**R3.考虑在主机A和主机B之间有一条TCP连接。假设从主机A传送到主机B的TCP报文段具有源端口号x和目的端口号y。对于从主机B传送到主机A的报文段，源端口号和目的端口号分别是多少？**

答：源端口号为y，目的端口号为x

**R7.假定在主机C上的一个进程有一个具有端口号6789的UDP套接字。假定主机A和主机B都用目的端口号6789向主机C发送一个UDP报文段。这两台主机的这些报文段在主机C都被描述为相同的套接字吗？如果是这样的话，在主机C的该进程将怎样知道源于两台不同主机的这两个报文段？**

答：会被描述为相同的套接字。通过源IP地址来进行区分。

**R14.是非判断题：**

**a.主机A经过一条TCP连接向主机B发送一个大文件。假设主机B没有数据发往主机A。因为主机B不能随数据捎带确认，所以主机B将不向主机A发送确认。**

**b.在连接的整个过程中，TCP的rwnd的长度决不会变化。**

**c.假设主机A通过一条TCP连接向主机B发送一个大文件。主机A发送但未被确认的字节数不会超过接收缓存的大小。**

**d.假设主机A通过一条TCP连接向主机B发送一个大文件。如果对于这条连接的一个报文段的序号为m，则对于后继报文段的序号将必然是m + 1。**

**e. TCP报文段在它的首部中有一个rwnd字段。**

**f.假定在一条TCP连接中最后的SampleRTT等于1秒，那么对于该连接的Timeoutlnterval的当前值必定大于等于1秒。**

**g.假设主机A通过一条TCP连接向主机B发送一个序号为38的4个字节的报文段。在这个相同的报文段中，确认号必定是42。**

答: a.错

b.错

c.对

d.错

e.对

f.错

g.错

**R15.假设主机A通过一条TCP连接向主机B发送两个紧接着的TCP报文段。第一个报文段的序号为90， 第二个报文段序号为110。**

**a.第一个报文段中有多少数据？**

**b.假设第一个报文段丢失而第二个报文段到达主机B。那么在主机B发往主机A的确认报文中, 确认号应该是多少？**

答：a.20字节

b.确认号为90

**P1.假设客户A向服务器S发起一个Telnet会话。与此同时，客户B也向服务器S发起一个Telnet会话。 给出下面报文段的源端口号和目的端口号：**

**a.从A向S发送的报文段。**

**b.从B向S发送的报文段。**

**c.从S向A发送的报文段。**

**d.从S向B发送的报文段。**

**e.如果A和B是不同的主机，那么从A向S发送的报文段的源端口号是否可能与从B向S发送的报文段的源端口号相同？**

**f.如果它们是同一台主机，情况会怎么样？**

答: a. A到S，源端口号467，目的端口号23

b. B到S，源端口号513，目的端口号23

c. S到A，源端口号23，目的端口号467

d. S到B，源端口号23，目的端口号513

e.可能相同

f.不可能相同

**P2.考虑图3-5。从服务器返回客户进程的报文流中的源端口号和目的端口号是多少？在承载运输层报文段的网络层数据报中，IP地址是多少？**

答：对于主机A，源端口号为80，源IP为B，目的端口号为26145，目的IP为A。

对于主机C左侧进程，源端口号为80，源IP为B，目的端口号为7532，目的IP为C。

对于主机C右侧进程，源端口号为80，源IP为B，目的端口号为26145，目的IP为C。

**P4.a.假定你有下列2个字节：01011100和01100101。这2个字节之和的反码是什么？**

**b.假定你有下列2个字节：11011010和01100101。这2个字节之和的反码是什么？**

**c.对于a中的字节，给出一个例子，使得这2个字节中的每一个都在一个比特反转时，其反码不会改变。**

答：a.00111110

b.10111111

c.第一个字节变为01010100，第二个字节变为01101101

**P7.在rdt3.0协议中，从接收方向发送方流动的ACK分组没有序号（尽管它们具有ACK字段，该字段包括了它们正在确认的分组的序号）。为什么这些ACK分组不需要序号呢？**

答：为了回答这个问题，首先考虑为什么我们需要序列号。我们看到发送方需要序列号以便于接收方能够区分出一个分组是不是己经接收到的分组的重复。考虑ACK信息，发送方不需要这个信息（也就是一个ACK的序列号）来告诉发送方检测到一个重复的ACK.因为当他接收到原始ACK信息后它就转入下一个状态，所以一个重复的ACK信息对rdt3.0的发送方是很明显的。重复的ACK信息不是发送方需要的ACK信息，因此被rdt3.0发送方忽略了。

