面向大数据处理的高精度多维计数布鲁姆过滤器

李 玮[1],张大方[1],黄 昆[2],谢 鲲[1]
(1.湖南大学信息科学与工程学院,湖南长沙 410082;2.中国科学院计算技术研究所,北京 100190)

摘 要: 分析了现有多维布鲁姆过滤器查询算法的工作原理和特点,针对大数据处理特点提出了一种基于双射函数的高精度多维计数布鲁姆过滤器(AMD-CBF)查询算法.AMD-CBF中元素表示和查找分两步进行,第1步将元素各属性哈希映射到各自对应的高精度计数布鲁姆过滤器(A-CBF)中;第2步将元素的所有属性通过双射函数转换为一个值来表示元素整体信息,然后将这个值哈希映射到联合计数布鲁姆过滤器中(C-CBF),完成元素整体的表示和查询确认.理论分析和仿真实验结果表明,AMD-CBF能够支持多维集合元素的高效表示和查询及删除,相比同类研究查询假阳性降低明显,查询精度大幅度提高.

关键词: 大数据处理; 多维布鲁姆过滤器; 双射函数; 高精度计数布鲁姆过滤器; 假阳性
中图分类号: TP393 **文献标识码:** A **文章编号:** 0372-2112 (2015)04-0652-06
电子学报URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2015.04.005

Accurate Multi-Dimension Counting Bloom Filter for Big Data Processing

LI Wei[1], ZHANG Da-fang[1], HUANG Kun[2], XIE Kun[1]
(1. *College of Computer Science and Electronics Engineering, Hunan University, Changsha, Hunan* 410082, *China*;
2. *Institute of Computing Technology, CAS, Beijing* 100190, *China*)

Abstract: Based on the analysis of multi-dimension Bloom filter presented before, an accurate multi-dimension counting Bloom filter(AMD-CBF) query algorithm based on bijective function is proposed for big data processing. When representing or querying an element, AMD-CBF needs two steps. The first step is to hash and map each attribute of the element to their corresponding accurate counting Bloom filter (A-CBF); The second step is to transform all attributes of the element into a value by bijective function to represent the overall information of the element, then the value is hashed and mapped into a combined counting Bloom filter (C-CBF) for completing the representation and query confirmation of the elements overall. Both theoretical analysis and experiment show that the AMD-CBF can support concise representation, approximate membership query and deletion of multi-dimension data set and significantly lower false positive rate and improve query accuracy compared to similar research.

Key words: big data processing; multi-dimension Bloom filter; bijective function; accurate counting Bloom filter; false positive

1 引言

未来的十年将是一个大数据引领的时代.大数据有三个典型特征:(1)数据结构复杂,元素属性多维化.如数字城市中空间数据具有三维坐标、地形等多维属性;网络Trace海量数据包具有源IP、目的IP、协议等多维属性;(2)数据价值密度低.价值密度的高低与数据总量的大小成反比.以视频为例,一部一小时的视频,在连续不间断监控过程中,可能有用的数据仅仅只有一两秒;(3)数据动态变化更新快.如何在快速变化的海量数据中通过高精度的数据查询算法迅速地完成数据的价值"提纯",成为有效进行大数据处理过程中极具挑战性的问题.

布鲁姆过滤器算法(Bloom Filter, BF)是一种结构精简的数据查询算法,虽然它存在稍许查询误判[1],但由于其查找的常数时间和存储空间开销小,从而使它具有很好的实用价值,已广泛应用于网络[2~4]、分布式计算[5]、云存储[6]等领域.但目前BF的研究主要集中在单维元素的表示和查询[7],如标准BF[1]、计数布鲁姆过滤器(Counting Bloom Filter, CBF)[8]、光谱BF[9],拆分型BF[10]、

