可交互流媒体服务中的应用层组播技术研究

陈建忠 李文中 司春峰 陆桑璐 陈道蓄

(南京大学计算机系计算机软件新技术国家重点实验室 南京 210093)

摘 要 在开放、动态的网络环境中,网络构件致力于如何有效地整合和共享多样化资源。近年来,流媒体应用在Internet上日趋流行,由此带来了资源共享和节约带宽消耗等一系列挑战性的问题。应用层组播被认为是解决大规模流媒体应用网络拥塞的一种有效技术。然而,流媒体交互操作会引起组播树的频繁重构,从而降低系统性能。提出了一种支持可交互操作的应用层组播树构建协议ISMT(Interactive Streaming Multicast Tree),可以降低用户响应延时和改善系统的扩展性。通过仿真实验验证了ISMT协议的有效性。

关键词 可交互,流媒体,IP组播,应用层组播

Research on Application Layer Multicast in Interactive Streaming Media

CHEN Jian-zhong LI Wen-zhong SI Chun-feng LU Sang-lu CHEN Dao-xu

(State Key Laboratory of Novel Software Technology, Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China)

Abstract Internetware tends to resolve the integrating and sharing of various kinds of resources in open, dynamic and changing Internet environment. In the recent years, streaming media applications becomes more and more popular on the Internet, which introduced challenging issues such as resource sharing and bandwidth saving. Application layer multicast has shown to be efficient in reducing network traffic for large-scale multimedia applications. However, interactive operations such as VCR may cause frequently reconstruction of multicast trees, which will degrade the system performance. We proposed a protocol called ISMT (Interactive Streaming Multicast Tree) to construct and maintain interactive application layer multicast trees, which aims at reducing user response delay and enhancing system scalability. The effectiveness of the ISMT protocol is validated by simulations.

Keywords Interactive, Streaming media, IP multicast, Application layer multicast

1 引言

组播是Internet上支持流媒体传输的有效途径,诸如文献[12,23]都提供了基于组播的流媒体解决方案,但它们都是在假设流媒体服务请求相对同步和稳定的前提下,最大可能地降低服务器端带宽占用、网络资源利用率以及请求响应时延。网络带宽的可使用量和个人计算机处理性能的持续提高,普通客户的流媒体服务需求已不再局限于被动地接受服务器提供的数据流,类似于直播和VoD (Video on Demand)的方式,各种VCR交互操作(如暂停、快进、跳转、后退等)的流媒体运用应运而生。

用户VCR操作的异构性(操作类型不同)和异步性(用户在不同的时间段内提出的两个同类型操作)使服务器对数据流的调度和维护变得异常复杂,很多实际系统都采用单播的方式提供服务。服务器端的带宽消耗随着接入用户的数量呈线性增长,很容易成为系统性能瓶颈。当用户数量剧增时,流数据消耗的带宽急剧增长,网络拥塞和端到端时延会急剧恶化。为降低服务器和网络负载,一个不彻底的解决方案就是分散服务,通过多个副本服务器或是一系列存放了部分源服务器文件副本的代理来提供服务,这并没有降低全局的网络资源使用量,而且其中任意一个副本服务器或是代理仍有可能成为系统瓶颈。所以,基于单播的系统缺乏可扩展性,只能部署于小型网络。目前,组播策略的利用有效地降低了服务器和网络负载。针对流媒体应用,一些研究采用客户端缓存技术或应急通道技术与基本组播技术的结合,提出了支持交互操作的算法。文献[24]采用客户端缓存技术,试图在不增加服务器带宽消耗的条件下支持交互功能。文献[25]在系统中引入了应急通道技术。前者受限于缓存的尺寸,一般只能支持小范围的跳转等操作,后者因为是在服务器端开辟临时传输通道,则会减少服务器可用带宽,降低系统整体性能。

基本组播技术包括IP组播[1,2]和应用层组播。IP组播应该是降低服务器和网络负载的最有效手段,但其部署仍受

到稿日期:2008-01-30 本文受国家高技术研究发展计划863项目(No. 2006AA01Z199),国家重点基础研究发展计划973项目(No. 2006CB303000),国家自然科学基金(No. 90718031, No. 60721002, No. 60573106)资助。

陈建忠 硕士研究生,研究方向为并行处理和分布式计算,E-mail:derychen@gmail.com;**李文中** 讲师,研究方向为并行处理和分布式计算;**司春峰** 硕士研究生,研究方向为并行处理和分布式计算;**陆桑璐** 教授,博士生导师,研究方向为并行处理和分布式计算;**陈道蓄** 教授,博士生导师,研究方向为并行处理和分布式计算。

到很大的限制[3,4]。本文采用了应用层组播的解决方案。应用层组播是基于P2P框架的，现有的研究大都针对节点的动态性，构建有效的路由算法，从而提高系统的可扩展性和网络资源利用率。但这些研究都没有将可交互作为主要目标，由这些协议构建的流媒体服务大都只能提供非交互的流媒体直播服务。

