ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.18, No.8, August 2007, pp.1933−1942
DOI: 10.1360/jos181933

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

虚拟计算环境中的可扩展分布式资源信息服务*

张一鸣+, 李东升, 卢锡城

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

Scalable Distributed Resource Information Service for Internet-Based Virtual Computing Environment

ZHANG Yi-Ming+, LI Dong-Sheng, LU Xi-Cheng

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

+ Corresponding author: Phn: +86-731-4575816, Fax: +86-731-4556650, E-mail: sdiris@gmail.com, http://www.kylinx.com/zym.htm

Zhang YM, Li DS, Lu XC. Scalable distributed resource information service for Internet-based virtual computing environment. *Journal of Software*, 2007,18(8):1933−1942. http://www.jos.org.cn/1000-9825/18/1933.htm

Abstract: Based on the characteristics of evolution, autonomy and diversity of Internet resources, researchers recently proposed to realize the publication and query of Internet resource information through common DHT (distributed Hash table) information services. However, current research on resource information services is insufficient in generality, usability and adaptability. Aiming at the needs of iVCE (Internet-based virtual computing environment) for resource aggregation, the SDIRIS (scalable distributed resource information service) is proposed to construct. First, the adaptive DHT (A-FissionE) infrastructure is presented, which supports to adapt to different system scale and stability transparently. Second, the MR-FissionE, which is an efficient multiple-attribute range query algorithm, is presented based on A-FissionE. Theoretical analysis and experimental results prove that SDIRIS can realize resource information publication and query efficiently.

Key words: iVCE (Internet-based virtual computing environment); resource information service; adaptive DHT (distributed Hash table); multiple-attribute range query; ZKT (Z-curve Kautz tree)

摘 要: 基于网络资源的“成长性”、“自治性”和“多样性”,近年来,人们提出以通用 DHT(distributed Hash table)信息服务的形式实现网络资源信息的发布和查询.然而,现有的资源信息服务在通用性、易用性和自适应性等方面仍存在不足.针对虚拟计算环境 iVCE(Internet-based virtual computing environment)的资源聚合需求,提出构建可扩展的分布式资源信息服务 SDIRIS(scalable distributed resource information service).首先,提出采用自适应 DHT(adaptive FissionE,简称 A-FissionE)底层架构,以对上层应用透明的方式适应不同的系统规模和稳定性;其次,基于自适应 DHT 提出高效的多属性区间搜索算法(multiple-attribute range FissionE,简称 MR-FissionE).理论分析和模拟结果表明,SDIRIS 能够高效地实现资源信息的发布与查询功能.

关键词: 虚拟计算环境;资源信息服务;自适应 DHT(distributed Hash table);多属性区间搜索;ZKT(Z-curve Kautz

* Supported by the National Natural Science Foundation of China under Grant Nos.60673167, 90412011 (国家自然科学基金); the National Basic Research Program of China under Grant No.2005CB321801 (国家重点基础研究发展计划(973))

Received 2007-03-02; Accepted 2007-04-26

tree)

中图法分类号: TP393 文献标识码: A

随着网络技术的发展,人们希望在开放的互联网基础设施之上为终端用户或应用系统提供和谐、可信、透明的虚拟计算环境(Internet-based virtual computing environment,简称iVCE)[1].传统的计算机资源管理模式具有明确的被管资源边界、全局的资源控制以及统一的资源描述.然而,互联网中的资源具有“成长性”、“自治性”和“多样性”特性.成长性是指资源的规模及其关联关系不断变化,使得资源的范围难以确定;自治性是指资源局部自治,难以进行全局的集中控制;多样性是指资源存在广泛差异,特征难以统一描述.互联网资源的上述自然特性给资源信息的发布和查询带来了巨大的挑战.

互联网中的节点数量巨大,而且节点加入、退出的频率(churn rate)较高.为了获得较好的可扩展性和容错性,近年来各种分布式应用通常基于分布式哈希表(distributed Hash table,简称DHT)技术来实现资源信息的发布和查询功能,例如网格信息服务、数据共享与缓存、Web服务放置、传感器网络以及Internet内存共享[2,3]等.然而在上述分布式应用中,每种应用都需要独立设计、实现和部署自己的信息发布和查询功能,给分布式应用的开发者带来了极大的负担.

针对该问题,人们提出以通用DHT服务的形式实现资源信息的发布和查询功能[4−6].OpenDHT提出在基础设施节点上运行DHT服务,应用程序无须关心DHT的部署问题,只需在基础设施以外通过RPC调用相关服 务[4].SDIMS提出在大规模系统中按照组织结构把节点组织成聚合树,允许应用程序在聚合树中“安装”所需的聚合查询功能,并指定在聚合树中信息发布与查询的传播范围[5].SWORD针对网格应用的需求,提出把网格资源信息描述为多个互连的小组,允许应用程序指定所需资源的组内和组间的特性要求[6].然而,上述DHT信息服务均没有提供底层DHT架构的自适应能力,上层应用开发者必须对针对不同的系统规模和稳定性来设置DHT运行参数(如路由表大小和更新频率等),并且在系统规模和稳定性发生变化时需要重新进行设置.

