软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2011,22(9):2121−2136 [doi: 10.3724/SP.J.1001.2011.03886]

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

eMule 网络最佳节点标识*

刘祥涛[1,2+], 程学旗[1], 李 洋[3], 陈小军[1], 白 硕[1], 刘 悦[1]

[1](中国科学院 计算技术研究所,北京 100190)

[2](中国科学院 研究生院,北京 100049)

[3](中国移动通信研究院,北京 100053)

Optimal Peer Identifier in eMule Network

LIU Xiang-Tao[1,2+], CHENG Xue-Qi[1], LI Yang[3], CHEN Xiao-Jun[1], BAI Shuo[1], LIU Yue[1]

[1](Institute of Computing Technology, The Chinese Academy of Sciences, Beijing 100190, China)

[2](Graduate University, The Chinese Academy of Sciences, Beijing 100049, China)

[3](China Mobile Research Institute, Beijing 100053, China)

+ Corresponding author: E-mail: liuxiangtao@sotware.ict.ac.cn

Liu XT, Cheng XQ, Li Y, Chen XJ, Bai S, Liu Y. Optimal peer identifier in eMule network. *Journal of Software*, 2011,22(9):2121−2136. http://www.jos.org.cn/1000-9825/3886.htm

Abstract: In recent years, eMule network, a kind of peer-to-peer (P2P) file-sharing network has become more and more popular. Along with its popularity, the demand to accurately determine the peer in eMule has also increased for two reasons: it is a critical step to accurately locate sources of files in P2P file-sharing networks, and the wanton spread of vulgar content makes it necessary to censor eMule. This demand allows everyone to put forward the problem of optimal peer identifier in eMule network. However, since Kad ID (the widely-used identifier in eMule network) can be freely changed by users of eMule, there exists Kad ID aliasing, a single peer may correspond to multiple Kad IDs; reversely, There also exists Kad ID repetition, which are multiple peers corresponding with a single Kad ID. Therefore, it is difficult to accurately determine the peer by using Kad ID. This paper attempts to solve this problem. First, the stability factor (SF) of peer identifier is defined to evaluate candidate identifiers. Then, a crawler named Rainbow is designed and implemented to collect peer information from multiple candidate identifiers' relationship in real eMule network. Note that Rainbow has been proved to be convergent and has low time and space complexity. Experimental results show that {userID} is the optimal peer identifier in peer identifier set $2^{\{\text{Kad ID,userID,IP}\}}-\{\Phi\}$ as {userID} has the largest SF value. Later on, in order to quantify the extent of Kad ID aliasing, the relationship between {userID} and {Kad ID} is discussed. Lastly, the effectiveness of the application of the optimal peer identifier is analyzed. Results show that peers are more accurately determined when using {userID} as the identifier of peers. All in all, the identification of optimal peer identifier provides a basis for future research of eMule network, and Rainbow serves as a useful tool for measuring real eMule network.

Key words: P2P network; eMule network; crawler; peer identifier; stability factor

* 基金项目: 国家自然科学基金(60803085, 60873245)
收稿时间: 2009-06-09; 修改时间: 2009-12-25; 定稿时间: 2010-05-18

摘 要: eMule 网络是近年来越来越流行的文件共享对等网络.一直以来,文件源的准确定位是文件共享对等网络的一个关键步骤;此外,不健康内容的肆意传播也使网络监管成为必需.这些都导致准确确定 eMule 网络中节点的需求,同时促使 eMule 网络最佳节点标识问题的提出.然而,eMule 网络中广泛使用的节点标识 Kad ID 因可被 eMule 用户任意更改,存在 Kad ID 别名,即单个节点对应多个 Kad ID 的情况,以及 Kad ID 重复,即多个节点对应同一个 Kad ID 的情况,从而使用传统 Kad ID 很难准确确定节点.为解决这一问题,首先定义候选节点标识的稳定因子用以评价候选标识;然后设计并实现一个证明可收敛且时空复杂度不高的 eMule 网络节点信息采集器——Rainbow,以获得实际 eMule 网络中节点的多个候选标识之间的对应关系信息.实验结果表明,{userID}的稳定因子最大,是节点标识集合 $2^{\{Kad\ ID,userID,IP\}}-\{\Phi\}$中的最佳节点标识;之后,为了量化 Kad ID 别名问题的程度,对{userID}与{Kad ID}的关系进行探讨;最后对最佳节点标识的应用有效性进行分析,说明采用{userID}作为节点标识能够更准确地确定节点.总之,所确定的最佳节点标识为 eMule 网络的研究奠定了基础,Rainbow 也为真实 eMule 网络测量提供了良好的工具.

关键词: 对等网络;eMule 网络;采集器;节点标识;稳定因子

中图法分类号: TP393 **文献标识码**: A

eMule 网络[1]是一种混合类型的文件共享对等网络,其网络由集中式网络和纯分布式网络组成两部分,纯分布式网络采用了一种结构化对等网络协议——Kademlia 协议[2],该网络主要用于 eMule 的文件和文件关键字的发布与查找并被一些研究者[3,4]称为 Kad 网络.Ipoque 2008~2009 年度的因特网流量报告[5]指出:依地域不同,eMule 流量占 P2P 总流量的 2%~47%,占总因特网流量的 1%~26%.而且相关研究[6,7]表明,P2P 流量在过去几年一直呈上涨趋势.

当在 eMule 网络中下载文件时,首先要找到拥有某文件的若干节点(eMule 网络中,节点是被指定了某个标识并参与到 eMule 网络的物理机器),然后进行文件传输.其中涉及到影响文件共享性能的一个关键步骤:资源的准确定位,即准确地找到拥有某文件的源节点.此外,被广泛应用的 eMule 网络为版权侵犯和不健康内容传播提供了温床,这促使了对其进行监管的需求.资源的准确定位和网络监管都必须准确地确定节点.然而在 eMule 网络中,Kad ID(由 eMule 用户在本地随机生成的一个长为 128 位的节点标识)作为节点的一种常用标识,可以被 eMule 用户任意更改.使得一个节点可能对应多个 Kad ID,导致 Kad ID 别名(Kad ID aliasing[4]);此外,多个节点也可能对应同一个 Kad ID,导致 Kad ID 重复(Kad ID repetition[3]).可以直观认为,Kad ID 并不是 eMule 网络的最佳节点标识.

本文对 eMule 网络最佳节点标识问题进行分析和探讨.首先提出标识合适性度量:稳定因子并对最佳节点标识进行形式化定义;然后设计一个 eMule 网络节点信息采集器:Rainbow 用于收集候选节点标识的对应信息;最后通过实验分析获得 eMule 网络的最佳节点标识,从而为资源定位和网络监管奠定基础.本文第 1 节介绍相关工作.第 2 节提出 eMule 网络节点标识相关度量和最佳节点标识定义,并对 eMule 中常见的两个标识{Kad ID}和{userID}的关系进行探讨.第 3 节介绍我们的实验工具:eMule 网络节点信息采集器——Rainbow 的设计思想和实现原理,并对其收敛性、时间复杂度和空间复杂度进行分析.第 4 节对实验结果进行分析.最后在第 5 节对本文主要工作进行总结.

1 相关工作

在对真实 P2P 网络进行测量分析时,一些研究[3,4,8−11]发现了某些节点标识(比如 IP,Kad ID)的不稳定现象,但并没有深入研究“最佳节点标识”问题.2003 年,Bhagwan 等人[8]在测量 Overnet 的节点可用性(peer availability,即节点在线时长与测量时长的比率)时,发现一个节点可能对应多个 IP 地址,即 IP 地址别名.他们指出,这将导致大大低估节点可用性.2005 年,Kutzner 等人[9]在测量 Overnet 的一些性能参数(比如网络大小、节点可用性)时,也发现 IP 地址别名.文献[4,10,11]发现了 Kad ID 别名,但没有深入分析其具体原因.2009 年,Yu 等人[3]测量了 Kad 网络中 Kad ID 重复程度,据他们的数据显示,19.5%的节点(包括在线节点和离线节点)和 4.5%的在线节点具有 Kad ID 重复,同时也指出,Kad ID 重复可能会导致对 Kad 网络测量的错误结果,其实验结果表明,Kad ID 重复会

导致 Kad 网络的发布和查找性能降级.据我们所知,本文首次提出 eMule 网络最佳节点标识问题,目的是为了找到 eMule 网络中的最佳节点标识,以设法减少测量错误等负面影响和精确确定节点.

节点标识别名和重复问题的测量需要采集器来获得数据源.2002 年,Saroiu 等人[12]率先使用主动测量方法对当时最为流行的 Gnutella 和 Napster 进行了拓扑测量.2005 年,Stutzbach 等人[13]在前人工作的基础上改进了主动测量方法,并开发出了快速分布式 Gnutella 拓扑采集器:Cruiser.实验表明,如果采集器采集速度太慢,节点震荡(peer churn,新节点加入和老节点离开网络的现象)可能会导致错误,从而使得提高 Crawler 的采集速度成为提高测量准确性的关键问题.2008 年,王勇等人[14]针对 Gnutella 网络设计了基于正反馈的分布式 Gnutella 拓扑采集器:D-Crawler,提出了度量采集器准确性和完整性的衡量指标,分析了 Gnutella 网络拓扑图的度等级分布特征、度频率分布特征以及小世界特性.

