” 够用吗？

❖ 方法：发送方等待“合理”时间

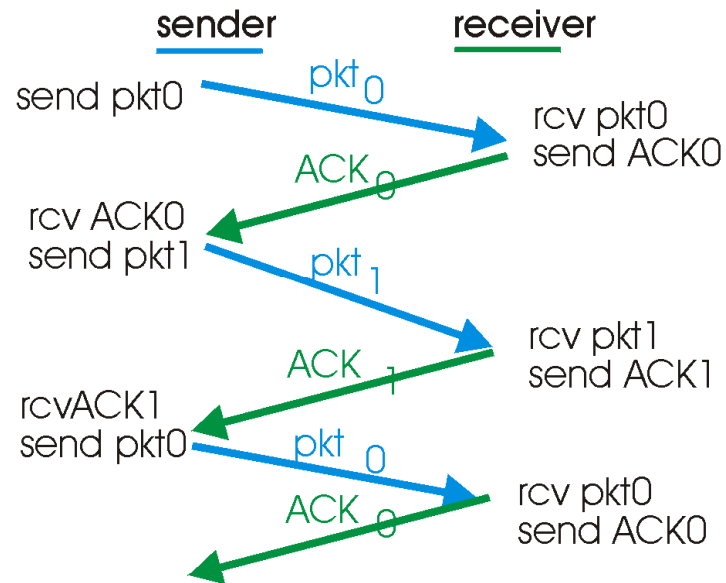
- 如果没收到ACK，重传
- 如果分组或ACK只是延迟而不是丢了
 - ⑩ 重传会产生重复，序列号机制能够处理
 - ⑩ 接收方需在ACK中显式告知所确认的分组
- 需要定时器



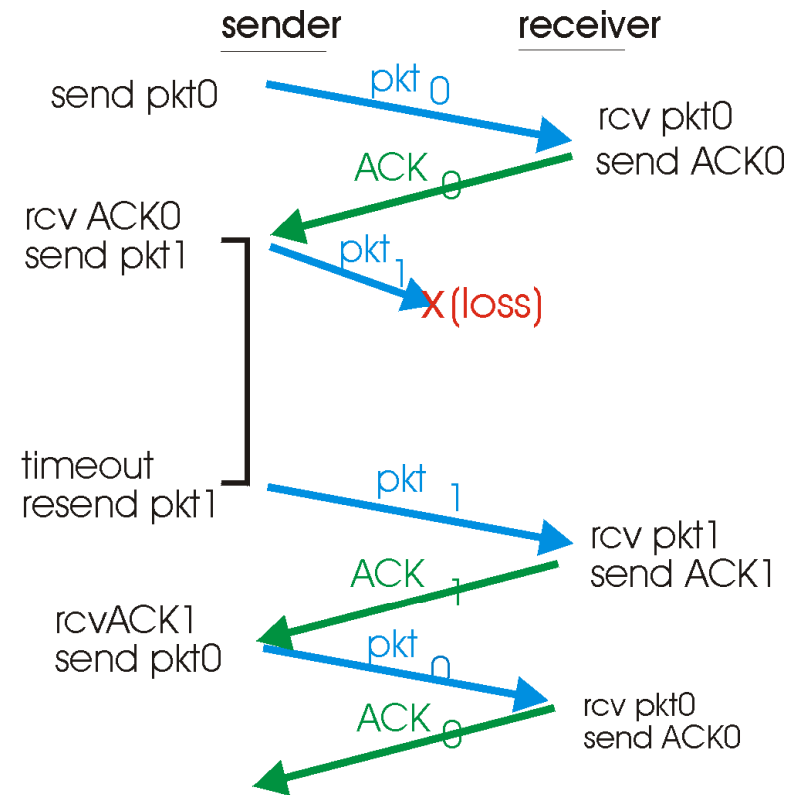
Rdt 3.0发送方FSM



Rdt 3.0示例(1)



