文章编号: 1007-791X(2015)01-0042-09

博客数据的属性拓扑分析

张 涛[1,*],李 慧[1],任宏雷[2]

(1. 燕山大学 信息科学与工程学院,河北 秦皇岛 066004; 2. 唐山启奥科技股份有限公司 系统测试部,河北 唐山 063000)

摘 要: 近些年博客逐渐呈现蓬勃发展趋势,但由于零门槛和缺少监管,博客信息斑驳繁杂,信息垃圾层出不穷。形式概念分析是数据分析与知识处理的有力工具,而属性拓扑作为形式背景的新型的表示方法,在背景表示可视化和概念计算可视化方面尤其有效。本文在子拓扑并行计算形式概念的理论基础上,加入一些条件约束,通过对博客数据的形式概念计算,对博主及其博客主题内容的相关信息进行合理的整合和深层次的挖掘,有利于摒弃无用信息,为博客使用者迅速发现对自己有利和感兴趣的博客内容以及了解博客作者的相关信息提供了理论依据。

关键词: 形式概念分析; 属性拓扑; 全路径搜索; 博客

中图分类号: TP182 文献标识码: A DOI: 10.3969/j.issn.1007-791X.2015.01.007

0 引言

博客是以自由、开放和共享为文化特征,通过图文音象等表现形式,围绕个人网络存在的五大功能,提供存取读写、组织沟通、评价交换等服务的一种社会化个人服务模式。它并不是纯粹的技术创新,而是一种逐渐演变的网络应用,一种形式的变化。博客的全民性,让它的传播方式成为所有人对所有人的传播。然而,博客并不如表面般繁荣,当博客毫不掩饰地在大众面前喧闹的时候,接踵而来的问题使得博客乱了方寸。博客参与者的盲目性导致了博客行为过程中的迷茫与厌倦;由于进入的零门槛和缺少监管,彻底颠覆互联网既有模式的博客,变成了新的信息垃圾场。博客正遭受低俗肤浅成风、网络侵权等因素的困扰。

作为数据分析与知识处理的有力工具,形式概念分析[1]以数学化的概念和概念层次为基础,已经应用在众多领域,如数据挖掘[2-3]、网络搜索[4-5]、软件工程[6-7]、本体分析[8-9]等,并仍然具有很大的潜在应用价值。

属性拓扑[10-12]作为一种新型的形式背景表示方法,可以有效地生成形式概念,本文在子拓扑计算形式概念[13]的基础上,加入一些条件约束,通过对博客数据的形式概念计算,对博客信息资源进行了科学的整合和发掘,对斑驳繁杂的博客信息进行了“过滤”,为博客使用者迅速发现对自己有利和感兴趣的博客内容以及了解博客作者的相关信息提供了理论依据,有利于摒弃无用信息,可以促进博客文化的科学管理和博客健康、有序的发展。

1 属性拓扑及形式概念计算

1.1 属性拓扑的表示

在形式背景 $K=(G,M,I)$ 中,定义 $T=(V,E)$ 为属性拓扑的邻接矩阵表示[10-12]。其中,$V=M$ 为拓扑的顶点集合,E 为拓扑中边的集合。则可以得到属性拓扑的表达式:

收稿日期: 2014-06-25 基金项目: 国家自然科学基金资助项目(61273019,61201111,81273740,81373767); 河北省自然科学基金资助项目(F2013203368,F2015203013)

作者简介: * 张涛(1979-),男,河北唐山人,博士,副教授,主要研究方向为信息融合、可视化模式识别、图像处理,Email: zhtao@ysu.edu.cn。

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

$$E(v_i,v_j)=\begin{cases} g(m_i)\cap g(m_j) & \text{其他} \\ \varnothing & g(m_i)-g(m_j)=\varnothing \text{ 或 } g(m_i)-g(m_j)=g(m_i)。\end{cases} \quad (1)$$

定义 $T=(V,E')$ 为属性拓扑的关联矩阵表示[1]。同样，$V=M$ 为拓扑中的顶点集合，E'为拓扑中边的集合。由此得到属性拓扑的另一个表达式为

$$E'(v_i,v_j)=\begin{cases} 1 & \text{其他} \\ 0 & g(m_i)-g(m_j)=g(m_i)。 \\ -1 & g(m_i)-g(m_j)=\varnothing \end{cases} \quad (2)$$

