ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.21, No.4, April 2010, pp.672−679
doi: 10.3724/SP.J.1001.2010.03430

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

基于 Hamming 范数的 XML 流相关性估测算法*

孙 贺[1+], 朱 洪[2]

[1](复旦大学 上海市智能信息处理重点实验室,上海 200433)

[2](华东师范大学 上海市高可信计算重点实验室,上海 200062)

Correlation Estimating Algorithm of XML Stream Based on Hamming Norms

SUN He[1+], ZHU Hong[2]

[1](Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai 200433, China)

[2](Shanghai Key Laboratory of Trustworthy Computing, East China Normal University, Shanghai 200062, China)

+ Corresponding author: E-mail: sunhe@fudan.edu.cn, http://www.tcs.fudan.edu.cn/~sun

Sun H, Zhu H. Correlation estimating algorithm of XML stream based on Hamming norms. *Journal of Software*, 2010,21(4):672−679. http://www.jos.org.cn/1000-9825/3430.htm

Abstract: It is of great importance to compare the correlation of different XML (extensible markup language) streams in the limited space in the Database Theory. In the study of these problems, several measures are proposed, e.g. the tree-edit distance, to show the difference of XML trees. This paper proposes a natural measure l_0 employing Hamming norms, i.e. the number of distinct sub-trees between two XML trees, to estimate the correlation. Furthermore, a probabilistic estimating algorithm involving space-bounded pseudorandom generators, stable distributions and hash functions has been presented in the data stream model. Theoretical time/space complexity analysis, correctness proof and experimental simulation show that this algorithm can give a desired approximation.

Key words: algorithm design; data stream; Hamming norm; stable distribution; XML (extensible markup language)

摘 要: 在数据库理论中,如何在较小的空间条件下快速地比较不同的 XML(extensible markup language)流的差异性是一个基本问题.在这一问题的研究中,人们提出了树编辑距离等测度来描述 XML 文本的差异性.提出了一种基于 Hamming 范数的 l_0 测度——即 XML 树的不同子树的个数,并以此来刻画 XML 文本的相关性.在数据流模型下,给出了基于空间有界伪随机数发生器、稳态分布于哈希函数的 l_0 测度的概率算法.理论上的时空复杂性分析、正确性证明与实验模拟结果表明,这一概率算法对问题的输入提供了一个理想的近似.

关键词: 算法设计;数据流;Hamming 范数;稳态分布;XML(extensible markup language)

中图法分类号: TP311 **文献标识码**: A

可扩展标记语言 XML(extensible markup language)是一种可以用来创建自己标记的标记语言.随着互联网技术的发展,目前,XML 已成为数据表示和交换的标准.在大多数情况下,XML 文档是一个半结构化的、有序、

* Supported by the National High-Tech Research and Development Plan of China under Grant No.2007AA01Z189 (国家高技术研究发展计划(863)); the Shanghai Leading Academic Discipline Project of China under Grant No.B412 (上海重点学科建设项目资助)

Received 2007-12-17; Revised 2008-03-27; Accepted 2008-07-02

带标签的树.在 Internet 中,为了提取出不同文档和数据库中的有用信息,一个自然的问题是如何在较小的空间条件下快速地衡量不同的 XML 文档的差异.

本文首先提出了定义不同的 XML 文档相似性的 Hamming 范数 l_0.对于任意两棵 XML 树,本文给出了计算 XML 文档 Hamming 范数的近似算法,并对算法的复杂性进行分析.

相关工作

在数据库理论和互联网技术的研究中,经常会遇到如下一些问题:

(1) 在数据库理论中:人们通常需要迅速获得一个大型数据库中的某些统计信息,比如:一个数据库中共含有多少条记录、在一个数据项中不同的记录总数等.这些记录为数据库的查询优化提供了重要的指标.我们知道,一个复杂的数据库查询操作一般包含一系列的集合运算,例如映射、连接等.因此,在查询操作之前,首先快速知道一个数据库的基数,可以帮助我们选择一个高效的查找策略,以此来优化数据库的查询操作;

(2) 在互联网技术中:由于网络安全的需要,一个路由器往往需要对特定时间段的 IP 地址进行统计,比如:有多少个不同的 IP 地址、哪些 IP 地址的访问量较大,等等.

从上面两个例子不难看出,数据流模型具有如下几个共同点:

(1) 对数据流中的元素进行少量的几次扫描,通常为一次扫描;

(2) 使用较小的空间,一般来讲空间为 $O(\text{polylog}U)$,其中,U 表示问题规模;

(3) 只需给出精确解的理想近似即可.