针对客户的交互操作给应用层组播树构建带来的影响，我们提出了一种可交互的应用层组播树构建协议——ISMT (Interactive Streaming Multicast Tree)。ISMT是一个适用于支持可交互流媒体服务的应用层组播树构建协议，包括节点加入、拓扑的维护、节点退出及拓扑的修复等部分。ISMT的核心思想在于：1) 利用节点的缓存满足部分交互操作请求，又能作为资源为其他节点服务；2) 用树优先的方法构建组播树，在构建过程中，将节点缓存的内容、节点的空余带宽资源以及节点间的距离作为父节点选择的依据；3) 采用祖先和一些子孙之间建立临时连接的机制，加快节点的加入过程，减少用户交互操作以后的响应延迟。

第2节介绍了有关构建组播树和交互式流媒体方面的相关工作，第3节给出了可交互流媒体服务的详细描述，接着给出了基于ISMT的流媒体系统的简单框架，最后针对ISMT做了一些相关的性能模拟。

2 相关工作

作为IP组播技术的一种替代技术，应用层组播技术的研究在近年来受到广泛关注，研究人员提出了一些应用层组播方案，如NICE[23]，Overcast[21]，同时出现了一些不成熟的产品，如SpredIt[22]，CoopNet[19]以及ZIGZAG[12]。

典型的组播树构造算法包括基于源节点的最短路径树(Shortest Path Tree，简称SPT)和Steiner树(Steiner Minimal Tree，简称SMT)[13,14]。SPT的优点在于它使源节点到每个目的节点的时延最小，而且易于计算和实现。DVRMP，MOSPF，CBT，PIM-SM[15-18]等协议都以构造SPT为协议目标。若树中除源节点外的所有节点都是接收数据的目的节点，Steiner树将演变成最小生成树(Minimum Spanning Tree，简称MST)。SMT试图提供代价最小的优化树，但SMT的优化计算是一个NPC[14]问题。

与网络路由器不同，终端主机节点的出口带宽有限，限制了节点的报文转发能力，因此节点带宽是影响应用层组播协议的一个主要因素。从QoS角度，应用层组播路由协议研究的核心问题是：针对具体的应用需求，构造受节点度约束的高性能组播树。

Yoid[20]最早用树优先法构建应用层组播树的协议，它为实现覆盖组播提出了完整的框架，其本质是一个协议集。Yoid在系统中生成两种拓扑：控制mesh和组播树。

在以提供直播和点播流媒体服务为目标的CoopNet[19]系统中，采用集中式的组播树构建协议，根节点中记录了所有节点的信息。当新节点加入时，由根节点根据树中节点的剩余带宽，在组播树中逐层下溯寻找一个可用节点，作为新节点的父节点为其传送数据。对于交互操作，CoopNet试图采用流分解编码技术(Multiple Description Coding)，在组播树中寻找若干个节点进行协作，同时为请求节点提供数据。

P2P流媒体系统已经在工业界和理论界都是一个研究的热点，和传统的C/S模式相比，它可以充分利用终端客户的存储和处理能力来提高系统的可扩展性，就如NICE[23]和ZIGZAG[12]，但是它们都设计用于大规模流传输的应用层组播协议。NICE采用层次式的簇聚集算法，较高层簇中的节点为下层节点的簇头，负责数据的组播。簇内可以根据需求采取独立的组播方案。ZIGZAG也是基于层次式的解决方案，但将组成员的管理和组播树的构建独立开来，簇头不负责本簇内的数据转发服务。

3 可交互应用层组播协议

3.1 ISMT目标

大部分应用层组播研究主要从提高网络覆盖效率、提高可扩展性出发，但都未考虑可交互功能的支持。本文提出一种支持可交互功能的组播树构建协议，其目标是：1) 支持可交互流媒体服务；2) 较小的响应延迟，包括新用户的加入延迟和用户进行交互操作后的响应延迟；3)用树优先的方式构建组播树，要求其有较高的组播树性能和网络覆盖率，并尽量减少底层链路的重复数据流，提高网络利用率；4) 提高系统的可扩展性，使系统可为更多的用户服务。

与已有的组播树构建方法相比，可交互操作会对数据组播树的构建产生一定的影响。首先，节点缓存的内容会影响父节点的选择，如P2CAST[5]。在组播树中，由父节点向其子节点传输流媒体数据，这个数据一般是由其接收到的数据立即进行复制并转发的。而在支持可交互功能的组播树中，父节点传输给子节点的流媒体数据一般是父节点在若干时间以前接收到并暂存在其缓存中的内容。而且，由于系统中用户进行交互操作后，可能会导致子节点需要的内容在其父节点中并未缓存的情况，子节点就必须重新选择父节点。因此，节点缓存内容成为父节点的选择因素之一。

其次，交互操作导致节点的频繁迁移。在普通的组播树中，节点的动态迁移通常是由于节点的失败、退出造成的，或者在有些系统中，会进行周期性的组播树结构优化来提高组播树的性能，也会导致节点的迁移。在支持可交互功能的组播树中，用户进行的交互操作也可能导致一个节点重新寻找其合适的父节点。因此，节点的迁移将更加频繁。

再次，端到端延迟不作为考虑因素。在很多的组播应用场景中，都将发送源到接收端的传输时延作为系统性能的衡量指标，诸如传统的视频会议和网上直播服务。本文的目标是构建适用于可交互流媒体服务的应用层组播树，每一个节点都是直接从其父节点中取得其所需要的数据，在同一时刻每一个用户所观看的媒体内容也不必同步，因此端对端时延不必作为建树的性能参考。

