ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.9, September 2008, pp.2376−2388
DOI: 10.3724/SP.J.1001.2008.02376

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

分点:无结构对等网络的拓扑关键点*

李振华, 陈贵海+, 邱彤庆

(南京大学 计算机软件新技术国家重点实验室,江苏 南京 210093)

Partition Nodes: Topologically-Critical Nodes of Unstructured Peer-to-Peer Networks

LI Zhen-Hua, CHEN Gui-Hai+, QIU Tong-Qing

(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China)

+ Corresponding author: E-mail: gchen@nju.edu.cn

Li ZH, Chen GH, Qiu TQ. Partition nodes: Topologically-Critical nodes of unstructured peer-to-peer networks. *Journal of Software*, 2008,19(9):2376−2388. http://www.jos.org.cn/1000-9825/19/2376.htm

Abstract: Although nodes in peer-to-peer networks are viewed as equal entities in function, some of them have much more importance than others in the aspect of overlay topology. This paper introduces the concept of "partition node" to describe the topologically-critical nodes, whose failure may potentially lead to overlay partitioning. Then a simple, effective and distributed method to detect and avoid partition nodes, is proposed. The results of simulation show that the proposed method can optimize the overlay topology and remarkably improve the fault tolerance of unstructured P2P systems under a dynamic environment.

Key words: unstructured peer-to-peer network; partition node; overlay partitioning; fault tolerance

摘 要: 虽然对等网络中的结点在功能上互相平等,但其中某些结点对于覆盖网拓扑却有重要意义.提出了分点概念来描述对等网络的拓扑关键点,这些结点的失效很可能导致覆盖网的分割.设计了一套简单、有效、分布式的分点检测和避免方法.模拟实验的结果表明,该方法可以优化覆盖网拓扑结构,并显著地提高无结构对等网络在动态环境下的容错性.

关键词: 无结构对等网络;分点;覆盖网分割;容错性

中图法分类号: TP393 **文献标识码**: A

分布式、自组织的对等网络在过去几年里以惊人的速度发展到巨大的规模,并被广泛应用到多个领域,如资源共享[1−4]、实时消息[5]、协同工作[6]、分布式计算[7]等等.虽然对等网络中的结点在功能上互相平等,但实际上,其中的某些结点对于整个网络却有着特殊意义.一方面,那些作为两个或多个独立子网之间唯一通道的结点对覆盖网拓扑结构有重要影响,其失效很可能导致覆盖网被分割,相反地,覆盖网被分割往往可以追溯到它们的失效.另一方面,那些成为系统性能"瓶颈"的结点,其能力影响到系统在动态环境下工作的多方面性能,如查询成功率、数据(结点)定位跳数、通信拥塞和时延等等.我们称上述具有特殊意义的结点为"拓扑关键点".如果能在

* Supported by the National Natural Science Foundation of China under Grant Nos.60573131, 60673154, 60721002 (国家自然科学基金); the National Basic Research Program of China under Grant No.2006CB303004 (国家重点基础研究发展计划(973)); the Jiangsu Province High-Tech Research Project of China under Grant No.BK2007039 (江苏省高技术研究项目)

Received 2007-01-27; Accepted 2007-08-03

对等网络中以分布式的方法有效地检测到拓扑关键点并加以合理地避免,就能从本质上增强系统对覆盖网分割的抵抗力,同时显著地提高系统的容错性.

最基本的问题是:什么样的结点是拓扑关键点?如果将对等网络看成一幅图(有向图或无向图,本文立出于简化通常考虑无向图,但所提出的概念、方法对有向图是同样适用的),很自然地,会想到图论中的“割点”.割点的概念可以表述如下:如果在一幅连通图 G 中删去点 C 后,图 G 被分割成两个或多个独立的连通子图,那么点 C 就是图 G 的一个割点.图 1 是割点的简单例子.

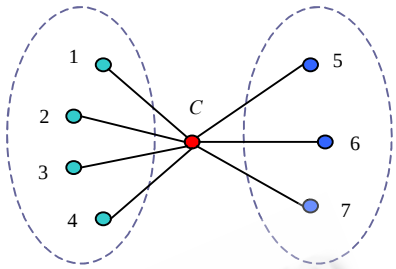


Fig.1 Node C is a cut node (only C's neighbors and edges are drawn here)

图 1 点 C 为割点(仅画出 C 的邻居和边,图中其他结点和边均未画出)

作为对等网络领域中重要而实用的部分,无结构对等网络[1−3]有其独特的性质:覆盖网组织松散、没有固定的结构,通常采用洪泛式的路由方法或者基于洪泛的改进方法(如扩展环、随机走,等等).为了限制通信开销,无结构对等网络通常会给路由消息设定跳数限制(time to live,简称 TTL),以使其局限在一定范围内,因此,其路由具有明显的局部性和不确定性,不能覆盖整个网络,也就无法准确地定位结点或者数据对象.

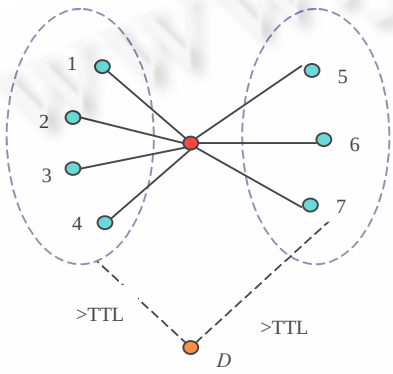


Fig.2 Node C is not a cut node, but it is a partition node and a real topologically-critical node

图 2 点 C 不是割点,但它是“分点”,并且是名副其实的拓扑关键点

鉴于无结构对等网络的工作特点,我们认为割点不能充分、合理地描述无结构对等网络的拓扑关键点,因为割点是全局的、静止的,而无结构对等网络则是在动态环境下分布式工作、局部性路由的,既不可能静止地获取网络全局状态,也不可能作全局性的定位.图 2 的简单例子说明了割点不适合用来描述无结构对等网络的拓扑关键点.

图 2 中,点 C 并非割点,因为还有点 D 连接着左右两个子图.但是,D 与左右两个子图的距离超过跳数限制 TTL,所以如果 C 失效,左右子图之间虽然从图论上看仍然连通,但对无结构对等网络而言,却是相互分割的,因为两个子图之间从此没有结点可以互相定位到.在这种情况下,D 的存在对左右子图并无意义,C 既是连接左右子图的唯一通道,也是左右子图间通信的性能瓶颈,是名副其实的拓扑关键点,此时,我们称 C 为分割结点,下文都简称为分点.

