

# 流媒体传输技术

陈珏奇, 刘 峰

(南京邮电大学 图像处理与图像通信江苏省重点实验室, 江苏 南京 210003)

**摘 要:**文中主要对流媒体传输技术的发展现状进行了研究。首先,分别介绍了 RTSP/RTP、HTTP 渐进式下载以及自适应传输技术并从实现难易度、支持业务及实时性等方面对这三种流媒体传输技术进行对比;其次,对现存的自适应流媒体传输技术提出新的分类方式:推式流媒体传输和拉式流媒体传输,详细分析了两类自适应传输关键技术,给出了相应的实例分析,并对两类自适应传输技术进行对比;再次,介绍了基于可分级视频编码的自适应传输技术;最后,对最新的流媒体传输实例进行详细的介绍。

**关键词:**流媒体;自适应传输;码率控制;SVC

**中图分类号:**TP39

**文献标识码:**A

**文章编号:**1673-629X(2012)03-0006-05

## Media Streaming Technology

CHEN Jue-qi, LIU Feng

(Key Lab of Image Processing and Image Communication of Jiangsu Province, Nanjing University of Posts and Telecommunications, Nanjing 210003, China)

**Abstract:** Study the development status of media streaming technology. First of all, introduce and compare RTSP/RTP, HTTP progressive download respectively; Then, classify the existing adaptive streaming technology into two categories: push-based and pull-based streaming, give detail instruction of two categories of adaptive streaming technology with examples and analysis of adaptive streaming, and compare two categories of adaptive streaming technology; Finally, give the examples of the latest media streaming technology in detail.

**Key words:** streaming media; adaptive streaming; bitrate control; SVC

## 0 引 言

流媒体(Streaming Media)是一种新兴的网络传输技术,它是指用户通过互联网边下载、边播放多媒体数据的一种工作方式。流媒体技术的主要特点是以“流(Streaming)”的形式在互联网上进行多媒体数据的实时、连续传播,客户端不需要花费很长时间将整个多媒体文件下载到本地后才能播放,而只需将起始几秒的数据先下载到本地缓冲区就可以播放了。同时,剩余数据仍持续不断地从服务器递送到客户端,从而维持播放的连续性。

在网络飞速发展的今天,流媒体技术的出现和发展必然会对人们的日常工作以及生活产生深远的影响。

收稿日期:2011-07-28;修回日期:2011-10-30

基金项目:国家自然科学基金项目(61071166);江苏省高校科技成果转化推广项目(JHB10-13)

作者简介:陈珏奇(1986-),男,硕士研究生,研究方向为多媒体通信;刘 峰,教授,博士生导师,从事图像处理与多媒体通信方面的教学和科研工作。

## 1 流媒体传输技术的发展

### 1.1 RTSP/RTP

RTSP/RTP 是目前业界最为流行和广为采用的实时流媒体协议,它实际上由一组 IETF 标准化的协议组成<sup>[1~3]</sup>,包括 RTSP<sup>[1]</sup>、SDP、RTP 以及针对不同编解码标准的 RTP 净载格式等,共同协作来构成一个流媒体协议栈<sup>[4,5]</sup>。

RTSP 是由 RealNetworks 和 Netscape 共同提出的,该协议定义了一对多应用程序如何有效地通过 IP 网络传送多媒体数据,是用来建立和控制一个或多个时间同步的连续音视频媒体流。

RTP 用于承载媒体数据并且为实时媒体数据交互提供端到端的传输服务。它真正实现了实时传输,客户端仅需要维持一个很小的解码缓冲区用于缓存视频解码所需的少数参考帧数据,从而大大缩短了其播放延时。

### 1.2 HTTP 渐进式下载

基于 HTTP 的渐进式下载(Progressive Download)是目前国内外主流的视频网站(如 YouTube、优酷、土

豆等)主要使用的技术。渐进式下载的客户端只需要在开始播放之前等待一小段时间来下载和缓冲媒体文件最前面的一小部分数据,之后便能边下载边播放。

渐进式下载是基于 HTTP 协议的,而 HTTP 协议又承载于 TCP 协议之上。TCP 协议设计的目标是在保证整个网络稳定性以及高吞吐量的前提下,使数据传输速率最大。为了实现这一目标,TCP 采用了慢启动算法,它以一个较低的速率开始传输数据,然后逐步提高传输速率直到接收到目的方的数据分组丢失反馈报告。此时 TCP 认为传输速率已达到最高带宽限制或发生网络拥塞,于是重复上面的过程,这个过程将如此重复地进行下去。TCP 通过重传丢失的数据分组来实现可靠传输。但是,对于流媒体来说,TCP 不能保证所有重传的数据分组能在它们的播放时刻之前到达客户端,而客户端也无法跳过这些数据分组直接播放后面的媒体数据。当这种情况出现时,就会导致播放器画面的停顿。

