一个通用最优的动态网络构建框架

袁 野[1),2)] 王国仁[1),2)] 郭得科[3)]

[1)](东北大学信息科学与工程学院 沈阳 110004)

[2)](医学影像计算教育部重点实验室(东北大学) 沈阳 110004)

[3)](国防科学技术大学信息与管理学院信息系统工程国家重点实验室 长沙 410073)

摘 要 覆盖网络的拓扑特性对 P2P 系统的性能至关重要.现有的覆盖网络大多基于静态互联网络,因为互联网络在静态环境下表现出良好的拓扑特性.Moore 下界给出这些静态网络的直径和结点度数的最佳折中理论值,但由于动态变化的网络,Moore 下界不适合现存 P2P 系统.为此,该文根据现有 P2P 系统的特点,给出在高度动态环境下新的网络直径和路由平均距离的下界.现有系统的路由性能不能超越此下界,因为它们不能很好地适应高度动态的网络——这一 P2P 系统最重要的特点.另外已被提出的覆盖网络都针对其相应静态结构有不同的维护机制,并没有统一的构建方法.为解决上述问题,该文提出了动态 Trie 树结构这一通用框架,任何静态互联网络都可以基于该框架构造出新的 P2P 系统,同时此通用框架又包含了一系列最优的设计策略.根据该构造方法,文章采用 deBruijn 和 Butterfly 图构建出两个新 P2P 系统,并且它们的性能可以超越文中给出的下界.经少许修改,构建 deBruijn 和 Butterfly 的方法也可应用到其它互联网络如 Hypercube、Kautz、Shuffle-exchange 和 CCC 等.

关键词 P2P;互联网络;下界;动态网络;路由

中图法分类号 TP393 **DOI 号**: 10.3724/SP.J.1016.2011.01536

A Universal and Optimal Dynamic Network Constructive Framework

YUAN Ye[1),2)] WANG Guo-Ren[1),2)] GUO De-Ke[3)]

[1)](*College of Information Science and Engineering, Northeastern University, Shenyang* 110004)

[2)](*Key Laboratory of Medical Image Computing (Northeastern University), Ministry of Education, Shenyang* 110004)

[3)](*National Laboratory for Information System Engineering, School of Information System and Management, National University of Defense Technology, Changsha* 410073)

Abstract The topological properties of overlay networks are critical for the performance of peer-to-peer (P2P) systems. Most existing overlay networks are based on static interconnect networks, since these networks behave good topological features in static environments. The Moore bound sets the optimal tradeoff between diameter and degree for these static networks, but the Moore bound cannot give a good description for existing P2P systems due to their dynamic features. In this study, therefore, we first prove new lower bounds of the network diameter and average query distance in a highly dynamic environment. The routing latency of the existing systems cannot be better than the new bounds. This is because the existing systems require a fixed number of peers known a priori, and P2P systems on the other hand, typically are dynamic, meaning that peers frequently join or leave. In addition, most proposed overlay networks have their unique maintenance mechanisms specific to the interconnect networks on which they are based. To solve the issues, we propose the dynamic multi-way Trie tree structure, a universal

收稿日期:2009-05-06;最终修改稿收到日期:2011-05-11.本课题得到国家自然科学基金-国家杰出青年科学基金(61025007)、国家自然科学基金重点项目(60933001)、国家自然科学基金面上项目(61073063)及中央高校基本科研业务费专项资金(N090404012)资助.
袁 野,男,1981 年生,博士,副教授,研究方向为图数据库、概率数据库、P2P 计算和数据隐私.E-mail: linuxyy@gmail.com. **王国仁**,男,1966 年生,教授,博士生导师,研究领域为 XML 数据管理技术、查询处理与优化、概率数据库和生物信息学. **郭得科**,男,1980 年生,博士,副教授,研究方向为无线多跳网络、对等计算、数据中心互联.

framework based on which any interconnect network can be adopted to construct a P2P system. The universal framework also consists of a series of optimal schemes for P2P systems. We show the power of the constructive scheme by applying it to deBruijn and Butterfly graphs to propose two new P2P systems whose performances can exceed the new bounds. Also, the schemes for deBruijn and Butterfly graphs can be easily applied to other interconnection networks after minimal modifications, such as, Hypercube, Kautz, Shuffle-exchange and CCC.

Keywords　P2P; interconnect network; lower bound; dynamic network; routing

1　引　言

随着 P2P 系统大量的涌现，它已经成为继 Internet 网之后信息系统的又一主要基础架构. 在 Napster 和 Gnutella 等[1]集中式和无结构 P2P 网络出现之后，为取得更好的可扩展性、查询准确性和高效的路由，大量基于分布式 Hash 表(DHT)的结构化网络相继出现[2-4].

两个最重要的参数决定了结构化 P2P 网络的性能和质量：(1)结点度数：结点路由表的大小. (2)网络直径：一个查询在网络里执行的最长路由跳数(hop). 早期的结构化网络如基于 Hypercube 的 Chord[3]和 Pastry[4]，它们的网络直径和结点度数都随着结点数目(n)的增加而以对数比例增长. 这些系统在发布和查询资源时需执行 $\log n$ 跳数，它们通常具有较大的维护代价和较差的扩展性. 为解决上述问题，一些基于常数度互联网络的 P2P 系统被提出，例如基于 Butterfly 的 Viceroy[5]和 Ulysses[6]；基于 CCC 的 Cycloid[7]；基于 d 维度 Torus 的 CAN[2]；基于 debruijn 的 D2B[8]、Koorde[9]和 ODRI[10]；基于 Kautz 的 FissionE[11]和 Moore[12]. 这些网络的直径在结点度数不变的情况下随网络规模以对数比例变化，表现出良好的可扩展性.

Moore 下界给出任何静态图的结点度数和直径的最佳折中理论值. 但由于 P2P 系统的动态特性，Moore 下界不能很好地给予 P2P 网络理论指引. deBruijn 和 Kautz 图是直径最接近 Moore 下界的互联网络结构，但基于它们的 P2P 系统，如 D2B、Koorde 和 FissionE 的直径远大于 Moore 下界. 其它一些系统如 Pastry、Viceroy 和 Cycloid 也远比它们相应的静态网络性能差得多. 主要原因在于现有系统不能很好地适应动态环境，这些网络通常只有在它们的直径和结点度数决定的网络规模下，才能表现出良好性能，但这对动态的 P2P 系统是不实际的. 因此本文根据 P2P 网络动态的特性，为现有系统计算出新的网络直径和查询平均距离的下界. Kademlia[13]系统使用 XOR 技术来解决动态问题，但它只在 Pastry 系统上实现了该方法，并且系统具有较高的维护代价. Zhang 等人[14]用 Distributed Line Graphs 技术来解决任何常数度拓扑的动态构造问题，但该方法实际只适用于 Kautz 和 deBruijn，并不适用于 Butterfly、CCC 和 Hypercube 等结构. 为了支持任何互联结构的动态构造，本文提出了一通用构建框架-基于动态 Trie 树的构造模型，基于该框架，任何静态拓扑都可以构造出 P2P 系统，并且这些系统的直径和平均距离可以超越本文给出的下界. 同时该构造方法也包含了一系列最优设计来优化系统的性能.