这些问题的广泛存在以及与一般算法设计截然不同的约束条件,使人们开始了对数据流模型的研究,这一研究起源于 20 世纪 80 年代由 Flajolet 完成的奠基性工作.在文献[1]中,Flajolet 首次提出了数据流的基本模型,形式化地表述了数据流中频率矩 F_0 估测的问题,并给出了解决该问题的概率算法.

1999 年,Alonn,Matias 和 Szegedy 对文献[1]中的问题进行了进一步研究.在文献[2]中,他们证明了运用两两独立哈希函数(pairwise independent hash function)来取代文献[1]中算法设计所假设的完美哈希函数,就可以给出 F_0 的常数近似比算法.此外,他们对数据流模型中的频率矩 $F_k(k=0,1,2,\ldots)$估测问题给出了近似算法,分析了在理想近似比下概率算法的空间复杂度下界.这些开创性工作使 3 位作者获得了 2005 年度的 Gödel 奖.

在文献[1]所做工作之后的 20 余年里,由于关系数据库理论、网络路由技术、计算几何等领域的需要,研究人员从不同的角度对数据流模型下的众多问题展开了全面研究.随着 XML 技术的广泛应用和实际需要,研究人员希望量化不同 XML 文本的相似程度,并给出在数据流模型下的算法.其中,文献[3]设计了一种针对改进的计算树编辑距离的近似算法.在文献[4]中,Polyzotis 等人指出了树编辑距离无法较好地反映 XML 树的相似性;在同一篇论文中,一种根据文档的边分布和全局路径结构的测度标准被提了出来.但是,这种测度不够直观和简洁.

本文的主要工作

在本文中,我们定义了一个基于 Hamming 范数的 XML 文本相似性测度.这一测度提出的思想如下:XML 文本用一棵有序标号树 T 来表示数据之间的层次结构和从属关系,数据之间的部分从属关系则由树 T 中从任一点 v 开始 v 以下的所有后代形成的子树来描述.由于树中完全相同的子树可能多次出现,例如在一个表示出版信息的 XML 树 T 中,作者信息用一个以〈Author〉为根的子树来表示,而该子树可能出现在 T 的〈Book〉,〈Article〉等多个子树中,树中信息在结构上存在着一定的冗余,如图 1 所示.因此,两个 XML 树中不同子树的个数便很自然地反映了 XML 文本的相关性.

考虑到两个向量 $\boldsymbol{a}$ 和 $\boldsymbol{b}$ 的 Hamming 距离 l_0 定义为$|\boldsymbol{a}-\boldsymbol{b}|_H=|\{i|a_i\neq b_i\}|$,我们将两棵树的 Hamming 距离定义如下:设 S^*,T^*为树 S 和 T 的任一子树,则树 S 和 T 的 Hamming 距离 l_0 定义为$||\{S^*|S^*\subseteq S\}\oplus\{T^*|T^*\subseteq T\}||$,其中,$S^*\subseteq S$ 表示 S^*是 S 的子树.特别地,我们将 XML 文本中基于 Hamming 范数所定义的测度称为 XML 流的 l_0 测度.

在本文中,我们借助 Nisan 伪随机数发生器、稳态分布和 KR(Karp-Rabin)签名算法等复杂性理论和算法设计技巧,设计了计算不同 XML 文档 l_0 测度的近似算法,并对算法的正确性、时空复杂性进行了分析.

