适用于多通道视频流传输的循环线性 2 维表排序算法

郭 伟 龙沪强
(上海交通大学电子工程系图像通信与信息处理研究所,上海 200240)

摘 要 多通道视频流传输的排序问题,在视频数据的传输领域有着重要的地位。提出了一种适用于多通道视频流的循环线性 2 维表排序方法,利用循环 2 维线性表的查表方式实现多通道的视频流分组进行排序处理。与传统的排序方法相比,它可以有效地消除时延不可控,排序复杂度过高,内存管理复杂等方面的缺陷。运用 PC 机对传统方法和该方法及其特点进行了测试,测试结果表明:传统排序方法在分组数量增加的条件下,排序时间呈指数式增长,而本文的方法的排序时间呈线性增长。与几种传统排序方法在排序时间的实验统计数据对比表明:在分组数量增加的情况下,该方法可以节约大量的排序时间。

关键词 视频流分组 数据报方式 线性表
中图法分类号: TP301.6 **文献标识码**: A **文章编号**: 1006-8961(2009)10-2149-05

Circular Linear Two Dimensional Table Sorting Algorithm for Multi-channel Video Stream Transmission

GUO Wei, LONG Hu-qiang
(*Institute of Image Communication and Information Processing, Department of Electronic Engineering, Shanghai Jiaotong University, Shanghai* 200240)

Abstract Sorting problem has an important role in the field of video stream transmission. A sorting algorithm for multi-channel video stream is proposed. It uses the method of look up table in the circular linear two-dimensional table to sort the video stream packets received from multi-channels which is different from a traditional sorting method. It efficiently solves the problems such as uncontrollable delay, high complexity both in sorting and in memory management, the result has been verified through PC simulation which indicates that there is a linear relationship between the increase of packet number and the increase of the time which consumed during the sorting, while in a traditional sorting algorithm, when the number of the packet increases the time consumption will experienced an exponential growth. The statistical datas between this algorithm and traditional algorithms have shown that it can reduce time consumption when the number of packet increases.

Keywords video stream packet, datagram mode, linear table

1 引 言

视频在 Internet 网络中的传输得到了越来越广泛地应用,例如,IP 视频监控系统、可视电话、会议电话系统等,对视频的实时性提出了更高的要求[1]。传输协议的选择也尤为重要,为了减少传输延时,通常采用基于数据报方式的传输协议。目前,采用数据报方式的分组交换的网络体系中,由于数据报协议采用的是无连接的工作方式,分组发送时一边选择传输路径,一边传送信息[2]。同一业务流的不同分组通常会沿着不同路径到达目的地,该路径事先无法预知,由于不同路径存在不同的传输时延,输出端分组流的排列次序会被打乱,无法保证信息的有序性。因此如何对分组进行排序成为迫切需要解决的问题之一[3]。

收稿日期:2009-06-17;**改回日期**:2009-07-14
第一作者简介:郭 伟(1985 ~),男。上海交通大学电子工程系硕士研究生。研究方向为多媒体视频传输。E-mail:ciappcia@sjtu.edu.cn

针对以上问题,提出了一种基于 2 维循环线形表的排序方法,该方法在多通道视频传输的应用中,不但极大地降低了排序的复杂度,缩短了排序过程所产生的时延,有效地应对了分组延时,而且防止了缓存溢出。通过对该方法的性能进行详细的测试与分析,以及与其他传统排序方法的性能比较,证明了该方法的优越性。