由形式背景所对应的邻接矩阵和关联矩阵，即可得到以所有属性为顶点的形式背景的属性拓扑[1]。

1.2　属性拓扑的分解与子拓扑的构造

依文献[2]可知，根据属性拓扑中各属性类别及其之间的关系情况，可判断拓扑是否可分解为若干子拓扑。将拓扑判定为可分解后，便可以进行下一步，即拓扑的分解与子拓扑的构造。

假设形式背景 $K=(G,M,I)$ 可分解，其属性拓扑中的顶层属性的个数为 n，则子拓扑数也为 n。拓扑的分解与子拓扑的构造流程图如图 1。

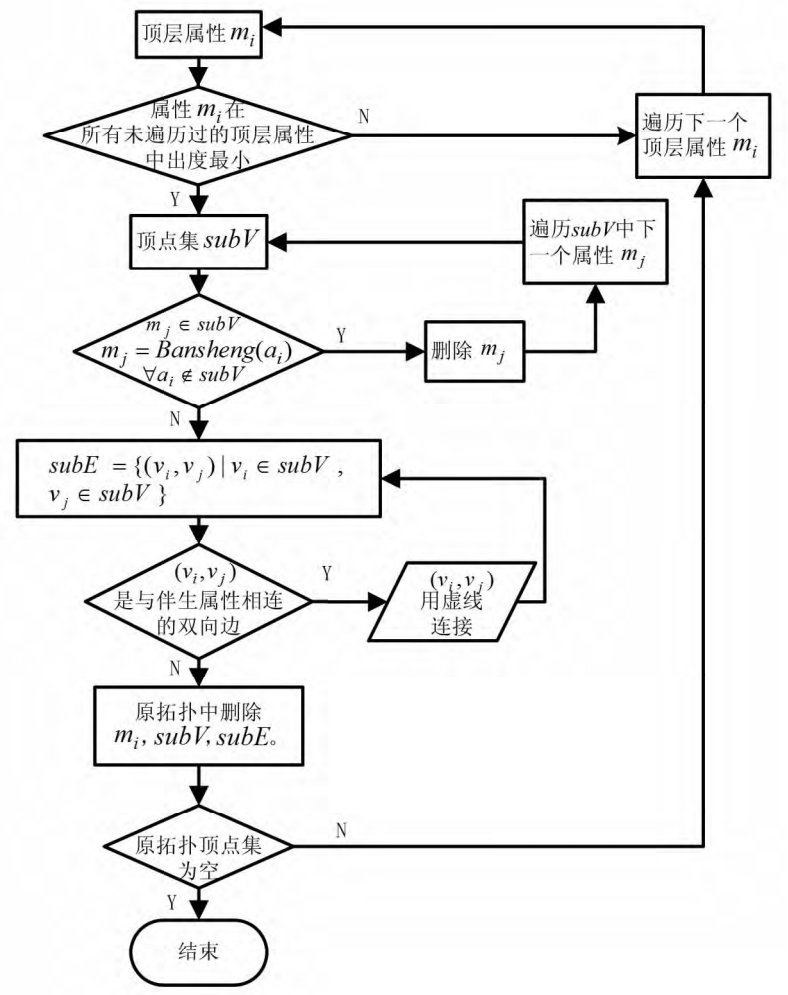

图 1　子拓扑的分解流程图

Fig. 1　Flow chart of decomposition of sub-topologies

1.3　基于子拓扑的形式概念计算算法

利用上述得到的子拓扑进行概念的计算，假设得到子拓扑分别为 $subT(V_1,E_1)$，$subT(V_2,E_2)$，…，$subT(V_n,E_n)$。

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

文献[2]通过对分解后的属性子拓扑分别进行全路径搜索实现概念的计算。为了将算法应用到实际数据的分析中，需要在文献[2]提出的算法实现的理论基础上加入一些约束条件，修改后的算法流程图如图2所示。

