文章编号:1001 - 9081(2009)02 - 0484 - 03

基于中心环旋转木马的应用层组播模型

许建真,许 强
(南京邮电大学 计算机学院,南京 210003)
(xqdxr999@163.com)

摘 要:提出了一种基于中心环旋转木马的高效分布式应用层组播模型(CRCL)。CRCL 采用层次环状结构,以中心环为基础,每个中心环节点组建自己的旋转木马,通过创建节点优先级,使高优先级节点位于上层环中。通过仿真实验对比表明:该模型在中小型流媒体应用中具有很高的数据传输率、较短的收敛时间,同时平均路径长度也较小。

关键词:中心环;旋转木马;应用层组播;流媒体;数据传输率
中图分类号: TP393 **文献标志码**:A

A distributed application layer multicast model based on central ring carousel

XU Jian-zhen, Xu Qiang
(*College of Computer, Nanjing University of Posts and Telecommunications, Nanjing Jiangsu* 210003, *China*)

Abstract: A distributed application layer multicast model based on Central Ring Carousel algorithm was proposed. The model employed ring as the basic topology unit, constructed carousel on each central node, and built its upper rings topology according to each node's priority. The simulation results indicate that CRCL can improve the performance of streaming media application efficiently, especially in data transmission, convergence time and average path length.

Key words: central ring; carousel; application layer multicast; steam media; data transmission rate

0 引言

随着 Internet 与计算机技术的发展,人们对视频点播、电视会议、远程学习等流媒体的应用需求越来越强烈,而且对这些应用的质量要求也越来越高。由于资源的限制,直接利用一个发送者和一个接收者的单播传送已经不能满足需求。这时组播的作用得以发挥,人们越来越发现它的有用性和重要性。IP 组播由于技术和商业驱动的问题使其没有在 Internet 上得到广泛部署。因此,应用层组播技术成为网络流媒体研究的热点。

Internet 的实际环境是复杂的,应用层组播技术作为一种实用技术必须能解决这些实际问题。在实际的网络环境中,参与应用层组播组中的成员节点性能差异也很多。近几年提出了许多基于应用层组播的模型,如 NICE[1],SCAMP[2] 以及 P4L[3] 等。

ALM 在某些应用上是用户越多,越能得到及时和优质的服务,但同时也带来了一些严重的问题,例如对网络带宽的无限制的浪费。于是如何提高应用层多播的效率成为迫切需要解决的问题。

本文提出了一种基于中心环旋转木马拓扑结构的应用层组播解决方案。该方案考虑了节点的性能,并将组播系统中性能优的节点组成中心环,减少了对 RP(Rendezvous Point)节点的负载和依赖性。在数据传输方面,本文方案由上层环中节点对下层节点进行分发,如果下层环中节点没能及时收到数据,则收到数据的邻居节点对其进行数据的传输,从而减轻上层节点的压力,加快了传输速度,提高了系统的整体性能。

1 CRCL 组播模型的设计

中心环旋转木马的高效分布式应用层组播模型(Central Ring Carousel,CRCL)采用分布式控制协议中的网优先方式。即首先建立一个覆盖网,然后在覆盖网上建立传输拓扑。

1.1 覆盖网的拓扑结构

CRCL 模型将组播中成员进行分层组织结构,并在每层不同的域中采用环结构。下层域中的环成员由上层环中某个特定的节点进行控制,这样每个上层环中节点都分散控制管理着自己的下层域中节点,从而减轻了 RP 节点的控制管理负担。该拓扑结构的最高层是 RP 节点,紧接着的高层是一个由中心节点组成的中心环。这些中心节点是通过计算节点的性能和优先级从普通节点中选出来的。本模型的拓扑结构如图 1 所示。

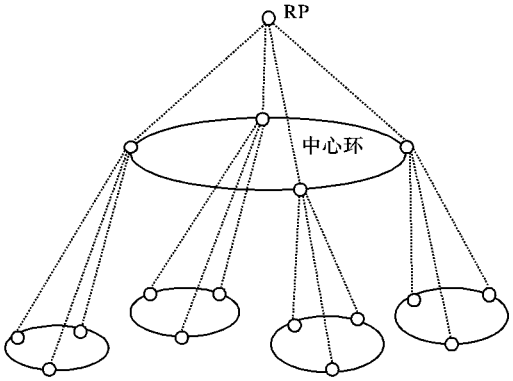

图 1 覆盖网拓扑结构

收稿日期:2008 - 08 - 07;**修回日期**:2008 - 09 - 23。

作者简介:许建真(1966 -),男(回族),安徽宿州人,副教授,主要研究方向:计算机通信与网间互联技术; 许强(1984 -),男(回族),安徽宿州人,硕士研究生,主要研究方向:计算机通信与网间互联技术。

