第 35 卷 第 2 期 计 算 机 工 程 2009 年 1 月
Vol.35 No.2 Computer Engineering January 2009
·软件技术与数据库· 文章编号：1000—3428(2009)02—0055—03 文献标识码：A 中图分类号：TP393

基于 PM-chord 算法的数据网格副本定位方法

王福业[1]，高敬阳[1]，危胜军[2]
（1. 北京化工大学信息科学与技术学院，北京 100029；2. 北京理工大学计算机网络攻防对抗技术实验室，北京 100081）

摘 要：提出一种数据网格环境下的分布式副本定位算法 PM-chord，以完成对所需数据副本的快速定位，降低存储和更新开销。PM-chord 取逆时针方向为 chord 环的正方向，对节点和数据编码后按前缀匹配的原则查询数据，同时增加前继副本机制。分析及实验表明，该算法具有很高的副本定位效率，能够有效解决网格中的查询“热点”问题，具有良好的可靠性和可行性。

关键词：数据网格；副本；副本定位

Replica Location Method Based on PM-chord Algorithm in Data Grids

WANG Fu-ye[1], GAO Jing-yang[1], WEI Sheng-jun[2]
(1. School of Information Science and Technology, Beijing University of Chemical Technology, Beijing 100029;
2. Lab of Computer Network Defense Technology, Beijing Institute of Technology, Beijing 100081)

【Abstract】 Data replication is a general mechanism to improve performance and availability for large-scale data-intensive applications. However, ensuring efficient and fast access to one or more replicas of desired data is a challenging problem. To solve this problem, a Prefix Matching-chord (PM-chord) algorithm which leads into chord protocol in P2P field is proposed. PM-chord takes anti-clockwise as positive, searches data by using prefix matching principle and adds the predecessor replication mechanism. Analysis and experiments show that PM-chord has good performance on replica location, is effective on hot spots question in data grids, and can achieve reliability and adaptability of implementation.

【Key words】 data grid; replica; replica location

1 概述

数据网格主要用于对高度异构、分布广泛、数量巨大的数据资源的访问、共享和处理。由于 Internet 较大的传输延迟和较低的传输速率降低了数据网格的性能，因此数据网格广泛采用副本技术提高数据的访问效率，改善数据网格的性能。

在数据网格中，一个数据文件可能存在多个副本。如何根据约束条件找到该数据文件的一个或多个副本的问题称为副本定位问题[1]。SRB(Storage Resource Broker)使用中央服务器提供副本定位服务，存在系统扩展困难和单点失效问题。文献[2]提出的分布式、自适应的副本定位方法尽管采用了 Bloom Filter技术，但存储和更新开销仍然很大。DSRL方法[3]采用动态均衡映射方法将全局副本定位信息均匀分布在多个宿主节点上，查询和扩展性能良好，但每个宿主节点要维护其他所有宿主节点的相关信息，存储和更新开销较大。本文提出的副本定位方法对chord[4]算法进行了改进，分离了节点查询和数据查询，增加了前继副本机制，不但副本定位效率高，有效解决了因节点失效而丢失数据的问题，而且减少了信息存储和更新开销。

2 关键概念

chord 是一种基于分布式哈希表(DHT)的查找算法，它使用的是折半查找方法，系统中的每个节点维护 $O(\log N)$个节点的路由信息(N 为系统中节点总数)，每次查找的最大跳数不超过 $O(\log N)$，具有负载均衡、分布、可扩展、容错等特性。Globus 设计了 RLS(Replication Location Service)机制来注册和发现数据副本。RLS 设计包括 2 部分：(1)LRCs(Local Replica Catalogs)：维护逻辑文件名(lfn)到物理文件名(pfn)的映射，物理文件名部分包含实际的文件名、存放地址和存取速度等信息。(2)RLIs (Replica Location Indices)：维护 lfn 到 LRC 的映射。只要查找到 lfn 所在的 RLI 节点，就可以定位数据副本的实际存储地址。RLS 结构如图 1 所示。

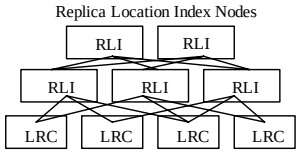

图 1 GT4 中RLS结构示例[5]