**P11.考虑在图3-14中的rdt2. 2接收方，在状态“等待来自下层的0”和状态“等待来自下层的1”中的自转换（即从某状态转换回自身）中生成一个新分组：sndpk = make\_ pkt ( ACK, 1，checksum)和 sndpk = make\_ pkt ( ACK, 0，checksum) 0如果这个动作从状态“等待来自下层的1”中的自转换中删除，该协议将正确工作吗？评估你的答案。在状态“等待来自下层的0”中的自转换中删除这个事件将会怎样？[提示：在后一种情况下，考虑如果第一个发送方到接收方的分组损坏的话，将会发生什么情况？**

答：会发生死锁。

发送方发送 pkt0, 进入“等待ACK0状态”, 等待一个接收方发回的确认包。

接收方在“等待来自下层的0”的状态，接收一个损坏的包，如果它不返回信息，就待在“等待来自下层的0”的状态。

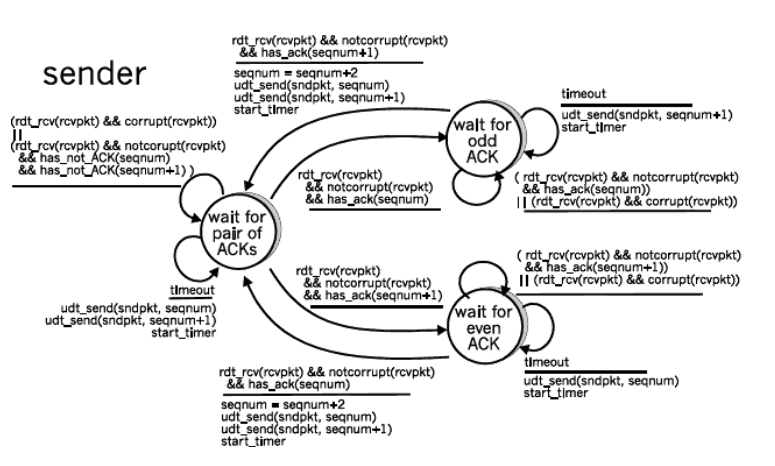
现在，发送方在等待来自接收方的确认，接收方在等待发送方的数据包，这样就会产生死锁。

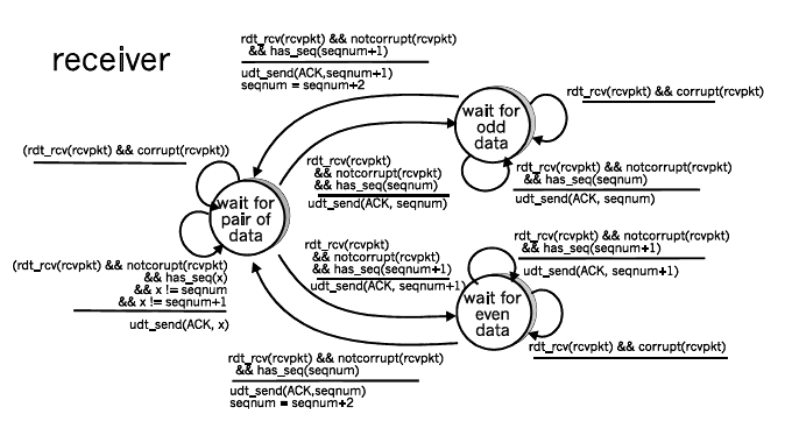
**P14.考虑一种仅使用否定确认的可靠数据传输协议。假定发送方只是偶尔发送数据。只用NAK的协议是否会比使用ACK的协议更好？为什么？现在我们假设发送方要发送大量的数据，并且该端到端连接很少丟包。在第二种情况下，只用NAK的协议是否会比使用ACK的协议更好？为什么？**