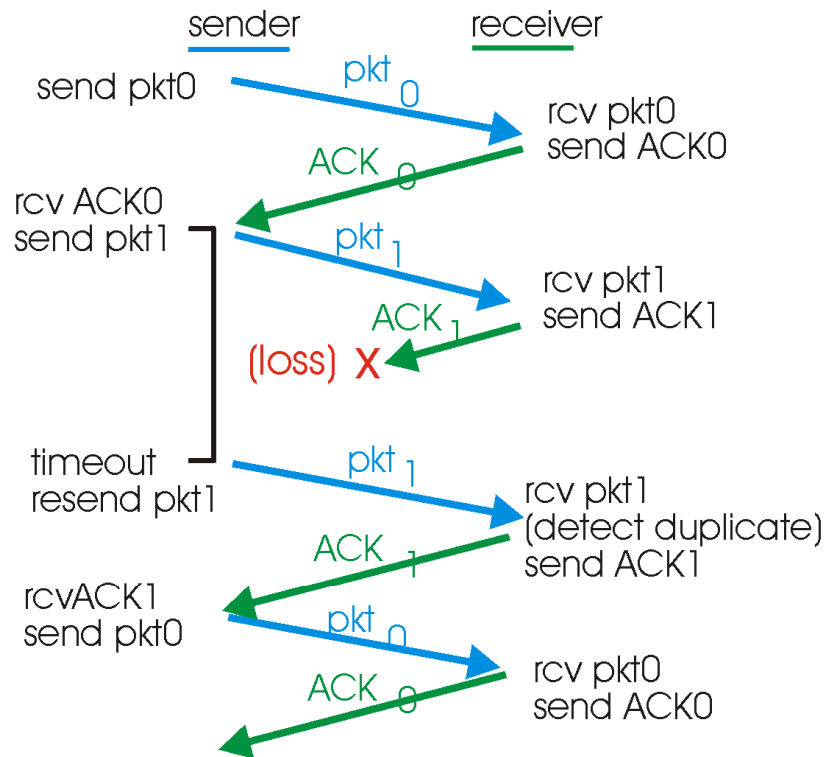
(a) operation with no loss



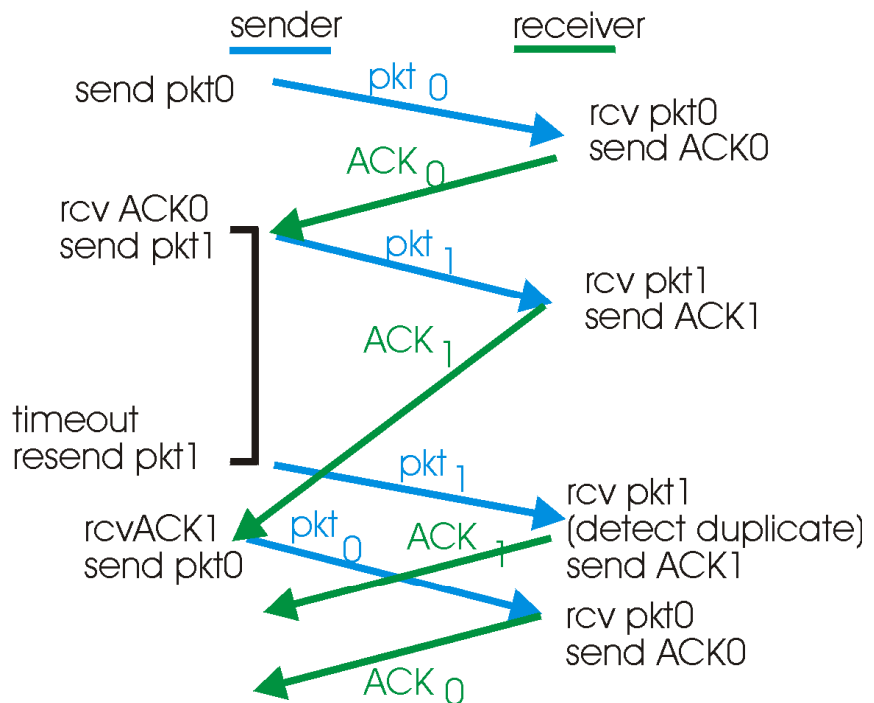
(b) lost packet



Rdt 3.0示例(2)



(c) lost ACK



(d) premature timeout



Rdt 3.0性能分析

- ❖ Rdt 3.0能够正确工作，但性能很差
- ❖ 示例：1Gbps链路，15ms端到端传播延迟，1KB分组

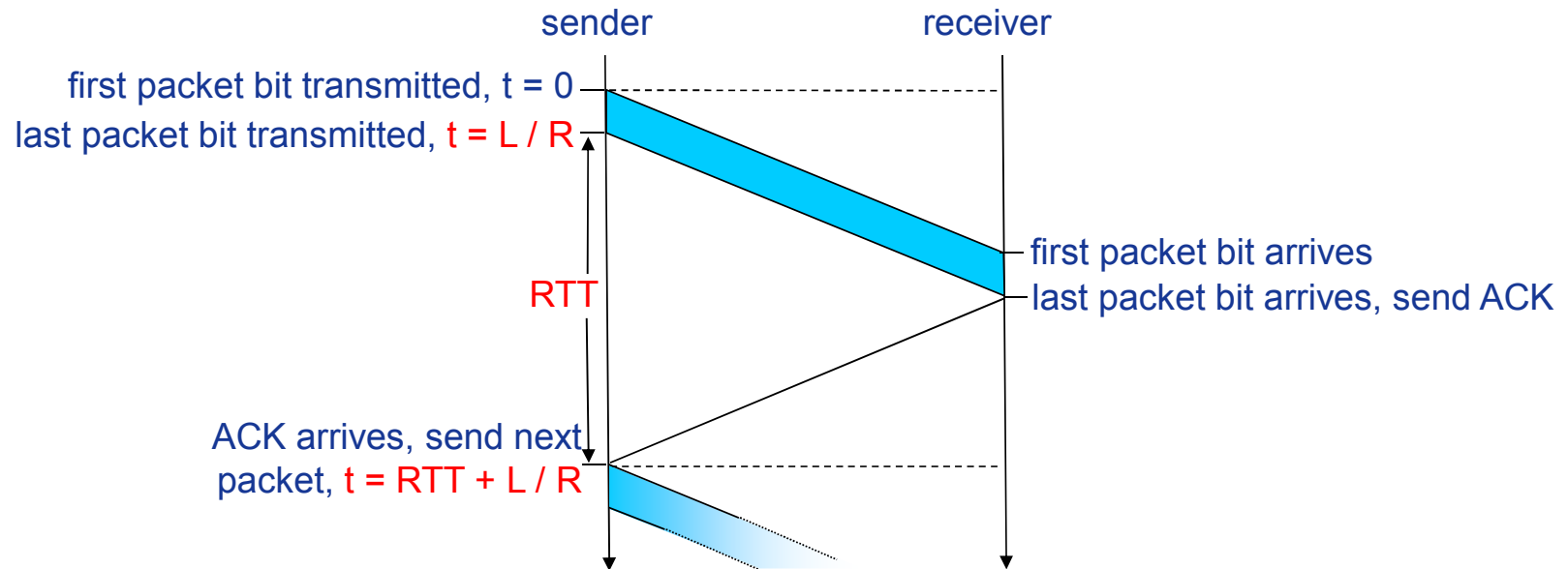
$$T_{\text{transmit}} = \frac{L \text{ (packet length in bits)}}{R \text{ (transmission rate, bps)}} = \frac{8\text{kb/pkt}}{10^9 \text{ b/sec}} = 8 \text{ microsec}$$

- 发送方利用率：发送方发送时间百分比

$$U_{\text{sender}} = \frac{L / R}{RTT + L / R} = \frac{.008}{30.008} = 0.00027$$

- 在1Gbps链路上每30毫秒才发送一个分组→33KB/sec
- 网络协议限制了物理资源的利用

Rdt 3.0: 停等操作



$$U_{\text{sender}} = \frac{L / R}{RTT + L / R} = \frac{.008}{30.008} = 0.00027$$





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



谢谢!