本文首先提出了分点这一概念来描述无结构对等网络的拓扑关键点,本质上它是对割点概念的扩展,使其真正符合无结构对等网络的工作特点和实际需求.基于分点的概念,我们设计了一套简单、有效、分布式的分点检测和避免方法,具体来说,该方法有如下特点:

(1) 分布式:不需要任何集中式的控制,每个结点独立地运行自己的检测与避免算法;

(2) 低开销:分点检测、避免的通信开销低,并且由分点避免操作导致的新分点数非常少;

(3) 抗分割:分点的避免能够有效地防止覆盖网被分割,这也是本文研究分点的初衷;

(4) 高效率:为了避免分点而对网络组织结构所作的调整,优化了覆盖网拓扑,减少了性能瓶颈,从而提升了系统查询成功率;

(5) 高容错:去掉覆盖网中分点这一关键而脆弱的元素后,即使面临结点不断失效的高动态性环境,系统查询成功率仍然能够保持基本稳定,不会出现急剧下降的情况;

(6) 异构性:我们为避免分点而给结点加边,当结点连接度达到上限时给结点减边,都是以异构性的开发作

为主要的指导原则,从而提高了系统的负载均衡,让每个成员尽量各尽所能.

本文第 1 节回顾对等网络领域针对覆盖网分割问题的研究状况.第 2 节解释分点的概念.第 3 节讲述分点检测的步骤.第 4 节讲述为避免分点而进行的加边、减边原则和方法,并分析分点检测和避免的开销.第 5 节模拟实验测量分点对覆盖网拓扑的意义、分点避免算法的有效性及其对系统各方面性能的提升.第 6 节总结全文并指出进一步的研究工作.

1 相关工作

对等网络的“覆盖网分割”问题及与之相关的容错性问题一直被研究者广泛重视.Saroiu 等人[8]通过细致的测量来研究无结构对等网络 Gnutella 的容错性,其实验结果表明:在 Gnutella 中,绝大多数结点连接度很低,只有少数结点连接度非常高.更具体地说,Gnutella 符合 α=2.3 的幂律(Power-Law)分布,对随机结点失效有高容错性,但是对恶意攻击的抵抗力则非常弱.尤其是高连接度的结点失效后,容易导致覆盖网被分割.虽然[8]认识到高连接度的结点对系统容错性的重要意义,但却没有能够进一步回答到底哪部分高连接度结点属于系统拓扑关键点?哪些结点失效会导致覆盖网分割?我们认为分点是这个问题的答案.

Ripeanu 和 Foster[9]指出:如果无结构对等网络中每个结点的连接度能够尽量高于某个常数下限,就能有效地提高对抗恶意攻击的能力.这里所说的“常数下限”实际上就是一幅图 G 的最小结点度$\delta(G)$,因为图论中有一个简单而重要的不等式:点连通度$\kappa(G)$≤边连通度$\lambda(G)$≤最小结点度$\delta(G)$,所以最小结点度实际上确定了点连通度和边连通度的上限,也就间接确定了图的连通性上限.本文为避免分点而选择合适的结点加边,选择所依据的第一条原则正基于此,让最小结点度$\delta(G)$尽可能地大.

Liu 等人[10]最先从割点的角度研究无结构覆盖网的分割问题.他们设计了一种称为 CAM(connection adjacency matrix)的分布式算法来检测割点.CAM 算法有效地检测到结点间的可达关系,避免了重复的探测工作,将通信开销降得很低.然而 CAM 有一些缺点.首先,CAM 算法无法准确检测割点,除非消息路由的跳数没有限制.实际上,CAM 所检测到的“割点”并非割点,而是我们提出的“分点”.其次,为避免割点,CAM 算法从每个连通分量中随机选取一个结点(我们称该结点为其所在连通分量的“代表结点”)将它们相连,我们认为,代表结点的选取非常重要,应基于某种策略,从某个角度提高网络性能,所以本文的分点避免遵循了一种兼顾容错性提升、异构性开发的策略来选择代表结点.再次,选择了代表结点后,如何将它们相连,也是很重要的.Liu 等人[10]列举了线性链连接的例子:将代表结点线性地串成一条闭合链.线性链连接方式虽然简单,但很脆弱,对网络连通性的提高也不充分.本文对几种不同的连接方式进行了讨论,并通过实验来比较线性链连接和我们提出的带弦环连接.在任何一个网络中,结点的连接度都不可能是无穷的,既然为了避免分点而给结点加边,就应该考虑结点度超过上限时给它减边,但文献[10]没有提及这个问题.本文考虑了减边的原则和方法,并在模拟实验中作了实现.

对等网络中存在着普遍的结点异构性:结点间虽然在功能上对等,在实际能力(带宽、存储容量、计算能力等)上却存在着巨大的差异.流行的对等网络 KaZaA[2]、eDonkey[3]通过显式地设置“超结点”来开发这种异构性,而 Gia 系统[11]则通过隐式的自适应调整以形成“集群效应”来开发异构性.无论显式隐式,这些方法的指导思想都在于让结点各尽所能——能力越高、负担就越大.本文为避免分点而给结点加边,当结点连接度达到上限时给结点减边,都是以异构性的开发作为主要原则的.

虽然本文研究的是无结构对等网络中的分点,但其中大多数概念、方法对结构化对等网络同样适用,所以我们也关注了结构化网络的相关研究.Liben-Nowell 等人[12]分析了 Chord[13]在动态网络环境下的容错性,并给出了维护 Chord 覆盖网连通性的邻居信息获取速率下限.Saia 等人[14]构建了一个动态高容错的覆盖网,通过让每个结点维护 $O(\log^3 N)$项路由信息以及每次路由需要发送 $O(\log^3 N)$条消息,使得在任何时刻,绝大多数结点都能成功获取绝大多数数据对象.Loguinov 等人[15]研究了一些结构化 P2P 覆盖网(Chord,CAN[16],de Bruijn 图)的图论属性:路由性能、集群属性、等分带宽等,并提出了一种基于最优直径 de Bruijn 图的网络架构.

对于已经提出的解决覆盖网分割问题的各种方法,我们从执行时间与覆盖网分割的先后顺序的角度将它

们分成两类:主动避免与被动修复.主动避免意味着采取措施主动避免可能出现的覆盖网分割,以尽可能地保证覆盖网一直连通;被动修复意味着当发现覆盖网分割时被动地修复以合并分裂的多个子网.

(1) 主动避免

Pandurangan 等人[17]提出了一种集中式的方法来维护 P2P 覆盖网的连通性,他们使用集中式的服务器来引导新加入覆盖网的结点和哪些现存结点建立连接,离开覆盖网的结点如何更新信息,前提是要求所有结点的加入和离开都必须预先通知服务器.很明显,这违反了 P2P 的思想.上面讲到的 CAM 算法和本文提出的解决覆盖网分割问题的方法,通过周期性地检测无结构覆盖网中的拓扑关键点,然后将它们调整成普通结点,以尽量避免拓扑关键点的存在,从而缓解无结构 P2P 覆盖网的分割问题.