1.3 自适应传输

自适应传输继承了渐进式下载的简单性,同时克服了它的不灵活性。自适应传输放弃了专门的流媒体服务器,使用标准的 Web 服务器来发送流媒体数据。服务商将容量巨大的连续流媒体文件分割成大量的小文件提供给服务器,这一过程称为“分片”。服务器提供许多内容相同但码率不同的流媒体分片,不同码率对应着不同的分辨率和质量级别。码率高的文件分辨率大、质量级别高、文件容量大,而码率低的文件则相反。

客户端可以在每次向服务器请求时来选择不同码率的流媒体分片。客户端根据当前的网络状况、设备性能等指标来决定分片的选择。

1.4 三种流媒体传输技术比较

●自适应传输技术与传统的 RTSP/RTP 不同之处在于:

- (1) RTSP/RTP 使用的是 RTP 来传输流媒体文件,需要专门的流媒体服务器,而自适应传输只使用标准的 Web 服务器;
- (2) 自适应传输技术降低了系统的部署难度,简化了客户端软件的开发复杂度;
- (3) 流媒体文件分片和索引使得带宽自适应的流间切换、服务器故障保护以及流媒体加密实现更方便。

●自适应传输技术与 HTTP 渐进式下载相比,不同之处在于:

- (1) 渐进式下载只支持点播而不支持直播,自适应传输两者均支持;
- (2) 自适应传输比渐进式下载更具版权保障性;
- (3) 渐进式下载不支持播放过程中的码率切换,

始终以固定码率进行播放,而自适应传输能在播放时随网络环境及设备性能的变化随时进行码率切换。

归纳起来,上述三种流媒体传输技术的综合比较如表 1 所示。

表 1 流媒体传输比较

	HTTP 渐进式下载	RTSP/RTP	自适应传输
服务器实现	Web 服务器	流媒体服务器	Web 服务器
客户端实现	容易	较难	容易
系统部署	简单	复杂	简单
支持业务	点播	直播、点播	直播、点播
实时性(起始延时)	>30s	<2s	约 30s
客户端缓冲区	硬盘,文件大小	内存,小	内存,较小
网络带宽适应	不支持	部分服务器支持	支持,灵活流间切换
服务器故障保护	不支持	不支持	支持

2 自适应传输技术

现在关于自适应传输技术已经存在很多解决方案,大致可以将其分为两类:推式(push-based)和拉式(pull-based)流媒体自适应传输。

2.1 推式流媒体自适应传输

2.1.1 原理

在使用推式流媒体协议时,当服务器与客户端建立了连接,服务器将不停地向客户端推送流媒体数据包直到客户端停止或中断连接。在推式流媒体协议中使用最广泛的就是 RTP 协议。

在通常的推式流媒体传输中,服务器使用编码速率作为发送速率,以便匹配客户端的消耗速率。这样就能确保客户端缓存的大小在一定时间内保持恒定不变,同时使得网络资源利用率最佳。然而,如果数据包丢失或者网络传送出现延时,则客户端数据包的恢复速率将会低于消耗速率,最终导致缓冲区下溢出,播放中断。在这里自适应传输技术能很好地解决这一问题。为了防止缓冲区下溢出,服务器会自动选择发送低码率的媒体流。当网络环境改善时,与之前相反,服务器会自动选择发送高码率的媒体流。通过自动监听可用带宽和缓冲区以及利用选择不同码率的媒体流实现传输速率的调节,推式流媒体传输实现了尽可能高的质量等级视频的平滑播放。

2.1.2 举例

图 1 所示是推式流媒体传输在第三代合作伙伴项目(3GPP)中的应用举例。

在会话建立阶段,服务器为了应答客户端的 RTSP DESCRIBE 消息,会通过会话描述协议(SDP)发送流媒体文件列表以及它们的一些属性,例如:码率、编解码器等。在接收完会话描述信息以后,客户端从提供的音/视频文件中选择最适合其连接速率和解码能力的文件。然后,客户端发送 RTSP SETUP 消息给服务器,让其准备好被选择的文件。RTSP 消息头部包含一

