基于应用层组播的实时视频传送协议关键技术研究①

Research on the Real – time Transfer Protocol Based on Application Layer Multicast

姚 玮 张 卫 (华东师范大学计算机科学技术系 200062)

摘要:本文基于应用层组播技术,针对实时视频传输这类应用的特点,在分析和讨论了协议设计过程中面临的若干关键技术后,提出了自己的实时组播传输协议(Multicast Real – time Transfer Protocol,文中简称 MRTP 协议)。

关键词:应用层组播 MRTP 叠加网 网络自组织

1 引言

应用层组播的主要思想是:保持 Internet 原有的"单播、尽力发送"模型,尽量不改变原来网络的体系结构,而主要通过增加端系统的功能来实现组播的功能。由于对网络本身的改变很少,应用层组播具有很好的灵活性。但是,端系统的稳定性一般不如专用网络设备,应用层组播在带宽利用效率方面也无法和 IP 组播相比。另外,应用层组播中的系统框架和很多细节技术也还在研究当中。这些问题的存在为应用层组播的研究提供了广阔的空间。

本文主要研究一种针对实时视频应用的协议(MRTP)。希望借助应用层组播实现 internet 上多达几百人同时收看某一节目。

2 叠加网(Overlay Network)的拓扑结构

应用层组播网的结点是组播成员主机,数据路由、复制、转发功能都由成员主机完成,成员主机之间建立一个叠加在 IP 网络之上的、实现组播业务逻辑的功能性网络,称为叠加网(Overlay Network)。

针对我们的视频直播系统的应用特点,在 MRTP 协议中我们选择最简单的树状拓扑结构。视频的数据源就是树的根节点。组播树中的每一个节点都维护一张子节点列表。数据从组播树的根结点发出,不断转发给各自子节点列表中的所有子节点,最后到达各叶子结点。

与其他的一些拓扑结构相比,树状拓扑因为每一个节点都有固定且明确的数据转发路径,容易做到较好的网络延时抖动特性。又由于免去了那些用于路由的通信开销,网络利用效率也相对较高。当然,树状拓扑也有其弱点,那就是健壮性不好,一旦转发树中的某一节点离开了,则它所有的子节点都没有办法收到组播数据了,因此我们需要父节点的切换机制,以确保在父节点退出网络时,子节点们能及时的切换到其它的节点下,继续接受数据。

3 网络的自组织

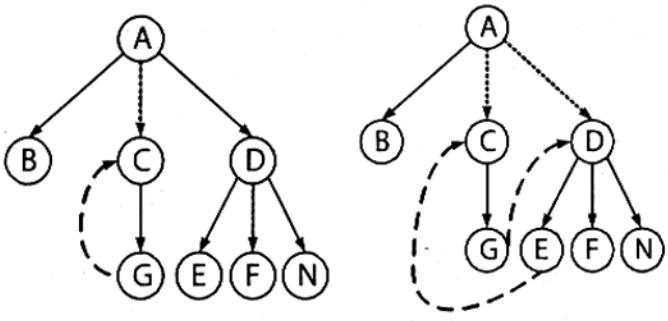
明确了需要组建树状的叠加网,接下来就要讨论如何一步一步地来构造这个网络。应用层组播网络自组织的一般步骤可以分为加入组,退出组两步。由于采用树状拓扑,因此组播网络自组织的过程还包括切换。下面详细介绍这三个步骤。

3.1 加入组

当某一个用户最初准备加入到组播组中时,它必须首先找到一个父节点,通知该父节点,使其在以后收到发往该组播组的数据时能够将数据转发给这个节点。通常有两种方法来找到自己的父节点,其一是分配模式:某一集中服务器为新加入的节点分配父节点,这样做的好处是该服务器可能对全局的把握更加好一些,容易找到对于全局来说较好的节点。其二是主动模式:指该新加入的节点通过某种渠道获得组中其它

① 华东师范大学远程教育研究院科研项目

的节点列表后，主动寻找可以接受自己的父节点，找出一个最好的作为自己的父节点，这种方式的优点是能够找到一个相对于自己来说较好的父节点。



A. 结点C加入自己的子结点 B. 分支C和D分别加入各自的叶子结点 C. 三个子结点互相切换

图 1 几种环路的构成情况

在 MRTP 协议中，为了能结合这两者的优点，我们折中了这两个方案。用户端启动后首先会在一台集中的服务器（在 MRTP 中被称为祖成员管理服务器）上注册自己，同时这台组成员管理服务器会为这个用户挑选出若干个可供选择的父节点。用户可以站在自己的角度上（比如说对其一一网络状况进行测试），再一次挑选出最好的那个节点作为自己的父节点。

