P2P 网络中基于 DHT 的自适应 Chord 风险模型

黄飞雪[1],陈 青[2],李志洁[3]
HUANG Fei-xue[1],CHEN Qing[2],LI Zhi-jie[3]

1.大连理工大学 经济系,辽宁 大连 116024
2.大连理工大学 软件学院,辽宁 大连 116024
3.大连理工大学 计算机科学与工程系,辽宁 大连 116024
1.Department of Economics,Dalian University of Technology,Dalian,Liaoning 116024,China
2.School of Software,Dalian University of Technology,Dalian,Liaoning 116024,China
3.Department of Computer Science and Engineering,Dalian University of Technology,Dalian,Liaoning 116024,China

HUANG Fei-xue,CHEN Qing,LI Zhi-jie.Self-adaptive Chord risk model based on DHT for Peer-to-Peer networks. Computer Engineering and Applications,2007,43(36):150-152.

Abstract: This study's objective is to solve the problem that the Chord model is not appropriate for dynamic network as it brings large numbers of messages.A self-adaptive Chord based on DHT (Distribute Hash Table) is proposed.When node join or leave,it does not maintain consistency of entire network's logic topology,but only update routing table of node's predecessor and successor.Besides,nodes update their routing table when they transmit messages.Therefore logic topology of entire network tends to a consistent state.The experimental results show that self-adaptive Chord both evidently lessen messages caused by node join or leave, and basically reserve efficient search performance which is close to Chord.The conclusion indicates that model could provide a candidate solution of ad hoc P2P network with high churn rate.

Key words: Distribute Hash Table(DHT);self-adaptive Chord;dynamic network;grid computing;Peer-to-Peer(P2P) computing

摘 要:针对 Chord 模型在节点加入或离开时产生大量消息,不适用于动态网络的问题,提出一种基于分布式哈希表(Distribute Hash Table,DHT)的自适应 Chord 模型,即 Self-adaptive Chord。方法是该模型在节点加入或离开的时候暂不考虑整个网络逻辑拓扑的一致性,只简单更新其前驱节点和后继节点的路由表,而在节点转发消息时动态地调整各节点路由表,使得网络逻辑拓扑动态地趋向于一致。通过实验对比评估了自适应 Chord 和 Chord 性能,结果表明自适应 Chord 能有效降低由于网络动荡引发的消息数量,同时基本保留了 Chord 的高效率查询。结论为自适应 Chord 提供了一种在节点动荡频繁的环境下的候选解决方案。

关键词:分布式哈希表;自适应 Chord;动态网络;网格计算;对等计算
文章编号:1002-8331(2007)36-0150-03 **文献标识码:**A **中图分类号:**TP393

1 引言

作为网格系统的一个重要组成部分,P2P(Peer-to-Peer)网络,即对等计算网络,提供了一种大规模异构环境下进行资源共享的有效途径。目前 P2P 网络大体共有三种基本结构:

(1)非完全分布式 P2P:采用中心目录服务器保存节点信息,资源查找在服务器上进行而后节点间自行通信,如 Napster[1]。这种结构虽然响应快速,但依赖于中心服务器会存在单点失效的问题。

(2)完全分布式 P2P:这种系统以 Gnutella[2]为代表,所有节点相互独立,资源查找需在整个网络中洪泛式查询,它克服了非纯 P2P 的单点依赖问题,但是洪泛式查询机制产生巨大的消息数量使系统不易扩展。

(3)结构化分布式 P2P:如 Chord[3]、CAN[4]、Tapestry[5]、Pastry[6],一般基于 DHT(Distribute Hash Table)技术[7,8],为节点建立一定结构的逻辑拓扑,搜索资源时有选择地进行消息转发,基于 DHT 的结构化 P2P 系统在可扩展性、容错能力、查找速度等方面都有很大提高,但它需要以很高代价维护既定拓扑,不能很好地适应结点高度动态的 P2P 环境。

在 Chord 的基础上提出基于分布式哈希表(Distribute Hash Table,DHT)的自适应 Chord 模型,即 Self-adaptive Chord。该模型是在节点加入或离开的时候暂不考虑整个网络逻辑拓扑的一致性,只简单更新节点前驱节点和后继节点的路由表,而是在节点转发消息时动态地调整各节点路由表,使得网络逻辑拓扑动态地趋向于一致。这将更适用于动态 P2P 网络环境。

