ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.3, March 2007, pp.625−635
DOI: 10.1360/jos180625

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

应用层组播用户的自私性研究*

李 丹+, 吴建平, 崔 勇

(清华大学 计算机科学与技术系, 北京 100084)

Study on Selfishness in Application-Layer Multicast

LI Dan+, WU Jian-Ping, CUI Yong

(Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

+ Corresponding author: Phn: +86-10-62795818 ext 6864, E-mail: lidan@csnet1.cs.tsinghua.edu.cn

Li D, Wu JP, Cui Y. Study on selfishness in application-layer multicast. *Journal of Software*, 2007,18(3): 625−635. http://www.jos.org.cn/1000-9825/18/625.htm

Abstract: Application-Layer multicast (ALM) is an important supplement to IP multicast. However, unlike in IP multicast, the participating nodes in ALM are selfish and strategic host users. In order to improve their own interests, selfish host users might not strictly obey the ALM protocols, because of which the overall performance of the multicast session could be affected. To design robust and trustworthy ALM protocols, it is necessary to study the selfishness in ALM. This paper surveys the recent research trends in this area, and classifies the researches into three categories according to the working steps of ALM protocols, that is, the selfishness in maintenance of control structure, the selfishness in collection of node information, and the selfishness in construction of data structure.
Key words: multicast, application-layer multicast, selfishness

摘 要: 应用层组播(application-layer multicast,简称 ALM)是网络层组播的重要补充.但与网络层组播不同的是,应用层组播结构的组成节点是具有独立利益和决策的主机用户.自私的主机用户为了提高自身利益,可能不严格遵守应用层组播协议的规定,从而对组播会话的整体性能带来影响.为了设计可信任的、鲁棒的应用层组播协议,对应用层组播的用户自私性进行研究是必要的.综述了这一领域的研究进展,并按照应用层组播协议的工作阶段把这些研究分为 3 类,即控制结构维护阶段的自私性研究、节点信息收集阶段的自私性研究以及数据结构构造阶段的自私性研究.
关键词: 组播;应用层组播;自私性
中图法分类号: TP391 文献标识码: A

随着互联网的飞速发展,一些新型应用,如网络会议、流媒体传播、网络电视、大规模交互式网络游戏等对高效率的组通信技术提出了要求.组播是一种比较理想的实现组通信的网络通信技术,它允许一台或多台主机作为组播源一次性地发送单一数据包到多台主机接收者[1].与传统的单播技术相比,组播能有效地节省网络带宽,减轻网络负载和服务器负载.

* Supported by the National Natural Science Foundation of China under Grant Nos.90104002, 60303006 (国家自然科学基金); the National Grand Fundamental Research 973 Program of China under Grant No.2003CB314801 (国家重点基础研究发展规划(973))
Received 2006-09-09; Accepted 2006-11-21

最先提出的组播技术是实现在网络层的,称为网络层组播[1].在网络层组播中,路由器作为组播数据的分发节点,组播树建立在路由器上,因此具有很高的效率和可扩展性.然而,网络层组播改变了传统互联网的基于单播的设计原则,而且与其相关的一些问题比如组播成员管理、组播拥塞控制、组播计费机制等至今并未得到很好的解决.因此,网络层组播一直没有在互联网上得到大规模的部署.

Francis[2]和 Zhang[3]在 1997 年和 1998 年各自独立地提出应用层组播的思想,建议在应用层来实现组播技术.与网络层组播不同,应用层组播中的数据分发节点是主机,而分发节点之间的数据传输是通过传统的单播技术来实现的.与网络层组播相比,应用层组播在效率、可扩展性等方面有着先天性的劣势,但是应用层组播的最大优点在于其易于部署.另外,由于应用层组播没有改变传统互联网的单播模型,不依赖于互联网的底层实现,用户可以根据自己的需求设计更加灵活的组播模型.可以说,应用层组播体现了组播在组通信中的技术优势和组播技术的可部署性之间的一种折衷.自从应用层组播的思想诞生以来,研究者已经设计了许多应用层组播协议[2−20].

与网络层组播相比,应用层组播的一个重要特点在于其数据分发节点是具有独立利益和策略的主机,而不是由网络运营商统一维护的路由器.因此,在网络层组播中,可以认为同一个自治域内部的组播分发节点即路由器之间是严格遵守网络协议并且是相互合作的,但这些假设在应用层组播中不再严格成立.应用层组播中自私的主机用户可能会采取独立的策略以在接收组播数据时获得较高的效用,或者承担较少的数据转发负担.比如,主机可以隐瞒真实出口带宽以减少为其他节点转发数据的机会,也可以谎报到源节点的距离以在组播树上获取更靠近源节点的位置,等等.由于现有的组播协议中对组播结构的优化算法都是建立在所有参与节点真实地报告私有信息并忠实执行协议规定的基础上的,主机用户的自私性行为在提高个体利益的同时,可能会对组播会话的总体性能造成影响.目前已经出现了许多关于应用层组播用户自私性的研究.这类研究主要包括两个方面,一是分析主机用户的自私性行为对应用层组播会话总体性能的影响,二是设计防御应用层组播中用户自私行为的机制和算法.本文综述了这一领域的最新研究进展,并对各种研究进行了分类和比较,并对下一步的研究方向作了讨论.

