布鲁姆过滤器代数运算探讨

谢 鲲[1,2], 张大方[3], 文吉刚[1], 谢高岗[4], 尤志强[3]
(1. 湖南大学计算机与通信学院, 湖南长沙 410082; 2. 香港理工大学电子计算学系, 香港;
3. 湖南大学软件学院, 湖南长沙 410082; 4. 中国科学院计算技术研究所, 北京 100080)

摘 要: 本文探讨布鲁姆过滤器的代数运算和集合查询的关系, 定义布鲁姆过滤器的"并","交","异或","补","差"代数运算, 从理论和实验两方面分析布鲁姆过滤器的代数运算和集合代数运算并集, 交集, 异或集, 补集, 差集的元素查询关系. 理论分析和实验结果表明, 布鲁姆过滤器的"并","交"运算能够支持集合并集交集的元素查询, 这一结论可以简化利用布鲁姆过滤器进行的系统设计.

关键词: 计算机网络; 分布式计算; 分布式消息系统; 集合元素查询; 代数运算
中图分类号: TP393 文献标识码: A 文章编号: 0372-2112 (2008) 05-0869-06

Algebraic Operations on Bloom Filters

XIE Kun[1,2], ZHANG Da-fang[3], WEN Ji-gang[1], XIE Gao-gang[4], YOU Zhi-Qiang[3]
(1. *School of Computer and Communication, Hunan University, Changsha, Hunan 410082, China;*
2. *Department of Computing, The Hong Kong Polytechnic University, Hong Kong;*
3. *School of Software, Hunan University, Changsha, Hunan 410082, China;*
4. *Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100080, China*)

Abstract: This paper discusses the relationship between algebraic operations on Bloom filters and algebraic operations on data sets. This paper completely define algebraic operations including OR, AND, XOR, NOT, MINUS on Bloom filter, and study the membership query performance on Bloom filter and data set. Theoretical analyses and simulation results show that the Bloom filter ORed (ANDed) from the original Bloom filters can support element membership query on data set ORed (ANDed) from the original data sets, which can be a trick to real application.

Key words: computer networks; distributed computing; distributed information system; set membership query; algebraic operations

1 引言

布鲁姆过滤器查询算法[1]作为一种高效简洁的查询算法, 已广泛应用于各种计算机和网络系统中[2~4]. 目前, 该算法的研究主要集中在针对具体应用需求的算法改进[5~10], 这对于扩展算法应用领域远远不够.

布鲁姆过滤器算法数据结构的本质是由"0""1"组成的位串. 直接从布鲁姆过滤器向量的串运算和操作角度出发, 探讨其算法性质, 这类研究还十分少见. 数据库作为表示和存储集合的数据结构, 研究代数运算下的查询关系, 该领域已经展开了不少关系代数的研究. 那么布鲁姆过滤器作为另外一种表示集合的数据结构, 它的位串运算也可能和集合的运算存在某种潜在的关系. 布鲁姆过滤器相关的运算方法或者其他泛函运算的研究是否一样可以优化集合的查询操作? 在集合进行"并""交"运算时, 布鲁姆过滤器是否可以进行"并""交"等布尔运算? 如果布鲁姆过滤器能够进行这类运算, 那么布鲁姆过滤器的布尔运算是否还具有集合查询的功能? 运算后的布鲁姆过滤器对查询算法性能又有何影响? 这正是本文探讨的问题. 就目前的研究来看, 文献[3]和文献[8]对布鲁姆过滤器运算的有过初浅的探讨, 却没有相关运算的系统定义, 也没有系统的从理论和实验方面验证过集合运算和布鲁姆过滤器运算的关系. 本文从布鲁姆过滤器向量的位串运算出发, 研究布鲁姆过滤器运算和集合查询的关系.

2 布鲁姆过滤器查询算法原理

标准布鲁姆过滤器查询算法[1]核心是一个 V 向量和一组哈希函数. 设集合 $S=\{s_1, s_2, \cdots, s_n\}$ 共有 n 个元素, 通过 k 个哈希函数 $h_1, h_2, \cdots, h_k$ 映射到长度为 m 的向量 V 中. 当元素插入集合时, 对于每一个元素 s_i, 计算 $h_j(s_i)(1 \leqslant j \leqslant k)$, 若 $h_j(s_i)=q$, 则令 $BF[q]=1$, 将向量对应位置置位, 完成元素的插入. 查询给定的元素 x 是否在集合, 只需要检查向量 V 的 k 个位置 $(h_1(x),$

收稿日期: 2007-10-22; 修回日期: 2008-01-20
基金项目: 国防基础科研"十一五"规划(No. A1420060162)

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

$h_2(x)$, …, $h_k(x)$)是否都为 1. 如果全是 1, x 可能在集合中, 但不一定; 否则肯定不在集合中. 此时可能出现所谓假阳性误判(false positive), 即将不属于集合的元素误判断成属于集合. 标准布鲁姆过滤器的假阳性误判率为[8, 10]:

$$f^{BF}(m, k, n) = (1-p)^k = \exp(k\ln(1-e^{-kn/m})) \quad (1)$$

使用标准布鲁姆过滤器完成集合存储, m 比特位串向量可以表示 n 个元素的集合, 每个元素平均保存 m/n 位, 大大节约存储空间, 这也是布鲁姆过滤器得以广泛应用的原因.

