# 带智能预测缓存的物联网标识解析器*

杜源峰[1,2,3], 孔 宁[1,3], 田 野[1,3], 毛 伟[1,3]

(1. 中国科学院 计算机网络信息中心, 北京 100190; 2. 中国科学院大学, 北京 100049; 3. 中国互联网络信息中心, 北京 100190)

**摘 要**: 针对现有物联网标识解析存在的性能缺陷,提出一种带智能预测缓存的物联网标识解析器,以 DNS 为基础,支持各种物联网标识编码类型的解析,并具有智能预测的缓存功能。预测时采用关联规则方法对日志数据进行挖掘,根据当前的查询请求预测下一个可能解析的标识,提前解析,缓存解析结果。同时结合 TTL 策略以及 LRU 策略作为缓存置换方法,具有高效性和时效性。实验表明,该解析器具有较高的命中率和较小的响应时间,且当缓存大小为 62 时获得最优性能。

**关键词**: 物联网; 标识解析器; 预测; 缓存

**中图分类号**: TP393 **文献标志码**: A **文章编号**: 1001-3695(2013)09-2819-04

doi:10.3969/j.issn.1001-3695.2013.09.066

# IoT identifier resolver with intelligent predictive cache

DU Yuan-feng[1,2,3], KONG Ning[1,3], TIAN Ye[1,3], MAO Wei[1,3]

(1. *Computer Network Information Center, Chinese Academy of Sciences, Beijing* 100190, *China*; 2. *University of Chinese Academy of Sciences, Beijing* 100049, *China*; 3. *China Internet Network Information Center, Beijing* 100190, *China*)

**Abstract**: In order to improve the performance of the traditional IoT identifier resolving methods, this paper proposed an IoT identifier resolver with intelligent predictive cache. The resolver was based on DNS, and supported multi-coding resolving and had a cache with the function of intelligent prediction. Mining the log data of the resolver with association rules, the resolver could predict the identifier to resolve based on the current request and pre-resolve and cache the coming requested code. The replace strategy of the resolver combined the TTL strategy with the LRU strategy at the same time, which provided with high performance and timeliness. The experimental results show that the resolver has a higher hit rate and quicker response and get the best performance when setting the cache size as 62.

**Key words**: IoT; identifier resolver; prediction; cache

物联网这个概念最早由 MIT Auto-ID 中心的 Ashton 教授于 1999 年提出,用于描述在供应链中使用 RFID 标签存储物品的编码信息。之后,物联网在各国得到大力发展。

在物联网中,每一个物品有一个唯一的物品标识,通过现有网络基础设施可以将物品连接起来,实现对物品的定位、追踪、监控和管理。互联网通过域名解析实现资源的定位,与之相似,物联网也有资源解析的需求。随着加入物联网资源数量的几何倍增,物联网标识的解析日益成为严重的问题。

目前比较流行的资源解析方案是 EPCglobal 提出的 ONS (object name service,对象名字服务)解析体系[1],在这个体系中每个物品用 EPC(electronic product code,电子产品编码)编码唯一标识。ONS 解析器实现了 EPC 编码到存储该物品信息的服务器地址之间的映射。

物联网中物品编码标准有很多种,每种编码标准的格式都不相同,即使是同一编码标准也存在许多不同的编码格式,ONS 解析器只支持 EPC 编码解析,因此无法满足物联网兼容多编码的解析需求。

ONS 是基于 DNS 实现的,底层缓存采用的是 DNS 的缓存机制。然而物联网的解析与互联网存在明显的差异。例如,互联网中解析的对象是域名,一个用户会经常访问同一个域名下的不同资源,解析器需要频繁解析同一个域名,采用缓存可以提高解析器的性能;但是在物联网中,解析的对象是物品编码,用户经常解析不同物品的编码,而且在一定时间段内一般不会重复查询同一个物品编码,因此物联网资源的标识解析具有其特殊性,导致物联网不能照搬互联网的缓存模式。由此可见,ONS 使用的 DNS 缓存方法无法满足物联网的特殊需求。