收稿日期:2013-01-05;修回日期:2014-07-28;责任编辑:李勇锋
基金项目:国家重点基础研究发展规划(973计划)(No.2012CB315805);国家自然科学基金(No.61173167);江苏省未来网络前瞻性研究项目(No.BY2013095-1-05);湖南省科技计划(No.2013SK3149)

几何 BF[11]等.这些算法从不同角度讨论和优化 BF 的设计以满足实际应用的不同需求.目前存在少数针对多维元素的表示和查询的 BF 研究,如 MDBF[12]、CMDBF[13]和 PBF-BF[14],但是这些算法由于没有对元素的多维属性进行有效关联,存在查询误判率高的缺点,无法应用于未来大数据环境下多维元素表示和查询需求.因此,针对大数据特点,设计出高精度的多维 BF 来完成多维元素高效表示和查询成为大数据处理中迫切解决的问题.

本文详细分析了 MDBF、CMDBF 和 PBF-BF 特点和不足之处,提出了一种面向大数据处理的高精度多维计数布鲁姆过滤器(Accurate Multi-Dimension Counting Bloom Filter, AMD-CBF).AMD-CBF 元素表示、查找和删除分为两步进行,第 1 步是将元素的多个属性分别表示 A-CBF,A-CBF 是一种基于分层结构的 CBF,通过高效利用计数器的存储空间来降低假阳性,提升其查询精度;同时为了有效完成多属性的关联,第 2 步采用双射函数将元素多维属性转换为一维数值来表示元素整体信息,将这个一维数值映射到一个联合计数布鲁姆过滤器(Combined Counting Bloom Filter, C-CBF)中,联合元素各属性值利用 C-CBF 进行确认.理论分析和仿真实验表明,与 CMDBF、PBF-BF 相比,AMD-CBF 查询假阳性显著降低.

2　相关工作

标准 BF 采用长度为 m 的比特向量 $\boldsymbol{V}$ 表示 n 个元素集合 $S=\{s_1,s_2,\cdots,s_n\}$,采用 k 个相互独立的散列函数 $h_1,h_2,\cdots,h_k$,其函数取值均匀分布在范围为[1…m].插入元素时,设置 $\boldsymbol{V}$ 中第 $h_1(s),h_2(s),\cdots,h_k(s)$ 位为 1.查询元素 u 时,检查 $\boldsymbol{V}$ 中第 $h_1(u),h_2(u),\cdots,h_k(u)$ 位是否全为 1,如果全为 1,则元素 u 在 S 中;否则,元素 u 不在 S 中.后面章节中采用三元组 $\{n,m,k\}$ 形式化表示单维属性 BF,用四元组 $\{n,m,k,w\}$ 表示多维属性 BF,n 为集合 S 中元素个数,m 为向量 $\boldsymbol{V}$ 的长度,k 为散列函数的个数,w 为元素属性维数.目前针对多维元素的表示和查询主要有三种布鲁姆过滤器算法.

文献[12]提出多维布鲁姆过滤器(Multi-Dimension Bloom Filter, MDBF),其采用和维数相同的多个 BF 组成,直接将多维元素的表示和查询分解为单属性值子集的表示查询.MDBF 判断元素是否从属集合,需要判断所有属性是否在对应的属性子集合.但即使 MDBF 每一维没有出现误判,也有可能将不属于该集合的元素误判为属于该集合.

文献[13]考虑到将元素的整体信息在 BF 中表示出来,必然可以减少由于单维属性误判而导致的元素整体误判的可能性,由此提出改进的联合多维布鲁姆过滤器(Combined Multi-Dimension Bloom Filter, CMDBF).CMDBF 由两部分 BF 组成,第 1 部分用 BF 表示各属性子集,采用和 MDBF 一样的机制;第 2 部分用一个联合布鲁姆过滤器(Combined Bloom Filter, C-BF)来表示各个属性值的联合,将每个属性对应散列映射地址经过异或函数 $Fun(X)$ 运算后得到一个数值来表示元素整体信息,再将该数值经过散列映射到 C-BF 中.但 CMDBF 不支持元素删除操作.