由于以上3个因素，支持可交互功能的ISMT组播树将有如下特点：1) 窄而高的树，即节点度有限、最长路径长度不限的组播树。在流媒体系统中，数据传输速率一般较大，而普通网络用户的出口带宽有限，一个节点通常只能为有限的其他节点服务，即其子节点数有限。当系统中的用户数越多时，最大路径长度不可避免地增加。2) 缓存中的内容和节点间的距离同时作为父节点的选择因素，前者是由支持可交互功能的特点决定的，后者是由提高组播树性能的目标决定的。在应用层组播树中，将物理距离较近的节点尽量聚集在一起，能提高组播树的网络覆盖率，减少底层网络链路的报文数量。

3.2 ISMT 组播树结构

ISMT 协议采用树优先的方法构建组播树。相比于网优先方法，树优先中的客户可以设计得尽量简单，且容易对树进行直接控制，例如节点的出度、根据缓存内容进行父节点的选择以及在节点失败时将结构变动限制在较小的范围。网优先协议需要所有客户端运行一个分布式算法来维护整个 mesh，还需要维护相应的路由算法。另外，频繁的节点加入和退出、交互所引起的树的重构操作会增加 mesh 的维护工作，以致影响整个系统的性能。

ISMT 的核心思想在于：1）利用节点的缓存满足部分交互操作请求，又能作为资源为其他节点服务；2）用树优先的方法构建组播树，在构建的过程中将节点缓存的内容、节点的空余带宽资源以及节点间的距离作为父节点选择的依据；3）采用在祖先和一些子孙之间建立临时连接的机制，加快节点的加入过程，减少用户交互操作的响应延迟。

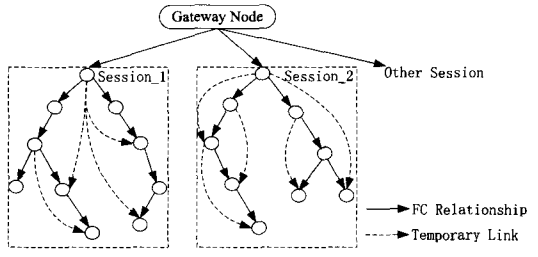

图 1　ISMT 结构示意图

ISMT 的组播树结构如图 1 所示。在组播树中，每个节点都有一定的缓存和空余带宽。缓存中可以存放部分流数据供用户观看，也可以为其他节点提供流媒体服务。我们将有直接数据连接的两个节点称为邻居，其中提供数据节点为父节点，接收传输数据的节点为子节点。

在由 ISMT 构成的组播树中，有一个门户节点，其地址是广为人知的。这个节点存有服务器所提供的全部流媒体文件，是系统中所有流数据的源，并维护了组播树的部分信息。由于普通的缓存限制，一个组播会话中的根只可能缓存部分流数据，当有额外的数据请求时，门户节点会开辟一个新的组播会话。这样，门户节点会维护多个组播树的信息。

在同一组播树中的节点间，其缓存中的内容是相近的。当一个新节点向门户节点发送数据请求时，门户节点将根据不同组播树中缓存的内容，将此节点分配到某个合适的组播树中，新节点通过一系列的信息交互定位其父节点并从其接收数据。父节点的选择标准系根据其有无被请求内容、有无空余带宽以及与该节点的距离作为衡量的标准。在应用层组播树中，相邻节点在底层链路的距离越近，则说明组播树的覆盖网络拓扑越符合底层拓扑结构，组播树的性能也就越好。在 ISMT 中，由于底层链路不可知，我们使用发送和接收数据的延迟来粗略估计节点间的距离。

流媒体传输所需的带宽资源消耗比较大，由于普通节点带宽的有限性，限制了 ISMT 系统中节点的出度。随着用户数量的增加，组播树的高度也会随之至少对数级地增加。若采用常用的父节点定位策略，即层层下溯，以节点间的延迟为衡量标准，寻找较近节点的方法，节点请求的响应延迟与组播树高度成正比。在 ISMT 中，我们在节点与其一部分子孙节点中，接入一些临时连接，使下溯过程在某些情况下可以跨越若干层，加快父节点的定位过程，减少响应延迟。

3.3 节点维护的信息

门户节点作为系统的入口，需要一些维护全局的组播会话信息。信息的结构如表 1 所示。

表 1　门户节点中的组播会话信息表结构

组播会话 Mid_i	文件 Fid_i	根节点地址 $Root_i$	起始时间 T_{start_i}	起始位置 D_{start_i}	发送速率 X_i
1	file5	202.119.36.54	16:06	0:00	1(播放)
2	file5	210.32.75.7	16:16	0:00	1(播放)
3	file5	210.28.131.159	16:48	35:15	2(快进)
4	file16	218.16.100.66	17:23	10:00	−1(回退)

在 ISMT 协议中，每个组播会话有一个唯一的 M_ID，这个组播会话中的所有节点，请求的都是同一个媒体文件，用 F_ID 表示。在同一时刻，一个文件可能对应系统中的若干组播会话，这些组播会话传输可以对应文件的不同片段。在门户节点中，并不记录系统中的所有节点的信息，只记录该组播树的根节点地址 Root。起始时间 T_{start} 为门户节点建立该组播会话的实际时间，起始位置 D_{start} 表示在该会话建立时传输的数据位置，可以该点在影片中的时间为单位。发送速率 X_i 即数据的传输速率，以正常播放速率为一个单位，这个参数通常与根节点当前 VCR 状态有关。