个 3GPP-Adaptation 参数和 3GPP-Link-Char 参数。3GPP-Adaptation 用来告知服务器客户端缓冲区的大小以及缓冲区最小需求 target-time, 3GPP-Link-Char 用来提供客户端的有效接收带宽的大小 (GBW)。在服务器正确获取了关于客户端缓冲区以及接收带宽的信息, 它开始建立一个客户端缓冲区模型, 用来模拟客户端缓冲区大小的变化。因为网络环境和缓冲区需求都在随时间变化而变化, 所以客户端不断通过周期性的 RTCP 反馈报告通知服务器。RTCP 反馈报告中的信息包括: 客户端接收缓冲区的剩余可用空间、播放延时以及检测的网络当前带宽。利用这些信息, 服务器可以建立一个精确的客户端接受缓存模型, 并且为何时选择何种质量级别的文件做出准确的决定。

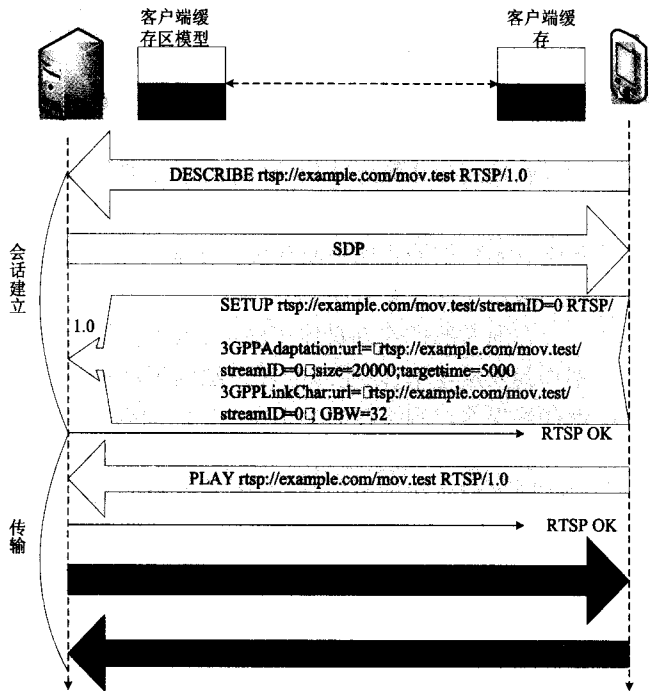


图 1 3GPP 中的推式流媒体传输

## 2.2 拉式流媒体自适应传输

### 2.2.1 原理

在使用拉式流媒体协议时, 客户端占据主导地位, 不断地向服务器请求数据, 服务器则不断响应客户端的请求。因此, 客户端接收数据的速率取决于客户端的请求以及网络带宽。在拉式流媒体协议中使用最广泛的是超文本传输协议 (HTTP)。

与推式流媒体传输方法类似, 拉式流媒体传输也使用码率自适应来防止缓冲区下溢出。图 2 所示是在拉式流媒体传输中使用码率自适应的例子。将媒体流分割成许多时长大约 2 到 3 秒的分片, 每个分片都编码成几种不同码率。当客户端逐个播放视频分片时能