为了解决元素的删除操作,文献[14]提出 PBF-BF(Parallel Bloom Filter-Bloom Filter),将多维 BF 的构造分为两个部分,第 1 部分用 CBF 表示各属性子集;第 2 部分是用一个 C-CBF 来表示各个属性值的联合,将不同属性的对应散列映射地址经过映射函数 $Fun(X)$ 运算后获得一个一维数值 v_a 表示元素整体信息,将 v_a 经过散列映射表示到 C-CBF 中.

相对于 MDBF, CMDBF、PBF-BF 将单维属性进一步关联,来降低多维元素查询的误判率,由于这两种算法采用的 $Fun(X)$ 均不是双射函数,因此对于同一个 Y 值,可能存在多个元素 $\{X_1,X_2,\cdots,X_k\}$ 与之对应,CMDBF 与 PBF-BF 假阳性率较高,导致查询精度大幅度降低.

3　高精度多维计数布鲁姆过滤器

3.1　AMD-CBF 结构和原理

为解决 CMDBF、PBF-BF 出现的问题,同时针对大数据特点,本文提出了 AMD-CBF 算法,其结构和原理如图 1 所示.

图 1 中,将 CMDBF、PBF-BF 中第 1 部分 BF 或 CBF 替换为 A-CBF,采用 A-CBF 表示元素的各属性子集;第 2 部分中映射函数 $Fun(X)$ 采用双射函数来表示元素整体信息,保证对于同一个 Y 值,只有唯一元素 X 与之对应,避免不同元素映射到第 2 部分 C-CBF 中相同位置.通过这两方面改进,降低假阳性,提高查询精度.下面详细论述这两方面的工作.

(1)A-CBF 设计

为解决标准 BF 删除操作,支持集合的动态变化,文献[7]提出了 CBF,如图 2(a)所示,用计数器替代标准 BF 中的每个 bit 位,一般情况下,每个计数器由 4bit 组成.元素的插入操作和删除操作分别将对应的 k 个计数器加 1 或者减 1.进行元素是否在集合中的判断时,只需要判断这 k 个计数器的值是否都大于 1.

在 CBF 实际应用中,大部分计数器的值只需 1 – 2 个 bit 来表示,从而造成内存空间浪费.由于 PBF-BF 采用了 $w+1$ 个 CBF,因此未能有效利用内存空间.为了解决此类问题,本文在多层压缩计数布鲁姆过滤器[15]

(MLC-CBF)基础上提出了一种高精度计数布鲁姆过滤器(Accurate Counting Bloom Filter, A-CBF), A-CBF 将标准 CBF 的空间 $4m$ 分为空间不等的多层($L_1, \cdots, L_d$),每层中每个位置(桶)由 1bit 组成, L_1 长度固定,其值为 $4m-kn$,其余各层及长度根据需要动态创建,原理如图 2 中(b)所示.