当一个新节点加入系统时，门户节点将根据信息表中的内容搜索一个合适的组播会话，将此组播树的根节点地址发送给新节点。如果现有的组播会话都不能满足新节点的请求，门户节点将新建一个组播会话，将此新节点作为该会话的根，并生成相应的组播会话记录，插入到门户节点的信息表中。当数据传输完毕或是节点离开，则撤销该组播会话，并从信息表中将相应记录删除。在某些情况下，门户节点返回的组播树根到请求节点的时延可能会很大(物理拓扑距离相距甚远)，这样可能要在为新节点开辟一个新的组播会话和容忍大时延之间做平衡。

当新节点通过门户节点获取一棵组播树的根节点地址后，就会通过一系列的查询交互过程在这个树中寻找一个合适的父节点。因此在普通节点中，也需要维护一些信息，这些信息的结构如表 2 所示。

表 2　普通节点维护的信息

父节点地址	61.158.95.63
子节点列表	219.45.172.85,221.10.124.34
临时连接列表	211.97.68.211,202.201.94.27,218.244.225.180
临时连接缓存	218.5.191.126,210.28.37.15,……
候选父节点集合	61.158.93.67,61.158.94.34,……
缓存信息	(23,45,92)
路径	202.119.36.54 * 219.149.76.19 * 61.158.95.63

在 ISMT 组播树中，除根节点外，每个节点都有一个父节点，并从这个父节点获取数据。每个非叶节点都有 0～d 个子节点，d 可以设置为一个常数，通常由节点的出口带宽决定。在 ISMT 组播树中，一些节点还会与它的一部分子孙节点建立临时连接，此临时连接的个数也可以人为设置。临时连接缓存中的节点作为临时连接的候补，当列表中的节点个数不足时，则用缓存中的候补节点补充，在每个节点中，都会记录自己的数据传送路径信息，即从该组播树根节点到自己所经过节点的列表。

3.4 新节点加入

在P2PIS协议中，当一个新节点加入时，将会首先向门户节点发送加入请求，得到一个合适的组播树根节点地址，然后从根节点开始，在组播树中找到一个合适的父节点。加入过程是有目标地选择一个符合下列条件的父节点：1）传送时延较小的节点，即父节点和子节点在底层拓扑上相距较近；2）父节点缓存有子节点需要的内容；3）父节点必须有满足新节点的服务的空闲带宽。

如图2所示，新节点K询问门户节点后，得到相应组播树的根节点A的地址，然后向A节点发送请求，得到A的所有子节点和临时连接邻居的地址B，G，E，H和J。在这些节点中，H有被请求内容且到K节点时延最小，因此被作为下一轮请求的节点。K向H发送请求后，得到I的地址，且I中有被请求的内容且其时延比H小，则I将被作为下一轮请求的节点。K向I请求后，得到J的地址，但J的时延比较大，因此K向I发送数据传送请求。因为I还有空余服务能力，因此将接收K作为其子节点。

在新节点的加入过程中，节点需要和门户节点及组播树中相关节点进行通信，此过程涉及的协议消息见表3。

表3 加入过程涉及的协议消息

MessageType	MessageFormat
Boot	(Boot, F_ID, POS, MODE)
BootReply	(BootReply, ROOT)
JoinReq	(JoinReq, TimeStamp, POS, MODE)
JoinReply	(JoinReply, TimeStamp, SpareDegree, NeighbourList)
JoinTest	(JoinTest, TimeStamp, POS, MODE)
TestReply	(TestReply, TimeStamp, SpareDegree)
Join	(Join, POS, MODE)
JoinOK	(JoinOK, PATH)
JoinRej	(JoinRej)

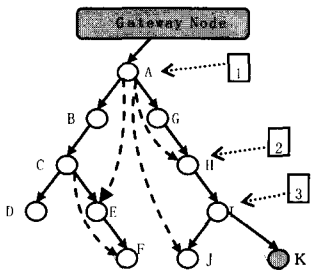

图2 新节点加入过程示意图

1）Boot/BootReply

当一个新节点向门户节点发送Boot消息请求数据 D_{req} 时，门户节点在其维护的组播会话表中寻找符合式(1)的组播会话，并将其根节点的地址返回。

$$\max(D_{start\_i}, D_{now\_i} - C_{buf}) < D_{req} < D_{now\_i} \quad (1)$$

其中 $D_{now\_i} = D_{start\_i} + (T_{now} - T_{start\_i}) * X_i$

如果没有合适的组播树，则新节点将会向门户节点发送Join消息，门户节点会新建一个组播会话，将此节点作为该组播会话的根节点，并将此组播会话信息接入信息表中。

2）JionReq/JoinReply

当一个节点收到JoinReq消息时，它会将其维护的直接子节点列表和临时连接列表中的所有节点地址都返回，并返回其空余出度。JoinReply消息中的时间戳是JoinReq中的时间戳的拷贝，以供发送请求的节点得到两节点之间的延迟的粗略估计。

若节点接收到另一节点的JoinReq消息，往往意味着这个新节点将成为自己的子孙节点，因此它会以一定的概率将此节点更新到自己的临时连接列表中，和该节点建立临时连接。