2 系统整体结构

传统的视频流传输系统主要包括视频采集部分、编码发送部分、传输通道、接收部分和解码显示部分,其系统架构如图 1 所示。

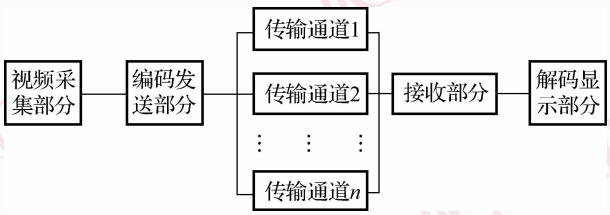

图 1 视频传输系统的总体结构

Fig. 1 The overall structure of video transmission system

在视频流的采集和传输过程中,采集到的视频流在编码发送部分经过编码形成数据帧,拆分数据帧成分组,对每个分组加入自定义的分组头信息,其中头信息里面的内容包含帧的分块总数,分组的长度,分组的帧内序号,分组隶属帧的类型(I,P,B),分组隶属帧的帧号;其中帧内序号是为了接收端重新排序而加入的分组索引号,分组长度是重排序完成后将分组组合成数据帧的必要信息,帧的分块总数是判断数据帧中所有的分组是否达到完全的依据;然后编码发送部分根据网络状况将分组发送入网络,这些分组可能通过不同的路径传输,由于不同的路径产生的时延各不相同,分组到达接收部分的顺序跟发送的顺序会不相同,造成分组的乱序现象。因此在接收部分要对乱序的分组进行排序,使得拆分的数据帧重构。

发送端的结构图如图 2 所示。

传统的排序方法包括选择排序、冒泡排序、快速排序等[4],这些排序方法里,不同的方法对于排序处理的时间和空间复杂度不尽相同,单一的某种实现方法很难对整体的性能做出最优规划。并且,算法对于所有发送的分组都是平等对待,然而,视频流

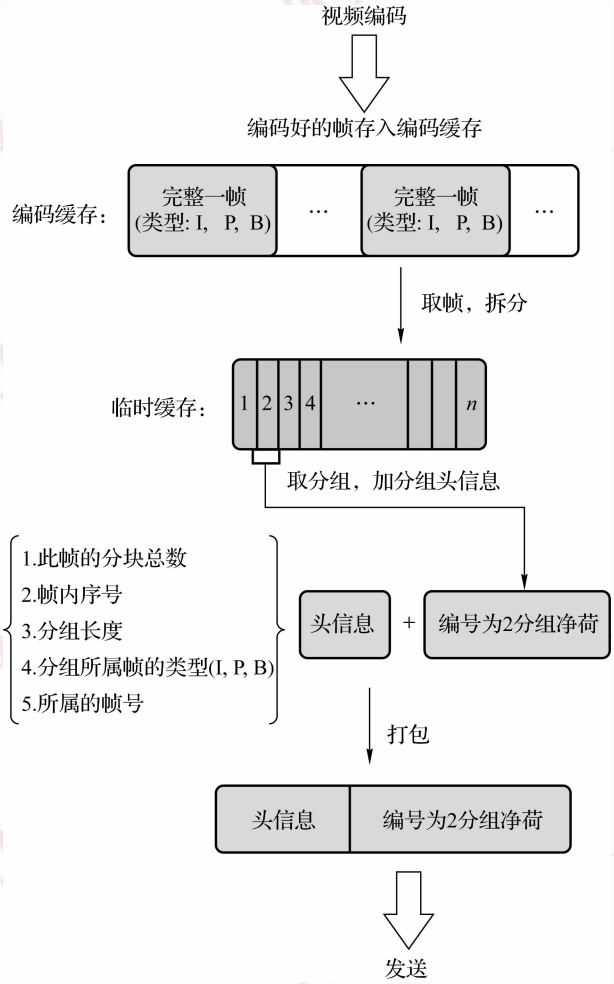

图 2 发送端的结构图

Fig. 2 Structure of sending end

中的分组隶属于一个个的数据帧结构,数据帧只有在其隶属的分组全部接收完全的情况下才可以进行后续解码操作,缺失其中某些分组,例如编码 slice 头、MB 头、运动向量等信息,数据帧无法复原;其次,以上算法也没有加入对于分组数据延时的控制处理。如果采用传统的排序方法,把接收的分组在内存中直接做排序处理,会出现下述问题:第一,不同于一般的数据分组,视频分组隶属于视频数据帧,并且它在帧与帧之间存在依赖性,分组间时延的波动对于计算机内存的分配和回收处理变得比较困难;第二,排序算法的复杂度会随着接收缓存所存的分组数目的增加而指数递增;第三,使用传统的排序方法排序成功,整体系统的延时变得不可控。在此背景下,设计了一种循环线形 2 维表来完成排序过程,可以弥补传统方法的不足。