Kademlia 协议的实现有 Overnet、eMule 网络和 Bittorrent[15]的 DHT 网络.文献[8,9,16]对 Overnet 进行了测量.2006 年,Stutzbach 等人[17]针对 eMule 网络提出了计算查询性能的分析框架,并开发出了两个软件:kFetch 和 kLookup 用于采集和计算 eMule 网络的查询性能.2006 年,Stutzbach 等人[18]对 3 个 P2P 网络:Gnutella、eMule 网络和 Bittorrent 进行了测量和相应的节点震荡分析.2007 年,Steiner 等人[4,10,11]设计了 eMule 网络采集器:Blizzard 并进行了为期 179 天的 eMule 网络采集,获得了节点的地理分布、会话时间、节点可用性和生命周期等测度的测量结果.2007 年,Falkner 等人[19]在 PlanetLab 实验条件下,对 Bittorrent 的一个客户端 Azureus 的 DHT 网络进行了测量.2006 年,刘琼等人[20]对当时的 P2P 测量研究现状进行了详细的综述,将 P2P 测量研究分为 3 类:P2P 拓扑特征的测量、P2P 流量特征的测量和 P2P 可用性(分为主机可用性和内容可用性)的测量.

P2P 网络采集目前有 3 种方式:完全采集(full crawl)、区域采集(zone crawl)以及本文采用的独特随机采集(random crawl).完全采集即快速地采集整个 P2P 网络的节点信息.文献[4]对 eMule 网络进行了多次完全采集.其优点是:获得信息完整;其缺点为:eMule 网络的用户规模已达百万级,完全采集耗费时间过长,节点震荡带来的影响加剧,使采集的节点信息有多个瞬态快照的堆叠现象.区域采集只采集 Kad ID 前 k 位相同的节点空间的节点信息.文献[18]进行了前 10/12 位相同的区域采集,文献[4]进行了前 8 位相同的区域采集.其优点是:采集时间远远小于一次完全采集的耗时,且因为 Kad ID 的随机生成性,使得一个区域的节点分布接近均匀分布[4],使区域采集具有一定的代表性.其缺点为:不能获得 eMule 网络节点的完整信息,例如前 8 位相同的区域采集只能产生完全采集的 1/256 的节点信息.随机采集不固定 k 位前缀,从节点总体中随机抽取一些节点来获得他们的信息.随机采集的优缺点与区域采集类似.此外,它还具有配置灵活的优点,即采集的节点数量是一个可以灵活配置的参数.

虽然已有的 P2P 测量工作取得了一定进展,但已有工作主要关注拓扑或节点特征(比如节点可用性)测量,而并没有针对最佳节点标识问题设计可用的采集器.本文设计实现了 eMule 网络节点信息采集器——Rainbow,以获得 eMule 网络中节点候选标识集合 S={Kad ID,userID,IP}的对应关系信息,并以此为数据集来试图解决 eMule 网络的最佳节点标识问题.

2 eMule 网络最佳节点标识

2.1 度量定义

Kad ID 别名问题的存在导致有寻找更好 eMule 网络标识的需求,本节提出了度量标识合适性的稳定因子等概念.

定义 1(稳定因子(stability factor,简称 SF). 给定候选标识 $\{x_i\}\in 2^{\{x_1,x_2,\ldots,x_n\}}-\{\Phi\}$,存在 m 个不同的取值$\{x_{i1}, x_{i2},\ldots,x_{im}\}$,且对 x_i 的任意一个取值 x_{ij},对应有 p_j 个元组(这些元组为以$\{x_1,x_2,\ldots,x_n\}$为标识的包含 x_{ij} 的取值组合).则$\{x_i\}$的稳定因子定义为

$$SF=\frac{\sum_{j=1}^{m}p_j}{m} \tag{1}$$

公式(1)中,$SF\in[1,+\infty)$,在 m 不变(或 $\sum_{j=1}^{m}p_j$ 不变)的情况下,$\sum_{j=1}^{m}p_j$ 越大(或 m 越小)表示有更多的元组与$\{x_i\}$对应,即以$\{x_i\}$作为节点标识越稳定越好.

例 1:假设人的候选标识集合为 $2^{\{姓名,学号\}}-\{\Phi\}$,现在有 3 个人:甲、乙、丙.其中:甲有 1 个姓名:李木,1 个学号:1423;乙有 1 个姓名:王小虎,2 个学号:2325,2985(不同学校具有不同的学号);丙有 2 个姓名:张意(曾用名)、张大全,3 个学号:3456,3213,3829.则姓名有 4 个不同的取值{李木,王小虎,张意,张大全},对李木这个取值,对应有 1 个元组:{李木,1423},即有 p_1=1;对王小虎这个取值,对应有 2 个元组:{王小虎,2325}和{王小虎,2985},故有 p_2=2.对张意,对应有 3 个元组:{张意,3456}、{张意,3213}、{张意,3829},即有 p_3=3;同理,对应张大全的 p_4=3.可求得{姓名}的稳定因子 $SF_{\{姓名\}}=(p_1+p_2+p_3+p_4)/4=(1+2+3+3)/4=9/4$.同理,{学号}的稳定因子 $SF_{\{学号\}}=(1+1+1+2+2+2)/6=3/2$,{姓名,学号}的稳定因子 $SF_{\{姓名,学号\}}=(1+1+1+1+1+1+1+1+1)/9=1$.此例中,{姓名}的稳定因子最大.

定义 2(最佳节点标识). 假设对节点的候选标识集合 $2^S-\{\Phi\}$,$S=\{x_1,x_2,\ldots,x_n\}$,$\exists a\in(2^S-\{\Phi\})$,使得 $\forall b\in(2^S-\{\Phi\})\rightarrow SF_a=\max(SF_b)$,则 a 是集合 S 中的最佳节点标识.

例 2:同例 1 的情景,S={姓名,学号},$2^S-\{\Phi\}$={{姓名},{学号},{姓名,学号}},$SF_{\{姓名\}}=9/4$,$SF_{\{学号\}}=3/2$,$SF_{\{姓名,学号\}}=1$,从而 $SF_{\{姓名\}}=\max(SF_{\{姓名\}},SF_{\{学号\}},SF_{\{姓名,学号\}})$,故{姓名}是 S 中的最佳节点标识.

选出集合 S 的最佳节点标识后,为对最佳节点标识与候选标识的关系正常程度进行归一化度量,提出最佳节点标识——候选标识 P 关系度量:候选标识正常度(candidate identifier normality,简称 *CIN*)的定义.

定义 3(候选标识正常度).

$$CIN=\frac{y_1}{\sum_{i=1}^{n}y_i\times i+\sum_{j=2}^{m}z_{1/j}\times j} \tag{2}$$

其中,y_i 为最佳节点标识与候选标识 P 的 1 对 i 关系的数量(例如,i=1 表示最佳节点标识与 P 的 1 对 1 关系,i=4 表示最佳节点标识与 P 的 1 对 4 关系),$z_{1/j}$ 为最佳节点标识与 P 的 j 对 1 关系的数量(例如,j=3 表示最佳节点标识与 P 的 3 对 1 关系).

$CIN\in[0,1]$反映了 eMule 网络中最佳节点标识——候选标识 P 关系的正常度.*CIN* 越大,说明最佳节点标识与候选标识的一一对应关系比率越高,正常度越高;反之,*CIN* 越小,则说明最佳节点标识与候选标识的一对多关系或者多对一关系比率越高,正常度越低.

例 3:同例 1 的情景,在选定{姓名}作为集合{姓名,学号}的最佳节点标识后,现对{学号}的 *CIN* 进行计算.{姓名}和{学号}的关系={(李木,1423),(王小虎,2325),(王小虎,2985),(张意,3456),(张意,3213),(张意,3829),(张大全,3456),(张大全,3213),(张大全,3829)},可得其中{姓名}与{学号}的 1 个 1 对 1 关系:(李木,1423),1 个 1 对 2 关系:(王小虎,2325),(王小虎,2985),2 个 1 对 3 关系:(张意,3456),(张意,3213),(张意,3829);(张大全,3456),(张大全,3213),(张大全,3829),3 个 2 对 1 关系(张意,3456),(张大全,3456);(张意,3213),(张大全,3213);(张意,3829),(张大全,3829).所以,$CIN_{\{学号\}}=\frac{1}{(1\times1+1\times2+2\times3)+3\times2}=1/15$.同理,容易算出 $CIN_{\{姓名\}}=1$,$CIN_{\{姓名,学号\}}=1$.由此例可见,因{姓名}和{学号}的对应关系中包含大量的一对多和多对一关系,导致 $CIN_{\{学号\}}$远低于 $CIN_{\{姓名\}}$和 $CIN_{\{姓名,学号\}}$.