3 布鲁姆过滤器形式化表示和相关定义

定义**1** 元素的布鲁姆过滤器表示. 对于全集 U 中的任意元素 x, 通过 k 个哈希函数, 表示到长度为 m 位的布鲁姆过滤器向量中, 向量记为 $BF^{k,m}(x)$, $BF^{k,m}(x)[hash_i(x)] = 1 (1 \leqslant i \leqslant k)$. 元素到布鲁姆过滤器向量的映射过程为 $x \xrightarrow{k,m} BF^{k,m}(x)$.

定义**2** 集合的布鲁姆过滤器表示. 对于全集 U 中的任意子集 $S = \{s_1, s_2, \cdots, s_n\}$ 中所有的元素都通过 k 个哈希函数, 表示到长度为 m 位的布鲁姆过滤器向量中, 向量记为 $BF^{k,m}(S)$. 集合到布鲁姆过滤器向量的映射过程为 $S \xrightarrow{k,m} BF^{k,m}(S)$.

定义**3** 同源布鲁姆过滤器. 对于全集 U 中的任意子集 S, 使用相同的 k 个哈希函数, 映射到相同长度为 m 位的布鲁姆过滤器向量中, 得到的一类布鲁姆过滤器称为同源布鲁姆过滤器, 记为 $BF^{k,m}$, 即 $BF^{(k_0, m_0)} = \{BF \mid \forall S \in U, S \xrightarrow{k=k_0,\ m=m_0} BF^{k_0,m_0}(S)\}$.

定义 **4** 同源布鲁姆过滤器的并运算“∪”直接通过位向量逻辑“或”完成. 即, S_a, S_b 为全集 U 中的任意子集, 它们的布鲁姆过滤器的并为 $BF^{(k,m)}(S_a)[i] \cup BF^{k,m}(S_b)[i] (0 \leqslant i \leqslant m-1)$. 如图 1 所示.

图 1 同源布鲁姆过滤器并运算

定义 **5** 同源布鲁姆过滤器的交运算“∩”直接通过位向量逻辑“与”完成. 即, S_a, S_b 为全集 U 中的任意子集, 它们的布鲁姆过滤器的交为 $BF^{k,m}(S_a)[i] \cap BF^{k,m}(S_b)[i] (0 \leqslant i \leqslant m-1)$. 如图 2 所示.

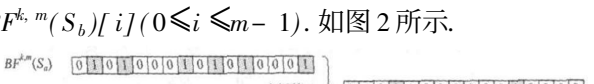

图 2 同源布鲁姆过滤器交运算

定义 **6** 同源布鲁姆过滤器的异或运算“⊕”直接通过位向量逻辑“异或”完成. 即, S_a, S_b 为全集 U 中的任意子集, 它们的布鲁姆过滤器的异或为 $BF^{k,m}(S_a)[i] \oplus BF^{k,m}(S_b)[i] (0 \leqslant i \leqslant m-1)$. 如图 3 所示.

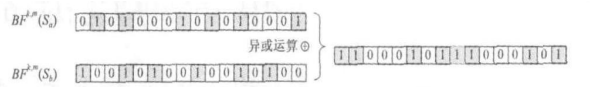

图 3 同源布鲁姆过滤器异或运算

定义 **7** 同源布鲁姆过滤器零元是指布鲁姆过滤器位串向量每个位都为“0”的布鲁姆过滤器, 用 0 表示.

定义 **8** 同源布鲁姆过滤器么元是指全集 U 的布鲁姆过滤器表示, 用 1 表示.

定义 **9** 同源布鲁姆过滤器的补运算“-”直接通过位向量和么元的逻辑“异或”完成. 即, S_a 为全集 U 中的任意子集, 它的布鲁姆过滤器表示的补记为 $\overline{BF^{(k,m)}(S_a)}$.

定义 **10** 同源布鲁姆过滤器的差运算“-”, 是交运算和异或运算的组合. 即, S_a, S_b 为全集 U 中的任意子集, 它们的布鲁姆过滤器的差为 $BF^{(k,m)}(S_a) - BF^{(k,m)}(S_b) = BF^{(k,m)}(S_a) \oplus (BF^{k,m}(S_a) \cap BF^{k,m}(S_b))$.