3　循环线性 2 维表的结构和操作

设计中,在接收端建立一个用于排序的线性数组结构,并用此线性数组的结构来存储各个分组在缓存中的地址和长度,每一个这样的线性数组与一个数据帧唯一对应,并且与一组标识变量相绑定,组成一个表的结构 *address*。标识变量的作用在于判断帧中的分组是否全部到齐,并且记录下数据帧的类型,它的信息成分包括:数据帧类型 *frame_type*,帧序号 *frame_num*, 隶属帧的分组总数 *sect_num* 和已到分组数 *already_arrived_number*。

在内存中建立一个固定表数量大小的循环线性表数组结构,按照每个表对应的帧序号大小依次将表存放入循环线性表数组结构中,组成一个 2 维结构,其 1 维部分的主体对应于一个数据帧中的所有分组的起始地址和分组长度,其 2 维部分对应于按照帧序号编排存放的多个表结构,同时,随着接收过程的继续,在 2 维部分做循环覆盖操作。并且,1 维部分的分组结构数量的大小按照发送部分切割分组时可能切割的最大数目的数值来确定,2 维部分的大小是为了控制网络延时而设计的门限宽度,2 维的维度越大,相应的能够容忍的延时门限也就越宽,比如,若 2 维部分的大小设定为 10,即表示接收部分能够承受的网络延时最大为 10 帧数据的时延。在 2 维部分的循环操作里面,分别用两个指针控制循环线性表的读和写操作,以此来保证数据帧的存入和取出是严格按照帧序号从小到大的顺序来进行的。循环 2 维线性表的结构图如图 3 所示。

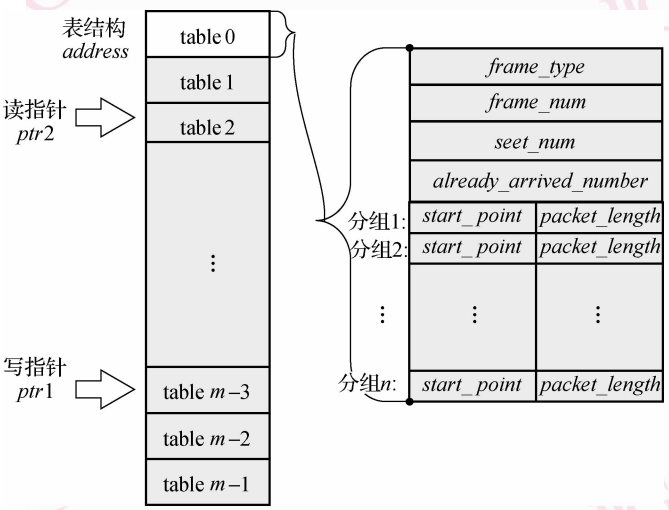

图 3　循环线性 2 维表的结构图

Fig. 3　Structure of the circular linear two dimensional table

循环线性 2 维表的操作如下:

当接收到一个发送数据帧中第一个到达的分组的时候,建立一个对应于此帧的表结构 *address*,初始化表结构,即对表结构 *address* 中的所有变量清零处理。然后从接收分组的头信息中解析得到:

(1)对应于此帧的分块总数, 并赋值给表中变量 *sect_num*;

(2)对表中的已接收到分组的数量变量 *already_arrived_number* 做自增一操作;

(3)对应于帧的序号,并赋值给表中变量 *frame_num*;

(4)对应于帧的类型,并赋值给表中变量 *frame_type*;