2.2 {userID}-{Kad ID}关系

eMule 网络中,userID[21]是通过随机数生成算法产生用于 eMule 网络的信誉系统(credit system)的 128 位标识符.本节主要探讨{userID}与{Kad ID}之间可能的复杂关系.

(1) 一对一关系:{userID}与{Kad ID}一一对应.这是{userID}与{Kad ID}之间的正常关系;

(2) 一对多关系:一个{userID}对应多个不同的{Kad ID}.

产生原因分析:a) NAT(network address translation,网络地址转换)后,即局域网内多个节点各自有不同的{Kad ID},通过共同的公网出口共享同一个 IP 地址和{userID},从而产生{userID}与{Kad ID}的一对多关系;b) eMule 网络中可能存在 Sybil 攻击(即一个节点伪造多个身份进行网络攻击)[22],该攻击行为在模拟多个客户

端时,可能会使用同一个 userID 而给每个模拟的客户端分派不同的 Kad ID,从而产生{userID}与{Kad ID}的一对多关系.

(3) 多对一关系:多个{userID}对应同一个{Kad ID}.

产生原因分析:a) 用户利用{Kad ID}可以任意指定的漏洞,为多个节点指定同一个{Kad ID}而让每个节点使用不同的 userID,用于 eMule 网络探测或攻击;b) 哈希冲突.在 2^{128} 空间产生相同哈希的可能性多大呢?即使假设 eMule 网络中有 10 亿个节点,每个节点随机生成的{Kad ID}产生冲突的可能性:

$$p=1-\frac{2^{128}\times(2^{128}-1)\times...\times(2^{128}-10^9+1)}{(2^{128})^{10^9}}\approx 0.$$

可见,哈希冲突的几率极低.

3 eMule 网络节点信息采集器

虽然到目前为止,对 P2P 网络采集器的研究已经很深入.但是如下两个原因促使本文进行 eMule 网络采集器的设计与实现:1) 现有的 eMule 网络采集器只能与节点进行 UDP 通信,只能获取有限的节点信息.而本文需要获得更丰富的节点信息(比如节点的 userID),必须与节点进行 TCP 通信;2) 没有公开可用的 eMule 网络采集器.基于以上原因,促使本文设计实现了自己的 eMule 网络节点信息采集器——Rainbow.

3.1 eMule网络节点信息采集器设计

我们提出如下问题:eMule 网络是否是可被采集的网络?设计 Rainbow 是否可行?首先,eMule 网络肯定是可被采集的,因为已经存在一些 eMule 采集器的实现(Cruiser[18],Blizzard[4]),这些都是很好的例证.就第 2 个问题,我们对 eMule 网络源码进行了详细分析,得出它具有两个主要特点:1) 前缀匹配路由:节点间的距离通过节点 Kad ID 的按位异或值来度量.例如:节点 *A* 和节点 *B* 的 Kad ID 分别为 1011 和 0010,则 *A* 与 *B* 的距离=1011⊕0010=1001.高位对距离的影响较大,如果两个节点高位相同位数(即前缀匹配)越多,则距离越小;2) 迭代式路由(iterative routing):当开始搜索某个目标时,在路由表查找到距离目标最近(局部最近)的一些节点信息(由 Kad ID、IP、TCP 端口和 UDP 端口组成),然后向它们发出查找请求.这样通常都能够得到一些距离更近的节点,然后再向那些更近的节点发送查找请求,按照这种方式进行路由,就能够得到距离目标最近(全局最近)的节点.这些特点使得 Rainbow 的设计成为可行.我们可以采用迭代式采集的方式,即从一批已知的初始节点开始,询问这些初始节点并等待返回下一批新节点信息,接着再询问返回的新节点以获得再下一批新节点信息.如此迭代,预计可以获得越来越多的节点信息.

考虑到和节点进行 TCP 通信的时间开销(包括建立 TCP 连接、发包后的延迟等待等)较大,而进行单次采集的时间又不可过长,因过长的采集时间会导致节点震荡的负面影响加剧.这就决定了单次采集的节点规模不可过大(典型值 10000 左右).本文设计 Rainbow 的目的是为了采集 eMule 网络中节点的候选标识集合{Kad ID, userID,IP}对应关系的信息,且单次采集获得的这些信息会以(Kad ID,userID,IP)为主键存入数据库表 PeerInfo.表的记录条数越多,对应的统计结果越有代表性.而区域采集固定了 Kad ID 的 *k* 位前缀,只能采集固有的 $1/(2^k)$ 的节点集合的候选标识集合对应关系信息.故本文采用了随机采集的方法,即单次采集不固定 *k* 位前缀,从节点总体中随机抽取一些节点以获得他们的信息.可知,单次采集的节点比区域采集获得的样本节点更具有随机性,而且,进行多次随机采集会比区域采集获得更多的记录条数.

3.2 eMule网络节点信息采集器实现

3.2.1 Rainbow 系统框架

如图 1 所示,Rainbow 系统框架由 3 个模块组成:节点采集模块(peer crawler)、节点信息收集模块(peer information gatherer)和写数据库模块(Write DB module).我们为 Rainbow 维护了一个种子节点(seed),其路由表会不断更新,节点采集模块首先与种子节点联系,获得 *u* 个初始节点(因种子节点的路由表不断更新,每次获得的 *u* 个初始节点一般会不同),然后与 eMule 网络进行 UDP 通信,展开迭代式采集以获得一个节点集合(简称 S_{UDP})

的部分信息(IP,Kad ID,UDP port 和 TCP port).节点信息收集模块和 S_{UDP} 中的节点进行 TCP 通信以获得更丰富的节点信息(userID).写数据库模块将 S_{UDP} 中的节点的完整信息写入数据库表 PeerInfo,并将其他的 log 信息(单次采集时长、采集节点数、在线节点比例)写入相应数据库表.

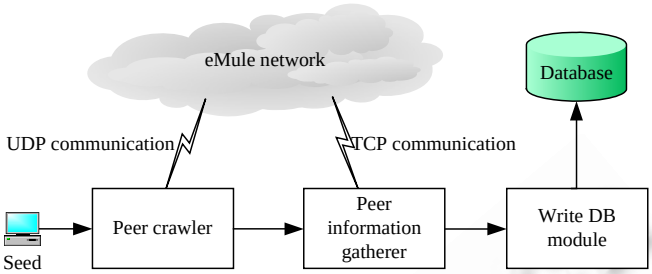


Fig.1 Framework of rainbow system

图 1 Rainbow 系统框架

3.2.2 Rainbow 采集算法

Rainbow 通过改造 eMule 客户端实现,模拟一个 eMule 网络正常节点,从种子节点获得初始节点列表后,加入 eMule 网络进行采集.Rainbow 采集算法采用了多线程实现,包括线程 1~线程 5,其详细的算法描述见算法 1.由描述可见,线程 1、线程 2 对应模块 1(节点采集模块)的功能,即和节点进行 UDP 通信,进行节点收集.其中:线程 1 负责给节点发送 UDP 查询消息 *bootstrap_req*,以进行迭代式采集;线程 2 负责分析 UDP 响应消息 *bootstrap_res*,将去重后的节点存储起来.线程 3、线程 4 对应模块 2(节点信息收集模块)的功能,即和节点进行 TCP 通信,获得节点的更丰富的信息.其中:线程 3 负责和节点建立 TCP 连接并给节点发送 TCP 查询消息 TCP *hello*,以获得节点更多的信息;线程 4 负责处理 TCP 响应消息 *TCP hello answer*,并将返回的 userID 信息写入相应节点中.线程 5 对应模块 3(写数据库模块)的功能,即将采集到的信息写入数据库.

算法 1. Rainbow 采集算法.

Input: seed;

Output: detailed information of peers above a threshold value *n*.