本文第 1 节对应用层组播的研究现状进行概述.第 2 节讨论应用层组播用户自私性研究的最新进展,并按照应用层组播协议的 3 个工作阶段进行分类.第 3~5 节按照 3 个类别分别介绍应用层组播用户自私性的研究情况.第 6 节对目前的这些研究进行比较,最后总结全文并展望下一步可能的研究方向.

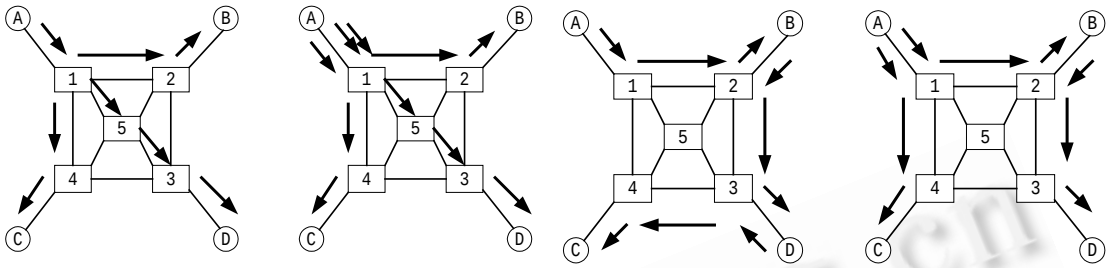
1 应用层组播研究现状概述

应用层组播的基本思想如图 1 所示.图中方框代表路由器,圆圈代表主机.图 1(a)是网络层组播的实现示意图.在网络层组播中,分组在网络中的路由器处进行复制,如果主机 A 想往主机 B,C,D 发送分组,则分组在路由器 1 处进行复制.而在应用层组播中,分组在主机处进行复制.由于应用层组播可能会在同一条链路上多次发送相同的分组,因此它的效率要低于网络层组播.另外,由于应用层组播不考虑网络本身的拓扑结构,因此应用层组播和网络层组播相比通常延时较大.

应用层组播数据结构的质量通常采用强度(stress)和伸展度(stretch)这两个参数来衡量:.强度定义为每个链路或者每个路由器在传输组播分组时发送相同分组的次数.在网络层组播中,由于组播分组并不进行多余的复制,因此,网络层组播中强度参数总是 1.伸展度定义为平均每个组成员在覆盖网络中的从源到目的的距离以及对应的单播路径距离的比值.图 1(b)中直接采用源结点 A 到其他结点的单播路径进行组播转发.这时,伸展度为 1,A 与路由器 1 之间链路的强度为 3,其他链路的强度均为 1.图 1(c)中采用的是环型组播方案.在这种方案中,链路的最大强度为 1,而平均伸展度为 $O(n)$,其中,n 为参与应用层组播会话的主机数量.图 1(d)中的方案是(b)和(c)中的方案的折衷,它的最大强度和平均伸展度都位于两者之间.

在应用层组播中,出于可扩展性的考虑,每个节点一般不会维护其他所有参加组播会话的节点,而只维护其中一部分节点作为邻居节点[20].所有节点之间的邻居关系构成了应用层组播的控制结构.组播数据结构就是在控制结构的基础上建立的,组播数据沿着组播数据结构从源节点向网络中传播.根据组播数据结构的特征,目前

提出的应用层组播协议可以大致分为两类,一类是基于树状的应用层组播,另一类是基于网状的应用层组播.下面分别介绍这两类应用层组播协议.



(a) (b) (c) (d)

Fig.1 IP multicast and application-layer multicast

图 1 网络层组播和应用层组播

1.1 基于树状的应用层组播

基于树状的应用层组播,其数据结构特点与网络层组播类似,都是一棵覆盖所有组播节点的分发树.在组播控制结构的基础上,所有参加组播会话的节点之间建立一种长期性的父子关系.每个节点在邻居节点中选择一个唯一的父节点,并在可用出口带宽范围内接受子节点.在建立好父子关系之后,由父节点转发自己接收到的所有数据给所有子节点.一旦所有节点之间的父子关系建立,也就是组播树建立好之后,节点之间除了传输组播数据之外,只需要根据控制结构的动态变化对组播树进行调整,并没有其他的额外控制负载.但如果一个节点离开了组播会话,或者由于某种原因不再正常工作,其所有子孙节点都将产生播放停顿,直到组播树重新建立好为止.基于树状的应用层组播数据结构如图 2 所示,其中节点 A 是组播源节点.

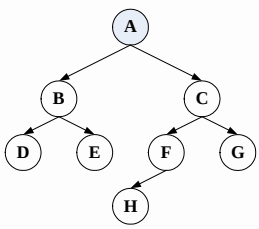


Fig.2 Data structure of tree-based overlay multicast

图 2 基于树状的应用层组播数据结构

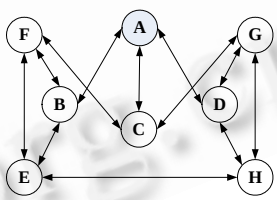


Fig.3 Data structure of mesh-based overlay multicast

图 3 基于网状的应用层组播数据结构

按照构造控制结构和组播树的顺序,基于树状的应用层组播协议可分为 3 类.第 1 类协议先构造控制结构,然后在控制结构上生成组播树,比如 NARADA[3],Scattercast[4],Kudos[5];第 2 类协议先构造组播树,然后在组播树的基础上维护额外的虚链路以组成控制结构,比如 Yoid[2],TAG[6],Host Multicast[7],ALMI[8],TBCP[9]等;第 3 类是采用面向大规模应用层网络的路由机制创建带有某些特殊属性的控制拓扑,在控制拓扑中就隐含定义了数据转发路径,比如 NICE[10],ZIGZAG[11],Scribe[12],Can-Multicast[13]等.

