一种可重构流媒体调度算法

黄 霄[①] 邬江兴[①] 张晓娟[①] 刘 阳[②]
[①](国家数字交换系统工程技术研究中心 郑州 450002)
[②](约克大学计算机科学系 约克 Y0105DD)

摘 要：针对现有流媒体算法在异构环境下性能恶化的问题，论文提出一种支持用户异构性的可重构流媒体调度算法——RSMS 算法。该算法引入了追赶流的概念，能重构追赶流的速率来服务于具有不同接收带宽的异构用户，并最终通过流合并达到资源共享的目的。分析了 RSMS 算法的最佳组播调度间隔、所需的平均服务器带宽和服务器带宽需求分布。仿真实验表明该算法简单高效，可扩展性好。

关键词：流媒体调度算法；用户异构性；追赶流；可重构
中图分类号：TP393 **文献标识码**：A **文章编号**：1009-5896(2010)02-0255-06
DOI: 10.3724/SP.J.1146.2009.00136

A Reconfigurable Streaming Media Scheduling Algorithm

Huang Xiao[①] Wu Jiang-xing[①] Zhang Xiao-juan[①] Liu Yang[②]
[①](*National Digital Switching System Engineering & Technology R&D Center*, *Zhengzhou* 450002, *China*)
[②](*Department of Computer Science*, *University of York*, *York*, Y0105DD, *UK*)

Abstract: Current stream scheduling algorithms have to compromise their performance in heterogeneous environments. A Reconfigurable Streaming Media Scheduling (RSMS) algorithm supporting user heterogeneity is proposed. The algorithm by using catching-up streams supports heterogeneous users reconfiguring catching-up streams at different transmission rates and achieves resources sharing ultimately through stream merging. The optimal multicast interval, the average server bandwidth and server bandwidth distribution for RSMS algorithm are all analyzed. The efficiency and scalability of RSMA algorithm are verified through simulation experiment.

Key words: Streaming media scheduling algorithm; User heterogeneity; Catching-up stream; Reconfiguration

1 引言

为降低流媒体服务器和网络带宽消耗，研究者提出了很多高效的流媒体调度算法。这些算法大致可分为两类。一类采用“用户拉”的方式，即以用户为中心，用户请求到达后才调度相应的资源进行视频传输，比如以补丁技术(Patching)为代表的流合并算法[1]。另一类采用“服务器推送”的方式来满足用户需求[2]，即服务器把要传输的视频分成多个片段，每个片段在特定的信道上周期性广播传输。理论上，广播算法可以支持无限多用户，具有很好的扩展性。但是，由于广播协议每隔一定时间才重复传输请求的视频，存在固有的访问延迟。因此，“服务器推送”方式仅适用于点播率高的视频，当视频访问率较低时，资源利用率低。上述算法均假定用户是同构的，即具有相同的接收带宽和存储能力。但是，用户终端在处理能力及接入带宽上的巨大差异使得传统同构环境下的流调度技术很难应用于异构环境。在用户异构条件下，这些算法要么不再适用，要么性能急剧恶化。

为应对上述挑战，研究者提出了不同的解决方案。代表性算法有 HeRO[3]，CAR[4]和 UHB[5]等。这些算法均基于服务器推送方式，除了继承广播类协议的固有缺点外，都假定用户侧有较大的接收带宽，当用户带宽较小时，用户服务延迟很大，降低了服务质量[6]。“用户拉”方式的流调度算法很少支持用户异构性，只有文献[6]提出了一类混合 Batching 和 Patching/ERMT 的算法来支持用户异构性。该类算法通过对用户侧接收带宽进行分类，当用户接收带宽等于视频播放率时，采用 Batching 算法；当大于或等于两倍视频播放率时,采用 Patching 或 ERMT 算法；当接收带宽介于两者之间时，通过引入自适应流来完成流合并。该类算法的代表为 EHS-E 算法，仿真结果表明 EHS-E 混合算法能够取得最佳性能，但是算法缺乏有效的理论分析。