Data: *qelem*: struct{IP,UDP port,TCP port}

Data: *key*: struct{IP,UDP port,Kad ID}

Data: *peer*: struct{IP,UDP port,TCP port,Kad ID,userID,status} /*status=UDP_Requested, UDP_Responded, TCP_Requested, TCP_Responded, TCP_Cantcon*/

Data: shared queue *qpeersUDP*=queue of qelem elements

Data: shared queue *qpeersTCP*=queue of qelem elements

Data: shared map *mpeers*=map of ⟨*key*,*peer*⟩ elements

Data: bool *UDPover*=0 /*the Boolean variable to ensure the convergence of UDP communication*/

Data: bool *TCPover*=0 /*the Boolean variable to ensure the convergence of TCP communication*/

1: the main thread start thread 1 and the iterative crawl begin

Thread 1. UDP send thread

2: contact seed, and initialize *qpeersUDP* with *u* initial peers returned from seed

3: start thread 2 and thread 3

4: **while** *size*(*mpeers*)<=*n* /**n* is the maximum peer number of a single crawl*/

5: **do while** *size*(*qpeersUDP*)>0

6: **do** send UDP message *bootstrap_req* to *qpeersUDP*'s first element *p*

7: remove *p* from *qpeersUDP*

8: **if** $p \in mpeers$
9: **then** $p.status \leftarrow UDP\_Requested$
10: $wait(T_w)$ /*T_w is the waiting time needed for the last response *bootstrap_res* message*/
11: $UDPover \leftarrow 1$
12: start thread 5

Thread 2. UDP receive thread

13: **while** true
14: **do if** *UDPover*
15: **then** exit thread
16: wait for *bootstrap_res* message
17: **if** $wait(bootstrap\_res) > T_i$ /*T_i is the timeout time waiting for *bootstrap_res* messages*/
18: **then** sample some peers from *mpeers* and send *bootstrap_req* messages to them /*incentive mechanism*/
19: **for** each $p \in bootstrap\_res\ message$
20: **do** $qelem \leftarrow p.info$, $key \leftarrow p.info$, $peer \leftarrow p.info$
21 $peer.status \leftarrow UDP\_Responded$
22: **if** $key \notin mpeers$
23: **then** $mpeers.add(\langle key, peer \rangle)$
24: $qpeersUDP.add(qelem)$
25: $qpeersTCP.add(qelem)$

Thread 3. TCP send thread

26: start thread 4
27: **while** true
28: **do if** *TCPover*
29: **then** exit thread
30: **while** $size(qpeersTCP) > 0$
31: **do** wait the number of current *TCP connections*>*m*
32: try to establish asynchronous TCP connection with *qpeersTCP*'s first element *p*, and send TCP hello message to *p*
33: $p.status \leftarrow TCP\_Requested$
34: remove *p* from *qpeersTCP*

Thread 4. TCP receive thread

35: **while** true
36: **do if** *TCPover*
37: **then** exit thread
38: wait for *TCP hello answer* message
39: update the userID value of corresponding peer *p* in *mpeers*
40: disconnect TCP connection with *p*
41: $p.status \leftarrow TCP\_Responded$
42: **for** each *TCP connection* $tc \in$ *currently established TCP connections*
43: **do if** *tc*'s *waiting time*>=*T* /**T* is the maximum time of *TCP connection**/
44: **then** disconnect *TCP connection tc*

45: set the status of corresponding peer as *TCP_Cantcon*

Thread 5. write database thread

46: **for** each *peer*∈*mpeers*

47: **do** wait for *peer.status*=*TCP_Responded* or *TCP_Cantcon*

48: write the information of *peer* into *PeerInfo*

49: *TCPover*←1

3.2.3 Rainbow 采集算法收敛性分析

我们考虑线程 1、线程 2 在用 map 去重的情况下,从 N 个节点的集合中随机采集 n 个节点($n\in[1,N]$)是否收敛,即是否能在有限时间内完成特定采集任务.若是,为多长?首先对该问题进行形式化,该问题等价于从一个包含 N 个元素的集合 S_1 中,进行随机放回抽样,每次抽取 c 个($c\in[1,n]$).在考虑去重情况下,抽取 n 个不同元素到集合 S_2 中,其抽取次数是否有限?若有限,其抽取次数为多少?

例如:从 S_1={1,2,3,4,5}中抽取大小 n=4 的集合 S_2,每次抽 2 个,则 N=5,c=2.去重情况指:第 1 次抽到{1,2},第 2 次抽到{2,3},则现在抽到的元素集合为{1,2,3}(去重后).第 3 次抽到{1,3},则现在抽到的元素集合还是{1,2,3}(去重后),仍没有达到 4 个元素的要求.在第 4 次抽到{3,5},抽到的元素集合{1,2,3,5}(去重后),才满足 n=4.在这个过程中,抽取次数是有限的,其抽取次数为 4.

在此,我们为了求解这一问题对这个问题进行简化,考虑 c=1 的特殊情形,即从 S_1 中每次只抽取一个元素,求总共需要抽取多少次.这也对应了采集在最坏情况下(通常情况下,一次会抽取较多相互不重复的元素,比如 20 个)所需要的抽取次数.

我们对简化后的问题(c=1)进行分析,得出如图 2 所示的状态转移图.图中的圆圈表示状态,圆圈内数字表示 S_2 的大小$|S_2|$,箭头上的数字表示从一个状态转移到另一个状态的概率.令待抽集合为 S_1,抽到的集合为 S_2.初始时$|S_2|$=0,转移到状态$|S_2|$=1 的概率为 1,而维持在状态$|S_2|$=0 的概率为 0;当$|S_2|$=1 时,转移到$|S_2|$=2 的概率为(N−1)/N,维持在状态$|S_2|$=1 的概率为 1/N;依此类推,当$|S_2|$=i−1 时,转移到状态$|S_2|$=i 的概率为(N−i+1)/N,维持在状态$|S_2|$=i−1 的概率为(i−1)/N.

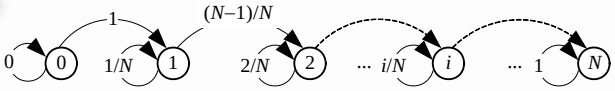


Fig.2 Chart of state transition

图 2 状态转移图

引理 1. 从包含 N 个元素的集合 S_1 中进行随机放回抽样到 S_2 中,令 q_i 为从状态$|S_2|$=i−1 转移到状态$|S_2|$=i 需要进行的抽取次数且 q_i 是随机变量,则 q_i 的数学期望满足:

$$E(q_i)=\frac{N}{N-i+1},\ i\in[1,N] \tag{3}$$

证明:由图 2 可知,抽 1 次从状态$|S_2|$=i−1 转移到状态$|S_2|$=i 的概率为 $\frac{N-i+1}{N}$,抽 2 次从状态$|S_2|$=i−1 转移到状态$|S_2|$=i 的概率为 $\frac{N-i+1}{N}\times\frac{i-1}{N}\times 2$,依此类推,则 $E(q_i)=\frac{N-i+1}{N}\sum_{j=1}^{\infty}\left[\left(\frac{i-1}{N}\right)^{j-1}\times j\right]\Rightarrow E(q_i)=\frac{N}{N-i+1}$. □

引理 2. 从大小为 N 的集合 S_1 中随机放回抽样 n 个不同的元素到 S_2 中,即$|S_2|$=n,$n\in[1,N]$.当每次只抽取一个元素即 c=1 时,令所需要的抽取次数为随机变量 $X(N,n,1)$,则 $X(N,n,1)$的数学期望满足:

$$E[X(N,n,1)]=\sum_{i=1}^{n}\frac{N}{N-i+1} \tag{4}$$

证明:该问题等价于计算图 2 中从状态$|S_2|$=0 转移到状态$|S_2|$=n 所需要的总抽取次数的数学期望.由引理 1

得知,单步的抽取次数的数学期望 $E(q_i)=\frac{N}{N-i+1}$, $i\in[1,n]$,易知,从状态$|S_2|=0$转移到状态$|S_2|=n$所需的n步的总抽取次数的数学期望 $E[X(N,n,1)]=E(q_1+q_2+...+q_n)=\sum_{i=1}^{n}E(q_i)=\sum_{i=1}^{n}\frac{N}{N-i+1}$. □

由公式(4)知,$E[X(N,1,1)]=1$;$n>1$ 时,$E[X(N,n,1)]>n$.

定理 1. 从大小为 N 的集合 S_1 中随机放回抽样 n 个不同的元素到 S_2 中,即$|S_2|=n,n\in[1,N]$.当每次只抽取 c 个元素时(c 个元素互不相同且 $c\in[1,n]$),令所需要的抽取次数为随机变量 $X(N,n,c)$,则 $X(N,n,c)$的数学期望满足:

$$E[X(N,n,c)]=\frac{E[X(N,n,1)]}{E[X(N,c,1)]} \tag{5}$$

证明:问题的最终效果是从集合 S_1 中抽取了 n 个不同的元素到 S_2 中,而获得该最终效果可有两种不同的抽样粒度:一种是一次从集合 S_1 中抽取 c 个不同的元素,即抽样粒度为 c(对应图 3 左侧椭圆的抽样方式);另一种是一次从集合 S_1 中抽取 1 个元素,即抽样粒度为 1(对应图 3 右侧椭圆的抽样方式).考虑抽样粒度为 c 的 $E[X(N,n,c)]$ 次抽取中的单步抽取(比如图 3 左侧小框 i),即考虑从大小为 N 的集合 S_1 中随机放回抽取 c 个不同元素的子问题,若抽样粒度为 1,则由引理 2 知,其抽取次数为 $E[X(N,c,1)]$.由图 3 易知

$$E[X(N,n,1)]=E[X(N,n,c)]\times E[X(N,c,1)]\Rightarrow E[X(N,n,c)]=\frac{E[X(N,n,1)]}{E[X(N,c,1)]}.$$ □