1.2　覆盖网的建立

在本组播模型中,覆盖网的建立和形成是系统功能执行的基础。它包括节点的加入和退出,以及各层环的维护。

初始时,系统只有一个 RP 节点,当有新节点加入时,新节点向 RP 发送加入请求,RP 收到请求后会记录新节点的信息,并将其加入中心环中,并给新加入节点发确认信息。RP 将根据节点的优先级动态地调整中心环的中心节点,使优先级高的节点在中心环中。

1)节点的加入。中心节点是中心环中的节点,是从普通节点中选择出来的。一个普通节点加入组播系统时,首先向 RP 节点发送 JOIN 信息,此 JOIN 信息包含节点所在网络状态以及节点自身物理位置等信息。在 RP 节点中记录着中心环中中心节点的数量,如果当中心节点数量未达到 RP 设定的中心节点数目时,加入的普通节点将成为中心节点,成为中心环的节点。RP 节点会记录新加入的节点的信息,并给节点发 ACK 确认信息。新加入中心环的节点将获得中心环中其他中心节点的信息,并储存这些信息。RP 节点也将使其他的中心节点也相应的更新自己的邻接中心节点列表。然而当中心节点数量达到了 RP 节点初始时设定的控制的中心节点数目时,RP 节点将把新加入的节点分配给中心环的合适的中心节点。中心节点将把新加入的节点放在自己的下层域的环中。

当节点加入某中心节点的下层环时,将更新自身的节点列表,该表记录包含同一环中的邻居节点信息,还记录了下层环的节点列表和上层中心节点列表。然后新加入的节点将向自身节点列表的节点发送更新请求信息。中心节点收到信息后将记录新加入节点的位置,其他收到更新请求的节点将根据具体的情况更新相应的列表。如果中心节点的下层环的节点数目都达到初始设定的数值时,中心环中的节点将把新加入的节点交给自己所属的下层环中的合适节点,由它来分配新加入节点的位置。它将根据需要,将新加入节点放在合适的环中。这就好像以中心节点为中心的旋转木马一样,每个中心节点构建自己下属的环,组建自己的旋转木马。

2)节点的退出。当中心环中的中心节点正常退出时,它将向 RP 节点和自己下属层环中的节点发送 QUIT 信息,RP 节点收到信息后将更新自己的节点记录,该中心节点下属层环收到信息后,将从环中选择一个节点成为中心节点,选择的依据是节点的性能和优先级,并同时更新相应节点的节点列表。

当其他环的节点正常退出时,上层环的节点将把它从自己的列表信息中删除,并通知退出节点所在环的其他节点更新自己的列表信息。

由于网络的实际环境,系统的节点可能发出非正常退出。所以系统中的节点会周期性地发送心跳消息(Heartbeat Message)给上层所属节点。如果一个上层节点在几个心跳周期内都没有收到下层节点发来的心跳消息,那么将把这个节点的状态视为离开,并将这个下层节点在列表中的表项删除,并通知其他相应的节点更新自己的节点列表。

3)各层环的维护。中心环中的中心节点由 RP 节点进行控制管理,RP 节点控制中心节点的加入和离开并在环节点失效后执行环的愈合操作。其他各层环的节点由自己所属的上层节点进行控制管理。

1.3　优先级的建立

在本文的模型中,中心节点的选取以及上层环中节点离开后下层节点选取上层节点时,都需要优先级的建立。所以本文采用一种动态分析的方法,根据各节点的资源占有量、时延、带宽以及在系统的存在时间等,计算节点相应的优先级。各层节点根据自己的优先级动态调整自己所在的层位置。让拥有大量资源和有效带宽的节点在上面的层中。这样性能好的节点都保持在上面的层次,有利于信息和数据的分发。

在 PBHM[4] 中建立了一种优先级算法。我们在此参考此优先级算法,并建立一种新的优先级算法。首先我们先考虑节点的一些性能,通过式(1)进行计算:

$$E = n \times \frac{upl_{\text{bandwidth}} \times downl_{\text{bandwidth}}}{T_d \times T_d} \tag{1}$$

其中 n 表示节点的度数;$upl_{\text{bandwidth}}$ 表示本节点剩余的上传带宽;$downl_{\text{bandwidth}}$ 表示本节点剩余的下载带宽;T_d 表示节点的时延。