2 可重构流媒体调度算法

本节首先介绍追赶流的概念，然后详述可重构

2009-02-02 收到，2009-09-28 改回
国家 973 计划项目(2007CB307102)和国家 863 计划项目(2008AA01A323)资助课题
通讯作者：黄霄 samue1309@sina.com

流媒体调度算法。文中主要符号及含义约定如下：r 为视频播放率，单位为 bit/min；B_c 为用户侧接收带宽，以视频播放率为单位，且假定 B_c>1；L 为视频长度，单位为 min；λ 为用户到达率，单位为请求/min，本文假定用户从头至尾观看节目，不考虑快进、快退等交互请求；W 为追赶流窗口阈值，也称为组播间隔，单位为 min；T 为平均两个连续组播流的时间间隔，单位为 min；N 为在 T 时间内到达系统的用户数目，$N=\lambda L$，在不引起混淆的情况下也称为用户到达率；B_u 为用户侧缓存，表示能容纳 B_u 时间单位的视频数据。

2.1 追赶流原理

图 1 介绍了追赶流的概念。当第 1 个用户到达系统，服务器发起一条常规组播流(Regular multicast Stream，RS)来为该请求服务。常规组播流 RS 从头至尾传输整个视频节目，如图 1 中 RS1 和 RS2 所示。P 秒钟后接收带宽为 B_c 的用户 A_1 到达系统并请求相同的视频。为响应用户请求，服务器发起一条速率为 $r\cdot B_c$ 的单播流来保证用户的零延迟播放，用户接收并缓存该单播流的数据，同时进行视频播放。由于当 B_c>1 时，该流不断“追赶”距离当前时刻最近的 RS，故称为追赶流(Catching-up Stream，CS)。当经过 T_{CS1} 时间后，CS 和 RS 中传输的视频帧相同，CS 终止，用户加入 RS 流接收并继续缓存视频数据，即 CS 与 RS 合并。这种类型的追赶流利用全部的用户侧带宽来接收初始的视频片段直至追赶上目标组播流，称为第 1 类追赶流，记为 CS1。易知 CS1 流的追赶时间为