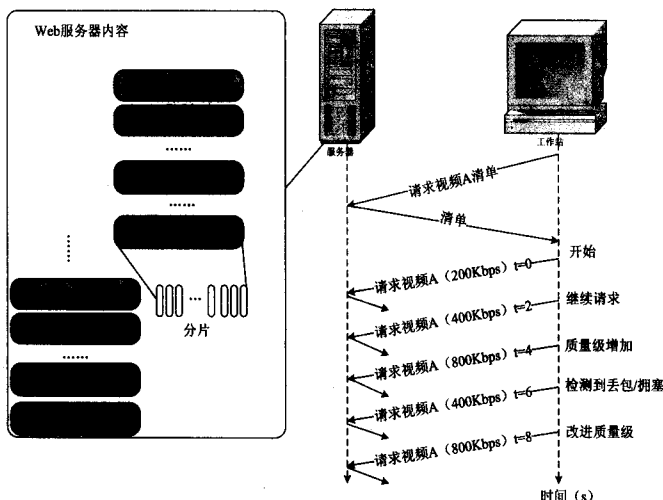


图 2 拉式流媒体传输中码率自适应举例

够无缝地将原媒体流重建。在下载过程中, 客户端根据当前可用带宽自动选择合适的码率视频分片。通过这种方式客户端实现了基于可用带宽的自适应传输。

虽然不同格式视频文件的分片结构是不同的, 但分片的基本原理是一样的。通常音频帧是由一些相同时长的音频采样组成的, 每个音频帧都能用音频编解码器单独解码。因此, 只需要将一定数量的音频帧组合成与分片时长相同的音频数据并填入分片。对于视频数据的处理完全不同, 因为视频帧不能单独解码。因此, 在分片中视频分区是以图片组的形式存在的。一个 GoP 是一组由一个帧内编码 I 帧和许多预测 P 帧组成的帧序列。编码器只需要简单地调整 GoP 中帧的数量使得 GoP 的时长和分片的时长相等。

### 2.2.2 举例

图 3 所示是客户端的拉式流媒体传输所需的基本配置图。如图所示:

- 客户端需要一个称为客户端列表文件或播放列表的文件, 用来对请求的分片进行查找定位;
- 播放缓存管理是所有自适应传输客户端必须具有的基本功能, 用来从特定码率的文件中选取分片;
- 拉式流媒体传输中使用 HTTP 传输客户端向服务器发送的请求。

当客户端所请求的分片定位后, 客户端从列表文件中找到对应的 URI, 然后与服务器建立一条或多条连接。客户端通过监视自己的缓冲区来决定选择何种码率的文件。

### 2.3 推式和拉式自适应传输比较

表 2 简要地给出了推式和拉式流媒体传输的不同点。它们之间一个最主要的不同就是构建服务器的复

杂度。如之前提到,在拉式流媒体传输中,码率控制通常是客户端的任务,这样很显而易见地简化了服务器。推式流媒体传输需要专门的服务器执行 RTSP 协议或者具有类似功能的协议,此协议需要有码率控制算法、重传机制以及高速缓存。这也使得推式流媒体传输没有拉式流媒体传输划算。

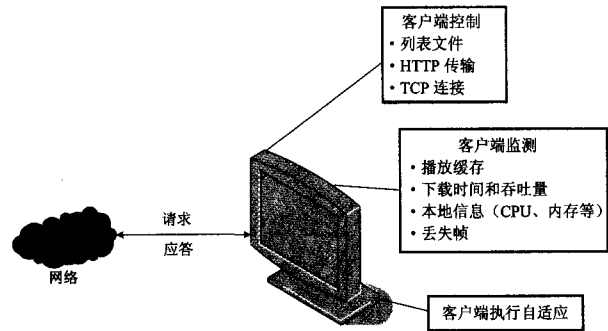


图3 客户端的拉式流媒体传输应用

表2 推式和拉式流媒体传输对比

	推式	拉式
媒体源	广播站和服务器 例如: Widows Media, Apple QuickTime, RealNetwork Helix, Cisco CDS/DCM	Web 服务器 例如: LAMP, Microsoft IIS, RealNetwork Helix, Cisco CDS
协议	RTSP, RTP, UDP	HTTP
带宽利用率	高	稍低
视频监视和用户跟踪	RTCP	私有
多播支持	支持	不支持

推式流媒体传输技术的一个优势就是支持多播传输。多播传输允许服务器只需给一组客户端发送一次数据包,数据包在网络传输时进行复制。客户端可以随时加入或离开多播组,如此一来,客户端就能只获得它们需要的数据包。拉式流媒体传输只支持单播传输,即在服务器和客户端之间建立点到点的连接。因此,服务器不得不逐一给客户端发送数据,同样的,网络节点也必须逐一给客户端发送数据。当客户端数量大时,就极大地减少网络效率。

3 基于 SVC 的自适应传输

前面已经讨论了基于对整个文件进行不同质量级编码的以及基于对分片进行不同质量级编码的自适应方法。现在介绍另一种方法,称为可分级视频编码(SVC)。SVC 是 H. 264/ MPEG4 AVC 视频压缩标准的扩展<sup>[6,7]</sup>,目前为止,它只支持视频。在 SVC 中,视频流是由分层结构组成的。基础层提供最低的质量级别,表现在帧率、分辨率以及信噪比(SNR)。在基础层上是增强层,它对上述的参数中的一个或多个进行增强。增强层能够单独存储或在网络中传输。因此,可以通过添加或移除增强层来改变视频流的码率<sup>[8,9]</sup>。