针对上述资源信息服务的不足,我们以国家科技部“973”计划项目“虚拟计算环境聚合与协同机理研究”为牵引,提出在 iVCE 中构建可扩展的分布式资源信息服务 SDIRIS(scalable distributed resource information service).本文的贡献现在于:首先,提出了自适应的 DHT 底层架构 A-FissionE(adaptive FissionE),以对上层应用透明的方式适应不同系统的规模和稳定性;其次,基于 A-FissionE 底层架构,提出了高效的多属性区间搜索算法 MR-FissionE(multiple-attribute range FissionE).理论分析和模拟结果表明,SDIRIS 能够高效地实现资源信息的发布与查询功能.

本文第 1 节介绍 SDIRIS 系统结构设计.第 2 节介绍自适应 DHT 方法 A-FissionE.第 3 节介绍高效的多属性区间搜索算法 MR-FissionE.第 4 节进行模拟验证.第 5 节对当前工作进行总结.

1 SDIRIS 系统结构

本节首先简要介绍 iVCE 的基本概念和 SDIRIS 在 iVCE 中所处的位置,然后介绍 SDIRIS 的系统结构.

针对开放环境下的按需聚合与分布自治资源的自主协同问题,iVCE 提出了自主元素、虚拟共同体和虚拟执行体的概念.自主元素对网络上的多种异构资源进行抽象封装,并赋予其环境感知、自主行为决策和协同等能力,为资源访问交互提供一致的接口.虚拟共同体是指具有共同目标、遵从共同原则的自主元素集合,规定了资源聚合的范围,描述了自主元素聚合的环境.网络资源根据虚拟共同体的目标和原则,按照 Join/Adapt 语义形成自主元素,并在协同承担共同任务时动态地组成虚拟执行体.为了按照任务需求对自主元素进行有效聚合,在 iVCE 中需要组织和管理大量自主元素的特征信息.针对这一需求,本文提出 SDIRIS 资源信息服务,以 SDIRIS 作为 iVCE 虚拟共同体中的重要信息基础设施,实现自主元素特征信息的有效管理和资源的按需聚合.

如图 1 所示,SDIRIS 资源信息服务的系统结构分为 DHT 架构层、信息服务功能层和信息服务接口层.

首先,不同分布式应用的规模和稳定性各不相同,同一分布式应用的规模和稳定性也可能随着时间的推移而发生显著变化.因此,SDIRIS 在底层 DHT 架构层的设计中提出了自适应 FissionE(adaptive FissionE,简称

A-FissionE)方法.A-FissionE 能够根据系统的规模和稳定性,自适应地调整节点度数和路由表更新策略,在信息服务的维护开销和查询延迟之间加以适当的折衷.A-FissionE 在系统规模较小且稳定性较好时采用较高的节点度数和路由表更新频率,以得到较低的查询延迟;而在系统规模较大且稳定性较差时采用较低的节点度数和路由表更新频率,从而将维护开销控制在一定范围以内.

其次,为了满足上层应用的各种查询需求,SDIRIS 信息服务功能层包括信息发布、精确匹配搜索、区间搜索、聚合查询以及 Top-*k* 查询等多个功能模块,并由各模块负责具体实现不同的查询请求.目前,SDIRIS 支持精确匹配搜索和区间搜索,我们将在以后的工作中逐步支持聚合查询、Top-*k* 搜索等更复杂的搜索方式.

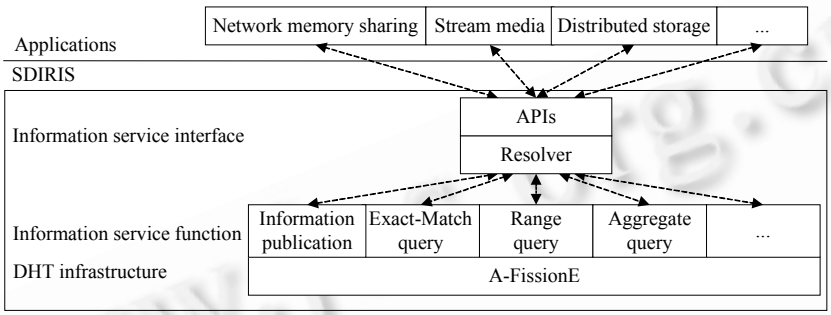

Fig.1 Architecture of SDIRIS

图 1 SDIRIS 系统结构

最后,为了在保持接口简单性的同时支持各种查询方式,SDIRIS 的资源信息服务接口层具有“窄腰(narrow waist)”的特征,通过声明性查询语言(declarative interface),支持上层应用以字符串的方式描述信息发布与查询要求.资源信息服务接口层向上层应用提供如下两类接口函数:

(1) 上层应用通过 *Publish*(*localInfo*)接口函数发布本地资源信息,其中,参数 *localInfo* 以字符串的形式对本地资源信息进行描述,例如,*localInfo*=“*attrName*1=*v*1,*attrName*2=*v*2,...”.

(2) 上层应用通过 *Lookup*(*sql*)接口函数查询网络中的资源信息,其中,参数 *sql* 以类 SQL 的方式表示资源请求节点的查询请求,例如,*sql*=“SELECT {*agg_func*(*expr*),*attr*} FROM {*commonwealth*} WHERE {*selPred*} GROUP BY {*attr*} HAVING {*havingPred*} FREQ *i* STOP IF *t*”.除了最后的 FREQ 和 STOP IF 语句以外,上面的语法与 SQL 查询语言非常类似,这里不再赘述.FREQ 语句用于指定查询的频率,表示每 *i* 秒返回一次结果;STOP IF 语句用于指定查询结束条件,可以用查询结束的时间或次数等表示.不同资源信息的变化频率差别很大,某些资源信息(如服务节点的 CPU 个数)几乎不会变化,而另一些资源信息(如服务节点的空闲内存容量)的变化则很频繁.因此,SDIRIS 通过 FREQ 和 STOP IF 语句支持应用程序显式地设定资源信息的查询频率和结束条件.在 Internet 分布式应用中,这种持续的信息流通常比单独的查询结果更有意义.