$$T_{\mathrm{CS1}}=\frac{P}{B_c-1} \tag{1}$$

完成流合并前，CS1 流的视频消耗(以比特为单位)为

$$C_{\mathrm{CS1}}=rB_cT_{\mathrm{CS1}}=\frac{rB_cP}{B_c-1} \tag{2}$$

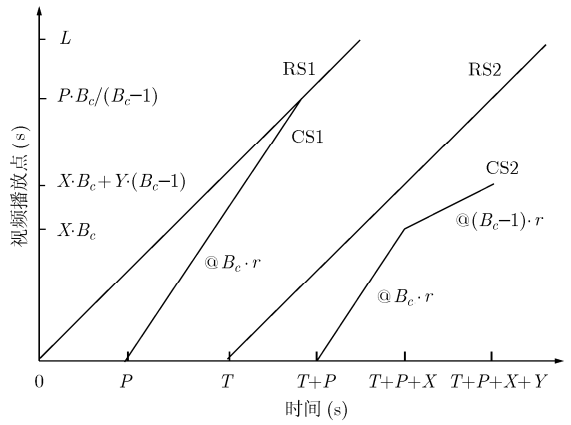


图 1 追赶流原理图

用户侧缓存需求为

$$\mathrm{Buf}_{\mathrm{CS1}}=r(B_c-1)T_{\mathrm{EP1}}=r\cdot P \tag{3}$$

第 1 类追赶流的缺点是视频消耗较大，视频数据在 RS1 和 CS1 中均被发送。借鉴文献[6]中增强自适应流的思想，可利用两阶段追赶的单播流来降低流消耗。图 1 中，用户 A_2 在 $T+P$ 时刻到达，距离最近发起的常规组播流 RS2 的时间为 P 秒。为服务该用户，服务器发起追赶流 CS2。CS2 分为两个阶段：第 1 阶段时长为 X，流速率为 $r\cdot B_c$；第 2 阶段时长为 Y，流速率被重构为$(B_c-1)\cdot r$，在第 2 阶段接收 CS2 的同时，接收 RS2 中的数据并缓存。把这类流称为第 2 类追赶流，记为 CS2，把 CS2 两个阶段的时长及其相应的传输速率称为 CS2 的特征参数。为了保证用户的连续播放，CS2 要满足两个条件：一方面，为最大程度的节省网络资源，CS2 传输的视频数据应该在 CS2 结束时刚好播放完，用户后续播放所需的视频数据均来自 RS2，即满足 $X\cdot B_c\cdot r+Y\cdot(B_c-1)\cdot r=(X+Y)\cdot r$；另一方面，为保证用户的不间断播放，CS2 传输的视频应该等于用户错过的 RS2 已传输的数据，即满足 $X\cdot B_c\cdot r+Y\cdot(B_c-1)\cdot r=(X+P)\cdot r$。由此可得

$$X=\begin{cases}\dfrac{2-B_c}{B_c-1}P, & 1<B_c<2\\ 0, & B_c\geq 2\end{cases}\text{和}Y=\begin{cases}P, & 1<B_c<2\\ \dfrac{P}{B_c-1}, & B_c\geq 2\end{cases} \tag{4}$$

需要说明的是，当 $B_c\geqslant 2$ 时，$X=0$ 意味着一开始用户便同时接收 CS2 和 RS2。CS2 持续的时间长度 T_{CS2} 为

$$T_{\mathrm{CS2}}=X+Y=\frac{P}{B_c-1},\ B_c>1 \tag{5}$$

CS2 消耗的视频数据 C_{CS2} 为

$$C_{\mathrm{CS2}}=X\cdot B_c\cdot r+Y\cdot(B_c-1)\cdot r=\begin{cases}\dfrac{r\cdot P}{B_c-1}, & 1<B_c<2\\ r\cdot P, & B_c\geq 2\end{cases} \tag{6}$$

用户侧缓存需求为

$$\mathrm{Buf}_{\mathrm{CS2}}=r(B_c-1)T_{\mathrm{CS2}}=r\cdot P \tag{7}$$

2.2 RSMS算法描述

对比式(2)和式(6)可知第 2 类追赶流效率高，带宽消耗小，因此采用第 2 类追赶流来设计 RSMS 算法。由式(6)知，当 B_c>2 时，CS2 的视频消耗与 B_c 无关。研究表明[7,8]，利用用户侧 $2r$ 的带宽即可获得较大的系统收益。即使采用更复杂的算法利用全部用户带宽，也只获得较低的额外收益，并且增大了实现复杂度。因此，RSMS 算法最多只需用户有 $2r$

的接收带宽。

由于用户对各个视频访问相互独立，不失一般性，假定用户访问相同的视频节目，且用户端缓存足够大(比如能够容纳整个视频节目)。RSMS 算法的处理流程如图 2 所示，某时刻，接收带宽为 B_c 的用户向系统请求视频服务，假定此时剩余服务器带宽资源为 C_n。如果系统中没有正在传输的视频组播流，则服务器发起一条 RS 组播流来为该用户服务。如果存在正在传输的组播流，且假定用户距最近的组播流的时间间隔为 P，则服务器为该请求发起一条 CS 来保证用户的实时播放，用户同时侦听服务器指定的 RS。当 CS 的高速传输阶段结束时，用户继续接收 CS 的同时，接收并缓存 RS 中的数据。CS 的特征参数可以由 B_c，P 和式(4)来确定。需要说明的是，CS 的持续时间 T_{CS} 应该小于视频长度 L，否则需要为用户发起一条新的视频组播流。同时，为了避免 CS 过长，当 P 超过一定的组播间隔 W 时，服务器发起新的 RS 组播流来服务用户请求。