依照图2所示算法流程图，分别对子拓扑 $subT(V_1,E_1)$，$subT(V_2,E_2)$，…，$subT(V_n,E_n)$ 进行全路径搜索，对每个子图的所有路径进行遍历后，可以得到各子图中对应所有的形式概念集 $\beta(subT(V_1,E_1))$，$\beta(subT(V_2,E_2))$，…，$\beta(subT(V_n,E_n))$。

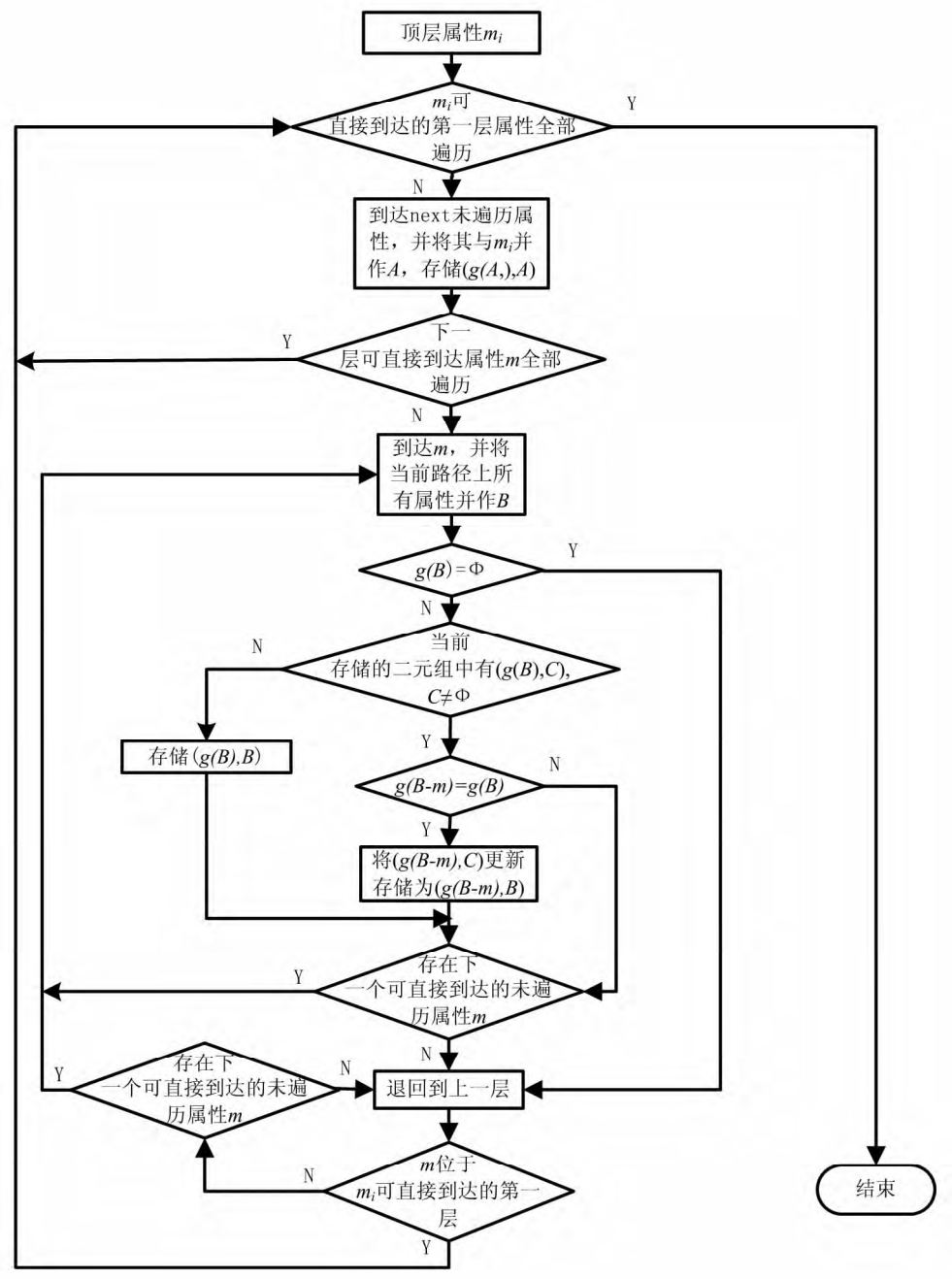

图2 子拓扑计算形式概念算法流程图

Fig. 2 Flow chart of calculation of formal concepts by sub-topology

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

2　基于属性拓扑的博客数据分析

2.1　拓扑的生成

经过二十多年的不断发展，目前的 UCI 数据库已经包含了 264 个不同种类的数据集。其中，在这些数据集中，大部分来自于各个领域专家真实的实验数据，涉及到生命科学、物理科学、工程学、社会科学等多个领域。为了测试和验证属性拓扑算法表示形式背景和计算形式概念的可行性，本文选取 UCI 机器学习数据库(UCI Machine Learning Repository) 中的最近更新的博客数据集进行测试，其包含了 6 个属性和 100 个对象，这些属性和对象的关系包括了形式背景中可能出现的所有关系，并且净化后的背景直观明确，适合本文中的实验。在实验中，本文研究的属性拓扑算法均在净化背景的基础上[10-12]，因此需要先对样本数据进行背景的净化和处理，将重复的对象合并整理，多值属性进行单值化，净化后的测试数据集包含 14 个属性和 41 个对象，如表 1 所示。

表 1 中的各字母代表的含义分别如下: a: 博主为高学历; b: 博主为中等学历; c: 博主学历较低; d: 政治立场为左派; e: 政治立场为中立; f: 政治立场为右派; g: 博客主题为感想; h: 博客主题为政治; i: 博客主题为旅游; j: 博客主题为新闻; k: 博客主题为科学; l: 博客被当地媒体转载; m: 地方，政治和社会空间; n: 该博主为临博主。

由第 2 节的邻接矩阵和关联矩阵计算公式，可以得到表 1 所示。形式背景的邻接矩阵和关联矩阵分别如下:

邻接矩阵:

表 1　净化后的测试数据集

Tab. 1　Test data set after purification

	a	b	c	d	e	f	g	h	i	j	k	l	m	n
1	×			×			×					×	×	×
2	×			×				×				×	×	×
3		×			×				×			×	×	×
4		×			×					×		×	×	×
5	×					×		×				×		×
6	×					×		×				×		
7		×				×			×			×		×
8	×					×			×			×	×	×
9		×		×						×		×		×
10	×			×				×			×	×		
11			×			×				×			×	
12		×		×			×					×	×	×
13		×		×				×				×	×	×
14	×					×		×				×	×	×
15	×					×			×			×	×	
16		×		×						×		×	×	×
17	×					×		×					×	
18			×	×					×			×		
19	×			×						×		×	×	×
20			×			×	×							×
21	×			×			×							×
22		×		×							×	×	×	
23			×	×							×	×	×	
24		×				×			×			×	×	
25			×			×		×				×	×	×
26	×			×			×					×		×
27		×		×					×			×		×
28		×			×						×	×		×
29		×			×		×						×	
30		×				×					×	×	×	
31		×			×	×							×	
32		×		×						×			×	×
33		×								×		×	×	
34			×		×		×					×		
35			×			×	×					×		
36		×				×			×			×	×	×
37	×			×			×					×	×	
38	×					×	×					×	×	
39		×				×		×					×	
40		×			×		×					×		×
41		×			×		×					×	×	×