1.2 基于网状的应用层组播

与基于树状的应用层组播不同,在基于网状的应用层组播中,其数据结构不再是一棵具有明确父子关系的树.每个节点都从控制结构的邻居节点中选择并维护一定数量的伙伴节点.所有节点之间的伙伴关系构成了组播数据结构,即一个网状结构.源节点上的流媒体数据被分成若干个数据段.每个节点向伙伴节点通告其拥有的数据段信息,并从拥有某个数据段的所有伙伴节点中选择一个来请求接收这个数据段.与基于树状的应用层组播相比,基于网状的应用层组播中的数据段通知和数据段请求都引入了额外的控制开销.但是基于网状的应用层组播最大的优点是它可以更好地容忍节点的动态变化,而这在应用层网络的环境中是非常重要的.由于每个

节点都可以从多个拥有某个数据段的伙伴节点中选择一个来接收数据,一个节点离开组播会话或者发生故障不会给它的伙伴节点带来太大的负面影响.在基于网状的应用层组播中,一个节点也有可能发生播放停顿,但产生播放停顿的原因是该节点的伙伴节点都不能提供到达播放时刻的某个数据段.基于网状的应用层组播的数据结构如图 3 所示,其中节点 A 是组播源节点.

根据数据结构的网状是否有隐含内部结构,基于网状的应用层组播也可以分为两类.一类是采用多树的方案来实现的,数据结构的网状实际上由多棵组播树构成,如 SplitStream[14],Bullet[15],CoopNet[16]等;另一类是采用完全无结构的方案来实现的,数据结构的网状没有任何隐含内部结构,如 PRO[17],DONet[18]和 PROMISE[19]等.

2 应用层组播用户的自私性问题

在网络层组播中,组成组播数据结构的节点是由运营商管理的路由器.因此,在同一个自治域内部,路由器是完全按照组播协议的规定来生成高效率的组播分发结构的.而对于应用层组播来说,情况却有所不同.组成应用层组播数据结构的节点是应用层的主机用户,它们都有各自的利益,具有自私性.在应用层组播中,主机用户一方面获取接收组播数据的利益,另一方面也要承担转发组播数据给其他主机用户的负担.因此,自私的主机用户总是希望在获得尽可能多的接收组播数据利益的同时,尽量减少转发组播数据的负担,而不一定严格按照组播协议的规定来执行.

应用层组播协议的工作阶段主要分为 3 个:控制结构的维护、节点信息的收集和数据结构的构造.在控制结构维护阶段,每个节点通过被其他节点告知或者随机探测[20,21]的方式获知其他参与组播会话的节点,并选择一部分节点作为自己的邻居节点.在节点信息收集阶段,每个节点通过被动通告或者主动测量的方式获取邻居节点以及与邻居节点之间的虚链路的有关信息,比如邻居节点获取组播数据的延迟,出口带宽,虚链路的延迟或者代价,等等.在数据结构构造阶段,根据收集到的节点相关信息,对基于树状的应用层组播而言,每个节点向某个邻居节点发出父节点请求并且在自己的出口带宽范围内接受其他节点发来的父节点请求;对基于网状的应用层组播而言,每个节点向某个邻居节点发出分段接收请求,并且在自己的出口带宽范围内接受其他节点发来的分段接收请求.

因此,主机用户的自私性研究也可以按照应用层组播协议的 3 个工作阶段来分别讨论.在这里,我们并没有讨论组播数据传输过程中的用户自私性问题,这是因为:对基于树状的应用层组播而言,在组播树生成之后,数据就沿着组播树单向传播,节点之间并不存在控制信息交互,节点的自私性行为完全可以在组播数据结构构造过程中体现;对基于网状的应用层组播而言,节点之间对数据段的请求和发送是放在数据结构的构造阶段来讨论的,因为网状数据结构中组播数据的实际传输路径只有通过每个分段的传播路径才能具体体现.下面我们分别分析主机用户在控制结构维护、节点信息收集和数据结构构造这 3 个阶段的自私性行为.

在控制结构维护阶段,一个节点的邻居节点数量越多,维护邻居节点的开销越大,但到达其他节点的伸展度会减小;反之,维护邻居节点的开销越小,而到达其他节点的伸展度会增大.自私节点会综合考虑维护邻居节点的开销和到达其他节点的伸展度,从而选择数量合适的节点作为邻居节点,以使得自己维护控制结构的开销最小.

在节点信息收集阶段,节点可以在通告自己的私有信息或者在其他节点做主动测量的过程中进行欺骗以期待在组播数据结构中获取更好的位置,在更好地接收组播数据的同时减少为其他节点转发数据的负担.比如,某个节点可以在通告自己接收数据的延迟时加以夸大,或者在其他节点通过 Ping 命令测量相互之间的距离时把 Ping 报文保存一小段时间再返回,这样都可以减少收到其他节点的父节点请求或者分段接收请求的机会;节点还可以隐瞒自己的出口带宽以减小数据转发代价;中继节点还可以夸大自己的中继代价以企图收取更多的费用.