2　相关工作

Moore 下界是任何图直径的下界，它的理论值由文献[15]给出：$d_m \geq \lceil \log_k(n(k-1)+1) \rceil - 1$. 另一个衡量网络路由效率的参数是图的平均距离，任何 k-正则图的平均距离下界是[16]：

$$d_{avg} \geq d_m - \frac{k(k^{d_m}-1)}{n(k-1)^2} + \frac{d_m}{n(k-1)} \approx d_m - \frac{1}{k-1}.$$

一个度数和直径为 d 和 k 的 deBruijn 图[8]含有 d^k 个结点，其中边的连接规则是从结点 $u=x_1x_2\cdots x_k$ 到结点 $v=x_2\cdots x_kx_{k+1}$ 连接一条有向边. 包含 kd^k 个结点的 Butterfly 结构也是有向图，其中它的直径和结点度数分别是 $2k-1$ 和 d. 当 $i\neq k-1$ 时，有向边从结点 $(x_0x_1\cdots x_{k-1};i)$ 连接到标识为 $(x_0\cdots x_i, y, x_{i+2}\cdots x_{k-1}; i+1)$ 的结点，否则连接到结点 $(y, x_1\cdots x_{k-1}; 0)$. 图 1 和图 2 分别给出 deBruijn 和 Butterfly 结构示例.

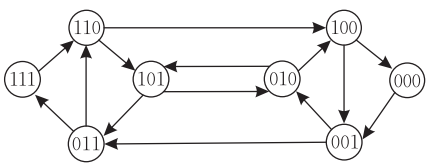


图 1　结点度数为 2 直径为 3 的 deBruijn 图

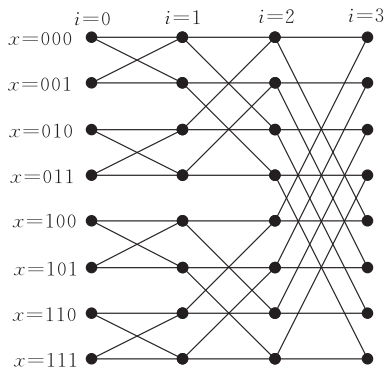


图 2 结点度数为 2 直径为 5 的 Butterfly 图

基于 deBruijn 图的系统如 D2B[8] 和 Koorde[9]，基于 Butterfly 的系统如 Viceroy[5] 和 Ulysses[6] 都给出解决动态网络问题的方法，但同时也带来了网络拓扑变形、维护代价高昂和结点度数分布倾斜等问题，这些变化都会严重影响网络的性能. 本文将给出解决动态环境问题的通用方法，同时将额外的代价降到最低，以保证系统高效的运行.

3 动态网络路由延迟下界

众所周知，Moore 下界给出任何图的直径下界，但它只适合描述静态拓扑. 对于动态网络，它的值太小，不适合描述 P2P 网络. 因此本节给出在动态环境下，现有 P2P 网络查询延迟的下界.

为计算新下界，考虑一颗 d 叉 Trie 树，它也可被看作路由树. 对于静态网络，从树根开始在 h 跳内，最多可达 $n-1$ 个结点，因此满足 $d+d^2+\cdots+d^h\geq n-1$，从此式便可推出 Moore 下界. 但在动态网络里，第 i 跳($1\leq i\leq h$)最多可达结点不一定是 d^i 了，也可看成路由树的每层不再是满叉的. 为了模拟动态网络，考虑一个标识符前缀为 x 的结点随机地加入和离开 Trie 树. 加入时从根结点开始，从 d 个孩子中找到一个与 x 向前最匹配的结点，并路由到该结点；接下去每一层都按照此规则路由，直到该结点路由到一个叶子结点，此叶子结点和 x 具有最长的公共前缀；最后该结点成为此叶子结点的孩子，加入到 Trie 树中. 随机离开的过程和加入的过程恰好相反. 这个动态过程是目前 DHT 的工作方式，因此可以用来模拟动态网络.

对网络直径的下界，我们有如下定理.

定理 1. 在概率 $1-o(1)$ 的保证下，动态网络的直径下界($d_{\max}$)由如下等式给出

$$d_{\max}=\lfloor \log_d n+\sqrt{d\log_d n}-1.5\rfloor \quad (1)$$

证明. $d_{\max}$ 可看作上述 Trie 树的最大高度，因此只需计算此高度. 为此，需考虑结点离开的过程. 这个过程等同于从最大高度开始一层层地砍掉叶子结点，这个过程与 Patricia Trie 树结构的构造十分类似，因为 Patricia Trie 树可看成是删除所有普通 Trie 树中只有一个孩子的中间结点后得到的结构. 从文献[17]可以得到在概率 $1-o(1)$ 的保证下，随机 Patricia Trie 树的高度收敛于 $\lfloor \log_d n+\sqrt{d\log_d n}-1.5\rfloor$. 因此得到式(1)给出的下界. 证毕.

网络直径只是查询在最坏情况下的路由跳数，而查询平均距离更能衡量一个网络的路由性能. 因此下面将给出动态网络的平均查询距离的下界.

设 h 表示 d 叉 Trie 树的最小深度. Trie 树由两部分构成，一部分是从根结点到 h 层，另一部分是由 $h+1$ 层到 $d_{\max}$ 层的结点组成. 很明显第 1 部分是一颗叉数为 d 的满叉树. 同时设 d_x 表示从根节结点到树任意结点的距离，接下来讨论 d_x 对树两部分的概率分布情况.

为计算第 1 部分的分布情况，我们定义一序列随机变量 A_i，$i\geq 0$. 其中当树的第 i 层是满的，有 $A_i=1$，否则 $A_i=0$. 一层是满的表示该层的结点都存在，并且不是叶子结点. 注意到对 $i\leq h$，如果有 $A_i=1$，那么对所有 $k<i$，可推出 $A_k=1$. 因此当 d_x 至少是 $k+1$ 时，所有的 $0\sim k$ 层都需是满的，并且满足

$$P(d_x\geq k+1)=P(\bigcap[A_i=1]) \quad (2)$$

从式(2)可推出：

$$P(d_x\geq k+1)=P(d_x\geq k)P(A_k|A_{k-1}) \quad (3)$$

其中 $P(d_x\geq 0)=1$ 并且 $P(A_k|A_{k-1})$ 是条件概率，表示如果第 k 层满，那么前 $0,\cdots,k-1$ 层都得是满的，因此有

$$P(A_k|A_{k-1})=P(A_k=1|h\geq d_x) \quad (4)$$

对于树的第 1 部分，有如下定理.

引理 1. 在概率 $1-o(1)$ 的保证下，d_x 的分布是 $\exp\left\{-d^k\exp\left\{\frac{1}{d-1}-\frac{n(d-1)+1}{d^{k+1}(d-1)}\right\}\right\}$.

证明. 首先树的第 1 部分是 h 层的满叉树，并且该满叉树包括 $\frac{n(d-1)+1}{d}$ 个叶子结点和 $\frac{n(d-1)+1-d}{d(d-1)}$ 个非叶子结点. 其次假设对于 $0\sim k-1$ 层都是满的，那么就有 $\frac{d^k-1}{d-1}$ 个结点已经加入到 $0,\cdots,k-1$ 层，并且还剩余 $\frac{n(d-1)+1-d}{d(d-1)}-\frac{d^k-1}{d-1}=\frac{n(d-1)+1-d^{k+1}}{d(d-1)}$ 个位置可以让新结点加

