基于节点状态模型的传输模式自适应选择算法*

柯 璘，张兴明，沈剑良
（国家数字交换系统工程技术研究中心，郑州 450002）

摘 要：针对现有的流媒体传输方法不能在高效分发数据的同时保证数据接收质量的问题，提出了一种传输模式自适应选择算法。该算法首先根据节点状态模型，将节点状态分为数据接收质量和综合状态值两部分，对各节点本身稳定状态和节点重要性进行综合判定，再结合现有数据缓冲模型进行传输模式自适应选择。仿真结果表明，该算法有效地提高了数据在传输过程中的分发效率，同时通过对服务器负载一定程度的控制，在保证数据接收质量的同时，提高了系统的可扩展性。

关键词：流媒体分发；节点状态模型；传输模式；自适应选择
中图分类号：TP37 **文献标志码**：A **文章编号**：1001-3695(2013)06-1798-04
doi:10.3969/j.issn.1001-3695.2013.06.051

Adaptive transmission mode selection algorithm with node state model

KE Lin, ZHANG Xing-ming, SHEN Jian-liang
(*National Digital Switching System Engineering & Technological R&D Center*, *Zhengzhou* 450002, *China*)

Abstract: Since the existing transmission mode for streaming media system can not guarantee the reception quality of the data while keeping the high distribution efficiency, this paper presented a new adaptive transmission mode selection algorithm. With a new node state model, this algorithm divided the node state into data receive quality and consolidated state value, so as to achieve the comprehensive determination for each node, and then selected the transmission mode adaptively based on the existing data buffer model. The simulation results show that this algorithm is effective to enhance the distribution efficiency of the data, meanwhile, the system scalability is improved under a certain degree control of the server load.

Key words: streaming media distribution; node state model; transmission mode; adaptive selection

0 引言

面对当前大规模流媒体分发服务需求，如何利用有限的资源条件来满足快速增长的用户需求，以及如何在提高数据分发效率的同时保障一定的服务质量，已经成为流媒体分发技术研究中的核心内容[1]。现有的流媒体系统中，数据的传输模式按照其工作原理一般可分为服务器推模式、用户拉模式和推拉结合的混合模式三类[2]。其中基于 CDN 架构的流媒体系统主要传输模式为推模式，基于 P2P 架构的流媒体系统主要传输模式为拉模式，而基于 CDN 和 P2P 混合架构的流媒体系统主要传输模式则为推拉混合模式。

在推模式中，数据节点在收到数据之后立即进行转发，以达到数据在网络中高效传播的目的。但是这种传输模式通常是基于单树或者多树的树状网络拓扑结构，抗抖动性能较差，在父节点出现故障或者退出以后，无法保证子节点数据的传输质量，需要引入复杂的父节点切换过程，过程中常导致节点向服务器发送超时紧急请求的数量急剧增加，由此加重了服务器的负载，乃至产生拥塞。

在拉模式中，节点根据其所需数据在邻居节点的分布等情况，主动向数据拥有的节点发送请求，而拥有数据的节点在收到数据后，并不立即进行转发，而是发布对数据的拥有情况，并根据收到的请求数据，回传给请求节点对应数据。这种传输模式具有较高的鲁棒性，能够适应复杂多变的网状拓扑结构，有效地保证数据节点的传输质量。但是拉模式传输的延迟较大，特别是这种延迟还会随着传播跳数的增加而不断累积，离源服务器跳数越多的节点，数据的传输延迟越大，整个系统的传输效率较低。

在现有的推拉混合模式中，服务器根据数据节点的请求返回对应的数据，或者是根据节点的数据预订推送对应的数据，而数据节点则通常是根据不同的情况进行数据请求或者是数据的预订，即通过拉模式或者是推模式获取数据。这种传输模式的目的在于结合推、拉模式各自的优点，相关的研究成为了近年来的热点[3~5]。但是两种传输模式之间切换时机和切换方式的选择极其复杂，因此现有算法还未能在最优实现高效分发数据的同时保证数据的接收质量。

针对现有推拉调度算法还不能最优地实现高效分发数据的同时保证数据接收质量的情况，分析其原因在于它只根据单个节点的数据接收情况或者其他标准单一片面地判定，并基于此作出传输模式的选择。本文提出了一种节点状态模型(node state model)。

1 相关研究

文献[6]在数据源节点的控制下，将数据的分发过程根据

收稿日期：2012-10-15；**修回日期**：2012-11-28 **基金项目**：国家“863”计划重点资助项目(2009AA012201)；上海市科技委员会重大科技攻关项目(08dz501600)

作者简介：柯璘(1986-)，男，福建莆田人，硕士研究生，主要研究方向为流媒体分发网络、大型数据服务器(vvvip111111@sina.com)；张兴明(1963-)，男，教授，主要研究方向为 Web 服务、SoC 设计等；沈剑良(1982-)，男，讲师，主要研究方向为嵌入式系统与 SoC、可重构计算等.