3）JoinTest/TestReply

当一个节点收到JoinTest消息请求数据 D_{req} 时，它将查看自己的缓存是否存在请求数据 D_{req}。如果存在被请求内容，则在TestReply中返回自己的空余出度；如果 D_{req} 不存在，则该节点不对JoinTest消息做任何响应，这样可以减少网络无效报文开销。TestReply消息中的时间戳为JoinTest中的时间戳的拷贝。

当发送测试请求的节点收到某个节点的TestReply消息时，即表明该节点有自己请求的内容，并可以根据TestReply中的时间戳和当前时间计算出被测试节点和本节点之间的延迟。

4）Join/JoinOK&JoinRej

当一个节点收到Join消息时，它会查看有无空余出度并试图为发出请求的节点建立服务，以被请求速率发送被请求的数据。若请求得到满足，则返回JoinOK消息；否则发送JoinRej消息。

JoinOK消息中的路径是自身的路径信息附上自己的地址，作为子节点的路径。当发出的请求的节点接收到JoinOK消息时，它将消息中的路径信息保存到自己的路径信息表中。

综上所述，加入算法使一个新节点通过遍历组播树的一部分试图寻找一个能提供较好转发性能的父节点，该算法终止于一个叶节点或者一个中间节点。由于加入了临时连接，该算法的收敛速度可以加快。

3.5 节点间连接的维护

用ISMT协议构建的组播树中，有两类连接：父节点与子节点间的数据连接、祖先与其部分子孙节点的临时连接。由于ISMT中的数据连接传输是连续的流媒体，因此不必用额外的消息在保持父节点与子节点之间的联系，即一旦发生故障，数据连接会发现数据发送失败或是接收超时。而由于祖先与子孙节点间的临时连接并不进行直接的数据传输，因此不能感知发生的变动。因此，ISMT中每隔一段时间用Ping/Pong消息来维持临时连接，并保证临时连接的两端是子孙关系。消息格式为PING (LocalIP)，PONG (NeighbourList)。

Ping消息是由祖先节点定期向其临时连接的子孙节点发送的，在其中会给出自己的地址。当子孙节点接收到Ping消息时，它会在自己的路径信息中查找是否存在该节点。若存在，则说明两者的子孙关系依然成立，返回Pong消息；若不存在，则说明由于拓扑变动等原因，子孙关系不存在，它将不对Ping消息做响应。

当祖先节点接收到Pong消息时，它将返回的节点列表加入到自己的临时连接缓存中。若为收到相应的Pong报文，则认为此子孙节点已不存在，并将其从临时连接列表中删除，同时从临时连接缓存中取出一个新地址，尝试与其建立临时连接。

3.6 节点的退出及组播树的修复

在ISMT组播树中，节点退出组播树有两种情况：主动退出或发生故障。当组播树中的某一个节点退出后，它的子树

就和组播树的其他部分分离，所以子树必须执行修复过程。另外，组播树中节点的服务质量可能由于链路状况的恶化而得不到满足，此时也应当对组播树进行适当调整。

1)节点正常退出

当某节点欲退出组播树时，它将向父节点和所有子节点发送 Leave 消息，Leave 消息格式见表 5。当父节点收到该节点的 LeaveParent 消息后，将从其子节点列表中删除该节点。

表 5　节点退出涉及的协议消息

MessageType	MessageFormat
LeaveParent	(eaveParent, LocalIP)
LeaveChild	(LeaveChile, LocalIP, SequentialChildrenList)

欲退出的节点向子节点发送的 LeaveChild 消息中，除了给出自己的地址外，还会发送子节点列表。子节点是按照当前数据发送点的位置来排列的，数据发送位置大的子节点放在列表前，即与父节点缓存内容之间偏移较小的节点优先于偏移较大的节点。这样排列的原因在于，一般来说，偏移较小的节点会有偏移较大的节点原本正从父节点接收、由于父节点的离开而需要重新请求的数据。

当子节点接收到 LeaveChild 消息后，将会查找自己在消息中的子节点在序列表中的位置。若处于列表的第一个，则将父节点从 Path 中删除，沿 Path 向上修复过程；否则，则将父节点从 Path 中删除后，将 LeaveChild 消息中的子节点列表中排在自己之前的节点依次附到 Path 之后，然后从最后一个节点开始沿 Path 向上执行修复过程。

修复算法其本质上就是一个不断上溯申请加入的过程。若根节点都未接收请求，则执行新节点加入算法。算法的目的在于恢复过程中使子节点能尽快找到一个父节点，避免太长的延迟影响子节点的服务质量。

2)非正常退出

当节点或底层链路发生故障后，它并未发送 Leave 或 LeaveChild 消息就退出组播树，不再接收或发送流数据。

当父节点发现数据发送失败，则认为其子节点非正常退出，将其从子节点列表删除。当子节点从父节点接收数据超时，则也认为其父节点可能已非正常退出。则会从父节点开始沿 Path 向上执行修复过程。