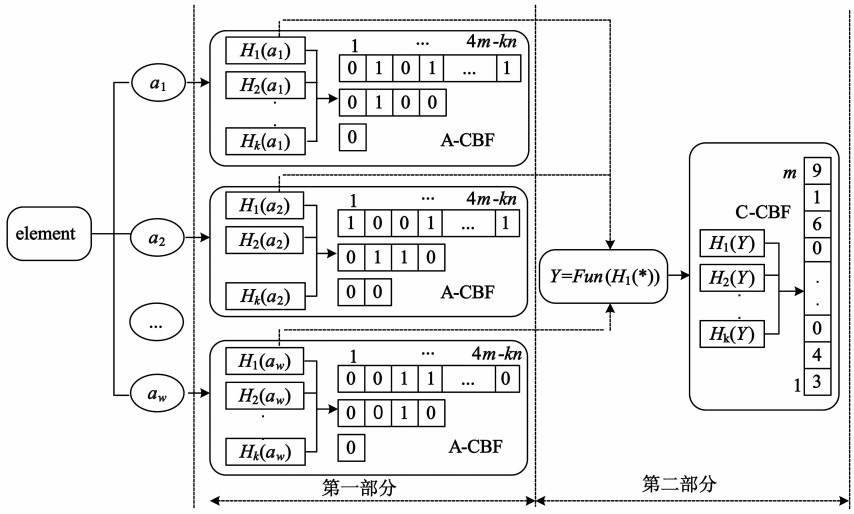

图1 AMD-CBF算法结构示意图

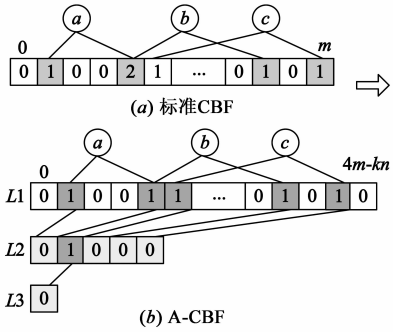

图2 标准CBF与A-CBF结构比较

图 2(b)中用哈夫曼编码 110 来表示图 2(a)标准 CBF 中第 5 个计数器的值 2;10 来表示值为 1 计数器. A-CBF 的第 2 层长度等于首层桶数为 1 的个数,第 3 层长度等于第 2 层桶数为 1 的个数,以此类推. A-CBF 查询某元素 u 时只需要判断元素 u 哈希到首层对应位置是否都为 1.

(2)映射函数 $Fun(X)$设计

本文第二个工作是采用双射函数 $Fun(X)$将元素 X 的多维属性($a_1, a_2, \cdots, a_w$)转换为一个数值 Y 来表示元素整体信息,并保证 X 与 Y 值的一一对应关系,降低 C-CBF 的假阳性,提高查询精度.

已经证明,两个自然数集合的笛卡尔积 $N \times N$ 与 N 之间是双射的[16],是实现多维数据与一维数据相互转换的有效方法,即函数 $f: N \times N \to N$ 为双射函数,其运算式为:

$$f(i,j) = (i+j)(i+j+1)/2 + i, (i,j) \in N \times N \quad (1)$$

根据 $N \times N$ 与 N 之间是双射函数 f 构造出从 N^n 到 N 的双射,N^n 表示 n 个集合 N 的笛卡尔积 $N \times N, \cdots, \times N$. 对于任意 n 元组 $(x_1, x_2, \cdots, x_n) \in N^n$,从 N^n 到 N 双射函数 $f_n: N^n \to N$ 为:

$$f_n(x_1, x_2, \cdots, x_n) = f_n(f_{n-1}(x_1, x_2, \cdots, x_{n-1}), x_n) \quad (2)$$

本文采用式(2)来设计图 1 中的映射函数 $Fun(X)$ 为:

$$\begin{aligned} &Fun_w(H_1(a_1), H_1(a_2), \cdots, H_1(a_w)) \\ &= Fun_w(Fun_{w-1}(H_1(a_1), \\ &H_1(a_2), \cdots, H_1(a_{w-1})), H_1(a_w)) \end{aligned} \quad (3)$$

但由于元素属性可能是非数字,因此采用散列函数将元素属性转换为数字,式(3)中 H_1 为 A-CBF 采用 k 个散列函数中第 1 个散列函数.

