# 17.排序器库

重新排序库提供了一种根据序列号重新排序mbufs的机制。

## 17.1.操作

重新排序库本质上是一个重新排列mbufs的缓冲区。用户将乱序的mbufs插入到重新排序缓冲区中，并从中输出顺序mbufs。

在给定的时间，重新排序缓冲区包含序列号在序列窗口内的mbufs。顺序窗口由缓冲区配置的可以维护的最小序列号和条目数决定。例如，给定具有200个条目并且最小序列号为350的重排序缓冲器，序列窗口分别具有350和550的低和高限制。

当插入mbufs时，重新排序库根据插入的mbuf的序列号区分valid，late和early的mbuf：

* Valid：序列号在有序窗口的限制内。
* Late：序列号在窗口限制外，小于下限。
* Early：序列号在窗口限制外，大于上限。

重新排序缓冲区直接返回late mbufs，并尝试适应early mbufs。

## 17.2.实现细节

重新排序库被实现为一对缓冲区，称为Order buffer和Ready buffer。

在插入调用时，valid mbufs将直接插入到Order buffer中，late mbufs将直接返回给用户错误。

对于early buffer的情况，重排序缓冲区将尝试移动窗口（递增最小序列号），以使mbuf成为有效的一个。为此，Order buffer中的mbufs被移动到就Ready buffer中。任何尚未到达的mbufs都被忽略，且将变成late mbufs。这意味着只要Ready buffer中有空间，窗口将被移动以适应early mbufs，否则将在重新排序窗口之外。

例如，假设我们有一个具有350个最小序列号的200个条目的缓冲区，并且我们需要插入一个具有565序列号的early mbuf。这意味着我们需要移动窗口至少15个位置来容纳mbuf。只要在Ready buffer中有空间，重新排序缓冲区将尝试将至少在Order buffer中的下一个15个槽中的mbufs移动到Ready buffer区。在这一点上的顺序缓冲区中的任何间隙将被跳过，并且这些数据包在报文到达时将被报告为late buffer的数据包。将数据包移动到Ready buffer的过程继续超出所需的最小值，直到遇到了缓冲区中的间隙，即缺少mbuf。

排出mbufs时，重新排序缓冲区首先返回Ready buffer中的mbufs，然后从Order buffer返回到尚未到达的mbufs。

## 17.3.用例：报文分发

使用DPDK数据包分发器的应用程序可以利用重新排序库以与它们相同的顺序传送数据包。

基本的报文分配器用例将由具有多个worker cors的分配器组成。worker对数据包的处理不能保证按顺序进行，因此可以使用重排序缓冲区来尽可能多地重排数据包。

在这种情况下，distributor将序列号分配给mbufs，然后再将其发送给工作人员。随着worker完成处理数据包，distributor将这些mbufs插入重排序缓冲区，最后传输排出的mbufs。