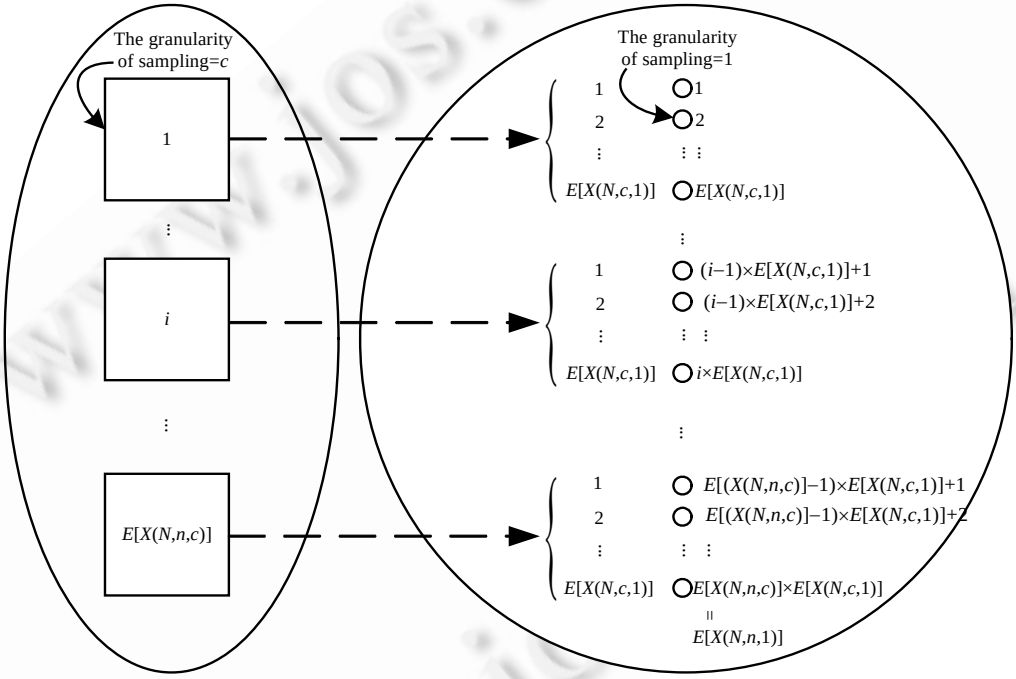


Fig.3 Chart of equivalent sample problem

图 3 抽取问题等价图

由公式(5)可见,当 $c=1$ 时, $E[X(N,n,c)]=\frac{E[X(N,n,1)]}{E[X(N,1,1)]}=E[X(N,n,1)]=$公式(4),所以公式(4)是公式(5)在 $c=1$ 时的特殊情形.当 $c=n$ 时, $E[X(N,n,n)]=\frac{E[X(N,n,1)]}{E[X(N,n,1)]}=1$,即当抽样粒度为 n 时,从大小为 N 的集合 S_1 中随机放回抽样 n 个不同元素的抽样次数的数学期望为 1,这也与实际情况相符.

定理 2. Rainbow 采集算法对节点集合 S_1($|S_1|=N$)进行随机采集、区域采集和完全采集都是收敛的.

证明:在不考虑去重所带来的额外抽取次数的情形下,Rainbow 采集算法的线程 1 的收敛性由线程 1 的语句 4 的阈值判断和线程 2 的语句 17、语句 18 引入的激励机制确保,线程 2 的收敛性由布尔量 *UDPover* 和线程 2 的语句 14、语句 15 确保;另一方面,在考虑去重所带来额外的抽取次数的情形下,从线程 1 的语句 6 可见,Rainbow 采集算法通过给节点 p 发送 UDP 消息 *bootstrap_req* 来进行迭代式采集,节点 p 在收到 *bootstrap_req* 消息时,会从本地路由表随机选择 20 个节点信息填充到 UDP 消息的 *bootstrap_res* 中返回.接着,线程 2 会处理返回的 *bootstrap_res* 消息,将其中 20 个节点信息去重存入 *mpeers* 中.综上,线程 1、线程 2 从节点集合 S_1 中进行采集的抽样粒度为 c=20.故由定理 1 得知,从 S_1 中抽取 n 个元素的抽取次数的数学期望为 $E[X(N,n,c)]=E[X(N,n,20)]$.当进行完全采集时,$n=N$,从而抽取次数的数学期望 $E[X(N,N,20)]=\dfrac{\sum_{i=1}^{N}\frac{N}{N-i+1}}{\sum_{i=1}^{20}\frac{N}{N-i+1}}$.因为采集总体大小$|S_1|=N\sim10^6>>20$,所以 $\sum_{i=1}^{20}\frac{N}{N-i+1}\approx 20$,故可得 $E[X(N,N,20)]\approx\dfrac{N(\ln(N+1)+\gamma)}{20}$($\gamma\approx0.577218$为欧拉常数).线程 1、线程 2 在考虑去重所带来额外的抽取次数的情形下进行完全采集是收敛的,且其收敛时间复杂度为 $O(N\ln N)$.因为随机采集和区域采集都只抽取一部分节点,故其抽取次数的数学期望都满足 $E[X(N,n,c)]<E[X(N,N,c)]$,$n<N$,所以线程 1、线程 2 在考虑去重所带来额外的抽取次数的情形下进行随机采集和区域采集也都是收敛的.

线程 3、线程 4 只是在线程 1、线程 2 收集了节点的前提下进一步和节点进行交互以获得更多的关于节点的信息,而线程 5 只是最后将收集到的信息写入数据库,所以线程 3~线程 5 都不需要考虑去重所带来额外的抽取次数的情形.由算法描述可见,线程 3 的收敛性由布尔量 *TCPover* 和线程 3 的语句 28、语句 29 确保,线程 4 的收敛性由布尔量 *TCPover* 和线程 4 的语句 36、语句 37 确保.此外,由线程 5 的算法描述可见,它只有一个简单的 for 循环,故线程 5 是收敛的.

综上,Rainbow 采集算法进行随机采集、区域采集和完全采集都是收敛的. □

由定理 2 可知,Rainbow 采集算法的线程 1、线程 2 是收敛的.本文在实验中令 $N=4\times10^6$,n=20000,则抽取次数的数学期望 $E[X(N,n,20)]=\dfrac{\sum_{i=1}^{20,000}\frac{4\times10^6}{4\times10^6-i+1}}{\sum_{i=1}^{20}\frac{4\times10^6}{4\times10^6-i+1}}\approx1003$,说明比不考虑去重的抽取次数(1000 次)仅多带来时间开销 0.3%.故下节在计算线程 1、线程 2(即 UDP 通信)的时间复杂度时不考虑去重所带来的时间开销.

3.2.4 Rainbow 采集算法时间复杂度分析

由线程 1~线程 5 可见,3 个模块——节点采集器、节点信息收集器和写数据库模块可以实现并发操作.为保证在写数据库时共享变量 *mpeers* 的大小固定以确保迭代器不出现指针异常,必须在 UDP 通信处理完成后才能启动写数据库线程,即模块 1、模块 3 需要串行操作.故可得 3 个模块的并发流水线图如图 4 所示.

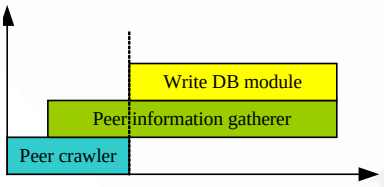


Fig.4 Pipeline chart of Rainbow's modules

图 4 Rainbow 各模块流水线图

UDP 通信(线程 1、线程 2)的时间复杂度计算如下:假设从种子节点获得的初始节点数为 u,向节点发 *bootstrap_req* 包的回包率为 p,令线程 1 中的语句 6~语句 9 所需时间为 t_1(纳秒级),UDP 通信延迟(从发送 *bootstrap_req* 包到收到 *bootstrap_res* 包所需延迟)为 t_2,则 UDP 通信获得 n 个节点所需要的时间 T_{UDP} 可通过 $T_{UDP}=(u\times t_1+t_2)+(20\times u\times p\times t_1+t_2)+[(20\times u)^2\times p\times t_1+t_2]+...+[(20\times u)^{(i-1)}\times p\times t_1+t_2]+T_w$ 推导出:

$$T_{UDP}=u\times t_1+i\times t_2+\frac{(20\times u)pt_1[(20u)^{i-1}-1]}{20u-1}+T_w \tag{6}$$

公式(6)中,i 满足$(20u)^i p>n\Rightarrow i=\left\lceil\frac{\log(n/p)}{\log 20u}\right\rceil$.而且,因 $t_2<<t_1$,所以 $T_{UDP}\approx i\times t_2+T_w$.