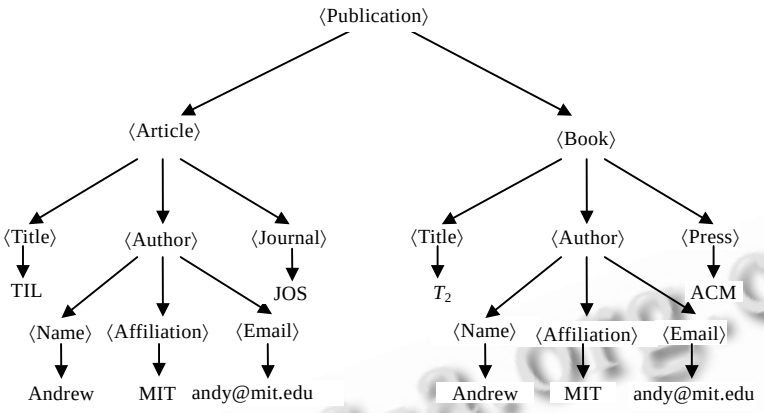

Fig.1 Examples of an XML tree

图 1 XML 树的实例

1 预备知识

1.1 数据流中的l_p范数估测

在数据流模型中,设输入数据流 S 为 $s_1s_2\ldots s_n$,流中的每个元素的形式为(i,a),其中,$i\in\{1,\ldots,U\}$,$a\in\{-1,+1\}$表示对数据 i 进行插入或删除操作,则数据流的 l_p 范数定义为 $l_p(S)=(\sum_{i\in[1,U]}|\sum_{(i,a)\in S}a|^p)^{1/p}$.由于数据流算法允许使用的空间有限,因此一般来讲,算法只需给出一个理想近似即可.具体来讲,对于给定的参数$\varepsilon,\delta>0$,若算法 A 的输出值 Z^*与精确值 Z 满足不等式 $\Pr[|Z^*-Z|>\varepsilon Z]<\delta$,则称算法 A 是对数值 Z 的一个(ε,δ)-近似.在本文中,算法的输出 Z 即为两棵 XML 树的 Hamming 距离.

由 l_p 的定义可知,l_0 范数就是数据流中不同的元素个数.这一问题在数据流算法的设计中占有重要的地位,在其研究中,人们发现了两种不同的算法设计方案:

第1种方案称为取样(sample)技术,参见文献[5,6].这种方法的基本思想如下:由于在数据流模型下无法存储所有输入元素,那么如果数据流中的元素是服从均匀分布并满足一定的独立性,则数据流中的部分元素就可以理想地反映所有元素的统计特性.因此,在基于取样技术的数据流算法设计中,需要解决的基本问题包括:

(1) 如何选取合适的哈希函数,以使通过哈希函数的数据呈现出一定的概率特性;

(2) 在数据流中选取哪些元素保留到取样集中?

另一种算法设计的方案被称为摘要(sketch)技术.与取样技术不同,基于摘要的算法并不直接存储数据流中的若干元素,而是利用数据流模型所允许的空间来存储数据流中元素的特定的统计信息.

本文所提出的数据流算法依赖于 Indyk 所提出的基于稳态分布的摘要设计技术.2000 年,Indyk 在那篇影响深远的文献[7]中首先揭示了稳态分布与数据流 $l_p(p=1,2)$估测问题的联系.2002 年,文献[8]扩展了 Indyk 在 2000 年提出的方法,并给出了数据流中 l_p 估测问题的(ε,δ)-近似算法.

1.2 稳态分布

稳态分布(stable distribution)是由 Lévy 于 20 世纪 20 年代在对若干独立同分布的随机变量的和的研究中引入的,其形式化定义如下.

定义 1(稳态分布). 对于给定的稳定指标$\alpha\in(0,2)$,偏斜因子$\beta\in[-1,1]$,衡量因子$\sigma>0$ 和位置参数$\mu\in\mathbb{R}$,如果随机变量 X 的特征函数满足:

$$\ln\phi(t)=\begin{cases}-\sigma^{\alpha}|t|^{\alpha}\left\{1-i\beta sign(t)\tan\dfrac{\pi\alpha}{2}\right\}+i\mu t, & \alpha\neq 1\\ -\sigma|t|\left\{1+i\beta sign(t)\dfrac{2}{\pi}\ln|t|\right\}+i\mu t, & \alpha=1\end{cases},$$

则称 X 服从由α,β,σ,μ决定的稳态分布,记为 $X\sim S_{\alpha}(\sigma,\beta,\mu)$.

非形式地说,对于$\alpha\geq 0$,称一个变量 X 为α-稳态,如果对于任意的 n 个实数 $a_1,\ldots,a_n\in\mathbb{R}$及服从分布 D 的独立的随机变量 $X_1,\ldots,X_n$,随机变量 $a_1X_1+\ldots+a_nX_n$ 与 $(\sum_{i=1}^{n}|a_i|^{\alpha})^{1/\alpha}X$ 同分布,其中 $X\sim D$.

对于$\alpha\in(0,2)$,目前已知$\alpha=1/2$时的分布为 Lévy 分布,$\alpha=1$时的分布为柯西分布,$\alpha=2$时的情况对应高斯分布.一般的α值对应的分布不存在紧的密度函数表示.

引理 1[9]. 若 $X\sim S_{\alpha}(\sigma,\beta,\mu)$,则对于任意常数 c,有

$$\lim_{\alpha\to 0^{+}} median(|cX|^{\alpha})=|c|^{\alpha}\,median(|X|^{\alpha})=\frac{|c|^{\alpha}}{\ln 2},$$

其中,*median* 为取中间值函数.