3 PM-chord 算法

3.1 算法描述

给定标识位长度为m($m \leq 128$, 2^m为系统最大容量)，节点集合Nodes{Node[1],Node[2],…,Node[x]}和副本逻辑文件名集合LFNs{lfn[1],lfn[2],…,lfn[y]}，(x和y均小于 2^m)，对上述 2 个集合中的所有元素实施MD5 运算得到 128 bit的数字标识，节点可以利用其IP地址和端口号进行运算，其十进制值即为该节点的chord值；lfn编码后的十进制值即为数据的id。根据

基金项目：国家部委基础研究基金资助项目
作者简介：王福业(1982－)，男，硕士研究生，主研方向：数据网格，分布式系统；高敬阳，副教授、硕士；危胜军，讲师、博士
收稿日期：2008-07-08 **E-mail：**willfcareer@sohu.com

节点的chord值以逆时针方向为正方向组成chord环，并且按照最大前缀匹配原则将所有lfn分配到合适的节点上。

定义 在 chord 环上所有的节点中，与要查询的数据值前缀匹配位数最大的节点称为该数据的最佳匹配点(optimal node)。如果所有节点匹配位数均为 0，则取数据值对应的后继为最佳匹配点。

表 1 给出了 Node[n]中变量的定义。

表 1　Node[n]中变量的定义

变量	定义
finger[k].start	$(n-2^{k-1}) \bmod 2^m$, $1 \le k \le m$
.interval	[finger[k].start , finger[k+1].start)
.node	first node$\le n$.finger[k].start
successor	chord 环上的下一个节点，也就是 finger[1].node
predecessor	chord 环上的前一个节点
.resp range	节点 n 所负责维护的数据范围(responsible range)，也就是最佳匹配点为 n 的所有数据的范围
.replicas_range	后继节点在 n 上存放的副本的总范围
.prede_list	节点 n 所维护的前继链表，大小为 $O(\log N)$
.prede_max	prede_list 中距离 n 最远的前继节点

当 m=3 且系统中存在 3 个节点(Node[0], Node[5], Node[7])时，Node[7]的路由间隔如图 2 所示，系统中各节点路由表及数据分配如图 3 所示。

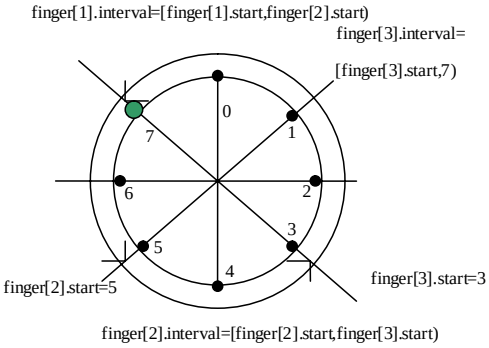

图 2　Node[7]的路由间隔示例

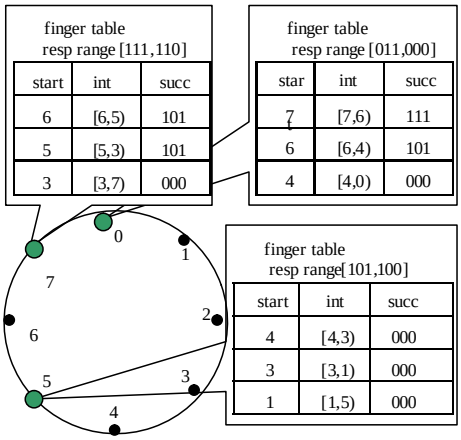

图 3　网络路由表和数据分配

3.2　节点加入和退出

节点加入和退出的原理及伪代码与原 chord 算法一致，节点加入或退出后需要重新划定前继或后继的 resp_range，并且移动数据到新的节点上，节点加入数据分配伪代码如下：

```
//node n joins the network
n.redistribute()
   if(n∈successor.resp range){
successor.resp_range = successor.resp range
– n.resp range;
// move keys from successor to n if keys∈ n.resp_range;
   }else{
predecessor.resp_range= predecessor.resp_range
– n.resp_range;
// move keys from predecessor to n if keys∈ n.resp range;}
```

