最终会变满，这将反过来引起对服务器的“反向压力”。具体而言，一旦客户应用缓存变满，字节不再从客户 TCP 接收缓存中删除，因此它也会变满。一旦客户 TCP 接收缓存变满，字节不再从服务器 TCP 发送缓存删除，因此它也变满。一旦客户 TCP 发送缓存变满，服务器不能向套接字中发送任何更多的字节。因此，如果用户暂停视频，服务器可能被迫停止传输，在这种情况下服务器被阻塞，直到用户恢复该视频。

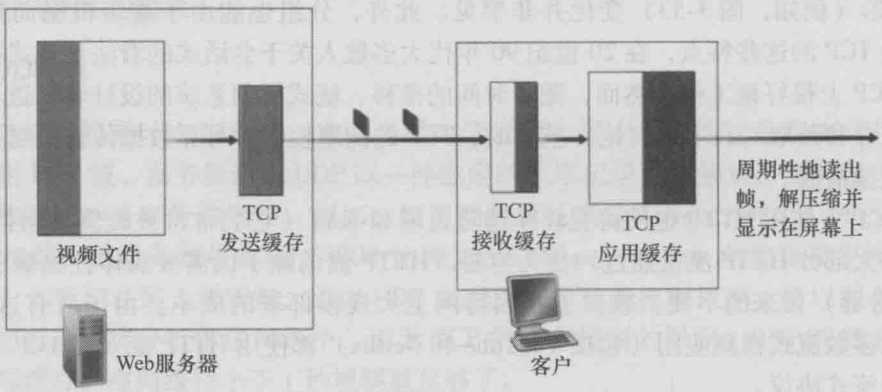

图 7-2 经 HTTP/TCP 的流式存储视频

事实上，甚至在常规的播放过程中（即没有暂停），如果客户应用缓存变满，反向压力将引起 TCP 缓存变满，这将迫使服务器降低其速率。为了决定其产生的速率，注意到当客户缓存删除 f 比特，它在客户应用缓存中产生了 f 比特的空间，这依次允许服务器发送额外的 f 比特。因此，服务器发送速率不能比客户端视频消耗速率更高。因此，当使用 HTTP 流时，一个满的客户应用缓存间接地对服务器到客户能够发送的视频速率施加了限制。

3. 流式视频的分析

某些简单的建模将有助于洞察由于应用缓存消耗所产生的初始播放时延和停滞。如图 7-3所示，B 表示客户应用缓存的长度（以比特计），Q 表示在客户应用缓存开始播放之前必须被缓存的比特数量。（当然，$Q < B$。）r 表示视频消耗速率，即客户在播放期间从客户应用缓存提取比特的速率。在此情况下，举例来说，如果视频的帧速率是 30 帧/秒，每（压缩）帧是 100 000 比特，则 $r = 3$Mbps。为了从细节看整体，我们将忽略 TCP 的发送和接收缓存。

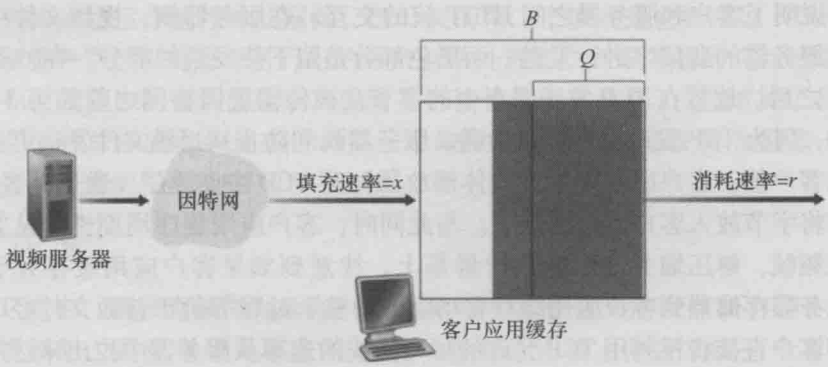

图 7-3 用户流式视频的客户端缓存的分析