1996 年,Weron 给出了生成满足任意参数的稳态分布 $Y\sim S_{\alpha}(\sigma,\beta,\mu)$的算法,其算法的形式化描述如下:

算法 1(Weron 算法).

算法输入:(0,1)中满足均匀分布的两个随机变量 r_1,r_2 及参数α,β,μ,σ.

算法输出:满足 $S_{\alpha}(\sigma,\beta,\mu)$分布的随机变量 Y.

算法描述:

1. 选取两个在(0,1)上均匀分布的随机变量 r_1,r_2,并计算 $V=\pi\left(r_1-\dfrac{1}{2}\right)$.

2. 作随机变量 $X(r_1,r_2)$

$$X=\begin{cases}S_{\alpha,\beta}\dfrac{\sin\{\alpha(V+B_{\alpha,\beta})\}}{\cos^{1/\alpha}V}\left[\dfrac{\cos\{V-\alpha(V+B_{\alpha,\beta})\}}{\ln r_2}\right]^{(1-\alpha)/\alpha}, & \alpha\neq 1\\ \dfrac{2}{\pi}\left(\dfrac{\pi}{2}+\beta V\right)\tan V-\beta\ln\left(\dfrac{-\dfrac{\pi}{2}\ln r_2\cos V}{\dfrac{\pi}{2}+\beta V}\right), & \alpha=1\end{cases},$$

其中, $B_{\alpha,\beta}=\dfrac{\arctan\left(\beta\tan\dfrac{\pi\alpha}{2}\right)}{\alpha}$, $S_{\alpha,\beta}=\left\{1+\beta^2\tan^2\left(\dfrac{\pi\alpha}{2}\right)\right\}^{1/2\alpha}$.

3. 作随机变量 $Y(r_1,r_2)$

$$Y=\begin{cases}\sigma X+\mu, & \alpha\neq 1\\ \sigma X+\dfrac{2}{\pi}\beta\sigma\ln\sigma+\mu, & \alpha=1\end{cases}.$$

Weron 在一份技术报告中证明了如上算法所构造的随机变量 Y 满足 $S_{\alpha}(\sigma,\beta,\mu)$.并且,这一算法的时间和空间复杂度均为 $O(1)$.

关于稳态分布的数学知识,可参考文献[10].

1.3 伪随机数发生器

伪随机数发生器这一概念起源于 20 世纪 80 年代 Blum 和 Yao 的开创性工作[11,12].简单地说,一个伪随机数发生器是一个可将“较短”的随机序列扩展为“较长”的近似随机序列的确定性多项式时间算法,其形式化定义如下:

定义 2(伪随机数发生器). 一个多项式时间算法 G 称为伪随机数发生器,若存在延展函数$\ell:\mathbb{N}\mapsto\mathbb{N}$,使得如下

两个概率总体 $\{G_n\}_{n\in\mathbb{N}}$ 和 $\{U_n\}_{n\in\mathbb{N}}$ 是计算不可区分的,其中:

(1) 概率分布 G_n 是长度为 $\ell(n)$的算法 G 的输出,其中输入序列均匀地取自$\{0,1\}^n$.

(2) 概率分布 U_n 是长度为 $\ell(n)>n$ 的均匀分布 $\{0,1\}^{\ell(n)}$, $\ell(n)>n$.

关于有限空间下的伪随机数发生器生成算法,Nisan 在文献[13]中证明了如下定理:

定理 1(Nisan,1990). 存在空间为 $O(S)$、参数为$\varepsilon=2^{-O(S)}$的伪随机数发生器 G,使得:(1) G 将 $O(S\log R)$位输入扩展为 $O(R)$;(2) G 的存储空间为 $O(S)$.

2 概率算法描述

在下面的讨论中,我们用$|S|$来表示树 S 的顶点个数.此外,树 S 和 T 的深度分别用 d_s 和 d_t 表示,且 $d=\max\{d_s,d_t\}$.在实际的 XML 数据集中,我们可作合理假设 $d=O(\text{polylog}U)$,$U=\max\{|S|,|T|\}$且树宽小于 d.

算法 2(XML 流相关性估测算法).

输入数据:两个 XML 数据流 $S=s_1s_2\ldots$,$T=t_1t_2\ldots$,分别表示 XML 树 S 和 T,数据流 S 和 T 中的顶点以前序遍历、从左到右的次序给出.

输出数据:XML 树 S 和 T 的 Hamming 距离.