图 4 为 Node[2]加入后系统中各节点的路由表及数据分配情况。

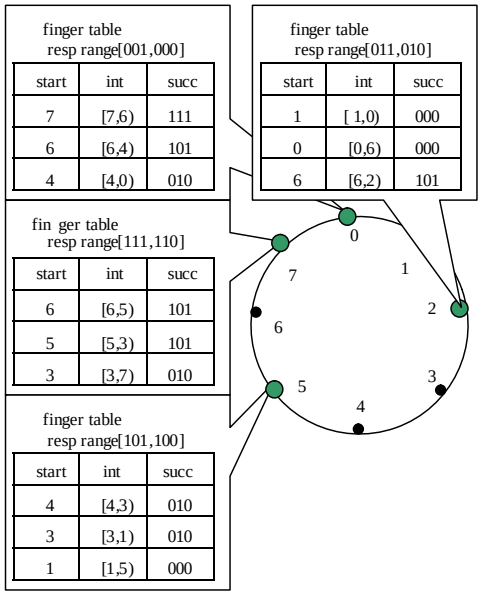

图 4　Node[2]加入后的网络路由表及数据分配

3.3　前继副本机制

为了解决因节点失效而丢失数据的问题并平衡查询负载，将每个节点上的数据做$O(\log N)$个副本分别存储在该节点的$O(\log N)$个前继节点上，采用软状态协议进行信息更新。当某节点失效时，数据可以从失效节点前继取得。因为在查询过程中总要经过目标节点的前继，所以在发现当前节点包含要查询数据的副本时，即可直接返回结果。这不但减少了查询跳数，而且可以解决网格中经常出现的查询“热点”问题，如图 5 所示，其中，节点Node[i]为根节点(“热点”)；k_1,k_2为自然数，$0<k_1<k_2 \le O(\log N)$。

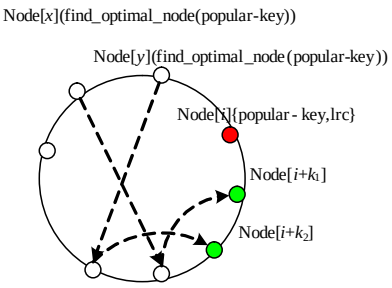

图 5　前继副本机制下数据查询示例

3.4　查询

PM-chord 算法下的数据不一定存储在其相应的后继节点。数据查询伪代码如下：

```
// ask node n to find id's optimal node or replica node
// initial value of np and ns are null
n.find_optimal_node(np,ns,id){
if(id∈(n.resp_range∪n.replicas range))
return n;
else if(id∈[n.prede_max,n) )
```

```
return optimal node from n.prede_list;
else if(n_p != null && n∈(n_p,id)){
        sort_finger_table(id);
        { //依次取排序后路由表中下一节点
  n' = n.finger_table_next;
}while (n_s != null && n_s ∈(n, n'));
        result = n'.find_optimal_node(n,n_s,id);
  }else{ //产生误跳，利用上级路由表搜索
        n' = n_p.finger_table_next;
        result = n'.find_optimal_node(n_p,n,id);
  }
  return result;
}
n.sort_finger_table(id){
```

将路由表项按如下优先级(1：与 id 匹配位数最大；2：距离 id 最近；3：前继优先)进行排序

}

4 分析评估与模拟实验

4.1 分析评估

根据 PM-chord 算法定义和折半查找的特性，可以得出如下结论：

理论 1 记 N 为系统节点总数，I 为全部数据范围，i 和 j 为 chord 环上的 2 个节点，有下面结论成立：

(1) $\forall(i,j), i \neq j,\ i.\text{resp range} \cap j.\text{resp range} = \varnothing$