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



计算机网络之探赜索隐

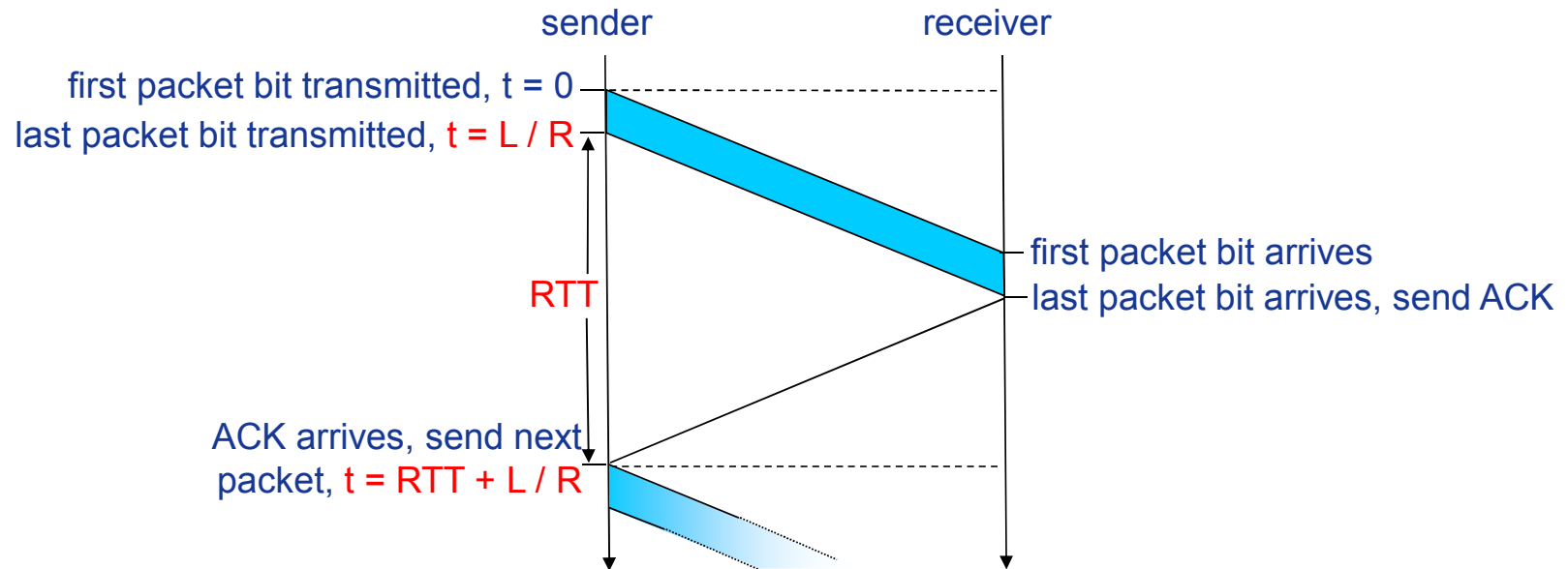
主讲人：聂兰顺

本讲主题

流水线机制与滑动窗口协议



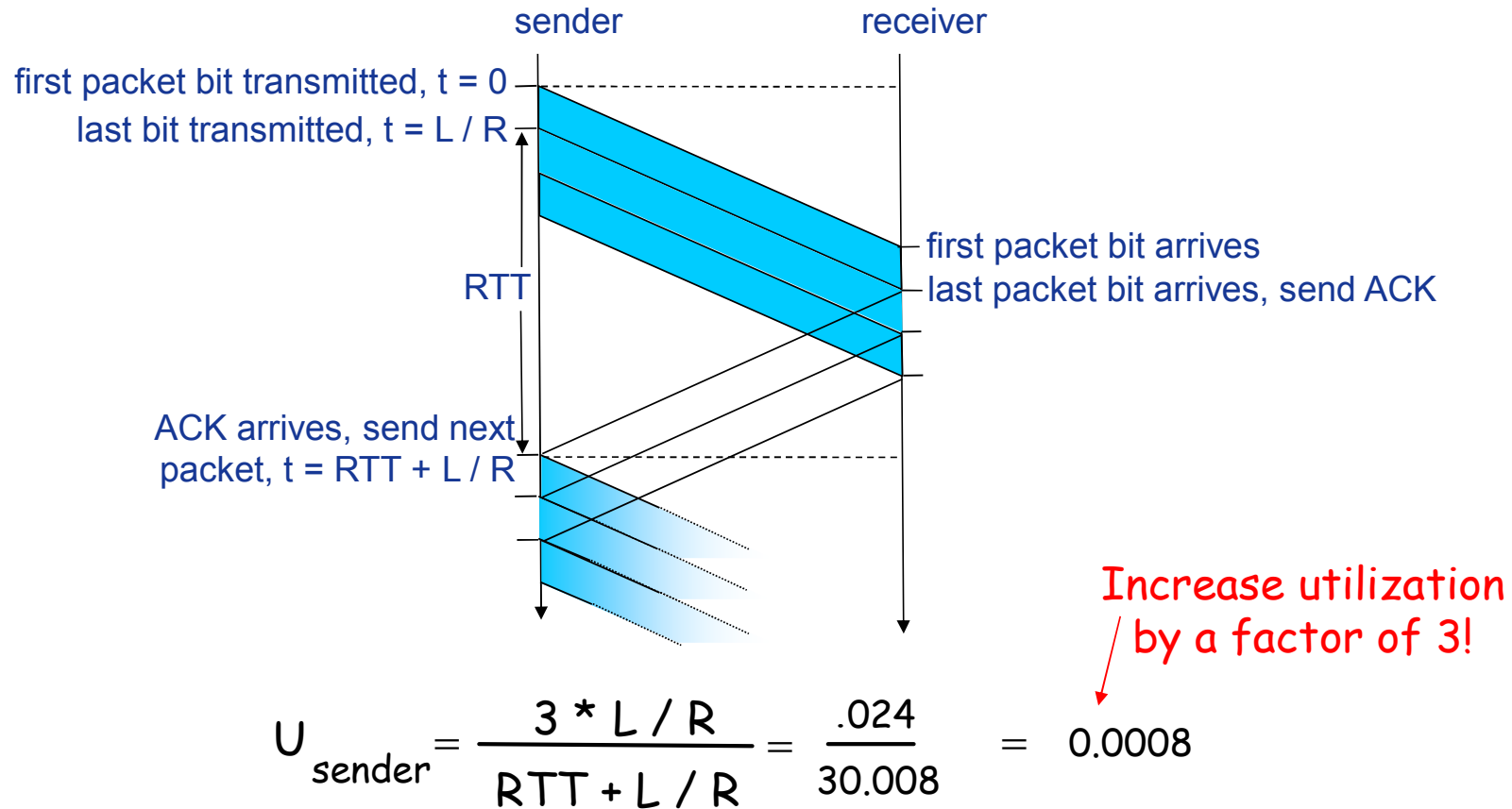
Rdt 3.0: 停等操作



$$U_{\text{sender}} = \frac{L / R}{RTT + L / R} = \frac{.008}{30.008} = 0.00027$$