在数据结构构造阶段,无论是基于树状的应用层组播中节点从多个可选的邻居节点里选择一个发送父节点请求,还是基于网状的应用层组播中节点从多个拥有某数据段的伙伴节点中选择一个来发送分段接收请求,在此选择过程中,自私节点都可以采用策略来尽可能地提高自身获取组播数据的质量.与此同时,基于树状的应

用层组播中节点在自己的出口带宽范围内接受父节点请求,以及基于网状的应用层组播中节点在自己的出口带宽范围内接受分段接收请求,自私节点也可以选择最优的接受策略以最大化自己在组播会话过程中的利益.

目前的各种关于应用层组播用户自私性的研究都可以归结到以上 3 类问题中的某一类.它们或者是分析主机用户的自私性对整体应用层组播会话的影响,或者是设计防御主机用户自私性行为的机制与算法.下面分别对这 3 类研究进行综述.

3 控制结构维护阶段的用户自私性研究

在控制结构维护阶段,用户自私性主要体现在选择合适数量的邻居节点来最小化自己维护控制结构的总开销[22−24].每个参与应用层组播会话的节点需要从其他节点中选取一部分作为邻居节点,以保证与所有其他节点之间的可达性.而每个节点维护邻居节点是需要付出一定开销的.假设节点 i 的邻居节点集合为 S_i,其维护一个邻居节点的开销为 α,节点 j 到节点 i 的伸展度(伸展度的定义见第 1 节)表示为 $Strech(j,i)$,则节点 i 维护控制结构的开销为

$$c_i = \alpha |S_i| + \sum_{j \neq i} Strech(j,i).$$

显然,节点 i 多维护一个邻居 k,k 到 i 的伸展度从 $Strech(k,i)>1$ 减少为 $Strech(k,i)=1$,而维护邻居的开销增加 α.因此,自私的节点 i 需要维护恰当的邻居,使得自己的开销最小.应用层组播会话控制结构维护的总开销是所有节点维护控制结构的开销之和,即

$$C = \sum_i c_i .$$

如果不考虑主机用户的自私性,应用层组播会话控制结构的维护总开销可以遵守组播协议规定而达到最小,即生成最优的控制结构.然而,由于自私用户相互之间并不合作,可能存在一种纳什均衡,此时,所有用户都不愿意改变自己的邻居维护策略(因为在假定其他用户都不改变策略的情况下,自己改变策略将使得自己维护控制结构的开销增大),但组播会话控制结构维护总开销却大于理想的相互合作的情况下的总开销.最坏纳什均衡下维护组播会话控制结构的总开销与理想情况下的比值称为“无秩序代价(price of anarchy)”.

Fabrikant 等人证明了当$\alpha<2$ 或$\alpha>n^2$ 时(这里,n 为参与组播会话的节点总数),无秩序代价是一个常数;而当$2\le\alpha\le n^2$ 时,无秩序代价的上限值是 $O\sqrt{\alpha}$ [22].Albers 等人进一步证明,当$\alpha\ge 12n\lceil \log n\rceil$时,无秩序代价不大于 1.5,并且随着 α的增大而趋近于 1;对于任意大小的α,无秩序代价的上限为 $O\left(1+\left(\min\left\{\frac{n^2}{\alpha},\frac{\alpha^2}{n}\right\}\right)^{1/3}\right)$ [23].因此,当 $\alpha\in O(\sqrt{n})$时,无秩序代价接近于一个常数;当 $\alpha\in[\sqrt{n},n]$时,无秩序代价逐渐增大,直到$\alpha=n$ 时达到最大值 $O(n^{1/3})$;当$\alpha>n$ 时,无秩序代价逐渐减小.Moscibroda 等人证明了在没有节点移动或翻转的情况下,节点的自私行为甚至有可能使网络的控制结构无法收敛到稳定的状态[24].但到目前为止,并没有一个解决控制结构维护中用户自私性问题的具体方案.

4 节点信息收集阶段的用户自私性研究

在节点信息收集阶段,节点之间通过大量的信息交互以获取其他节点以及节点之间的虚链路的相关信息,为数据结构的构造做准备.自私用户为了提高自己的利益,在这一阶段可能存在着大量的欺骗行为,比如进行距离欺骗[25,26]、吞吐量欺骗[27,28]、中继代价欺骗[29]等.目前的这些研究有的是对用户自私性行为对应用层组播整体性能的影响进行分析,有的则提出了防御这些用户自私性行为的机制与算法.

4.1 距离欺骗

在许多应用层组播协议中,出于可扩展性的考虑,要求每个节点测量自己与其他节点之间的距离,并把这些距离测量值报告给某些中心节点或者利用这些测量值来进行决策.在距离测量的过程中,一些节点可能进行距离欺骗,以在组播树上获取更好的位置.这样的应用层组播协议包括 TBCP[9],NICE[10]等.欺骗节点进行距离欺骗

的原则是尽量减少到数据源节点的距离以增大成为源节点的子节点的几率,同时尽量夸大到其他普通接收节点的距离以减少成为其他节点的父节点的机会.欺骗节点的一个简单欺骗方法是:把自己与源节点之间的距离总是谎报为 0;把自己与其他普通接收节点之间的距离总是加上 10 秒,同时在收到其他节点的测量报文时延迟 10 秒回复,以避免反向路径延迟相差太大而被发现欺骗行为.

Mathy 等人通过模拟实验发现,简单的距离欺骗对应用层组播的性能有着比较大的负面影响[25].在部分节点进行距离欺骗时,应用层组播的链路强度可能达到节点诚实情况下的 3 倍左右,而伸展度可能达到节点诚实情况下的 30 倍左右.