为了计算的准确性,我们用节点的平均时延来代替时延。根据排队模型,可推导出的时延公式:

$$T_d = \frac{\frac{\gamma}{u} + \gamma\sigma_{T_s}^2 u}{2(1-\gamma)} \tag{2}$$

其中 γ 表示为信道的利用率;$T_s = 1/\mu$ 为报文的处理时间,$\sigma_{T_s}^2$ 为其方差。

一段时间 Δt 内的平均效率为:

$$\rho = \frac{D_{\text{sent}}(n_t)}{D_{\text{sent}}(n_t) + D_{\text{receive}}(n_t)} \tag{3}$$

其中 $D_{\text{sent}}(n_t)$ 表示节点 n_t 发送的数据。$D_{\text{receive}}(n_t)$ 表示节点 n_t 接收的数据。

$P\_effc(\rho)$ 表示随 ρ 的变化,组成员优先级的变化趋势。

$$P\_effc(\rho) = \frac{\rho}{1-\rho} e^{\beta\frac{\rho^2}{1-\rho^2}},\ 0 \leqslant \beta \leqslant 1 \tag{4}$$

另外我们再考虑节点的在线时长,在线时间长和性能优的节点应尽量在系统中的中心环上。这样有利于系统信息的传播和数据的分发,同时还能使系统尽快地处于稳定状态。设 T 为系统稳定的时间,t_0 为节点在系统中存在的时间。

考虑上面几方面的因素,可推导出的优先级公式:

$$pri(n_t) = P\_effc(\sigma) + \zeta E + \tau t_0/T;0 \leqslant \zeta,\tau \leqslant 1 \tag{5}$$

其中,ζ,τ 是添加的系数。在实际网络环境中,优先级的建立可根据具体的情况设定不同的参数。例如在节点加入退出比较频繁的情况下,我们可以增大 ζ,τ 的值,提高在线时间长且性能优的节点的优先级,从而加快系统信息的分发,使系统更快的稳定下来,提高系统的稳定性。

1.4　数据的传输

在传输流媒体时,服务器首先将流媒体数据进行分块,再将分块的数据进行传输。因为流媒体是视频、声音和数据通过实时传输协议以连续流方式顺序从源端向目的地传输,目的地只需接收到一定数据缓存后就可以立即播放的多媒体应用,它采用"边下载边运行"的模式,所以对系统的实时性要求比较高。在数据块传输时,服务器首先将数据块传给中心环中的中心节点。接着由中心环中的中心节点来负责自己所属的下层环中节点的传输。

中心环的中心节点向下层环中的节点进行数据块的传输,下层环中的节点收到数据后再向自己所属的下层环中节点进行传输。在上层节点向下层环中节点进行数据传输时,下层环中节点不一定都能及时地收到上层节点发送的数据块。所以我们采用一种下层环中节点辅助传输的方法,这也是本文下层域中节点设计成环的目的所在。

下层环中收到上层节点数据块的节点,会向自己的邻居节点发送 REQUEST 信息。REQUEST 信息的内容是询问邻居

节点有没有收到数据块,如果已经收到从上层节点发送的数据块,则发送 REFUSE 信息,不需要从其接收数据块。如果没有收到上层节点发送的数据块,则需要从其邻居节点接收数据块,并给上层的节点发送 REFUSE 信息,表示已经准备从邻居节点接收本次的数据块不需要上层节点再发送了。采用此传输算法不仅减少了上层节点传输的压力,而且保证了节点总能在较短的时间内接收到数据,满足了多媒体应用对实时性的要求。

2 模型的性能与仿真分析

2.1 性能参数定义

一个好的 ALM 协议,除了要有好的网络资源利用率和传输效率之外,还应该有适中的控制负载、良好的网络负载平衡,并且要能满足接收者的性能需求。本文从下面几个方面考查系统的性能。

1)链路压力:指在某一条链路上同时要转发的相同的数据的数目。平均链路压力(Average Link Stress, ALS)可用式(6)表示:

$$ALS = \frac{S_{total}}{L_{total-rec}} \tag{6}$$

其中 S_{total} 表示总的数据传输量; $L_{total-rec}$ 表示节点接收到的有用数据量。

2)平均路径长度:系统中节点距离中心节点的距离即经过的节点个数的平均值。

3)平均数据传输率:节点收到的组播数据包的数目与源节点发送的总的数据包的比值。

2.2 仿真分析

本文在一台 Linux 主机上搭建了仿真平台,由 GT-ITM 拓扑产生器生成实验所需的拓扑结构。首先我们让 300 个节点参加多播组中,来比较 CRCL 和 NICE 在失败节点恢复时链路压力方面的性能。从图 2 的仿真结果来看,CRCL 比 NICE 的链路压力要小,而且 CRCL 趋于稳定的时间要短。这是因为 CRCL 采用了优先级,优先级高和性能好的节点能快速控制和管理自己域中环的节点,减少了处理时间,优化系统性能,使系统能较快恢复并趋于稳定。

