

基于码流自适应分段的离线码率平滑*

王子磊, 奚宏生, 许书彬

中国科学技术大学网络传播系统与控制联合实验室 网络传播系统与控制安徽省重点实验室, 合肥 230027

E-mail: zlwang@ustc.edu.cn

摘要: 码率平滑通过降低发送码流的突发性能够有效提高网络带宽利用率, 利用原始码流在平均码流上的变化趋势, 提出了一种对发送码率进行自适应分段的平滑算法。该算法能够有效降低客户端缓冲区的需求, 分析函数上的平滑结果显示, 该算法的缓冲区需求不足平均分段PCRTT算法的50%。在针对MPEG4编码影片的实验评估中, 将自适应分段方法用于PCRTT和e-PCRTT后, 波动较大影片的缓冲区需求明显降低。

关键词: 视频点播, 变码率比特流, 码率平滑, 视频传输与发送, 服务质量, 恒码率

Off-line Bandwidth Smoothing with Adaptive-Segmented Traffic

Wang Zilei, Xi Hongsheng, Xu Shubin

Joint lab of Network Communication System and Control, Key lab of Anhui Network Communication System and Control,

USTC, Hefei 230027, P.R.China

E-mail: zlwang@ustc.edu.cn

Abstract: Bandwidth smoothing can effectively increase the utility of the network bandwidth with reducing the bursting of the traffic. Making use of the trend of the changing of the play traffic around the average rate line, an adaptive-segmented smoothing algorithm was proposed in this paper, which can reduce the buffer required of the client. The traffic smoothing result on the analysis function showed the buffer size required by the algorithm is less half than that of the PCRTT. In the following experiment, the MPEG4 movie was used for comparing the performance of the variable smoothing algorithms. The results demonstrate that the PCRTT and the e-PCRTT after using adaptive segment outperform the before with the same number of the segments for the burst movie.

Key Words: Video on demand, Variable bit rate, Bandwidth smoothing, Video transmission and transport, Quality of service, Constant bit rate

1 引言(Introduction)

视频多媒体在现代通信、娱乐等方面具有越来越多的应用, 如视频会议、VOD 点播、时移电视等。基于传统 C/S 架构的流媒体服务器从媒体文件库中提取数据进行流化封包, 通过接入网络以一定的码率发送给客户端, 客户端进行一段时间的缓存(引起播放的时延)后开始播放, 并在播放的同时接收服务器发送的数据包, 直到此次播放的数据包全部发送完成为止。

现代视频服务中, 人们期望享受高质量画面的服务^[1, 2], 因此, 商业的 VOD 系统大多都采用具有图像质量保证的 VBR(Variable Bit Rate)视频编码文件。然而, VBR 视频码率在不同时间尺度上的强突发特性^[3, 4]增加了网络传输的复杂性, 降低了网络带宽利用率^[4], 为此, 研究者们提出了码率平滑的方法^[2], 它利用客户端缓冲区, 在实际播放前将媒体数据预先发送到客户端进行缓存, 从而在较大帧播放前预先发送数据, 降低大数据帧传输的码率。

在离线码率平滑中, VOD 影片的帧信息是可以预先获取的, 因此可以利用这一有用信息优化发送码

流的平滑度, 不同的研究者针对不同的平滑指标提出了相应的策略。Feng 等人提出了 MCBA(Minimum Changes Bandwidth Allocation)方法^[5], 它能够最小化码率变化次数并具有最低的峰值码率; Salehi 等人依据“盖理论”提出了 MVBA (Minimum Variability Bandwidth Allocation)算法^[6], 它在保持峰值码率最低的前提下, 最小化发送码流的变化率, 并较好地支持用户的交互操作^[7]。无论哪一种平滑方法, 都是在发送时间内将码流划分为若干段恒定码率(Constant Bit Rate, CBR)的流进行传输, 这样能够简化网络传输的管理, 提高网络资源利用率^[4]。McManus 和 Ross^[8]依此为基础提出了具有最好平滑性能的 CRTT(Constant Rate Transmission and Transport)传输方法, 并在缓冲区受限的情况下提出了 PCRTT(Piecewise Constant Rate Transmission and Transport)方法, 该方法将发送过程划分为相等的若干段, 并在每一段上采用恒定码率进行发送。PCRTT 与其它码率平滑相比, 具有较小的计算复杂度, 其周期性的带宽变化有利于带宽分配的管理, 且其峰值码率接近于最优峰值码率^[2]。