4 布鲁姆过滤器代数运算和集合查询的关系

4.1 布鲁姆过滤器并运算查询算法

定理 **1** 对于全集 U 中任意的子集 S_a 和 S_b, $BF^{(k,m)}(S_a)$ 和 $BF^{(k,m)}(S_b)$ 分别为它们的布鲁姆过滤器表示, 则集合 S_a 和 S_b 并集的布鲁姆过滤器表示为它们的布鲁姆过滤器的并, 即

$$BF^{(k,m)}(S_a \cup S_b) = BF^{k,m}(S_a) \cup BF^{k,m}(S_b) \quad (2)$$

证明

(1) $\forall x \in S_a \cup S_b$

$$\left.\begin{array}{l}(a)\ x \in S_a \cup S_b \rightarrow BF^{(k,m)}(S_a \cup S_b)[hash_i(x)] = 1 (1 \leqslant i \leqslant k) \\ (b)\ x \in S_a \cup S_b \rightarrow x \in S_a \text{ 或者 } x \in S_b \rightarrow BF^{k,m}(S_a)[hash_i(x)] \\ \cup BF^{k,m}(S_b)[hash_i(x)] = 1 (1 \leqslant i \leqslant k)\end{array}\right\}$$

$$\rightarrow BF^{k,m}(S_a \cup S_b) \Rightarrow BF^{k,m}(S_a) \cup BF^{k,m}(S_b)$$

(2) $\forall x \in S_a$ 或者 $x \in S_b$

$$\left.\begin{array}{l}(a)\ x \in S_a \text{ 或者 } x \in S_b \rightarrow BF^{k,m}(S_a)[hash_i(x)] \\ \cup BF^{k,m}(S_b)[hash_i(x)] = 1 (1 \leqslant i \leqslant k) \\ (b)\ x \in S_a \text{ 或者 } x \in S_b \rightarrow x \in S_a \cup S_b \rightarrow BF^{k,m} \\ (S_a \cup S_b)[hash_i(x)] = 1 (1 \leqslant i \leqslant k)\end{array}\right\}$$

$$\rightarrow BF^{k,m}(S_a) \cup BF^{k,m}(S_b) \Rightarrow BF^{k,m}(S_a \cup S_b)$$

定理 **2** 布鲁姆过滤器并运算的元素查询假阳性误判率大于等于单个布鲁姆过滤器的查询假阳性误判率, 其中布鲁姆过滤器并运算元素查询假阳性误判率为

$$f^{BF}(m, k, n_\cup) = (1 - e^{-kn_\cup/m})^k \quad (3)$$

$n_\cup$ 是集合并集的规模.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

证明　对于全集 U 中任意的子集 S_a 和 S_b, $BF^{k,m}(S_a)$ 和 $BF^{k,m}(S_b)$ 分别为它们的布鲁姆过滤器表示, $BF^{k,m}(S_a \cup S_b)$ 为 S_a 和 S_b 并集的布鲁姆过滤器表示. 令集合规模 $n_a = |S_a|$, $n_b = |S_b|$, $n_\cup = |S_a \cup S_b|$.

由式(1), 得布鲁姆过滤器 $BF^{k,m}(S_a \cup S_b)$, $BF^{k,m}(S_a)$ 和 $BF^{k,m}(S_b)$ 的元素查询假阳性误判率分别为: $f^{BF}(m, k, n_\cup) = (1 - e^{-kn_\cup/m})^k$, $f^{BF}(m, k, n_a) = (1 - e^{-kn_a/m})^k$, $f^{BF}(m, k, n_b) = (1 - e^{-kn_b/m})^k$.

$n_\cup \geqslant \max(n_a, n_b) \rightarrow f^{BF}(m, k, n_\cup) \geqslant f^{BF}(m, k, n_a)$ 且 $f^{BF}(m, k, n_\cup) \geqslant f^{BF}(m, k, n_b)$

所以, 集合并集的布鲁姆过滤器查询假阳性误判率大于等于单个集合的布鲁姆过滤器表示的查询假阳性误判率. 由定理 1 知 $BF^{k,m}(S_a \cup S_b) = BF^{k,m}(S_a) \cup BF^{k,m}(S_b)$, 定理 2 即证.

4.2　布鲁姆过滤器交运算查询算法

定义 11　对于全集 U 中任意的子集 S_a 和 S_b, $BF^{k,m}(S_a)$ 和 $BF^{k,m}(S_b)$ 分别为它们的布鲁姆过滤器表示. 如果 $\forall i \in \{0, \cdots, m-1\}$ 都有当 $BF^{k,m}(S_b)[i] = 1$ 时 $BF^{k,m}(S_a)[i] = 1$, 则布鲁姆过滤器 $BF^{k,m}(S_a)$ 强于 $BF^{k,m}(S_b)$ 记为 $BF^{k,m}(S_a) \geqslant BF^{k,m}(S_b)$, $BF^{k,m}(S_b)$ 弱于 $BF^{k,m}(S_a)$ 记为 $BF^{k,m}(S_a) \leqslant BF^{k,m}(S_b)$.