(2) $I = \bigcup_{i=1}^{N} i.\text{resp range}$

理论2 任何节点的加入或者退出只影响其直接前继或者直接后继的resp_range。

理论3 所有数据的最佳匹配点都是唯一的，并且所有数据均存储在自己的最佳匹配点上。

理论4 在查找过程中，每次递归都将更靠近最佳匹配点，并且不超过$O(\log N)$跳就能够到达最佳匹配点。

4.2 模拟实验

在一台Pentium 4 计算机(CPU 2.2 GHz，内存 1 GB)上对PM-chord算法进行模拟。用Java独立进程模拟RLI节点，通信方式为socket。共对 1~15 个RLI节点、分别预装载全局映射记录总数为 2^{20}(=1 048 576)和 2^{24}(=16 777 216)的 30 种情况(标识位长度m取 128)进行了模拟。每种情况进行 1 000 次查询操作，得到了RLI查询操作的平均查询响应时间和上下界，如图 6 所示。从结果中可以得出如下结论：(1)PM-chord算法具有较好的数据查询性能。(2)随着RLI节点数量的增加，平均查询相应时间仅呈对数关系增加，并且在全局映射总数相差很大的情况下，查询相应时间却变化不大，因此，系统具有良好的可扩展性和稳定性。

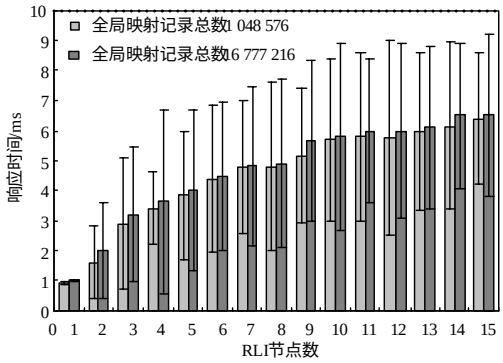

图 6 查询响应模拟结果

5 结束语

PM-chord 算法基于 chord 理论实现，具有很强的稳定性、灵活性、可扩展性。前缀匹配机制进一步提高了查询效率。前继副本机制使其具有更强的容错性能。下一步将在实际环境中对 PM-chord 的性能进行更全面的分析、实验和模拟，同时将 PM-chord 应用到正在开发的数据网格系统中。

参考文献

[1] 陈绍宇, 宋佳兴, 刘卫东, 等. 基于 P2P 覆盖网络的数据网格副本定位机制[J]. 计算机工程 2006, 32(13): 111-113.

[2] Ripeanu M, Foster I. A Decentralized, Adaptive, Replica Location Mechanism[C]//Proc. of the 11th IEEE International Symposium on High Performance Distributed Computing. Edinburgh, Scotland: IEEE Press, 2002.

[3] Li Dongsheng, Xiao Nong, Lu Xicheng, et al. Dynamic Self-adaptive Replica Location Method in Data Grids[C]//Proceedings of the IEEE International Conference on Cluster Computing. [S. l.]: IEEE Press, 2003.

[4] Stoica I, Morris R, Karger D, et al. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications[C]//Proceedings of ACM SIGCOMM'01. [S. l.]: ACM Press, 2001.

[5] Chervenak A L. Applying Peer-to-Peer Techniques to Grid Replica Location Services[J]. Journal of Grid Computing, 2006, 4(1): 49-69.

(上接第 54 页)

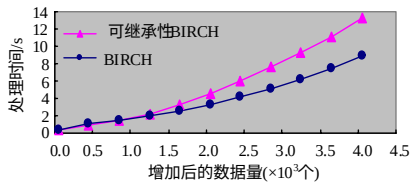

图 5 多次增加数据时可继承性 BIRCH 与 BIRCH 时间性能比较

4 结束语

本文结合 BIRCH 算法和 XML 提出一种新的数据处理模型，实现了 BIRCH 算法的可继承性。相关实验结果验证了算法的正确性和有效性。下一步工作将把可继承性 BIRCH 应用到 QAR 数据的处理方面，并作进一步完善。

参考文献

[1] 邵峰晶, 于忠清. 数据挖掘原理与算法[M]. 北京: 中国水利水电出版社, 2003.

[2] Ramakrishnan R, Livny M. Birch: A New Data Clustering Algorithm and Its Application[J]. Data Mining and Knowledge Discovery, 1997, 1(2): 141-182.

[3] Han Jiawei, Micheline K. 数据挖掘概念与技术[M]. 北京: 机械工业出版社, 2005.

[4] 万常选. XML 数据库技术[M]. 北京: 清华大学出版社, 2005.

[5] Peter G. 微软 XML 技术指南[M]. 北京: 中国电力出版社, 2003.