软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2011,22(4):722−735 [doi: 10.3724/SP.J.1001.2011.03770]

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

Laurel:一种混合式数据分发覆盖网*

郑 重+, 王意洁, 马行空

(国防科学技术大学 计算机学院 并行与分布处理国家重点实验室,湖南 长沙 410073)

Laurel: A Hybrid Overlay Network for Data Distribution

ZHENG Zhong+, WANG Yi-Jie, MA Xing-Kong

(National Key Laboratory for Parallel and Distributed Processing, School of Computer, National University of Defense Technology, Changsha 410073, China)

+ Corresponding author: E-mail: zhengzhong@nudt.edu.cn

Zheng Z, Wang YJ, Ma XK. Laurel: A hybrid overlay network for data distribution. *Journal of Software*, 2011,22(4):722−735. http://www.jos.org.cn/1000-9825/3770.htm

Abstract: As an infrastructure for data distribution, overlay networks must incorporate efficient routing and adequate robustness in order to achieve fast and accurate data distribution in an environment with a high node churn. Considering that the existing overlay networks mostly focus on a single optimization objective and fail to ensure routing efficiency and robustness. Simultaneously, a hybrid overlay network for data distribution, Laurel, is proposed in this paper. Laurel achieves a better trade-off between routing efficiency and robustness by combining the inter-cluster multiple structured topologies with the intra-cluster unstructured topologies. Laurel also provides mechanisms for a dynamic, concurrent cluster creation, cluster departure, and load balance to make data distribution more adaptive to the dynamic network environment. Experimental results show that compared with existing overlay networks, Laurel can support faster and more accurate data distribution, even when a large amount of nodes fail in the system and balance the load within clusters.

Key words: hierarchical overlay network; hybrid overlay network; clustering; data distribution; publish/subscribe; P2P

摘 要: 覆盖网是各种数据分发应用的基础架构.在节点波动的网络环境中实现快速而准确的数据分发,对覆盖网提出了两个要求:高效的数据路由;较强的系统鲁棒性.已有的覆盖网构建方法多侧重于某个方面的优化,因而未能充分权衡数据路由效率与系统鲁棒性.提出了一种混合式数据分发覆盖网——Laurel.Laurel 通过簇间多重结构化拓扑与簇内非结构化拓扑的结合,实现了路由效率与鲁棒性的高效折衷,并通过簇动态创建、退出以及负载平衡机制增强了对动态变化环境的适应能力.实验结果表明,相对于已有方法,Laurel 即使在节点频繁波动的网络环境中也能快速而准确地分发数据,并且具有较好的负载平衡效果.

关键词: 层次式覆盖网;混合式覆盖网;分簇;数据分发;发布/订阅;P2P

中图法分类号: TP393 **文献标识码**: A

* 基金项目: 国家自然科学基金(60873215); 国家重点基础研究发展计划(973)(2011CB302601); 湖南省自然科学杰出青年基金(S2010J5050); 高等学校博士学科点专项科研基金(200899980003)

收稿时间: 2009-05-04; 定稿时间: 2009-10-23

随着网络技术的发展,Internet 已经汇聚了大量的资源,正逐步演变为无处不在的计算平台.为了满足不断增长的对信息共享的需求,产生了很多基于数据分发的应用,如文件共享、视频会议、新闻分发、电子商务、环境监测等.因此,各种数据分发技术成为关注的热点,如应用层组播[1]、发布/订阅技术[2]等.而覆盖网则是在缺乏集中控制的网络环境中应用这些技术所必需的基础架构.

各种数据分发应用的共同需求是将数据源产生的数据在广域网的范围内快速而准确地分发至不断波动的用户群体.特别地,如紧急情况处理[3]、网络中心战[4]等应用,格外强调数据分发的准确性、时效性以及系统对网络波动的适应能力.因此,这些应用对其覆盖网的构建与维护提出了两个要求:支持根据需求快速而准确定位数据分发的目的节点;适应节点频繁加入、退出的动态网络环境.

根据拓扑结构,传统的覆盖网可分为两大类:非结构化覆盖网和结构化覆盖网.非结构化覆盖网没有固定、严格的拓扑结构,因而维护开销较低且具有较强的鲁棒性,但也因此缺乏对节点及资源定位的高效支持,通常都是采用泛洪、gossip[5]等方法分发数据,典型的系统有 Gnutella[6],Freenet[7]等.结构化覆盖网的拓扑由确定性的算法严格控制,并且资源或资源元信息也按照特定规则存放在特定的节点上,因而能够支持节点及资源的高效定位,但维护开销较大,鲁棒性较差,典型的方法有 Chord[8],CAN[9],Pastry[10],Tapestry[11].

传统的覆盖网都是扁平结构.出于提高性能的目的,产生了很多层次式的覆盖网[12−22].这些覆盖网根据不同的原则将节点组织成若干簇,每一个簇组织成一个覆盖网,同时,簇间也通过超级节点组织成另一层覆盖网.这些方法根据其目的采用了不同的构建策略:基于超级节点组织高层网络以利用节点能力的异构性;基于邻近性分簇以降低路由延迟;基于节点兴趣分簇以减少不必要的数据分发;等等.但这些覆盖网各自侧重于某个方面的优化——或者路由效率较高而鲁棒性较差,或者鲁棒性较好而路由效率较低,缺少对数据路由效率与系统鲁棒性的充分权衡.

本文提出了一种混合式数据分发覆盖网 Laurel.Laurel 根据兴趣对节点进行分簇以减少不必要的数据分发,通过在簇间采用结构化拓扑以实现高效的数据路由,通过在簇间采用多重连接及在簇内采用非结构化拓扑以保证较高的系统鲁棒性.相对已有的覆盖网而言,Laurel 保证即使在节点频繁波动的网络环境中也能够快速而准确地分发数据.

1 相关工作

根据特定的优化目标,已有的层次式覆盖网协议在设计的各个方面采用了不同的策略.基于对这些工作的分析,整个层次式覆盖网的设计空间大致可以分为 4 个维度:簇间拓扑结构、簇内拓扑结构、超级节点数目以及分簇原则.

早期的层次式覆盖网的主要优化目标是充分利用节点能力的异构性,于是,计算或通信能力强的节点被选作超级节点,并基于非结构化拓扑组成上层覆盖网;而能力普通的节点则基于星形拓扑以某个超级节点为中心簇聚,典型的如 KaZaA[12],eMule[13].KaZaA 在节点选择超级节点簇聚时使用了邻近性原则,eMule 则没有明确的分簇原则.

为了提高数据查询效率,有的层次式覆盖网基于结构化拓扑构建簇间连接,并在簇内采用星形或结构化的拓扑,典型的如 TBSN[14],Grapse[15],Jelly[16],Canicula[17],Omicron[18],以及 Garces-Erice 等人提出的层次式 P2P 系统框架[19].TBSN 在簇内采用星形拓扑,并根据节点资源的语义相似性进行节点分簇.其余的方法则采用结构化的簇内拓扑,其中:Grapse,Jelly,Canicula 都根据邻近性进行分簇,且每个簇只有一个超级节点;Omicron 没有明确的分簇原则,并通过在每个簇内设置多超级节点以增强系统鲁棒性;Garces-Erice 等人提出的一般框架也是根据邻近性分簇,而且通过使得簇内超级节点数目可任意设置来提高系统的鲁棒性.