Li 等人进一步分析了距离欺骗对应用层组播树稳定性的影响[26].距离欺骗对应用层组播树稳定性的影响可以从两个方面来衡量.一是欺骗本身造成的整棵组播树结构的改变,二是欺骗后各个节点与欺骗前相比在组播树中位置的变化.前者从宏观上反映了节点欺骗给应用层组播带来的组播树切换的开销,而后者从微观上反映了各个节点在欺骗前后的变化,主要是它们从节点欺骗中获取的利益或者损失的利益.模拟实验表明,在部分节点进行距离欺骗的情况下,应用层组播树中甚至会有超过 50%的虚链路发生改变.欺骗节点将会在组播树中取得更好的位置,获取更多利益,而其他诚实节点和源节点的利益将被损害.具体来讲,欺骗节点在组播树中会更靠近源节点,孩子数量会减少,而其他诚实节点将会在组播树上远离源节点,并且诚实节点和源节点的孩子都会增加.由于欺骗节点从距离欺骗中能获取更多利益,因此它们会不断进行欺骗,造成组播树的不断重建,而重建组播树的开销是比较大的.

应用层组播的一个重要的设计初衷是为了减少服务器的负载,但距离欺骗不但影响了组播会话的总体性能,还大大增加了源节点的负担,这与应用层组播的初衷是相悖的.因此,距离欺骗无论对于应用层组播树的性能,还是对应用层组播树的稳定性,都有着很大的危害.但目前还没有设计具体的防御距离欺骗的方法,这将是下一步值得研究的工作.

4.2 吞吐量欺骗

在应用层组播中,节点需要贡献自己的转发资源才能使整个应用层组播会话得以实现.只有在所有节点都如实贡献自己的吞吐量时,应用层组播会话的总体性能才能达到最优.但自私节点可能为了减少转发开销而不愿意如实贡献自己的转发资源,这可以理解为节点对自己的吞吐量进行欺骗.实验表明,如果节点之间资源贡献的合作率低于 30%,应用层组播会话的质量就会变得非常差.为了防御自私节点的吞吐量欺骗,研究者进行了相应的机制与算法设计.

Habib 等人设计了一种基于区分服务的鼓励用户贡献转发资源的激励机制[27].其主要思想是:记录每个节点给其他节点转发数据的贡献并对应为一定的分值,又把分值对应到一定的优先级,并被应用在每个节点选择 Peer 节点的过程中.规定每个节点都可以选择贡献分值与自己相当或比自己低的节点作为 Peer 节点为自己提供数据.因此,贡献分值越高的节点,选择 Peer 节点的范围越广,能选择到更好的 Peer 节点的机会也就越大.这样,做转发贡献较多的节点将可能获得较好的服务质量.自私节点还可以根据预期的服务质量来动态调整自己的贡献级别.具体方法如下:节点通过采样其他节点的贡献分值来预测自己的贡献分值在所有用户中的优先级情况,以此进一步预测自己在应用层组播会话中获得的服务质量情况.反之,也就可以根据预期的服务质量来调整自己的贡献级别.实验表明,在引入该激励机制以后,应用层组播会话的总体性能得到了很大提高.但这一激励机制存在一个重要的缺陷:假设节点已经得到了预期服务质量,就没有动力进一步贡献资源了.在一些节点能力差距较大的网络环境中,这种方法不能最大限度地利用能力较强的节点的转发资源以优化组播会话.

Yuen 等人则基于博弈论中的 VCG 机制[30−32]设计了一种基于补偿的激励机制[28].其主要思想是:每个节点的总利益由两部分组成:获取组播数据的利益和额外支付的补偿.额外支付的补偿通过 VCG 机制来计算.VCG 机制的基本算法是:每个个体的补偿根据其对所有其他个体的影响来计算.假设 s 表示所有个体采取的策略的集合,s^{-i} 表示除了节点 i 之外所有个体采取的策略的集合,$U_k(s)$表示节点 k 在策略集合 s 下的效益,$P_k(s)$表示节点 k 在策略集合 s 下得到的补偿,则 VCG 机制下节点 i 的补偿为

$$P_i(s)=\sum_{j\neq i}U_j(s)-\sum_{j\neq i}U_j(s^{-i}).$$

基于 VCG 机制的补偿算法设计保证了只有在节点诚实通告私有信息的情况下,其总利益才能达到最大.因此,在这种机制下,每个理性节点都会诚实地通告其私有信息.值得注意的是,由于一个节点的补偿是根据其对其他节点的重要性来计算的,它可能为正,也可能为负.如果由于补偿为负值而导致节点的总利益为负,则节点可以选择离开组播会话.我们还设计了分布式的补偿计算算法,由每个节点计算自己应该得到的补偿.随着应用层网络环境的缓慢动态变化,分布式补偿计算算法会在有限步骤内收敛,计算出每个节点的稳定的补偿值,应用层组播树也会向理想的组播树收敛.但这种机制存在的缺点是每个节点的补偿由节点自身来计算,因此自私节点可能进一步欺骗自己的补偿值以获取更多的利益.