本文针对物联网标识解析的特殊性需求,提出了一种带智能预测缓存的物联网标识解析器。

## 1 相关工作

### 1.1 物联网名字解析服务

物联网的标识解析服务已经有很多相关研究成果[2~4],其中 EPCglobal 提出的 ONS[1] 是当前最为流行的名字解析服务,ONS 利用 DNS 现有的域名解析器,把物品的信息服务器地址信息存在 DNS 的 NAPTR 记录中,ONS 把 EPC 编码转换成域名,实现 EPC 编码向资源地址的映射。

然而 ONS 无法兼容多编码。Kong[2] 和 Liu[3] 等人针对这

**收稿日期**: 2012-12-31; **修回日期**: 2013-02-28 **基金项目**: 一种带智能缓存模块的物联网标识解析方法(专利号 201210328283.3)

**作者简介**:杜源峰(1988-),男,广东汕头人,硕士研究生,主要研究方向为资源地址解析、数据挖掘(duyuanfeng@cnnic.cn);孔宁(1980-),男,副研究员,博士研究生,主要研究方向为资源地址解析、DNS、物联网;田野(1979-),男,副研究员,博士研究生,主要研究方向为移动网络安全、物联网;毛伟(1968-),男,研究员,博导,博士研究生,主要研究方向为下一代互联网、网络安全、资源地址解析.

个问题分别提出了各自的解决方案。Kong 等人提出一个通用分层模型[2],该模型扩展了互联网的分层迭代模型,基于这个通用分层模型提出了一个支持多编码的通用资源解析系统。该系统通过把上一层的地址解析结果作为下一层的输入,把原始的编码转换成该层解析器可以解析的格式作为该层解析器的输入,解析得到该层的解析结果,通过不断地迭代实现多编码解析。

Liu 等人提出一个基于物联网名字服务的资源管理模型[3],采用两段式编码结构解决多编码解析问题。其中:SID(standard identifier)用于唯一标识物品编码所属的物品编码标准;RID(resource identifier)用于标识一个资源,RID 可以是各种编码标准的编码。通过先解析 SID 找到对应 RID 采用的编码标准,然后用对应的编码标准对 RID 进行解析。为了解决 SID 频繁解析造成集中式解析带来的根服务器负载过重问题,Liu 等人采用 DHT 方式获取 SID,有效解决了根服务器负载过重的问题。

### 1.2 缓存

Kong 和 Liu 等人提出的方案解决了 ONS 中的多编码兼容问题,然而缺乏对缓存的考量。缓存是把之前用户的请求结果暂时存于内存中,当用户下次再次发起相同的请求时,如果缓存中有之前用户请求的记录,并且该记录还有效,那么就把缓存中的记录直接返回给用户,减少用户等待结果的时间,提高查询解析的效率。已经有许多国内外的学者在 DNS 缓存和 Web 缓存方面进行了很多研究。本文在对比 DNS 缓存和 Web 缓存的基础上,提出了物联网标识解析器智能预测缓存的设计。

关于 DNS 缓存的性能已经有很多学者作了研究[4~8],Cohen 等人[4]提出了把缓存中过期的缓存重新刷新请求的方法,通过主动地把缓存中已经过期的记录重新请求,达到缓存更新的目的,但这个方法会增加网络负载,同时也增加了服务器的负担,因为这个刷新过程会涉及到很多与请求相关的服务器。

为了避免增加网络负担,Jang 等人[5]提出一种方法,当缓存过期时,如果过期记录对应的权威记录没有过期,则当有其他查询同一权威记录的查询时把该过期的缓存记录也搭载上一起查询。这种方法可以减少查询时间,同时减少网络的负载,但是必须在当前 DNS 协议基础上进行小的改动,不利于推广。

相比于 DNS 缓存只是刷新过期的缓存,Web 缓存加入了预测的机制,可以通过学习获得网页的访问模式,根据这些模式预取相关的网页文件对象[9~12]。Yang 等人[9]提出用挖掘 Web 日志得到的关联规则改进 GDSF[13]缓存和预取策略,有效地提高了 Web 访问的性能。Bonchi 等人[11]提出用关联规则和决策树方法扩展传统的 LRU 策略预测缓存,实验结果表明缓存命中率比 LRU 策略高。