为了提高系统鲁棒性,新出现的一些层次式覆盖网在其不同层次引入非结构化拓扑,如 Yang 等人提出的混合式 P2P 系统[20],HP2P[21],TERA[22].Yang 等人提出的混合式 P2P 系统可以根据不同的应用需求采用不同的分簇原则,簇间采用结构化拓扑,簇内采用非结构化拓扑,并且每个簇内只有一个超级节点.HP2P 则根据邻近性分簇,也同样采用簇间结构化、簇内非结构化的混合式拓扑,但簇内可以设置多个超级节点.TERA 则根据节点兴

趣分簇,簇间、簇内均采用非结构化拓扑,并且没有超级节点.

根据簇间拓扑、簇内拓扑两个主要设计维度,上述层次式覆盖网可以大致归纳如图 1 所示.

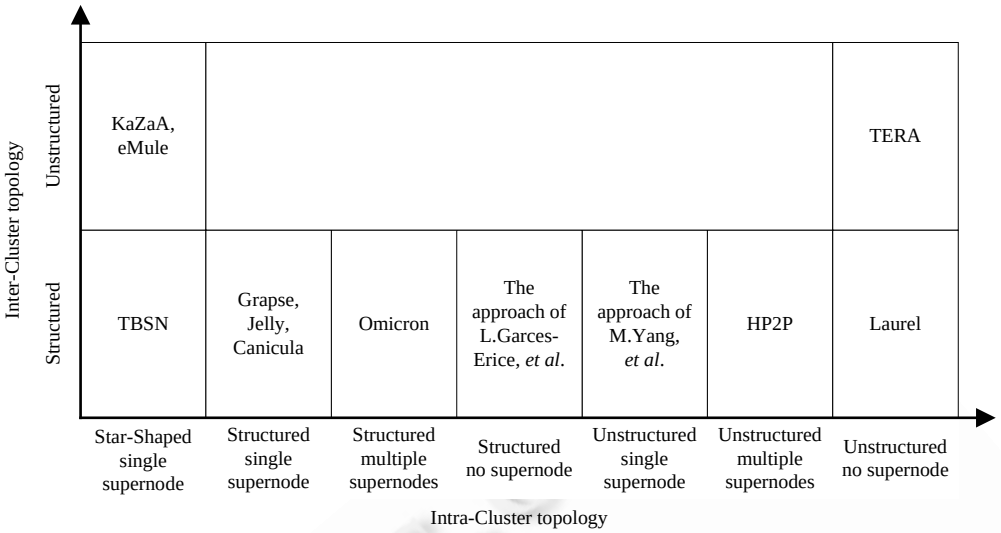


Fig.1 Categories of hierarchical overlay networks

图 1 层次式覆盖网分类

就趋势而言,图 1 中的覆盖网由左至右鲁棒性递增,由下至上则簇间路由效率递减,鲁棒性递增.由图 1 可知,已有的覆盖网协议并未穷尽设计空间.而且,前述簇内采用多个或没有超级节点的覆盖网协议,除了 TERA,都未考虑簇的并发创建、意外失效及超级节点间的负载平衡等问题,而这些问题实际上会影响到簇内采用多超级节点在增强系统鲁棒性方面的实际效果.Laurel 的设计思路就是在这个设计空间中寻找路由效率与鲁棒性的高效折衷,并通过解决簇的并发创建、意外失效及负载平衡等问题增强系统的鲁棒性.

2 Laurel 体系结构

2.1 基本结构

Laurel 中的节点根据其兴趣主题(topic)组织成相应的簇(cluster),每个簇内部采用非结构化拓扑,而簇间采用结构化拓扑.因为 Chord 的简单、高效,Laurel 用其作为簇间结构化拓扑.实际上,基于某些特定需求,也可在簇间采用其他结构化拓扑来改造 Laurel.簇间的连接由每个簇内任意数目的节点负责,这些节点被称为骨干节点(bone node),其余的节点被称为叶子节点(leaf node).簇内骨干节点与叶子节点的实际比例可以根据特定需要加以调整,而簇内每个节点都充当骨干节点,即相当于簇内没有超级节点.图 2 显示了 Laurel 的基本结构.

Laurel 以簇为单位组织成 Chord 环,即利用 Hash 函数根据簇对应的兴趣主题为每个簇分配标识(cluster id),而每个簇则根据其标识确定自己在 Chord 环空间中的位置.簇之间的前驱、后继、指向表等连接关系实际上通过各个簇的骨干节点之间的连接来实现,而簇内多骨干节点的存在则使得两个簇间的连接往往不止一条,这意味着各个簇实际上是由多个交错的 Chord 环串联在一起.簇间的结构化连接有助于提高簇间路由的效率,而建立多重结构化连接则有助于减轻骨干节点失效所产生的影响.

Laurel 簇内所有节点组织成一个非结构化覆盖网,称为簇覆盖网(cluster overlay).另外,簇内骨干节点间需要彼此通信以更新、修复其维护的簇间连接.为了消除它们之间彼此寻找的困难以及由此带来的通信开销,可以在它们之间维护专门的连接,即在簇内再构建一个非结构化的骨干节点覆盖网,称为骨干覆盖网(bone overlay).因而,在 Laurel 的每个簇中实际上有两层非结构化覆盖网,其中一个由所有节点组成,用于数据的分发;

另一个由所有骨干节点组成,用于簇间连接的维护.这两层非结构化覆盖网的构建与维护都可以使用很多已有的方法,如 CYCLON[23].CYCLON 的基本思想是,每个节点周期性地选择一个邻居节点来交换各自随机选择的部分邻居,这种邻居的周期性交换使得相应的覆盖网拓扑近似随机图,因而具有很高的连通性、鲁棒性以及较低的直径.

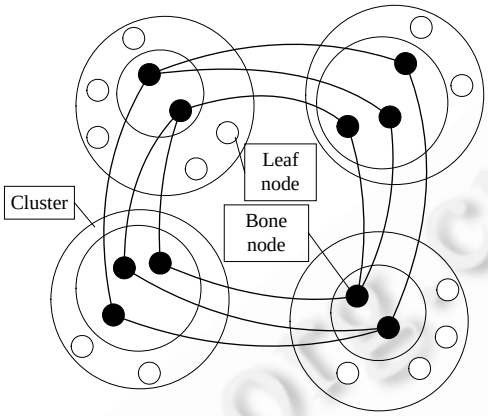


Fig.2 Architecture of Laurel

图 2 Laurel 体系结构

2.2 数据分发

Laurel 不仅适用于较小数据(如发布/订阅系统中的事件),也适用于较大数据(如文件)的分发.分发的数据有其所属的主题,对某个主题感兴趣的节点可以通过加入相应的簇来接收该主题的数据.这个过程相当于应用层组播中节点加入某个组播组,或发布/订阅系统中节点订阅某个主题的数据.