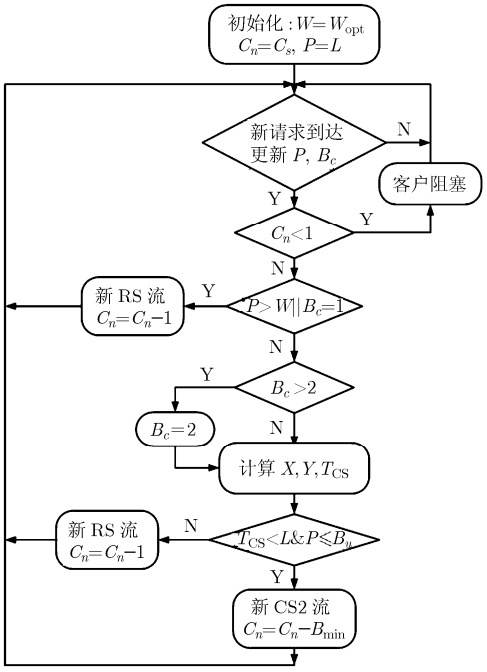


图 2 RSMS 算法的请求处理流程

3　RSMS 算法分析

本节对 RSMS 算法进行分析。假定用户到达服从参数为 λ 的 Poisson 分布，文中所提到的带宽均以视频播放率为单位。

3.1 最佳组播间隔

假定服务器容量不受限，最佳组播间隔 W_{opt} 定义为给定用户到达率 λ 和视频长度 L 情况下，使得服务器平均带宽消耗最小的组播阈值。

假定随机过程$\{S(t):t>0\}$，其中 $S(t)$为从 0 到时刻 t 请求的视频总量(以比特计)。定义服务器平均带宽为 $B_{\mathrm{RSMS}}=\lim\limits_{t\to\infty}\dfrac{S(t)}{r\cdot t}$。由 2.2 节知，为了避免 CS 过长，当用户的到达时间距最近的 RS 流的时间间隔 P 大于 W 时，触发一个新的组播流。$\{t_j\}_{j=0}^{\infty}$ $(t_0=0)$ 表示 RS 的触发时刻。则 S(t)为更新过程，t_j为更新点，意味着系统在 t 时刻$(t\geqslant t_j)$的行为与 t_j 之前的行为无关。把连续两个 RS 组播流 t_j-1 和 t_j 的时间间隔$[t_{j-1},\ t_j)$称为第 j 个周期，T_j表示第 j 个周期的长度，R_j表示第 j 个周期请求传输的总视频比特。由更新过程知，对任意 j，有 $B_{\mathrm{RSMS}}=\dfrac{E(R_j)}{r\cdot E(T_j)}$。不失一般性，只考虑第 j 个周期并忽略下标，且假定组播流的发起时刻为 0。易知，

$$E[T]=W+1/\lambda \tag{8}$$

在 T 时间内平均服务的用户数目为 $N=1+\lambda W$，其中第 1 个用户触发 RS 组播流。假定第 i 个用户的接收带宽为 B_i。由于 RSMS 算法最多只需要两倍视频播放率，因此，当 $B_i>2$ 时，令 $B_i=2$。假定服务器容量能够保证为这些用户实时提供服务，则服务器需要传输的视频总量为

$$E[R]=Lr+r\sum\nolimits_{i=1}^{N-1}\frac{P_i}{\min\{B_i,2\}-1} \tag{9}$$

式(9)右边第 1 项为常规组播流的消耗，第 2 项为所有单播追赶流的消耗。则 RSMS 算法所需的平均服务器带宽为

$$B_{\mathrm{RSMS}}=\frac{E(R)}{r\cdot E(T)}=\frac{L+\sum\nolimits_{i=1}^{N-1}\dfrac{P_i}{\min\{B_i,2\}-1}}{1/\lambda+W} \tag{10}$$

假定用户带宽为一随机变量，最小带宽为 1，最大带宽为 $b(b\geqslant 2)$，其概率密度函数为 $f(x)$。定义 B^* 为

$$B^*=\int_1^2 xf(x)\mathrm{d}x+2\int_2^b f(x)\mathrm{d}x \tag{11}$$

用 B^*来对 B_i进行估算，且考虑到 P_i的均值为 $W/2$，则

$$B_{\mathrm{RSMS}}\approx\frac{L+\lambda\dfrac{W^2}{2}\dfrac{1}{B^*-1}}{W+1/\lambda} \tag{12}$$