TCP 通信(线程 3、线程 4)的时间复杂度计算如下:假设能同时建立的异步 TCP 连接数阈值为 m,TCP 连接超时时间为 T,则线程 3、线程 4 中循环段中语句所需的 CPU 耗时以及 TCP 通信延迟(TCP 包在网络中的耗时)相比 T 都可以忽略不计.则与 n 个节点进行 TCP 通信在最坏情况下(每个节点都需要时间 T 才获得响应)所需时间 T_{TCP} 可计算为

$$T_{TCP}=\left\lceil\frac{n}{m}\right\rceil\times T \tag{7}$$

写数据库线程(线程 5)的时间复杂度计算为:假设写一条数据库记录耗时为 t_3(可变值,与数据库表现有大小有关).线程 5 循环段中语句所需 CPU 耗时相比 t_3 可以忽略不计,则将 n 条节点信息去重写入 PeerInfo 所需时间 $T_{WriteDB}$,计算如下:

$$T_{WriteDB}=n\times t_3 \tag{8}$$

3.2.5 Rainbow 采集算法空间复杂度分析

我们从全局来考虑 Rainbow 采集算法的空间开销(包括线程 1~线程 5),其空间开销主要由 3 个全局结构带来,分别是 *qpeersUDP*,*qpeersTCP* 和 *mpeers*(见 rainbow 采集算法全局量部分的第 3 行~第 5 行).其中,*qpeersUDP*,*qpeersTCP* 都是元素为 *qelem* 的队列,而 *qelem* 是由 IP,UDP port 和 TCP port 构成的结构体,占 4B+2B+2B=8B(1B=1 字节)内存.*mpeers* 是由结构体 *key* 和 *peer* 组成的 map,其中,*key* 为由 IP,UDP port 和 Kad ID 组成的结构体,占 4B+2B+16B=22B,而 *peer* 为由 IP,UDP port,TCP port,Kad ID,userID 和 status 构成的结构体,占 4B+2B+2B+16B+16B+1B=41B 内存.算法中的 map 用红黑树实现,一般而言,其空间开销=r×用静态数组来存储的大小(r 的典型值大约为 3~4).综上,Rainbow 采集算法在进行 n 个节点的采集时,在最坏情况下(情况较好时,*qpeersUDP*,*qpeersTCP* 在算法中一般会及时清空)的空间开销 *SE* 可通过:$SE=n\times\{[2\times(8+4)]+r\times(22+41)\}$推导出:

$$SE=n\times(24+63\times r) \tag{9}$$

3.3 小 结

为从 eMule 网络中采集 3 个候选标识集合{Kad ID,userID,IP}之间的对应关系信息,设计实现了 eMule 网络节点信息采集器——Rainbow,并对其采集算法进行了详细介绍,同时对其收敛性和时空复杂度进行了详细的理论分析.总之,Rainbow 采集器具有如下特点:

1) 可以对 eMule 网络进行随机采集.此外,经过适当的配置,也可以进行完全采集和区域采集;
2) 可以和节点进行 TCP 通信,获得相比传统采集器更加丰富的信息;
3) 理论分析表明,采集算法是收敛的(定理 2),且时空复杂度(公式(6)~公式(9))不高.

4 实验结果及分析

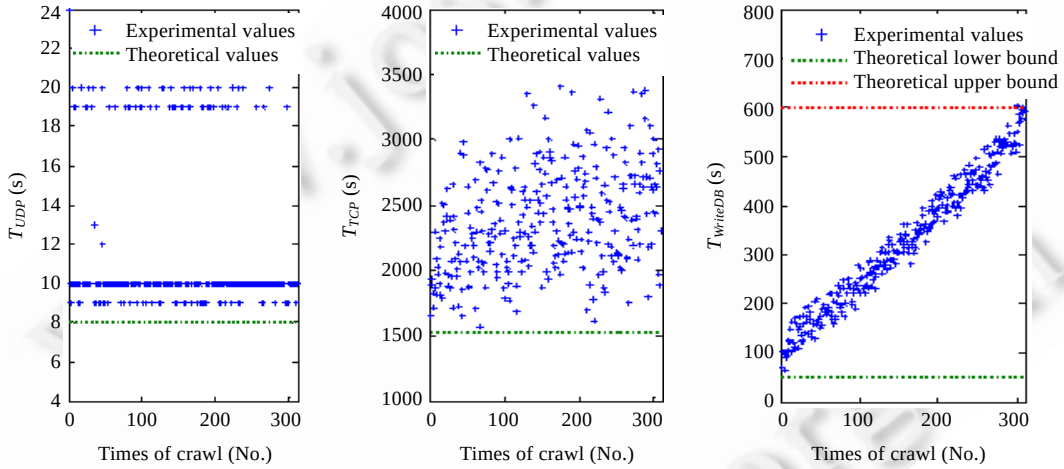
4.1 实验数据

应用 Rainbow 在如下配置的两台机器上进行了数据采集.硬件环境:2.21GHz 双核处理器/8GB 内存/100Mb/s 局域网带宽服务器、2.8GHz 双核处理器/2GB 内存/2Mb/s 带宽 ADSL PC 各一台.软件环境:Windows Server 2003 SP2,SQL Server 2005 Developer Edition.为了进一步加快数据采集速度,我们让 Rainbow 在这两台机器上持续运行了 9 天(5/3/2009~5/11/2009),然后将数据库进行合并.我们一共进行了 310 次随机采集,PeerInfo 表共获得以(Kad ID,userID,IP)为主键的约 320 万条去重节点信息记录,其中,可进行 TCP 通信并获取了 userID 信息的记录有 1 166 784 条.

4.2 Rainbow采集性能

本节就 Rainbow 采集器在变量 $p,u,n,m,r,T_w,T,t_1,t_2,t_3$ 相关取值下进行时间与空间复杂度的理论值与实验值对比分析.公式(6)中,考虑较坏情况,令 t_1~10^{-9}s,t_2=1s,则 $t_2 \propto 10^9 t_1$.据我们实验观察 p 的取值约为 0.2.同时,若取 u=20,n=20000,T_w=5s,则可求得 T_{UDP}=8.00003202s≈8s.公式(7)中,若取 T=45s,n=20000,m=600,可求得 T_{TCP}=1,530s.公式(8)中:当 PeerInfo 表小于 10 万条记录时,t_3 较小大概取值 0.0025s;当 PeerInfo 表大于 300 万条记录时,t_3 较大大概取值 0.03s.若取 n=20000,可求得 $T_{WriteDB}$=50s~600s.公式(9)中,若取 n=20000,r=4,可求得 SE≈5.26MB.

就 310 次随机采集的性能数据对 Rainbow 采集器的收敛性、时间复杂度和空间复杂度进行实验分析,并与相应的理论值进行对比.实验结果显示,310 次随机采集都是收敛的,所以收敛性获证.3 个模块的实验耗时 T_{UDP},T_{TCP},$T_{WriteDB}$ 与公式(6)~公式(8)的理论取值的对比如图 5 所示.由图可见,实验值和理论值相差不大,310 次随机采集的 T_{UDP} 实验值的算数平均值≈11.78s,约为理论值的 1.47 倍.T_{TCP} 实验值的算数平均值≈2379.41s,约为理论值的 1.55 倍.$T_{WriteDB}$ 实验值的算数平均值≈325.34s,基本处于理论值的上下界范围之内.此外,经我们实测观察,Rainbow 采集器在运行时所占的内存空间在 12MB 左右波动,约为公式(9)理论值的 2.28 倍.考虑到采集器在运行时会加载一些库从而带来额外内存开销,而且 Rainbow 的总耗内存很小,所以 Rainbow 的空间复杂度是可以接受的.总之,实验显示 Rainbow 采集器是收敛的,时空复杂度的实验值和理论值基本处于同一数量级.此外,由 T_{UDP},T_{TCP},$T_{WriteDB}$ 理论值和实验值可见,模块 2(节点信息收集模块)的处理时间较长,为主要的瓶颈段.故模块 1、模块 3 的串行操作并不会带来额外的时间开销(如图 4 所示).



(a) T_{UDP} (b) T_{TCP} (c) $T_{WriteDB}$

Fig.5 Rainbow's performance comparison between experimental and theoretical values

图 5 Rainbow 性能实验值与理论值对比

4.3 稳定因子(SF)计算与分析

从数据表 PeerInfo 中抽出 3 个标识:Kad ID,userID,IP,并以他们为主键构建新表 KadID_userID_IP,共获得 1 166 784 条不重复记录,在表 KadID_userID_IP 上根据公式(1)计算它们及其组合的稳定因子(SF)值.获得 SF 计算结果及 ID 合适度排名和原因分析见表 1.由表 1 比较 SF 值可见,{userID}的取值最大,根据定义 2,{userID}是候选节点标识集合 $2^{\{\text{Kad ID,userID,IP}\}}-\{\Phi\}$中的最佳节点标识.