对于较小数据的分发,如果数据源节点是叶子节点,那么它可以在簇内利用随机行走的方式[24]将数据发送到簇内的骨干节点,然后由骨干节点利用簇间结构化连接将数据路由到数据主题对应的簇,最后在该簇内通过泛洪或 gossip 的方式分发数据;如果数据源节点是骨干节点,那么它可以直接通过簇间连接将数据路由到相应的簇.图 3 显示了数据发送的基本过程.

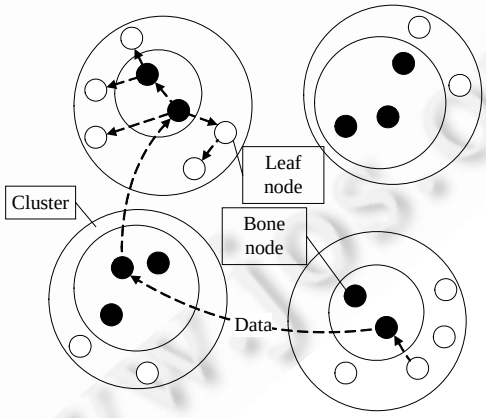


Fig.3 Data distribution over Laurel

图 3 Laurel 数据分发

对于较大数据的分发,数据源节点可以首先利用前述方法将数据的元信息(数据名称、大小、分块情况等)发送至数据主题对应的簇.然后,数据源节点临时加入该簇的簇覆盖网,并利用类似 CREW[25]的方法在簇内进行数据分发,即簇内各节点根据已获取的数据元信息,周期性地随机选择某个簇覆盖网邻居,向其请求自己缺少

的数据块.如此反复,直到获取完整的数据为止.

Laurel 的簇间结构化连接保证了簇间数据路由的跳数只与簇个数 C 有关,即为 $O(\log C)$.对于数据源是骨干节点的情况,$O(\log C)$就是从源到目的簇的路由跳数.对于数据源是叶子节点的情况,数据从源到本簇骨干节点的传输还会带来额外的路由跳数,而这个跳数的具体大小则与源所在簇内的骨干节点比例有关.若某簇内骨干节点相对于所有节点的比例为 b,考虑到簇覆盖网的拓扑近似随机图,因此,该簇内的某个叶子节点发送数据至任一骨干节点所需跳数的期望为 $1/b$.这就意味着,当簇内骨干节点比例超过 50%时,从某个叶子节点发送数据至任一骨干节点的平均跳数不到 2 步.

3 Laurel 的构建与维护协议

3.1 节点邻居关系

3.1.1 簇内邻居关系

簇内每个节点都各自维护一个簇内邻居表(cluster neighbor list),这形成了一个包含簇内所有节点的非结构化覆盖网,即簇覆盖网.除此之外,簇内每个骨干节点还各自维护一个簇内骨干邻居表(bone neighbor list),这就形成了一个骨干节点覆盖网,即骨干覆盖网.图 4 显示了簇内的双层拓扑结构.

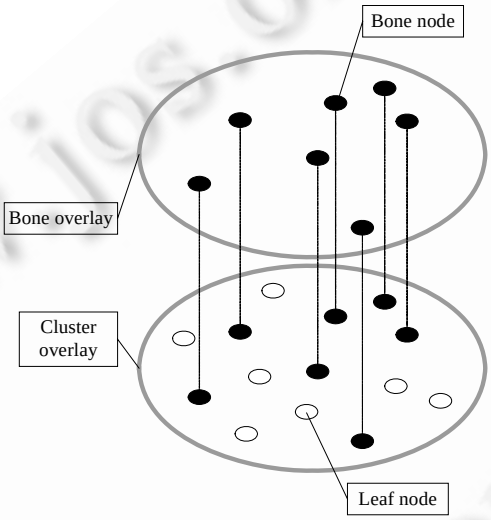


Fig.4 Topology in the cluster of Laurel

图 4 Laurel 簇内拓扑结构

所有节点根据簇内邻居表选择节点转发数据,骨干节点则利用骨干邻居表来彼此交换信息以更新、修复簇间连接.两种邻居表都利用 CYCLON 方法维护,其基本思想是,通过相邻节点间周期性地交换部分邻居来保持节点波动条件下的拓扑连通性.

3.1.2 簇间邻居关系

簇间邻居关系根据簇在 Chord 环空间中的位置来确定,每个簇有自己的前驱簇、后继簇以及指向表等,这些关系通过不同的簇中骨干节点间的关系来实现.每个骨干节点负责维护一个前驱表(predecessor list)、后继表(successor list)、后备后继表(backup successor list)以及一个指向表(finger list).与 Chord 的数据结构不同,为了增强鲁棒性,Laurel 的前驱表、后继表都包含多个表项而非一个,而后备后继表则是一个二维数组.图 5 显示了 Laurel 协议的基本数据结构.

```
Class LaurelProtocol {
   …
   BigInteger clusterId;                    //Cluster id
   IntraNeighbor clusterNeighborList[c];    //Cluster neighbor list
   IntraNeighbor boneNeighborList[b];       //Bone neighbor list
   Node predecessorList[p];                 //Predecessor list
   Node successorList[s];                   //Successor list
   Node fingerList[f];                      //Finger list
   Node backupSuccessorList[l][s];          //Backup successor list
   …
}

Class IntraNeighbor {
   int age;              //The age of this item
   Node neighbor;   //Intra-Cluster neighbor
}

Class Node {
   …
   //The protocol object running at this node
   LaurelProtocol localProtocol;
   …
}
```

Fig.5 Basic data structure of Laurel

图 5 Laurel 的基本数据结构

前驱表所含节点为前驱簇中的若干骨干节点,后继表所含节点为后继簇中的若干骨干节点.后备后继表第 1 维对应本簇的后继簇之后的顺次若干个簇,而第 2 维则对应这些簇中的若干骨干节点.指向表的组织与 Chord 中的基本一致,只是其中每一项的节点为 Chord 环相应位置上的簇中的某个骨干节点.

3.2 节点加入

节点可以根据自身能力来决定是否作为骨干节点加入 Laurel.当节点加入时,首先需要查询当前覆盖网中是否已有自己兴趣主题对应的簇存在:如果有,则直接作为骨干节点或叶子节点加入该簇;如果没有,且自己是作为骨干节点加入,则创建一个新簇.也就是说,Laurel 中没有显式的簇创建.新加入节点根据当前覆盖网中是否存在自己对应的簇及自己加入的角色来自动判断是否启动一个簇创建过程.

在 Laurel 系统的初始化阶段,应至少已存在 3 个簇组成 Chord 环结构,其中每个簇应至少包含一个骨干节点.这种初始拓扑可以由管理员手工配置,而之后,系统的扩展过程则根据节点加入的相关协议自组织地进行.下面就按照节点加入 Laurel 的主要步骤来介绍相关的协议细节.