入.在前 $k-1$ 层已经被加满后，第 k 层每个位置被新结点加入的概率是 d^{-k}.这样我们的问题就等同于把 $u=\frac{n(d-1)+1-d^{k+1}}{d(d-1)}$ 个球随机并且均匀地放到 $m=d^k$ 个盒子里，要求每个盒子至少一个球的概率是多少.这个问题有很多种解法，其中文献[18]把它看做是经典的优惠券收集问题，下面将用到其中的结果.定义 $Z(u)$ 是在 u 个球被放到 m 个盒子后剩下的非空盒子个数，因此有 $P(A_k \mid A_{k-1})=P(Z(u)=m)$.由文献[18]知，

$$P(Z(u)=m)=\sum(-1_j)(m/j)\left(1-\frac{j}{m}\right)^u \quad (5)$$

因 u 很大，所以 $(1-j/m)^u$ 近似等于 $e^{-uj/m}$，因此式(5)可写成

$$P(Z(u)=m)\approx\sum(-1_j)(m/j)e^{-uj/m}=1-e^{-u/m} \quad (6)$$

对于 P2P 系统，我们对较大的 $m=o(\log_d n)$ 更感兴趣，因此上式又可写成

$$P(Z(u)=m)\approx e^{-me^{-u/m}}=\exp\left\{-d^k\exp\left\{\frac{1}{d-1}-\frac{n(d-1)+1}{d^{k+1}(d-1)}\right\}\right\} \quad (7)$$

因此从式(3)～式(7)得

$$P(d_x\geq k+1)=P(d_x\geq k)P(A_k\mid A_{k-1})\approx\exp\left\{-d^k\exp\left\{\frac{1}{d-1}-\frac{n(d-1)+1}{d^{k+1}(d-1)}\right\}\right\}.$$ 证毕.

从引理 1 知，在概率 $1-n^{-\varepsilon}(\varepsilon\leq 1)$ 的保证下有

$$h=\max(d_x)=\log_d n-\log_d((1+\varepsilon)\log n-o(\log\log n)) \quad (8)$$

对于 d_x 到树第 2 部分的分布可以从文献[17]得到，因此有如下引理.

引理 2.　在概率 $1-o(1)$ 的保证下，d_x 对 PATRICIA Trie 树的分布是

$$d_x\sim\sqrt{1+d\varepsilon\Phi'(\varepsilon)+\varepsilon^d\Phi''(\varepsilon)}e^{-n\Phi'(\varepsilon)} \quad (9)$$

其中 $\varepsilon=nd^{-k}(0<\varepsilon<1)$，并且

$$\Phi(\varepsilon)\sim\frac{1}{d}\rho_0 e^{\varphi(\log_d\varepsilon)}\varepsilon^{3/2}\exp\left(-\frac{\log^d\varepsilon}{d\log d}\right).$$

从 d_x 到树两部分的分布，有如下定理.

定理 2.　在概率 $1-n^{-\varepsilon}(\varepsilon\leq 1)$ 的保证下，动态网络查询平均距离的下界是 $\log_d n+\sqrt{\log_d n/d}$.

证明.　由上述结论可得，

$$E(d_x)=\sum_{k=0}^{h}k\cdot P_1(d_x)+\sum_{k=h+1}^{d_{\max}}k\cdot P_2(d_x)=\sum_{k=0}^{h}k\cdot\exp\left\{-d^k\exp\left\{\frac{1}{d-1}-\frac{n(d-1)+1}{d^{k+1}(d-1)}\right\}\right\}+\sum_{k=h+1}^{d_{\max}}k\cdot\sqrt{1+d\varepsilon\Phi'(\varepsilon)+\varepsilon^d\Phi''(\varepsilon)}e^{-n\Phi'(\varepsilon)}\approx\log_d n+\sqrt{\frac{\log_d n}{d}}.$$ 证毕.

4　动态网络通用构建框架

任何互联网络具有良好的拓扑特性的条件是当且仅当它的所有结点都存在并且稳定，例如 deBruijn 和 Butterfly 结构要具有优良的拓扑特性需有 d^k 和 $d^k k$ 个结点.但这种要求对以互联网络为基础的动态 P2P 系统是不现实的，为此本文提出了“动态多叉 Trie 树”来构建 P2P 系统，使其有良好的拓扑特性.

4.1　动态多叉 Trie 树

定义 1.　动态 d 叉 Trie 树结构是一颗深度为 k 的树，每个结点至多有 d 个孩子结点.每个结点和其出边都被赋予唯一的标识符，结点的标识符是从根结点到其自身边标识符的叠加，并且边和结点的标识符满足如下规则：(1) 根结点的出边被标识为 $x_1^i=i(0\leq i\leq d-1)$，它的第 i 个孩子结点也被赋予标识符 $x_1^i=i$，并且孩子结点从左至右地排列，根结点不包括任何标识符.(2) 结点 x_1 的出边标识为 $x_2^i=i(0\leq i\leq d-1)$，它的第 i 个孩子结点被标识为 $x_1x_2^i$，并且从左至右排序.(3) 结点 $x_1x_2\cdots x_{k-1}$ 的出边标识为 $x_k^i=i(0\leq i\leq d-1)$，孩子结点标识为 $x_1x_2\cdots x_{k-1}x_k$，并且从左至右排序.(4) 每个结点同时也指向其父亲，并且同层的结点构成一个环结构.

在动态环境下，Trie 树可能是不平衡的.平衡 Tire 树被定义为所有的叶子结点都在同一层，当叶子结点都存在时，树被称为完全 Trie 树.图 3 给出一颗高度为 3 的完全 Trie 树，当结点 011 失效，树变成不完全的，但是平衡的.如果结点 010 和 011 都失效，Tire 树变成不平衡树.

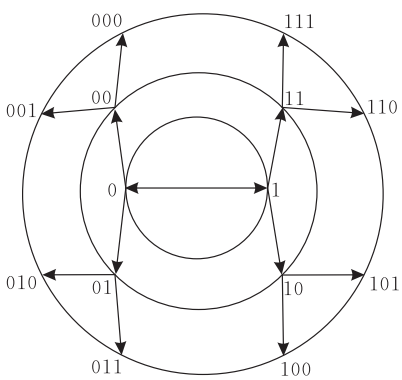


图 3　二叉 Trie 树结构

从 Tire 树定义可知，同层结点构成环结构，此环称之为 Trie 环.因此每个结点需连接其前驱和后

继结点以构成环.

定义 2. 动态 Trie 树一结点的前驱是以顺时针方向沿 Trie 环第一个存在的结点,同理它的后继结点是以逆时针方向沿 Trie 环第一个存活的结点.

从定义 2 看出,一个结点的前驱和后继可能不是在标识符空间中的邻居,因为动态网络不能保证所有邻居结点都存活.下面给出结点 $x_1x_2\cdots x_k$ 确定其前驱的算法.

算法 1. Predecessor($x_1x_2\cdots x_k$).

1. 结点 $x_1x_2\cdots x_k$ 沿向上路径发送"确定前驱"消息,该消息直到发送到 $x_1x_2\cdots x_k$,某个祖先 $x_1x_2\cdots x_i$ 已经有前驱并且该祖先不是其父的第 1 个孩子为止.

2. 结点 $x_1x_2\cdots x_i$ 将消息转发给其前驱结点,接着该消息一直沿着最后一个向下链接被转发,直到结点 $x_1x_2\cdots x_k$ 所在的层,并把该结点作为 $x_1x_2\cdots x_k$ 的前驱.

该算法的效率由如下定理给出.

定理 3. 在概率 $1-o(1)$ 的保证下,算法 1 的代价最多是 $\sqrt{d\log_d n}+0.5+\log_d((1+\varepsilon)\log n-o(\log\log n))$.

算法 1 给出确定结点前驱的步骤,同理也可以确定结点的后继 Successor(x).

4.2 静态网络到 Trie 树的映射

本节给出最优 P2P 网络的构建方法,使网络在动态环境下保持静态互联网络的拓扑特性.