在 SDIRIS 系统结构中,资源信息服务接口层偏重于工程实现,相对较为简单,因此,下文将重点对 DHT 架构层和信息服务功能层进行详细介绍.

2 自适应 DHT 方法

在虚拟计算环境中,不同分布式应用的规模和稳定性各不相同,同一分布式应用的规模和稳定性也可能随着时间的推移而发生显著变化.在小规模、稳定性好的系统中,使用较高的节点度数和路由表更新频率能够获得较低的查询延迟,但是这种方式在大规模、稳定性差的系统中将消耗大量带宽,造成系统崩溃;反之,在大规模、稳定性差的系统中,使用较低的节点度数和路由表更新频率能够降低系统维护开销,但是这种方式在小规模、稳定性好的系统中将显著增加不必要的查询延迟.针对iVCE应用的上述特点,SDIRIS基于概率路由表[7]的思想对FissionE方法[8]进行改进,提出一种自适应DHT方法A-FissionE.如图 1 所示,在SDIRIS系统结构中,A-FissionE既是其他各种信息服务的基础架构,同时也直接向上层应用提供精确匹配搜索服务.

下面首先简要回顾 FissionE 的基本特点,然后介绍 SDIRIS 对 FissionE 的改进.

2.1 FissionE简介

分布式哈希表(DHT)是实现资源信息服务的关键技术.由于常量度数DHT方法具有维护开销小、易于配置等优点,近年来人们提出了多种常量度数的DHT方法,其中,CAN[9]的路由延迟为$O(N^{1/d})$;Koorde[10]的路由延迟为$O(\log N)$,但是具有较大的拥塞.我们在以前的工作中提出了基于Kautz图的DHT方法FissionE[8],节点度数为$O(1)$,网络直径为$O(\log N)$,拥塞特性为$(1+o(1))$.

在FissionE中,对于串$\xi=a_1a_2...a_k$,若$a_i\in\{0,1,2,...,d\}(1\le i\le k)$且$a_i\ne a_{i+1}(1\le i\le k-1)$,则称$\xi$是基底为$d$、长度为$k$的Kautz串.Kautz空间$KautzSpace(d,k)$是指所有长度为$k$、基底为$d$的Kautz串的集合,即$KautzSpace(d,k)=\{a_1a_2...a_k|a_i\in\{0,1,2,...,d\}(1\le i\le k)$且$a_i\ne a_i+1(1\le i\le k-1)\}$.对给定的整数$d$和$k$,Kautz图$K(d,k)$是一个节点出度和入度都为$d$、网络直径为$k$的有向图.

在FissionE中,每个节点的标识都是Kautz空间中的一个基底为 2 的Kautz串.初始时,FissionE中的节点组成了以 2 为基底的一个静态Kautz图(如$K(2,1)$).随着节点的不断加入或退出,FissionE中各节点标识以及邻居关系也会进行相应的调整,各节点标识的长度可能是不同的,并且随着节点的加入或退出而动态变化.设节点U 的标识为Kautz串$u_1u_2...u_k$,节点U的路由表中维护了如下两类邻居的信息:出边邻居信息,即系统中所有标识为$V=u_2u_3...u_kq_1...q_m(0\le m\le 2)$的节点是$U$的出边邻居;入边邻居信息,即系统中所有标识为$W=au_1u_2...u_i(k-2\le i\le k)$的节点是$U$的入边邻居.

2.2 自适应FissionE

与 FissionE 相比,A-FissionE 的资源命名算法、消息路由策略以及节点加入、退出时的区域拆分与合并机制没有变化.为了适应不同的系统规模和稳定性,A-FissionE 改进了 FissionE 的拓扑构造策略,并且使用了新的节点失效预测方法.

2.2.1 拓扑构造

在FissionE中,每个节点的入度为 2,出度在 1 和 4 之间,平均节点度数为 4,从而所需的维护开销较小,适合于低带宽、大规模、稳定性差的系统.然而,在高带宽、小规模、稳定性好的系统中,这种设计将导致不必要的信息查询延迟($\log_2N$,N为节点个数).与FissionE不同,A-FissionE中的节点在链路带宽允许的情况下将自动提高节点度数,在路由表中增加新的邻居信息,进而提高资源搜索性能.

A-FissionE 基于"小世界(small-world)"分布特性选择邻居节点:节点 B 与节点 A 的距离越近,B 成为 A 的邻居的可能性就越大.

定义 1(最大匹配度). 对任意两个Kautz串$\alpha=u_1u_2...u_m$和$\beta=v_1v_2...v_n$,一个度数为$i(0\le i\le\min(m,n))$的匹配是指:对任意的$j(1\le j\le i)$,有$u_{m-i+j}=v_j$.我们称i的最大取值为最大匹配度$MD(\alpha,\beta)=\mathrm{Max}(i)$.

例如,设$\alpha=10121$,$\beta=12102$,则它们之间有一个度数为 1 的匹配(1)以及一个度数为 3 的匹配(121),因此,$MD(\alpha,\beta)=3$.