3.2.1 节点查询簇

为了加入 Laurel,新节点必须知道已在 Laurel 中的某个节点.该已知节点的获取有多种途径,因其具体实现超出了本文的关注范围,故这里不再赘述.新节点首先用 Hash 函数获得自己兴趣主题对应的簇标识 $cluster\_id_{new}$,再通过已知节点以随机行走的方式获取其簇内的某个骨干节点(若已知节点是骨干节点,则直接返回其自身),然后从这个骨干节点出发,沿 Chord 环查询自己所属簇标识在环上的后继簇($successor(cluster\_id_{new})$),称为目标簇(target cluster).实际返回的查询结果是目标簇中的某个骨干节点,称为目标节点(target node).根据该目标节点对应的簇标识与新节点自身簇标识的关系,Laurel 分别启动节点加入簇以及节点创建簇的过程.

3.2.2 节点加入簇

如果目标节点的簇标识与新节点的相同,则说明目标簇就是新节点要加入的簇,此时,根据新节点是作为骨干节点还是叶子节点而加入,分别采取下述行动:

- 若作为叶子节点加入,则通过目标节点加入目标簇的簇覆盖网即可;
- 若作为骨干节点加入,则首先通过目标节点加入目标簇的簇覆盖网及骨干覆盖网,然后从自己的骨干覆盖网邻居处各取一个前驱、后继及一列后备后继加入自己的前驱表、后继表、后备后继表,并直接复制目标节点的指向表为自己的指向表.

3.2.3 节点创建簇

如果目标节点的簇标识与新节点的不同,则说明新节点所归属的簇尚未创建,此时,根据新节点是作为骨干节点还是叶子节点而加入,分别采取下述行动:

- 若作为叶子节点加入,则返回失败,或作为骨干节点重启加入过程以创建新簇;
- 若作为骨干节点加入,则创建新簇,即通过目标节点从目标簇中随机挑选若干骨干节点加入自己的后

继表,并从目标簇的前驱簇中随机挑选若干骨干节点加入自己的前驱表;通过自己的前驱节点泛洪通知其所在簇内骨干覆盖网中的骨干节点,将其后继修改为本节点;通过自己的后继节点泛洪通知其所在簇内骨干覆盖网中的骨干节点,将其前驱修改为本节点;建立自己的指向表和后备后继表.由此引起的其他簇中骨干节点所维护的指向表、后备后继表中的错误,由那些节点自身通过周期性的检测更新纠正.

3.2.4 对并发簇创建的处理

对于新节点创建簇的情况,可能存在并发操作的问题,即在很短的时间内有一个以上的新节点企图通过同一个目标簇启动簇创建.如图 6 所示,如果这些新节点具有相同的兴趣主题,则会导致在 Chord 环空间的同一位置上出现多个具有同一标识但却分裂的簇;如果这些新节点的兴趣主题不一样,则会导致簇间连接的错误.因为 Laurel 的簇间存在多重连接,所以无法直接使用 Chord 的自适应算法来解决并发问题.

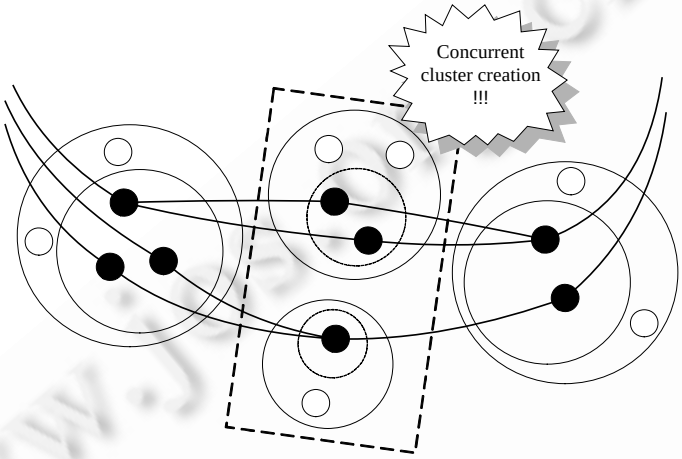


Fig.6 Result of concurrent cluster creation

图 6 并发簇创建的后果

为了解决这个问题,Laurel 引入了簇令牌(cluster token).每个簇都有唯一一个簇令牌,由其中某个骨干节点维护.令牌内容为当前 Chord 环空间中以该簇作为后继的簇标识的范围(*leftBoundary*,*rightBoundary*],其中,*leftBoundary* 为该簇前驱簇的标识,*rightBoundary* 为该簇标识.当新节点需要创建簇时,需按以下步骤进行:

(1) 通过目标节点在其骨干覆盖网中利用泛洪查询获取簇令牌节点及令牌;

(2) 判断自己的簇标识 $cluster\_id_{new}$ 是否落在簇令牌所含标识范围(*leftBoundary*,*rightBoundary*)内:

- 若是,则通过该簇令牌节点(而不是目标节点)创建新簇,并创建新簇对应令牌为(*leftBoundary*, $cluster\_id_{new}$),修改原令牌为($cluster\_id_{new}$,*rightBoundary*),并将其归还给原令牌节点;
- 若不是,则放弃创建簇,等待固定时间后重新启动加入过程.

通过设置令牌,簇并发创建节点得以协调动作,抢先获取令牌的节点先启动簇创建过程;而其他节点等待一段时间,估计新簇稳定后再重新启动加入过程.另外,为了增强鲁棒性,簇令牌在簇中可以存放多个备份.上述方法能否有效地处理簇的并发创建,依赖于其能否保证新簇创建的过程是原子操作.故此,我们证明以下定理.

定理 1. Laurel 协议能够保证新簇创建过程是原子操作.

证明:不失一般性,假设有两个新加入节点 *A*(簇标识为 $cluster\_id_A$)与 *B*(簇标识为 $cluster\_id_B$)试图通过同一个目标簇(簇标识为 $cluster\_id_C$)创建新簇,即该目标簇同时是节点 *A* 与节点 *B* 两者在 Chord 环空间中的后继.根据 *A* 与 *B* 两个新节点簇标识的关系,分两种情况来证明:

(1) 若 $A=successor(cluster\_id_B)$

假设此时节点 *A* 已从目标簇中的节点 *C* 处获取令牌($cluster\_id_{pre}$,$cluster\_id_C$),则节点 *B* 必须等待节点 *A* 将

修改后的令牌($cluster\_id_A$,$cluster\_id_C$)归还节点 C 后才能获取该令牌.当节点 B 获取该令牌,由于 $cluster\_id_B$ 不属于($cluster\_id_A$,$cluster\_id_C$),于是,节点 B 等待固定时间后重启加入过程.因此,此种情况下,Laurel 可以保证新簇创建是原子操作.

(2) 若 $B=successor(cluster\_id_A)$

假设此时节点 A 已从目标簇中的节点 C 处获取令牌($cluster\_id_{pre}$,$cluster\_id_C$),则节点 B 必须等待节点 A 将修改后的令牌($cluster\_id_A$,$cluster\_id_C$)归还节点 C 后才能获取该令牌.当节点 B 获取该令牌时,若 $cluster\_id_B=cluster\_id_A$,则 $cluster\_id_B$ 不属于($cluster\_id_A$,$cluster\_id_C$),所以节点 B 等待固定时间后重启加入过程;否则,$cluster\_id_B$ 属于($cluster\_id_A$,$cluster\_id_C$),所以,节点 B 通过节点 C 启动簇创建过程.由于节点 A 已通过节点 C 启动簇创建过程,节点 C 的前驱已修改为节点 A,因而,此时节点 B 启动的新簇创建过程实际上是在节点 A 的新簇创建过程完成之后进行的.因此,此种情况下,Laurel 同样可以保证新簇创建是原子操作.