网络使用率分为两个部分,在网络使用率较低的时间段内,使用推模式进行分发传输。之后进入拉模式阶段,需求节点需要根据所需要的数据查找拥有对应数据的节点,向该节点发送请求,拉回数据。针对推拉混合传输模式的特点,文献[7]设计的混合流媒体分发模型由 CDN 服务器及 P2P 网络组成的自治域构成,初始分发阶段用户请求较少时采用传统 C/S 模式向节点提供快速响应;后续分发中,将自治域内的节点组成中心化的 P2P 网络,转为采用 P2P 的拉模式,而 CDN 服务器承担着内容服务器和索引服务器双重角色;接下来若用户不断增加,仅靠 P2P 足以将流媒体的分发过程持续下去,CDN 则完全成为索引服务器。这样既保证流媒体服务的质量,又将流量控制在自治域范围内,可有效降低骨干网络带宽和服务器部署成本。

在文献[8,9]中,推拉结合策略采用的是节点开始时先通过拉模式进行下载,当到达稳定状态后再切换到推模式。这里提出节点将数据根据时间分成不同的时间片,在不同的时间片内分别采用拉模式或者推模式进行工作。在推模式下,节点对各邻居在上一个时间片内的评估,向不同的邻居节点定制当前时间片内所需要的数据。这样数据节点在接收到新的数据后,就可以根据数据的定制情况立刻决定是否转发该数据包,即推送数据包。而若是某个时间片的数据接近播放时间点却还未完成下载,则改成采用拉模式,向拥有对应数据的节点紧急拉取数据。在此基础上,文献[10]明确了缓冲区的推拉边界,避免了数据的重复请求,提出可以根据数据接收质量,动态调整边界,以此减小服务器负载。

文献[11]对于 CDN 的树状拓扑和 P2P 的网状拓扑两种服务方式进行了详细的分析和比较;文献[12]提出了两种结合的建议。文献[11]中将数据节点的传输通路进行了分类,从一定角度分析了将网状拓扑视为基于多树拓扑变形的可行性;文献[12]中提出,为了减小 CDN 服务器负载的问题,将 P2P 网络中的节点进行了分类,CDN 服务器主要的服务对象为节点状态较好的节点,其他节点则主要是 P2P 网络中的其他节点获取数据。

通过对当前主要混合分发模式的分析,其传输模式选择的本质在于对数据节点状态的判断。文献[6,7]等传输模式将数据分发开始阶段进行推模式传输是基于分发开始阶段,数据节点数量较少、状态较为稳定,拓扑结构上较为稳定,能够很好地发挥推模式传输高效性的优点,而分发到达一定阶段,数据节点数量众多,拓扑结构上动态性特点突出,因此使用网状拓扑对应的拉模式传输,保证数据的接收质量。文献[8~10]则视节点开始传输的时候,拓扑新建,是动态不稳定的,因此才用拉模式传输,而在传输一定的时间后,某个节点所用的传输关系将会固定,形成较为稳定的拓扑结构,适用于高效分发的推模式。前后两种策略的区别在于对稳定状态的理解是从不同的角度出发,前者是从整个系统所有节点的角度,而后者是从特定节点在网络拓扑结构中相关数据链路的角度。这两种策略都只是片面地从某一个角度出发,并不能够全面、充分地结合两种传输模式的优点。

本文根据分析的结论,提出一种针对数据节点状态的判断模型,首先从每个节点的角度初步判断其基本的数据接收质量,如果节点接收质量好,再从整个系统的角度判断节点在分发网络中的综合状态情况。数据分发的过程中同时根据实时的节点状态判断结果,结合当前的数据缓冲模型,提出一种自适应的传输模式选择算法,从而更充分、有效地结合两种传输模式各自的优点。

2　节点状态模型(node state model)