## 2 带智能预测缓存的物联网标识解析器

如前所述,用户在一定的时间段内一般不会重复解析同一个物品编码,因此只是简单缓存已经解析过的物品编码对应的资源地址信息不能有效地实现缓存的目的。

现有的物联网解析体系并没有针对物联网特性的有效缓存机制,导致解析效率不高。因此,本文在参考对比 DNS 缓存和 Web 缓存,以及 Kong 和 Liu 等人的多编码解析工作基础上,在物联网解析体系中引入了带预测机制的智能缓存。

鉴于物联网解析的特殊性,本文通过研究用户的历史解析请求,挖掘出用户解析习惯,根据用户当前的解析请求和用户的习惯,预测用户下一次可能解析的物品编码,提前解析用户要请求解析的编码,加快解析器的响应速度,提高解析效率。

### 2.1 解析器结构设计

基于名字服务的物联网应用体系如图 1 所示。客户端的应用程序把物品编码查询解析请求发给解析器,解析器对编码进行必要的转换[2,3],转换成 DNS 可以识别的域名,然后解析器用该域名与服务器进行交互查询,把编码的查询解析结果返回给应用程序,应用程序收到解析结果后与物品相关的应用连接。解析器同时会对每次查询进行日志记录,方便以后的日志查询和利用。解析器的架构如图 2 所示,包含编码查询、智能缓存、通用查询、配置、日志五大模块,各个模块的功能如下所示:

a)编码查询模块。负责编码的转换和查询,会调用其他三个模块的功能。编码转换过程包括:

(a)把 SID 转换成 SID 域名;

(b)通过 DNS 查询 SID 域名,获得转换 RID 的正则表达式;

(c)用正则表达式转换 RID,把 RID 转换成 RID 域名;

(d)通过 DNS 查询 RID 域名,得到资源的地址信息。

通过 SID 可以根据不同的编码标准转换各种编码,实现兼容多编码。

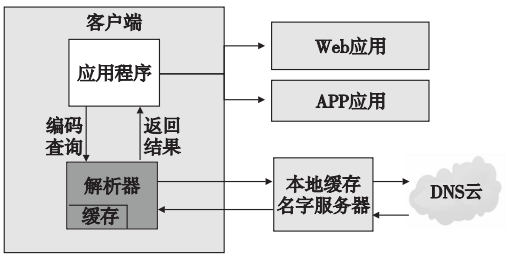


**图1 基于名字服务的物联网应用体系**

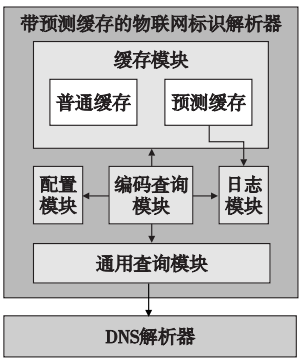


**图2 解析器架构**

b)智能缓存模块。分为普通缓存和预测缓存。

(a)普通缓存。解析器在客户端内存中开辟一段空间,保存应用程序每次请求查询的结果,当应用程序再次请求相同的编码解析时直接从缓存中返回结果,不用重复原来的转换域名操作和 DNS 查询。

(b)智能预测缓存。采用数据挖掘算法,对解析器生成的日志进行挖掘,分析应用程序使用该解析器的习惯,得到应用程序使用解析器的行为模式,预测应用程序下一个可能查询的物品编码,提前解析该编码,把结果放入缓存,当应用程序发来该编码的解析请求时可直接从缓存中取得结果。

c)通用查询模块。负责与 DNS 解析器交互的通用查询。

d)配置模块。记录解析器一些必要的配置,解析器启动时可以从该模块中获取配置信息,如转换 SID 需要的正则表达式等。

e)日志模块。记录应用程序使用解析器的解析历史数据,预测缓存通过挖掘日志得到用户的使用习惯。

### 2.2　基于关联规则的预测算法

本文采用关联规则算法[14]对解析器的日志进行挖掘,预测出解析器下一个可能解析的物品编码。

设集合 $I=\{i_1,i_2,\cdots,i_m\}$,其中的元素称为项(item)。记 $D$ 为事务(transaction)$T$ 的集合,其中 $T$ 为项的集合,且 $T\subseteq I$。