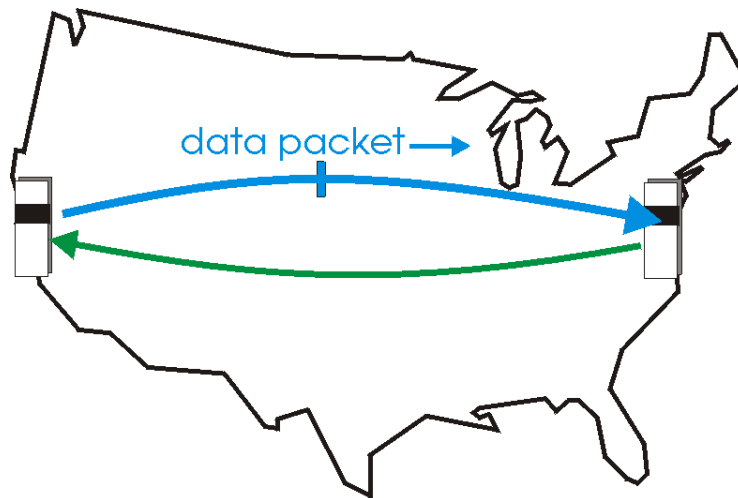


流水线机制：提高资源利用率

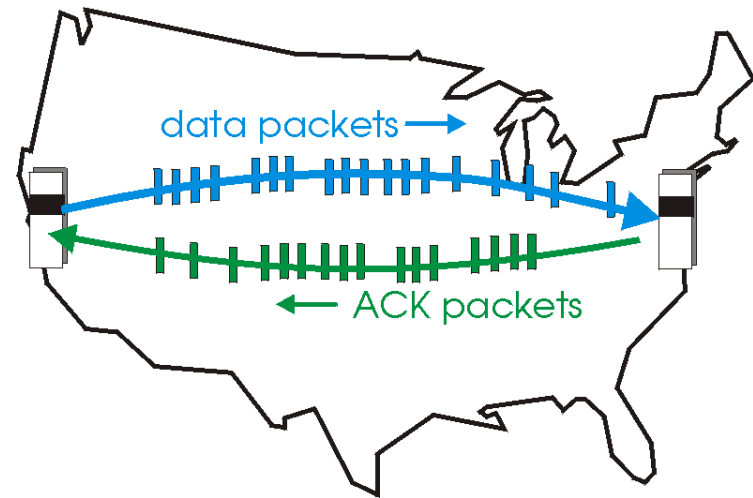


流水线协议

- ❖ 允许发送方在收到ACK之前连续发送多个分组
 - 更大的序列号范围
 - 发送方和/或接收方需要更大的存储空间以缓存分组



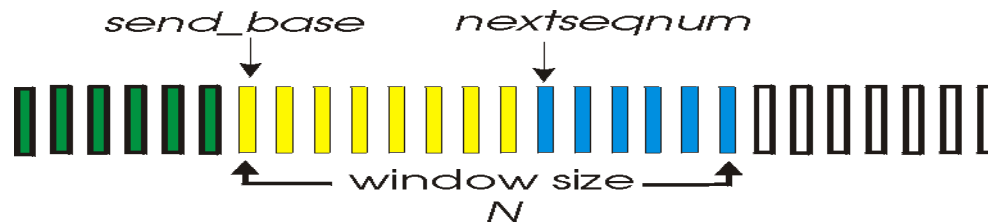
(a) a stop-and-wait protocol in operation



(b) a pipelined protocol in operation



滑动窗口协议



❖ 滑动窗口协议: Sliding-window protocol

❖ 窗口

- 允许使用的序列号范围
- 窗口尺寸为 N : 最多有 N 个等待确认的消息

❖ 滑动窗口

- 随着协议的运行, 窗口在序列号空间内向前滑动

❖ 滑动窗口协议: GBN, SR





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



谢谢!