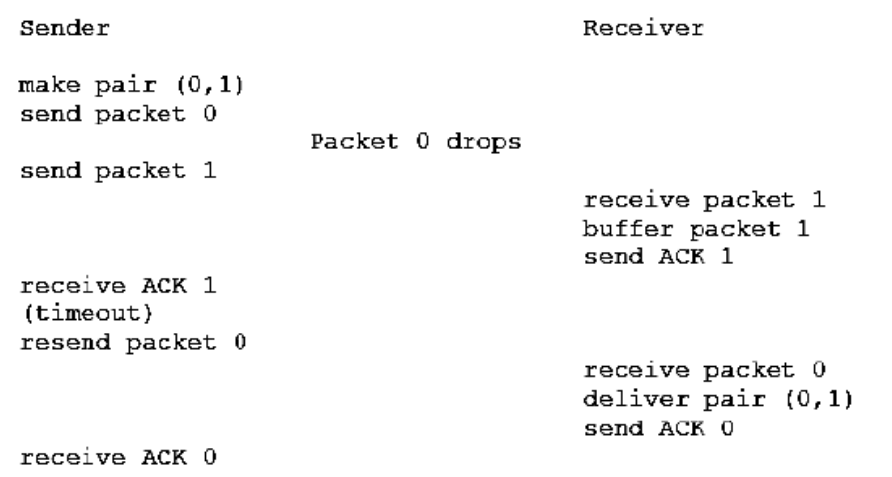
答：在仅使用NAK的协议中，只有当接收到分组x+1时才能检测到分组x的丢失.也就是说接收方接收到x-1然后接 收到x+1,只有当接收方接收到x+1时才发现x的丢失.如果在传输x和传输x+1之间有很长时间的延时，那么在只有NAK的协议中，x的修复要花费很长的时间.另一方面，如果要发送大量的数据，那么在只有NAK的协议中修复的速度将很快.并且，如果错误很少，那么NAK只是偶尔发送,并且从不发送ACK.与只有ACK的情况相比，只有 NAK的情况将明显减少反馈时间.

**P20. 考虑一种主机A和主机B要向主机C发送报文的情况。主机A和C通过一条报文能够丢失和损坏（但不重排序）的信道相连接。主机B和C由另一条（与连接A和C的信道独立）具有相同性质的信道连接。 在主机C上的运输层，在向上层交付来自主机A和B的报文时应当交替进行（即它应当首先交付来自A 的分组中的数据，然后是来自B的分组中的数据，等等)。设计一个类似于停等协议的差错控制协议，以可靠地向C传输来自A和B的分组，同时以前面描述的方式在C处交替地交付。给出A和C的FSM描述。（提示：B的FSM大体上应当与A的相同。）同时，给出所使用的报文格式的描述。**

答：在我们的解决方案中,发送方在接收到一对报文的ACK(seqnum和seqnum+1)后才开始发送下一对报文，数据分组携带有两bit的序列码，也就是说，有效的序列号是0，1，2，3。ACK信息携带己经确认的数据分组的序列号。接收方和发送方的FSM由下图所示.注意发送状态记录(1)当前对没有收到ACKs,(2)只收到seqnum的ACK或只收到 seqnum+1的ACK。在本图中,我们假设seqnum由0起始,发送方己经发送第一对数据.







**P24. 对下面的问题判断是非，并简要地证实你的回答；**

**a对于SR协议，发送方可能会收到落在其当前窗口之外的分组的ACK。**

**b.对于GBN协议，发送方可能会收到落在其当前窗口之外的分组的ACK。**

**c.当发送方和接收方窗口长度都为I时，比特交替协议与SR协议相同。**

**d.当发送方和接收方窗口长度都为1时，比特交替协议与GBN协议相同。**

答：a)正确，假设发送方窗门大小为3，在时刻发送分组1,2,3.在 (>)时刻接收方确认1,2,3.在 (>)时刻发送 方计时器超时，重发1,2,3.在时刻接收到重复的分组并重新确认1,2,3.在时刻发送方接收到接收方在时刻发送的ACK，并将其窗口前移到4,5,6在时刻发送方接收到接收方在发送的ACK 1，2，3这些ACK是在当前窗口之外的报文的ACK

b)是的，本质上同a中是一样的.

c)正确

d)正确.当窗口尺寸为1时，SR.GBN，的比特交替协议在功能上相同。窗口尺寸1排除了失序分组的可能性，在这种情况下，一个累积的ACK就是一个普通的ACK.因为在窗口内它只能与一个分组有关.