4.3 中继代价欺骗

在一些应用层组播模型中,组播中继节点和组播接收者的角色是分离的.组播接收者在接收数据的同时,需要向中继节点付转发费用.在这种情况下,中继节点可能欺骗自己的转发代价以获取较高的费用.为了获得组播会话整体性能的最优,需要防御中继节点的这种欺骗行为.Wang 等人设计了合适的支付中继节点的付费算法来解决这一问题[29].在这种应用层组播模型中,从所有参与节点组成的拓扑中生成一棵覆盖所有接收者的最优组播树是一个 Steiner Tree 问题[33,34].众所周知,这一问题是 NP 难问题,而 VCG 机制所适用的情景要求产出函数必须是最优的,而不能是近似最优的[35].因此,VCG 机制并不适合于这种网络模型,需要设计一种新的激励性的付费算法.我们证明了这种付费算法的存在并进行了设计.

还需要讨论由谁来支付这些中继节点的费用.一种收费方法是,由第三方如所有接收者所属的组织来支付,另一种收费方法是,由所有数据接收者公平地分担数据的转发费用.第 1 种方法比较简单,只要根据激励性的付费算法计算出了每个中继节点应该得到的费用即可.而第 2 种方法相对比较复杂.如果采用第 2 种方法,费用分担算法应该满足以下原则:1) 每个接收者的付费不应该为负值,即接收者不能因为接收数据而获取费用;2) 随着接收者集合的增大,单个节点的付费不应该增加;3) 如果接收者的集合为 R,单个接收者单播情况下付费为 c,则该接收者的付费不能小于 $c/|R|$;4) 向所有中继节点支付的费用应该等于向所有接收者收取的费用.但我们证明,在任意给定一种网络拓扑和对应的激励性付费算法的情况下,可能并不存在一种这样的公平的收费方法;不过可以针对某些具体的例子设计对应的收费算法.但这一算法也存在一个缺点,即不能保证费用计算算法本身能被自私接收者忠实地执行.

5 数据结构构造阶段的用户自私性研究

在数据结构构造阶段,对基于树状的应用层组播而言,节点进行父节点选择并且接受其他节点发来的父节点请求;对基于网状的应用层组播而言,节点选择某个数据段的提供节点并接受其他节点发来的数据段接收请求.这一过程中,节点的自私性体现在采用有利于提高自己在组播会话中的总体利益的策略.

Tan 等人对引入激励机制后应用层组播的数据结构构造阶段的用户自私性进行了讨论[36].在我们的应用层组播模型中,组播数据的发送分为若干个周期,每个周期内节点请求一个数据段并且接受其他节点的数据段请求.每个节点维护一个积分,在向其他节点发送数据段时挣取积分,而在请求数据段时耗用积分.数据段的请求和发送采用“市场”机制,即在每个周期开始时,每个数据段请求节点分别向数据段提供节点出价,而在数据段提供节点的带宽允许范围内,那些出价较高的请求节点将赢得此次数据段的发送.节点在每个周期内赢得的积分可以累加,并在后续周期内都可以使用.为了鼓励即将退出组播会话的节点继续作出贡献,节点重新加入组播会话后可以使用其在以前的会话中累积的积分.

该模型中数据结构的构造包括数据段请求和数据段发送.对数据段发送而言,每个数据段提供节点采用的策略很简单,如上所述,在带宽允许范围内向那些出价较高的请求节点发送数据段,即“最高竞价”策略.而对于数据段请求而言,每个请求节点需要从候选的数据段提供节点中选择一个来发送请求.至少有 3 种请求策略:请求离源节点最近的节点(一旦被接受,则获取的效益最大)、请求可用带宽最大的节点(被接受的概率最大),以及前

两种策略的混合.自私节点一般会选择第 1 种策略以保证自己获取的利益最大,但我们证明了存在一种纳什均衡,在这种状态下所有节点都会选择第 2 种来最大化自己的利益.从实际情况来看,自私节点往往会采用第 3 种策略,即首先选择第 1 种策略,如果请求被数据段提供节点拒绝,再选择第 2 种策略.从应用层组播会话的整体性能来看,显然第 2 种策略更优,因为这种情况下可以形成一棵比较平衡的组播树.

Li 等人对没有任何激励机制情况下应用层组播数据结构构造阶段的用户自私性问题进行了分析[37].以基于树状的应用层组播为例,分为父节点请求和孩子节点接受两个阶段.在父节点请求阶段,自私节点有随机选择和基于 QoS(quality of service)的选择两种方式,前者从控制结构的所有邻居节点中随机选择一个作为父节点,而后者从邻居节点中选择一个最能优化自己 QoS 的节点作为父节点.显然,自私的节点都会选择后一种方式,而这种方式也有利于组播会话整体性能的优化,因为其不但提高了选择节点的 QoS,也间接提高了那些将来可能选择这个选择节点作为父节点的节点的 QoS.

而在孩子节点接受阶段,自私节点有 4 种典型的处理策略:1) 不接受任何孩子节点;2) 在出口带宽范围内随机接受孩子节点;3) 在出口带宽范围内优先选择那些“能力”强的节点为孩子节点;4) 在出口带宽范围内优先选择那些曾经为自己做贡献比较多的节点为孩子节点.由于每个节点的策略及其邻居节点的策略都可能影响节点自身将来在组播会话中的利益,因此可以用一个多方参与的博弈来分析这一问题.我们通过占优策略分析,证明了每个自私节点都将选择策略 3)来优化自己将来在组播会话中的利益,而这一策略又恰好与应用层组播会话总体利益的优化相一致.

从以上研究来看,与用户在控制结构维护阶段和节点信息收集阶段的自私性不同,用户在数据结构构造阶段的自私策略往往对应用层组播会话整体利益的优化是有利的.