关联规则表示为 $A\Rightarrow B[S,C]$,其中 $A\subset I,B\subset I$,且 $A\cap B=\varnothing$;$S$ 是该规则的支持度,表示为包含 $A$ 和 $B$ 的事务占总事务数的比例,即

$$S(A\Rightarrow B)=\frac{|\{T:A\cup B\subseteq T,T\in D\}|}{|D|} \tag{1}$$

表示规则中所有项同时出现在一个事务中的概率;$C$ 为该规则的置信度,表示为包含 $A$ 和 $B$ 的事务数占包含 $A$ 的事务数的比例,即

$$C(A\Rightarrow B)=\frac{|\{T:A\cup B\subseteq T,T\in D\}|}{|\{T:A\subseteq T,T\in D\}|} \tag{2}$$

表示一个事务中如果 $A$ 中的项出现,$B$ 中的项也出现的条件概率。通常会给支持度和置信度设置一个最小的阈值,只有支持度和置信度超过这两个阈值的规则才能作为强规则被使用。

在物联网标识解析器的关联规则中,项对应用户查询的物品编码,事务对应一个用户查询会话,即一段连续时间间隔内的用户查询的编码集合。

通过使用关联规则算法学习解析器的日志数据,可以得到一个类似 $A\Rightarrow B$ 的关联规则集合,其中 $A$ 和 $B$ 分别表示物品编码集合,表示如果在一个查询会话内用户如果查询解析了 $A$ 集合内的物品编码,那么解析器预测在该会话中用户下一次可能会解析 $B$ 集合内的物品编码,一般 $B$ 集合包含一个编码。

### 2.3　缓存工作流程

该物联网标识解析器的缓存分为两部分,一部分为普通缓存,另一部分为预测缓存。普通缓存存放应用程序每次查询解析的结果,预测缓存存放预测的结果。

#### 2.3.1　普通缓存工作流程

普通缓存的工作流程如图 3 所示,当应用程序发起一个编码查询请求,解析器首先检查普通缓存中是否存在该记录。如果不存在则到预测缓存中查找,如果命中则把记录从预测缓存转移到普通缓存,否则进行正常的编码转换和 DNS 查询解析,把结果存入普通缓存。

#### 2.3.2　普通缓存置换策略

在普通缓存的工作流程中,记录从预测缓存移动到普通缓存以及正常解析后把解析结果存入普通缓存的这两个步骤可能遇到缓存空间已满,需要置换缓存中其他记录的情况。

与 Web 缓存不同,Web 缓存置换时不需要考虑记录的时效性,可以直接使用 LRU(least recently used)策略。而本文物联网标识解析器是基于 DNS 架构,每条记录有一个 TTL(time to live)值,当记录缓存的时间超过该 TTL 值,则记录失效,需要重新解析该记录。因此不能直接使用 LRU 策略,还需要考虑记录的 TTL 值。

在该带智能预测缓存的物联网标识解析器中,当需要置换缓存记录时,优先置换 TTL 已经过期的记录。如果存在多条 TTL 过期的记录,则根据 LRU 策略,置换最近最少访问的记录。如果缓存中所有记录的 TTL 均未过期,同样根据 LRU 策略,把最近最少访问的记录置换掉。

#### 2.3.3　预测缓存的工作流程

预测缓存采用 FIFO 队列实现,在队列的尾部插入新记录,在队列的头部删除记录。

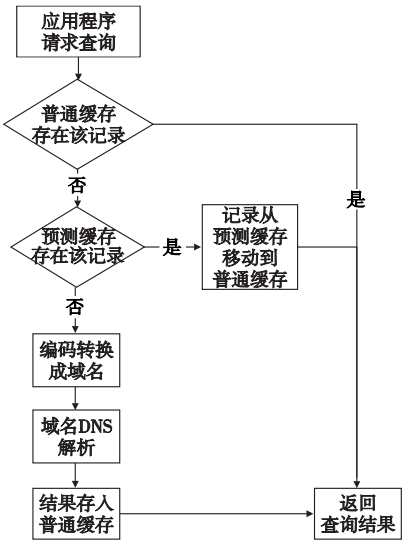


**图3　普通缓存工作流程**

缓存预测过程如图 4 所示,先用关联规则算法对解析器的日志进行分析,得出关联规则集合,形如 $A\Rightarrow B$,表示在一次查询会话中,如果集合 $A$ 中的编码被解析了,那么可以预测 $B$ 中的编码将可能被解析。