算法描述:

1. 在初始化阶段,算法执行如下操作:
   (1) 随机构造一个 $m=O\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)$行 $k=O(U^c)$列的矩阵 $\boldsymbol{B}$,矩阵中的每个元素 $b_{i,j}$服从稳定参数为α的稳态分布.其中,$c\in\mathbb{N}$,$0<\alpha<1$ 为常数.用 $\boldsymbol{B}(i)$代表矩阵 $\boldsymbol{B}$ 的第 i 列.
   (2) 初始化一个包含 m 个元素的零向量 $\boldsymbol{SK}$,作为流算法的摘要来存放数据流的统计信息.
   (3) 构造两个规模为 $O(d)$的堆栈 *StackS* 和 *StackT*.
2. 当数据流中的元素依次到来时,算法用 SAX(simple API for XML)对 XML 数据流进行解析,在此过程中,算法根据到来元素的不同类型,执行下列操作:
   (1) 若解析器遇见树 S 的开始标签$\langle e_s\rangle$,则将字符串"$\langle e_s\rangle$("压入堆栈 *StackS*.
   (2) 若解析器遇见树 T 的开始标签$\langle e_t\rangle$,则将字符串"$\langle e_t\rangle$("压入堆栈 *StackT*.
   (3) 若解析器遇见树 S 的结束标签$\langle /e_s\rangle$,则执行如下操作:① ")$\langle /e_s\rangle$"入栈;堆栈中的元素依次出栈,直到遇到相应的元素开始标签$\langle e_s\rangle$为止,设出栈元素的逆序$\langle e_s\rangle\ldots\langle /e_s\rangle$为 *String_Stack_S*;② 运用哈希函数$\sigma:\Sigma^*\to\mathbb{N}$计算字符串 *String_Stack_S* 的指纹 $Sig:=\sigma(String\_Stack\_S)$,并将 *Sig* 压入栈中.Karp-Rabin 算法[8]所使用的函数等都可作为本算法的指纹函数;③ $\boldsymbol{SK}\leftarrow\boldsymbol{SK}+\boldsymbol{B}(Sig)$.
   (4) 若解析器遇见树 T 的结束标签$\langle /e_t\rangle$,则执行如下操作:① ")$\langle /e_t\rangle$"入栈;堆栈中的元素依次出栈,直到遇到相应元素的开始标签$\langle e_t\rangle$为止,设出栈元素的逆序$\langle e_t\rangle\ldots\langle /e_t\rangle$为 *String_Stack_T*;② 运用函数$\sigma:\Sigma^*\to\mathbb{N}$计算字符串 *String_Stack_T* 的指纹 $Sig:=\sigma(String\_Stack\_T)$,并将 *Sig* 压入栈中;③ $\boldsymbol{SK}\leftarrow\boldsymbol{SK}-\boldsymbol{B}(Sig)$.
3. 当询问 XML 流的相关性时,算法输出值 $Z^*=\ln 2\cdot(median\{\boldsymbol{SK}(1),\boldsymbol{SK}(2),\ldots,\boldsymbol{SK}(m)\})^{\alpha}$,其中,*median* 表示取中间值元素的函数.

3 算法复杂度分析与正确性证明

定理 2. 算法的空间复杂度为 $O\left(d\log U+\frac{1}{\varepsilon^2}\log U\log\frac{1}{\delta}\right)$.

证明:算法的空间需求有两个方面:首先,算法需要存储堆栈 *StackS* 和 *StackT* 中的元素.因为树 S 和 T 的最大顶点数为 U 且指纹函数的值域为$[0,k]$,因此存储每一个顶点只需要空间 $O(\log U)$.又因为树的最大深度为 d,所以存储路径的空间需求为 $O(d\log U)$.

其次,我们需要计算摘要 ***SK*** 和矩阵 ***B*** 的空间需求.由于 ***SK*** 中有 $m=O\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)$ 个元素,矩阵 ***B*** 中含有 $m\times k$ 个元素.由数据流算法的空间上界 $O(\text{polylog}U)$可知,我们无法将矩阵 ***B*** 的所有元素事先完整地存放在内存中,因此我们采取的方法是:当需要矩阵 ***B*** 的某一列 $\boldsymbol{B}_j$ 时,在线随机生成该列的所有元素.由于 $\boldsymbol{B}_j$ 可能会多次使用,因此算法需保证:对于 ***B*** 中任一固定的列 $\boldsymbol{B}_j$,每次生成的元素必须相同.为了达到这个目的,我们使用 Nisan 伪随机数发生器算法.这一算法的优势在于:(1) 只需要 $O(\log U)$的空间即可生成所需要的伪随机数;(2) 当需要 $\boldsymbol{B}_j$ 列的元素时,以 j 为参数调用 Nisan 伪随机数发生器算法,使得每次生成的 $\boldsymbol{B}_j$ 中的元素相同.