6 已有研究的比较

应用层组播协议主要有 3 个工作阶段:控制结构的维护、节点信息的收集和数据结构的构造.前面 3 节所综述的应用层组播用户自私性的研究也主要集中在这 3 个阶段.表 1 对前面提到的各种关于应用层组播用户自私性的研究进行了比较.

从表 1 可以看出,目前的研究大部分是对用户的自私行为及其对整体组播会话的影响进行分析.研究表明,用户的不同自私行为对应用层组播会话的整体性能的影响可能是消极的,也可能是积极的.对于可能带来消极影响的用户自私性行为,一些研究还提出了具体的解决方案.这些防御机制主要可以分为两类,一类是对自私节点进行区分服务,另一类是对自私节点提供额外的补偿或付费.

7 结论和下一步的研究展望

随着互联网的飞速发展,组播及其支撑的应用必将得到越来越广泛的应用.应用层组播不但比网络层组播更容易部署,而且也更加灵活,可以作为网络层组播的重要补充.但与网络层组播相比,应用层组播的一个重要特点是组成组播结构的节点是具有独立利益和决策的主机,因此应用层组播协议的设计应该充分考虑到主机用户的自私性.目前,关于应用层组播用户自私性的研究主要集中在应用层组播协议的 3 个工作阶段:控制结构的维护、节点信息的收集和数据结构的构造.这些研究或者是分析用户自私性行为对应用层组播会话整体性能的影响,或者是设计防御用户自私行为的机制与算法.

用户自私性的研究对设计可信任的、鲁棒的应用层组播会话极为重要.虽然目前已有不少这方面的研究,但还有以下几方面值得作进一步的探索.第一,对于一些消极的用户自私行为还没有很好的解决方案.比如,文献[26,27]所讨论的距离欺骗还有明确的防御机制.而已经提出的一些自私行为防御机制往往都需要较大的开销,如文献[28]所采用的区分服务机制中,需要有可行的量化节点转发贡献的机制,而在文献[29,30]所采用的额外补偿的机制中,又需要应用层组播会话维护一个“费用”流通系统.这些方案往往还没有考虑激励机制本身能否被自私用户忠实地执行.第二,缺乏应用层组播用户信任的总体框架.目前的研究主要都是分析一些具体的自私性问题或者给出具体的解决方案,而没有提出一个总体框架来解决应用层组播用户在各个阶段可能的自私

性行为,包括联合欺骗行为等.第三,还没有一个充分考虑用户自私性问题的应用层组播协议.现有的研究大多只是针对问题本身.设计一个可运行的、可信任的应用层组播协议,是应用层组播用户自私性研究的根本目的所在,也是下一步的重要研究方向.

Table 1 Comparison of researches on selfishness in application-layer multicast

表 1 应用层组播用户自私性研究比较

Literature	Selfish behavior of individual nodes	Working step of ALM	Impact of selfish behavior on ALM session	Defending solution proposed?	Solution type
[23]	Selfishly selecting neighbors	Maintenance of control structure	Negative	No	–
[24]	Selfishly selecting neighbors	Maintenance of control structure	Negative	No	–
[25]	Selfishly selecting neighbors	Maintenance of control structure	Negative	No	–
[26]	Distance cheating	Collection of node information	Negative	No	–
[27]	Distance cheating	Collection of node information	Negative	No	–
[28]	Node throughput cheating	Collection of node information	Negative	Yes	Service differentiation
[29]	Node throughput cheating	Collection of node information	Negative	Yes	Additional payment
[30]	Relaying cost cheating	Collection of node information	Negative	Yes	Additional payment
[37]	Selfish segment request and segment response	Construction of data structure	Positive	–	–
[38]	Selfish parent request and children acceptance	Construction of data structure	Positive	–	–

References:

[1] Deering S. Multicast routing in internetworks and extended LANs. In: Landweber L, ed. Proc. of the Communications Architectures and Protocols. Stanford: ACM Press, 1988. 55–64.

[2] Francis P. Yoid: Extending the Internet multicast architecture. 2006. http://www.icir.org/yoid/

[3] Chu YH, Rao SG, Zhang H. A case for end system multicast. IEEE Journal on Selected Areas in Communication, 2002,20(8):1456−1471.

[4] Chawathe Y. Scattercast: An architecture for internet broadcast distribution as an infrastructure service [Ph.D. Thesis]. Berkekey: University of California, 2000.

[5] Jain S, Mahajan R, Wetherall D, Borriello G. Scalable self-organizing overlay. Technical Report, Washington University, 2000.

[6] Kwon M, Fahmy S. Topology-Aware overlay networks for group communication. 2006. http://www.cs.rit.edu/~jmk/papers/pub.html

[7] Zhang B, Jamin S, Zhang L. Host multicast: A framework for delivering multicast to end users. 2006. http://www.peer-to-peer.info/bibliography/zhang2002hmtp

[8] Pendarakis D, Shi S, Verma D, Waldvogel M. ALMI: An application level multicast infrastructure. 2006. http://citeseer.ist.psu.edu/pendarakis00almi.html

[9] Mathy L, Canonico R, Hutchison D. An overlay tree building control protocol. 2006. http://portal.acm.org/citation.cfm?id=747484&dl=ACM&coll=&CFID=15151515&CFTOKEN=6184618

[10] Banerjee S, Bhattacharjee B, Kommareddy C. Scalable application layer multicast. 2006. http://portal.acm.org/citation.cfm?coll=GUIDE&dl=GUIDE&id=633045

[11] Tran DA, Hua KA, Do T. Zigzag: An efficient peer-to-peer scheme for media streaming. 2006. http://citeseer.ist.psu.edu/tran03zigzag.html

[12] Castro M, Druschel P, Kermarrec AM, Rowstron A. SCRIBE: A large-scale and decentralized application-level multicast infrastructure. IEEE Journal on Selected Areas in Communications. 2002,20(8):1489−1499.