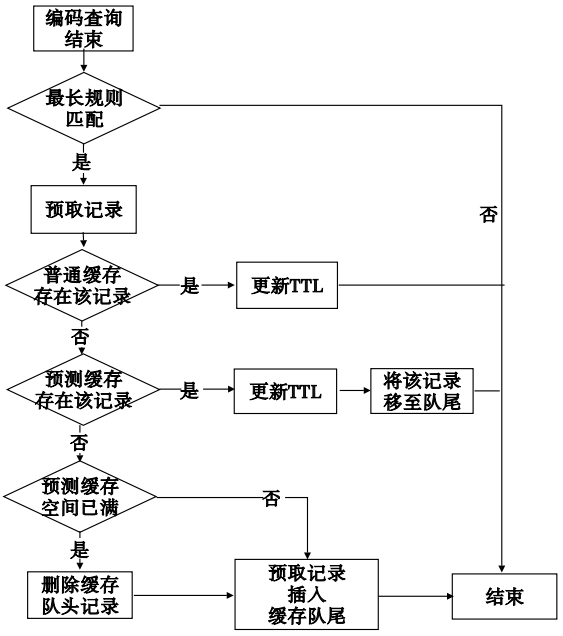


**图4　预测缓存工作流程**

当用户完成一个物品编码查询时,如果当前会话内的最长项集在关联规则集合中存在相应的记录,则取出该规则右边的编码(即 $B$),预先解析右边的编码。如果普通缓存或者预测缓存中存在预取的编码,则更新它们的 TTL 值。对于预测缓存,还要把该记录移至队尾,否则把该记录插入到预测缓存的队尾,如果缓存空间已满,还要删除队头的记录。

## 3　实验设计及结果分析

### 3.1　实验设计

为了验证预测缓存的有效性,采用用户解析日志数据进行实验,实验数据包含 551 个用户会话,共 2 705 个解析请求,平均每个会话包含 5 个解析请求,会话之间的时间间隔为 16 s。

实验主要从命中率和响应时间两个方面来验证算法的性能。

### 3.2 命中率

通过设置不同的缓存大小,分别对比默认使用 DNS 缓存解析器和带智能预测缓存解析器的缓存命中率。

默认使用 DNS 缓存解析器的缓存命中率随缓存大小变化的情况如图 5 所示。从实验结果可以看出,缓存命中率随缓存大小的变化成 log 函数形状,当缓存大小超过 30(即存放 30 条记录)时命中率变化趋于平缓,保持在 26% 左右,处于基本不变的状态,这也验证了文献[10]的结论。

由于当缓存大小超过 30 时,默认使用 DNS 缓存解析器的缓存命中率基本不变,现增加智能预测缓存,把普通缓存大小设为 30,其缓存命中率随预测缓存大小的变化情况如图 6 所示。从实验结果可以看出,缓存命中率随缓存大小的变化成 log 函数形状,当缓存大小超过 32 时命中率变化趋于平缓,保持在 49.37% 左右,处于基本不变的状态;同时可以看出,带智能预测缓存解析器的缓存命中率比默认使用 DNS 缓存解析器的要高将近一倍。

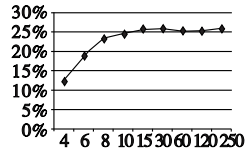


图5 默认使用DNS缓存解析器的缓存命中率

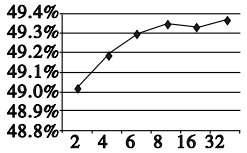


图6 带智能预测缓存解析器的缓存命中率

现对两者的命中率进行分析,缓存命中率的计算公式如下:

$$\text{HitRate} = \frac{\text{Count}_{\text{hit}}}{\text{Count}_{\text{all}}} \times 100\% \tag{3}$$

其中:$\text{Count}_{\text{all}}$表示所有的查询解析请求的数量,$\text{Count}_{\text{hit}}$表示所有查询解析请求中在缓存命中的数量。

用 NResolver(normal resolver)表示默认使用 DNS 缓存的解析器;用 IResolver(intelligent resolver)表示带智能预测缓存的解析器;用 $\text{Count}_{n\text{-hit}}$表示 NResolver 的命中记录数量;$\text{Count}_{n\text{-}i\text{-hit}}$表示 IResolver 中普通缓存的命中记录数量,$\text{Count}_{i\text{-}i\text{-hit}}$表示 IResolver 中预测缓存的命中记录数量。