3)网络状况的恶化

ISMT 客户通过持续监测输入流来监测网络故障。当输入流的质量低于某个阈值时，客户端就发送 Test_Error 给其直接父节点，同时可以启动一个定时器。

a)无应答消息。直到定时器超时也未收到应答消息，客户重发 Test_Error 并启动定时器，若重复 3 次都未在定时时间内收到应答，则节点沿 Path 路径向上进行修复。

b)收到 Network_Recovery 消息。当节点收到来自直接子节点的 Test_Error 消息后，节点试图提高流的传输质量。若子节点持续发送 Test_Error 消息，则认为它们之间的链路已经不能满足传输需求。节点发送 Network_Recovery 消息给子节点，子节点执行修复算法。

c)接收 Wait 消息。当节点执行修复算法或加入算法时，节点向其代表的子树的所有其他节点发送 Wait 消息并触发其启动 Test_Error 消息。

4　用 ISMT 实现可交互流媒体系统

4.1　系统结构

ISMT 本质上是一个应用层组播协议，其设计目的是实现一个可交互的流媒体服务系统。系统由许多自主性的节点构成，系统(节点)结构如图 3 所示。

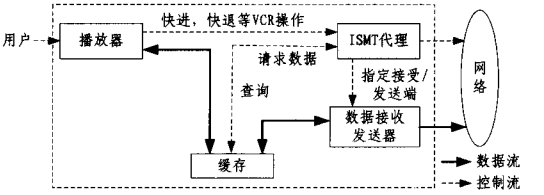

图 3　ISMT 支持的系统结构图

ISMT 代理是实现 ISMT 协议的模块，用于和系统中的其他节点进行信息交互。当缓存向代理请求数据时，ISMT 会根据当前的状态执行一定的过程和其它节点通信，以获取相应数据。同时，代理还负责为其他节点服务，响应其请求。

流媒体数据的接收和发送由相应的收发器模块完成，收发器模块与父子节点建立直接的数据连接，并负责与缓存进行直接数据传输。

4.2　VCR 操作对缓存的影响

为简化本地数据检索和存取，本系统采用了一种类似于环形列表结构的缓存。假设缓存分成 N 块，每块对应了时间长度为 t 的媒体片段，则整个缓存的容量为 $C_{buf} = N * t$。

VCR 操作对缓存的影响是巨大的，主要有以下 3 种可能情况：

a)VCR 操作所需的内容在缓存中存在

这通常是由于小幅度的 VCR 操作所致，如短时间慢退、小范围跳转等。此时，ISMT 无需做任何动作，数据收发器仍然继续接收父节点缓存内容并向子节点发送请求数据。

b)缓存溢出

这通常是由于暂停、慢放持续时间较长所致。此时 ISMT 代理将执行退出组播树操作，向父子节点发送退出消息，待用户取消暂停时加入组播树。

c)缓存中无播放所需的内容

如大范围跳转、长时间快进等，将会导致这种情况。此时，缓存会向 ISMT 代理请求数据。ISMT 代理收到请求后，将离开现在所在组播树，重新执行加入过程，向门户节点请求数据。

5　性能模拟

5.1　模拟环境

我们用 gt-item[7] 在 ns2[6] 上生成了 1000 个模拟 Internet 拓扑结构的节点，每个节点代表一个路由器。在每个路由器节点上随机生成 5～10 个叶节点，代表终端主机。指定一个主机为系统的门户节点，在其余节点中随机抽取若干，以符合平均到达率 λ 的泊松分布加入系统。为实验方便，我们只在门户节点上放置了一部影片，这并不会影响实验的有效性，因为从单个影片的实验中得到的数据很容易扩展到多个影片的情况。影片的长度为 100min。假设所有节点用户缓存大小相同，并限定每个节点的出度最大值为 2(终端主机的性能不是很高，所以假设的出度比较小)。

ISMT 的设计目标是在支持可交互功能的前提下，提高系统可扩展性、提高组播树的性能以及减少用户操作的响应延迟。因此，我们将从门户节点的可扩展性、普通节点的可扩展性、组播树的压力度以及 VCR 操作的响应延迟等几方面来进行分析和验证。

5.2 门户节点的可扩展性

门户节点作为系统的入口，当新节点加入系统或已加入的节点在 VCR 操作后发现父节点无法提供所需要的数据时，都会向门户节点发送查询请求。当已存在的组播树都无法满足请求时，门户节点会创建一个新的组播会话。因此，门户节点需要维护所有组播会话信息，并为每个组播会话的根节点提供数据。

假设门户节点一共维护了 n 个组播会话，对每个节点的查询请求进行处理时，需要在这些组播会话中挑选一个合适的组播会话（见式(1)），这个算法的时间复杂度为 $O(N)$。门户节点为每个组播会话维护一个如表 1 所示结构的信息，因此门户节点内所需的空间复杂度也为 $O(N)$。门户节点还需要为每个组播会话的根节点以单播的方式传输数据。假设每个流的速率大小基本相同，本质上与流媒体文件的编码速率有关。因此，门户节点用于传输数据而使用的带宽也为 $O(N)$。在 ISMT 系统中，用于构建组播树而传递的协议消息大小要比流媒体数据小得多，可以忽略不计。

综上所述，门户节点的可扩展性与系统中的组播会话个数有关。在模拟实验中，我们分别考察了用户数量（以新用户的平均到达率 λ 衡量）以及节点的缓存大小对门户节点中组播会话个数的影响。