这一技术的具体实现如下:不失一般性,我们只对树 T 进行讨论.对于任一点 v,当以 v 为根的完全子树 T^*出现时,需要根据矩阵 $\boldsymbol{B}_j$ 列的值更新摘要 ***SK***,其中 $j=\sigma(String\_Stack\_T)$.因为矩阵 ***B*** 的列数为 $O(U^c)$,在流模型下无法完整地将矩阵 ***B*** 保存在内存中.这里,我们采用文献[4]的方法.在算法执行的过程中,算法只保存摘要 ***SK*** 中 m 个元素的值和两个深度为 $O(\log U)$的堆栈信息.当 v 的完全子树 T^*出现时,算法执行如下操作:(1) 以 $\sigma(String_Stack_T)$为参数,利用 Nisan 伪随机数发生器来生成 $2m$ 个随机数组成的序列 A;(2) 以序列 A 中相邻的两个数 $a_{2t-1},a_{2t}(1\le t\le m)$为参数,运用 Weron 算法,在常数的时间和空间内生成满足 α-稳定系数的随机变量,作为 ***B*** 中第 t 行第$\sigma(String_Stack_T)$列的值.因此,这一部分的空间需求为 $O\left(\frac{1}{\varepsilon^2}\log U\log\frac{1}{\delta}\right)$.

综合两部分的空间需求可知,算法的空间复杂度为 $O\left(d\log U+\frac{1}{\varepsilon^2}\log U\log\frac{1}{\delta}\right)$. □

定理 3. 数据流中算法对每个元素的更新时间为 $O\left(d\log U+\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)$.

证明:当数据流中的每个元素到来时,我们分情况讨论更新时间.不失一般性,假设到来的元素在树 T 中.

情况 1. 如果到来元素为$\langle e_t\rangle$,则将"$\langle e_t\rangle$ ("放入堆栈 *StackT* 中.在此情况下,算法对元素的更新时间为 $O(1)$.

情况 2. 如果到来元素为$\langle/e_t\rangle$,此时新元素的到来使得 T 形成了一个以$\langle e_t\rangle$为根的子树 T^*,算法应执行如下操作:(1) 堆栈中的若干元素退栈.由于堆栈中的元素总数为 $O(d)$,因此这一步的时间复杂度为 $O(d)$;(2) 计算子树 T^*的指纹.由文献[10]的算法描述可知,这一步的时间复杂度为 $O(d\log U)$;(3) 更新摘要 ***SK*** 中元素的值.由 Nisan 伪随机数发生器的构造可知,算法可在 $O\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)$ 的时间内生成所需要的服从均匀分布的随机数.利用 Weron 算法,我们可以在线性时间里产生 m 个服从稳定系数α的随机变量,并更新向量 ***SK*** 的值.因此,时间复杂度为 $O\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)$.

综上所述,算法对每个元素的更新时间为 $O\left(d\log U+\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)$. □

在给出算法 2 的正确性证明之前,我们首先介绍 Chernoff/Hoeffding 不等式,通常也称为 Chernoff 不等式.

引理 2(Chernoff/Hoeffding 不等式). 对于完全独立的随机变量 $X_1,\ldots,X_m$ 及参数δ,其中每个随机变量满足 $X_i\in[a,b],i=1,\ldots,m$,如下不等式成立:

$$\Pr\left[\left|\left(\frac{1}{m}\sum_{i=1}^{m}x_i\right)-\left(\frac{1}{m}\sum_{i=1}^{m}E(x_i)\right)\right|\ge\delta\right]\le 2\mathrm{e}^{-\frac{2\delta^2}{(b-a)^2}\cdot m}.$$

证明:不失一般性,我们只讨论 $a=0,b=1$ 时的情况.对任一随机变量 X_i,定义 $p_i=\Pr[X_i=1]$及 $\overline{X_i}=X_i-p_i$,则对任意的$\lambda>0$,有

$$\Pr\left[\frac{1}{m}\sum_{i=1}^{m}\overline{X_i}>\delta\right]=\Pr[\mathrm{e}^{\lambda\sum_{i=1}^{m}\overline{X_i}}>\mathrm{e}^{\lambda\delta m}].$$

由 Markov 不等式,上式可变换成:

$$\Pr\left[\frac{1}{m}\sum_{i=1}^{m}\overline{X_i}>\delta\right]=\frac{E[e^{\lambda\sum_{i=1}^{m}\overline{X_i}}]}{e^{\lambda\delta m}}.$$