(2) 被动修复

很多结构化 P2P 网络的工作是基于环结构的,如果大环被分割成多个独立的小环(简称分环),就不能正常工作.Mahajan 等人[18]设计了一种能够有效检测和修复分环的方法,并对 Pastry[19]作了具体的实现.Harvey 等人[20]专门针对 SkipNet[21]的分环问题作了研究,但其设计的方法只适用于 SikpNet 这样的提供数据语义、消息路由双重局部性的特殊网络.Sit 和 Morris[22]提出了一种交叉验证路由表的方法来缓解覆盖网分割问题:每个结点每隔一段时间随机地向其路由表中的邻居发送校验查询消息,让其邻居执行此查询,然后比较自己查询的结果和邻居返回的结果,如果两者一致,可以近似地认为网络没有被分割;否则,就需要启动相应的修复机制.经典的结构化 P2P 网络 Chord 和 Pastry 中都有谈及覆盖网分割问题.在 Chord 中,一个结点 N 周期性地要求其他结点查询自己,如果其他结点不能正确地找到 N,就认为可能存在网络分割.而在 Pastry 中,结点周期性地使用 IP 多播,以扩展环路由的方式来搜索邻近的结点,如果存在网络分割,那么被分离开的结点之间很有可能互相搜索到,那时再来作子网合并.

2 分点的概念

首先定义无结构对等网络中的两种关系:定位关系和可达关系,然后基于上述关系定义分点.

定义1(定位关系). 在无结构对等网络中,如果结点A通过发送路由消息能够找到结点B,则称A能定位B,记作$A\rightarrow B$.

定位关系是不传递的,一般也不对称(除非网络可看成无向图).图3是定位关系的例子.

定义 2(可达关系). 在无结构对等网络中,如果结点 A 能够定位到结点 B,结点 B 能够定位到结点 C,则称 A 到 C 是可达的,记作 $A\rightarrow\rightarrow C$.

可达关系是传递的,如果 $P\rightarrow\rightarrow Q, Q\rightarrow\rightarrow R$,那么 $P\rightarrow\rightarrow R$;可达关系一般是不对称的(除非网络可以看成无向图).图 4 是可达关系的例子.可达关系是检测分点的基础:一个结点 n 是否为分点,取决于 n 被删去后其邻居间是否仍然都可达.

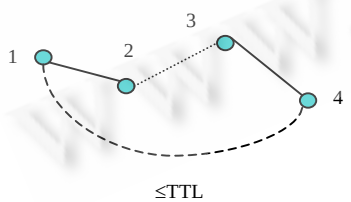


Fig.3 Node 1 can locate nodes 2, 3, 4

图3 结点1能够定位结点2,3,4

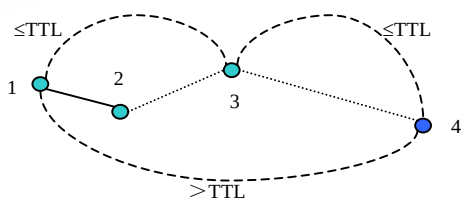


Fig.4 Node 1 can locate 2, 3, but can not locate 4; node 3 can locate 4; so node 1 can reach 4

图4 结点1能够定位2,3,但不能定位4;而结点3能够定位4;故结点1可达4

定义3(分点). 在无结构对等网络中,如果删去结点C后,C的邻居集将被分成两个或多个互不可达的子集:S_1,S_2,…,S_n(n>1),就称C为分点.

任意两个子集间的结点都不可达,而每个子集内的结点是可达的.图 5 是分点的例子,每个子集内部的边和网络中其他结点均未画出.

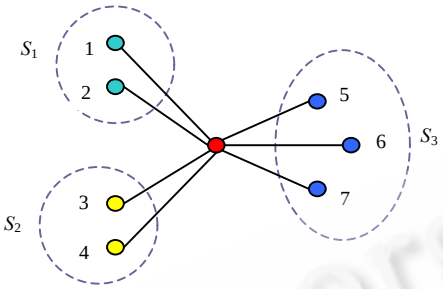


Fig.5 The partition node C's neighbor set is divided into three unreachable subsets S_1,S_2,S_3

图5 分点C的邻居集被分成3个互不可达的子集S_1,S_2,S_3

从上述定义可以看出,分点既是其邻居子集间互相连接的唯一通道,也是邻居子集间互相通信的性能瓶颈,所以分点是无结构对等网络中真正的拓扑关键点.因此,避免了分点这一脆弱而关键的元素,就能够从本质上提高系统对覆盖网分割的抵抗力,有效地提升系统容错性.

我们对无结构对等网络中分点的定义还有如下特点:

(1) 分布式:分点可以分布式地进行有效的检测和避免,检测、避免所需的信息交换只限于邻居结点之间;

(2) 局部性:分点符合无结构对等网络路由局部性的工作特点,因为所用到的定位关系、可达关系都基于局部性的路由,比起割点更具有实际意义;

(3) 兼容性:兼容割点,因为分点包含了割点.如果对 TTL 没有限制,那么分点实际上就是割点.

3 分点的检测

基于前一部分对分点的定义,分点检测的关键就是在候选分点 C 不参与消息路由的前提下,探测其邻居间的可达关系,再依据可达关系将邻居集分成互不可达的子集;若子集只有一个,那么 C 不是分点,否则,C 是分点.这就是我们设计的分点检测方法的思想.具体来说,C 为了检测自己是否为分点,分如下几步进行:

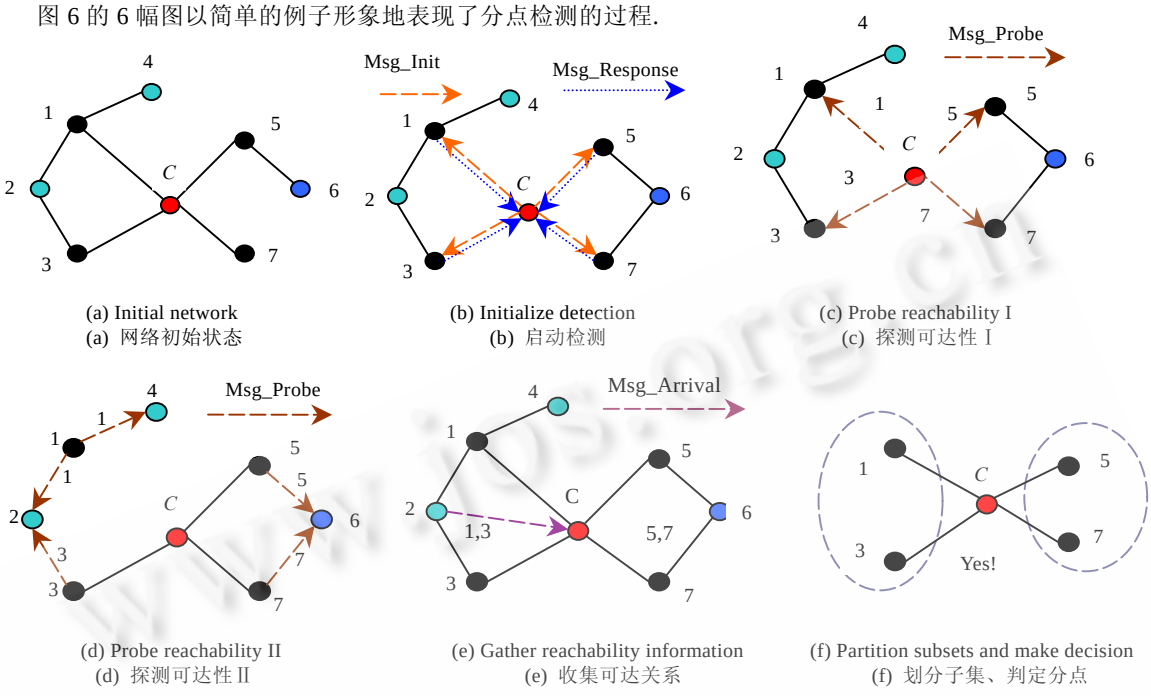
(1) 启动检测:C 向每个邻居 N_1,N_2,…,N_n 发送消息 Msg_Init 以启动检测,收到消息的邻居必须回复 Msg_Response,其中包含它的 IP 地址、带宽、时延、连接度等信息(如图 6(1)所示),这些信息是以后的工作(分点避免)所必需的.如果 C 没有收到邻居 N_i 的回复,就认为 N_i 失效,将其从路由表中删除.