4.2.1 最优拓扑构建规则

欲保持静态互联网络的拓扑特性,主要需解决 P2P 网络动态变化问题.为此本节给出基于 Trie 树的方法可使任何类型的互联网络结构都能被构建成高效的覆盖网络.

对任意 d-ary 互联网络的一个结点 $x=x_1x_2\cdots x_k$,$\sigma^i(x)(1\leq i\leq d)$,表示结点 x 的一个邻居,则基于动态 d 叉 Trie 树的构建规则如下:

(1) 如果结点 $\sigma^i(x)$ 存在于覆盖网络,那么它是结点 x 的第 i 个邻居.

(2) 否则,如果结点 $\sigma^i(x)$ 和它 Trie 树中前驱结点 Predecessor(x) 的标识符有共同长度为 $k-1$ 的前缀,那么结点 Predecessor(x) 是结点 x 的第 i 个邻居,并且 Predecessor(x) 保留标识符 $\sigma^i(x)$ 以代替结点 $\sigma^i(x)$.

(3) 否则,如果结点 $\sigma^i(x)$ 和它 Trie 树中后继结点 Successor(x) 的标识符有共同长度为 $k-1$ 的前缀,那么 Successor(x) 是结点 x 的第 i 个邻居,并且 Successor(x) 保留标识符 $\sigma^i(x)$ 以代替结点 $\sigma^i(x)$.

(4) 否则最年轻存活的 $\sigma^i(x)$ 祖先结点作为结点 x 的第 i 个邻居,并且该祖先保留标识符 $\sigma^i(x)$.

该构建规则可保证互联网络结点在动态环境下一定能按其规则连接到它的邻居 $\sigma^i(x)$,并且保证在 $\sigma^i(x)$ 不存在时,替代结点和 $\sigma^i(x)$ 的标识符具有最大匹配,以保证路由算法以最小代价完成动态路由(路由算法将在后面介绍).例如将 deBruijn 图嵌入到图 3 所示的树中,按规则结点 010 连接 100 和 101,如果结点 100 失效,010 将连接到 101.如果结点 100 和 101 都失效,其父结点 10 替代它们作为 010 的邻居.从该构建规则很容易得到如下定理.

定理 4. 设一个基于上述构建规则的 P2P 系统的中间结点和叶子结点分别为 x 和 y,可得如下结论:

(1) 当 Trie 树平衡时,结点 x 和 y 分别有 $2d+3$ 和 $d+3$ 个邻居.

(2) 当 Trie 树不平衡时,结点 x 和 y 分别至少有 $2d+1$ 和 $d+1$ 个邻居.

在设计实际系统时,通常选取较大的参数 d 来保证系统有高效的路由和较好的连通性,但同时也增加了系统维护代价,反之亦然.因此参数 d 对系统性能有很大影响,选取一个最佳的参数值是一个折中问题.

P2P 系统执行的操作主要包括查询和更新,因此可以基于它们的操作代价计算出系统总的代价.设查询操作所占比例为 $\lambda(0\leq\lambda\leq1)$,更新操作比例为 $1-\lambda$,则总代价为 $C=\lambda A+(1-\lambda)B$,其中 A 为查询代价,B 为更新代价.它们的值可分别近似为 $\log_d n$ 和 d,因此有总代价,

$$C(d)=\lambda\log_d n+(1-\lambda)d,$$

对该式求导得

$$(\ln d)^2 d=\frac{\lambda\ln n}{1-\lambda} \tag{10}$$

很容易利用牛顿迭代计算方法求得满足方程(10)中 d 的数值解.

4.2.2 最优资源放置策略

资源放置策略对 P2P 系统至关重要,因为它直接影响动态路由的效率和正确性.现有的策略在网络高度震荡时并不能同时满足上述两个特性.本节将给出基于 Trie 树的资源放置策略,它能保证在大量结点失效时,接管其资源的结点标识符和该资源有最大匹配,以保证路由的正确和高效.

设一资源为 $x=x_1x_2\cdots x_k\cdots$,它是根据某种 Hash 算法得到的,其标识符通常比它所在结点标识符要长.则放置规则如下:

(1) 如果结点 $x'=x_1x_2\cdots x_k$ 存活于 P2P 网络,则结点 x' 存储资源 x.

(2) 否则，如果结点 x' 和其前驱 Predecessor(x) 在 Trie 树上有共同的父亲，则结点 Predecessor(x) 存储资源 x.

(3) 否则，如果结点 x' 和其后继 Successor(x) 在 Trie 树上有共同的父亲，则结点 Successor(x) 存储资源 x.

(4) 否则最年轻的 x' 存活祖先结点存储资源 x.

根据构造和放置策略可看出，系统结点和动态 Trie 树结点是一一对应的，所有资源都放置在树叶子结点上，中间结点只作为路由结点. 此策略和以往的方法不同(如文献[19])，它们都是把虚拟树结点以组为单位或以树枝为一个整体将虚拟树影射成覆盖网络. 这样会使结点度数增大一个对数数量级，从而破坏了互联网络的基本特性，例如 deBruijn 结点的常数度，而本文的方法最大限度地保持了原有结构的特征. 另外叶子的数量几乎是中间结点数目的 $(d-1)$ 倍，因此叶子结点足够容纳系统发布的资源. 如图 3 所示，资源 100… 存储在结点 100，如果 100 失效，资源 100… 存储在结点 101；如果结点 100 和 101 都失效，资源发布到它们父亲结点 10 上.

传统 P2P 系统都是把所有资源发布到网络中，这样会带来大量维护代价，但减少发布的资源会使某些查询失效，为此本文接下来给出一个最优的折中方案.

设 G 表示一结点 p_i 存储的所有资源的数量，设 a_j 表示第 j 个资源所占空间大小(以字节为单位). 这里假设对资源 j 的成功查询概率 q_j 已知，并有 $q_1+q_2+\cdots q_G=1$. 设 y_j 是布尔变量，当结点 p_i 存储资源 j 时，$y_j=1$，否则 $y_j=0$. 因此结点 p_i 被成功查询的概率为

$$P_{\text{query}}=\sum_{j=1}^{G}q_jy_j.$$

P2P 系统对资源的维护代价主要包括数据更新代价和检查代价，检查代价主要查看资源是否和本地数据一致，并且查看资源是否由于网络的变化而丢失. 假设检查消息平均每 T 秒发送到结点 p_i，则检查代价可以近似为

$$C_{\text{refresh}}=\frac{1}{T}\log_dn\sum_{j=1}^{G}y_j,$$

假设每秒有 f 个资源插入到结点 P_i(或删除). 此更新操作大约消耗 $C_{\text{refresh}}=f\log_dn$ 代价. 综上系统总的代价为

$$C_{\text{refresh}}=\frac{1}{T}\log_dn\sum_{j=1}^{G}y_j+f\log_dn \tag{11}$$

另外结点所存资源的量需满足其存储能力，变量 y_j 还需满足

$$\sum_{j=1}^{G}a_jy_j\leq S \tag{12}$$

其中 S 是结点 P_i 的最大存储容量. 对结点 P_i 应该最大化其成功查询概率，最小化网络维护代价，因此最优资源存储策略是多目标优化问题. 根据实际 P2P 系统的特点，可以建立如下一个目标函数来求解，

$$C_{\text{virual}}=\lambda P_{\text{query}}\log_dn-(1-\lambda)C_{\text{total}} \tag{13}$$

其中 λ 表示查询操作所占比例，$(1-\lambda)$ 是维护操作比例. 可通过求式(13)的最大值来确定最优策略，由式(11)、式(13)得

$$\begin{aligned}C_{\text{virual}}&=\lambda\log_dn\sum_{j=1}^{G}q_jy_j-(1-\lambda)\cdot\\&\left(\frac{1}{T}\log_dn\sum_{j=1}^{G}y_j+f\log_dn\right)=\\&\log_dn\left(\sum_{j=1}^{G}y_j\left(\lambda q_j-\frac{1-\lambda}{T}\right)-(1-\lambda)f\right)\end{aligned} \tag{14}$$

本文用动态规划来求解式(14)的最优值，结合上述分析可得规划函数，

$$\begin{aligned}\underset{1\leq j\leq G,0<i\leq S}{c}(j,i)=\max\Big\{&C(j-1,i-a_j)+\\&\lambda q_j-\frac{1-\lambda}{T},C(j-1,i)\Big\},\end{aligned}$$

我们设计一个递归程序来计算规划函数，在程序执行过程中，把变量 y_j 的取值记录下来，从而得出最优资源存储策略. 系统中任何结点都可以在本地完成该最优化求解，不需增加网络代价.