2 Chord 算法

Chord 是由 MIT 提出的一种较有效的分布式路由算法。它采用一致性杂凑哈希(Consistent hashing)函数[9]为每个节点或查找关键值分配一个 m 位的标识符(key,本文中 key 即代表标示符),如使用 SHA-1[10]基本杂凑函数,每个节点的 key 由节点 IP 地址计算得到,每个查找关键值的 key 则由关键值计算得到。一致性杂凑函数能保证节点在整个标识符空间中基本均匀分布,并且不同的查找关键值的 key 不相等。Chord 中节点按如下规则保存关键字:节点按照其 key 值组成一个环,key 值为 K

基金项目:大连理工大学人文社会科学研究基金(No.DUTHS2007321)。

的关键值保存在环中第一个 *key* 值等于 *K* 或者后继于 *K* 的节点上，即从 *K* 开始在环上顺时针方向前进，第一个遇到的节点。该节点成为 *key* 值 *K* 的后继(successor)节点。

为了加快 *key* 值查询，节点除了保存自身的前驱(predecessor)节点、后继(successor)节点外,还需要保存一个 *m* 位的路由表(成为 Finger Table),如图 1 所示。表中第 *i* 项为与自身 *key* 间距 2^{i-1} 的后继节点路由信息,即:

$$finger[i]=successor(key[p]+2^{i-1}) \quad (1)$$

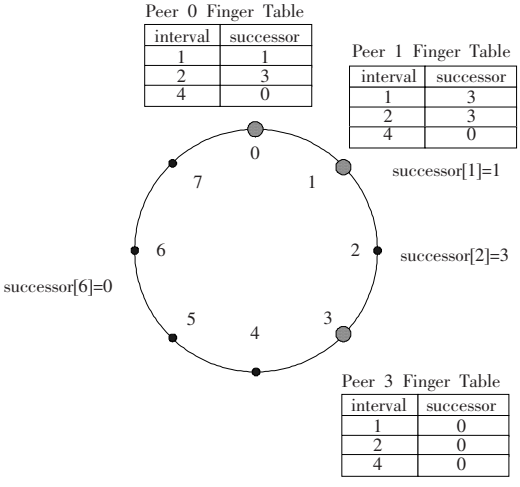

图 1　Chord 示意图

特殊的是 Finger Table 的第一项即为后继节点。

Chord 一次查询发送的消息数量是 $O(\log N)$[7],*N* 为网络规模。在新节点加入的时候,为了保证 Finger Table 的一致性,网络中所有可能需要更新 Finger Table 的节点都需要进行相应操作以反映出新加入的节点,同样地对于节点离开,这些节点也要更新 Finger Table。

假设节点 *P* 加入网络,*Q* 为网络中已有节点,若 *Q* 的 Finger Table 中第 *i* 项可能指向 *P*,则 *Q* 的 *key* 领先于 *P* 的 *key* 至少 2^{i-1}，故节点 *P* 加入网络后需要发出 *Key* 为 *k* 的查询以通知其它节点更新 Finger Table,其中

$$k=P.key-2^{i-1} \quad (i\in[1,m]) \quad (2)$$

所以节点加入产生的消息数量是 $O(\log^2 N)$。同样节点离开操作产生的消息数量也是 $O(\log^2 N)$[7]。由此可见 Chord 在用户稳定性不高的环境中，会由于节点频繁加入离开而产生大量消息，急剧加大系统的负荷。

3　改进后的自适应 Chord 模型

3.1　改进策略

Chord 在节点加入或离开时产生大量消息的根本原因在于它实时维护网络逻辑拓扑，更新所有可能需要更新节点的 Finger Table，从而保证查询效率使每次查询消息发送数量稳定在 $O(\log N)$。自适应 Chord 的改进思路为:由于查询进行时消息会在不同节点之间进行转发,故在节点加入或离开的时候考虑不维护网络逻辑拓扑的一致性，允许各节点 Finger Table 路由信息不一致，而在转发消息的时候动态地调整 Finger Table 路由信息,使得网络逻辑拓扑不断趋向于一致,这样可以大大减少由于网络动荡引发的消息数量,而随着网络运行查询效率会逐渐接近于 Chord。自适应 Chord 中消息格式定义如表 1 所示。