定理 3　对于全集 U 中任意的子集 S_a 和 S_b, $BF^{k,m}(S_a)$ 和 $BF^{k,m}(S_b)$ 分别为它们的布鲁姆过滤器表示, 则集合 S_a 和 S_b 的交集的布鲁姆过滤器表示弱于它们的布鲁姆过滤器的交, 即:

$$BF^{k,m}(S_a \cap S_b) \leqslant BF^{k,m}(S_a) \cap BF^{k,m}(S_b) \tag{4}$$

证明　同源布鲁姆过滤器交运算和单个布鲁姆过滤器向量的关系如下:

$$BF^{k,m}(S_a) \cap BF^{k,m}(S_b)[i] = \begin{cases} 1 \text{ 当 } x \in S_a \cap S_b \text{ 对于所有 } j(1 \leqslant j \leqslant k) \\ \quad i = hash_j(x),\ BF^{k,m}(S_a)[i] = BF^{k,m}(S_b)[i] = 1 \\ 1 \text{ 当 } x \in S_a - S_a \cap S_b,\ y \in S_b - S_a \cap S_b, \\ \quad \text{存在 } j(1 \leqslant j \leqslant k) \text{ 和 } j'(1 \leqslant j' \leqslant k) \\ \quad i = hash_j(x) = hash_{j'}(y),\ BF^{k,m}(S_a)[i] \\ \quad = BF^{k,m}(S_b)[i] = 1 \\ 0 \quad \text{不是上面两种情况} \end{cases} \tag{5}$$

因此, $\forall i \in \{0, \cdots, m-1\}$ 都有当 $BF^{k,m}(S_a \cap S_b)[i] = 1$ 时, $BF^{k,m}(S_a) \cap BF^{k,m}(S_b)[i] = 1$.

定理 4　布鲁姆过滤器交运算的元素查询假阳性误判率大于等于集合交集的布鲁姆过滤器表示的查询假阳性误判率, 其中布鲁姆过滤器交运算的元素查询假阳性误判率为

$$f^{BF}(m, k, n_\cap) = (1 - e^{-kn_a/m} - e^{-kn_b/m} + e^{-k(n_a + n_b - n_\cap)/m})^k \tag{6}$$

n_a, n_b, $n_\cap$ 分别是单个集合和交集的规模.

证明　对于全集 U 中任意的子集 S_a 和 S_b, $BF^{k,m}(S_a)$ 和 $BF^{k,m}(S_b)$ 分别为它们的布鲁姆过滤器表示. 令 $n_a = |S_a|$, $n_b = |S_b|$, $n_\cap = |S_a \cap S_b|$. 具体的, $BF^{k,m}(S_a) \cap BF^{k,m}(S_b)$ 任一位为 1 的概率为:

$$p = (1 - (1 - 1/m)^{kn_\cap}) + (1 - 1/m)^{kn_\cap}(1 - (1 - 1/m)^{k(n_a - n_\cap)})(1 - (1 - 1/m)^{k(n_b - n_\cap)}) \tag{7}$$

因此, 布鲁姆过滤器交运算的元素查询假阳性误判率为

$$f^{BF}(m, k, n_\cap) = p^k = (1 - e^{-kn_a/m} - e^{-kn_b/m} + e^{-k(n_a + n_b - n_\cap)/m})^k$$

布鲁姆过滤器交运算的元素查询假阳性误判率大于等于集合交集的布鲁姆过滤器表示的查询假阳性误判率.

4.3　布鲁姆过滤器异或运算查询算法

定理 5　对于全集 U 中任意的子集 S_a 和 S_b, $BF^{k,m}(S_a)$ 和 $BF^{k,m}(S_b)$ 分别为它们的布鲁姆过滤器表示, 则集合 S_a 和 S_b 的异或的布鲁姆过滤器表示强于它们的布鲁姆过滤器的异或, 即:

$$BF^{k,m}(S_a \oplus S_b) \geqslant BF^{k,m}(S_a) \oplus BF^{k,m}(S_b) \tag{8}$$