3.2 退出组

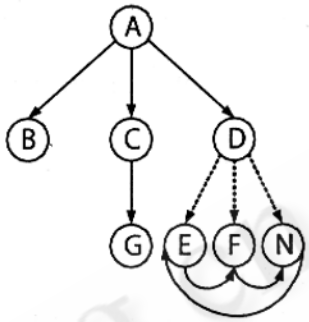
有两种情况会导致某一节点退出组，其一是该节点主动要求退出组播组，以后不再从父节点获取任何数据，也不再承担任何为子节点转发数据的任务。其二是因为该节点由于电脑故障或者网络故障，突然失去响应，无法承担数据转发的任务时，被判定为离开了组播组。

在 MRTP 协议中，一个节点在加入到组播网络中后，每隔 10 秒钟必须向组成员管理服务器发送 keep alive 报文。一旦组成员管理服务器在 30 秒时间内一次也未收到某成员的 keep alive 报文时，则判定该成员退出了组播组。组成员管理服务器只需简单的将该成员从它的组成员列表中删除，以后不再为新加入的其它成员分配它作为父节点即可。

除了组成员管理服务器外，某一节点的退出组还会被他的父节点和子节点们感知到，这是通过父子节点间的另一组 keep alive 实现的。一旦父节点感知到子节点退出时，会从自己的子节点列表中删除该子节点，并且停止向该节点发送组数据。而子节点感知到父节点退出时则会引发给予树状拓扑的网络自组织的切换行为。

3.3 切换

在基于树状拓扑的应用层组播中，父节点退出后，或者子节点感知到与父节点之间的网络特性变差，便会引发切换。子节点脱离与父节点之间的父子关系，另外选择自己的父节点。



在切换的过程中可能会出现一种情况就是若干个节点在整个转发网中形成了一个闭包。他们循环的加对方为父节点，因此谁也得不到数据。对于产生环路的情况，大概有以下几种原因造成：

A. 结点选择自己的子孙结点作为新的父结点。

B. 两个分支同时加入对方的叶子结点。

C. 多个结点同时互相加入的情况。

因此我们需要一种避免出现这种情况的方法。即使没有方法避免，也必须要有一种检测和恢复的手段。

在 MRTP 协议中，我们采用了继承 + 委派的机制来避免切换时形成环路。如图 2 – A 所示，当节点 D 退出时，由子节点 E，F，G 继承父节点 D 的位置，直接向祖父节点 A 发出加入请求。A 如果没有能力加所有的 EFG 为子节点，则会将 F，G 分别委派给 B，C 节点。当 EFG 切换时他们不能接受别的节点的加入请求，直到切换完成。最终完成切换后的状态如图 2 – B 所示。通过这套机制，我们就能够有效的避免循环拓扑的形成。

4 网络的优化调整

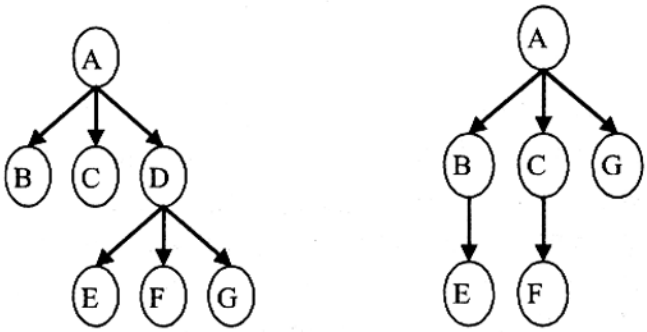
在应用层组播中，另一个重要的关键技术就是如何根据端结点间的网络的具体特性，不断优化调整叠加网的拓扑结构，以获得更好的应用层组播网络性能。

首先我们需要一些标准用于考量应用层组播网络的性能。由于应用层组播算法的一个特点是它的设计针对某种应用进行优化。所以，在对其评价的时候也就没有统一的标准。下面列出目前常用的一些评价标准，但是这些标准并不是针对每种应用层组播算法都适用的。