4.3　特例研究

前面已给出基于互联网络的通用 P2P 系统构建方法，本节将以 deBruijn 和 Butterfly 结构为例基于上述方法构造出 DPhoenix 和 BPhoenix 两个新 P2P 系统，因为 deBruijn 代表了单协议的互联网络如 Kautz、Hypercube 和 Shuffle exchange 等，Butterfly 可用来代表混合互联网络结构如 CCC 等.

4.2 节给出的网络拓扑构造规则是整个框架的核心，该方法可直接应用于 deBruijn 图. 对 Butterfly 结构(相关工作介绍了 Butterfly 结构)，系统包含若干棵 Trie 树，第 i 棵树的结点 $x=(x_0x_1\cdots x_{k-1};i)$ 把第 $i+1$ 棵树上结点 $\sigma^i(x)=(x_0x_1\cdots x_i,y,x_{i+2}\cdots x_{k-1};i+1)$ 作为其邻居. 由于 P2P 系统结点数目的动态性，第 $i+1$ 棵树可能为空，此时结点 x 作为该树的树根.

最优参数 d 的选取对 DPhoenix 和 BPhoenix 的拓扑性能很重要，此方法可以不经任何修改地应用到 deBruijn 和 Butterfly 结构上. 同样，最优资源放置策略也可被 DPhoenix 和 BPhoenix 系统直接采纳. 通过两系统的构造可看出本文的方法具有通

用性，构造 deBruijn 和 Butterfly 的方法可应用到其它互联网络结构.

5 系统的动态操作

P2P 系统是高度动态的网络，结点频繁地加入、离开和失败. 这些动作对网络拓扑有很大影响，从而影响路由. 因此对这些动作需健壮的处理方法，本节将给出基于动态 Trie 树系统的动态操作处理算法.

5.1 结点加入

欲保证路由算法的正确性，结点必须加入到正确的位置，及时更新路由表入口项. 同时也需以较低代价降低 Trie 树的倾斜度以平衡系统的负载. 为满足上述需求，本文给出算法 3.

算法 2. $Join(Peer x=x_0x_1\cdots x_k)$.

1. 加入结点首先通过 Hash 函数获得标识符 x，之后路由到其标识符所在的位置. 如果该位置有结点 x'(没有设其父结点为 x')查看其邻居 $\sigma^i(x')$是否至少包含两个标识符，则结点 x 路由到位置 $\sigma^i(x')$. 重复此过程($x'=\sigma^i(x')$)直到结点 x'的邻居$\sigma^i(x')$只包含一个标识符，那么结点分享 x'一个标识符并最后加入到该标识符所在位置.

2. 结点按拓扑构造规则连接其邻居，其中它互联网络的邻居可以根据其父结点的路由表快速定位.

新加入结点总是能选择 Trie 树低层空缺位置加入，因为构造规则使存活结点接管失败者时保存它们的标识符代替它们，这些多余的标识符揭示出系统的空缺，也标识出树的不平衡信息，加入结点一直沿这些连接找到最低层的空缺位置加入，从而降低树的不平衡性. 例如在图 3 中(假设结点 100 和 101 失败，结点 010 连接到 10)，新结点 0101 欲成为 010 的孩子，发现它的邻居 10 存储了其它的标识符，则它会选择 100 或 101 的位置加入替代它们.

算法步 2 根据其父结点快速定位互联网络邻居是有如下定理保证的.

定理 5. 对于结点 x 和它的 d 个互联网络邻居 $\sigma^i(x)$，结点 x 后代的互联网络邻居也是结点 $\sigma^i(x)$的后代.

虽然加入过程是一个迭代过程，但代价也是较低的并由定理 5 给出.

定理 6. 在概率 $1-o(1)$的保证下，结点加入算法消耗的代价至多是

$$\lfloor\sqrt{d\log_d n}-1.5+\log_d((1+\varepsilon)\log n-o(\log\log n))\rfloor+\log_d n.$$

5.2 结点离开

高效快速处理结点离开的方法可保证网络拓扑持久的连通性. 如果叶子结点离开系统，它可以随时离开，其保存的资源根据资源放置规则重新分配给其邻居，这个过程并不影响系统的拓扑形状. 但树的中间结点欲离开，此动作会给拓扑带来较大改变，算法 4 给出高效处理方法. 对于有向互联网络，设$(\sigma^i(x))^{-1}$表示结点 x 的入边，对无向网络有$(\sigma^i(x))^{-1}=\sigma^i(x)$.

算法 3. Departure(Peer x).

1. 结点 x 发送“替代”消息给它的一个后代叶子结点. 如果该后代结点 u 至少包含两个标识符，则 u 将消息转发给其邻居$(\sigma^i(u))^{-1}$，重复此过程($u=(\sigma^i(u))^{-1}$)直到结点 u 只包含一个标识符. 最后结点 u 被用来代替结点 x.

2. 更新结点 x Trie 树上邻居，并且根据结点 x 父亲路由表更新 x 的 d 个$(\sigma^i(x))^{-1}$邻居.

步 1 也像加入算法一样降低树的不平衡，“替代”消息会选择一个高层叶子结点来接替低层离开的中间结点. 同时在步 2 中，也可以利用定理 4 给出的性质快速更新互联网络邻居. 下面定理给出此算法的代价.

定理 7. 在概率 $1-o(1)$的保证下，结点 x 离开的代价至多是

$$\lfloor\log_d n+\sqrt{d\log_d n}-1.5\rfloor-Length(x).$$

P2P 网络经常也有结点失败，本文也按传统的方法让结点周期地查看其邻居是否失效，一旦发现就按离开结点处理. 这里不再单独给出结点失败的处理算法.

5.3 负载平衡

从前面介绍的方法可看出用基于 Trie 树构建的 P2P 系统有很多优良特性，但动态的网络会造成树的不平衡从而影响系统性能. 为此本节给出动态 Tire 树的负载平衡算法，本文以 DPhoenix 系统为例展示负载平衡算法.

如图 4 所示，根据构造规则子树 e 的所有结点都会找到替代它们不存在邻居的结点，由定理 4 知，这些结点都会连接到结点 c，当结点 c 过载时，它激发负载平衡算法. 如同结点加入和离开算法所述，指向 c 的连接揭示了不平衡信息，结点 c 就是利用这

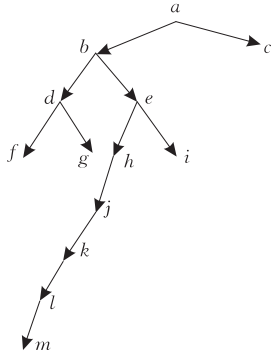


图 4 不平衡树，结点 c 过载

个信息来执行负载平衡算法，此时 c 存储了子树 e 所有信息. 算法包含两步，第一步采用类似 AVL 树旋转的方法处理不平衡. 首先结点 c 对子树 e 的所有结点按中序遍历排序得 m,l,k,j,h,e,i. 之后 c 开始按此顺序旋转子树，结点 m 替代 l；l 替代 k；k 替代 j 等等，最后结点 e 变成 i 的一个孩子，如图 5(a)所示. 用同样的中序遍历对该结构旋转得到如图 5(b)所示结果. 最后子树变成平衡树，图 5(c)给出得到的平衡树.