定义 2(距离). 设节点A的标识为$\alpha=u_1u_2...u_m$,节点B的标识为$\beta=v_1v_2...v_n$,则节点A和B的距离

$$D(A,B)=m-1-MD(\alpha,\beta).$$

例如,$D(10121,12102)=1$,$D(12120,0210)=3$.根据定义 2,如果在 FissionE 拓扑中,节点 B 是节点 A 的出边邻居,那么 $D(A,B)=0$.

在A-FissionE中,路由表的表项符合x^{-2}分布.也就是说,如果$D(A,B)=x$,那么,节点A的路由表中包含节点B的概率$p_{A,B}$正比于x^{-2}.当$D(A,B)=0$时,$p_{A,B}=1$.文献[11]证明,通过使用符合上述small-world分布的路由表将使系统获得最小的平均路由延迟.

A-FissionE 使用与 FissionE 相同的路由策略.然而,当中间节点 B 转发目标为 k 的查询消息时,B 的下一跳节点 C 将返回一个确认消息(acknowledgement),该确认消息将包括节点 C 路由表中随机选择的一些位于 C 和 k 之间的表项.这种方式使节点 B 能够在转发查询消息的过程中学习符合 small-world 分布的新邻居.

在上层应用的查询负载较轻的情况下,如果链路上有空闲带宽,那么节点将发起一些针对随机Kautz串的主动探测消息.与前面的过程类似,节点将选择符合x^{-2}分布的新节点加到自己的路由表中.在网络带宽允许的情况下,节点将逐渐增加主动探测消息的频率f.一旦发现负载超过了链路的承受能力(例如发生了报文丢失或查询负载突增等),将立刻停止发送主动探测消息.

2.2.2 预测节点失效

在FissionE中,一旦发现有节点失效将通知所有该节点的邻居节点.然而在A-FissionE中,由于路由表中的表项是随机选择的,因此,当一个节点(例如节点B)失效时,无法知道哪些节点的路由表中包含B,从而无法发送类似的更新信息.因此,A-FissionE采用Pareto概率预测[7]的方式来判断邻居节点是否失效.

一个表项的有效性可以用节点有效的概率p来表示.A-FissionE将删除路由表中有效概率小于阈值p_{del}的所有表项.在实际系统中,节点寿命的分布通常符合Pareto分布[12].Pareto分布的生存时间概率分布函数(PDF)分布见公式(1).

$$f(x)=\frac{ab^a}{(x+b)^{a+1}} \tag{1}$$

假设邻居节点B在时刻t_0加入系统,最近一次发现节点B有效的时刻为t_1,基于公式(1),可以得到当前时刻t节点B仍然有效的条件概率,见公式(2).从公式(2)可以看出,Pareto分布是heavy-tailed,当前时间距上次确认时间$(t-t_1)$越短,已知生存时间(t_1-t_0)越长,则在当前时间t节点B仍然有效的概率p就越高.各节点可以通过路由消息和确认消息得到其他节点的生存状态(liveness).为了计算p,每个节点需要知道邻居节点的t_1和t.每个节点记录自己的t_1并在所有发送和转发的消息中包含t_1和自己加入系统的时间t_0.

$$p=\frac{1-\int_{t_0}^{t}f(x)\mathrm{d}x}{1-\int_{t_0}^{t_1}f(x)\mathrm{d}x}=\left(1-\frac{t-t_1}{t-t_0+b}\right)^a \tag{2}$$

由于A-FissionE只能通过预测的方式了解邻居节点是否有效,因此,中间节点在转发消息时必须进行超时估计.为了计算超时,A-FissionE中的节点记录了每个邻居的延迟记录,并采用TCP协议的超时计算方法[13], RTO超时的计算如公式(3)所示,其中,AVG是历史平均RTO,VAR是RTO的均方差.

$$RTO=AVG+4\times VAR \tag{3}$$

A-FissionE 能够根据系统的规模和稳定性,自适应地调整节点度数和路由表更新策略,在信息服务的维护开销和查询延迟之间进行适当的折衷.根据前面介绍的拓扑构造策略(增加路由表项)和预测节点失效方法(删除路由表项),在系统规模较大且动态性较强时,各节点的路由表在最差情况下仅包含距离为 0 的表项,此时,A-FissionE的搜索延迟为$O(\log N)$,与FissionE相同;而在系统规模较小且动态性较弱时,各节点的路由表将包含 $O(N)$个邻居节点,此时,A-FissionE 将达到全互联拓扑的搜索延迟 $O(1)$.

3 多属性区间搜索技术

本节首先介绍基于 ZK 树(Z-curve Kautz tree,简称 ZKT)实现资源命名与信息发布的方法,然后介绍基于A-FissionE 的多属性区间搜索算法 MR-FissionE.

3.1 资源命名与信息发布

我们基于PHT技术[14]提出了用于实现多维属性线性化的ZK树.设资源对象X共有m个属性$X_i(0\le i\le m-1)$,资源对象的属性值$x=\{x_i\}$表示属性$X_i=x_i$,各属性的取值范围是已知的,$x_{i(\min)}\le x_i\le x_{i(\max)}$.我们以$k$位二进制数表示一个属性的值对$X$进行标准化(normalization),见公式(4).

$$x_i'=\frac{x_i-x_{i(\min)}}{x_{i(\max)}-x_{i(\min)}}\times(2^k-1),\ 0\le x_i'\le 2^k-1 \tag{4}$$

为了便于叙述,下文中不再区分x_i和x_i',并认为各属性的取值范围均为$0\le x_i\le 2^k-1$.为了实现多属性值到资源对象名称的映射,首先需要对多属性值进行线性化(linearization).我们通过Z-curve方法[15]来实现多属性值的线

性化.