实验结果表明,当缓存大小 size ≥ 30 时,NResolver 的 $\text{Count}_{\text{hit}}$基本不变;当 size ≥ 62 时,IResolver 的 $\text{Count}_{\text{hit}}$基本不变。分别令 NResolver 的缓存大小为 size = $m + n$,令 IResolver 的普通缓存大小为 size = $m$,预测缓存大小为 size = $n$,其中 $m$ = 30,$n$ = 32,则 IResolver 缓存的总大小为 size = $m + n$ = 62,与 NResolver 相同。

因为当 size ≥ $m$ 时,普通缓存的 $\text{Count}_{\text{hit}}$基本不会变,所以 $\text{Count}_{n\text{-hit}} = \text{Count}_{n\text{-}i\text{-hit}}$。而 IResolver 具有智能预测功能,故 $\text{Count}_{i\text{-}i\text{-hit}} \geqslant 0$,分别代入式(3)可得

$$\frac{\text{Count}_{n\text{-}i\text{-hit}} + \text{Count}_{i\text{-}i\text{-hit}}}{\text{Count}_{\text{all}}} \geqslant \frac{\text{Count}_{n\text{-hit}}}{\text{Count}_{\text{all}}}$$

综上所述,当缓存大小超过 62 时,带智能预测缓存解析器的命中率比默认使用 DNS 缓存解析器的命中率高。且随着缓存大小的增加,缓存命中率成 log 函数形状,即当缓存大小超过一定阈值(实验表明为 62)时,缓存命中率基本不变。

### 3.3 响应时间

由命中率实验可知,当缓存大小为 62 时,两种解析器的缓存命中率都达到最优,且带智能预测缓存解析器的命中率比默认使用 DNS 缓存解析器的高。现对比缓存大小在最优临界值时两者的请求响应时间,如表 1 所示。

表 1 两种解析器缓存大小为 62 时的响应时间/ms

| 类型 | 命中时响应时间 | 平均响应时间 |
|---|---|---|
| 不带预测 | 0 | 545.92 |
| 带预测 | 0 | 126.28 |

从表 1 结果可以看出,当缓存命中时,解析请求从缓存直接取得结果,读取缓存的时间很短,响应时间几乎为 0;同时从表中可以看出,带智能预测缓存解析器的响应时间比不带预测的快 3.3 倍。现对两者的响应时间进行分析,默认使用 DNS 缓存解析器的响应时间计算公式如下:

$$T_n = T_{\text{trans}} + \text{hitRate}_n \times t_{\text{cache}} + (1 - \text{hitRate}_n) \times t_{\text{dns}} \tag{4}$$

其中:$T_{\text{trans}}$表示转换编码需要的时间;$\text{hitRate}_n$ 表示默认使用 DNS 缓存解析器的命中率;$t_{\text{cache}}$表示读取缓存需要的时间;$t_{\text{dns}}$表示进行 DNS 查询需要的时间。式(4)第二项表示 DNS 缓存命中时需要耗费的时间,第三项表示缓存没有命中时需要耗费的时间。

带预测缓存的解析器的响应时间计算公式如下:

$$T_i = \text{hitRate}_i \times t_{\text{cache}} + (1 - \text{hitRate}_i) \times T_n \tag{5}$$

其中:$\text{hitRate}_i$ 表示带预测缓存的解析器的命中率;$t_{\text{cache}}$表示读取缓存需要的时间;$T_n$ 表示缓存不命中的情况下解析需要的时间,因为不命中时需要进行编码转换和 DNS 解析。该过程和默认使用 DNS 缓存解析器的解析过程相同,故缓存不命中时需要的时间为 $T_n$。令

$$\Delta = T_i - T_n \tag{6}$$

把式(5)代入式(6)得

$$\Delta = \text{hitRate}_i \times t_{\text{cache}} - \text{hitRate}_i \times T_n \tag{7}$$

把式(4)代入式(7)得

$$\Delta = \text{hitRate}_i \times [(1 - \text{hitRate}_n) \times (t_{\text{cache}} - t_{\text{dns}}) - T_{\text{trans}}] \tag{8}$$