PCRTT 的分段和码率求解直接影响影片的平滑效果, 因此, McManus 在[9]中应用动态规划方法求解 PCRTT 的这类参数, 但计算复杂度较高。Hadar^[10]在码率计算方面利用 e-PCRTT 切线方法计算各时间段的码率, 与原始的 PCRTT 算法比具有较低的缓冲

*此项工作得到国家自然科学基金资助, 项目批准号: 60574065, 60774038。

区要求;在时间段的划分中,[11,12]将播放码率曲线的数据量进行平均分段,并转换到时间尺度上。然而,在实际的应用中,分段数仍然无法自动确定,且该方法没有直接可解释的物理意义作为参考,与原始PCRTT方法一样需要确定划分参数。

本文以PCRTT码率平滑模型为基础,提出了一种自适应的分段方法。该方法根据码流曲线在平均码率曲线上的分布,自动选择分段数和具体的分段结果,它利用了码流的全局分布和局部分布信息,折中码率平滑和码流跟踪两个目标,从而能够在平滑码率的同时跟踪原始流的变化,降低客户端缓冲区的需求。在性能分析中,自适应分段的缓冲区需求不足平均分段的50%,而针对MPEG4编码视频流的实验中,无论是PCRTT还是e-PCRTT,自适应分段大大降低了码流波动影片的缓冲区需求。

2 离线码率平滑(off-line bandwidth smoothing)

码率平滑的基本思想是利用客户端缓冲区,通过引入初始时延,在播放之前预先发送媒体流,从而达到平滑发送码流的目的。码率平滑算法采用“帧时间” $\Delta T = 1/F$ (F 为影片的帧率,通常为25或30)进行平滑^[2,13],假定影片共 N 帧,每帧大小为 s_i 字节,总大小为 $C = \sum_{i=1}^N s_i$ 。以开始播放时刻为零时刻,则在 i 时刻共播放 $L_i = \sum_{j=1}^i s_j$ 字节的数据,令 B 表示客户端缓冲区的容量, $a_0 (\leq B)$ 表示开始播放时刻已经传输的数据量, a_i 为第 i 帧时间传输的数据量,则第 i 帧时刻传输的总数据量为 $A_i = \sum_{j=1}^i a_j + a_0$ 。在整个播放过程中,为了满足客户端的缓冲区要求,即不发生上溢和下溢,则对任意的时刻 $0 \leq i \leq N$,传输数据量 A_i 必须满足^[2]:

$$L_i \leq A_i \leq \min(L_{i-1} + B, C) \quad (1)$$

令 $U_i = \min(L_{i-1} + B, C)$ 表示上溢曲线。针对第 i 帧时刻,其传输码率为 $r_i = a_i F = (A_{i+1} - A_i) F$ 。由(1)式可以看出,任意可行的码流发送方案必须位于由 L_i 、 $U_i, i=1,2,\dots,N$ 曲线构成的“河”中^[2],如图1所示。图中给出了一种可行的码流发送方案,该曲线有三段构成,分别在 t_1 和 t_2 时刻变化发送码率。

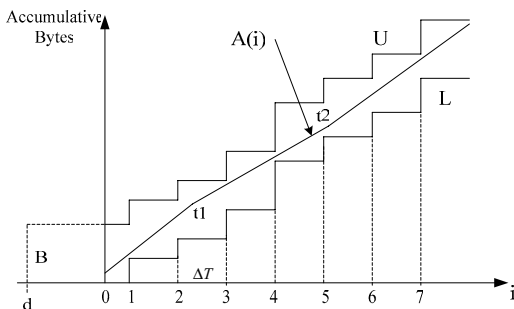


图1 离线码率平滑原理图

当前的各种离线码率平滑方法都是在图1的“河”中构造分段的线性发送曲线,整个码率发送策略由若干个CBR段构成。不同的方法依据不同的平滑准则进行构造,主要的平滑准则包括码率的变化次数、峰值码率、码流变化率等。其中,CBA算法^[13]、MCBA算法^[5]、MVBA算法^[6]从起始点开始,在“河”中画一条射线,并使该射线与 L 或 U 曲线的相交点最远,从而使得每个线段的长度最大化,降低码率变化的次数。该射线构成的码率发送曲线与 L 和 U 曲线相切或相交,从而在相切点和相交点之间构成该段发送曲线的frontier段^[5],用于下一步的计算,各平滑算法的主要区别就在初始点的选择上,导致它们达到不同的性能指标。