综上,定理得证. □

3.3 失效与错误处理

对连接失效及错误的处理机制是影响系统鲁棒性的关键因素,而 Laurel 的混合式拓扑结构为高效地处理失效和错误提供了良好的基础.Laurel 中没有显式的节点及簇退出机制,而是直接将其作为意外失效来处理.

3.3.1 簇内连接失效的处理

簇内两层覆盖网的维护均使用 CYCLON 方法,该方法能够通过节点间周期性的邻居交换检测剔除失效节点,从而在节点波动的条件下保持覆盖网的连通性.对于骨干节点,当其发现自己没有活着的簇内邻居或骨干邻居时,还可以通过自己前驱的后继来获取自己簇内的其他节点作为邻居.

3.3.2 簇间连接失效与错误的处理

骨干节点的失效会破坏簇间连接,而簇的动态创建则会导致部分骨干节点指向表及后备后继表的错误.因此,骨干节点需要周期性地检测、更新自己的簇间连接,尽力保证簇间后继关系始终正确.事实上,只要簇间后继关系正确,簇间路由就一定能够保证正确.

Laurel 对骨干节点指向表中的失效或错误的处理与 Chord 类似,即骨干节点周期性地通过 Chord 环查询更新指向表中某一随机表项.对骨干节点前驱表、后继表、后备后继表的失效与错误,则可以利用簇间的多重连接来进行更新、修复.下面就从失效与错误处理两个方面分别介绍 Laurel 如何更新修复这 3 种簇间连接.

(1) 失效处理

- 当骨干节点发现自己的某个前驱(后继)失效,则从自己在簇内骨干覆盖网中任一邻居处获取它的前驱(后继),将其加入自己的前驱表(后继表);
- 当骨干节点发现自己所有的后继都失效,并且从本簇内的骨干节点处也无法获得新的后继时,则从自己的指向表中任一活着的节点出发在 Chord 环上搜索可以作自己后继的骨干节点:
  - ✓ 若找到,则将其加入自己的后继表;
  - ✓ 若未找到,则确认后继簇失效,将后备后继作为新的后继.

(2) 错误处理

骨干节点周期性地检查其前驱表、后继表、后备后继表的正确性,并通过基于 Chord 环的查询来更新错误的表项.

3.4 负载平衡

簇间路由实际上主要是通过骨干节点的指向表来进行的,而指向表的构建与维护则依赖于前驱表和后继表.因此,为了保证同一个簇内的骨干节点间的路由负载平衡,需要使它们有同等的机会出现在其他簇骨干节点的指向表、前驱表、后继表中.特别是对于新加入的骨干节点,应使其他簇的骨干节点及时获知它的存在.

归根结底,避免簇间连接集中于少量骨干节点,不仅有助于簇内骨干节点间的负载平衡,而且也能够增强系统的鲁棒性.为此,每个骨干节点周期性地随机选择指向表(前驱表、后继表)的某一表项,用其在相应簇内随机选

择的某个骨干邻居替换该表项本身.需要注意的是,簇内骨干节点覆盖网的维护方法 CYCLON 可以看作是一个节点随机采样方法,因此,随机选择某个骨干覆盖网邻居,也就是从该骨干覆盖网中随机采样节点.

4 性能评价

Laurel 协议的目标是实现路由效率与鲁棒性的高效折衷.具体来说,就是在保证较高的路由效率的前提下尽可能地提高系统的鲁棒性.为了检验 Laurel 在簇间路由效率与系统鲁棒性之间的折衷效果以及负载平衡机制的性能,我们将 Laurel 与 TERA,HP2P 以及 Yang 等人的混合式 P2P 系统进行比较.我们基于 P2P 协议模拟器 PeerSim[26]实现了这些方法,并进行了以下的对比实验:

(1) 簇间路由效率,即在不同节点规模与簇个数条件下,数据簇间路由的平均所需跳步数;

(2) 系统鲁棒性,即在连续节点失效及瞬时节点失效的情况下,数据簇间路由错误率的变化情况;

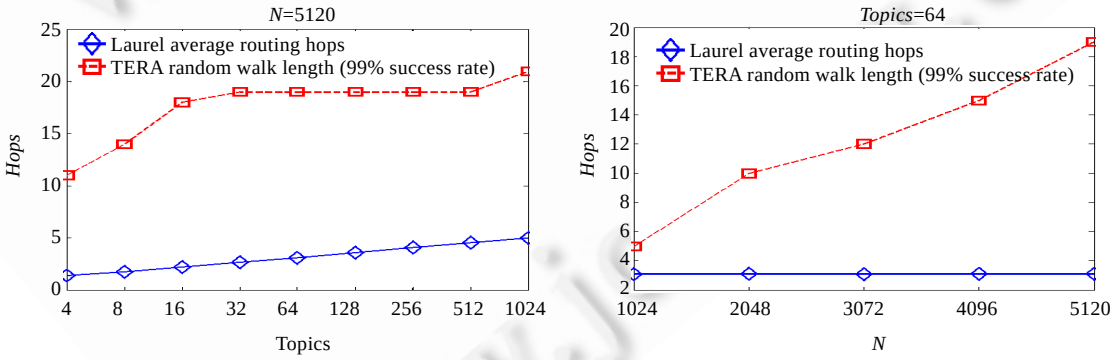
(3) 簇内负载平衡,即同一簇内的簇间路由负载平衡程度.

TERA 的簇间路由性能依赖于每个节点维护的簇访问入口点表(access point table,简称 APT)的大小,在实验中,我们将其设为 100,这个数值实际上已使得 TERA 中节点的存储开销超过 Laurel.原 HP2P 协议中每个簇根据节点性能选择少数节点充当超级节点,为公平起见,在实验中我们扩展了 HP2P,使其所有节点均充当超级节点,同时,Laurel 中所有节点也均作为骨干节点加入.Yang 等人的方法中每个簇只有一个超级节点,簇间连接完全由这些超级节点负责,为方便起见,下面我们称该方法为超级节点网络(supernode network).另外,实验中各簇节点及各种主题的数据均按照 Zipf 分布随机生成,即值 i 出现的概率为 i^{-a},其中,a 为 Zipf 参数,实验中设为 1.

4.1 路由效率

Laurel,HP2P 以及超级节点网络的簇间路由都通过结构化连接进行,理论上可以保证只需 $O(\log C)$(C 为簇个数)跳即可将数据分发至相应的簇.TERA 的簇间路由通过簇间的并发随机行走实现,因此不能保证一定能够将数据分发至相应簇.尽管通过增大随机行走的长度可以提高路由成功率,但在网络规模较大的情况下还是无法保证 100%地成功.我们分两类情况进行路由效率的对比:节点总数一定而簇个数发生变化;簇个数一定而节点总数发生变化.