(5)对应于此分组的帧内序号 k,在表中的绑定数组位置 k 处写入此分组在接收缓存中的位置 *start_point*,以及此分组的长度 *packet_length*。此后,每收到一个分组,解析分组头信息,看此分组是否属于新的一帧里面的,如果是,则建立一个新表,然后对分组里面的信息按照上述方式赋值;如果接收到的分组所属于的帧在 2 维循环线形表数组里面已经存在,则继续往已存在的对应的表位置写入分组的地址和长度信息,当表的标识变量中的分组总数等于已到达的分组数时,一帧数据的所有分组完全到达,此时,按照数组下标从小到大顺序依次到内存寻址取出各个分组,组合成帧,送入解码和显示部分,再对表中的各个变量进行清空,空出表空间来存放将来到达的数据帧。如果分组所属于的帧序号比在循环 2 维线形表数组中存在的最原始的帧序号还要提前,则证明此分组到达时刻超过了时延的门限,统计反馈信息后做主动丢弃处理。

此外,为了控制延时在一定范围内,同时保证表结构 *address* 在内存中的存放数量不随不同帧中分组的到达累计而膨胀,在内存中构造一个固定表数量大小的循环线性表结构体数组,通过读指针 *ptr*2 和写指针 *ptr*1 的配合,根据读指针 *ptr*2 指向的表,对表中的分组地址数组进行内存寻址组合成帧操作,取完后释放表空间,同时根据写指针 *ptr*1,对于将要构造的新表进行线性顺序存放,当存放的新表到达数组底部的时候,写指针 *ptr*1 循环回表数组起始部分,然后继续下一轮循环,同理,读指针 *ptr*2 也是这样一个循环过程。

设循环线性 2 维表在 2 维上的大小为 m,即表

结构 *address* 在内存中的个数不超过 *m* 个,最大记录 *m* 个顺序帧的组成分组信息,并把这 *m* 个表结构配置成循环线性数组。*ptr*1 代表写指针,指向循环线性 2 维表数组中已存放的最新表位置的下一个位置,*ptr*2 代表读表指针,因为每次读表的时候都是从所存表中最原始的那个表开始读,所以 *ptr*2 始终指向的循环线性表数组中最原始的表位置;两指针初始值都是 0,指向 *table* 0 的位置。同时,设置标识 *flag* 来指示两指针翻转情况,指示可能的表数组填充空间,*flag* 初始值为 0。当 *ptr*1 大于 *ptr*2 的时候,*flag* 等于 0;反之,*flag* 等于 1。

4 循环线性 2 维表的分组排序处理流程

在循环 2 维线性表已经在内存中建立好的基础上,因为分组的到达是无序的,所以针对接收到的分组可以分以下几种情况处理(假定循环 2 维线性表的大小为 *m*,读写指针的翻转用 *flag* 来标识):