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



计算机网络之探赜索隐

主讲人：聂兰顺

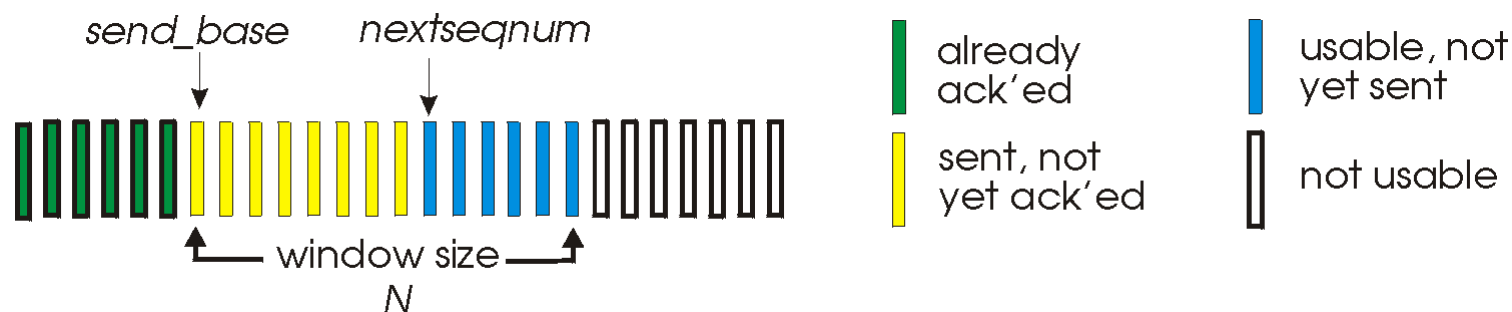
本讲主题

Go-Back-N协议



Go-Back-N(GBN)协议: 发送方

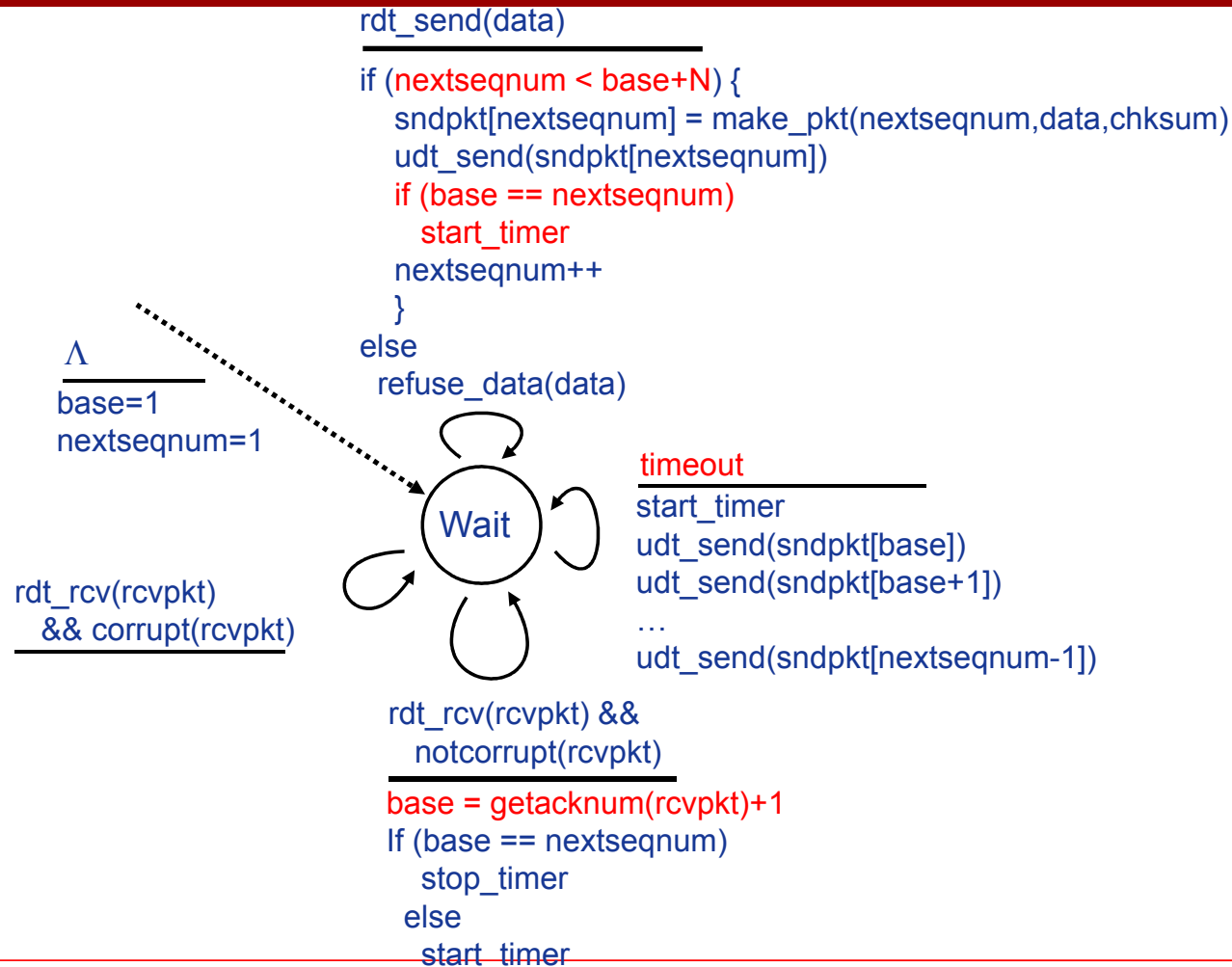
- ❖ 分组头部包含k-bit序列号
- ❖ 窗口尺寸为N，最多允许N个分组未确认



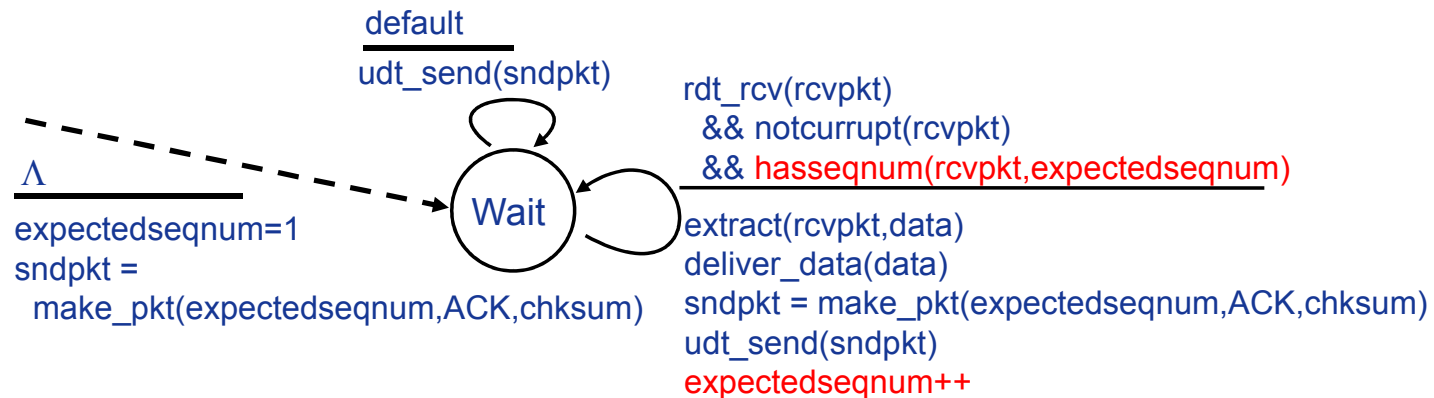
- ❖ ACK(n): 确认到序列号n(包含n)的分组均已被正确接收
 - 可能收到重复ACK
- ❖ 为空中的分组设置**计时器(timer)**
- ❖ 超时Timeout(n)事件: 重传序列号大于等于n, 还未收到ACK的所有分组