与理论预期相符,实验结果表明,Laurel,HP2P 与超级节点网络的路由效率的数据曲线基本重合.因此,为了图片清晰,我们在图 7 中没有画出 HP2P 与超级节点网络的数据曲线.



(a) Number of topics changes
(a) 簇个数变化

(b) Network size changes
(b) 节点总数变化

Fig.7 Routing efficiency

图 7 路由效率

图 7(a)显示了在节点总数(N)为 5 120、簇个数(topics)从 4 增加到 1 024 的条件下,Laurel 的平均簇间路由跳数以及 TERA 保证 99%路由成功率所必需的随机行走长度的变化情况.随着簇个数呈指数级的增多,Laurel

的平均路由跳数呈线性增长,这与对 Laurel 结构化簇间连接性能的理论预期是一致的;TERA 为实现 99%成功率的路由所必需的随机行走长度也随着簇个数的增长而增长,且始终远大于 Laurel 的平均路由跳数.

图 7(b)显示了在簇个数为 64、节点总数(N)从 1 024 增加到 5 120 的条件下,Laurel 的平均簇间路由跳数以及 TERA 保证 99%路由成功率所必需的随机行走长度的变化情况.随着节点总数的增多,Laurel 的平均路由跳数几乎没有发生变化,这也与理论预期相一致;TERA 为实现 99%成功率的路由所必需的随机行走长度却随着节点总数的增长而显著增长.

需要注意的是,在实验中,为了保证成功率与低延迟,TERA 实际上并发进行 4 个随机行走,这意味着其实际的路由通信开销应为图中随机行走长度的 4 倍左右.综上,Laurel,HP2P 及超级节点网络的簇间路由效率基本一致,且在各种情况下均显著高于 TERA.

4.2 鲁棒性

在基于分簇的覆盖网上进行数据分发,簇间路由是从源节点到目标簇的整个数据传输过程的核心步骤.因此,我们通过检测节点失效对簇间路由错误率的影响来衡量系统的鲁棒性.具体而言,分两类情况进行鲁棒性的对比:每隔一段时间,固定比例的在线节点失效;大量节点瞬时失效.在两种情况中,节点总数均设为 1 024,簇个数(主题数)均设为 64,TERA 的随机行走长度均设为 11.

图 8 显示了从 12 000 时刻开始,每隔 1 500 单位时间随机选择的 5%的在线节点失效条件下,各种方法簇间路由错误率的变化情况.由图 8 可知,超级节点网络的错误率在 12 000 时刻后即迅速上升,HP2P 的错误率在 48 000 时刻后才开始逐步上升,TERA 的错误率保持在 0.05 以下,而 Laurel 的错误率则始终保持为 0.采用单一簇间连接的超级节点网络以及采用双层非结构化拓扑的 TERA 在此条件下的性能符合预期.尽管拓扑结构与 Laurel 类似,但由于其缺乏簇意外失效及簇内拓扑动态维护机制,HP2P 在失效节点逐步增多后便难以修复损坏的簇间连接,从而导致路由性能逐步恶化.而通过非结构化的簇内拓扑与多重簇间连接的结合以及相应的失效与错误处理机制,Laurel 获得了最优的表现.

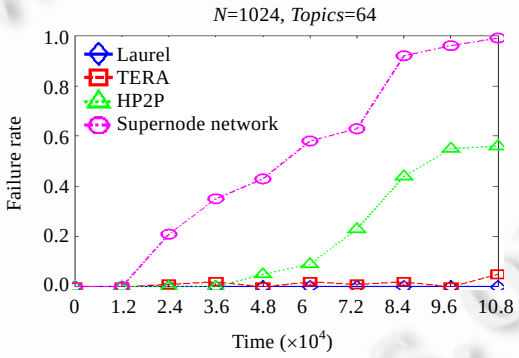
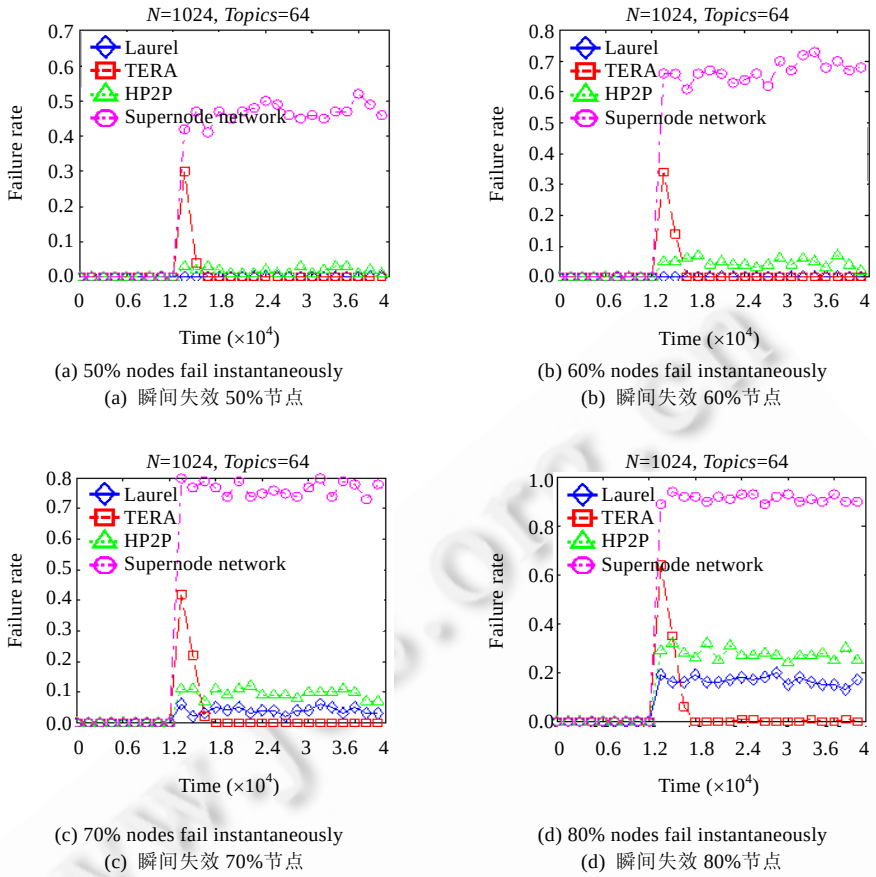


Fig.8 Routing failure rate with continuous node failure

图 8 连续节点失效时的簇间路由错误率

图 9 显示了在 12 000 时刻,分别有随机选择的 50%,60%,70%,80%的节点瞬时失效对簇间路由错误率产生的影响.图 9(a)和图 9(b)显示,在不超过 60%的节点失效的情况下,Laurel 的错误率始终保持在最低水平,并且错误率在节点失效时刻的附近也没有明显波动.图 9(c)和图 9(d)则显示,在 70%以上节点失效的情况下,尽管 Laurel 瞬时的错误率低于 TERA,但其恢复能力不如 TERA.虽然类似的拓扑结构为 HP2P 带来了接近 Laurel 的性能,但簇意外失效及簇内拓扑动态维护机制的缺乏,导致其无法在大量节点瞬时失效后修复拓扑,因而其错误率在 4 种情况下均高于 Laurel.另外,意料之中的是,超级节点网络在各种条件下均表现出最差的鲁棒性.