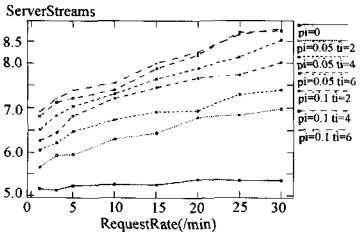

图 4 用户 VCR 操作对门户节点带宽的影响

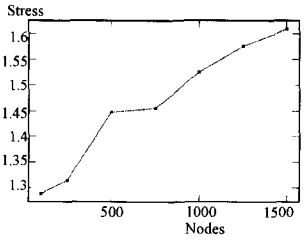

图 5 组播树节点对压力度的影响

图 4 显示了当缓存大小固定为 20min 时，在不同的 VCR 操作频度下门户节点中组播会话的情况（p_i 为各 VCR 操作的概率、t_i 为操作的持续时间）。从图中可以看出，首先，在固定 VCR 操作频度下，门户节点所需要创建的组播树数量受 λ 的影响不大。其次，当 VCR 操作频度加大时，门户节点的会话个数也会随之增长，但其增长幅度并不大。

5.3 普通节点的可扩展性

由于应用层组播树中组播树的建立与维护、数据的转发都依靠应用层节点完成，因此协议要通过节点保存部分（或全部）其他节点的状态信息来维护网络的完整性。以树优先法构造的组播树中，普通节点的可扩展性通常和节点的平均控制负载及其节点度相关。

由于 ISMT 协议在构造组播树时限定了节点的出度，因此树中节点出度为常数，这个常数是由节点的出口带宽与流媒体文件的编码速率的比值决定的。

ISMT 中，普通节点需要维护组播树中其他一部分节点的信息，包括其父子节点的信息、路径信息以及一些临时连接的信息。其中，父子节点的数量属于常数级，临时连接的数量也可以控制在常数范围内。因此，ISMT 中普通节点维护信息所需的空间是常数级的，平均控制负载很低，可扩展性较优。

5.4 组播树的性能

组播树的性能可以用压力度和伸展度来衡量。压力度反映了数据包在底层链路传输的平均重复次数，压力度越小，宽带利用率越高。伸展度体现了节点到根节点的端到端传输延迟。在本系统中，由于我们关注于提供可交互功能的点播式流媒体服务，而不是关注端到端传输延迟较高的直播式服务，系统中的每个节点在同一时刻播放的内容并不需要尽量地同步。在系统中，未将伸展度作为衡量标准。

图 5 显示了基于 ISMT 协议的系统中链路的平均压力度。可以看出，树中的压力度仅为 1.3～1.6 左右，带宽利用率较高。且随用户数量的增加，压力度的变化不大，这是由于 ISMT 将两个节点之间的延迟作为父节点选择的依据之一，从而得到较好的组播树性能。

5.5 VCR 操作的响应延迟

响应延迟是流媒体系统服务中一个至关重要的性能指标。当新用户加入系统时，若等待时间过长，则有可能导致用户直接退出系统；而当用户进行 VCR 操作后，若响应延迟较长，则会导致停顿时间过长、组播树重新调整操作。为了减少响应延迟，ISMT 采用了临时连接的机制，试图加快父节点的定位过程。

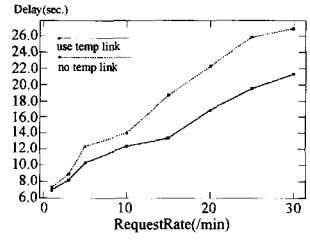

图 6 不同策略对响应延迟的影响

图 6 显示了 ISMT 系统中的平均响应延迟。我们分别考察了在使用临时连接机制和不使用这种机制两种情况下的平均响应延迟。可以看出，使用临时连接机制后，平均响应延迟减少了 25%～30%左右。

结束语 流媒体的固有特性对传统的网络传输方式提出了挑战，针对流媒体的研究已方兴未艾。利用现有的网络基础设施，在规模可伸缩的系统中提供预期的流媒体服务一直是研究的热点，如文献[8-11]。本文提出了一种适用于可交互流媒体系统的应用层组播协议 ISMT，由其构建的组播树能较好地支持交互操作，并有良好的节点聚集性，从而可以有效地降低传输时延。而且，由于节点的平均控制负载维护较低，系统有较高的可扩展性。

（下转第 125 页）

[3] Wittenmark B, Nilsson J, Torngren M. Timing problems in real-time control systems // Proc. The 1995 American Control Conference. Seattle, Washington

[4] Liu J W S. Real-Time Systems[M]. Published by Higher Education Press arrangement with the original publisher, Pearson Education, Inc. , Beijing, 2002: 195-218

[5] Bernat G, ABurns. New results on fixed priority aperiodic server [C] // Proc. of the 12th IEEE Real-Time Systems Symposium. Phoenix, Arizona: IEEE Computer Society Press, 1999: 68-78

[6] Lin Suzhen , Manimaran G . A FeedBack - Based Adaptive Algorithm for Combined Scheduling with Fault-Tolerance in Real-Time Systems [J] // Proc. Conference on High Performance Computing (HiPC). Bangalore, India, Dec. 2004: 101-110

[7] Lu C, Stankvoic J A. Design and Evaluation of a Feedback Control EDF Scheduling Algorithm[J] // IEEE Real-Time Systems Symposium. Phoenix, AZ, Dec. 1999

[8] Abeni L , Buttazzo G . Intergrating multimedia applications in hard real-time systems[J] // Proc. 19th IEEE Real-Time SystemsSymposium. Madrid, Spain