(2) 探测可达性:C 给每个邻居发送探测消息 Msg_Probe(如图 6(2a)所示),此消息中包含结点 C 的地址、跳数限制 TTL 和邻居号 N_i.网络中每个结点维护一张"连接表",每个候选分点在连接表中都有其对应的一项,形如⟨候选分点地址,邻居号 1,邻居号 2,…⟩.结点 N 的连接表中,候选分点 C 所对应的项表明 C 的哪些邻居是互相可达的.当某个结点收到 Msg_Probe 消息时,如果 Msg_Probe 中的邻居号是新的,它就把该消息发送给每个邻居(除了消息发送者)(如图 6(2b)所示),并且将消息中的邻居号 N_i 添加到候选分点 C 的连接表项中.如果连接表项的邻居号不止 1 个,就给候选分点发送 Msg_Arrival 消息,此消息包含该表项中所有的邻居号,实际上就是告诉候选分点哪些邻居可达(如图 6(3)所示).候选分点给其不同邻居发出的探测消息所到达的范围是不重叠的(只在相交的那一点回发 Msg_Arrival 消息报告邻居间的可达关系),避免了重复的可达性探测.上述探测过程借鉴了 CAM[10]探测割点的思想[10],其中有更为详细的介绍和理论分析.

(3) 划分子集:C 不断收集 Msg_Arrival 消息,计算其邻居之间的可达关系,将可达的结点划分到同一个子集(这一步实际上是在计算等价类).鉴于对等网络的动态性,少数 Msg_Arrival 信息可能会丢失,所以 C 可以设置一个超时值 Timeout,在 Timeout 后认为得到了最终的划分.

(4) 判定分点:完成子集的划分后,C 根据子集的数目来判定自己是否为分点:如果所有邻居归到同一个子集,那么 C 不是分点;否则,C 是分点,进入下一步——分点的避免.

图 6 的 6 幅图以简单的例子形象地表现了分点检测的过程.



(a) Initial network
(a) 网络初始状态

(b) Initialize detection
(b) 启动检测

(c) Probe reachability I
(c) 探测可达性 I

(d) Probe reachability II
(d) 探测可达性 II

(e) Gather reachability information
(e) 收集可达关系

(f) Partition subsets and make decision
(f) 划分子集、判定分点

Fig.6 The process of partition node detection

图 6 分点检测的过程

4 分点的避免

4.1 加边以避免分点

当结点 C 判定自己是分点后,为避免成为分点,C 需要给其邻居加边以合并互不可达的多个子集.假设 C 的邻居被分成多个子集:$S_1,S_2,\ldots,S_n$,那么 C 从每个子集 S_i 中找到最适合加边的代表结点 N_i,然后以某种方式将所有代表结点相连.为了找到最“适合”加边的代表结点,我们考虑如下原则:

(1) 使每个结点的连接度尽量高于某个常数下限(也就是尽可能提高对等网络的最小结点度),从而提高系统容错性;

(2) 使每个结点尽量各尽所能,也就是使负载因子(=结点度/结点能力)尽量趋于一致,从而开发对等网络中普遍存在的结点异构性.

基于上述原则,分点 C 首先找到子集 S_i 中连接度最低的结点,看其结点度是否低于常数下限 Min_Degree,如果是,则将此结点作为 S_i 的代表结点 N_i;否则,计算每个结点的负载因子,将负载因子最小的结点(也就是负担最轻的结点)作为代表结点 N_i.

选择好代表结点后,代表结点之间以何种方式相连?最简单的连接方式是线性链连接:给每个 N_i 加一条到 N_{i+1} 的边(给 N_n 加一条到 N_1 的边),这样,所有的代表结点就形成了一条闭合链,从而合并了互不可达的多个子集,避免了分点的存在.图 7 是线性链连接的例子.

以线性链方式连接代表结点虽然简单,但很脆弱,以后如果有结点失效,则这条链很可能断裂,从而又形成新的分点.因此,我们考虑稍微复杂的连接方式,比如对于代表结点 $N_1,N_2,\ldots,N_n$,让每个 N_i 不仅连接 $N_{(n+1)\bmod n}$,还连接 $N_{(n+2)\bmod n}$,也就是首先连成环,然后在环上给每个代表结点加一条到对面的弦(如图 8 所示).模拟实验部分

比较了带弦环连接和线性链连接的性能,测量结果表明,前者通常要优于后者.

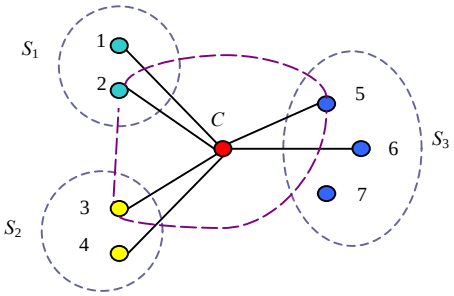


Fig.7 Linear chain connection

图 7 线性链连接各代表结点

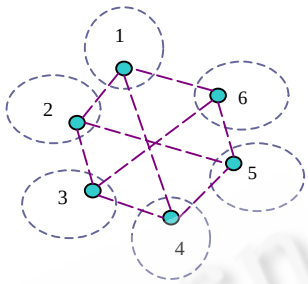


Fig.8 Chordal ring connection

图 8 带弦环连接各代表结点

如果给每个结点再加一些边,比如像 Chord[14]那样,给每个结点 i 加一系列指数距离的边 $i+1,i+2^1,i+2^2,i+2^3,\ldots$,那么连通性的提升自然更多;但是这却带来了更多额外的开销,实现起来也更为复杂.上面所讲的带弦环连接方式,简单实用,也不会带来很多额外开销,我们认为它是介于线性链连接方式和 Chord 所采取的连接方式之间的一个合理折衷.