3.2 AMD-CBF 理论假阳性计算

为了更好地比较 CMDBF、PBF-BF、AMD-CBF 查询精度,下面从理论上来分析这几种多维 BF 的假阳性. 对于 CMDBF、PBF-BF、AMD-CBF 来说,整体理论假阳性等于第 1 部分假阳性 f_1 与第 2 部分假阳性 f_2 乘积,即:

$$\begin{aligned} f = f_1 \times f_2 &= [P_1 \times f_{11} + P_2 \times f_{11}^2 + \cdots, + \\ &P_w \times f_{11}^w] \times f_2 = \left[\sum_{i=1}^{w} P_i \times f_{11}^i\right] \times f_2 \end{aligned} \quad (4)$$

式(4)中 f_{11}表示元素各个属性对应 BF 的假阳性; P_i 代表第 i 个属性对应 BF 同时出现误判时的概率,其计算式为:

$$P_i = \binom{w}{i} / \binom{w}{1} + \binom{w}{2} + \cdots, + \binom{w}{w} = \binom{w}{i} / \sum_{j=1}^{w} \binom{w}{j} \quad (5)$$

对于 CMDBF 和 PBF-BF,式(4)中 f_{11}与 f_2 计算式为[4]:

$$f_{11}=f_2=f_{\mathrm{CBF}}(m,k,n)=f_{\mathrm{BF}}(m,k,n)=(1-\mathrm{e}^{-kn/m})^k \tag{6}$$

根据式(4)和(6),CMDBF 和 PBF-BF 整体假阳性计算式为:

$$f_{\mathrm{CMDBF}}=f_{\mathrm{PBF\text{-}BF}}=\Big[\sum_{i=1}^{w}P_i\times(1-\mathrm{e}^{-kn/m})^{k\cdot i}\Big]\times(1-\mathrm{e}^{-kn/m})^k \tag{7}$$

对于 AMD-CBF,式(4)中 $f_2=f_{\mathrm{BF}}$,f_{11}计算式为:

$$f_{11}=f_{\mathrm{A-CBF}}((4m-kn),k,n)=f_{\mathrm{BF}}((4m-kn),k,n)=(1-\mathrm{e}^{-kn/(4m-kn)})^k \tag{8}$$

根据式(4)、(6)和(8),AMD-CBF 整体假阳性 $f_{\mathrm{AMD\text{-}CBF}}$计算公式为:

$$f_{\mathrm{AMD\text{-}CBF}}=\Big[\sum_{i=1}^{w}P_i\times(1-\mathrm{e}^{-kn/(4m-kn)})^{k\cdot i}\Big]\times(1-\mathrm{e}^{-kn/m})^k \tag{9}$$

通过以上理论分析,根据式(6)得知 m 值的大小与 BF 假阳性 $f_{\mathrm{BF}}(m,k,n)$成反比,因此当 $4m-kn>m$,即 $kn<3m$ 时,根据式(7)和式(9)可以得知 AMD-CBF 的假阳性明显低于 CMDBF、PBF-BF.

3.3　AMD-CBF 算法实现

AMD-CBF 中元素插入算法如算法 1 伪代码所示所示.元素加入时,首先分离元素各个属性值,将每个属性值映射到对应的 A-CBF;同时将该元素所有属性值经过双射函数转换为一个值,通过散列映射到 C-CBF 中,完成元素插入操作.

算法 1　AMD-CBF 元素插入算法

```
1  int Addr[t][k],CAddr[k]//k 为 A-CBF 的散列函数个数,t 为 A-CBF 层数
2  for(i=0;i<e.attri_length;i++)
3    ACBF=GetACBF(i)//为每维属性创建一个 A-CBF
4    Hash1=GetHash1(i)//为每维属性创建一组散列函数
5    for(j=0;j<k;j++)
6      Addr[0][j]=Hash1[j](e.attr[i]) //计算属性在 A-CBF 首层的 k 个散列值
7      while(ACBF[i].S[t](Addr[t][j])=1)//检查 A-CBF 第 t 层对应位置的值
8        ++t//继续轮询 A-CBF 第 t+1 层
9        Addr[t][j]=countOne(S[t],Addr[t-1][j])//计算当前层置 1 的位置
10     end while
11     Set(ACBF[i].S(t),Addr[t][j])=1//将 A-CBF 第 t 层指定位置设为 1
12     Addr[t+1][j]=countOne(S[t],Addr[t][j])//计算下层增加 bit 位置
13     Insert(ACBF[i].S(t+1),Addr[t+1][j],0)//t+1 层指定位置增加 1 比特
14   end for
15   end for
16   CCBF=GetCCBF()//创建 C-CBF
17   Y=Fun(Hash1[1](e))//双射函数将元素所有属性转换为一个值 Y
18   Hash2=GetHash2()//为 Y 创建一组散列函数
19   for(i=0;i<k;i++)
20     Caddr[i]=Hash2(Y)//计算 Y 在 C-CBF 中的 k 个散列值
21     AddOne(CCBF,Caddr[i])//将 Y 插入 C-CBF 中
22   end for
```