[9] Cervin A , Eker J . Control - Scheduling Codesign of Real - Time Systems: The Control Server Approach[J] // Proc. Journal of embedded computing. 2004

[10] VxWorks-Programmer's Guide 5. 5, 2002 Wind River Systems [EB/OL]. NC. http://www. windriver. com

[11] Goahead Software Foundation[EB]. http://www. goahead. com

[12] Lehoczky J P. Fixed priority scheduling of periodic task sets with arbitrary deadlines // Proc. of the IEEE Real-time Systems Symposium. Dec. 1990

[13] Audsley N, Burns A, Tindell K. Applying a new scheduling theory to static priority preemptive scheduling. Software Engineering Journal, 1993, 5(5): 284-292

[14] 唐应辉，唐小我. 排队论基础与分析技术[M]. 北京：科学出版社，2006：57-61

（上接第 110 页）

用户交互操作的不可预知性和过于频繁的 VCR 操作，会使系统组播树处于剧烈振荡的状态。下一步研究将着重于使节点的 VCR 操作引起的组播树结构变动限制在一个较小的范围内，以改善组播树性能。另外，无线终端的普及也有力地推进了无线流媒体研究，可以将本文的研究工作推广到无线网络。

参 考 文 献

[1] Deering S. Host extendion for i Pmulticasting. RFC－1112. August 1989

[2] Quinn B, Almeroth K. IP multicast applications: Challenges and solutions. Internet Engineering task Force (IETF) Internet Draft, March 2001

[3] Chu Yang-Hua, Rao S G, Zhang Hui. A case for end system multicast // ACM SIGMETRICS. 2000: 1-12

[4] Jannotti J, Gifford D K, Johnson K L. Overcast: Reliable multicasting with an overlay network // USENIX Symposium on Operating System Design and Implementation. San Diego, CA, October 2000

[5] Guo Y, Suh K, Kurose, et al. P2Cast: P2P patching scheme for vod service// WWW2003. May 20-24

[6] The ns manual. July 23 2003. http://www. isi. edu/nsname/ns/

[7] GT-ITM. http://www. cc. gatech. edu/fac/Ellen. Zegura/graph. html

[8] Deshpande H, Bawa M, Garcia-Molina H. Streaming live media over a peer-to-peer network// Submitted for publication. 2002

[9] Padmanabhan V N, Wang H J, Chou P A, et al. Distributing streaming media content using cooperative networking // ACM/IEEE NOSSSAV. Miami, FL, USA, May 2002

[10] Sheu S, Hua K A, Tavanapong W. Chaining: A generalized batching technique for video-on demand // Proc. of the IEEE Ith'l Conf. on Multimedia Computing and System. Ottawa, Ontario, Canada, 1997: 110-117

[11] Castro M, Druschel P, Kermarrec A M, et al. SplitStream: High-bandwidth content distribution in cooperatie environment // IPTPS'03. Berkeley, CA, USA, 2003

[12] Tran D, Hua K, Do T. ZIGZAG: an efficient peer-to-peer scheme for media streaming // Proc. of IEEE INFOCOM'03. San Francisco, CA, USA, April 2003

[13] Diot C, Dabbous W, Crowcroft J. Multipoint communication: A Survey of protocols, functions, and mechanisms. IEEE Journal on Selected Areas in Communications, 1997, 15(3): 227-190

[14] Wei L, Estrin D. A comparison of multicast trees and algorithms // IEEE INFOCOM. Toronto, Canada, June 1994

[15] Waitzman D, Partridge C, Deering S. Distance Vector Multicast Routing Protocol. RFC 1075. 1998

[16] Moy J. Multicast Extension to OSPF. RFC 1584(1994)

[17] Ballardie J, Francis F, Crowcroft J. Core Based Trees // Proceedings of the ACM SIGCOMM. San Francisco, 1993

[18] Estrin D , Farinacci D , Helmy A , et al . Protocol Independent Multicast-Spare Mode (PIM-SM): Protocol Specification. Proposed Experimental RFC. Sept, 1996

[19] Padmanabhan V N, Wang H J, Chou P A, et al. Distributing Streaming Media Content Using Cooperative Networking // NOSSDAV 02. Miami, Florida, USA, May 2002

[20] Fracis P. Yoid: Extending the Internet Multicast Architecture. White paper. http://www. aciri. org/yoid, April 2000

[21] annotti J, Gifford D K, ohnson K L, et al. Overcast: Reliable Multicasting with an overlay network // USENIX Symposium on Operating System Design and Implementation. San Diego, CA, Oct. 2000

[22] Deshpande H, Bawa M, Garcia-Molina H. Streaming Live Media over a Peer-to-Peer Network. Technical report. Stanford University, 2001

[23] Banerjee S, Bhattacharjee B, Kommareddy C. Scalable application layer multicast // Proc. of ACM SIGCOMM. August 2002

[24] Kwon J B, Yeom H Y. Providing VCR Functionality in Staggered Video Broadcasting. Transaction on Cosumer Electronics (SCI), 2002, 48(1): 41-48

[25] Almeroth K C, Ammar M H. On the Use of Mlticast Delivery to Provide a Scalable and Interactive Video-on-Demand Service. IEEE Journal of Selected Areas in Coummunication (NGC'99), Nov. 1999: 152-169