GBN: 发送方扩展FSM



GBN: 接收方扩展FSM



❖ ACK机制: 发送拥有最高序列号的、已被正确接收的分组的ACK

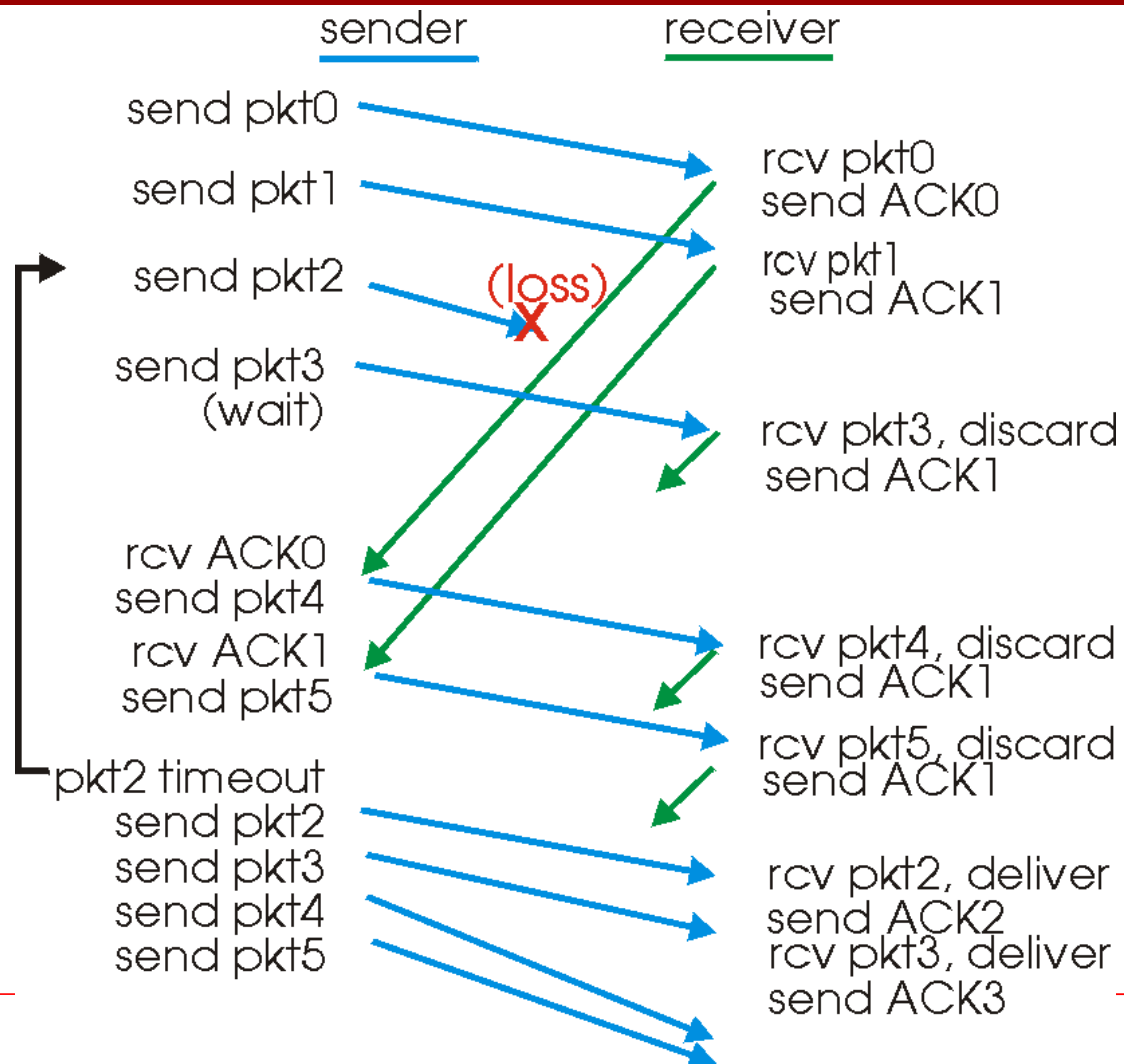
- 可能产生重复ACK
- 只需要记住唯一的**expectedseqnum**

❖ 乱序到达的分组:

- 直接丢弃→接收方没有缓存
- 重新确认序列号最大的、按序到达的分组



GBN示例



练习题

❑ 数据链路层采用后退N帧（GBN）协议，发送方已经发送了编号为0~7的帧。当计时器超时时，若发送方只收到0、2、3号帧的确认，则发送方需要重发的帧数是多少？分别是那几个帧？

❑ 解：根据GBN协议工作原理，GBN协议的确认是累积确认，所以此时发送端需要重发的帧数是4个，依次分别是4、5、6、7号帧。





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



谢谢!



哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



计算机网络之探赜索隐

主讲人：聂兰顺

本讲主题

Selective Repeat协议

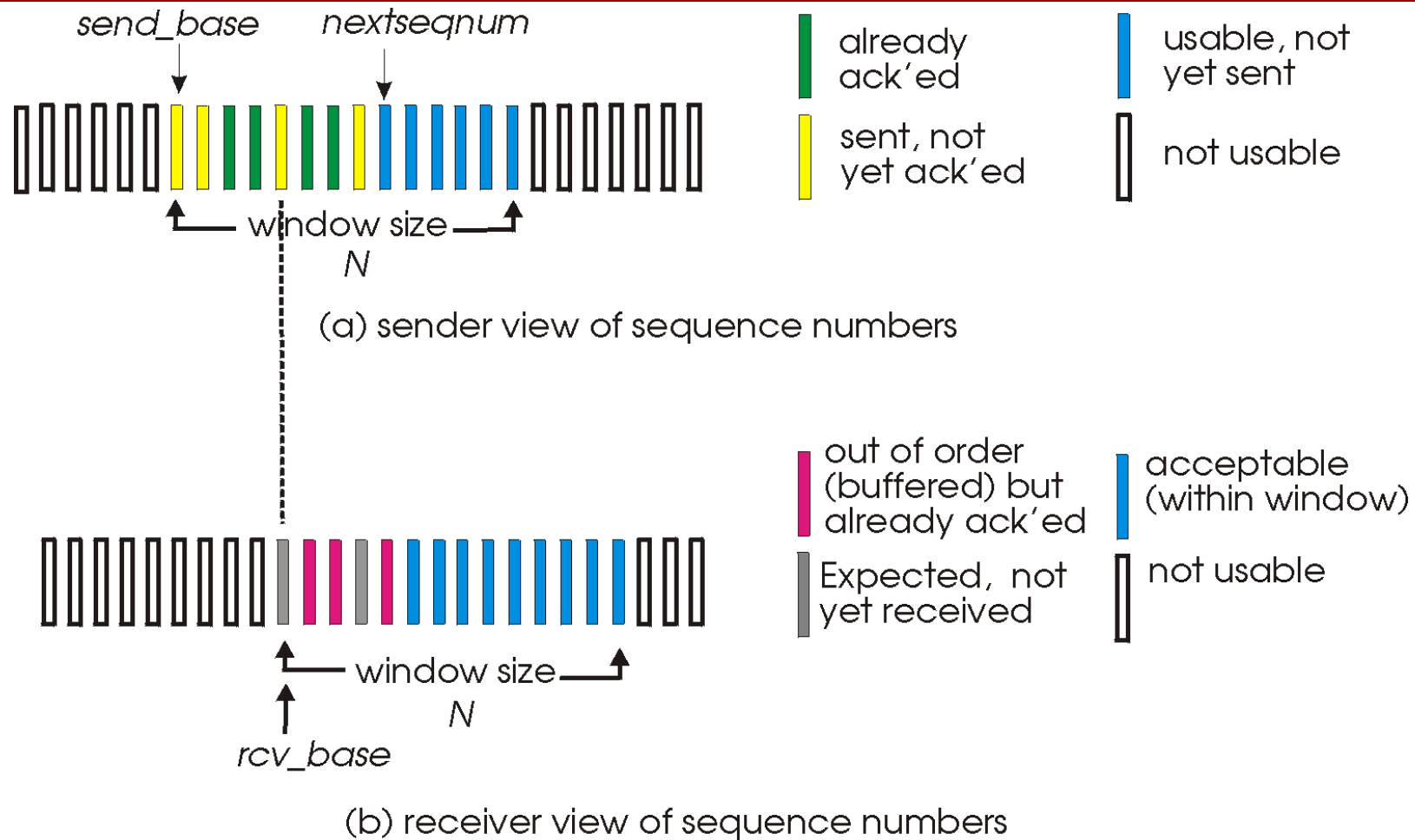


Selective Repeat协议

- ❖ **GBN有什么缺陷？**
- ❖ 接收方对每个分组单独进行确认
 - 设置缓存机制，缓存乱序到达的分组
- ❖ 发送方只重传那些没收到ACK的分组
 - 为每个分组设置定时器
- ❖ 发送方窗口
 - N个连续的序列号
 - 限制已发送且未确认的分组



Selective Repeat: 发送方/接收方窗口



SR协议

sender

data from above :

- ❖ if next available seq # in window, send pkt

timeout(n):

- ❖ resend pkt n, restart timer

ACK(n) in [sendbase, sendbase+N]:

- ❖ mark pkt n as received
- ❖ if n smallest unACKed pkt, advance window base to next unACKed seq #

receiver

pkt n in [rcvbase, rcvbase+N-1]

- ❑ send ACK(n)
- ❑ out-of-order: buffer
- ❑ in-order: deliver (also deliver buffered, in-order pkts), advance window to next not-yet-received pkt

pkt n in [rcvbase-N, rcvbase-1]

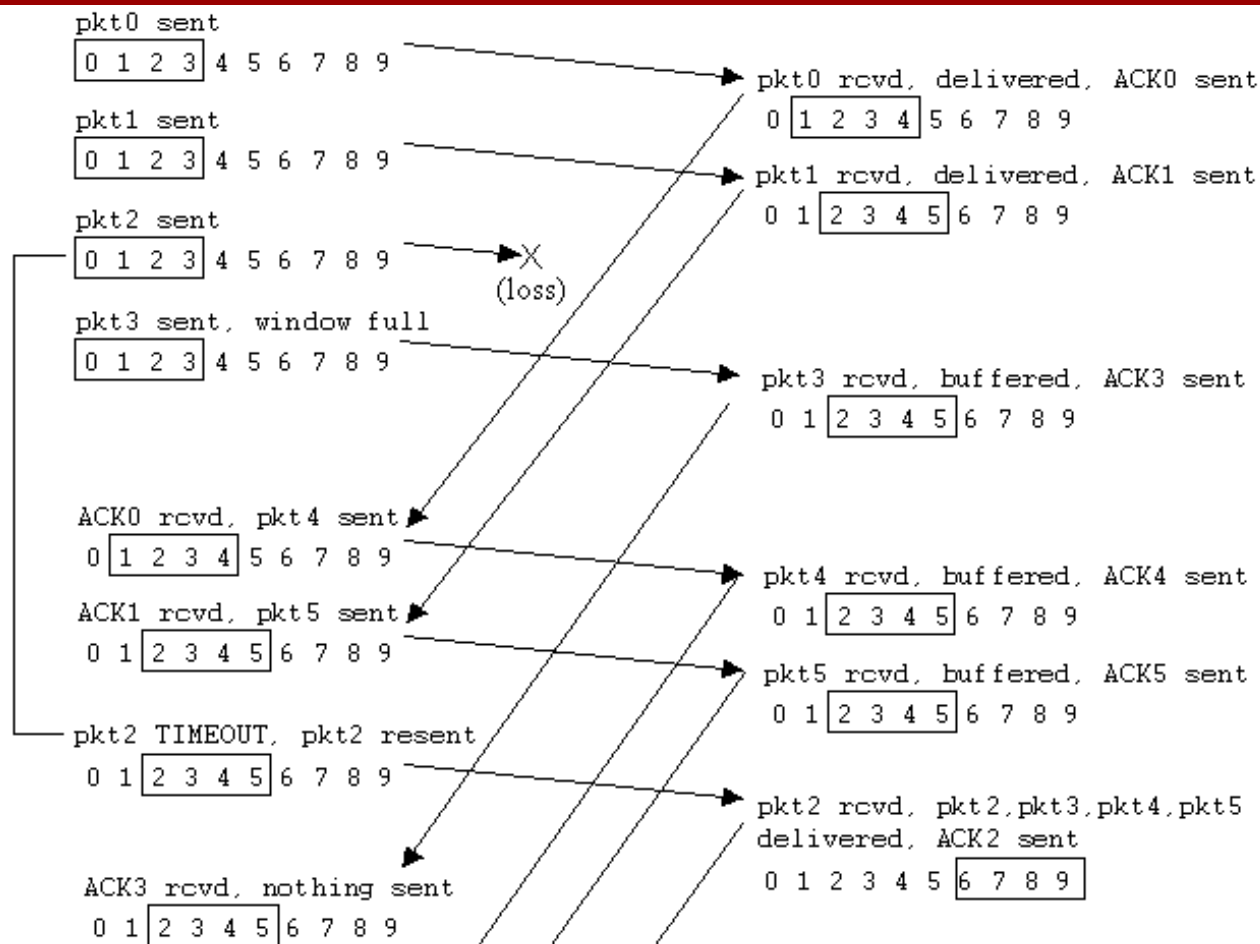
- ❑ ACK(n)