(1)如果(分组对应的帧序号 - 循环表数组中最新表的序号)≥*m*;则必须建立新表来存放新到的分组,并且循环 2 维线形表数组内所有已存放的表都已在超

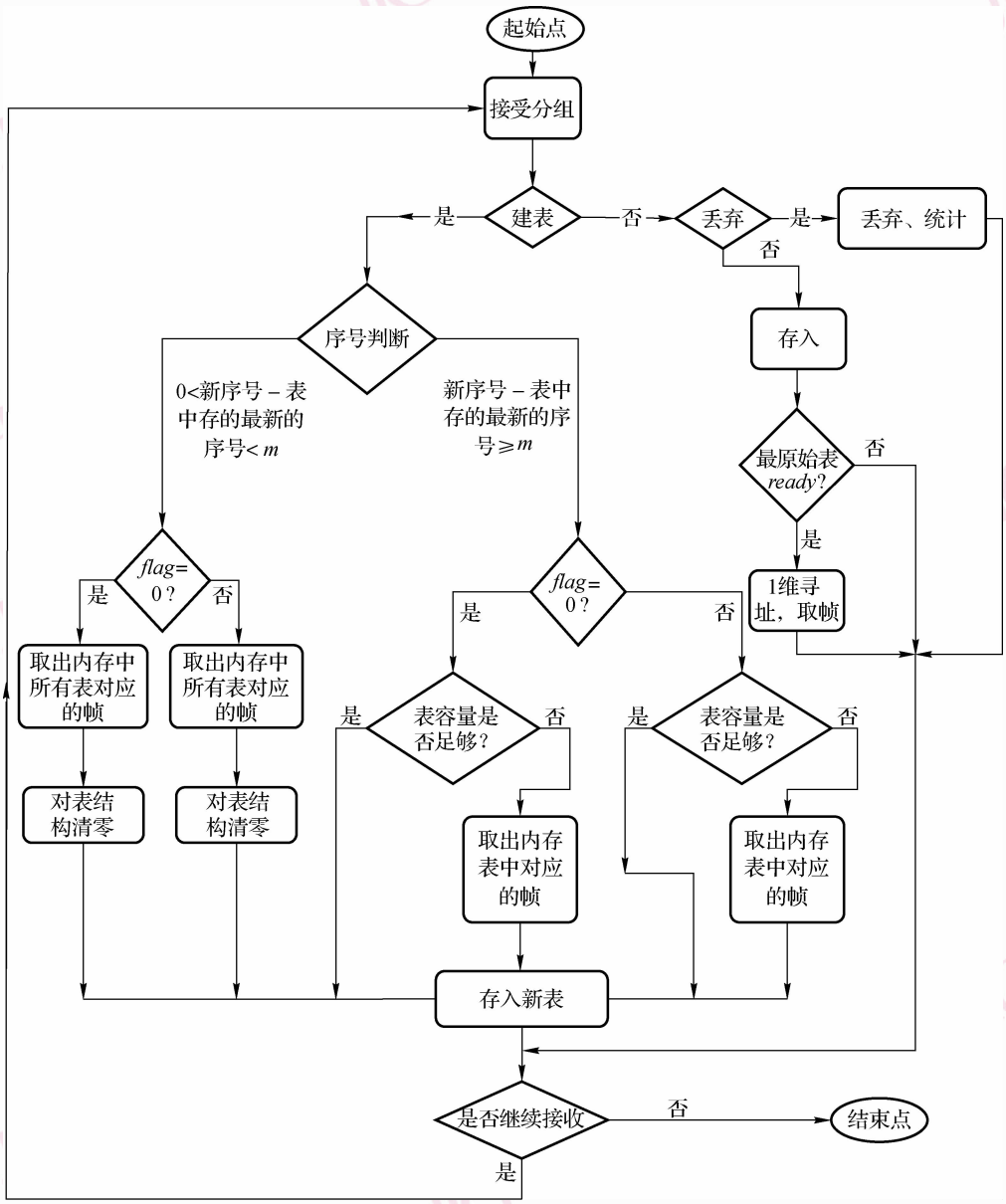

图 4 接收端的分组处理流程图

Fig. 4 The packet processing flow chart of receiving end

时门限以上。对循环 2 维表内所有表依照帧序号从小到大进行判断,寻址,组帧操作,然后将循环 2 维表数据清零,最后写入新的表;

(2)如果 0 <(分组对应的帧序号 - 循环表数组中最新表的序号) < m;则对循环线性表中已超时的表的处理要根据 *flag* 的值来分情况讨论。对超时的表进行判断,寻址,取帧操作,然后将其清零,最后写入新的表;

(3)如果(分组对应的帧序号 < 循环表数组中最原始的帧序号);循环线性表内所有表对应的帧的序号都要大于此接收分组对应的帧序号,所接收到的分组视做超过超时门限到达,统计反馈信息,并做丢弃处理;