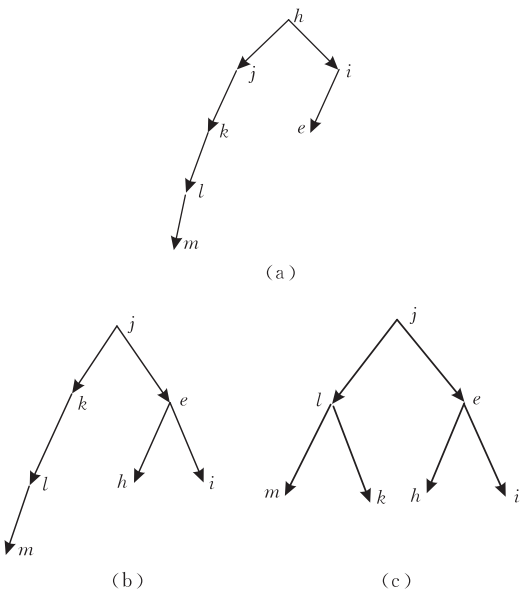


图 5　平衡子树 e 的过程

步 2 是自底向上地把平衡子树 j 的结点移给子树 c，做为其后代结点，直到两颗树的叶子结点在整个 Trie 树的同一层为止. 这个过程是结点 m,k 成为 c 的孩子；结点 h,i 成为 m 的孩子，最后的结果由图 6 给出.

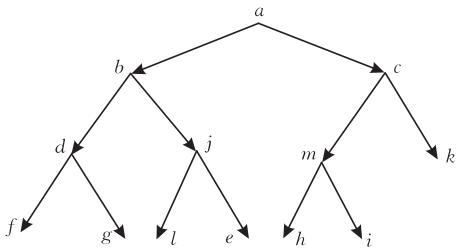


图 6　执行负载平衡算法后的结果

在算法执行过程中，数据移动代价较低，因为采用的旋转方法充分利用了系统拓扑的特性. 对参与的结点需要花费 $o(\log n)$ 代价去路由和更新路由表，但此代价是可以接受的. 由算法的步骤可看出，执行过程中不需要全局信息，也不需要整个系统都参与，只是某个结点过载时，它自己发起算法并在整个过程中控制所有参与结点，同时该算法并不很大程度地改变系统拓扑. 以上这些特性对高度动态环境下的树结构是很必要的.

5.4　路由算法

P2P 系统里的消息包括 3 类：(1)发布或查找资源的消息. (2)维护网络拓扑的消息. (3)两结点之间的路由消息. 为了正确并高效地路由上述 3 种消息，Trie 树的每个结点需按本文给出的拓扑规则保持连接. 但由于 P2P 网络的动态性和网络维护的延迟，拓扑不能始终按规则保持连接，这就要求路由算法具有容错特性.

为满足上述需求，结合拓扑构造和资源放置的特点，本文设计了算法 5 用来支持高效容错的路由算法. 其中 x 表示请求结点或当前结点，y 是终点或资源标识符.

算法 4.　Route(x, y).

case 1: Length(x)=Length(y) or (y is a resource and x is not a prefix y)
if ($x=y$) then return available.
else if Comprefix(x, Successor(x))=$k-1$ and Successor(x) is less than y in the ring, then Forward the message to peer Successor(x).
else if Comprefix(x, Predecessor(x))=$k-1$ and Predecessor(x) is larger than y in the ring, then Forward the message to peer Predecessor(x).
else Peer x forwards message to its neighbor $\sigma^i(x)$ which has the largest value Comprefix($\sigma^i(x)$, y) among all the neighbors.

case 2: Length(x)<Length(y) or (y is a resource and x is not a prefix y)
if (x contains y) then return available.
else if (Peer x has at least one child), then Forward the message to its child z which has the largest value of Comprefix(z,y).
else Peer x forwards message to its neighbor $\sigma^i(x)$ which has the largest value Comprefix($\sigma^i(x)$, y) among all the neighbors.

case 3: Length(x)>Length(y)
if (x has a link to its parent) then Peer x forwards message to its parent.
else Peer x forwards message to its neighbor $\sigma^i(x)$ which has the largest value Comprefix($\sigma^i(x)$, y) among all the neighbors.

对资源的查询该算法主要包括两步，首先消息从请求结点路由到同一层结点，该结点的标识符是资源 y 的前缀，其次查询消息沿向下路径路由到叶子结点去查找资源 y. 例如对 DPhoenix 系统，如图 3 所示，结点 00 欲查获资源 111…，查询被沿路径 00-01-11 转发，接着路由到叶子结点 111. 算法支持

的资源查询和任意结点间的路由操作都有一共同特点:消息可以像在静态互联网络里路由一样,不必考虑网络拓扑的变化.因为拓扑构造方法保证一定会有结点替代失效或离开者并保存其标识符,同时资源放置策略也是按构造方法的规则处理资源的.这样就保证了路由的正确性,同时前面的一系列最优策略保证了路由的高效性,定理 7 给出了量化的路由效率.

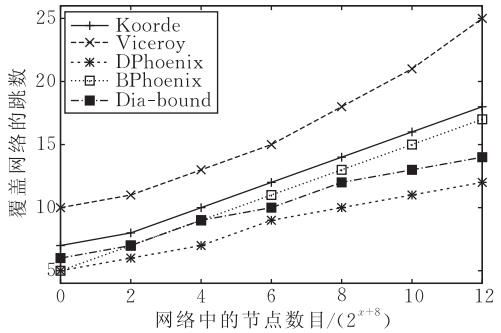
定理 8. 动态环境下,以概率 $1-o(1)$ 为保证,系统 DPhoenix 和 BPhoenix 的最坏路由延迟(网络直径)分别是 $\lfloor \log_d n \rfloor$ 和 $2\lfloor \log_d n \rfloor - 1$.

6 性能分析

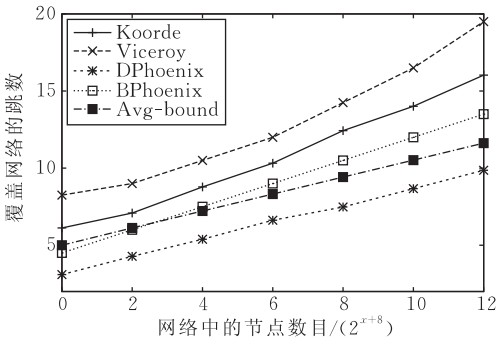
文本给出的所有算法都在我们开发的 P2P 模拟器上实现了,该模拟器和算法代码都用 C++编写,实验的运行环境是奔腾 IV 3.0 GHz CPU,2 GB 内存.我们在模拟器上实现了 DPhoenix 和 BPhoenix 系统,作为比较,基于 deBruijn 的 Koorde 系统和基于 Butterfly 的 Viceroy 系统也被在模拟器上实现.为公平比较,对 d-ary 的 deBruijn 和 Butterfly 的 d 值取 4,这样所有系统结点度数都相同,同时每组实验网络包含 256 至 1 百万个结点,所有查询在网络里被随机且均匀地激发.本文所有实验都是在上述条件下进行的.