定义 3(Z映射). 设资源对象X共有m个属性$X_i=x_i(0\le i\le m-1)$,令x_i为形如$x_{i0}x_{i1}\ldots x_{ik}$的二进制串,则$Z(X)=x_{00}x_{10}\ldots x_{(m-1)0}x_{01}x_{11}\ldots x_{(m-1)1}\ldots x_{0k}x_{1k}\ldots x_{(m-1)k}$称为资源对象$X$从多维属性空间到一维空间的$Z$映射.显然,$Z(X)$的长度$D=k\times m$,$0\le Z(X)\le 2^D-1$.

ZK树中的每个节点表示一个多维空间,该节点对应一个该多维空间中所有点Z映射值的共同前缀的字符串,并使用该字符串的Kautz_Hash[8]值作为其标识.ZK树的具体形成规则如下:根节点表示资源属性的整个多值空间$x_{i(\min)}\le x_i\le x_{i(\max)}$,$0\le i\le m-1$,根节点处于第0层,其字符串和标识均为空.ZK树中的每个节点有0或2个子节点.例如,设ZK树的第h层节点A对应的字符串为L,标识为$Hash(L)$.如果节点A有两个子节点B和C,则节点B对应的字符串为$L0$,标识为$Hash(L0)$,节点C对应的字符串为$L1$,标识为$Hash(L1)$.取$i=h \bmod m$,则子节点B和C均匀划分节点A所负责的属性X_i的子区间(即节点A负责的多值空间的第i维被均匀划分成2等份,子节点B和C各负责一份).子节点B和C所负责的其他属性$X_j(j\ne i)$的子区间与A相同.

所有资源信息都通过Z映射值的前缀对应到ZK树的叶节点上.每个叶节点最多容纳N_{thresh}条信息.例如,当ZK树中处于第h层的叶节点A所对应的资源信息数超过N_{thresh}时,A将按照前面介绍的规则生成两个子叶节点B和C,并根据各资源Z映射值的前缀把所有资源信息分配给B和C,同时,节点A变为中间节点,不再具有任何信息.资源对象将以所对应的ZK树叶节点的标识作为其标识.

资源对象的命名算法(Z-Hash)如下:

```
Procedure Z-Hash (Resource X)
1.  z=Z(X);                                  //通过 Z 映射得到一维二进制串
2.  A=GetCorrespondLeaf(z);                  //查找对应字符串为 z 的前缀的叶节点
3.  L=BinaryString(A);                       //叶节点 A 对应的二进制串
4.  l=Length(L);                             //该二进制串的长度为 l
5.  n=CurrentInfoNo(A);                      //叶节点 A 当前容纳的信息数
6.  if (n+1≤N_thresh) {                          //无须分裂成两个子叶节点
7.      s=Kautz_Hash(L);                     //直接使用 FissionE 的哈希算法
8.  } else {                                 //分裂成两个子叶节点
9.      AddToLeaf(X,A)                       //X 暂时先加入节点 A
10.     B=GenerateLChild(A);                 //B 所对应的二进制串为 L0
11.     C=GenerateRChild(A);                 //C 所对应的二进制串为 L1
12.     for each Y∈Resource(A) {             //对 A 的每个资源
13.         if (Z(Y)_{l+1}==0) {             //第l+1 位为 0
14.             HandOver(Y,B,A);             //把 Y 从 A 移到 B
15.             if (Y==X) s=Kautz_Hash(L0);  //命名
16.         } else {                         //第 l+1 位为 1
17.             HandOver(Y,C,A);             //把 Y 从 A 移到 B
18.             if (Y==X) s=Kautz_Hash(L1);}}}
19. return s;                                //返回 X 的命名结果
```

设发布资源对象X的节点的Kautz标识为$ID(X)$,则资源信息的形式为$\{ID(X),x_0,x_1,\ldots,x_{m-1}\}$.该信息将通过FissionE路由机制发布到负责Kautz串Z-$Hash(X)$的Kautz空间中的节点上.

3.2 区间搜索

我们将资源对象X看作是m维空间中的一点,并将待搜索的多值空间看作m维空间中的一块连续区域$\langle x_{i(\min)},x_{i(\max)}\rangle(0\le i\le m-1)$.可以使用$X_{\min}\le X\le X_{\max}$表示搜索请求,其中,$X_{\min}=\{x_{i(\min)}\}$,$X_{\max}=\{x_{i(\max)}\}$.

设$Z_{\min}=Z(X_{\min})$,$Z_{\max}=Z(X_{\max})$,$Z_{\min}$和$Z_{\max}$的共同前缀为$Prefix(Z_{\min},Z_{\max})$.为了充分利用对多个属性值的限制

条件,减少搜索范围并加快搜索速度,在执行多属性区间搜索时,首先需要定位在ZK树中负责$Prefix(Z_{min},Z_{max})$的节点(设该节点为A).需要注意的是,该节点可能为ZK树的中间节点.

设$Prefix(Z_{min},Z_{max})=z_1z_2...z_p$的位数为$p$,则负责该前缀的ZKT节点所对应的字符串$L$共有$(p+1)$种可能,分别为$z_1,z_1z_2,\ldots,z_1z_2...z_p$.设系统中共有$N$个节点,令$s=Kautz_Hash(L)$,对负责每个Kautz串$s$的Kautz节点发送请求,则通过二分搜索最多需要$\log_2 p\leq\log\log N$步测试即可定位到节点$A$.如果带宽允许每步最多测试$2^q$个ZKT节点,那么最多需要$(\log_2 p)/q\leq(\log\log N)/q$步即可定位到节点$A$.