由于 $E[e^{\lambda\sum_{i=1}^{m}\overline{X_i}}]=\prod_{i=1}^{m}E[e^{\lambda\overline{X_i}}]<(e^{\lambda^{2/8}})^m$,因此令$\lambda$=4$\delta$,则有

$$\Pr\left[\frac{1}{m}\sum_{i=1}^{m}\overline{X_i}>\delta\right]<e^{-2\delta^2 m}.$$ □

定理 4. 对于给定的近似参数ε和置信参数δ,在忽略哈希函数碰撞概率的条件下,算法返回值 Z^*与精确值 Z 的关系满足:

$$\Pr[|Z^*-Z|>\varepsilon Z]<\delta.$$

证明:不失一般性,只考虑摘要 ***SK*** 和矩阵 ***B*** 的第 i 行元素.易知 $\boldsymbol{SK}(i)=c_1b_{i,1}+\ldots+c_kb_{i,k}$,其中,$c_j$ 为元素 j 出现的次数.

由稳态分布的性质可知:

$$\sum_{j=1}^{k}c_jb_{i,j}\sim(\sum_{i=1}^{k}|c_i|^{\alpha})^{1/\alpha}b^*,$$

其中,b^*与矩阵 ***B*** 中的元素同分布.由引理 1 可知:

$$\lim_{\alpha\to0^+}median(|cb^*|^{\alpha})=|c|^{\alpha}\ median(|b^*|^{\alpha})=\frac{|c|^{\alpha}}{\ln 2}.$$

因此,在算法设计中,只需令稳定系数 $\alpha\le\frac{\log(1+\varepsilon)}{\log U}$,则有 $U^{\alpha}\le\log(1+\varepsilon)$.

另一方面,对于给定的非负实数α,$c_i\in\{1,\ldots,U\}$,我们考察精确值 $Z=\sum_{i=1}^{U}|c_i|^0$ 与摘要 ***SK*** 中的每个 $\boldsymbol{SK}(i)$的关系.不难看出:

$$\ln 2\cdot Z=\ln 2\cdot\sum_{i=1}^{U}|c_i|^0\le\ln 2\cdot\sum_{i=1}^{U}|c_i|^{\alpha}=\ln 2\cdot\boldsymbol{SK}(i)\le\ln 2\cdot\sum_{i=1}^{U}|c_i|^0\ U^{\alpha}\le\ln 2\cdot(1+\varepsilon)Z.$$

设 $Y_1,\ldots,Y_m$ 为 $\boldsymbol{SK}(1),\boldsymbol{SK}(2),\ldots,\boldsymbol{SK}(m)$按照值的递增顺序的一个排列.对于 $1\le i\le m$,作指标随机变量如下:

$$X_i=\begin{cases}1, & Y_i\le Y_{(1/2-\varepsilon)m}\\ 0, & \text{否则}\end{cases},$$

则有 $\Pr[X_i=1]=\frac{\lfloor(1/2-\varepsilon)m\rfloor}{m}\simeq\frac{1}{2}-\varepsilon$, $E[X_i]=\frac{1}{2}-\varepsilon$.因此,

$$\Pr\left[median\{\boldsymbol{SK}(1),\boldsymbol{SK}(2),...,\boldsymbol{SK}(m)\}<\left(\frac{1}{2}-\varepsilon\right)m\right]\le\Pr\left[\sum_{i=1}^{m}X_m>\frac{m}{2}\right]=\Pr\left[\overline{X}>\frac{1}{2}\right],$$

其中,$\overline{X}=\frac{1}{m}\sum_{i=1}^{m}X_m$.

由于 $E(\overline{X})\cdot(1+2\varepsilon)<\frac{1}{2}$,所以在 $m=O\left(\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)$的条件下,由引理 2(Chernoff/Hoeffding 不等式)可得:

$$\Pr\left[\overline{X}>\frac{1}{2}\right]\le\Pr[\overline{X}>E(\overline{X})(1+2\cdot\varepsilon)].$$

因为 $E[X]=\frac{1}{m}\sum_{i=1}^{m}E[X_i]$,我们可得 $\Pr[|m\overline{X}-mE[\overline{X}]>2\varepsilon mE[\overline{X}]|]<\delta$.