(a) 50% nodes fail instantaneously
(a) 瞬间失效 50%节点

(b) 60% nodes fail instantaneously
(b) 瞬间失效 60%节点

(c) 70% nodes fail instantaneously
(c) 瞬间失效 70%节点

(d) 80% nodes fail instantaneously
(d) 瞬间失效 80%节点

Fig.9 Routing failure rate with instantaneous node failure

图 9 瞬间节点失效时的簇间路由错误率

综上,Laurel 在各种节点失效的情况下均表现出优于超级节点网络与 HP2P 的鲁棒性,并且除了 70%以上节点瞬时失效的情况以外,其表现还不低于甚至优于采用双层非结构化拓扑的 TERA.

4.3 负载平衡

由于超级节点网络的每个簇内只有一个超级节点,而簇间路由又完全由其负责,因此,其簇内负载平衡性能显然低于其他 3 种方法.所以,在这里的讨论中我们将其排除在外.下面,我们就在节点总数一定(1 024)的情况下,设置不同的簇个数(16,32,64),以实验对比 Laurel,TERA,HP2P 的簇内负载平衡性能.

图 10 显示了不同簇个数条件下,Laurel,TERA,HP2P 各簇内部的簇间路由负载的相对标准差(relative standard deviation,简称 RSD).相对标准差即样本标准差与均值的商,大致反映了样本值相对均值的平均偏离程度.我们用这个指标来衡量负载平衡程度,其值越低,则表示负载越平衡.由图 10 可以看出:Laurel 的各簇簇内负载相对标准差最大也就在 0.2 左右,大多低于 TERA 的相应值或与之相差不大;而 HP2P 的各簇簇内负载相对标准差则大多显著高于 Laurel 与 TERA 的相应值.这表明,Laurel 通过簇间连接的周期性更新,的确能够获得较好的簇内负载平衡性能;而 TERA 的簇间拓扑中节点入度的不均衡,则削弱了其簇间非结构化拓扑在负载均衡方面的优势. HP2P 的簇内负载不平衡状况则归因于其簇间连接的构建维护机制.HP2P 只有在发现节点失效或连接错误时才更新簇间连接,这导致其容易忽视新加入的节点而倾向在在线时间更长的节点间建立簇间连接,从而使簇间路由更多地由这些节点承担,最终造成簇内负载的不平衡.

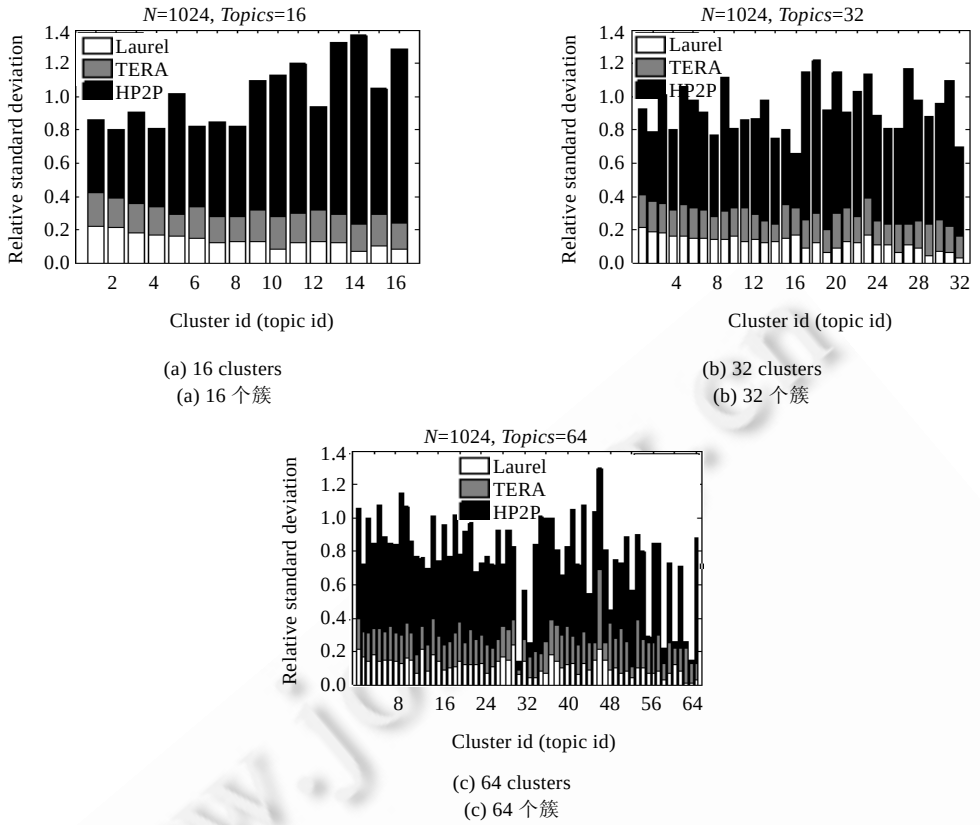


Fig.10 Load balance in the cluster

图 10 簇内负载平衡

综上,Laurel 具有优于 TERA,HP2P 及超级节点网络的簇内负载平衡性能.

5 总 结

本文提出了一种结合非结构化与结构化拓扑优点的混合式数据分发覆盖网——Laurel.通过基于分簇的结构化簇间连接,Laurel 能够保证较高的数据路由效率;通过多重的簇间连接、非结构化的簇内拓扑以及簇的动态创建与退出机制,Laurel 能够保证较强的系统鲁棒性;通过负载平衡机制,Laurel 能够保证簇间路由负载在簇内节点间均衡分配.

实验结果表明,相对于簇间采用结构化拓扑的层次式覆盖网,Laurel 在保持相同路由效率的情况下具有显著更强的鲁棒性;相对于基于双层非结构化拓扑的 TERA,Laurel 则保证在鲁棒性没有显著差距的情况下具有显著更高的路由效率;并且相对于已有方法,Laurel 也表现出更好的簇内负载平衡性能.因此,Laurel 的确实现了路由效率与鲁棒性的更好的折衷.

下一步值得研究的问题是,Laurel 中骨干节点与叶子节点角色的自适应转换及簇间负载平衡.这些问题的解决,将有助于 Laurel 根据系统负载的动态变化自动地调整拓扑结构及路由策略,从而进一步增强对动态变化的工作环境的适应能力.

References:

[1] Zhang M, Xu M, Wu J. Survey on application layer multicast. Acta Electronica Sinica, 2004,32(12A):22−25 (in Chinese with English abstract).