定位到节点A后,如果节点A为叶节点,那么把区间搜索请求发送给节点A即可;否则,需要从节点A开始沿ZK树向下进行剪枝搜索.多属性区间搜索算法(MR-FissionE)如下:

```
Procedure MR-FissionE (Value X_min,Value X_max)
1.  Z_min=Z(X_min); Z_max=Z(X_max);          //进行Z映射
2.  Z_pre=Prefix(Z_min,Z_max);               //得到最大共同前缀
3.  p=Length(Z_pre);                         //得到共同前缀长度
4.  for (i=0; i<=p; i++) {
5.      s=Kautz_Hash(Z_pre,i);               //得到对应Kautz串
6.      if ((A=Test(s))!=NULL) break;}       //探测该 ZKT 节点是否存在
7.  PruneSearch(A,X_min,X_max)               //在节点A进行剪枝搜索
8.  return 0;
```

定义 4(多维空间上的序 <). 对m维空间上的任意两个多属性值$X=\{x_i\},Y=\{y_i\},X<Y$当且仅当存在$i(0\leq j\leq m-1)$满足:对任意的$0\leq j<i,x_j=y_j$并且$x_i<y_i$[16].

定义 5(维序映射). 设F是从m维空间D到一维空间E的一个映射.F是一个维序映射,当且仅当对定义域D中任意两个元素X和Y,若$X<Y$,则$F(X)=F(Y)$或$F(X)<F(Y)$.

定义 6(空间维序映射). 设F是从m维空间D到一维空间E的一个维序映射.F是一个空间维序映射,当且仅当对多值空间D中的任意连续区域$\langle X_{min},X_{max}\rangle$,通过映射$F$在$E$上的值域为$[F(X_{min}),F(X_{max})]$.

定理 1. 当$m=1$时,Z映射是从多维属性空间到一维空间$[0,2^{km}-1]$的空间维序映射;当$m>1$时,Z映射是从多维属性空间到一维空间$[0,2^{km}-1]$的维序映射,但不是空间维序映射.

证明:当$m=1$时,$X=x,Z$映射$Z(X)=X$.若$X<Y$,则$x<y$,即$Z(X)<Z(Y)$,因此,该映射为维序映射.同理,任给$X_{min}\leq X\leq X_{max}$,即$x_{min}\leq x\leq x_{max}$,有$Z(X_{min})\leq Z(X)\leq Z(X_{max})$.易知,此时$Z$映射为双射,进而$Z$映射为空间维序映射.

当$m>1$时,设$X=\{x_i\},Y=\{y_i\}(0\leq i\leq m-1)$.若$X<Y$,则存在$i$满足:对任意的$0\leq j<i,x_j=y_j$并且$x_i<y_i$.令$x_i=x_{i0}x_{i1}\ldots x_{ik}$,$y_i=y_{i0}y_{i1}\ldots y_{ik}$,则$x_i<y_i$等价于存在$g$满足:对任意的$0\leq h<g,x_{jh}=y_{jh}$并且$x_{ig}<y_{ig}$.又$Z(X)=x_{00}x_{10}\ldots x_{(i-1)g}x_{ig}x_{(i+1)g}\ldots x_{(m-1)k}$,$Z(Y)=y_{00}y_{10}\ldots y_{(i-1)g}y_{ig}y_{(i+1)g}\ldots y_{(m-1)k}$,所以有$Z(X)<Z(Y)$.因此$Z$映射为维序映射.

设$m=2$,且多值空间为$\langle X_{min},X_{max}\rangle,X_{min}=(0,0),X_{max}=(2,1)$.显然,$X'=(0,2)$不在该多值空间之内.但是,$Z(X')$为$2k$位二进制数$Z(X')=00\ldots0100$,且小于二进制数$Z(X_{max})=00\ldots01001$,即$Z(X_{min})\leq Z(X')\leq Z(X_{max})$.因此,$\langle X_{min},X_{max}\rangle$通过$Z$映射的值域不是$[Z(X_{min}),Z(X_{max})]$,即$m=2$时,$Z$映射不是空间维序映射.

当$m>2$时的证明过程与此基本类似,这里不再赘述. □

由定理1可知,当资源属性个数大于1时,如果$X_{min}\leq X\leq X_{max}$,那么,$X$通过$Z$映射得到的值$Z(X)$的值域并不一定为$[Z(X_{min}),Z(X_{max})]$,而很可能只是$[Z(X_{min}),Z(X_{max})]$的一个子集.因此,在节点$A$沿ZK树向下搜索时,可以采用如下的剪枝搜索方法:

(1) 在某中间节点G,检查其左子节点B负责的区域与目标查询区域是否有重叠;

(2) 如果有重叠,则把范围查询请求发送给节点B对应的Kautz节点;否则,说明节点B(及其所负责的子ZK树)无须进一步搜索;

(3) 同样的检查并行地在节点G的右子节点C进行.

设节点A所负责的子树高度为d,由于ZK树的高度为$\log_2 N$,因此,上述剪枝搜索方法最多需要搜索$d\leq\log_2 N$

步.