证明　同源布鲁姆过滤器异或运算和单个布鲁姆过滤器向量的关系如下:

$$BF^{k,m}(S_a) \cap BF^{k,m}(S_b)[i] = \begin{cases} 0 \quad BF^{k,m}(S_a)[i] = BF^{k,m}(S_b)[i] = 0 \\ 0 \text{ 当 } x \in S_a \cap S_b, \text{ 对于所有 } j(1 \leqslant j \leqslant k) \\ \quad i = hash_j(x),\ BF^{k,m}(S_a)[i] = BF^{k,m}(S_b)[i] = 1 \\ 0 \text{ 当 } x \in S_a - S_a \cap S_b,\ y \in S_b - S_a \cap S_b, \\ \quad \text{存在 } j(1 \leqslant j \leqslant k) \text{ 和 } j'(1 \leqslant j' \leqslant k) \\ \quad i = hash_j(x) = hash_{j'}(y),\ BF^{k,m}(S_a)[i] \\ \quad = BF^{k,m}(S_b)[i] = 1 \\ 1 \quad x \in S_a - S_a \cap S_b,\ y \in S_b - S_a \cap S_b \\ \quad i = hash_j(x), \text{ 不存在 } j'(1 \leqslant j' \leqslant k), \\ \quad \text{使得 } hash_{j'}(y) = i \end{cases} \tag{9}$$

因此, $\forall i \in \{0, \cdots, m-1\}$ 都有当 $BF^{k,m}(S_a) \oplus BF^{k,m}(S_b)[i] = 1$ 时, $BF^{k,m}(S_a \oplus S_b) = 1$ 成立.

由于 $BF^{k,m}(S_a) \oplus BF^{k,m}(S_b)$ 置 1 位个数可能少于 $BF^{k,m}(S_a \oplus S_b)$, 使用 $BF^{k,m}(S_a) \oplus BF^{k,m}(S_b)$ 查询元素 $x \in S_a \oplus S_b$ 时, 会出现假阴性误判. 从直觉看来, 假阴性误判只需要元素对应过滤器任何一位出现 0 就会发生, 这种误判概率可能比较大, 能否用 $BF^{k,m}(S_a) \oplus BF^{k,m}(S_b)$ 替代完成异或集合的元素查询, 需要进一步验证. 数学分析得, $BF^{k,m}(S_a) \oplus BF^{k,m}(S_b)$ 任一位为 1 的概率为:

$$t_\oplus = (1 - e^{-k(n_a + n_b - n_\cap)/m}) - (1 - e^{-kn_a/m} - e^{-kn_b/m}$$

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

$$+ e^{-k(n_a+n_b-n_\cap)/m}) \quad (10)$$

则 $BF^{k,m}(S_a) \oplus BF^{k,m}(S_b)$ 的假阳性误判率为:

$$f^{BF}(m,k,n_\oplus)=(t_\oplus)^k=(e^{-kn_a/m}+e^{-kn_b/m} - 2e^{-k(n_a+n_b-n_\cap)/m})^k \quad (11)$$

$x \in S_a \oplus S_b$ 使用 $BF^{k,m}(S_a) \oplus BF^{k,m}(S_b)$ 查询出现假阴性误判, 是因为元素 x 对应 k 个位置, 出现对应位置本该为 1 而错误置为 0 的情况发生. $BF^{k,m}(S_a+S_b)$ 任一位置为 1 的概率为:

$$t=(1-(1-1/m)^{k|n_a+n_b-2n_a|}) \approx 1-e^{-k(n_a+n_b-2n_\cap)/m} \quad (12)$$

则 $BF^{k,m}(S_a) \oplus BF^{k,m}(S_b)$ 进行元素查询出现假阴性误判的概率为:

$$f_{neg}(m,k,n_\oplus)=1-(1-t(1-t_\oplus))^k \quad (13)$$

因为 $0 \leqslant (1-t(1-t_\oplus)) \leqslant 1$, $f_{neg}(m,k,n_\oplus)$ 过大, 导致查询失效. 不能用 $BF^{k,m}(S_a) \oplus BF^{k,m}(S_b)$ 替代 $BF^{k,m}(S_a \oplus S_b)$ 完成 $S_a \oplus S_b$ 的元素查询, 第 5 节的仿真实验进行相关验证.

4.4 布鲁姆过滤器补运算查询算法

定理 6 对于全集 U 中任意的子集 S_a, $BF^{k,m}(S_a)$ 为它的布鲁姆过滤器表示, 则集合 S_a 补集的布鲁姆过滤器表示强于它的布鲁姆过滤器的补, 即:

$$BF^{k,m}(\overline{S_a}) \geqslant \overline{BF^{k,m}(S_a)} \quad (14)$$

定理证明和布鲁姆过滤器差运算的查询误判率分析类似于 4.3 节, 略.