(4)如果分组都不属于前 3 种情况;则分组隶属的帧对应的表在内存中的循环 2 维线性表数组内存在,直接把分组的信息写入表中。

接受端的分组处理流程图如图 4 所示。

5　实验结果分析

为了验证循环线性 2 维表排序算法在降低排序复杂度方面的有效性,进行了以下测试,在配置为 Intel Pentium Dual 1.80GHz,1.0G memory 的计算机实验平台上,发送 44 400 帧编码数据,在不同的分组大小情况下,采用不同的排序方法所产生的排序时延,统计结果如表 1 所示。

表 1　多种排序方法产生的排序时延

Tab. 1　Time delay caused by variety of methods

单位:ms

序号	排序方法	排序时间(分组大小/字节:1 k)	排序时间(分组大小/字节:0.5k)	排序时间(分组大小/字节:0.25 k)	排序时间(分组大小/字节:0.125 k)
1	循环 2 维线性表	5	13	31	54
2	冒泡排序	11	53	397	1 611
3	直接排序	10	67	354	1 369
4	希尔排序	21	45	230	1 121
5	快速排序	30	87	199	466

从图 5 可以明显地看到,采用循环 2 维线性表的排序方法排序时间基本上是随着分组数目的加倍线性增加的,其他 4 种传统排序方法的排序时间是随着分组数目的增加呈现指数增长。

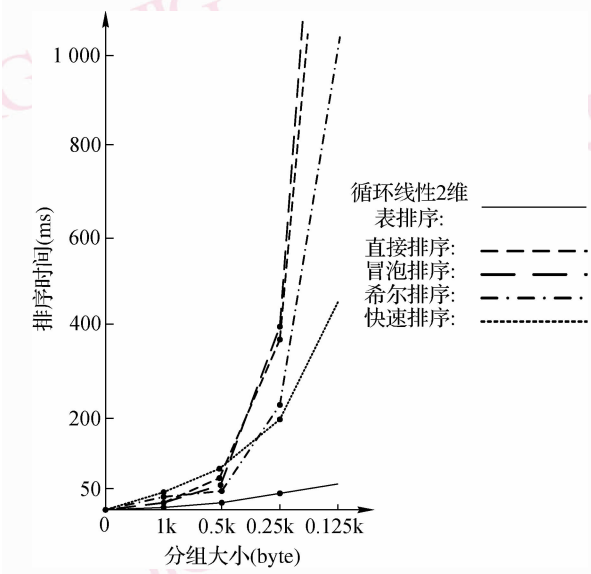

图 5　传统排序和本文排序方法的排序时间对比图

Fig. 5　Comparison chart of sorting time between traditional and the proposed methods

6　结　论

采用循环 2 维线性表的查表的方式实现多通道的视频流分组进行排序处理,可以有效地消除时延不可控,排序复杂度过高,内存管理复杂等方面的缺陷。从实验结果的分析可以得到,本文的设计大大缩短了排序的时间,并且随着分组数目的增加算法的优越性更加明显,这样的设计提高传输的实时性,解决了因为排序时延过大造成了系统瓶颈问题。由于循环线性 2 维表的滚动覆盖操作,克服了缓存溢出问题,同时把网络造成的延时控制在一定的门限范围之内,保证数据的实时性。

参考文献(References)

1　WU Da-peng, Hou Y W T, Zhang Ya-qin. Transporting real-time video over the internet: challenges and approaches[J]. Proceedings of the IEEE, 2000, **88**(12): 1855-1874.

2　Etoh M, Yoshimura T. Advances in wireless video delivery[J]. Proceedings of the IEEE, 2005, **93**(1): 111-122.

3　Zhang Qian, Zhu Wen-wu, Zhang Ya-qin. End-to-end QoS for video delivery over wirelss internet[J]. Proceedings of the IEEE, 2005, **93**(1): 123-133.

4　Wang Li. Comparison and selection of internal sorting algorithm[J]. Software Guide, 2006, (01): 45-46. [王莉. 常用内部排序算法的比较与选择[J]. 软件导刊, 2006, (01): 45-46.]