因此,根据算法 ***SK*** 中 m 个元素的中间值,即可获得一个理想估计. □

4 结束语

XML 流的相关性比较在数据库应用中有着重要的意义.在本文中,我们提出了一种衡量 XML 文本相关性的新测度.与树编辑距离等测度相比,基于 Hamming 范数的测度更加自然、直观,能够较好地衡量不同 XML 文本的相关性.其次,我们给出了计算 XML 文本 Hamming 测度的近似算法.由于本文中所设计算法的一般性,该算法不仅可以用于 XML 流的比较,而且还可以应用在如下更一般的情况:即在数据流模型下比较两个树的相似

性.以 IBM XMLGenerator 生成的大规模 XML 流作为输入,算法的实验结果显示:随着两个 XML 流 DTD(document type definition)结构信息的逐渐到来,本文提出的算法可以对 XML 流的相关性给出理想估测.

与此同时,我们也注意到:由于 KR 指纹函数等哈希函数的理想特性(参见文献[3]的应用),我们在算法的理论分析时忽略了指纹函数的碰撞概率.在考虑碰撞概率的情况下研究参数 c,ε与δ的关系以及在考虑碰撞概率的情况下设计此问题的(ε,δ)-算法,这是我们深入研究的问题.

致谢 感谢麻省理工学院 Piotr Indyk 教授和香港城市大学 Chung Keung Poon 教授在作者从事数据流领域的研究工作中所给予的帮助.

References:

[1] Flajolet P, Martin GN. Probabilistic counting algorithms for data base applications. Journal of Computer and System Sciences, 1985,31(2):182−209. [doi: 10.1016/0022-0000(85)90041-8]

[2] Alon N, Matias Y, Szegedy M. The space complexity of approximating the frequency moments. Journal of Computer and System Sciences, 1999,58(1):137−147. [doi: 10.1006/jcss.1997.1545]

[3] Garofalakis M, Kumar A. XML stream processing using tree-edit distance embeddings. ACM Trans. on Database Systems, 2005, 30(1):279−332. [doi: 10.1145/1061318.1061326]

[4] Polyzotis N, Garofalakis M, Ioannidis Y. Approximate XML query answers. In: Weikum G, ed. Proc. of the 2004 ACM SIGMOD Int'l Conf. on Management of Data. New York: ACM Press, 2004. 263−274.

[5] Bar-Yossef Z, Jayram TS, Kumar R, Sivakumar D, Trevisan L. Counting distinct elements in a data stream. In: Rolim JDP, Vadhan S, eds. Proc. of the 6th Int'l Workshop on Randomization and Approximation Techniques in Computer Science. Berlin, Heidelberg: Springer-Verlag, 2002. 1−10.

[6] Gibbons PB, Tirthapura S. Estimating simple functions on the union of data streams. In: Rosenberg A, ed. Proc. of the Symp. on Parallel Algorithms and Architectures. New York: ACM Press, 2001. 281−291.

[7] Indyk P. Stable distributions, pseudorandom generators, embeddings and data stream computation. In: Proc. of the 41st Symp. on Foundations of Computer Science. Cambridge: IEEE Computer Society Press, 2000. 189−197. http://portal.acm.org/citation.cfm?id=795666.796606

[8] Cormode G, Datar M, Indyk P, Muthukrishnan S. Comparing data streams using Hamming norms (how to zero in). In: Ramakrishnan R, ed. Proc. of the 28th Int'l Conf. on Very Large Data Bases. New York: VLDB Press, 2002. 335−345.

[9] Cormode G, Muthukrishnan S. Estimating dominance norms of multiple data streams. In: Battista GD, Zwick U, eds. Proc. of the 11th European Symp. on Algorithms. Berlin, Heidelberg: Springer-Verlag, 2003. 148−160.

[10] Zolotarev V. One-Dimensional Stable Distributions. In: Vol.65 of Translations of Mathematical Monographs. American Mathematical Society Press, 1986.

[11] Blum M, Micali S. How to generate cryptographically strong sequences of pseudorandom bit. SIAM Journal on Computing, 1984, 13(4):850−864. [doi: 10.1137/0213053]

[12] Yao AC. Theory and applications of trapdoor functions. In: Fischer MJ, ed. Proc. of the 23rd IEEE Symp. on Foundations of Computer Science. Cambridge: IEEE Computer Society Press, 1982. 80−91.

[13] Nisan N. Pseudorandom generators for space-bounded computation. In: Ortiz H, ed. Proc. of the 22nd Symp. on Theory of Computation. ACM Press, 1990. 204−212.

孙贺(1984－),男,吉林长春人,博士,主要研究领域为算法设计与分析,计算复杂性,数据流理论.

朱洪(1939－),男,教授,博士生导师,主要研究领域为算法设计与分析,计算复杂性,密码学.