对式(12)求关于 W 的导数并令结果为零，可得最佳阈值 W_{opt} 为

$$W_{\mathrm{opt}}=\frac{\sqrt{1+2\lambda L(B^*-1)}-1}{\lambda} \tag{13}$$

3.2 平均服务器带宽

RSMS 算法平均服务器带宽需求 B_{ave} 定义为最

佳组播间隔下，为保证用户实时播放所需的平均服务器带宽。由式(12)和式(13)，RSMS 算法的平均服务器带宽需求为

$$B_{\text{ave}}=\frac{1}{B^*-1}\left(\sqrt{1+2\lambda L(B^*-1)}-1\right)=\frac{\lambda\cdot W_{\text{opt}}}{B^*-1} \quad (14)$$

由式(14)知，当 W 增大时，B_{ave} 逐渐下降，这是因为越来越多的用户共享同一个组播流；但当 W 增大到某一值时，B_{ave} 反而逐渐上升，这是由于追赶流过长从而导致系统资源利用率降低。因此，存在一个使得 B_{ave} 最小的 W 值，即 W_{opt}。这说明随意的选择 W 值可能造成资源浪费。

3.3 服务器带宽需求分布

上一小节提供了平均服务器带宽的计算方法，但平均服务器带宽并不一定能够精确描述一段时间间隔内的用户带宽需求，基于平均服务器带宽设计服务器容量可能使得用户服务质量下降。因此，本节研究服务器带宽需求的概率分布函数，确定满足一定服务质量要求的服务器带宽需求。具体地说，在一个组播周期内，假定用户所请求的总的服务器带宽为 B_{total}，在给定 λ，W 和 L 条件下，确定分布函数 $F_B(c)=P\{B_{\text{total}}>c\}$，其中 c 为任意给定的服务器带宽。根据该概率分布函数可以确定基于 B_{ave} 设计的视频服务器能否满足用户需求以及满足一定服务质量需求的服务器带宽值。

实际上，由 3.1 节描述的更新过程知，在一个周期内到达的第 1 个请求触发组播流 RS，假定该时刻为 0。在$(0,W)$时间内假设有 n 个用户请求到达。为服务这 n 个请求，发起的追赶流的总带宽消耗 $G(n)$ 为

$$G(n)=\left(\sum_{i=1}^{n}\frac{r\cdot P_i}{\min\{B_i,2\}-1}\right)\Big/r\approx\frac{1}{B^*-1}\sum_{i=1}^{n}P_i \quad (15)$$

由式(10)，服务这 n 个用户所需的服务器带宽为

$$B_{\text{RSMS}}^{n}=\frac{L+G(n)}{W+1/\lambda} \quad (16)$$

进而

$$\begin{aligned}P\{B_{\text{RSMS}}^{n}>c\}&\triangleq P_B(n,c)=P\left\{\frac{L+G(n)}{W+1/\lambda}>c\right\}\\&=P\{G(n)>cW+c/\lambda-L\}\\&\triangleq P_G(n,\gamma)\end{aligned} \quad (17)$$

其中 $\gamma=cW+c/\lambda-L$。也就是说，一个周期内如果有 n 个用户到达，则这 n 个用户的带宽需求分布由追赶流的总带宽消耗 $G(n)$的分布完全确定。由全概率公式，服务器带宽需求分布为

$$F_B(c)=\sum_{n=0}^{\infty}P(n)\cdot P_B(n,c)=\sum_{n=0}^{\infty}P(n)\cdot P_G(n,\gamma) \quad (18)$$

其中 $P(n)=\dfrac{(\lambda W)^n}{n!}e^{-\lambda W}$ 为$(0,W)$时间内到达 n 个用户的概率，$P_G(n,\gamma)$ 由如下定理确定：

定理 $P_G(n,\gamma)=\Theta(\gamma,\boldsymbol{I}(1,n+1))$，其中 $\Theta(\gamma,I(1,n+1))$ 可由递归确定：

$$\begin{aligned}\Theta(\gamma,\boldsymbol{I}(\alpha,\beta))=&\frac{(n+1-\alpha)-\gamma\cdot(B^*-1)/W}{\beta-1}\Theta(\gamma,\boldsymbol{I}(\alpha,\\&\beta-1))+\frac{\gamma\cdot(B^*-1)/W-(n+2-\alpha-\beta)}{\beta-1}\\&\cdot\Theta(\gamma,\boldsymbol{I}(\alpha+1,\beta-1))\end{aligned} \quad (19)$$