4.2 减边以限制邻居数

每个结点所可以维护的连接数(结点度)是有限制的.连接度越高,更新路由表的自适应开销就越大,因此,每个结点只能根据自己的能力(带宽、存储容量、计算能力等)维持一定数目的连接.所以,当加边操作导致结点连接度超过其上限时,该结点需要减边以限制邻居数.减边遵循如下原则:

(1) 为避免分点而新加的那些边不能减;

(2) 对于其他边而言,计算每条边所连接的结点的负载因子,将负载因子最大的结点(也就是负担最重的结点)对应的那条边删去.

上述两条原则的理由是明显的,而实现方法也很直观,不再赘述.需要注意的是:原则(1)优先于原则(2),否则可能会产生覆盖网因减边再次分割的问题.

4.3 分点检测和避免的总开销

分点检测的过程中,候选分点给其不同邻居发出的探测消息所到达的范围是不重叠的(只在相交的那一点回发 Msg_Arrival 消息报告邻居间的可达关系),避免了重复的可达性探测.假设无结构对等网络中共有 n 个结点,c 为平均结点度,t 为跳数限制 TTL,那么每个候选分点检测的开销为 $\min(O(c^t),O(nc))$(文献[10]对此作了更细致的分析),所以检测的总开销(即所需消息数)为 $\min(O(nc^t),O(n^2c))$.

如果某个结点确认自己是分点,假设其邻居被分成 k 个子集,那么无论采用线性链连接还是带弦环连接,加边的开销都为 $O(k)$.由于 k 必定小于等于分点的度 d,所以加边开销也可写为 $O(d)$.考虑最坏情况,每个结点都是分点,此时加边的总开销为 $O(nc)$.同理,减边的总开销也为 $O(nc)$.因此,分点避免的总开销为 $O(nc)$,它相比分点检测的总开销 $\min(O(nc^t),O(n^2c))$而言要少得多,可见分点检测和避免算法的开销基本上都在检测这一步.综上所述,分点检测和避免的总开销为 $\min(O(nc^t),O(n^2c))$.特别地,当平均结点度 c 和跳数限制 t 都较小时,总开销即为 $O(n),O(n)$的线性开销一般认为是高效的.

5 模拟实验

5.1 实验方法和参数

实验是基于模拟的,随机产生一张连通的无向图作为覆盖网拓扑的基础,每个结点连接到其他哪些结点是随机的,但平均结点度有所规定(平均结点度=6).模拟的网络结点总数 N=1000,跳数限制 TTL 取 2~5 不等.在第

4.1 节,给结点加边的第 1 条原则,是要使每个结点的连接度尽量高于某个常数下限,实验中,我们设定结点度下限 Min_*Degree*=3(平均结点度的一半).给每个结点随机分配其能力值(1~20 不等),代表它可以负担的连接度.为了模拟高动态性的网络环境,我们让网络中的结点不断失效,而每隔 *T* 个结点失效后作一次分点检测和避免以测量其效果.实验中分别取 *T*=5,10,20,50,*T* 越小,分点检测和避免做得越频繁,通常效果就越好,当然开销也相应会越大.

5.2 实验过程与测量结果

实验 1:分点对覆盖网拓扑的意义.

这个实验的目的在于证实分点在覆盖网中的拓扑关键性.对相同的初始网络,首先让随机结点逐个失效,记录多少个结点失效后覆盖网被分割;然后让分点逐个失效,记录多少个分点失效后覆盖网被分割.*TTL* 分别取 3 和 4,多次实验取平均值.

由图 9 可以观察到:对同样的覆盖网,如果要让它分割,失效的分点数要远少于失效的随机结点数.覆盖网被分割时失效的随机结点数在 1~150 之间,而失效的分点数,*TTL*=3 时在 1~35 之间,*TTL*=4 时在 1~2 之间(这说明,*TTL*=4 时检测到的大多数分点也是割点).由此可以看出,分点在覆盖网中的拓扑关键性:只要让极少数的分点失效,覆盖网就会被分割.

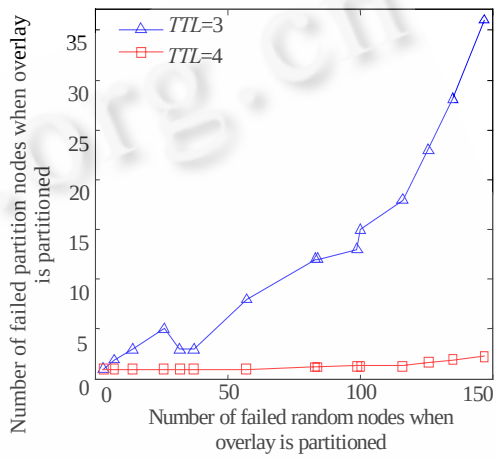


Fig.9 The importance of partition nodes

图 9 分点对覆盖网拓扑的意义

实验 2:分点避免算法的有效性.

为了测量分点检测和避免算法的有效性,我们首先检测网络初始的分点数,然后作一次分点避免并检测避免后的分点数,最后再作一次分点避免并检测第 2 次避免后的分点数,以观察分点数的变化趋势.测量结果都是多次实验的平均值,*TTL* 分别取 2,3,4,5.

表 1 的测量结果和图 10 说明,每一次分点避免后,网络分点数均有明显下降,尤其是当 *TTL*=2 时最为突出:初始时 1 000 个结点中几乎所有结点都是分点,第 1 次避免后减少到 50 附近,第 2 次避免后减少到 20 附近.初始时几乎所有结点都是分点,原因在于 *TTL*=2 很小,只探测到极其有限的可达性关系,所以结点的邻居集基本上被划分到多个子集中,从而几乎每个结点都认为自己是分点.*TTL*=3 时,初始分点数在 160 左右,第 1 次避免后降到 55 左右,是初始的约 1/3,第 2 次避免后降到 20 左右,又是第 1 次避免后的约 1/3.*TTL*=4 时,初始分点数稳定在 15 左右,第 1 次避免后降到 1,第 2 次避免后接近 0.由于 *TTL*=2 时初始分点数太多,没有将它画在图 10 中,*TTL*=5 时的实验结果与 *TTL*=4 时非常相似,所以也没有画出.总体上讲,*TTL* 越大,探测到的可达性关系就越充分,分点数相应越少.但无论 *TTL* 取多少,算法本身的有效性是非常明显的.