SVC 的一个关键优势就是它可以在各层之间以最

小的附加冗余传输信息。然而,使用 SVC 编码,每层之间信息的相同处很少。这使得 SVC 非常利于存储不同质量级别的媒体流。SVC 的另一个优势在于,当网络环境突然变化造成增强层信息不能发送,不需要服务器或客户端的介入,SVC 就能自动地降低媒体流质量级别。这和多码率编码形成了鲜明的对比,因为在网络环境变化时,多码率编码需要服务器或客户端进行质量级别的选择。虽然 SVC 优势显著,但是实现复杂并且对编解码器限制性增加,这使得 SVC 一直发展缓慢。

4 最新流媒体传输实例介绍

4.1 微软流媒体平滑传输

微软流媒体平滑传输(smooth streaming)<sup>[10]</sup>的实现是基于受保护的互操作文件格式(PIFF)<sup>[11]</sup>,它是 MP4 文件格式的扩展。在平滑传输中,拥有相同码率的分片存放在一个 MP4 文件中。因此,不同码率都有一个独立的文件。分片一般设计时长为 2 秒。

图 4 所示是一个分片式 MP4 文件。每个分片式 MP4 文件是以阶层式数据结构组成的,阶层中最基本构造块称为数据盒(box),它里面存放了音频/视频数据以及元数据。每个数据盒的类型用四个字母作为标识符表示。文件开始存放的是一个文件类型数据盒(ftyp box),用来描述文件的版本信息,后面紧跟着一个视频元数据盒(moov box)。每个分片的音频/视频数据存放在媒体数据盒(mdat box)中。在分片式 MP4 的封装结构中,紧挨着媒体数据盒前的是视频分片盒(moof box),该盒中存放着信号信息,包括:分片的顺序计数器、分片中采样的数量以及每个采样的时长。

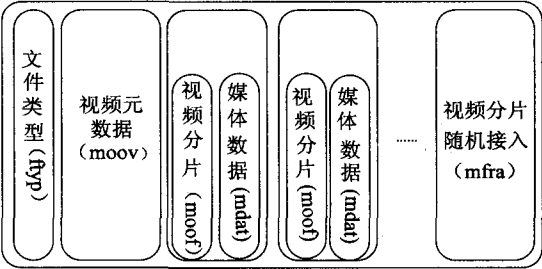


图4 分片式 MP4 文件

要使客户端成功请求到特定码率的分片,首先必须知道服务器上提供哪些分片。这个信息是在客户端与服务器会话开始时通过列表文件提供的。当客户端获得了列表文件,它利用其中的信息开始向服务器进行 HTTP 请求。在 HTTP 请求信息的报头包含两个信息:请求分片的码率以及时间戳。当服务器收到客户端对特定码率分片的 HTTP 请求,首先它需要决定从哪个 MP4 文件中搜索分片。决定的依据由服务器端的另一个列表文件给出,该列表文件描述了每个 MP4

文件中包含分片的码率。当服务器选择好 MP4 文件之后,开始从中搜索请求的分片。如图 4 所示,每个 MP4 文件中包含一个视频分片随机接入数据盒(mfra box),其中包含每个分片的位置以及时间戳。服务器通过请求中提供的时间戳信息与其进行匹配从而找出需要的分片。当找到分片后,将其媒体数据盒和视频元数据盒发送给客户端。

## 4.2 苹果 HTTP 实时流媒体传输

苹果 HTTP 实时流媒体传输(live streaming)<sup>[12]</sup>采用了一种基于 MPEG2 流媒体格式的分片存储方法。媒体流进行分割时,不断地读取流媒体传输文件并将其分割成等时长的分片(默认 10 秒)。长分片拥有更多的时间冗余,易于压缩。然而,长分片同样会减少音视频块的间隔尺寸,意味着将不能很好地对网络带宽变化产生自适应。

与微软方案类似,客户端的列表文件也保存了服务器提供的分片信息。这个列表文件的格式是 MP3 列表文件格式的扩展,不同的是,该列表文件头部有一个 EXTM3U 标示符。图 5 所示是一个由两个分层等级组成的列表文件示例。图中左边的文件是一个高等级文件,它关联着右边两个低等级文件。每个低等级文件的关联是由 URI 提供并且前面有一个标示符 EXT-X-STREAM-INF。这个标示符的属性名称为 BANDWIDTH,用来指示对应的低等级文件分片的编码方式。在示例中,分片有两种编码方式:一种低质量等级的编码(100Kbps),另一种高质量等级的编码(2Mbps)。