[13] Ratnasamy S, Handley M, Karp M, Shenker S. Application-Level multicast using content-addressable networks. 2006 http://citeseer.ist.psu.edu/ratnasamy01applicationlevel.html

[14] Castro M, Druschel P, Kermarrec A, Nandi A, Rowstron A, Singh A. SplitStream: High-Bandwidth content distribution in cooperative environments. 2006. http://citeseer.ist.psu.edu/castro03splitstream.html

[15] Kostic D, Rodriguez A, Albrecht J, Vahdat A. Bullet: High bandwidth data dissemination using an overlay mesh. 2006. http://citeseer.ist.psu.edu/684715.html

[16] Padmanabhan VN, Wang HJ, Chou PA, Sripanidkulchai K. Distributing streaming media content using cooperative networking. Technical Report, MSR-TR-2002-37, Microsoft Research, 2002.

[17] Rejaie R, Stafford S. A framework for architecting peer-to-peer receiver-driven overlays. 2006. http://portal.acm.org/citation.cfm?id=1005858&dl=acm&coll=&CFID=15151515&CFTOKEN=6184618

[18] Zhang X, Liu J, Li B, Yum TP. DONet: A data-driven overlay network for efficient live media streaming. 2006. http://citeseer.ist.psu.edu/zhang05coolstreamingdonet.html

[19] Hefeeda M, Habib A, Botev B, Xu D, Bhargava B. PROMISE: Peer-to-Peer media streaming using CollectCast. 2006. http://citeseer.ist.psu.edu/672832.html

[20] Banerjee S, Lee S, Bhattacharjee B, Srinivasan A. Resilient multicast using overlays. 2006. http://www.cs.umd.edu/~slee/pubs/prm-sigmetrics03.pdf

[21] Eugster P, Guerraoui R, Kermarrec AM, Massoulie L. From epidemics to distributed computing. IEEE Computer, 2004,37(5):60−67.

[22] Fabrikant A, Luthra A, Maneva E, Papadimitriou CH, Shenker S. On a network creation game. 2006. http://www.cs.berkeley.edu/~alexf/papers/flmps03.pdf

[23] Albers S, Eilts S, EvenDar E, Mansour Y, Roditty L. On nash equilibria for a network creation game. 2006. http://www.aladdin.cs.cmu.edu/workshops/netdes2/slides/albers.pdf

[24] Moscibroda T, Schmid S, Wattenhofer R. On the topologies formed by selfish peers. 2006. http://iptps06.cs.ucsb.edu/papers/Schmid06.pdf

[25] Mathy L, Blundell N. Impact of simple cheating in application-level multicast. 2006. http://www.ieee-infocom.org/2004/Papers/28_3.pdf

[26] Li D, Cui Y, Xu K, Wu J. Impact of receiver cheating on the stability of ALM tree. 2006. http://ieeexplore.ieee.org/iel5/10511/33286/01577725.pdf?arnumber=1577725

[27] Habib A, Chuang J. Incentive mechanism for peer-to-peer media streaming. 2006. http://p2pecon.berkeley.edu/pub/HC-IWQOS04.pdf

[28] Yuen S, Li B. Strategyproof mechanisms for dynamic multicast tree formation in overlay networks. 2006. http://www.eecg.toronto.edu/~bli/papers/yuen-infocom05.pdf

[29] Wang W, Li X, Suny Z, Wang Y. Design multicast protocols for non-cooperative networks. 2006. http://www.cs.iit.edu/~xli/paper/Conf/multicast-INFO05.pdf

[30] Vickrey W. Counterspeculation, auctions and competitive sealed tenders. Journal of Finance, 1961,16(1):8−37.

[31] Clarke EH. Multipart pricing of public goods. Public Choice, 1971,11(2):17−33.

[32] Groves T. Incentives in teams. Econometrica, 1973,41(4):617−631.

[33] Klein P, Ravi R. A nearly best-possible approximation algorithm for node-weighted steiner trees. Technology Report, S-92-54, 1992.

[34] Guha S, Khuller S. Improved methods for approximating node weighted steiner trees and connected dominating sets. 2006. http://www.springerlink.com/index/XJ2CGKEJGDWDC1TP.pdf

[35] Colell AM, Whinston M, Green J. Microeconomic Theory. New York: Oxford University Press, 1995.

[36] Tan G, Jarvis SA. A payment-based incentive and service differentiation mechanism for peer-to-peer streaming broadcast. 2006. http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=4015719&arnumber=4015732&count=54&index=11

[37] Li D, Wu J, Cui Y, Liu J. QoS-Aware streaming in overlay multicast considering the selfishness in construction action. 2006. http://netlab.cs.tsinghua.edu.cn/~lidan

李丹(1981－),男,四川阆中人,博士生,主要研究领域为计算机网络体系结构,P2P,应用层组播.

吴建平(1953－),男,教授,博士生导师,CCF 高级会员,主要研究领域为计算机网络体系结构,协议工程学,互联网络.

崔勇(1976－),男,博士,助理研究员,主要研究领域为计算机网络体系结构,服务质量控制,路由算法,性能评价.