在第二种情况下，如图3-27b所示，对前3个分组的ACK都被正确交付。因此发送方向前移动窗口并发送第4、5、6个分组，其序号分别为3、0、1。序号为3的分组丢失，但序号为0的分组到达（一个包含新数据的分组）。

现在考虑一下图3-27中接收方的观点，在发送方和接收方之间有一副假想的帘子，因为接收方不能“看见”发送方采取的动作。接收方所能观察到的是它从信道中收到的以及它向信道中发出报文序列。就其所关注的而言，图3-27中的两种情况是等同的。没有办法区分是第一个分组的重传还是第5个分组的初次传输。显然，窗口长度比序号空间小1时协议无法工作。但窗口必须多小呢？本章后面的一道习题请你说明为何对于SR协议而言，窗口长度比须小于或等于序号空间大小的一半。

在本书配套的网站上，可以找到一个模仿SR协议运行的Java小程序。尝试进行你以前对GBN Java小程序所进行的相同的实验。这些结果与你期望的一致吗？

至此我们结束了对可靠数据传输协议的讨论。我们已涵盖许多基础知识，并介绍了多种机制，这些机制可一起提供可靠数据传输。表3-1总结这些机制。既然我们已经学习了所有这些运行中的机制，并能看到“全景”，我们建议你再复习一遍本节内容，看看这些机制是怎样逐步被添加进来，以涵盖复杂性渐增的（现实的）连接发送方与接收方的各种信道模型的，或者如何改善协议性能的。

**表3-1 可靠数据传输机制及其用途的总结**

| 机制 | 用途和说明 |
| --- | --- |
| 检验和 | 用于检测在一个传输分组中的比特错误 |
| 定时器 | 用于超时/重传一个分组，可能因为该分组（或其ACK）在信道中丢失了。由于当一个分组延时但未丢失（过早超时），或当一个分组已被接收方收到但从接收方到发送方的ACK丢失时，可能产生超时事件，所以接收方可能会收到一个分组的多个冗余副本 |
| 序号 | 用于为从发送方流向接收方的数据分组按顺序编号。所接收分组的序号间的空隙可使接收方检测出丢失的分组。具有相同序号的分组可使接收方检测出一个分组的冗余副本 |
| 确认 | 接收方用于告诉发送方一个分组或一组分组已被正确地接收到了。确认报文通常携带着被确认的分组或多个分组的序号。确认可以是逐个的或累积的，这取决于协议 |
| 否定确认 | 接收方用于告诉发送方某个分组未被正确地接收。否定确认报文通常携带着未被正确接收的分组的序号 |
| 窗口、流水线 | 发送方也许被限制仅发送那些序号落在一个指定范围内的分组。通过允许一次发送多个分组但未被确认，发送方的利用率可在停等操作模式的基础上得到增加。我们很快将会看到，窗口长度可根据接收方接收和缓存报文的能力、网络中的拥塞程度或两者情况来进行设置 |

我们通过考虑在底层信道模型中的一个遗留假设来结束对可靠数据传输协议的讨论。前面讲过，我们曾假定分组在发送方与接收方之间的信道中不能被重新排序。这在发送方与接收方由单段物理线路相连的情况下，通常是一个合理的假设。然而，当连接两端的“信道”是一个网络时，分组重新排序是可能会发生的。分组重新排序的一个表现就是，一个具有序号或确认号 $x$ 的分组的旧副本可能会出现，即使发送方或接收方的窗口中都没有包含 $x$。对于分组重新排序，信道可被看成基本上是在缓存分组，并在将来任意时刻自然地释放出这些分组。由于序号可以被重新使用，那么必须小心，以免出现这样的冗余分组。实际应用中采用的方法是，确保一个序号不被重新使用，直到发送方“确信”任何先前发送的序号为 $x$ 的分组都不再在网络中为止。通过假定一个分组在网络中的“存活”时间不会超过某个固定最大时间量来做到这一点。在高速网络的TCP扩展中，最长的分组寿