4.5 布鲁姆过滤器差运算查询算法

定理 7 对于全集 U 中任意的子集 S_a 和 S_b, $BF^{k,m}(S_a)$ 和 $BF^{k,m}(S_b)$ 分别为它们的布鲁姆过滤器表示, 则集合 S_a 和 S_b 的差的布鲁姆过滤器表示强于它们的布鲁姆过滤器的差, 即:

$$BF^{k,m}(S_a-S_b) \geqslant BF^{k,m}(S_a)-BF^{k,m}(S_b) \quad (15)$$

定理证明和布鲁姆过滤器差运算的查询误判率分析类似于 4.3 节, 略.

5 实验与性能分析

下面进行仿真实验评估验证布鲁姆过滤器代数运算的查询性能. 实验中, 元素的插入和查询直接在 PC 机上完成. 为了简化实验过程, 采用 32bit 整数作为集合的元素, 数据集合的元素是由计算机随机产生的 32bit 的无符号整数, 元素范围为 $\{0, 2^{32}-1\}$.

理论上任何分布均匀的哈希函数均可以完成, 实验中我们使用 $H3$ 哈希函数作为实现函数. 随机产生全集和两个数据子集合 S_1, S_2, 全集规模为 n, 两子集合规模分别为 n_1 和 n_2, 交集接规模为 $n_\cap$. 理论值是通过上述公式直接计算而得. 查询假阳性误判率实验值是通过统计 10,000 个不在对应集合运算后的集合的元素, 通过 k 个哈希函数, 完成元素查询, 统计被误判的元素个数, 再将此统计值除以 10,000; 查询假阴性误判率通过统计理应在集合中的元素查询误判个数占集合规模的比率. 对于每个实验过程和实验参数的组合, 随机产生 100 次数据集合, 完成 100 次实验, 实验结果取 100 次的平均值. 部分实验结果如下:

表 1 布鲁姆过滤器代数运算查询假阳性误判率

k	m	n	n_1	n_2	$n_\cap$	BF_{or}		sBF_{or}		BF_{and}		sBF_{and}		BF_{xor}		sBF_{xor}		BF_{not}	sBF_{not}	BF_{minus}	sBF_{minu}
						仿真值	理论值	仿真值	理论值	仿真值	理论值	仿真值	理论值	仿真值	理论值	仿真值	理论值	仿真值	仿真值	仿真值	仿真值
4	131072	20000	10000	10000	5000	0.0182	0.0182	0.0182	0.0182	0.0006	0.0006	0.0004	0.0004	0.0019	0.0019	0.0048	0.0048	0.0014	0.0049	1E-04	0.0004
	65536	20000	5000	5000	1000	0.0318	0.0319	0.0318	0.0319	0.0001	0.0001	1E-05	1.2E-05	0.0103	0.0104	0.0221	0.0223	0.0382	0.1294	6E-04	0.0022
8	131072	20000	8000	8000	1000	0.0168	0.0167	0.0168	0.0167	1E-06	1E-06	0	0	0.0011	0.0011	0.012	0.0119	0.0001	0.0052	5E-06	0.0002
	131072	20000	10000	10000	1000	0.0493	0.0493	0.0493	0.0493	2E-06	7E-06	0	0	0.002	0.002	0.039	0.039	1E-05	0.0018	2E-05	0.001

表 2 布鲁姆过滤器代数运算假阴性误判率

k	m	n	n_1	n_2	$n_\cap$	BF_{or}	BF_{or}	BF_{and}	sBF_{and}	sBF_{xors}		sBF_{xor}	BF_{nots}	sBF_{not}	BF_{minuss}	sBF_{minus}
										仿真值	理论值					
4	131072	20000	10000	10000	5000	0	0	0	0	0.7049	0.6068	0	0.7048	0	0.7056	0
	65536	20000	5000	5000	1000	0	0	0	0	0.705	0.7049	0	0.7044	0	0.7057	0
8	131072	20000	8000	8000	1000	0	0	0	0	0.9797	0.9591	0	0.98	0	0.9795	0
	131072	20000	10000	10000	1000	0	0	0	0	0.9923	0.9721	0	0.9923	0	0.9923	0

(1) 表 1 和表 2 可以看出, 使用布鲁姆过滤器并运算得到的 BF_{or} 比直接用表示集合并集的 sBF_{or} 相比, 查询假阳性误判率相同, 并不产生假阴性误判. 验证定理 2 内容. 可以用布鲁姆过滤器并运算的位串完成集合并集的元素查询.

(2) 从表 1 和表 2 可以看出: 使用布鲁姆过滤器交运算得到 BF_{and}, 与直接表示集合交集的布鲁姆过滤器 sBF_{and} 查询性能相比: 增加稍许假阳性误判率, 不产生假阴性误判. 使用 BF_{and} 可以完成集合交集的元素查询, 但是会增加稍许假阳性误判. 验证了定理 3 和定理 4 结论.