判断元素是否属于该集合时,分两个步骤进行.第 1 步检查该元素的各个属性值是否在各个对应的 A-CBF 中,如果此轮检查发现该元素的各个属性值都已经映射到 A-CBF 中,再进行第 2 步检查;第 2 步检查各维属性值经双射函数转换后的值散列到 C-CBF 中的相应位置是否都大于零,如大于零,则判断该元素属于该集合,否则判断该元素不属于该集合.AMD-CBF 的查询算法如算法 2 中伪代码所示.

算法 2　AMD-CBF 元素查询算法

```
1  int Addr[t][k],CAddr[k]//k 为 A-CBF 的散列函数个数,t 为 A-CBF 层数
2  for(i=0;i<e.attri_length;i++)
3    ACBF=Get ACBF(i)//为每维属性创建一个 ACBF
4    Hash1=GetHash1(i)//为每维属性创建一组散列函数
5    for(j=0;j<k;j++)
6      Addr[0][j]=Hash1[j](e.attr[i])//计算每个属性的 k 个散列值
7      if(ACBF[i].S[0](Addr[0][j])==0)//检查哈希值在 A-CBF 首层中对应值
8        return false//k 个位置只要有 1 个为 0,则返回假值
9      end if
10   end for
11 end for
12 CCBF=GetCCBF()//创建 C-CBF
13 Y=Fun(Hash1[1](e))//双射函数将元素所有属性转换为一个值 Y
14 Hash2=GetHash2()//为 Y 创建一组散列函数
15 for(i=0;i<k;i++)
16   Caddr[i]=Hash2(Y) //计算 Y 在 C-CBF 中的 k 个散列值
17   if(CCBF.Caddr[i]==0) 检查散列值在 C-CBF 中对应值
18     return false//k 个位置只要有 1 个为 0,则返回假值
19 end for
20 return true
```

4　实验评估

本文采用美国应用网络研究国家实验室 NLANR 的 Trace 数据[17]进行 IP 黑名单过滤.实验数据来源是加州大学的圣迭戈超级计算中心(SDSC)到 Abilene 的 OC12 链路.随机选取 30 组数据源,每组数据源采集持续时间都为 90 秒左右,每组待过滤的数据包个数平均值为 665029 个,每组 IP 黑名单规则个数平均值为 1330.

实验方法是将源 IP 和目的 IP 组成的二维数据 IP 黑名单集分别插入 CMDBF、PBF-BF、AMD-CBF 中,然后从 Trace 中获得数据包,判断数据包中目的 IP 和源 IP 是否在 IP 黑名单集中,如在则该数据包则被过滤. CMDBF、PBF-BF、AMD-CBF 各自假阳性如图 3 和图 4 所示(30 组数据源取平均值).