表 1　Self-adaptive Chord 消息格式定义

属性名	定义
ID	消息标示符
TYPE	消息类型标示
SOURCE	消息产生源路由信息
PARAMETER	消息附带参数

考虑到一致性杂凑函数的特性：对于任意集合的 *N* 个节点和 *K* 个关键值，其中每个节点负责保存的 *key* 范围近似相同,而节点发起一个查询的时候,由杂凑函数计算出来的 *key* 值在整个标识符空间等概率分布。令 *P* 为网络中任意节点,则 *P* 向网络中任何节点发送查询消息的概率是相等的,同样理论上 *P* 可能接收到网络中任意节点发出的消息。因此可在节点转发消息的时候,由消息产生源来更新本地 Finger Table,使 finger[*i*] 逐渐逼近于 $successor(key+2^{i-1})$。这样 Self-adaptive Chord 的查询效率会不断接近于 Chord。

3.2　节点增删算法

Self-adaptive Chord 在节点加入或离开的时候不同步更新其他节点 Finger Table,只简单维护其前驱后继节点的一致性。

节点加入操作描述如下:

(1)寻找到加入点,即加入节点 *key* 的前驱节点。

(2)根据前驱节点的后继来初始化新加入节点 Finger Table,若满足 finger[*i*]条件则 finger[*i*]指向前驱节点的后继,不满足则指向新节点自身。

(3)通知前驱节点更改其后继为新加入节点。

(4)通知后继节点更改其前驱为新加入节点。

(5)移动索引,加入网络。

节点的加入过程只需发送两个消息。

离开操作同样只需要通知前驱后继节点，发送两个消息,描述如下:

(1)更改前驱节点后继为本节点后继。

(2)更改后继节点前驱为本节点前驱。

(3)移动索引至后继节点,离开。

如图 2 所示节点 6 加入,Finger 中更新了的记录用斜体加粗标出。

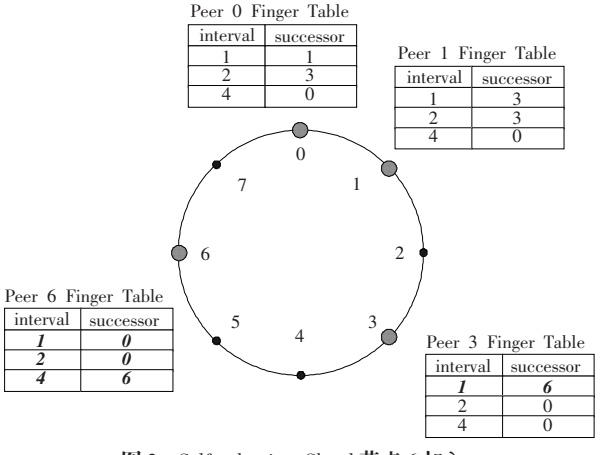

图 2　Self-adaptive Chord 节点 6 加入

3.3　节点转发算法

节点转发消息前根据消息源对本地的 Finger Table 进行更新,更新操作伪代码描述如下:

//节点 *n* 更新本地 Finger Table,*p* 为发送消息来源节点

```
n.updateFinger(p){
    for(int i=m-1;i>=0;i--){
        if(p.key∈[n.key+finger[i].interval,finger[i].key))
          finger[i]=p;
        else break;
    }
}
```

节点转发流程图如图 3 所示。

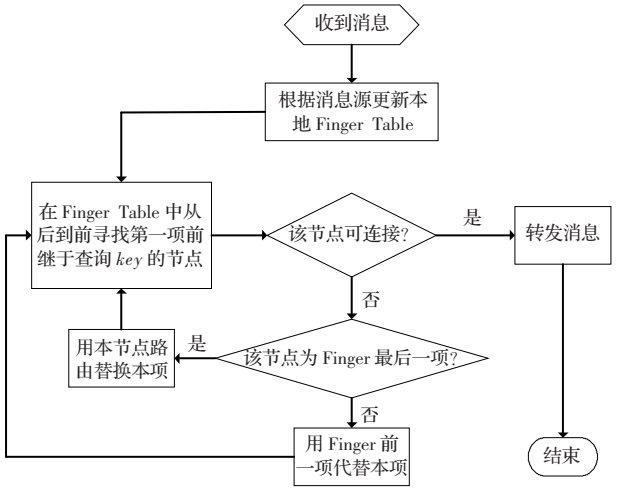

图 3　节点转发流程图

4　实验

4.1　实验目的与环境设置

实验目的是为了评估 Self-adaptive Chord 与 Chord 相比哪个更适用于动态网络。

本文采用 JAVA 开发了一个 P2P 网络模拟实验环境。运行在硬件为主频 1.86 G CPU、内存 1 G 的 PC 机；软件为 Windows XP,JDK1.4.2。