相关研究表明,传输模式的切换主要取决于节点状态的判定,即数据节点状态的判定对混合传输模式的过程起到指导性的作用。本文将节点状态的相关参数分为静态值和动态值两种。其中静态值包括节点的上传带宽和下载带宽;动态值包括节点实时的播放质量、持续时间、数据的平均到达速率和上传速率等。

模型首先通过数据接收质量 RQ_i 对数据节点的接收状态进行初步判定,对节点的传输模式初步调整,然后根据需求情况依公式求得节点的综合状态值 SV_i,作为进一步的综合判定,再参照当前服务器的输出状态 OSV_{Severs} 来作出二次调整。

具体对于节点 i,每隔 Δt 的时间间隔用以下公式计算其对应状态值:

$$RQ_i = \text{DataReceive}_{\text{Received}} / \text{DataReceive}_{\text{Should}} \tag{1}$$

式(1)中,数据接收质量 RQ_i 为在 Δt 的时间间隔内求得的节点 i 的实际接收数据量 $\text{DataReceive}_{\text{Received}}$ 与应该接收数据量 $\text{DataReceive}_{\text{Should}}$ 之间的比例,作为对节点当前相关链路情况的初步判定。

$$PQ_i = \sum_{\Delta t} \text{DataPlayback}^i_{\text{played}} / \sum_{\Delta t} \text{DataPlayback}^i_{\text{should}} \tag{2}$$

式(2)中,视频回放质量 PQ_i 为在 Δt 的时间间隔内求得的节点 i 实际回放数据量 $\sum_{\Delta t} \text{DataPlayback}^i_{\text{played}}$ 与应该回放数据量 $\sum_{\Delta t} \text{DataPlayback}^i_{\text{should}}$ 之间的比例。

$$SC_i = \frac{\sum_{\Delta t} \text{Data}^i_{\text{out}}}{\Delta t} / \max\left(\frac{\sum_{\Delta t} \text{Data}^j_{\text{out}}}{\Delta t}, j \mid j \in \text{PlayList}\right) \tag{3}$$

式(3)中,数据节点数据上传数据的能力 SC_i 为在 Δt 的时间间隔内求得节点 i 的平均上传速率 $\sum_{\Delta t} \text{Data}^i_{\text{out}} / \Delta t$ 和所有回放节点中最大的平均上传速率 $\max(\sum_{\Delta t} \text{Data}^j_{\text{out}} / \Delta t, j \mid j \in \text{PlayList})$ 之间的比例。

$$ODC_i = \sum_{\Delta t} \text{Degree}^i_{\text{out}} / \max\left(\sum_{\Delta t} \text{Degree}^j_{\text{out}}, j \mid j \in \text{PushList}\right) \tag{4}$$

式(4)中,ODC_i 为节点重要性,用在 Δt 的时间间隔内的输出度 $\sum_{\Delta t} \text{Degree}^i_{\text{out}}$ 与所有推模式传输树结构中的最大输出度 $\max(\sum_{\Delta t} \text{Degree}^j_{\text{out}}, j \mid j \in \text{PushList})$ 之间的比值表示,其中输出度越高的节点在整个分发树中的重要性越高。

$$OP_i = 1 - e^{-t} \tag{5}$$

式(5)中,OP_i 为节点 i 开始回放时间 t 后继续在线的概率,在线时间越长的节点持续在线的概率就越大。

$$SV_i = (a \times PQ_i + b \times SC_i + c \times ODC_i) \times OP_i \tag{6}$$

式(6)中,SV_i 为综合状态值,表示在 Δt 的时间间隔内,将上面的四个状态值通过运算作为节点 i 在整个网络中状态的综合判定(其中 a、b、c 为加权系数,且 $a + b + c = 1$)。

$$OSV_{\text{Severs}} = \text{DataOut}_{\text{Severs}} / \text{OutBandwidth}_{\text{Severs}} \tag{7}$$

式(7)表示在 Δt 的时间间隔内,服务器的输出状态值 OSV_{Severs}。

该节点模型从单个节点和整个分发网络两个角度出发,全面综合地进行节点状态的判定。比起现有混合传输模式中片面判定的方式,能够更好地指导两种模式的切换,达到更好地结合两种传输模式优点的目的。

3　节点缓冲模型(node caching model)