分析第 1 次分点避免后为什么还存在分点,我们认为原因在于分点避免时所做的加边、减边操作改变了覆盖网拓扑结构,从而带来了新的分点.不过,新的分点数通常都是很少的,并会慢慢趋于 0,网络组织结构也相应趋于稳定.

Fig.1 Number of partition nodes in different states

表 1 不同状态下的分点数目

Number of partition nodes	Initial state	After the 1st round operations	After the 2nd round operations
TTL=2	981	55.6	21.6
TTL=3	165.8	55.4	20.8
TTL=4	15.4	1	0.4
TTL=5	10.6	0.8	0.2

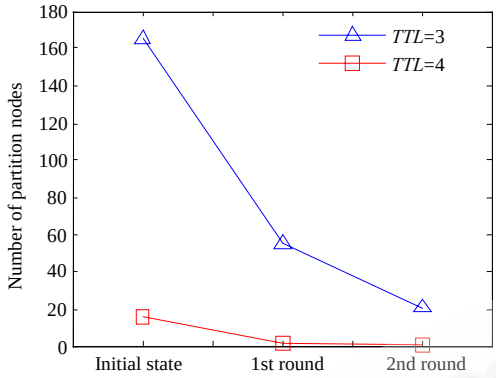


Fig.10 Effectiveness of the partition node avoidance

图 10 分点避免算法的有效性

实验 3:系统对覆盖网分割的抵抗力增强.

为了观察分点避免后系统对覆盖网分割的抵抗力增强,我们首先让网络结点逐个失效并且不采取任何补救措施,测得 58 个结点失效后覆盖网被分割.同样的网络,让同样的结点逐个失效,但每 T 个结点失效后作一次分点避免,记录多少个结点失效后覆盖网被分割.实验中,T=5,10,20,50 表示分点避免算法执行的频率高低.

表 2 的测量结果和图 11 表明:在 TTL=3 的情况下,当 T=5 时,超过 900 个结点失效,覆盖网才被分割,这说明分点避免做得及时、有效,使覆盖网一直保持很好的连通性;当 T=10,20,50 时,分别有约 1/3,1/4,1/10 的结点失效,覆盖网被分割.在 TTL=4 的情况下,当 T=5,10,20,50 时,分别有约 1/3,1/5,1/6,1/10 的结点失效,覆盖网被分割.

Fig.2 Number of failed nodes with different failure periods

表 2 不同失效周期下的失效结点数目

Number of failed nodes	T=5	T=10	T=20	T=50
TTL=3	58→926	58→341.6	58→237.4	58→117
TTL=4	58→357	58→218.2	58→170	58→110

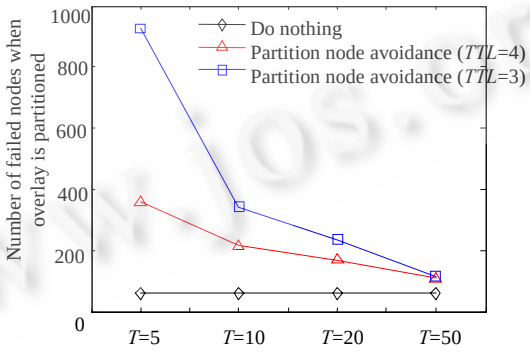


Fig.11 Resistance improvement to overlay partition

图 11 系统对覆盖网分割的抵抗力增强

随着 T 的变大,分点避免执行的频率变低,网络拓扑结构得不到及时、有效的修复,覆盖网分割时的失效结点数也随之减少,说明系统对覆盖网分割的抵抗力变弱了.而当 TTL 从 3 变大到 4 后,由于每次检测到的分点数变少(从实验 2 可以看出),对网络结构的调整也相应变少,所以对覆盖网分割的抵抗力也变弱了.总体上讲,T 越小,TTL 越小,系统对覆盖网分割的抵抗力就越强,当然,通信开销也相应增大.

实验 4:系统查询成功率提升.

我们首先测量初始网络的查询成功率(查询对象随机指定),然后进行一次分点避免,并且测量查询成功率(同一结点每次都做同样的查询),最后进行第 2 次分点避免,再次测量查询成功率.注意,网络环境是静态的,不存在结点失效的情况,查询的跳数限制 *TTL* 与分点检测的 *TTL* 是一样的.

从表 3 的测量结果和图 12 可以看到:无论 *TTL* 取 3 还是 4,第 1 次分点避免后的查询成功率相对初始网络都有所提高,*TTL* 越小,提高越多.联系实验 2 来分析其原因,*TTL* 越小,检测到的分点就越多,对网络所作的调整也越多.这同时也表明:分点避免对网络结构的调整是有利于系统查询成功率的,是对覆盖网拓扑的一种优化.第 2 次分点避免对查询成功率的提升很小,甚至没有,因为第 1 次分点避免带来的新分点数非常少.

Fig.3 Query success rate in different states

表 3 不同状态下的查询成功率

Query success rate	Initial state	After the 1st round operations	After the 2nd round operations
TTL=3	0.225	0.267	0.274
TTL=4	0.684	0.706	0.706

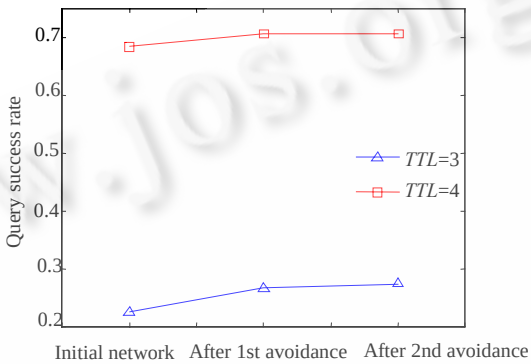


Fig.12 Increment of query success rate

图 12 系统查询成功率的提升

实验 5:系统容错性提升.

我们将系统容错性理解为结点不断失效的高动态性环境下系统的查询成功率变化趋势,希望采取了周期性的分点避免后,查询成功率总体上能够比较稳定.首先让网络结点逐个失效但不采取任何补救措施,每 *T* 个结点失效后测量 1 次查询成功率(同一结点每次都做同样的查询).然后,对同样的网络,让同样的结点失效,但每 *T* 个结点失效后做 1 次分点避免,同时测量系统查询成功率.

取 *TTL*=3,*TTL*=4,*T*=10,*TTL*=50,因此就有 2×2=4 种组合,需要作 4 次实验.查询的跳数限制 *TTL* 与分点检测的 *TTL* 是一样的.图 13 显示了 *TTL*=3 时的实验结果:随着结点不断失效,作分点避免相对于不采取任何补救措施而言,查询成功率要更高、更稳定,并且不会出现急剧的下降.*TTL* 越大,*T* 越大,容错性的提升就越不明显,前者是因为 *TTL* 越大,检测到的分点越少,所以对覆盖网的修复也越少;后者是因为 *T* 越大,对覆盖网的修复越不及时,所以容错性相应降低了.