初始条件为

$$\Theta(\gamma,\boldsymbol{I}(\alpha,\beta))=\begin{cases}0, & n+1-\alpha\le\gamma\cdot(B^*-1)/W\\1, & n+2-\alpha-\beta>\gamma\cdot(B^*-1)/W\end{cases}$$

其中 $\boldsymbol{I}(\alpha,\beta)=(i_1,i_2,i_{n+1})$，为 $n+1$ 维向量，并且 $i_\alpha=i_{\alpha+1}=\cdots=i_{\alpha+\beta-1}=1$，其余分量均为 0。

证明 由于用户到达服从 Possion 分布，P_i 为第 i 个$(1\le i\le n)$用户的到达时间，则这 n 个到达时间 $P_1<P_2\cdots<P_n$ 与相应于$(0,W)$上均匀分布的独立随机变量的顺序统计量有相同的分布。由式(16)知，G 为均匀顺序统计量的线性组合。定义 $\xi=(c_1,c_2,c_3,\cdots,c_{n+1})$，其中 $c_i=(n+1-i)/(B^*-1)$，则 $c_1>c_2>\cdots>c_n>c_{n+1}=0$。由 $\boldsymbol{I}(\alpha,\beta)$ 的定义知，第 1 个非零分量为第 α 项，最后一个非零分量为第 $\alpha+\beta-1$ 项。易知 ξ 对应的第 α 和 $\alpha+\beta-1$ 个分量的值分别为 $c_F=(n+1-\alpha)/(B^*-1)$ 和 $c_L=(n+2-\alpha-\beta)/(B^*-1)$。对 P_i 进行归一化处理，由 ξ 分量的递增特性及文献[9]的定理 3 知：

$$\begin{aligned}\Theta(\gamma,\boldsymbol{I}(\alpha,\beta))=&\frac{c_F-\gamma'}{c_F-c_L}\Theta(\gamma',\boldsymbol{I}(\alpha,\beta-1))\\&+\frac{\gamma'-c_L}{c_F-c_L}\Theta(\gamma',\boldsymbol{I}(\alpha+1,\beta-1))\end{aligned} \quad (20)$$

初始条件为 $\Theta(\gamma,\boldsymbol{I}(\alpha,\beta))=\begin{cases}0, & c_F\le\gamma'\\1, & c_L>\gamma'\end{cases}$，其中 $\gamma'=\gamma\cdot(B^*-1)/W$。把 c_F，c_L 代入式(20)化简即可。

证毕

式(18)中，需要有一个标准来确定 n 的上限值进而使得计算结果在一定误差范围之内。假定$(0, W)$区间的 n 个用户到达把该区间划分为 n 个小区间，每个小区间的长度 $d=W/n$，选取 n 的标准是使得长度为 d 的区间内到达两个或两个以上用户的概率 $P(n\ge2)$小于 ε (比如 $\varepsilon=0.05$)。由理论计算和仿真实验可得，取 n=40 即可保证 $P(n\ge2)<0.05$。

4 仿真实验

在 NS2 的仿真环境中，假定用户带宽服从均值

为 μ=1.5，方差为 σ=0.5 的截断正态分布，用户带宽最小为 1，最大为 10。视频数目为 M，视频长度为 L 分钟。视频点播率服从参数为 θ 的 Zipf 分布。用户到达率为 λ 请求/min，在 L 时间内的平均用户到达数为 $N=\lambda \cdot L$。

由于用户对各个视频的访问互相独立，先考虑单个视频的情况。图 3 分别描述了当 N=25，100 时，不同组播间隔下的服务器容量需求分布 $F_B(c)$。从图中可以得到如下结论：