因为 $1 - \text{hitRate}_n > 0$,$t_{\text{cache}} < t_{\text{dns}}$,所以 $\Delta < 0$,即 $T_i < T_n$,带智能预测缓存的解析器的响应时间比不带预测的短。

另外,由于预测需要预先发送解析请求,有些预测的请求并没有真正被用户请求到,这个过程会增加额外的网络开销。但由于预测和请求解析是两个并行的过程,所以预测不会影响请求解析的响应速度。

综上所述,带智能预测缓存解析器的响应时间比默认使用 DNS 缓存解析器的响应时间短。实验结果表明,带智能预测的解析器比不带预测的快 3.3 倍。

## 4 结束语

本文介绍了用于解析物联网中物品编码的解析器,该解析器建立在 DNS 架构的基础上,可以解析多种编码格式,同时带有智能预测缓存。与现有的 DNS 和 Web 缓存不同,带智能预测缓存的物联网标识解析器的缓存采用关联规则的算法进行预测,提前解析可能被查询的物品编码,同时采用结合 LRU 和 DNS 的 TTL 的置换策略。经分析证明,当本文解析器缓存大小超过一定阈值时比没带预测缓存的解析器具有更好的性能,有相对较高的命中率和更短的响应时间。

**参考文献:**

[1] EPCglobal. The EPCglobal architecture framework [EB/OL]. (2010). http://www.gs1.org/gsmp/kc/epcglobal/architecture.

[2] KONG Ning, LI Xiao-dong, YAN Bao-ping. A model supporting any product code standard for the resource addressing in the Internet of things[C]//Proc of International Workshop on Intelligent Networks and Intelligent Systems. 2008:233-238.

(下转第 2834 页)

## 4 结束语

本文首先阐述了在大规模分布式系统中处理复杂查询是将对等计算技术运用于云计算中的重要问题,然后结合云对等网络的特点,提出使用区间查询,并解释了区间查询在云计算中的优势。在比较了三种多属性区间查询的典型方法后,又分析了多属性区间查询的研究现状;接着给出了云资源的定义,简述了本算法的基本思想与方法。本算法提出用 $n$ 元组来为云资源建立索引,并选择用各个属性各区间的组合来建立索引。针对多属性组合索引个数太多,导致云资源信息更新的网络开销过大的缺点,本算法提出依据索引中属性的个数对全部索引进行归类存储。最后叙述了在索引分类存储的对等网络中如何查找云资源,还比较了索引未归类与归类后,云资源信息更新在网络中产生的信息数量。通过对比发现,归类后云资源信息更新在网络中产生的信息数量远远小于未归类的情况。仿真实验表明,本文算法在资源更新时产生的消息数目和查找资源时产生的网络开销都远远小于其他基于对等网络的多属性区间查询算法;更重要的是本文算法在网络开销上非常稳定,满足云对等网络实际需要。

**参考文献:**


[1] CAI Min, FRANK M, CHEN Jin-bo, *et al*. MAAN: a multi-attribute addressable network for grid information services[C]//Proc of the 4th International Workshop on Grid Computing. 2003:3-14.

[2] LI Dong-sheng, CAO Jian-nong, LU Xi-cheng, *et al*. Delay bounded range queries in DHT-based peer-to-peer systems[C]//Proc of the 26th IEEE International Conference on Distributed Computing Systems. Washington DC:IEEE Computer Society,2006:184-191.

[3] BHARAMBE A R, AGRAWAL M, SESHAN S. Mercury: supporting scalable multi-attribute range queries[C]//Proc of Conference on Community Architectures, Protocols & Applications. New York:ACM Press,2004:353-366.

[4] 王必晴,贺鹏. H-Chord:基于层次划分的 Chord 路由模型及算法实现[J]. 计算机工程与应用,2007,43(36):141-143.

[5] 段世惠,王劲林. 基于有限范围组播的 Chord 路由算法[J]. 计算机应用,2009,29(2):514-517.

[6] OPPENHEIMER D, ALBRECHT J, PATTERSON D, *et al*. Distributed resource discovery on planetlab with sword[C]//Proc of the 1st Workshop on Real, Large Distributed Systems. New York: ACM Press,2004.