**P27. 主机A和B经一条TCP连接通信，并且主机B已经收到了来自A的最长为126字节的所有字节。 假定主机A随后向主机B发送两个紧接着的报文段。第一个和第二个报文段分别包含了 80字节和 40字节的数据。在第一个报文段中，序号是127,源端口号是302，目的地端口号是80。无论何时 主机B接收到来自主机A的报文段，它都会发送确认。**

**a.在从主机A发往B的第二个报文段中，序号、源端口号和目的端口号各是什么？**

**b.如果第一个报文段在第二个报文段之前到达，在第一个到达报文段的确认中，确认号、源端口号 和目的端口号各是什么？**

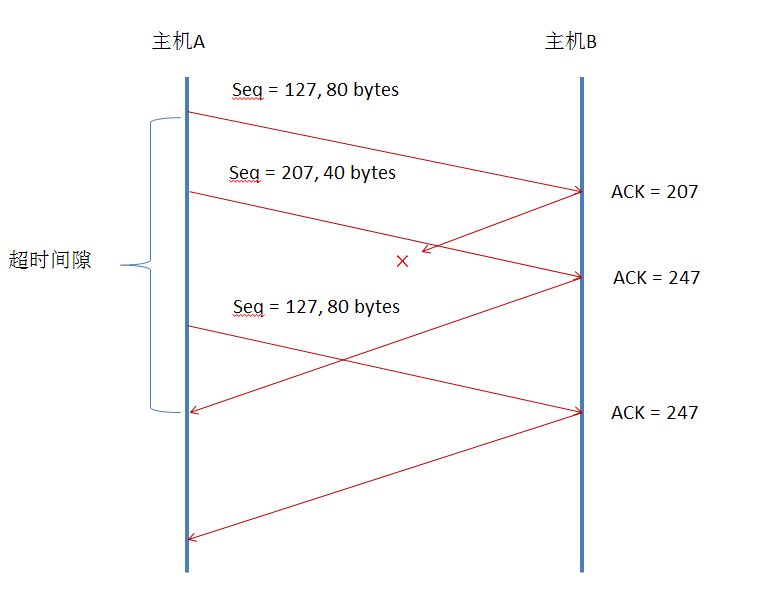
**c.如果第二个报文段在第一个报文段之前到达，在第一个到达报文段的确认中，确认号是什么？**

**d.假定由A发送的两个报文段按序到达B。第一个确认丟失了而第二个确认在第一个超时间隔之后到达。画出时序图，显示这些报文段和发送的所有其他报文段和确认。（假设没有其他分组丢失。）对于图上每个报文段，标出序号和数据的字节数量；对于你增加的每个应答，标出确认号。**

答：a.序号207，源端口号302，目的端口号80.

b.确认号207，源端口号80，目的端口号302.

c.确认号127.

d.

**P28. 主机A和B直接经一条100Mbps链路连接。在这两台主机之间有一条TCP连接。主机A经这条连接向主机B发送一个大文件。主机A能够向它的TCP套接字以高达120Mbps的速率发送应用数据, 而主机B能够以最大50Mbps的速率从它的TCP接收缓存中读出数据。描述TCP流量控制的影响。**

答：主机A向接收缓存中发送数据的速度比主机B从缓存中读出数据的速度快。接收缓存大致以50Mbps的速度填充。当缓冲区满的时候，主机B通过将RcvWindow设置为0来向主机A发送停止传输数据的信号。于是主机A停止传输数据直到接收到一个带有RcvWindow > 0的TCP报文段。主机A就这样根据RcvWindow 的值反复地停止和开始传输数据。平均来说，主机A向主机B发送数据的速率不会超过50Mbps。

**P31.假设测量的5个SampleRTT值（参见3.5.3节）是106ms、120ms 、140ms, 90ms和115ms。在获得了每个SampleRTT值后计算EstimatedRTT，使用= 0. 125并且假设在刚获得前5个样本之后EstimatedRTT的值为100ms。在获得每个样本之后，也计算DevRTT，假设=0.25，并且假设在刚获得前5个样本之后DevRTT的值为5ms。最后，在获得这些样本之后计算TCP TimeoutInterval。**

**P32.考虑TCP估计RTT的过程。假设a =0.1，令置为最新样本RTT，令设置为 下一个最新样本RTT，等等。**

**a.对于一个给定的TCP连接，假定4个确认报文相继到达,带有4个对应的RTT值：、 、、。根据这 4 个样本 RTT 表示。**

**b.将你得到的公式一般化到n个RTT样本的情况。**

**c.对于在(b)中得到的公式，令n趋于无穷。试说明为什么这个平均过程被称为指数移动平均。**

答:(a)将定义为在n个sample之后的估计值.