(1)随着 N 的增大，组播间隔对服务器容量需求分布的影响增大。

当 N=25 时，最佳组播间隔 W_{opt}=17.4 min，而组播间隔 w 在 15 min 至 20 min 之间变化时，对系统性能影响不大，组播间隔变化范围为 5 min。但随着 N的增大，W对系统性能的影响也随之增大。当 N=100 时，最佳组播间隔 W_{opt}=9.5 min，而组播间隔 w 的变化范围在 8 min 至 10 min 之间。

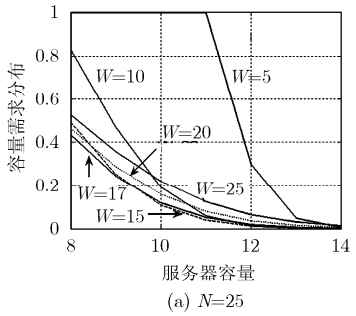
(2)在最佳组播间隔下，系统并不一定能够取得最佳性能。

当 N=25，在 $W=W_{\text{opt}}$=17.4 min 和 w=15 min 下，要使用户容量需求分布满足 $F_B(c)<0.001$，需要的服务器容量分别 15 和 14。即当 $w=W_{\text{opt}}$ 时反而会使系统性能降低，用户阻塞率增大。

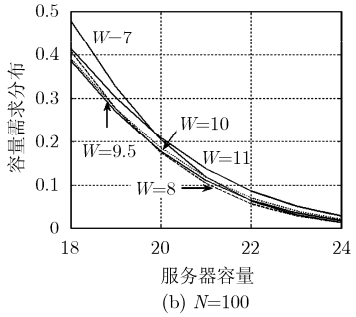
(3)基于平均带宽设计服务器容量，难以满足用户服务质量需求，用户阻塞率高。

由式(14)，当 N=25 和 N=100 时，可得平均服务容量 C_{ave} 分别为 8 和 18。如果以平均带宽来设计服务器容量，必然造成较高的用户阻塞率，降低用户的服务体验质量。图 4 给出了不同用户到达率情况下，基于平均带宽(由式(14)计算)设计服务器容量时，RSMS 算法用户阻塞率(用“RSMS”表示)的仿真结果。图 4(a)为单个视频即 M=1 的情况，同时给出了用户请求的服务器容量大于平均带宽的概率计算结果，即由式(18)计算 $F_B(C_{\text{ave}})$的值，用“Calcu”表示；图 4(b)为多个视频的情况。可以看出，在单个视频情况下，用户阻塞率在 30%–45%之间；在多个视频情况下，用户阻塞率有所下降，但用户阻塞率在所考虑的各种情况下仍在 10%–30%之间。

图 5 给出了 RSMS 与 Patching，基于单播的 TVoD 和 EHS-E 算法的性能比较结果。图 5(a)为客户遇忙则阻塞离开的情况。当客户带宽小于 2 时，Patching 不能充分利用客户侧带宽，只能为该客户发起一条组播流，降低了系统资源利用率，增大了客户阻塞率，其性能接近于 TVoD。而 RSMS 通过重构追赶流速率，能充分利用客户侧带宽进行流合并，显著提高了系统性能。EHS-E 略优于 RSMS，原因是当客户带宽大于 2r 时，EHS-E 采用 ERMT 算法，而 RSMS 本质上采用 Patching 算法。但客户的异构性使得 ERMT 算法的潜在收益降低很多。图 5(b)以客户食言率为性能指标，如果系统没有空闲资源则客户进入等待队列。客户等待容忍时间服从均值为 2 min，方差为 1 min 的正态分布。可以看出，由于 EHS-E 采用 MCF 调度视频队列，没有考虑客户的等待时间属性，当服务器容量较小时，其食言率明显高于 RSMS。这说明在异构环境下，选择适当的请求调度策略非常重要。