（1）带宽的使用。应用层组播会比 IP 组播消耗更多的带宽，不同方案消耗的带宽不同。

（2）延迟。由于在应用层组播中经过的跳数可能

更多，而且主机的性能并不能保证，延迟方面的性能比 IP 组播要差。



A. 节点 D 准备退出，切换发生前　　B. 切换完成，EFG 分别找到自己的父节点

图 2　MRTP 协议中节点的切换过程示意

（3）可扩展性。这个指标并不是所有应用层组播算法都需要的，只有在设计针对大规模组播组的算法时才需要考虑。

（4）健壮性。应用层组播系统的鲁棒性无法和使用路由器或者专用服务器的组播系统相比，但是可以通过一定的机制提供一定程度的健壮性。而且对于不同的应用，要求的健壮性也有所不同。

（5）易推广性。应用层组播提出的一个推动力就是原有的 IP 组播在推广使用方面存在问题，所以在应用层组播的研究中把"易推广性"作为一个很重要的目标。"易推广性"不仅表现在不依赖于对现有网络体系结构的改变，还表现在要解决目前网络中 NAT 和防火墙（它们破坏了 Internet 原来的"扁平"模型）所带来的问题。

在 MRTP 协议中，我们为了优化网络传输，加入了以下一些策略：

4.1 父节点优选

当用户在组成员管理服务器上注册时，组成员管理服务器会在当前已注册的所有成员中，按一定的法则筛选出若干可供该新组成员连接的父节点，这一个过程称为父节点优选，它有利于节点快速找到一个相对较好的父节点。优选时要参考的度量有 ip 地标相似度，节点所处层次深度和节点剩余的带宽量。对于某已加入组的节点 X 和请求加入节点 A 的具体计算公式如下：

Goodness(X) = T1 (IP(X) ^ IP(A)) + T2 (Depth(X)) + T3 (Ability(X))

其中 Goodness(X) 是将节点 X 分配给节点 A 作为父节点的合适性，IP(X) ^ IP(A) 是将节点 X 与节点 A 的 IP 地址求异或，Depth(X) 是节点 X 在树状叠加网络中所处的层次，Ability(X) 是节点 X 还剩余的上行带宽能力。T1，T2，T3 是权值调整函数。优选时，组成员管理服务器对于每一个已注册的节点 X 计算其 Goodness(X) 的值，找出数值最大的那几个节点，将他们的信息返回给请求加入者。

4.2 节点加入过程优化

当用户在收到组成员管理服务器给出的若干个可选父节点时，需要进行父节点连接测试，以确定究竟选哪个节点作为自己的父节点。连接测试主要考量一段时间内该节点与可选父节点间的 RTT 时延，抖动时间，丢包率。最后找到考量指标最好的那个节点作为自己的父节点，同时测量的结果将被用于确定缓冲区的初始大小。

4.3 数据传输过程优化

针对实时视频应用的特点，要求数据传输的时延不能太长，组播数据允许部分丢失，但不能丢失太多。在一般的 Internet 应用层应用中，数据的传输可以通过 UDP 或者 TCP。通过 UDP 来传输数据的特点是实时性好，效率较高，但是传输不可靠，容易丢包和乱序。通过 TCP 来传输数据则能保证数据的正确按序到达，但不能保证实时性。

在 MRTP 中我们综合了两者的优点，数据的传输是通过 UDP 来进行的，同时加入了一次重传和随机重传的机制。一次重传请求是指子节点发现某个数据包丢失，马上向父节点请求重传该数据包，但是处于实时性考虑仅请求一次。假设在网络链路上有 10% 的可能性发生丢包，则经一次重传后数据包仍无法恢复的可能性为 (1 - (0.9 * 0.9)) * 0.1 = 0.02，即 98% 的数据包能正确得到达，如果只有 5% 的可能性发生丢包，则 99.5% 的数据包经过一次重传能正确得到达。随机重传请求则是当父节点要向子节点转发一个自己都没有的数据包时，向子节点发送一个特定的空数据包，子节点在接收到的时候，向组中的某一随机节点发送请求重传。

通过基于 UDP 的一次重传和随机重传，基本能够做到数据实时并且尽可能正确的延树状叠加网络向下传递。

4.4 自适应缓冲区管理

缓冲区的管理指当从父节点接受的数据后，何时播放以及何时向子节点转发该数据。因此缓冲区管理的好坏，很大程度上影响了系统的实时性以及播放的流利性。

如图 3 所示，缓冲区总大小应该为 RTT + 4 * Jitter，其中 RTT 是子节点与父节点之间往返时延(Round Trip Time)，Jitter 是抖动时延，即多次 RTT 测量值的标准差。其中在警戒线和提交线中留出的 RTT + 2 * Jitter 是留给用于一次重传的时间。