(3) 表 1 和表 2 可以看出: 使用布鲁姆过滤器异或运算得到 BF_{xor}, 它和 sBF_{xor} 相比, 虽然会降低稍许假阳性概率, 但是增加了假阴性误判概率, 而且引入的假阴

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

性概率比较大. 因此, 不能使用 BF_{xor} 完成集合异或集的元素查询, 在考察补运算和差运算时, 也是由于过多的引入了查询假阴性误判, 不能使用 BF_{not} 和 BF_{minus} 来完成集合补集和集合差集的元素查询.

因此, 直接通过过滤器的并即可完成集合并集的元素查询, 直接使用过滤器交即可完成集合交集的元素查询, 这是布鲁姆过滤器代数运算的使用技巧, 如例 1 所示.

例**1**　全集 U 中的两子集 $S_1=\{a, b, c\}$ 和 $S_2=\{c, d, e\}$, 它们的布鲁姆过滤器表示分别为 $BF^{k, m}(S_1)$ 和 $BF^{k, m}(S_2)$, 如图 4 所示. 查询元素 a 是否在并集 $S=S_1 \cup S_2$ 中, 利用 $BF^{k, m}(S_1) \cup BF^{k, m}(S_2)$ 可以直接得到 a 在集合的结论; 查询 f 是否在集合中, $BF^{k, m}(S_1) \cup BF^{k, m}(S_2)$ 可以得出 f 不在集合的结论. 查询元素 g 是否在交集 $S=S_1 \cap S_2$ 中, 利用 $BF^{k, m}(S_1) \cap BF^{k, m}(S_2)$ 可以得出 g 不在集合中的结论; 查询 c 是否在集合交集 $S=S_1 \cap S_2$ 中, $BF^{k, m}(S_1) \cap BF^{k, m}(S_2)$, 得出 c 在集合交集 $S=S_1 \cap S_2$ 的结论.

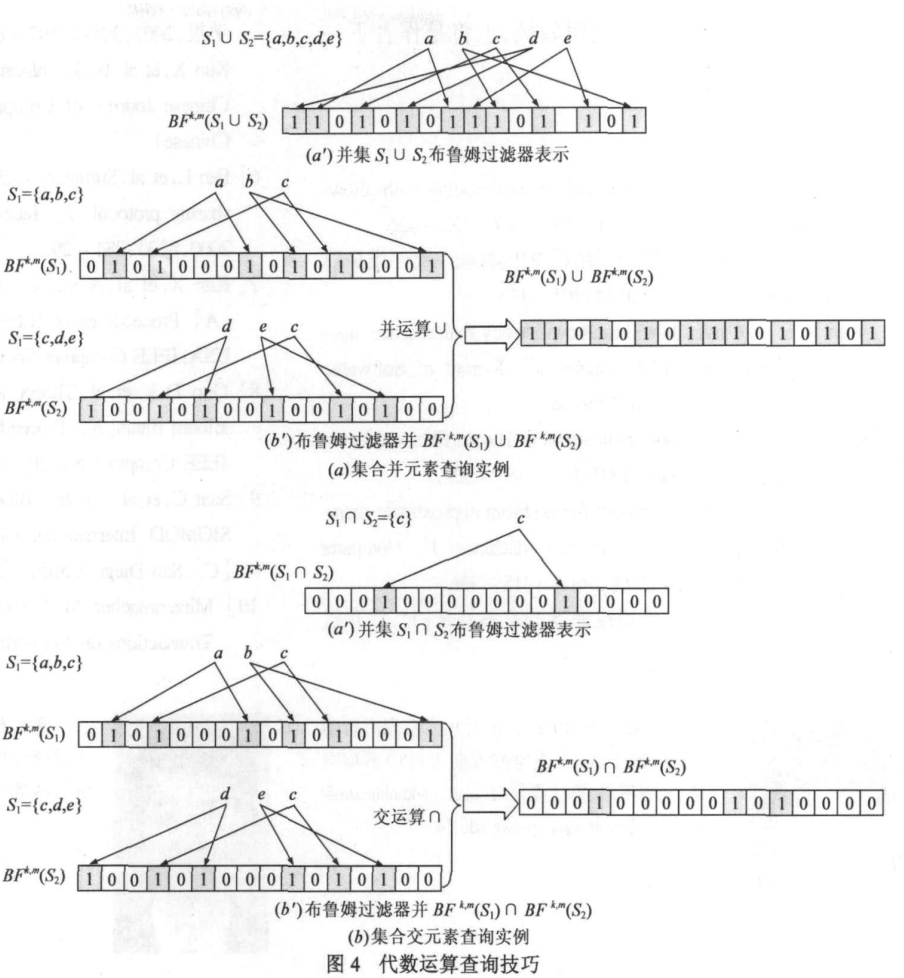

图 4　代数运算查询技巧

布鲁姆过滤器在实际中通常是用来表示具体应用中的数据集合, 借助布鲁姆过滤器完成集合查询操作. 而在实际的应用中, 查询可能又会涉及集合相关操作的查询, 利用布鲁姆过滤器代数运算, 直接通过运算后的布鲁姆过滤器完成查询操作, 与直接使用两个原始的布鲁姆过滤器来说, 可节约哈希计算次数. 而且, 本文的工作也提供了一条处理集合的思路, 尤其对于一些转瞬即逝的流数据集合, 数据集合具有不可重现性, 我们可以利用布鲁姆过滤器的代数运算反应出来的集合的关系, 完成相关的集合查询操作, 这在实际中有积极的意义.