= + (1 - )[ + (1 - )[ + (1 - )[]]

= + (1 - ) + +

= + 0.09 + +

(b) =

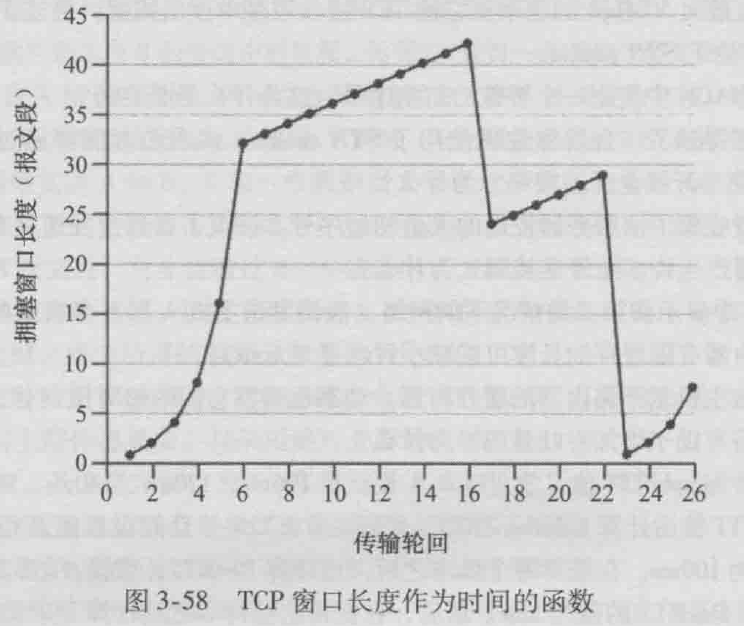
+

(c) =

=

之前的sample的权重是指数减少的.

**P40.考虑图3-58，假设TCP Reno是一个经历如上所示行为的协议，回答下列问题。在各种情况中，简要地论证你的回答。**



**a.指出TCP慢启动运行时的时间间隔。**

**b.指出TCP拥塞避免运行时的时间间隔。**

**c.在第16个传输轮回之后，报文段的丢失是根据3个冗余ACK还是根据超时检测出来的？**

**d.在第22个传输轮回之后，报文段的丢失是根据3个冗余ACK还是根据超时检测出来的？**

**e.在第1个传输轮回里，ssthresh的初始值设置为多少？**

**f.在第18个传输轮回里，ssthresh的值设置为多少？**

**g.在第24个传输轮回里，ssthresh的值设置为多少？**

**h.在哪个传输轮回内发送第70个报文段？**

**i.假定在第26个传输轮回后，通过收到3个冗余ACK检测出有分组丢失，拥塞的窗口长度和 ssthresh的值应当是多少？**

**j.假定使用TCP Tahoe (而不是TCP Reno),并假定在第16个传输轮回收到3个冗余ACK。在第19 个传输轮回，ssthresh和拥塞窗口长度是什么？**

**k.再次假设使用TCP Tahoe,在第22个传输轮回有一个超时事件。从第17个传输轮回到第22个传输轮回（包括这两个传输轮回）,一共发送了多少分组？**

答：a)运行TCP慢启动的时间间隔是[1,6]和[23,26]

b)运行TCP避免拥塞时的时间间隔是[1,6]和[17,22]

c)在第16个传输周期后，通过3个冗余ACK能够检测到一个报文段丢失。如果有一个超时，拥塞窗口尺寸将减小为1。

d)在第22个传输周期后，因为超时能够检测到一个报文段丢失，因此拥塞窗口的尺寸被设罝为1。

e)Threshold的初始值设置为32，因为在这个窗口尺寸是慢启动停止，避免拥塞开始。

f)当检测到报文段丢失时，threshold被设罝为拥塞窗口值的一半。当在第16个周期检测到丢失时，拥塞窗口的大小是42，因此在第18个传输周期时threshoki值为21。

g)当检测到报文段丢失时，threshold被设罝为拥塞窗门值的一半。当在第22个周期检测到丢失时，拥塞窗口的大小是29，因此在第24个传输周期时threshold值为14。

h)在第一个传输周期内，报文段1被传送：在第二个传输周期发送报文段2—3:在第3个传输周期发送报文段4—7，在第四个传输周期发送8—15:在第五个传输周期发送16—31;在第六个传输周期发送32—63;在 第7个传输周期发送64—96。因此，报文段70在第7个传输周期内发送。

i)当丢失出现时ssthresh的值被设罝为当前拥塞窗口长度8的一半，拥塞窗口的长度还要再加3。因此ssthresh的值为4，拥塞窗口长度为7。

j) threshold是21, 拥塞窗口长度为1。

k) 轮回17, 1个分组; 轮回18, 2个分组; 轮回19, 4个分组; 轮回20, 8个分组; 轮回21, 16个分组; 轮回22, 21个分组. 所以一共发送了52个分组。