实验 6:线性链连接和带弦环连接的各方面性能比较.

分点避免的重要步骤是给互不可达的各个子集的代表结点加边将它们连接,但是连接方式哪一个好,哪一个坏是一个问题.实验中我们在其他条件(*TTL*、*T*、初始的网络拓扑结构、失效结点)相同的情况下,分别采用线性链连接和带弦环连接两种不同的连接方式,比较它们的多方面性能:(1) 分点避免算法的有效性(如图 14(a)所示);(2) 系统对覆盖网分割的抵抗力增强(如图 14(b)所示);(3) 系统容错性提升(如图 14(c)所示).

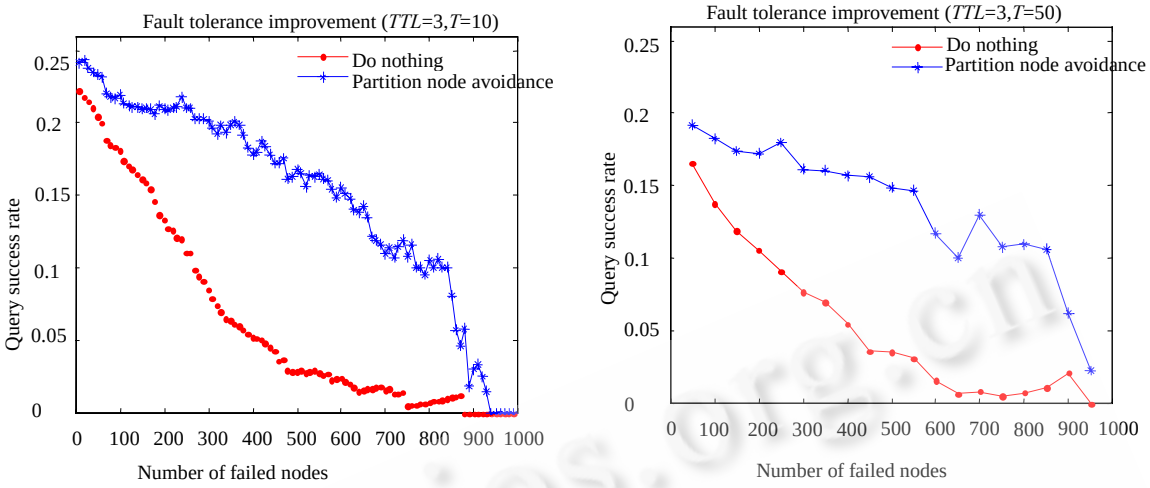
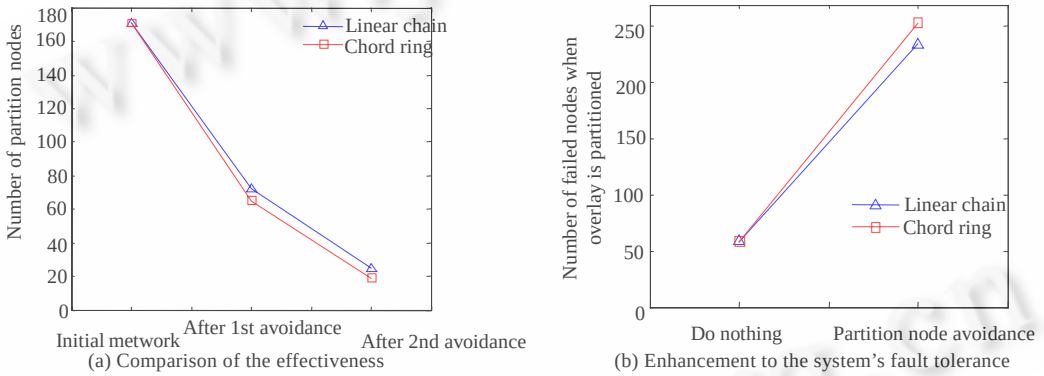


Fig.13 Improvement of system fault tolerance ($TTL=3$)

图 13 系统容错性的提升($TTL=3$)

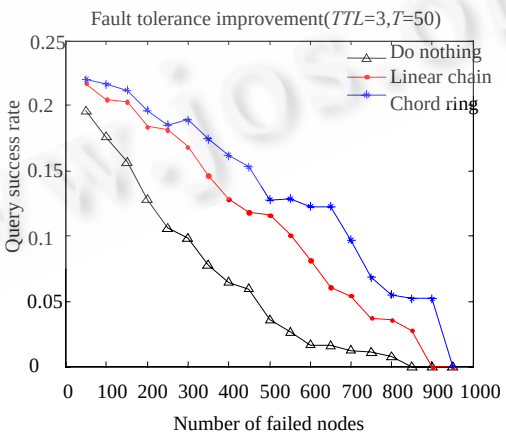


(a) Comparison of the effectiveness

(a) 算法本身的有效性比较

(b) Enhancement to the system's fault tolerance

(b) 系统对覆盖网分割抵抗力的增强



(c) Comparison of the fault tolerance

(c) 线性链连接和带弦环连接的容错性比较

Fig.14 Performance comparisons of linear chain connection and chordal ring connection

图 14 线性链连接和带弦环连接的各方面性能比较

从表 4、表 5 的测量结果和图 14 可以看出:不论就哪个方面的性能而言,带弦环连接通常都比线性链连接有所提升.原因是很直观的:带弦环连接能够将同样的代表结点连接得更好、连通性更高,而线性链连接形成的单环相对脆弱,很可能因为以后的结点失效而分裂.

Table 4 The effectiveness of our partition node avoidance algorithm (*TTL*=3)

表 4 分点避免算法的有效性(取 *TTL*=3)

TTL=3	Initial state	After the 1st round operations	After the 2nd round operations
Linear chain connection	171	72	25
Chordal ring connection	171	65	19

Table 5 The enhancement of system resilience to overlay partition, *TTL*=3, *T*=50

表 5 系统对覆盖网分割抵抗力的增强.取 *TTL*=3,*T*=50

TTL=3,*T*=50	Number of failed nodes
Linear chain connection	58 → 235
Chordal ring connection	58 → 254

实验 7:系统异构性开发.

我们为避免分点而给结点加边,当结点连接度达到上限时给结点减边,都是以异构性的开发作为主要的指导原则,从而提高系统的负载均衡,让每个成员尽量各尽所能.为了更具体地量化系统异构性的开发,我们测量一个实验网络从开始到稳定期的结点负载因子分布,包括:最小负载因子、最大负载因子、平均负载因子、负载因子方差.取 *TTL*=4,加边采用"带弦环连接",结点能力值取 1~20 之间的随机数,平均结点度为 6,最小结点度为 3.