在视频流传输中,CRTT是最平滑的传输方式,带宽管理也相对简单。然而,对VBR视频流来说,CRTT传输要求的客户端缓冲区非常大、时延较长,且带宽利用率较低^[8]。为了解决CRTT传输的以上问题,[8]中提出了PCRTT方法,它首先将总发送时间划分为相等的 M 份,在每个时间段内,采用该段的平均码率作为发送码率。由于这样连接的发送码率线可能在 L 曲线下,将此码率线向上整体平移,使得平移后的码率线全部在 L 曲线上,平移后的码率线就是最终发送码率曲线。PCRTT以固定周期进行码率分配,其达到的峰值码率接近于最优值,计算复杂度较低,且这种码率变化形式有利于周期性的带宽管理^[4,8]。

MCBA、MVBA等算法在码率求解过程中同时确定相应的持续时长,PCRTT则将码率平滑问题的两个过程进行分解,每个过程中采用不同的求解方法会使算法达到不同的平滑性能。e-PCRTT(Enhanced PCRTT)算法^[10]在求解给定段的发送码率时,首先在几何上获得起始点在上下溢曲线上的切线,然后在切线与终止点的两个交点处选取中间点作为下一步计算的起始点。在相同的分段下,该方法比PCRTT具有更低的缓冲区需求,而在相同的缓冲区下能够达到更少的码率变化次数^[10]。但它与PCRTT算法一样,没有一种确切的分段方法,需要进行很多不同的尝试^[8,10]来选取满足要求的一组分段。

3 基于原始码流的自适应分段方法(Adaptive-Segmented Smoothing Based on Play Traffic)

3.1 算法(Algorithm)

MCBA、MVBA、PCRTT等算法依据不同的分段和码率计算方法分别达到了不同的优化性能指标。其中,MCBA和MVBA算法将分段决策和码率计算同时完成,而PCRTT将两个过程分离,首先对发送码流进行分段,然后在不同的时间段内分别计算该段的发送码率,从而简化码率平滑过程,降低算法计算复杂度。对PCRTT来说,时间段划分方法和码率计算方法直接影响了PCRTT结果的性能^[9],本文针对

PCRTT 的时间段划分方法，利用原始码流的变化走势，给出一种自适应的时间段划分方法，它能够自主地计算分段数量和具体的分段结果，而不需要设置分段数等参数。

自适应分段算法的核心思想是，在有限分段的情况下，使发送码率尽可能地反映原始流的变化，从而使发送码率曲线和 L 曲线更加紧密，减小需要的缓冲区大小。基于这一原理，算法首先分析原始码率曲线 L 的变化趋势，在趋势的变化点进行分段，使同一变化趋势的点划分到同一时间段内，在发送码率曲线规划时，使其与原始码流紧密地结合，从而降低客户端缓冲区需求和初始时延。具体过程如图 2 所示。

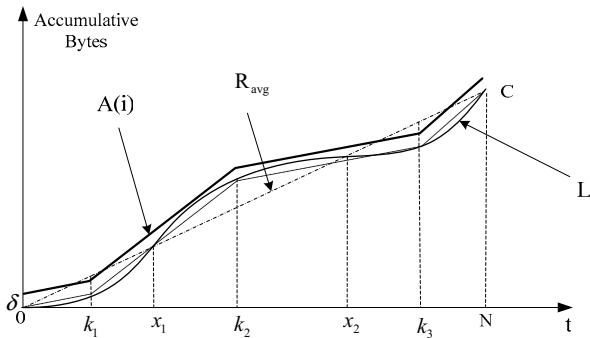


图 2 自适应分段方法原理示意图

首先，连接播放起始点和终止点 (N, C) 得到平均码率曲线 R_{avg} ，由于 VBR 视频流的波动性，原始码流曲线 L 与 R_{avg} 有交点 $x_i, i=0, 1, \dots, M$ ，且 $x_0=0, x_M=N$ ，则所有的 x_i 点将 L 曲线划分为 M 段，称 x_i 点之前的时间段为 x_i 段。针对每个时间段 $(x_{i-1}, x_i), i=1, 2, \dots, M$ ，计算该段内 L 曲线与 R_{avg} 曲线差别最大的点 k_i ：

$$k_i = \arg \max_{x \in (x_{i-1}, x_i)} \{ |L_x - R_{avg} x| \} \quad (2)$$

容易看出，原始码流与平均发送码率相比，在 k_i 点发生趋势变化，因此，选取 k_i 点作为分段边界点，将整个影片划分为 $M+1$ 段，其中第 i 段为包含 x_i 点的 $(k_i, k_{i+1}), i=1, 2, \dots, M-1$ ，第 0 段为 $(0, k_1)$ ，最后一段为 (k_M, N) 。