从图 3 和图 4 得知:$k=3$、$m/n=10$ 时,同 PBF-BF 相比,AMD-CBF 假阳性降低了 11.7 倍;同 CMDBF 相比,AMD-CBF 假阳性降低了 18 倍. $k=3$、$m/n=12$ 时,同 PBF-BF 相比,AMD-CBF 假阳性降低了 12.7 倍;同 CMDBF 相比,AMD-CBF 假阳性降低了 21.7 倍. $k=6$、$m/n=10$ 时,同 PBF-BF 相比,AMD-CBF 假阳性降低了 11.9 倍;同 CDMBF 相比,AMD-CBF 假阳性降低了 20.1 倍. $k=6$、$m/n=12$ 时,同 PBF-BF 相比,AMD-CBF 假阳性降低了 16.4 倍;同 CDMBF 相比,AMD-CBF 假阳性降低了 22.3 倍. AMD-CBF 算法查询精度大幅度提高.

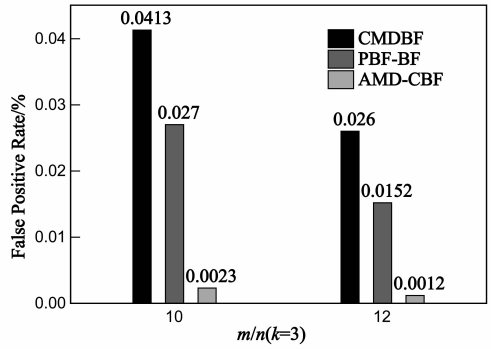

图3 IP黑名单过滤假阳性对比(k=3)

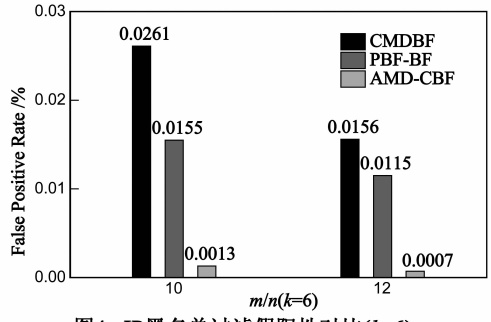

图4 IP黑名单过滤假阳性对比(k=6)

5 总结

本文针对大数据处理特点,研究如何将 BF 由一维扩展到多维的方法,仔细分析现有的 MDBF、CMDBF、PBF-BF 算法的运行机理,从提高查询精度角度提出了一种基于双射函数的 AMD-CBF 算法.

通过理论分析和仿真实验结果表明,AMD-CBF 对大规模多维数据集表示和查找很有效,查询精度大幅度提高.实际数据实验中,AMD-CBF 同 PBF-BF 相比,假阳性最高可降低 16.4 倍($k=6$, $m/n=12$),同 CMDBF 相比,假阳性最高可降低 22.3 倍. AMD-CBF 查询执行时间同 PBF-BF 相比,最高降低了 45%左右($k=6$).

参考文献

[1] Andrei B, Michael M. Network applications of Bloom filters: a survey[J]. Internet Mathematics, 2005, 1(4): 485 – 509.

[2] 谢鲲,赵姣姣,张大方,等.基于计数布鲁姆过滤器的快速多维包分类算法[J].电子学报,2010,38(5):1046 – 1052.
Xie Kun, Zhao Jiaojiao, Zhang Dafang, et al. A fast multi-dimensional packet classification algorithm using counting Bloom filter[J]. Acta Electronica Sinica, 2010, 38(5): 1046 – 1052. (in Chinese)

[3] 刘威,郭渊博,黄鹏.基于 Bloom filter 的多模式匹配引擎[J].电子学报,2010,38(5):1095 – 1099.
Liu Wei, Guo Yuanbo, Huang Peng. Multi-pattern matching engine based on Bloom filter[J]. Acta Electronica Sinica, 2010, 38(5): 1095 – 1099. (in Chinese)

[4] Heeyeol Yu, Rabi N. A power and throughput-efficient packet classification with n Bloom filters[J]. IEEE Trans on Computers, 2011, 60(8): 1182 – 1193.

[5] Siyuan Liu, Lei Kang, Lei Chen, et al. Distributed incomplete pattern matching via a novel weighted Bloom filter[A]. Proc of the 32nd International Conference on Distributed Computing Systems[C]. Piscataway, NJ: IEEE, 2012. 122 – 131.