{1, 2, 5, 6, 8, 10, 14, 15, 17, 19, 21, 26, 37, 38	Φ	Φ	{1, 2, 10, 19, 21, 26, 37}	Φ	{5, 6, 8, 14, 15, 17, 38}	{1, 21, 26, 37, 38}	{2, 5, 6, 10, 14, 17}	{8, 15}	{19}	Φ	{1, 2, 5, 6, 8, 10, 14, 15, 19, 26, 37, 38}	{1, 2, 8, 10, 14, 15, 17, 19, 37, 38}	{1, 2, 5, 8, 14, 19, 21, 26}
Φ	{3, 4, 7, 9, 12, 13, 16, 22, 24, 27-33, 36, 39, 40, 41}	Φ	{9, 12, 13, 16, 19, 22, 27, 31, 32}	{3, 4, 28, 29, 40, 41}	{7, 24, 30, 33, 36, 39}	{12, 29, 31, 40, 41}	{13, 39}	{3, 7, 24, 27, 36}	{4, 9, 16, 32, 33}	{22, 28, 30}	{3, 4, 7, 9, 12, 13, 16, 22, 24, 27, 28, 30, 33, 36, 40, 41}	{3, 4, 12, 13, 16, 22, 24, 29, 30, 32, 33, 36, 39, 41}	{3, 4, 7, 7, 12, 13, 16, 27, 28, 31, 32, 36, 40, 41}
Φ	Φ	{11, 18, 20, 23, 25, 34, 35}	{18, 23}	{34}	{11, 20, 25, 35}	{20, 34, 35}	{25}	{18}	{11}	{23}	{18, 23, 25, 34, 35}	{11, 23, 25}	{20, 25}
{1, 2, 10, 19, 21, 26, 37}	{9, 12, 13, 16, 19, 22, 27, 31, 32}	{18, 23}	{1, 2, 9, 10, 12, 13, 16, 18, 19, 21, 22, 23, 26, 27, 31, 32, 37}	Φ	Φ	{1, 12, 21, 26, 31, 37}	{2, 10, 13}	{18}	{9, 16, 19, 32}	{22, 23}	{1, 2, 9, 12, 13, 16, 18, 19, 22, 23, 26, 27, 37}	{1, 2, 10, 12, 13, 16, 19, 22, 23, 32, 37}	{1, 2, 9, 12, 13, 16, 19, 21, 26, 27, 31, 32}
Φ	{3, 4, 28, 29, 40, 41}	{34}	Φ	{3, 4, 28, 29, 34, 40, 41}	Φ	{29, 34, 40, 41}	Φ	{3}	{4}	{28}	{3, 4, 28, 34, 40, 41}	{3, 4, 29, 41}	{3, 4, 28, 40, 41}
{5, 6, 8, 14, 15, 17, 38}	{7, 24, 30, 33, 36, 39}	{11, 20, 25, 35}	Φ	Φ	{5-8, 11, 14, 15, 17, 20, 24, 25, 30, 33, 35, 36, 38, 39}	{20, 35, 38}	{5, 6, 14, 17, 25, 39}	{7, 8, 15, 24, 36}	{11, 33}	{30}	{5-8, 14, 15, 24, 25, 30, 33, 35, 36, 38}	{8, 11, 14, 15, 17, 24, 25, 30, 33, 36, 38, 39}	{5, 7, 8, 14, 20, 25, 36}
{1, 21, 26, 37, 38}	{12, 29, 31, 40, 41}	{20, 34, 35}	{1, 12, 21, 26, 31, 37}	{29, 34, 40, 41}	{20, 35, 38}	{1, 12, 20, 21, 26, 29, 31, 34, 35, 37, 38, 40, 41}	Φ	Φ	Φ	Φ	{1, 12, 26, 34, 35, 37, 38, 40, 41}	{1, 12, 29, 37, 38, 41}	{1, 12, 20, 21, 26, 31, 40, 41}