针对划分后的时间段 (k_i, k_{i+1}) ，采用该段的平均码率 r_i^{avg} （或 e-PCRTT 码率）作为该段的发送码率：

$$r_i^{avg} = \sum_{j=k_i+1}^{k_{i+1}} x_j / (k_{i+1} - k_i) \quad (3)$$

则在时间段 (k_i, k_{i+1}) 内， L 曲线在平均码率线之上的最大距离 $offset_i$ 和平均码率线在 L 曲线之上的最大距离 $buffer_i$ 为：

$$offset_i = \max_{j \in (k_i, k_{i+1})} \left\{ \sum_{l=k_i+1}^j x_l - r_i^{avg} (j - k_i) \right\} \quad (4)$$

$$buffer_i = \max_{j \in (k_i, k_{i+1})} \left\{ r_i^{avg} (j - k_i) - \sum_{l=k_i+1}^j x_l + x_j \right\}$$

为了防止缓冲区发生下溢，码率曲线需要向上平移，由于原始码率曲线中各段的码率曲线是直接相联的（图 2），则只需将初始码率曲线向上平移

$$offset_{min} = \max_i \{ offset_i \}, i=0, 1, \dots, M \quad (5)$$

此时需要的客户端缓冲区最小：

$$B_{min} = offset_{min} + \max_{i \in \{0, 1, \dots, M\}} \{ buffer_i \} \quad (6)$$

上述发送方案要求缓冲区初始具有 $offset_{min}$ 的数据，它在实际播放前以任意可行的码率 r_0 发送，则启动时延为 $t_d = offset_{min} / r_0$ 。与原始 PCRTT 方法的 $O(1)$ 分段相比，自适应分段方法引入了 $O(N)$ 复杂度的分段过程，但与码率求解和平移连接过程的 $O(N)$ 相比，自适应分段并没有显著增加计算复杂度，它仍是一种简单的码率平滑方法。另外，初始码率曲线的向上平移使码率发送到影片播放结束前的 t_w 时刻即可，令 w_k 表示 t_w 前最近的一个分段点，则 t_w 可表示为：

$$t_w = \arg \min_t \{ offset_{min} + \sum_{j=1}^w r_j k + r_{w+1} (t - wk) \geq C \} \quad (7)$$

3.2 性能分析(Performance Analysis)

自适应分段方法利用了原始码率的走势，更加有效地跟踪码率的变化，降低客户端缓冲区 B_{min} 的需求，我们以分析函数 $y = x^n$ (n 为奇数) 作为原始码流曲线为例^[10]，以此说明自适应分段方法达到的效果，如图 3 所示。

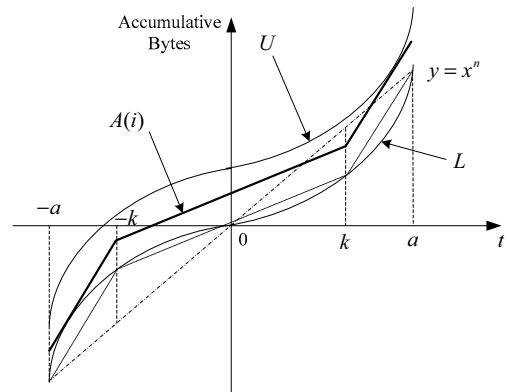


图 3 PCRTT 在 $y = x^n$ 上的自适应分段和平均分

在此例中，以原点对称函数为例，假定视频流范围为 $[-a, a]$ ，在此范围内分别采用平均分和自适应分段，并采用 PCRTT 方法求解发送码率，计算所需的客户端缓冲区。应用自适应分段方法，码流的分

段点为 k 和 $-k$ ，其中 $k = a/n^{1/(n-1)}$ ，它将原始码流的时间划分为 $(-a, -k), (-k, k), (k, a)$ 三段，为了不发生后溢，则码率曲线必须向上平移 $\text{offset}^{\text{ad}}$ ，此时需要的最小缓冲区大小为：

$$B_{\min}^{\text{ad}} = 2 \cdot \text{offset}^{\text{ad}} = 2 \left[k^n + \frac{a^n - k^n}{a - k} (p - k) - p^n \right] \quad (8)$$