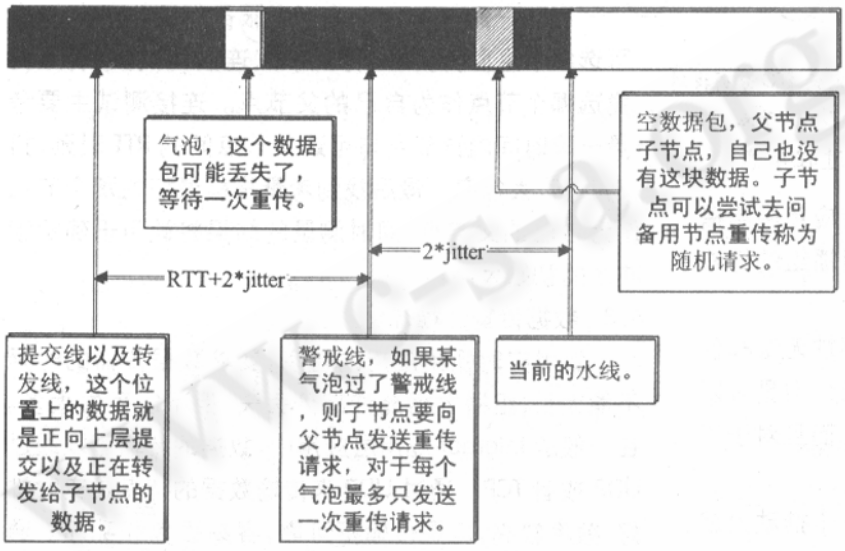


图 3　MRTP 缓冲区设计示意

由于在 Internet 中，父节点和子节点间实际的 RTT 和 Jitter 可能会各不相同，即使是同两个节点之间，不同的时间也有可能会有比较大的差别。因此 MRTP 在数据传输过程中会不断的测量 RTT 和 Jitter 的值，以此来动态的调整缓冲区的大小，确保最好的传输效果和时延。

4.5 内网穿透机制

MRTP 还支持有限的内网穿透，即位于内网的节点可以加位于同一个内网中的节点或者公网上的节点为父节点。但是位于公网上的节点，却无法加位于内网中的某一个节点为父节点。

为了实现这种机制，就必须有种方法来区分哪些节点是内网的节点，哪些是公网的节点，是内网的那些节点是否是同一个内网的？在 MRTP 中，每个节点在向组成员管理服务器注册时会报告自己的 ip 地址。如果经过了 NAT 地址转换则组成员管理服务器就能够感知到。他在进行父节点优选时就可以将这个因素也考虑进去，保证分配给加入节点的可选父节点是可以连接的节点。

5　结束语

本文介绍了一种面向实时视频组播的应用层组播系统 MRTP，并且详细分析了它的三个关键技术：叠加网的拓扑结构、网络的自组织、网络的优化调整，提出了在 MRTP 中的相应解决方案。我们已经在 WINDOWS 平台上实现了该协议，在这个基础上完成了一个传输流量约为 300kbs 简单的视频应用。测试中最多有 40 多个节点参与，整体时延小于 5 秒。今后的工作重点：一是对文中列出的 5 个网络优化策略继续研究和调整，以获得更好的性能。二是在现有的协议平台上实现一个完整的应用，并且进行更大规模的测试，以更好的验证协议的性能。

参考文献

1　一个单源的应用层组播协议的设计和实现，方奕、张卫，计算机应用，2005 年 2 期.

2　应用层组播综述，李珺晟、余镇危、潘耘、李霞、曹建华、武浦军，计算机应用研究，2004 年 21 卷 11 期.

3　Yang – hua Chu, Sanjay G. Rao, Srinivasan Seshan and Hui Zhang. A Case for End System Multicast [C]. Proc. of ACM SIGMETRICS'00, pages 1 – 12, June 2000.

4　J. Jannotti, D. Gifford, K. Johnson, M. Kaashoek, and J. O'Toole. Overcast: Reliable Multicasting with an Overlay Network [Z]. The 4th Symposium on Operating Systems Design and Implementation, October 2000.

5　Suman Banerjee, Bobby Bhattacharjee. A Comparative Study of Application – layer Multicast Protocols [M]. Submitted for Publication, 2002.

6　Zebra: Peer To Peer Multicast for Live Streaming Video, Maya Dobuzhskaya, Rose Liu, Jim Roewe, Nidhi Sharma, 2004.6.