文献[8~10]将节点的数据缓冲区分为推区域和拉区域两个部分,如图 1 所示。根据在进行调度时数据在缓冲区中所处的位置选择对应的传输模式。其中拉区域又分为紧急区和非紧急区。如果紧急区中还有回放数据尚未填充,则向服务器发送紧急数据请求,拉取数据;当紧急区中的数据已经完全填充后,才以拉模式填充非紧急区中缺少的数据、通过推模式获得推区域中未填充的数据。

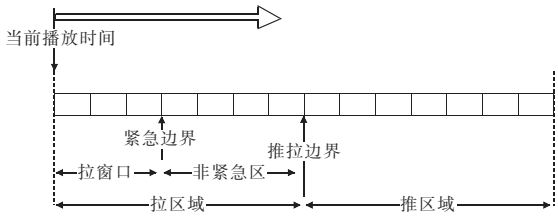

图 1　节点缓冲区结构

由于流媒体数据的不断回放,播放节点不断随着时间的增长向右滑动,整个缓冲区作为一个固定窗口也向右不断地滑动。流媒体数据按照时间片划分,每个时间片都是先进入推区域,数据的传输首选推模式,当节点状态稳定、数据接收情况好时,推送的数据可以及时有效地接收,则该时间片的数据在进入拉区域时就已经被正确填充;反之,节点状态不稳定、数据接收情况不好,数据在一定时间内未能有效地接收,则该时间片数据在进入拉区域时很可能未被完全填充,那么节点将会根据数据在网络中的资源分布情况,发出请求拉取对应数据。如果在数据进入紧急区还未完全填充,则向服务器发出紧急请求,请求拉取对应数据。

4　自适应传输模式选择算法(adaptive transmission mode selection algorithm,ATMS)

对于刚刚加入播放的节点,首先进入开始缓冲。这个过程中,整个数据缓冲窗口为拉窗口,数据的接收模式为拉模式。

在节点开始回放后,根据式(1)计算刚刚过去的 Δt 时间内的数据接收质量,设定数据接收质量阈值 ξ。若数据接收质量 RQ_i 小于阈值 ξ,说明数据接收质量较差,需要增大节点缓冲区的范围,则缓冲区的推拉边界向右调整,调整的大小为实际接收数据量 $\mathrm{DataReceive}_{\mathrm{Received}}$ 与应该接收数据量 $\mathrm{DataReceive}_{\mathrm{Should}}$ 的差值同等大小的数据时间片段,直至所有的时间片都进入拉区域;反之,若数据接收质量 RQ_i 大于阈值 ξ,则反向调整,直至非紧急区消失。

传输模式从拉模式向推模式切换的具体过程为:数据节点作为子节点,向其进行拉请求的父节点进行数据的订阅,父节点收到数据后检查子节点的订阅范围,若数据在订阅范围内则父节点直接推送数据到子节点。

由于数据节点的不稳定性,数据推送的树结构中的树干节点的状态变动将会对众多子节点产生较大的影响。因此使用式(2)~(8)等公式进行更精细的稳定性判断,对于数据接收质量较好的节点,进行进一步的处理。

$$SV_i \geqslant d \times OSV_{\mathrm{Severs}} \tag{8}$$

式(8)中的 d 为参数,如果成立则表示在 Δt 的时间间隔内数据节点 i 的状态较好,且在整个流媒体分发拓扑中为稳定的关键树干节点。为了保证该节点稳定性,该节点向服务器发送特定请求,节点数据由服务器直接推送。而服务器则实时地根据其状态变化,对推送的节点列表进行对应的删选,满足式(8)的加入列表或保持,不满足的则退出列表,并发给变动节点对应通知。

算法 1　节点的具体算法

```
if 节点是服务器的推送节点
  {
  将收到的数据,根据子节点的订阅情况进行转发;
  根据收到的数据请求,返回对应的数据;
   计算节点的状态值;
   发送当前状态值至服务器;
  }
else
  {
  将收到的数据,根据子节点的订阅情况进行转发;
  根据收到的数据请求,返回对应的数据;
  if 紧急区域已经填满
    {
     运算 Δt 时间内的数据接收质量,并进行对应的缓冲区调整;
     根据调整后的区域,发送对应的请求或者订阅消息;
     if 数据接收质量大于等于阈值
        {
        进一步计算节点的状态值;
        if 式(8)成立
          {
          请求成为服务器的推送节点;
          }
        end if
       }
     }
     end if
   else
     {
     向服务器发送对应数据的紧急请求;
     }
  end if
  }
end if
```