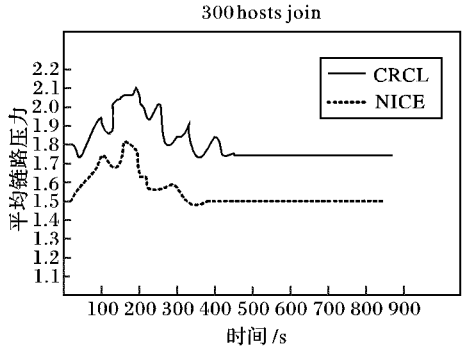

图2 链路压力比较

在平均路径长度方面,CRCL 的性能要优于 NICE。这是因为 CRCL 中每个下层环中的节点都由上层环中的某个节点直接控制,所以从源节点只需经过几个特定的中间节点就可以到系统中的节点。设系统中的节点总数为 M,环中的节点数最大值为 N_{max},确定的层数为 L。则有如下关系:

$$L = \sqrt[N_{max}]{M} = M^{\frac{1}{N_{max}}} \tag{7}$$

例如系统中有 300 个节点,环中的节点数最大值我们设为 7,则系统的层数只有三层。从源节点到系统中的节点可以很快遍历到。而 NICE 则不是这样,最差可以达到 $O(k \log M)$,M 为系统中节点总数。CRCL 和 NICE 的平均路径长度仿真结果如图 3 所示。

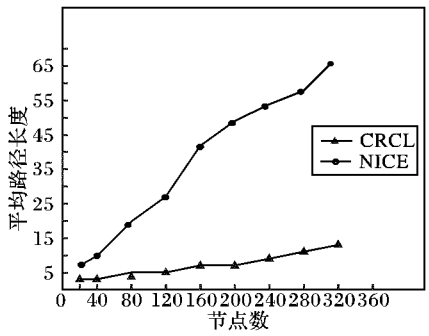

图3 平均路径长度比较

如图 4 所示,CRCL 比 NICE 的数据传输率要高。这是因为 CRCL 由上层环中节点对下层环中节点进行控制传输,当上层节点不能及时传输数据到下层节点时,下层环中收到数据的节点会及时传输数据给自己的邻居环中节点。当上层节点失效时下层环中节点会根据优先级快速选择一个上层节点出来,而 NICE 中上层节点失效对下层节点影响比较大,要进行拓扑结构重组。从而数据传输率低于 CRCL。

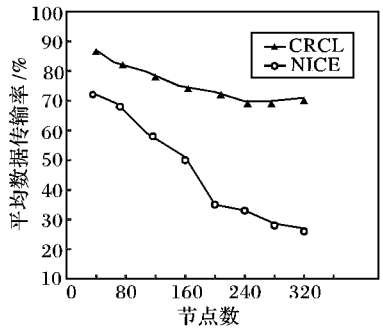

图4 平均数据传输率的比较

3 结语

本文提出一种基于中心环旋转木马的应用层组播方案,该解决方案适合中小型的流媒体系统。在中小型的流媒体系统中运用此方案,能提高系统的数据传输效率,减少系统收敛的时间,优化了系统性能,提高了在流媒体应用的效率。虽然该模型在流媒体应用中有上述的优势,但系统的安全性没有充分考虑,下一步将会重点予以考虑。

参考文献:

[1] BANERJEE S, BHATTACHARJEE B, KOMMAREDDY C. Scalable application layer multicast[EB/OL]. [2008-06-10]. http://pages.cs.wisc.edu/~suman/pubs/sigcomm02.pdf.

[2] AMAD M, MEDDAHI A. P4L: A four layers P2P model for optimizing resources discovery and localization[C]// APNOMS 2006. berlin: Springer-Verlag, 2006: 342-351.

[3] GANESH A, KERMARREC A M, MASSOULI L. Peer-to-peer membership management for gossip-based protocols[C]// IEEE Transactions on Computers. Washington, DC: IEEE Computer Society, 2003: 139-149.

[4] XU JIANZHEN, XU MIHUA, ZHANG FUYAN. Priority-based hierarchical application layer multicast management model[J]. International Journal Computer Science and Network Security 2007, 7(4): 242-249.

[5] AMAD M, MEDDAHI A. A scalable P2P model for optimizing application layer multicast[C]// IEEE/ACS International Conference on Computer Systems and Applications. New York: IEEE, 2008: 407-413.