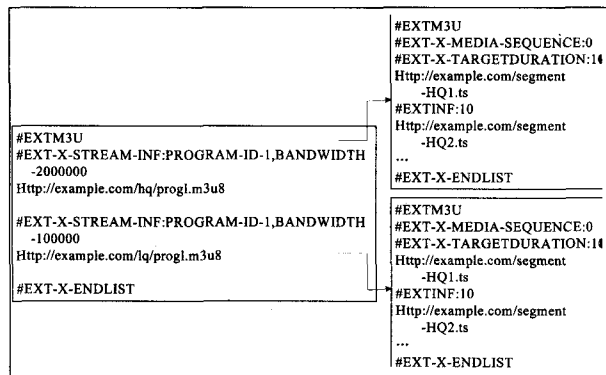


图 5 HTTP 实时流媒体传输列表文件

在图中右侧的低等级文件显示了可供下载分片的链接。在 EXTM3U 标示符之后是一个 EXT-X-MEDIA-SEQUENCE 标示符,它列出了在播放列表中第一个分片的序列号。每个分片的 URI 都用 EXTINF 标示符进行标示。这个标示符有一个属性值表示对应分片的时长。在此示例中,每个分片的时长为 10 秒。

对于固定时长的媒体流,EXT-X-ENDLIST 标示符标示了播放列表的结束。但是,在实时流媒体传输

中这个标示符不存在。因为此时播放列表是动态变化的。在实时流媒体传输中,客户端必须周期地更新列表文件。列表文件更新周期由列表文件最后一次重载时是否变化决定。若变化了,更新周期则为播放列表最后一个分片的时长;若没变化,更新周期则为 EXT-X-TARGET-DURATION 标示符标示的时长的倍数。

## 5 结束语

随着网络的发展和普及,人们对丰富多彩的流媒体的需求将越来越大,流媒体必将成为未来社会的主流媒体形式之一。

流媒体在中国的宽带建设中被列为最主要的应用之一,越来越多的网络服务商开始提供网络媒体服务解决方案,以增强其在网络服务上的优势。相信自适应传输技术的研究一定会推动流媒体的迅猛发展,为广大的用户提供更加丰富多彩的多媒体服务体验。这在对现存商业模式造成巨大冲击的同时,也带来了巨大的商机。

## 参考文献:

- [1] Schulzrinne H, Rao A, Lanphier R. Real Time Streaming Protocol (RTSP)[S]. RFC 2326, 1998.
- [2] Handley M, Jacobson V. SDP: Session description protocol [S]. RFC 2327, 1998.
- [3] Schulzrinne H, Casner S, Frederick R, et al. RTP: A Transport Protocol for Real-Time Applications[S]. RFC 3550, 2003.
- [4] ISMA 2.0: Internet Streaming Media Alliance Implementation Specification, Version 2.0[S]. 2005.
- [5] Transparent End-to-End Packet-Switched Streaming Service (PSS): Protocols and Codecs, 3GPP TS 26.234[EB/OL]. 2010-12. <http://ftp.3gpp.org/specs/html-info/26234.htm>.
- [6] Schwarz H, Marpe D, Wigand T. Overview of the scalable video coding extension of the H.264/AVC standard[J]. IEEE Trans on CSVT, 2007, 17(9): 1103-1120.
- [7] ISO/IEC 14496-10:2009-Information technology-Coding of audio audio-visual objects-Part 10: Advanced Video Coding, Annex G(Scalable Video Coding)[S]. 2009.
- [8] 逢 栋, 姜昌金. 流媒体技术及其开发方法[J]. 计算机技术与发展, 2006, 16(2): 145-147.
- [9] 韩礼国, 才书训. 流媒体 QoS 端到端自适应控制策略综述[J]. 计算机技术与发展, 2006, 16(11): 246-249.
- [10] Microsoft Corporation. IIS Smooth Streaming[EB/OL]. 2009. <http://www.iis.net/overview>.
- [11] Microsoft Corporation. Protected Interoperable File Format [EB/OL]. 2010. <http://go.microsoft.com/?linkid=9682897>.
- [12] Pantos R. HTTP Live Streaming[EB/OL]. 2009. <http://tool.ietf.org/html/draft-pantos-http-live-streaming>.