算法 2　服务器的具体算法

```
根据式(7)运算当前服务器的输出状态值;
将接收到的加入推送列表的请求进行处理
  {
  将收到的请求和原队列里有的节点状态值用式(8)进行判断;
  将满足要求的加入或保持在列表里,不满足的退出列表;
  发送对应的回复或通知;
  根据更新后的列表推送数据;
  if 有空余输出带宽
     {
     根据收到的紧急请求返回紧急数据;
     }
  end if
  if 有空余输出带宽
     {
    根据收到的数据订阅推送对应的数据;
     }
  end if
  if 有空余输出带宽
     {
     根据收到的数据请求返回对应的数据;
     }
end if
}
```

5　仿真分析

为了证明自适应传输模式选择算法对于性能的提高,使用开放源代码的仿真器 P2Pstrmsim(peer_to_peer stream simulator)对系统分发过程中所有节点的平均延迟和服务器负载进

行仿真研究。这里的视频速率设为 100 kbps,系统观测时间为 300 s,系统网络结构为 30 个路由节点随机生成的网状结构,两台 CDN 代理服务器随机接入其中的两个路由节点,其上行带宽 10 Mbps,下行带宽 2 Mbps,用户节点一共有 1 000 个,其中 600 个观测的时候就已经在观看节目,另外 400 个节点在观测时间为 150 ~ 160 s 的时候开始观看,用户节点的带宽分别有 15% 的概率是上行带宽 400 kbps,下行带宽 1 Mbps,35% 的概率是上行带宽 200 kbps,下行带宽 400 kbps,50% 的概率是上行带宽 200 kbps,下行带宽 200 kbps,每个节点的缓冲区大小为 20 s。

将提出的自适应传输模式选择调度(ATMS)对比传统的 P2P 调度和 iGridMedia 的延迟,结果如图 2 所示。在数据节点较为稳定的情况下,ATMS 的传输延迟最小,iGridMedia 次之,P2P 最长,而且在数据节点数量剧增后,明显发现平均延迟的增加上,ATMS 和 iGridMedia 增加较小,平均增加 800 ms,而 P2P 增加较大,平均增加 2 400 ms。而在数据节点激增的 150 s 至稳定前的 220 s 之间,ATMS 和 iGridMedia 延迟较小而 P2P 较大,其中 ATMS 比 iGridMedia 的延迟略大一点,在 160 ~ 200 s 平均大了 240 ms。这是由于节点较为稳定的时候,ATMS 和 iGridMedia 中大量节点和数据的传输采用了推模式,而 P2P 中使用的是拉模式,在延迟上,尤其是离数据源节点跳数较多的节点,延迟的增加是累积的。而 ATMS 比起 iGridMedia 在节点稳定的情况下,数据缓冲窗口的拉窗口减小,推窗口增加,以此达到加大数据被推送的比例,减小数据的传输延迟。此外,ATMS 中实时选择传输过程中,综合状态较好、对系统分发较为重要的节点由服务器直接推送数据,以此达到从整体角度减小一定传输延迟的作用。但是由于 ATMS 中对于服务器推送节点的删选,与服务器负载相关,因此出现了 ATMS 在节点激增时比 iGridMedia 的延迟略大的情况。

ATMS 对比传统的 P2P 和 iGridMedia 的服务器负载,结果如图 3 所示。在数据节点较为稳定的情况下 P2P 负载最小,平均 4 Mbps;ATMS 次之,平均 12.1 Mbps;iGridMedia 最大,平均 13.3 Mbps。而在数据节点数量剧增后,P2P 依旧最小,而 ATMS 变成比 iGridMedia 略大,在 160 s 时大了 0.04 Mbps。分析其主要原因,由于 P2P 中节点间的数据分发主要靠拉模式传输,数据父节点的选择上对于服务器的依赖性较小,仅在某些情况下向服务器发送一些紧急请求,因此服务器负载较小;而 ATMS 和 iGridMedia 的服务器除了返回紧急请求的数据,还担负着更多的数据分发任务,在服务器负载情况较好时,ATMS 会根据情况选择性地只对分发网络中起到较为重要作用的节点进行数据的推送,而如果负载较重,则会减少推送节点的数量,因此在稳定的情况下,负载较 iGridMedia 略轻,且在节点的多少对于负载影响较小,前后均值差为 0.03 Mbps。但是在节点剧增的过程中,由于向服务器发送的紧急请求数量的增加,这时候对于负载的减轻起到的作用较网络稳定情况时就比较微弱了。整个过程中 ATMS 就是通过对服务器推送节点的挑选和数量的控制,达到减小服务器负载的目的。