从表 6 的实验结果我们可以看出,处于稳定期的网络与初始网络具有相同的最小、最大负载因子,平均负载因子略大,而负载因子方差则有明显减少.这与我们预期的结果基本相符.由于实验规模为 1 000 个结点,出现能力值为 20 而连接度仅为 1 的结点(即最小负载因子的结点)的概率是较高的,即使在稳定期仍然如此.最大负载因子 1.0 的出现几乎是必然的,因为很多结点的连接度可以达到其能力上限.由于避免分点时加边情形的发生概率要高于减边,所以不难理解稳定期的平均负载因子要略大于初始网络.最后,由于无论加边、减边时我们都考虑到了负载均衡,所以稳定期的负载因子方差要明显小于初始网络.

Table 6 Different load factors in different states

表 6 不同状态下的各负载因子数值

	Min load factor	Max load factor	Average load factor	Load factor variance
Initial state	0.05	1.0	0.619	0.341
Steady state	0.05	1.0	0.635	0.227

6 结论与进一步的工作

本文首先指出,对松散化组织、局部性路由的无结构对等网络而言,静态的、全局意义的割点既难以准确检测,也远非系统拓扑关键点的全部,所以我们提出了分点这一概念来描述无结构对等网络的拓扑关键点.然后从分点的定义出发,设计了一套简单、有效、分布式的分点检测和避免方法.模拟实验的结果表明:我们的方法能够有效地检测并避免分点,使系统对覆盖网分割的抵抗力得到本质的增强;分点避免后,系统查询成功率有所上升,在高动态性网络环境下系统的容错性得到明显提高.在作分点避免时,我们对代表结点的选取做了仔细的考虑,兼顾了容错性和异构性;对代表结点的连接,则采取了线性链连接和带弦环连接两种方式,实验表明,后者在性能上通常要更优.

我们下一步的工作将把分点的概念、检测、避免应用到结构化对等网络(如 Chord、Pastry 等)中,以增强结构化对等网络对覆盖网分割的抵抗力并提高其容错性.鉴于结构化对等网络和无结构对等网络在组织结构、路由方式等多方面的不同,本文的许多地方需要作相应的修改,尤其是结构化对等网络的覆盖网不能看作无向图.虽然如此,分点作为对等网络的拓扑关键点,对优化其覆盖网拓扑结构有着重要意义.

References:

[1] Gnutella protocol specification. 2007. http://rfc-gnutella.sourceforge.net

[2] KaZaA website. 2007. http://www.kazaa.com

[3] eDonkey website. 2007. http://www.edonkey.com

[4] BitTorrent website. 2007. http://www.bittorrent.com

[5] Skype website. 2007 http://www.skype.com

[6] Groove website. 2007. http://www.groove.net

[7] GPU project website. 2007. http://gpu.sourceforge.net

[8] Saroiu S, Gummadi P, Gribble S. Measuring and analyzing the characteristics of napster and gnutella hosts. Multimedia Systems 9, Berlin: Springer-Verlag, 2003. 170−184.

[9] Ripeanu M. Peer-to-Peer architecture case study: Gnutella network. In: Proc. of the 1st IEEE Int'l Conf. on Peer-to-Peer Computing (IEEE P2P). Linkoping: IEEE Computer Society, 2001. 99−100.

[10] Liu X, Xiao L, Kreling A, Liu Y. Optimizing overlay topology by reducing cut vertices. In: Proc. of the ACM Int'l Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV). Newport: ACM Special Interest Group on Multimedia. 2006. http://portal.acm.org/citation.cfm?id=1378213&jmp=references&coll=ACM&dl=ACM

[11] Chawathe Y, Ratnasamy S, Breslau L, Lanham N, Shenker S. Making gnutella-like P2P systems scalable. In: Proc. of the Annual Conf. of the Special Interest Group on Data Communication (SIGCOMM). Karlsruhe: ACM Special Interest Group on Data Communication, 2003. 407−418.

[12] Liben-Nowell D, Balakrishnan H, Karger D. Analysis of the evolution of peer-to-peer systems. In: Proc. of the 21st Annual Symp. on Principles of Distributed Computing (PODC). Monterey: ACM SIGOPS and ACM SIGACT, 2002. 233−242.

[13] Saia J, Fiat A, Gribble S, Karlin A, Saroiu S. Dynamically fault-tolerant content addressable networks. In: Proc. of the 1st Int'l Workshop on Peer-to-Peer Systems (IPTPS). Cambridge: Springer-Verlag, 2002.

[14] Stoica I, Morris R, Karger D, Kaashoek MF, Balakrishnan H. Chord: A scalable peer-to-peer lookup service for Internet applications. In: Proc. of the Annual Conf. of the Special Interest Group on Data Communication (SIGCOMM). San Diego: ACM Special Interest Group on Data Communication, 2001. 149−160.

[15] Loguinov D, Casas J, Wang X. Graph-Theoretic analysis of structured peer-to-peer systems: Routing distances and fault resilience. IEEE/ACM Trans. on Networking, 2005,13(5):395−406.

[16] Ratnasamy S, Francis P, Handley M, Karp R, Shenker S. A scalable content addressable network. In: Proc. of the Annual Conf. of the Special Interest Group on Data Communication (SIGCOMM). San Diego: ACM Special Interest Group on Data Communication, 2001. 161−172.

[17] Pandurangan G, Raghavan P, Upfal E. Building low-diameter P2P networks. In: Proc. of the 42nd IEEE Symp. on Foundations of Computer Science (FOCS). Las Vegas: IEEE Computer Society 2001. 492−499.

[18] Rowstron A, Druschel P. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In: Proc. of the 18th IFIP/ACM Int'l Conf. on Distributed Systems Platforms (Middleware). Heidelberg: Springer-Verlag 2001. 329−350.

[19] Mahajan R, Castro M, Rowstron A. Controlling the cost of reliability in peer-to-peer overlays. In: Proc. of the 2nd Int'l Workshop on Peer-to-Peer Systems (IPTPS). Berkeley: Springer Verlag, 2003.

[20] Harvey N, Jones MB, Theimer M, Wolman A. Efficient recovery from organizational disconnects in SkipNet. In: Proc. of the 2nd Int'l Workshop on Peer-to-Peer Systems (IPTPS). Berkeley: Springer-Verlag, 2003.

[21] Harvey N, Dunagan J, Jones MB, Saroiu S, Theimer M, Wolman A. SkipNet: A scalable overlay network with practical locality properties. In: Proc. of the USENIX Symp. on Internet Technologies and Systems (USITS). Seattle: USENIX Association, 2003.

[22] Sit E, Morris R. Security considerations for peer-to-peer distributed hash tables. In: Proc. of the 1st Int'l Workshop on Peer-to-Peer Systems (IPTPS). Cambridge: Springer-Verlag, 2002.

李振华(1983－),男,江苏盐城人,硕士生,主要研究领域为对等计算.

邱彤庆(1981－),男,硕士生,主要研究领域为对等计算.

陈贵海(1963－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为对等计算,网络系统,传感器网络,并行计算.