6.1 动态网络路由效率

本节给出两组实验以检验网络在动态环境下的最大和平均路由延迟.在第 1 组实验中,为模拟动态网络,结点总数大约 10%的结点被设置成加入和离开网络.最大路由跳数和平均路由跳数在此条件下被记录,同时第 2 节给出的最大查询延迟理论下界(dia-bound)和平均延迟下界(avg-bound)也在结果中给出,用来和实际系统的结果作以比较.图 7 给出第一组实验结果,图 7(a)给出最大延迟,平均查询效率由图 7(b)给出.从结果看出,DPhoenix 的最大和平均延迟超越了下界,而 BPhoenix 的效率略大于理论下界,因为其内嵌结构 Butterfly 不像 DPhoenix 的 deBruijn 有最优直径.同时 DPhoenix 和 BPhoenix 的查询效率远高于 Koorde 和 Viceroy,并且它们的跳数也少于理论下界.此结果验证了本文给出的理论下界对现有 P2P 系统的有效性,而基于 Trie 树构造的系统可以超越下界,因为它们能够更好地适应动态环境.



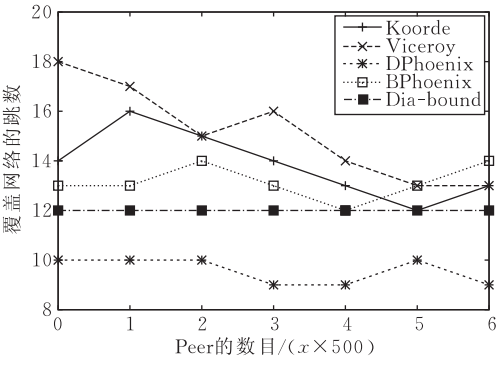
(a) 最大查询距离



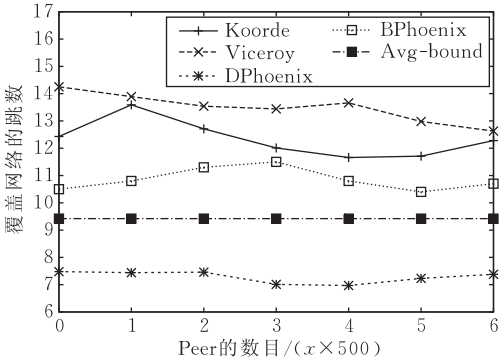
(b) 平均查询距离

图 7 动态网络路由效率

第 2 组实验中,每个网络起初有 64 K 个结点,后允许一定数目结点离开,同时设置相同数目的结点加入,从而网络总结点数几乎不变.图 8 给出此环境下各系统的路由跳数.从图中看出不论是最坏情



(a) 最大查询距离



(b) 平均查询距离

图 8 震荡网络路由效率的变化

况下还是平均情况下，DPhoenix 和 BPhoenix 系统跳数的波动都好于 Koorde 和 Viceory. 此结果说明本文的构建方法能使 P2P 系统很好地适应结点频繁的加入和退出，而现有系统的查询性能在网络震荡时受较大影响.

6.2　系统最优化

本节给出对 DPhoenix 和 BPhoenix 两系统应用最优化设计后的性能评测结果. 对最优资源放置策略，本实验假设每结点最大负载是 10^{2+x}，x 是和结点数目相关的整数，测试时分别取 $x=0,\cdots,12$. 系统查询和更新操作比例被设置成 60%和 40%. 根据这些实验条件，我们可得优化资源放置策略和拓扑构造策略后的网络代价，其中图 9 给出每个结点平均消耗的路由代价. 该结果表明在网络规模扩展的同时，优化后的系统 OPT-DPhoenix 和 OPT-BPhoenix 比原系统的路由显著变小. 但系统维护代价的实验结果（如图 10 所示）刚好和路由代价相反，基于最优设计的网络代价比非优化的网络要略高些. 因为，正如在最优化设计方法中指出的，不论是最优放置策略还是最优拓扑构建策略都是一个最优折中设计问题.

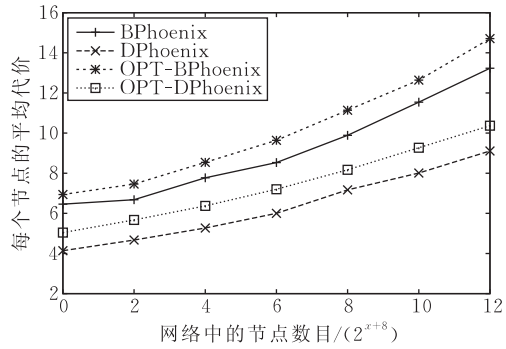


图 9　最优和非最优系统的平均路由代价

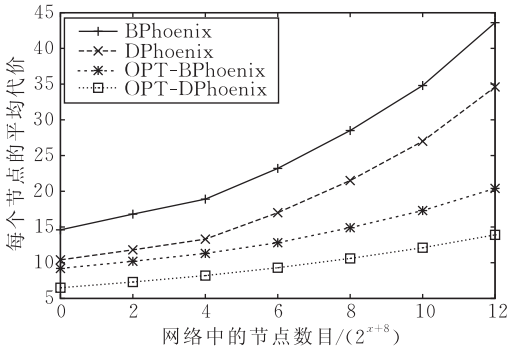


图 10　最优和非最优系统的平均维护代价

6.3　负载平衡

通常希望 P2P 系统的资源平均地分布到每个结点上，这样可以使结点收到其它结点均衡的访问，以至于系统总的访问代价和存储代价由每个结点均匀地来承担. 本文采用结点所存资源数量等于平衡负载的数目占总结点数的百分比来衡量系统平衡负载的能力，其中允许 5%的误差，例如平均负载是每结点 100 个资源，那么存储 95～105 个资源的结点也被计算在比例中.

实验中，每系统有 10 000 个结点，系统资源数是 10^5 至 10^6 并以 10^5 的数量递增. 如图 11 展示的结果，随资源数目的递增，结点所占比例下降. 同时该结果也显示出，由于负载平衡算法的作用，DPhoenix 和 BPhoenix 的曲线比其它网络更稳定，而且比例也是高于其它网络的. 反而由于 Viceroy 和 Koorde 没有有效的负载平衡算法，网络资源分配的很不平均. 图 12 给出应用本文的负载平衡方法后，DPhoenix 和 BPhoenix 系统所产生的平均网络代价. 从图中看出，产生的消息数目略多一些，但这个代价是可接受的，因为负载平衡算法只作用于 Trie 树的一小部分.