6　结束语

本文提出的自适应传输模式选择算法(ATMS)基于新的节点状态模型。该自适应调整节点数据的传输模式通过对数据接收质量和综合状态值两个值作出对应的调整,达到在具体节点和整个网络的数据分发中减小延迟的目的。仿真表明:ATMS 在网络拓扑结构较为稳定的情况下,对稳定节点进行数据推送和对网络中重要节点进行数据推送,达到提高分发效率的目的,而在网络拓扑发生剧烈变化的情况下,通过减少服务器直接推送的节点,降低分发效率,优先保证紧急请求的数据传输,整个过程通过一定程度地控制服务器负载,保证了系统的可扩展性。这种传输模式的自适应选择较现有的流媒体调度模式,能够更好地达到“在高效分发数据的同时保证数据的接收质量”的目的。但是,由于研究的复杂性,对于具体模型的研究仍有待深入。另外,后续工作上还需要结合具体的数据选择算法,达到更良好的数据调度效果。

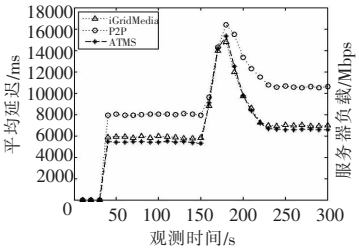

图 2　延迟对比

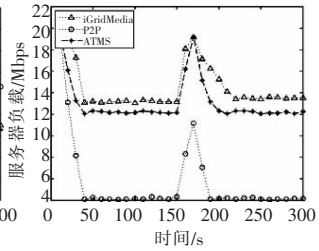

图 3　服务器负载对比

参考文献:

[1] 秦丰林,刘琚. P2P 网络流媒体关键技术[J]. 电子信息学报,2011,39(4):919-927.

[2] 郝敏,苗彦超,周应超,等. 视频流调度策略的研究[J]. 计算机工程,2004,30(24):141-143.

[3] MANSY A, AMMAR M. Analysis of adaptive streaming for hybrid CDN/P2P live video systems[C]//Proc of the 19th IEEE International Conference on Network Protocols. 2011: 276-285.

[4] LUO Jun. Practical algorithm for minimum delay peer-to-peer media streaming[C]//Proc of IEEE International Conference on Multimedia and Expo. 2010:986-991.

[5] YIN Hao, LIU Xue-ning, ZHAN Tong-yu, *et al*. Design and deployment of a hybrid CDN-P2P system for live video streaming: experience with LiveSky[C]//Proc of the 17th ACM International Conference on Multimedia. New York: ACM Press, 2009:25-34.

[6] ZHANG Meng, LUO Jian-guang, ZHAO Li, *et al*. A peer-to-peer network for live media streaming using a push-pull approach[C]//Proc of the 13th Annual ACM International Conference on Multimedia. New York: ACM Press, 2005:287-290.

[7] 臧运港,陈光喜. 基于 P2P 和 CDN 的混合流媒体分发模型及分析[J]. 计算机应用研究, 2010,27(6):2208-2214.

[8] ZHANG Meng. iGridMedia: providing dela-guaranteed peer-to-peer live streaming service on Internet[C]//Proc of Global Telecommunications Conference. 2008:1741-1745.

[9] ZHANG Meng, SUN Li-feng, XI Xiao-lu, *et al*. iGridMedia: the system to provide low delay peer-to-peer live streaming service over Internet [C]//Proc of IEEE Global Telecommunications Conference. 2008: 12-24.

[10] 任浩,王劲林,尤佳莉. 基于数据接收质量的 P2P 流媒体自适应推拉调度算法[J]. 微计算机应用,2011,30(4):1-8.

[11] MAGHAREI N, REJAIE R, GUO Yang. Mesh or multiple-tree: a comparative study of live streaming approaches[C]//Proc of the 26th IEEE International Conference on Computer Communications. 2007: 1424-1432.

[12] LU Zhi-hui, GAO Xiao-hong, HUANG Si-jia, *et al*. Scalable and reliable live streaming service through coordinating CDN and P2P[C]// Proc of the 17th International Conference on Parallel and Distributed Systems. 2011:581-588.