其中， $p = \left[\frac{a^n - k^n}{n(a - k)} \right]^{1/(n-1)}$ 。

与[10]中的方法类似，利用 PCRTT 分为相等的三段，即 $(-a, -a/3), (-a/3, a/3), (a/3, a)$ ，它需要的最小缓冲区大小为

$$B_{\min}^{\text{eq}} = 2 \left[(a/3)^n + \frac{3}{2} a^{n-1} (1 - 3^{-n}) (q - a/3) - q^n \right] \quad (9)$$

其中， $q = \left[\frac{3a^{n-1} (1 - 3^{-n})}{2n} \right]^{1/(n-1)}$ 。

从(8)和(9)两式可以看出， B_{\min}^{ad} 比 B_{\min}^{eq} 要小，考察在不同的 n 值下，两者的比例关系 $B_{\min}^{\text{eq}} / B_{\min}^{\text{ad}}$ 如表 1 所示：

表 1 自适应分段方法需要的缓冲区大小比较

n 值	3	5	7	9
比值	2.1133	2.3756	2.4392	2.4450

从表 1 可以看出，自适应分段方法需要的缓冲区大小不足平均分段 PCRTT 的 50%，且在一定范围内

表 2 Star Wars IV 影片在不同算法下的平滑结果

	PCRTT equal	with adaptive	PCRTT adaptive	with equal	e-PCRTT equal	with adaptive	e-PCRTT with adaptive
Required buffer size (MB)	6.741		1.1014		4.8492		1.2142
Peak rate (Bytes/frame)	69612		95149		98360		94678
Average buffer utilization	0.4208		0.3894		0.4519		0.5129
Buffer utilization variability	0.0672		0.0637		0.1074		0.0486

从表 3 可以看出，针对帧大小波动较大的 MPEG4 影片 Star Wars IV，无论是 PCRTT 还是 e-PCRTT，采用自适应分段方法后其需求的缓冲区大小都明显降低，PCRTT 不足 17%，e-PCRTT 不足 26%。在峰值码率方面，PCRTT 采用自适应分段后有少许增加，因为自适应分段使发送码流与原始码流更紧密，而对于已经采用切线方法的 e-PCRTT，自适应分段对峰值码率影响不大。在缓冲区占用率方面，e-PCRTT 通常比 PCRTT 具有更大的平均值和波动性^[4, 10]，但对自适应分段并不成立，自适应分段而是从时间分段的角度提高了平滑算法的性能。

因此，自适应分段对波动较大的影片具有更明

随着 n 值的变大两者的差距在加大，这主要在于自适应分段方法随着码流曲线的变化其分段结果随之变化，而原始 PCRTT 的分段方式是固定的，不能反映码流曲线的变化情况。

4 实验性能比较(Experimental Performance Comparison)

本节通过实验的方法对提出的自适应分段方法进行性能分析。针对广泛采用的 MPEG4 编码的影片，分别采用平均分段和自适应分段方式，并分别应用 PCRTT 和 e-PCRTT 的码率求解方法，组合出四种码率平滑算法，对它们达到的平滑性能进行比较。

选取 Star Wars IV 影片（可以从 <http://trace.eas.asu.edu/h264/index.html> 处下载该影片编码后的帧信息，并能取得各种的统计信息）作为实验样本数据，该影片共有 53953 帧，GoP 大小为 16，平均帧大小为 14791Bytes，而最小和最大帧分别为 6Bytes 和 60665Bytes。可以看出，Star Wars IV 的最大帧和最小帧偏差较大，具有很强的波动性。不失一般性，在实验中选择影片的前 500 帧进行平滑，并重点说明自适应分段对客户端缓冲区需求的影响。

4.1 性能评估(Performance Evaluation)

自适应分段方法根据影片的不同特性对码率发送曲线进行自主划分，而原始的 PCRTT 和 e-PCRTT 算法的分段数通过事先指定。实验中，平均分段和自适应分段采用相同的分段数，在此情况下比较两者的缓冲区需求、峰值码率、平均缓冲区占用率和缓冲区占用率方差。针对 Star Wars IV 影片，分别应用四种平滑后的结果统计如表 2 所示：

显的平滑效果，为了更清楚地了解自适应分段的平滑效果，我们给出具体的平滑曲线。Star Wars IV 影片相应的发送码率曲线如图 4 所示，并假定四种平滑方法的缓冲区约束均为 PCRTT 需求的缓冲区大小：

从图 4 可以看出，在相同的分段数下，自适应分段比平均分段方式具有更低的缓冲区需求，主要是因为自适应分段使分段结果更能体现原始码率曲线的走势，增加发送曲线与原始码流的耦合性。如图 4 中的 (50,150) 段，原始码流的走势发生了重大变化，自适应分段根据码流的变化形式在 100 处引入了一个分界点，而平均分段方式只是根据指定的段数在 167 点才引入分界点，这就造成 PCRTT 在变