[6] Jiansheng Wei, Ke Zhou. Efficiently representing membership for variable large data sets[J]. IEEE Trans on Parallel and Distributed Systems, 2014, 25(4): 960 – 970.

[7] Yan Qiao, Tao Li, Shigang Chen. Fast Bloom filters and their generalization[J]. IEEE Trans on Parallel and Distributed Systems, 2014, 25(1): 93 – 103.

[8] Fan Li, Cao Pei, Almeida J, et al. Summary cache: a scalable wide-area web cache sharing protocol[J]. IEEE Trans on Networking, 2000, 8(3): 281 – 293.

[9] Saar Cohen, Yossi Matias. Spectral Bloom filters[A]. Proc of ACM SIGMOD International Conference on Management of Data[C]. New York: ACM, 2003. 241 – 252.

[10] 肖明忠,代亚非,李晓明.拆分型 Bloom Filter[J].电子学报,2004,32(2):241 – 245.
Xiao Mingzhong, Dai Yafei, Li Xiaoming. Split Bloom filter [J]. Acta Electronica Sinica, 2004, 32(2): 241 – 245. (in Chinese)

[11] 张震,汪斌强,陈庶樵,等.几何布鲁姆过滤器的设计与分析[J].电子学报,2012,40(9):1852 – 1857.
Zhang Zhen, Wang Binqiang, Chen Shuqiao, et al. Geometric Bloom filter designing and its analysis[J]. Acta Electronica Sinica, 2012, 40(9): 1852 – 1857. (in Chinese)

[12] Deke Guo, Jie Wu, Honghui Chen, et al. Theory and network

application of dynamic Bloom filters[A]. Proc of the 25th Annual International Conference on Computer Communications[C]. Piscataway, NJ: IEEE, 2006. 1 – 12.

[13] 谢鲲,秦拯,文吉刚,等.联合多维布鲁姆过滤器查询算法[J].通信学报,2008,29(1):56 – 64.
Xie Kun, Qin Zheng, Wen Jigang, et al. Combine multi-dimension Bloom filter for membership queries[J]. Journal on Communications, 2008, 29(1): 56 – 64. (in Chinese)

[14] Bin Xiao, Yu Hua. Using parallel Bloom filters for multi-attribute representation on network services[J]. IEEE Trans on Parallel and Distributed Systems, 2010, 21(1): 20 – 32.

[15] Domenico Ficara, Stefano Giordano, Gregorio Procissi, et al. Multilayer compressed counting Bloom filters[A]. Proc of the 27th Annual International Conference on Computer Communications[C]. Piscataway, NJ: IEEE, 2008. 932 – 941.

[16] 左孝凌,刘永才.离散数学[M].上海:上海科学技术文献出版社,1982. 166 – 168.
Zuo Xiaoling, Liu Yongcai. Discrete Mathematics[M]. Shanghai: Shanghai Scientific & Technological Literature Publishing House, 1982. 166 – 168. (in Chinese)

[17] NLANR. Index of /datasets/wits. cs. waikato. ac. nz/pma/Traces[EB /OL]. https://labs. ripe. net/datarepository/datasets/nlanr-pma-data, 2005-07-01.

作者简介

李　玮　男,1972 年生,助理教授,博士生,研究方向大数据处理、软件工程、可信系统与网络.
E-mail: rj_ wli@hnu. edu. cn

张大方　男,1959 年生,教授、博导,博士,研究方向可信系统与网络、软件工程、大数据处理.
E-mail: dfzhang@hnu. edu. cn

黄　昆　男,1978 年生,助理研究员,博士,研究方向可信系统与网络、大数据处理.
E-mail: huangkun09@ict. ac. cn

谢　鲲　女,1978 年生,副教授,博士,研究方向为分布式计算、协作路由、大数据处理.
E-mail: xiekun@hnu. edu. cn