6　小节和研究展望

本文直接从布鲁姆过滤器位串向量的串运算和操作出发, 探讨了布鲁姆过滤器的布尔代数, 定义了布鲁姆过滤器的并、交、异或、补、差等代数运算. 从理论和实验的角度探讨布鲁姆过滤器代数运算和集合查询的关系, 证明了布鲁姆过滤器的并、交仍然可以支持集合元素查询, 而其他运算由于引入过多的假阴性误判而不再支持集合元素查询. 本文的结论可以简化利用布鲁姆过滤器进行的系统设计, 可用于大多数布鲁姆过滤器的应用场合, 为优化布鲁姆过滤器应用提供新的思路.

本文围绕同源布鲁姆过滤器展开代数运算探讨, 通过理论分析和实验获得和验证布鲁姆过滤器并、交、异或、补、差等代数的相关性质. 进一步探讨不同长度, 不同哈希函数下的非同源布鲁姆过滤器的如何进行相关代数运算, 成为作者的进一步研究工作: 如 $2\times m$ 长的过滤器如何与 $1\times m$ 长的过滤器进行运算来支持集合查询, 这对于将布鲁姆过滤器的代数运算扩展到动态的可扩展数据集合的查询操作有积极的意义. 另一个重要的工作展望就是将本文的研究成果应用于实际系统中, 在文献[2] 中, 我们借助布鲁姆过滤器“并”运算, 提出了一种非结构化的 P2P 一致性维护算法, 但是, 如何详细地描述布鲁姆过滤器在某种应用中的实

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

际效果,并扩展算法到新的应用领域,也将是作者下一步的工作重点.

参考文献:

[1] Burton H B. Space/time trade-offs in hash coding with allowable errors[J]. Commun. ACM, 1970, 13(7): 422- 426.

[2] 谢鲲,等. 基于轨迹标签的无结构 P2P 副本一致性维护算法[J]. 软件学报, 2007, 18(1): 105- 116.
Kun X, et al. A trace label based consistency maintenance algorithm in unstructured P2P systems[J]. Journal of Software, 2007, 18(1): 105- 116. (in Chinese)

[3] Broder A, et al. Network applications of Bloom filters: a survey [J]. Internet Mathematics, 2003, 1(4): 485- 509.

[4] Bonomi F, et al. Beyond bloom filters: From approximate membership checks to approximate state machines[J]. Computer Communication Review, 2006, 36(4): 315- 326.

[5] 谢鲲,等. 基于分档布鲁姆过滤器的查询算法[J]. 计算机学报, 2007, 30(4): 597- 607.
Kun X, et al. Basket bloom filters for membership queries[J]. Chinese Journal of Computers, 2007, 30(4): 597- 607. (in Chinese)

[6] Fan L, et al. Summary cache: A scalable wide-area Web cache sharing protocol[J]. Ieee-Acm Transactions on Networking, 2000, 8(3): 281- 293.

[7] Kun X, et al. A scalable bloom filter for membership queries [A]. Proceedings of IEEE Globecom[C]. Washington D. C. USA: IEEE Computer Society, 2007.

[8] Guo D k, et al. Theory and network application of dynamic Bloom filters[A]. Proceedings of IEEE Infocom[C]. SPAIN: IEEE Computer Society, 2006. 1- 10.

[9] Saar C, et al. Spectral Bloom filters[A]. Proceedings of ACM SIGMOD International Conference on Management of Data [C]. San Diego, California: ACM Press, 2003. 241- 252.

[10] Mitzenmacher M. Compressed bloom filters[J]. Ieee-Acm Transactions on Networking, 2002, 10(5): 604- 612.

作者简介:

谢 鲲 女, 1978 年 10 月出生于湖南省黔阳,博士,博士后. 主要研究方向为可信系统与网络、无线网络、算法设计. E-mail: xiekunkunxie@163.com; cskxie@comp.polyu.edu.hk

张大方 男, 1959 年 4 月出生于上海,博士,教授,博士生导师. 主要研究方向为可信系统与网络、网络测试、软件容错、软件测试.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net