化点100处需要较大的缓冲区。

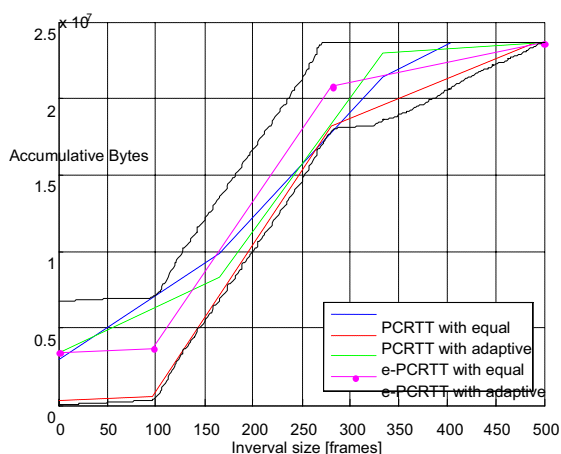


图4 四种不同码率平滑方法达到的平滑曲线

5 结论(Conclusion)

考虑 VOD 影片的离线码流平滑问题, 通过对 MCBA、MVBA、PCRTT 等算法的比较, 提出了一种对 PCRTT 和 e-PCRTT 算法的自适应分段方法。该方法利用原始码流的变化趋势, 更紧凑的跟踪原始码流的变化, 有效地折中码流跟踪和码率平滑的矛盾, 从而降低了客户端缓冲区的需求。分析码流线的平滑结果表明, 自适应分段 PCRTT 需求的缓冲区明显低于平均分段 PCRTT。在针对 MPEG4 编码影片的实验评估中, 自适应分段方法能够有效降低帧波动较大影片的缓冲区需求。然而, 针对给定影片, 其自适应分段的数量和相应的缓冲区需求是固定的, 能够配置分段数和缓冲区需求的码率平滑方法仍需进一步研究。

参考文献(References)

[1] KRUNZ M, HUGHES H. A traffic model for MPEG-coded VBR streams. Proceedings of the 1995 ACM SIGMETRICS joint international conference on Measurement and modeling

of computer systems, 1995:47-55.

[2] FENG WC, REXFORD J. Performance evaluation of smoothing algorithms for transmitting prerecorded variable-bit-rate video. IEEE Transactions on Multimedia, 1999,1(3):302-312.

[3] HADAR O, GREENBERG S. Statistical multiplexing and admission control policy for smoothed video streams using e-PCRTT algorithm. Information Technology: Coding and Computing, Proceedings. International Conference on, 2000:272-277.

[4] HADAR O, GREENBERG S, SEGAL M. EPCRTT-based smoothing and multiplexing of VBR video traffic. Multimedia Tools and Applications, 2007:1-17.

[5] FENG W, JAHANIAN F, SECHREST S. An optimal bandwidth allocation strategy for the delivery of compressed pre-recorded video. Multimedia Systems, 1997,5(5): 297-309.

[6] SALEHI J D, ZHANG ZL, Kurose J *et al.* Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing. Networking, IEEE/ACM Transactions on, 1998,6(4):397-410.

[7] RAO S G, RAGHAVAN S V. Fast techniques for the optimal smoothing of stored video. Multimedia Systems, 1999,7(3): 222-233.

[8] MCMANUS J M, ROSS K W. Video-on-demand over ATM: constant-rate transmission and transport. Selected Areas in Communications, IEEE Journal on, 1996,14(6):1087-1098.

[9] MCMANUS J, ROSS K W. A dynamic programming methodology for managing prerecorded VBR sources in packet-switched networks. Telecommunication Systems, 1998,9(2):133-152.

[10] HADAR O, COHEN R. PCRTT enhancement for off-line video smoothing. Real-Time Imaging, 2001,7(3):301-314.

[11] 马兴. 存储类压缩视频流量平滑技术. 重庆邮电学院学报 (自然科学版), 2002,14(4):42-46.

[12] ZHANG L, FU H. A novel scheme of transporting pre-stored MPEG video to support video-on-demand (VoD) services. Computer Communications, 2000,23(2):133-148.

[13] FENG W, SECHREST S. Critical Bandwidth Allocation for the Delivery of Compressed Video. Computer Communications (Special Issue on System Support for Multimedia Computing), 1995,18(10):709-717.