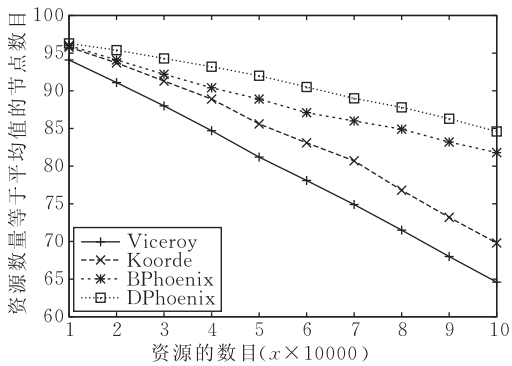


图 11　资源分布的变化

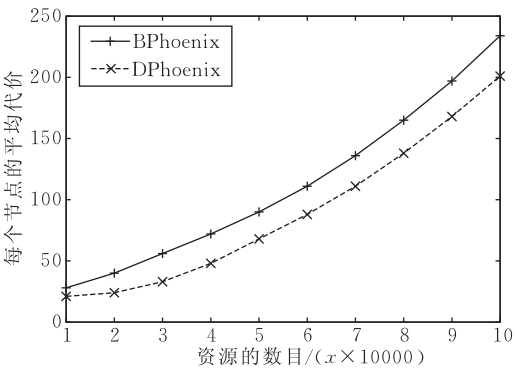


图 12　负载平衡算法平均代价

6.4　容错性

测试容错性时，4000 个查询被均匀地在含有 10 000 个结点的网络里激发，每个结点以 0.1～0.3 的概率失效. 图 13 给出此条件下各系统的成功查询

比例. 从图中易看出,在相同失效概率的情况下,DPhoenix 和 BPhoenix 的成功查询百分比远高于其它系统并变化稳定,因为在本文给出的构建方法中,每环节都考虑了容错设计,并且都是相互对应的. 其中 BPhoenix 要比 DPhoenix 的鲁棒性差些,因为消息要在不同 Trie 树间路由会增加查询失败的概率. 尽管 Viceory,Koorde 和 DPhoenix,BPhoenix 是基于相同互联网络结构的,但它们遭遇到更多的失败,主要原因是它们的路由协议只是不断地选择离目标最近的邻居转发消息,不允许沿其它路径路由.

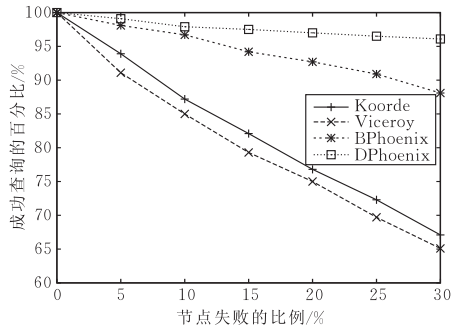


图 13 成功查询比例

7 结束语

本文首先对现有 P2P 系统给出动态环境下网络直径和平均距离的理论下界. 其次提出基于动态 Trie 树的通用构建框架,任何静态互联网络基于此框架都可被构建成 P2P 系统,并使系统在动态环境下保持静态网络的优点,同时对系统的参数给出最优化设计. deBruijn 和 Butterfly 作为互联网络的代表用通用方法构建出 DPhoenix 和 BPhoenix 两个 P2P 系统,两系统拥有的拓扑维护算法和路由算法保证了资源的平均分布、健壮的网络和高效的路由. 实验验证了本文的设计,DPhoenix 和 BPhoenix 路由效率渐进地接近其相应的互联网络,并不受到理论下界的限制. 将来,我们将开发出 DPhoenix 原型系统,并在其上构建大规模分布式应用.

参 考 文 献

[1] Chawathe Y, Ratnasamy S, Breslau L. Making gnutella-like P2P systems scalable//Proceedings of the ACM SIGCOMM. Karlsruhe, Germany, 2003: 407-418

[2] Ratnasamy S, Francis P, Handley M. A scalable content-addressable network//Proceedings of the ACM SIGCOMM. San Diego, USA, 2001: 234-245

[3] Stoica I, Morris R, Karger D et al. Chord: A scalable peer-to-peer lookup service for internet applications//Proceedings of the ACM SIGCOMM. San Diego, USA, 2001: 446-457

[4] Rowstron A, Druschel P. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. Lecture Notes in Computer Science, 2001, 12(8): 2218-2237

[5] Malkhi D, Naor M, Ratajczak D. Viceroy: Scalable and dynamic emulation of the butterfly//Proceedings of the 21st ACM Symposium on Principles of Distributed Computing. Monterey, USA, 2002: 45-56

[6] Xu J, Kumar A, Yu X. On the fundamental tradeoffs between routing table size and network diameter in peer-to-peer networks. IEEE Journal on Selected Areas in Communications, 2004, 22(1): 151-163

[7] Shen H, Xu C, Chen G. Cycloid: A constant-degree and lookup efficient P2P overlay network//Proceedings of the 18th International Parallel and Distributed Processing Symposium. USA, 2004: 356-366

[8] Fraigniaud P, Gauron P. D2B: A de bruijn based content-addressable network. Theory Computing, 2006, 35(1): 65-79

[9] Kaashoek F, Karger D R. Koorde: A simple degree-optimal hash table//Proceedings of the 2nd International Peer-To-Peer Systems Workshop (IPTPS). Berkeley, CA, USA, 2003: 78-87

[10] Loguinov D, Kumar A, Rai V et al. Graph-theoretic analysis of structured peer-to-peer systems: Routing distances and fault resilience//Proceedings of the ACM SIGCOMM. Karlsruhe, Germany, 2003: 258-269

[11] Li D, Lu X, Wu J. Fissione: A scalable constant degree and low congestion dht scheme based on kautz graphs//Proceedings of the IEEE INFOCOM. USA, 2005: 1677-1688

[12] Guo D, Wu J, Chen H. Moore: An extendable P2P network based on incomplete kautz digraph with constant degree//Proceedings of the 26th IEEE INFOCOM. 2007: 1567-1578

[13] Maymounkov P, Mazieres D. Kademlia: A peer-to-peer information system based on the xor metric//Proceedings of the IPTPS. Cambridge, USA, 2002: 649-657

[14] Zhang Y, Liu L, Li D. Distributed line graphs: A universal framework for building DHTs based on arbitrary constant-degree graphs//Proceedings of the ICDCS. 2008: 1179-1189

[15] Bridges W G, Toueg S. On the impossibility of directed moore graphs. Combinatorial Theory, 1980, 29(3): 1107-1120

[16] Sivarajan K N, Ramaswami R. Lightwave networks based on de Bruijn graphs. IEEE/ACM Transactions on Networking, 1994, 22(8): 765-779

[17] Szpankowski W. Patricia trees again revisited. Journal of the ACM, 1990, 37(8): 637-745

[18] DeGroot M H. Probability and Statistics. Washington, USA: Addison-Wesley, 2001

[19] Aberer K. P-Grid: A self-organizing access structure for P2P information systems//Proceedings of the 6th International Conference on Cooperative Information Systems. Torino, Italy, 2001: 638-647

YUAN Ye, born in 1981, Ph. D., associate professor. His research interests include graph database, probabilistic database, P2P computing, and data piracy.

WANG Guo-Ren, born in 1966, professor, Ph. D. supervisor. His research interests include XML data management, query processing and optimization, probabilistic database, and bioinformatics.

GUO De-Ke, born in 1980, Ph. D., associate professor. His research interests include multi-hop wireless networks, peer-to-peer networks, and data center networking.

Background

The growing popularity of P2P systems makes them very likely substrate for future large-scale information architectures, and thus many studies are proposed to construct P2P systems. Most P2P systems are based on overlay networks. For example, Chord and Pastry are based on the Hypercube topology; Viceroy and Ulysses are based on the Butterfly topology; Cycloid is based on the CCC topology; CAN is based on the d-dimensional torus topology; Koorde and D2B are based on the deBruijn topology; Moore and BAKE are based on the Kautz topology. However, one critical requirement of these systems is that the number of nodes must be some given values determined by the node degree and the network diameter. Hence, the corresponding approaches are often impractical when nodes frequently join, leave, and fail. Therefore, in this paper, we propose the dynamic multi-way Trie tree structure, a universal framework based on which any interconnect network can be adopted to construct a P2P system.

This research was supported by the National Natural Science Funds for Distinguished Young Scholar (Grant No. 61025007), National Natural Science Foundation of China Key Program (Grant No. 60933001), National Natural Science Foundation of China (Grant No. 61073063) and the Fundamental Research Funds for the Central Universities (Grant No. N090404012).