根据上面的分析,在最差情况下(每个节点的路由表中仅有距离为 0 的邻居信息,并且没有并行搜索),通过二分搜索最多需要$\log\log N$步测试即可定位到节点A,因此,最多需要$\log N+\log\log N$步即可完成一次多属性区间搜索.考虑到ZK树中的每一步搜索对应下层A-FissionE的一次给定目标Kautz串的路由,因此,MR-FissionE的搜索延迟小于$\log^2 N+\log N\log\log N$.

4 模拟验证

我们在FissionE模拟器[8]上模拟了A-FissionE精确匹配搜索算法.文献[8]证明,与其他DHT算法相比,FissionE具有最优的性能,因此,我们仅对A-FissionE和FissionE进行了比较.由于A-FissionE的资源命名方法、消息路由策略以及节点加入、退出时的区域拆分、合并机制与FissionE完全相同,因此,我们仅对A-FissionE算法的平均搜索延迟和节点度数分布进行了模拟.

由于目前 FissionE 模拟器还不支持对不同振荡频率(churn rate)的分布式系统的模拟,因此,我们在模拟中进行了简化,假设路由表中每条记录的维护需要 5 字节/秒的带宽,我们将在不同带宽限制下模拟 A-FissionE 算法的性能.对 FissionE 模拟器的改进将是我们下一步的主要工作之一.

我们首先评估了 A-FissionE 的平均路由延迟.实验中模拟的节点规模为 4K,各节点的带宽限制从 10 字节每秒到 20K 字节/秒.对每种带宽限制下的 P2P 网络,进行 5 000 次模拟,每次模拟随机选择两个节点进行路由,然后计算 5 000 次路由的平均路由延迟,模拟结果如图 2 所示.从图中可以看出,与 FissionE 相比,A-FissionE 能够随着带宽的增加而自适应地增加路由表项,从而获得更低的延迟.

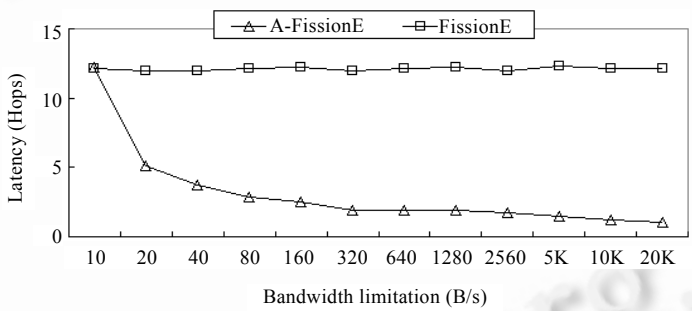

Fig.2 Average search latency of A-FissionE algorithm under different bandwidth limitations

图 2 不同带宽限制下 A-FissionE 算法的平均搜索延迟

然后,我们评估了 A-FissionE 方法的节点度数(出度)分布.实验中模拟的节点规模为 4K,各节点的带宽限制为 20 字节/秒和 40 字节/秒,其节点度数分布如图 3 所示.从图中可以看出,A-FissionE 能够自适应地随着带宽的变化调整节点度数.

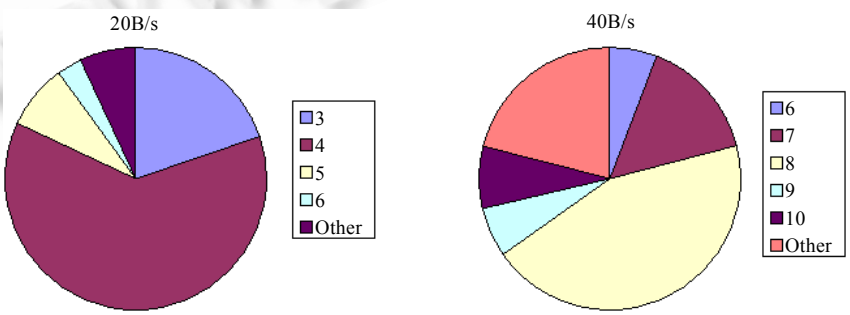

Fig.3 Distribution of node degree under different bandwidth limitations

图 3 不同带宽限制下的节点度数分布

我们在FissionE模拟器[8]上模拟了MR-FissionE多属性区间搜索算法.由于目前还没有同类的基于可变度数DHT的多属性区间搜索算法可作比较,因此我们只对MR-FissionE算法的搜索延迟进行了评估.模拟中,属性个数m=6.各节点的带宽限制从20字节/秒到160字节/秒.节点规模从1 000变化到10 000.对于每种规模进行1 000次模拟,最终取 1 000 次模拟的平均值.每次模拟随机选取一个节点作为初始节点发起区间搜索请求.各属性值总区间大小都为 1 000,搜索请求中各属性的搜索区间从[0,1000]中随机选择,搜索区间的大小在 100~200 之间随机选择.

图4显示了不同带宽限制下的模拟结果.可以看出,MR-FissionE算法具有较好的搜索延迟特性,搜索延迟在各种节点规模和带宽限制下都小于其理论上界$\log^2 N+\log N\log\log N$.