{2, 5, 6, 10, 14, 17}	{13, 39}	{25}	{2, 10, 13}	Φ	{5, 6, 14, 17, 25, 39}	Φ	{2, 5, 6, 10, 13, 14, 17, 25, 39}	Φ	Φ	Φ	{2, 5, 6, 10, 13, 14, 25}	{2, 10, 13, 14, 17, 25, 39}	{2, 5, 13, 14, 25}
{8, 15}	{3, 7, 24, 27, 36}	{18}	{18}	{3}	{7, 8, 15, 24, 36}	Φ	Φ	{3, 7, 8, 15, 18, 24, 27, 36}	Φ	Φ	Φ	{3, 8, 15, 24, 36}	{3, 7, 8, 27, 36}
{19}	{4, 9, 16, 32, 33}	{11}	{9, 16, 19, 32}	{4}	{11, 33}	Φ	Φ	Φ	{4, 9, 11, 16, 19, 32, 33}	Φ	{4, 9, 16, 19, 33}	{4, 11, 16, 19, 32, 33}	{4, 9, 16, 19, 32}
Φ	{22, 28, 30}	{23}	{22, 23}	{28}	{30}	Φ	Φ	Φ	Φ	{22, 23, 28, 30}	Φ	{22, 23, 30}	{28}
{1, 2, 5, 6, 8, 10, 14, 15, 19, 26, 37, 38}	{3, 4, 7, 9, 12, 13, 16, 22, 24, 27, 28, 30, 33, 36, 40, 41}	{18, 23, 25, 34, 35}	{1, 2, 9, 12, 13, 16, 18, 19, 22, 23, 26, 27, 37}	{3, 4, 28, 34, 40, 41}	{5-8, 14, 15, 24, 25, 30, 33, 35, 36, 38}	{1, 12, 26, 34, 35, 37, 38, 40, 41}	{2, 5, 6, 10, 13, 14, 25}	{3, 7, 8, 15, 18, 24, 27, 36}	{4, 9, 16, 19, 33}	{22, 23, 28, 30}	{1-10, 12-16, 18, 19, 22-28, 30, 33-38, 40, 41}	{1-4, 8, 10, 12-16, 19, 22-25, 30, 33, 36-38, 41}	{1-5, 7-9, 12-14, 16, 19, 25-28, 36, 40, 41}
{1, 2, 8, 10, 14, 15, 17, 19, 37, 38}	{3, 4, 12, 13, 16, 22, 24, 29, 30, 32, 33, 36, 39, 41}	{11, 23, 25}	{1, 2, 10, 12, 13, 16, 19, 22, 23, 32, 37}	{3, 4, 29, 41}	{8, 11, 14, 15, 17, 24, 25, 30, 33, 36, 38, 39}	{1, 12, 29, 37, 38, 41}	{2, 10, 13, 14, 17, 25, 39}	{3, 8, 15, 24, 36}	{4, 11, 16, 19, 32, 33}	{22, 23, 30}	{1-4, 8, 10, 12-16, 19, 22-25, 30, 33, 36-38, 41}	{1-4, 8, 10-17, 19, 22-25, 29, 30, 32, 33, 36-39, 41}	{1-4, 8, 10, 12-16, 19, 22-25, 30, 33, 36-38, 41}
{1, 2, 5, 8, 14, 19, 21, 26}	{3, 4, 7, 7, 12, 13, 16, 27, 28, 31, 32, 36, 40, 41}	{20, 25}	{1, 2, 9, 12, 13, 16, 19, 21, 26, 27, 31, 32}	{3, 4, 28, 40, 41}	{5, 7, 8, 14, 20, 25, 36}	{1, 12, 20, 21, 26, 31, 40, 41}	{2, 5, 13, 14, 25}	{3, 7, 8, 27, 36}	{4, 9, 16, 19, 32}	{28}	{1-5, 7-9, 12-14, 16, 19, 25-28, 36, 40, 41}	{1-4, 8, 10, 12-16, 19, 22-25, 30, 33, 36-38, 41}	{1-5, 7-9, 12-14, 16, 19-21, 25-28, 31, 32, 36, 40, 41}