Table 1 *SF* value comparison

表 1 *SF* 值比较

Candidate identifier	*SF*	ID fitness rank	Cause
{Kad ID}	1.166 540	2	Kad ID can be freely changed, thus {Kad ID} is less stable than {userID}
{IP}	1.012 907	4	DHCP (dynamic host configuration protocol) makes IP changes a lot, therefore {IP} has a low *SF* value
{userID}	1.271 760	1	Most stable as userID has public key encryption algorithm and credit system to ensure its stability
{Kad ID,IP}	1.000 448	6	{Kad ID,IP} has a low *SF* value as it changes a lot
{userID,IP}	1.001 140	5	Even if userID is stable, DHCP makes IP changes frequently
{Kad ID,userID}	1.163 981	3	Even if userID is stable, the instability Kad ID makes {Kad ID,userID} change
{Kad ID,userID,IP}	1	7	Since table KadID_userID_IP is composed of (Kad ID,userID,IP), therefore its *SF* equals to 1

4.4 候选标识正常度(CIN)计算与分析

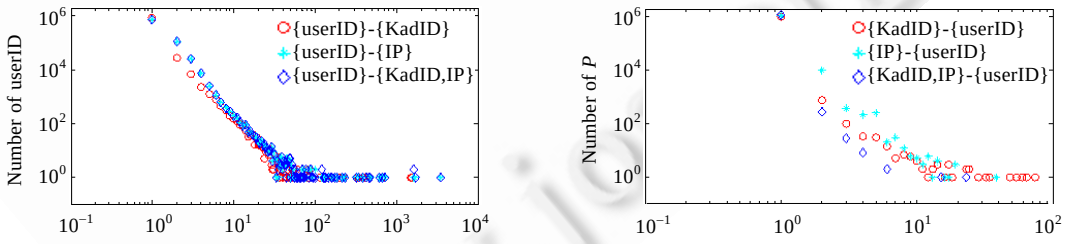
采用{userID}作为最佳节点标识后,利用公式(2)对 *CIN* 值进行了计算,获得结果见表 2.计算结果表明,*CIN* 值满足如下关系:$CIN_{\{userID\}}=CIN_{\{Kad\ ID,userID\}}=CIN_{\{userID,IP\}}=CIN_{\{Kad\ ID,userID,IP\}}>CIN_{\{Kad\ ID\}}>CIN_{\{Kad\ ID,IP\}}>CIN_{\{IP\}}$.其中,$CIN_{\{Kad\ ID,IP\}}$比 $CIN_{\{IP\}}$稍大是因为{Kad ID}比{IP}更稳定.例如,实验数据显示,{userID}与其他标识的多对一关系中,与{Kad ID,IP}的多对一关系只有 312 个,而与{IP}的多对一关系有 11 178 个,较前者多 10 866 个.

Table 2 *CIN* value

表 2 *CIN* 值

Candidate identifier	*CIN* value	Candidate identifier	*CIN* value
{userID}	1	{Kad ID}	0.869 784
{Kad ID,userID}	1	{Kad ID,IP}	0.652 680
{userID,IP}	1	{IP}	0.641 180
{Kad ID,userID,IP}	1		

由实验数据可以获得最佳节点标识{userID}与其他 *CIN*<1 的候选标识 *P*(*P*={Kad ID},{IP}或{Kad ID,IP})的关系,如图 6 所示.图 6(a)反映了{userID}与 *P* 的一对一关系和一对多关系数量,图 6(b)反映了 *P* 与{userID}的一对一关系和一对多关系数量(因{userID}与 *P* 之间的关系复杂,{userID}与 *P* 的一对一关系数量并不等于 *P* 与{userID}的一对一关系数量).*CIN* 越大,对应的散点图越陡、尾巴越短,表示{userID}与 *P* 关系的一一对应关系比率越高,即关系越正常.由图可见,图 6(b)的 *P* 与{userID}的一对多关系数量远小于图 6(a)的{userID}与 *P* 的一对多关系,故可以主要观察图 6(a),得关系正常度排序(从好到差)为:{userID}-{Kad ID}关系,{userID}-{Kad ID,IP}关系,{userID}-{IP}关系,这验证了表 2 的 *CIN* 计算值排序结果.



(a) {userID}-*P* relation
(a) {userID}与 *P* 的关系

(b) *P*-{userID} relation
(b) *P* 与{userID}的关系

Fig.6 Relationship between {userID} and other candidate peer identifiers *P*

图 6 {userID}与其他候选节点标识 *P* 之间的关系

4.5 {userID}-{Kad ID}关系统计分析

本节定量分析 Kad ID 别名,主要对{userID}与{Kad ID}的复杂关系进行统计分析,以获得{userID}与{Kad

ID}之间的相互关系(见表 3).

1) 一对一关系.图 6(a)最左上角的圆圈表示{userID}与{Kad ID}的一对一关系,有 874 650 个,占{userID}与{Kad ID}关系的 95.33%;图 6(b)最左上角的圆圈表示{Kad ID}与{userID}的一对一关系,有 999 222 个,占{Kad ID}与{userID}关系的 99.9%;

2) 一对多关系(包括瞬态和累积一对多关系)反映在图 6(a)中.实验观察到,{userID}与{Kad ID}有 42 806 个一对多关系,占{userID}与{Kad ID}关系的 4.67%,说明存在相当比率的一对多关系;

3) 多对一关系(包括瞬态和累积多对一关系)反映在图 6(b)中.实验观察到,{userID}与{Kad ID}有 987 个多对一关系,只占{userID}与{Kad ID}关系的 0.1%,说明多个{userID}对应同一个{Kad ID}的情况极少.

总之,虽然非正常情形的多对一和一对多关系所占总关系数量的比率不高,但是在节点规模以百万计的 eMule 网络中,这个数目是不容忽略的.例如,图 6(a)中有 29 660 个 1 对 2 关系,1 对 19 关系也有 31 个.表明 eMule 网络中节点(由{userID}标识)与{Kad ID}关系非一一对应性相当显著,反映出 Kad ID 别名问题很严重.同时也表明,eMule 网络中的安全隐患(即 Kad ID 可以随意指定)仍然存在,从而方便攻击者进行 eMule 网络攻击,例如 Sybil 攻击和蒙蔽攻击(eclipse attack,模拟大量恶意节点进行网络分割以进行信息截获与欺骗)[23,24].

Table 3 {userID} vs. {Kad ID} relation

表 3 {userID}与{Kad ID}相互关系

Relation	Number of {userID}-{Kad ID} relations	Ratio (%)	Number of {Kad ID}-{userID} relations	Ratio (%)
One to one	874 650	95.33	999 222	99.9
One to multiple	42 806	4.67	987	0.1
Total	917 456	100	1 000 209	100

Sybil 攻击的防御方法目前存在集中式和分布式两类,其中:集中式防御方法违背了 eMule 网络的分布式特性而实用性差;分布式防御方法[25−27]当前仍处于探索阶段,问题还没有获得完美解决.蒙蔽攻击的防御方法[23,24]目前较少.考虑到 userID 不易变性是因为采用了 RSA 公/私钥加密算法的挑战响应机制来防止 userID 欺骗和冒充,是否可以采用同样的方式来解决 Kad ID 可以任意指定的安全隐患,是一个值得深入思考的问题.

4.6 {userID}的应用有效性

{userID}作为最佳节点标识可以在如下场景下应用:eMule 网络文件源的准确定位;eMule 网络的节点跟踪;eMule 网络节点行为分析等.由第 4.3 节 *SF* 值计算可知,{userID}是集合 $2^{\{\text{Kad ID,userID,IP}\}}-\{\Phi\}$中最稳定的标识.

为进一步证明这一点,本节采用 Rainbow 对 eMule 网络节点的共享文件信息进行了为期 6 天的探测(5/24/2009~5/29/2009),共获得节点拥有的文件条数约 550 万条,然后分别采用{Kad ID},{IP},{userID},{Kad ID,IP},{userID,IP},{Kad ID,userID},{Kad ID,userID,IP}作为标识对节点所拥有的文件进行了统计,实验结果如图 7 的箱线图所示.

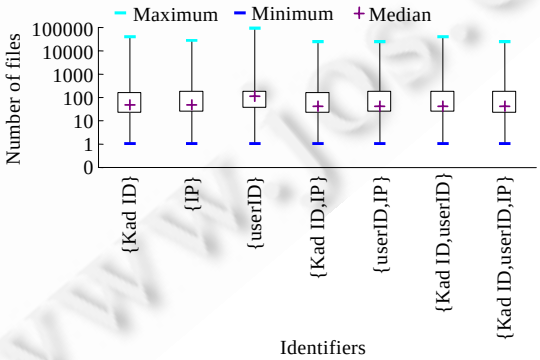


Fig.7 Boxplot of number of files corresponding to different identifiers

图 7 不同标识对应文件数量箱线图

由图 7 可见,{userID}在短期(只有 6 天)的测量中也能对应最多的文件数量,故采用{userID}作为节点标识可以更准确地探测节点所拥有的文件情况.从而证明了{userID}是集合{Kad ID,userID,IP}中最稳定(佳)的节点标识,同时也证明了稳定因子是一个能够准确度量标识合适性的测度.