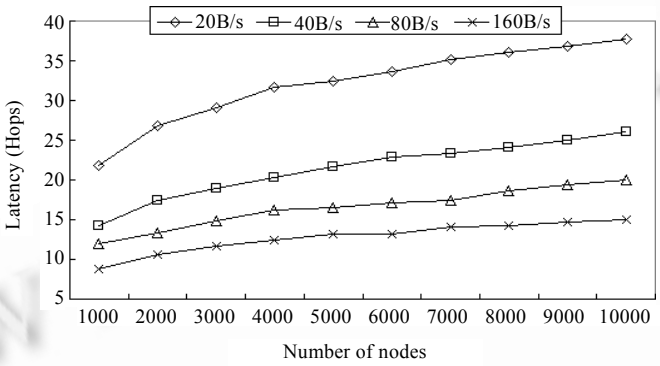

Fig.4 Average search latency of MR-FissionE algorithm under different bandwidth limitations

图 4 不同带宽限制下 MR-FissionE 算法的平均搜索延迟

5 总 结

针对现有资源信息服务在通用性、易用性和自适应性等方面的不足,本文提出在 iVCE 中构建可扩展的分布式资源信息服务 SDIRIS.首先,本文提出采用自适应 DHT(A-FissionE)底层架构,以对上层应用透明的方式适应不同的系统规模和稳定性;其次,本文基于自适应 DHT 提出高效的多属性区间搜索算法 MR-FissionE.理论分析和模拟结果表明,SDIRIS 能够高效地实现资源信息的发布与查询功能.

SDIRIS 下一步将在 PlanetLab 平台上运行与测试.聚合查询、Top-k 查询等复杂查询功能的实现以及 FissionE 模拟器的改进,也将是 SDIRIS 未来的主要工作.

References:

[1] Lu XC, Wang HM, Wang J. Virtual computing environment (IVCE): Concept and architecture. Science in China (Series E), 2006,36(10):1081−1099 (in Chinese with English abstract).

[2] Chu R, Xiao N, Zhuang YZ, Liu YH, Lu XC. A distributed paging RAM grid system for wide-area memory sharing. In: Proc. of the IPDPS 2006. Toronto: X-CD Technologies Inc., 2006. 88.

[3] Zhang YM, Li DS, Chu R, Xiao N, Lu XC. PIBUS: A network memory-based peer-to-peer IO buffering service. In: Proc. of the IFIP Networking 2007. Atlanta: Springer-Verlag, 2007. 1237−1240.

[4] Rhea S, Godfrey B, Karp B, Kubiatowicz J, Ratnasamy S, Shenker S, Stoica I, Yu H. OpenDHT: A public DHT service and its uses. In: Proc. of the ACM SIGCOMM 2005. Philadelphia: ACM Press, 2005. 73−84.

[5] Yalagandula P, Dahlin M. A scalable distributed information management system. In: Proc. of the ACM SIGCOMM 2004. Portland: ACM Press, 2004. 379−390.

[6] Oppenheimer D, Albrecht J, Patterson D, Vahdat A. Design and implementation tradeoffs for wide-area resource discovery. In: Proc. of the HPDC 2005. Washington: IEEE Computer Society, 2005. 113−124.

[7] Li JY, Stribling J, Morris R, Kaashoek FM. Bandwidth-Efficient management of DHT routing tables. In: Proc. of the NSDI 2005.

2005. 99−114.

[8] Li DS, Lu XC, Wu J. FISSIONE: A scalable constant degree and low congestion DHT scheme based on Kautz graphs. In: Proc. of the IEEE INFOCOM 2005. Miami: IEEE Press, 2005. 1677−1688.

[9] Ratnasamy S, Francis P, Handley M, Karp R, Shenker S. A scalable content-addressable network. In: Proc. of the ACM SIGCOMM 2001. San Diego: ACM Press, 2001. 161−172.

[10] Kaashoek FM, Karger DR. Koorde: A simple degree-optimal distributed hash table. In: Proc. of the IPTPS 2003. Berkeley: Springer-Verlag, 2003. 98−107.

[11] Kleinberg J. The small-world phenomenon: An algorithmic perspective. In: Proc. of the STOC 2000. Portland: ACM Press, 2000. 820−828.

[12] Godfrey BP, Shenker S, Stoica I. Minimizing churn in distributed systems. In: Proc. of the ACM SIGCOMM 2006. Pisa: ACM Press, 2006. 147−158.

[13] Rhea S, Geels D, Roscoe T, Kubiatowicz J. Handling churn in a DHT. In: Proc. of the USENIX Annual Technical Conf. Boston: USENIX Association Press, 2004. 127−140.

[14] Chawathe Y, Ramabhadran S, Ratnasamy S, LaMarca A, Shenker S, Hellerstein J. A case study in building layered DHT applications. In: Proc. of the SIGCOMM 2005. Philadelphia: ACM Press, 2005. 97−108.

[15] Jagadish HV. Linear clustering of objects with multiple attributes. In: Proc. of the ACM SIGMOD 1990. Atlantic City: ACM Press, 1990. 332−342.

[16] Li DS. Research on peer-to-peer resource location in large-scale distributed systems [Ph.D. Thesis]. Changsha: National University of Defense Technology, 2005 (in Chinese with English abstract).

附中文参考文献:

[1] 卢锡城,王怀民,王戟.虚拟计算环境 iVCE:概念与体系结构.中国科学(E 辑),2006,36(10):1081−1099.

[16] 李东升.基于对等模式的资源定位技术研究[博士学位论文].长沙:国防科学技术大学,2005.

张一鸣(1978－),男,山东济南人,博士生,主要研究领域为 P2P 资源定位技术,高性能并行分布处理技术.

卢锡城(1946－),男,教授,博士生导师,中国工程院院士,CCF 高级会员,主要研究领域为分布处理技术,网络技术.

李东升(1978－),男,博士,主要研究领域为计算机网络,P2P 资源定位技术.