[7] KLEIS M, LUA E K, ZHOU X. Hierarchical peer-to-peer networks using lightweight super peer topologies[C]//Proc of the 10th Symposium on Computers and Communications. Washington DC: IEEE Computer Society,2005:143-148.

[8] SHEN H, CHEN G. Cycloid: a constant-degree and lookup-efficient P2P overlay network[C]//Proc of the 18th International Parallel and Distributed Processing Symposium. Washington DC:IEEE Computer Society,2004:1-10.

[9] GUPTA I,BIRMAN K,LNGA P,*et al*. Kelips: building an efficient and stable P2P DHT through increased memory and background overhead[C]//Lecture Notes in Computer Science,Vol 2735. New York: ACM Press, 2003:160-169.

[10] GUPTA A,LISKOV B,RODRIGUES R. One-hop lookups for peer-to-peer overlays[C]//Proc of the 9th Conference on Hot Topics in Operating Systems. Berkeley:USENIX Association,2003:2.

[11] XU Ke, SONG Mei-na, ZHANG Xiao-qi, *et al*. A cloud computing platform based on P2P[C]// Proc of IEEE International Symposium on IT in Medicine & Education. Washington DC:IEEE Computer Society,2009:427-432.

[12] ZHAO Peng, HUANG Ting-lei, LIU Cai-xia, *et al*. Research of P2P architecture based on cloud computing[C]//Proc of International Conference on Intelligent Computing and Integrated Systems. Washington DC:IEEE Computer Society,2010:652-655.

[13] SOTIRIADIS S,BESSIS N,ANTONOPOULOS N. Using self-led critical friend topology based on P2P Chord algorithm for node localization within cloud communities[C]//Proc of International Conference on Intelligent Computing and Integrated Systems. Washington DC:IEEE Computer Society,2011:490-495.

[14] HUANG Li-can. Semantic P2P networks: future architecture of cloud computing[C]//Proc of the 2nd International Conference on Networking and Distributed Computing. Washington DC:IEEE Computer Society,2011:336-339.


(上接第 2822 页)


[3] LIU Yang, TIAN Ye, SHEN Shuo, *et al*. A compatible and equitable resolution service for IoT resource management[C]//Proc of the 6th IEEE International Conference on RFID. 2012.

[4] COHEN E, KAPLAN H. Proactive caching of DNS records: addressing a performance bottleneck[C]//Proc of Symposium on Applications and the Internet. 2001:85-94.

[5] JANG B, LEE D, CHON K, *et al*. DNS resolution with renewal using piggyback[J]. Journal of Communication and Networks, 2009,11(4):416-427.

[6] JUNG J, SIT E, BALAKRISHNAN H, *et al*. DNS performance and the effectiveness of caching[J]. Internet Measurement Workshop, 2002,10(5):589-603.

[7] BHATTI S N, ATKINSON R. Reducing DNS caching[C]//Proc of IEEE Conference on Computer Communications Workshops. 2011: 792-797.

[8] CHEN Xin, WANG Hai-ning, ZHANG Xiao-dong. Maintaining strong cache consistency for the domain name system[J]. IEEE Trans on Knowledge and Data Engineering,2007,19(8):1057-1071.

[9] YANG Qiang, ZHANG H H, LI Tian-yi. Mining Web logs for prediction models in WWW caching and prefetching[C]//Proc of the 7th ACM SIGKOD International Conference on Knowledge Discovery and Data Mining. 2001:473-478.

[10] YANG Qiang, ZHANG H H. Web-Log mining for predictive Web caching[J]. IEEE Trans on Knowledge and Data Engineering, 2003,15(4):1050-1053.

[11] BONCHI F, GIANNOTTI F, MANCO G, *et al*. Data mining for intelligent Web caching[C]//Proc of International Symposium on Information Technology. 2001:599-603.

[12] FU A Y, WU Xiao, FU Hao-huan. A statistics-based Web cache prediction model[C]//Proc of the 6th ACM-HK Postgraduate Research. 2005.

[13] CHERKASOVA L. Improving WWW proxies performance with greedy-dual-size-frequency caching policy, R.1[R]. [S.l.]:HP Laboratories,1998.

[14] HAN Jia-wei, PEI Jian, YIN Yi-wen. Mining frequent patterns without candidate generation[C]//Proc of ACM SIGMOD International Conference on Management of Data. 2000:1-12.