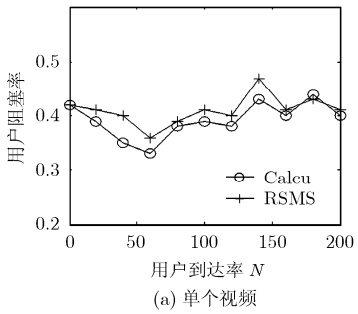


(a) N=25

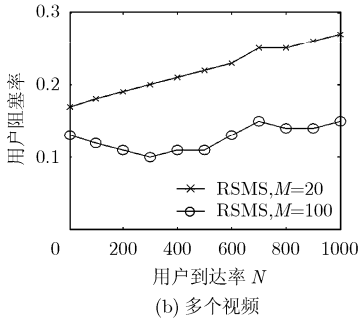


(b) N=100

图 3 服务器容量分布

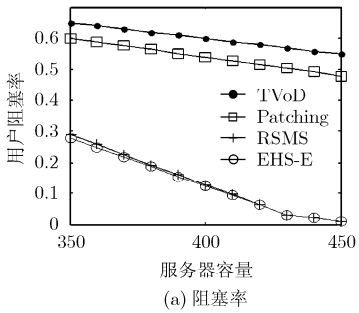


(a) 单个视频

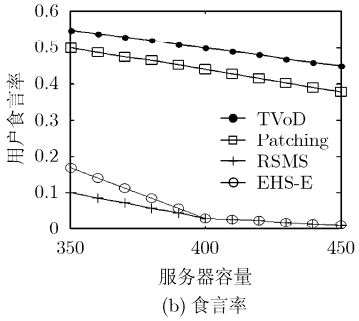


(b) 多个视频

图 4 平均服务器带宽下的用户阻塞率



(a) 阻塞率



(b) 食言率

图 5 多种算法的性能比较

5 结束语

本文提出并分析了一种简单高效的可重构流媒体调度算法，该算法通过重构追赶流的速率来支持流媒体用户的异构性。追赶流使得后申请流媒体服务的用户能够“追赶”上正在播放该流媒体视频的组播流，进而加入该组播组并实现流合并，达到资源共享的目的，保证了可重构流媒体算法的可扩展性。

参 考 文 献

[1] Eager D L, Vernon M K, and Zahorjan J. Minimizing bandwidth requirements for on-demand data delivery. *IEEE Transactions on Knowledge and Data Engineering*, 2001, 13(5): 742–757.

[2] Hua K A. Video delivery technologies for large-scale video delivery technologies deployment of multimedia applications. *Proceedings of the IEEE*, 2004, 92(9): 1439–1451.

[3] Bagouet O. Aperiodic broadcast protocol for heterogeneous receivers. SPIE Conf. Multimedia Computing and Networking 2003 (MMCN'03), Santa Clara, California, Jan. 2003: 220–231.

[4] Lin C T and Ding J W. CAR: A low latency video-on-demand broadcasting scheme for heterogeneous receivers. *IEEE Transactions on Broadcasting*, 2006, 52(3): 336–349.

[5] Ding J W, Lin C T, and Lan S Y. A unified approach to heterogenous video-on-demand broadcasting. *IEEE Transactions on Broadcasting*, 2008, 54(1): 14–23.

[6] Qudah B and Sarhan N J. Towards scalable delivery of video streams to heterogeneous receivers. Proc. ACM Multimedia, California, USA, October. 2006: 369–375.

[7] Zhi Y J and Wang B J. Urgency-based batching policy for streaming media. HPCC, Dalian China, 2008: 1722–1730.

[8] 智英建,等. 最大紧迫度优先的流媒体批调度算法. 电子与信息学报, 2008, 30(12): 3018–3022.
Zhi Y J, *et al.*. The maximum urgency first batching algorithm for streaming media. *Journal of Electronics & Information Technology*, 2008, 30(12): 3018–3022.

[9] Diniz M C, De Souza e Silva E, and Gail H R. Calculating the distribution of a linear combination of uniform order statistics. *INFORMS Journal on Computing*, 2002, 14(2): 23–29.

黄 霄： 男，1981 年生，博士生，从事下一代网络技术与流媒体调度方面的研究.

邬江兴： 男，1953 年生，教授，中国工程院院士，从事网络与交换技术方面的研究.

张晓娟： 女，1980 年生，双硕士，工程师，从事宽带信息网络技术方面的研究.

刘 阳： 男，1980 年生，博士，工程师，从事人工智能与流媒体调度方面的研究.