5 结束语

本文主要解决 eMule 网络中的最佳节点标识问题.首先定义了节点标识稳定性测度——稳定因子,并给出了节点标识集合中最佳节点标识的形式化定义;随后,设计并实现了可采用随机采集方式工作的 eMule 网络节点信息采集器——Rainbow,并对其收敛性、时间复杂度和空间复杂度进行了分析;运用 Rainbow 对真实 eMule 网络进行一段时间的采集后,我们获得 eMule 网络标识:Kad ID,userID,IP 之间对应关系的去重记录 1 166 784 条,通过稳定因子计算得出{userID}是最佳节点标识;之后,对{userID}与{Kad ID}的关系进行了统计分析;最后,对{userID}的应用场景进行了举例说明.本文针对 eMule 网络的主要贡献有:(1) 设计实现了一个节点信息采集工具——Rainbow,并从理论和实验角度证明了 Rainbow 采集算法是一个收敛且时空复杂度不高的算法;(2) 提出了标识的稳定性测度:稳定因子用于度量标识的合适度;(3) 通过稳定因子计算和实验验证得出{userID}是 eMule 网络的最佳节点标识,为 eMule 网络的研究奠定了基础.

eMule 网络的研究受到越来越多的关注,更进一步的研究工作可以围绕如何提高 eMule 网络的服务质量展开,比如 eMule 网络中粗俗文件的鉴别与过滤、eMule 网络完全拓扑图的采集与分析.

致谢 感谢淘宝软件有限公司的龚才春先生,我们从与他的交流讨论中得到许多启发.

References:

[1] eMule. 2009. http://www.emule-project.net

[2] Maymounkov P, Mazieres D. Kademlia: A peer-to-peer information system based on the XOR metric. In: Proc. of the Int'l Workshop on Peer-to-Peer Systems (IPTPS). 2002. 53−65.

[3] Yu J, Fang CF, Xu J, Chang EC, Li ZJ. ID repetition in Kad. In: Proc. of the 9th IEEE Int'l Conf. on Peer-to-Peer Computing (P2P). 2009. 111−120. [doi: 10.1109/P2P.2009.5284551]

[4] Steiner M, Ennajjary T, Biersack EW. Long term study of peer behavior in the KAD DHT. IEEE/ACM Trans. on Networking, 2009, 17(5):1371−1384. [doi: 10.1109/TNET.2008.2009053]

[5] Ipoque. 2009. http://www.ipoque.com/resources/internet-studies/

[6] Karagiannis T, Broido A, Faloutsos M, Claffy K. Transport layer identification of P2P traffic. In: Proc. of the ACM SIGCOMM/USENIX Internet Measurement Conf. (IMC). 2004. 121−134. [doi: 10.1145/1028788.1028804]

[7] Sen S, Spatscheck O, Wang DM. Accurate, scalable in-network identification of P2P traffic using application signatures. In: Proc. of the 13th Int'l World Wide Web Conf. (WWW). 2004. 512−521. [doi: 10.1145/988672.988742]

[8] Bhagwan R, Savage S, Voelker GM. Understanding availability. In: Proc. of the 2nd Int'l Workshop on Peer-to-Peer Systems (IPTPS). 2003. 256−267.

[9] Kutzner K, Fuhrmann T. Measuring large overlay networks—The overnet example. In: Proc. of the 14th KiVS. 2005. 193−204. [doi: 10.1007/3-540-27301-8_16]

[10] Steiner M, En Najjary T, Biersack EW. A global view of KAD. In: Proc. of the Internet Measurement Conf. (IMC). 2007. 117−122. [doi: 10.1145/1298306.1298323]

[11] Steiner M, Biersack EW, Ennajjary T. Actively monitoring peers in KAD. In: Proc. of the 6th Int'l Workshop on Peer-to-Peer Systems (IPTPS). 2007.

[12] Saroiu S, Gummadi PK, Gribble SD. A measurement study of peer-to-peer file sharing systems. In: Proc. of the Multimedia Computing and Networking (MMCN). 2002. 156−170.

[13] Stutzbach D, Rejaie R, Sen S. Characterizing unstructured overlay topologies in modern P2P file-sharing systems. In: Proc. of the 5th ACM SIGCOMM Conf. on Internet Measurement (IMC). 2005. 49−62. [doi: 10.1109/TNET.2007.900406]

[14] Wang Y, Yun XC, Li YF. Measuring and characterizing topologies of P2P networks. Journal of Software, 2008,19(4):981−992 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/19/981.htm [doi: 10.3724/SP.J.1001.2008.00981]

[15] Bittorrent. 2009. http://www.bittorrent.com

[16] Naoumov N, Ross K. Exploiting P2P systems for DDoS attacks. In: Proc. of the 1st Int'l Conf. on Scalable Information Systems (Infoscale). 2006. .[doi: 10.1145/1146847.1146894]

[17] Stutzbach D, Rejaie R. Improving lookup performance over a widely-deployed DHT. In: Proc. of the IEEE Infocom. 2006. [doi: 10.1109/INFOCOM.2006.329]

[18] Stutzbach D, Rejaie R. Understanding churn in peer-to-peer networks. In: Proc. of the 6th ACM SIGCOMM Conf. on Internet Measurement (IMC). 2006. 189−202. [doi: 10.1145/1177080.1177105]

[19] Falkner J, Piatek M, John JP, Krishnamurthy A, Anderson T. Profiling a million user DHT. In: Proc. of the Internet Measurement Conf. (IMC). 2007. 129−134. [doi: 10.1145/1298306.1298325]

[20] Liu Q, Xu P, Yang HT, Peng Y. Research on measurement of peer-to-peer file sharing system. Journal of Software, 2006,17(10): 2131−2140 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/2131.htm [doi: 10.1360/jos172131]

[21] Kulbak Y, Bickson D. The eMule protocol specification. 2005. http://www.cs.huji.ac.il/labs/danss/p2p/resources/emule.pdf

[22] Douceur JR. The sybil attack. In: Proc. of the 1st Int'l Workshop on Peer-to-Peer Systems (IPTPS). 2002. 251−260.

[23] Singh A, Ngan TW, Druschel P, Wallach DS. Eclipse attacks on overlay networks: Threats and defenses. In: Proc. of the IEEE Infocom. 2006. [doi: 10.1109/INFOCOM.2006.231]

[24] Singh A, Castro M, Druschel P, Rowstron A. Defending against eclipse attacks on overlay networks. In: Proc of the 11th European SIGOPS Workshop. 2004. [doi: 10.1145/1133572.1133613]

[25] Wang HH, Zhu YW, Hu YM. An efficient and secure peer-to-peer overlay network. In: Proc. of the 30th Annual IEEE Conf. on Local Computer Networks (LCN). 2005. 764−771. [doi: 10.1109/LCN.2005.27]

[26] Yu HF, Kaminsky M, Gibbons PB, Flaxman A. SybilGuard: Defending against sybil attacks via social networks. IEEE/ACM Trans. on Networking, 2008,16(2008):576−589. [doi: 10.1109/TNET.2008.923723]

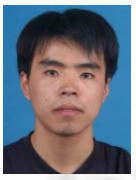
[27] Yu HF, Gibbons PB, Kaminsky M, Xiao F. SybilLimit: A near-optimal social network defense against sybil attacks. In: Proc. of the IEEE Symp. on Security and Privacy. 2008. 3−17. [doi: 10.1109/SP.2008.13]

附中文参考文献:

[14] 王勇,云晓春,李奕飞.对等网络拓扑测量与特性分析.软件学报,2008,19(4):981−992. http://www.jos.org.cn/1000-9825/19/981.htm [doi: 10.3724/SP.J.1001.2008.00981]

[20] 刘琼,徐鹏,杨海涛,彭芸.Peer-to-Peer 文件共享系统的测量研究.软件学报,2006,17(10):2131−2140. http://www.jos.org.cn/1000-9825/17/2131.htm [doi: 10.1360/jos172131]

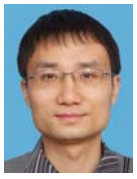
刘祥涛(1977－),男,湖南涟源人,博士生,CCF 学生会员,主要研究领域为 P2P 网络,数据挖掘.



陈小军(1979－),男,博士生,主要研究领域为网络信息安全,P2P 网络安全.

程学旗(1971－),男,博士,研究员,博士生导师,主要研究领域为社会计算,分布式计算.

白硕(1956－),男,博士,研究员,博士生导师,主要研究领域为自然语言处理,网络安全.



李洋(1978－),男,博士,主要研究领域为信息安全,分布式计算,P2P 网络.

刘悦(1971－),女,博士,副研究员,主要研究领域为信息检索,社区挖掘与分析,分布式计算.