otherwise:

- ❑ ignore



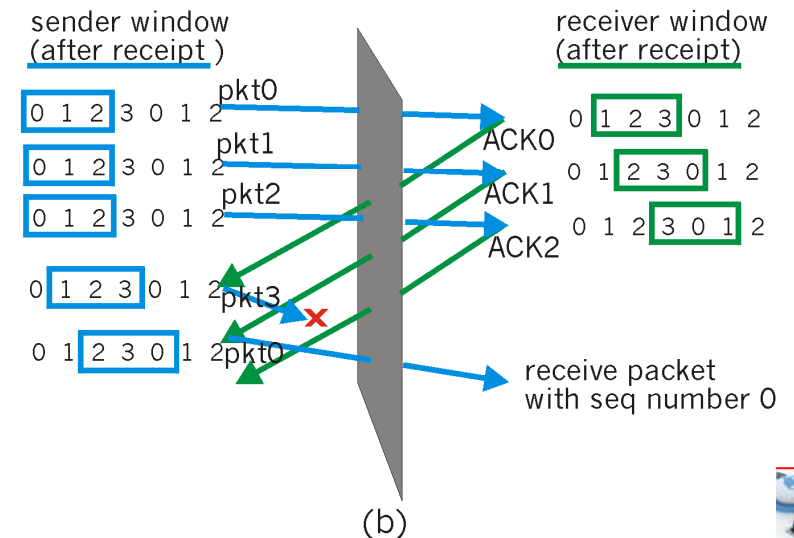
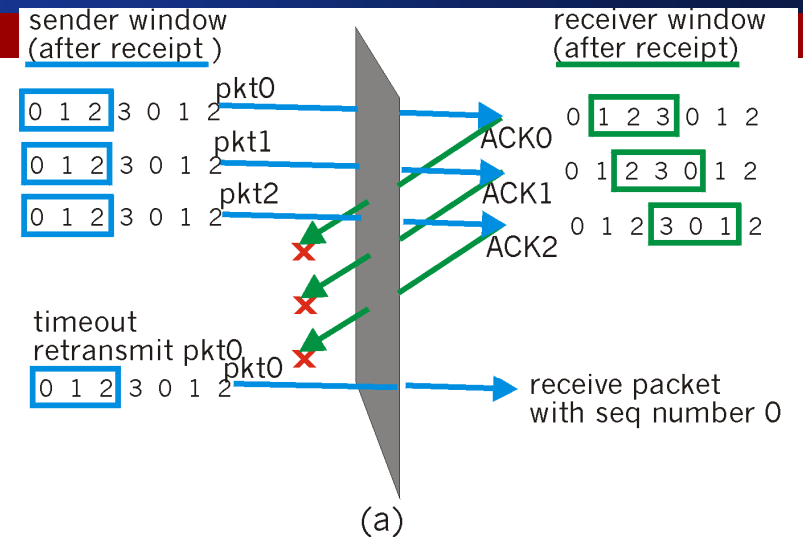
SR协议示例



SR协议：困境

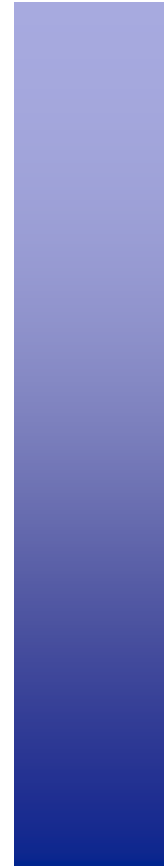
- ❖ 序列号: 0, 1, 2, 3
- ❖ 窗口尺寸: 3
- ❖ 接收方能区分开右侧两种不同的场景吗?
- ❖ (a)中, 发送方重发分组0, 接收方收到后会如何处理?
- ❖ 问题: 序列号空间大小与窗口尺寸需满足什么关系?

$$N_S + N_R \leq 2^k$$



可靠数据传输原理与协议回顾

- ❖ 信道的(不可靠)特性
- ❖ 可靠数据传输的需求
- ❖ Rdt 1.0
- ❖ Rdt 2.0, rdt 2.1, rdt 2.2
- ❖ Rdt 3.0
- ❖ 流水线与滑动窗口协议
- ❖ GBN
- ❖ SR





哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

立足航天，服务国防，面向国民经济主战场



谢谢!