[2] Ma J, Huang T, Wang J, Xu G, Ye D. Underlying techniques for large-scale distributed computing oriented publish/subscribe system. Journal of Software, 2006,17(1):134−147 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/134.htm [doi: 10.1360/jos170134]

[3] National incident management system. 2011. http://www.fema.gov/emergency/nims/

[4] Network centric warfare. 2011. http://www.dodccrp.org/html4/research_ncw.html

[5] Eugster PT, Guerraoui R, Kermarrec AM, Massoulie L. Epidemic information dissemination in distributed systems. IEEE Computer, 2004,37(5):60−67. [doi: 10.1109/MC.2004.1297243]

[6] Gnutella. 2011. http://wiki.limewire.org/index.php?title=GDF

[7] Clarke I, Sandberg O, Wiley B, Hong T. Freenet: A distributed anonymous information storage and retrieval system. In: Federrath H , ed. Proc. of the Int'l Workshop on Design Privacy Enhancing Technologies. New York: Springer-Verlag, 2001. 46−66.

[8] Stoica I, Morris R, Liben-Nowell D, Karger DR, Kaashoek MF, Balakrishnan H. Chord: A scalable peer-to-peer lookup protocol for Internet applications. IEEE/ACM Trans. on Networking, 2003,11(1):17−32. [doi: 10.1109/TNET.2002.808407]

[9] Ratnasamy S, Francis P, Handley M, Karp R, Shenker S. A scalable content-addressable network. In: Cruz P, Varghese P, eds. Proc. of the 2001 Conf. on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM 2001). New York: ACM, 2001. 161−172. [doi: 10.1145/383059.383072]

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

[11] Zhao BY, Huang L, Stribling J, Rhea SC, Joseph AD, Kubiatowicz JD. Tapestry: A resilient global-scale overlay for service deployment. IEEE Journal on Selected Areas in Communications, 2004,22(1):41−53. [doi: 10.1109/JSAC.2003.818784]

[12] KaZaA. http://www.kazaa.com/

[13] eMule. http://www.emule.org/

[14] Qiao B, Wang G, Ding L. TBSN: A taxonomy hierarchy based P2P network. Journal of Computer Research and Development, 2008,45(5):803−909 (in Chinese with English abstract).

[15] Shin K, Lee S, Lim G, Yoon H, Ma JS. Grapes: Topology-Based hierarchical virtual network for peer-to-peer lookup services. In: Olariu S, ed. Proc. of the 2002 Int'l Conf. on Parallel Processing Workshops (ICPPW 2002). Washington: IEEE Computer Society, 2002. 159−164. [doi: 10.1109/ICPPW.2002.1039726]

[16] Hsiao R, Wang S. Jelly: A dynamic hierarchical P2P overlay network with load balance and locality. In: Tzeng NF, Takizawa M, eds. Proc. of the 24th Int'l Conf. on Distributed Computing Systems Workshops (ICDCSW 2004). Washington: IEEE Computer Society, 2004. 534−540. [doi: 10.1109/ICDCSW.2004.1284084]

[17] Chen Y, Deng B, Li X. Canicula: An improved hybrid overlay networks. In: Wong LWC, Lau LY, eds. Proc. of the 14th IEEE Int'l Conf. on Networks (ICON 2006). Washington: IEEE Computer Society, 2006. 1−6. [doi: 10.1109/ICON.2006.302672]

[18] Darlagiannis V, Mauthe A, Steinmetz R. Overlay design mechanisms for heterogeneous, large scale, dynamic P2P systems. Journal of Networks and System Management, 2004,12(3):371−395. [doi: 10.1023/B:JONS.0000043686.04679.03]

[19] Garcés-Erice L, Biersack EW, Felber PA, Ross KW, Urvoy-Keller G. Hierarchical peer-to-peer systems. In: Kosch H, Boszormenyi L, Hellwagner H, eds. Proc. of the 2003 Int'l Conf. on Parallel Processing (Euro-Par 2003). Berlin: Springer-Verlag, 2003. 1230−1239. [doi: 10.1142/S0129626403001574]

[20] Yang M, Yang Y. An efficient hybrid peer-to-peer system for distributed data sharing. In: Wu J, ed. Proc. of the 22nd IEEE Int'l Symp. on Parallel and Distributed Processing (IPDPS 2008). Washington: IEEE Computer Society, 2008. 1−10. [doi: 10.1109/IPDPS.2008.4536271]

[21] Peng Z, Duan Z, Qi J, Cao Y, Lv E. HP2P: A hybrid hierarchical P2P network. In: Dini P, ed. Proc. of the 1st Int'l Conf. on the Digital Society (ICDS 2007). Washington: IEEE Computer Society, 2007. http://ieeexplore.ieee.org/search/freesrchabstract.jsp?tp=&arnumber=4063779&queryText%3DHP2P:+A+Hybrid+Hierarchical+P2P+Network%26openedRefinements%3D*%26searchField%3DSearch+All [doi: 10.1109/ICDS.2007.20]

[22] Baldoni R, Beraldi R, Quema V, Querzoni L, Tucci-Piergiovanni S. TERA: Topic-Based event routing for peer-to-peer architectures. In: Jacobsen P, ed. Proc. of the 2007 Inaugural Int'l Conf. on Distributed Event-Based Systems (DEBS 2007). New York: ACM, 2007. 2–13. [doi: 10.1145/1266894.1266898]

[23] Voulgaris S, Gavidia D, van Steen M. CYCLON: Inexpensive membership management for unstructured P2P overlays. Journal of Network and Systems Management, 2005,13(2):197–217. [doi: 10.1007/s10922-005-4441-x]

[24] Gkantsidis C, Mihail M, Saberi A. Random walks in peer-to-peer networks. In: Li VOK, ed. Proc. of the 23rd Annual Joint Conf. of the IEEE Computer and Communications Societies (INFOCOM 2004). Washington: IEEE Computer Society, 2004. 120–130. [doi: 10.1109/INFCOM.2004.1354487]

[25] Deshpande M, Xing B, Lazardis I, Hore B, Venkatasubramanian N, Mehrotra S. CREW: A gossip-based flash-dissemination system. In: Raynal M, Ichikawa H, eds. Proc. of the 26th IEEE Int'l Conf. on Distributed Computing Systems (ICDCS 2006). Washington: IEEE Computer Society, 2006. http://ieeexplore.ieee.org/search/freesrchabstract.jsp?tp=&arnumber=1648832&queryText%3DCREW:+A+Gossip-based+Flash-Dissemination+System%26openedRefinements%3D*%26searchField%3DSearch+All [doi: 10.1109/ICDCS.2006.24]

[26] PeerSim. 2011. http://peersim.sourceforge.net/

附中文参考文献:

[1] 章淼,徐明伟,吴建平.应用层组播研究综述.电子学报,2004,32(12A):22–25.

[2] 马建刚,黄涛,汪锦岭,徐罡,叶丹.面向大规模分布式计算发布订阅系统核心技术.软件学报,2006,17(1):134–147. http://www.jos.org.cn/1000-9825/17/134.htm [doi: 10.1360/jos170134]

[14] 乔百友,王国仁,丁琳琳.TBSN:一种基于分类层次的 P2P 网络.计算机研究与发展,2008,45(5):803–809.

郑重(1982－),男,江苏扬州人,博士生,CCF 学生会员,主要研究领域为网络计算,数据分发技术.

王意洁(1971－),女,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为网络计算,数据库技术,移动计算.

马行空(1987－),男,硕士生,CCF 学生会员,主要研究领域为网络计算,数据分发技术.