实验中模拟一致性杂凑函数采用标识符的长度是 10 位。节点加入网络是随机选取一个网络中已有的节点加入。一次查询是随机选择一个网络中已有节点，进行一个随机 *key* 的查询。由于实验采取随机抽取的策略,故实验结果会有细微差距,下面出现的消息数量实验数据，均为同样环境下进行 10 次实验的平均值。

4.2　实验结果与讨论

4.2.1　无网络动荡时 Self-adaptive Chord 与 Chord 比较

比较 Self-adaptive Chord 和 Chord 在不存在网络动荡时的查询效率,实验步骤为 Chord 和 Self-adaptive Chord 分别生成 500 个节点的网络,然后分别顺序执行 20 组查询,其中每组 500 个查询。

如图 4 所示,Chord 执行查询的效率是稳定的，每 500 个查询产生消息 3 000 多个，平均每个查询产生消息 6 个左右,这是由于 Chord 一次查询产生消息数量为 O(log *N*)。而 Self-adaptive Chord 在前几组查询的时候效率是不如 Chord 的,因为在刚生成的网络里面,大部分节点的本地 Finger Table 只是相对于它的后继做出初始化,这些节点不知道整个网络的逻辑拓扑,在刚生成的网络里面进行查询,很可能是按节点的后继节点一步步遍历整个环,故效率不如 Chord。在进行了一些查询以后，节点通过转发消息动态地更新了本地的路由信息,可以看到越往后执行的查询组，其产生的消息数量越接近

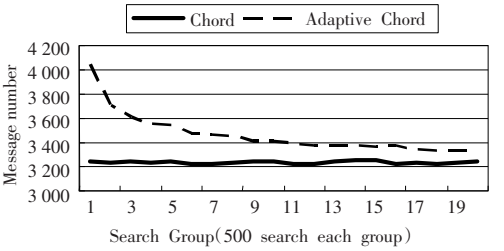

图 4　Chord 同 Self-adaptive Chord 产生消息数量比较(无网络动荡)

Chord,可见 Self-adaptive Chord 随着网络的运行,其查询效率会越来越接近于 Chord。

4.2.2　一般网络动荡时 Self-adaptive Chord 与 Chord 比较

实验步骤为 Chord 和 Self-adaptive Chord 分别生成 500 个节点的网络,然后分别顺序执行 20 组查询,其中每组 500 个查询,并且每组查询时加入一定数量的节点。

如图 5 所示,每组查询时加入 5 个节点,可以看到,相比于图 4,Self-adaptive Chord 受网络动荡(在这里为节点加入)的影响很小，每组查询产生的消息数量越来越趋于稳定。而 Chord 相比于图 4 可以看到，网络动荡对于 Chord 产生很大影响，无网络动荡时,Chord 查询产生消息数量稳定在 3 200 左右，但是每组查询中加入 5 个节点后,Chord 产生消息数量明显增多,并且消息数量曲线呈上升趋势,可见随着网络节点的增多,节点的加入离开操作产生的消息有增加趋势。

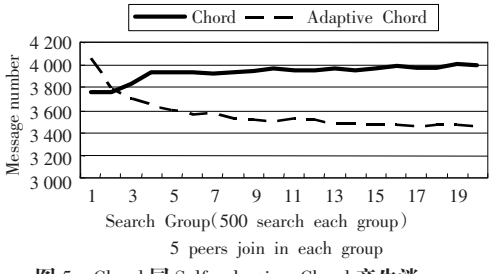

图 5　Chord 同 Self-adaptive Chord 产生消息数量比较(一般网络动荡)

5　结论

本文针对 Chord 模型在节点加入或离开时产生大量消息,不适用于网络动荡频繁的环境的问题,提出了一种基于分布式哈希表(Distribute Hash Table,DHT)的自适应 Chord 模型,该模型在节点加入或离开时暂不考虑整个网络逻辑拓扑的一致性,只简单更新其前驱节点和后继节点的路由表,而在节点转发消息时动态地调整各节点路由表,使得网络逻辑拓扑动态地趋向于一致。

自适应的 Chord 模型具有如下两方面特点:

(1)有效减少了由于网络动荡而引起的消息数量。

(2)查询效率基本接近于 Chord。

所以自适应的 Chord 模型更适用于动态的 P2P 网络环境。

(收稿日期:2007 年 5 月)

参考文献:

[1] Saroiu S,Gummadi K P,Gribble S D.Measuring and analyzing the characteristics of Napster and Gnutella hosts[J].Multimedia Systems,2003,9(2):170-184.

(下转 196 页)