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

关联矩阵:

$$\begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & -1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{bmatrix}$$

由关联矩阵确定图中各边的走向,即关联矩阵中若两个属性所在位置交叉处对应的两个值均为0,则这两个属性无关联,若值为1和-1,则这两个属性为包含关系[10],用单向边连接,若两个值均为1,则这两个属性为相容关系[10],拓扑中用双向边连接;邻接矩阵确定图中边的权值,可以得到表1的属性拓扑如图3所示,图中包含了形式背景中的所有属性以及属性间的关联方式和关联强度,其为带权值的有向图。为表示方便,图中将各边的权值省略。

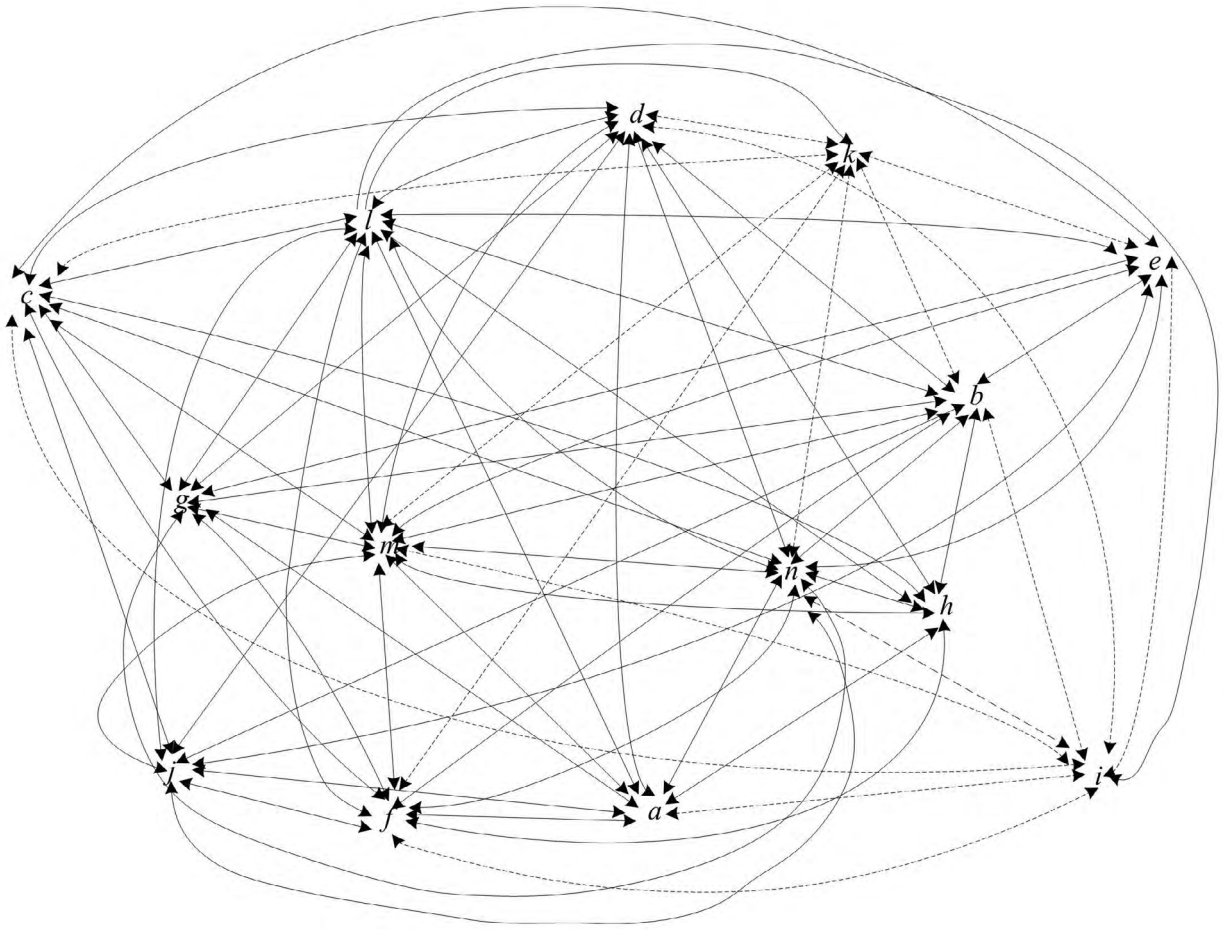

图3 表1的属性拓扑图

Fig. 3 The attribute topology for Tab. 1

依文献[10]中顶层属性和文献[12]中伴生属性的定义可知,图3中,顶层属性为:a,b,c,d,e,f,j,h,j,l,m,n,伴生属性为:j,k,且属性i,k均为属性l的伴生属性。与伴生属性i,k相连的双向边用虚线连接,表示该双向边连接的两个属性在概念计算时不能直接关联,需要通过伴生属性的父属性产生关联[13]。

图4为表1所示背景的其中一种概念格。显

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

然,相对于概念格,从属性拓扑图中可以更明显的看出各个属性间是否有关联,及其产生关联的对象集合,具有更强的可视性,使得属性间的关系一目了然。

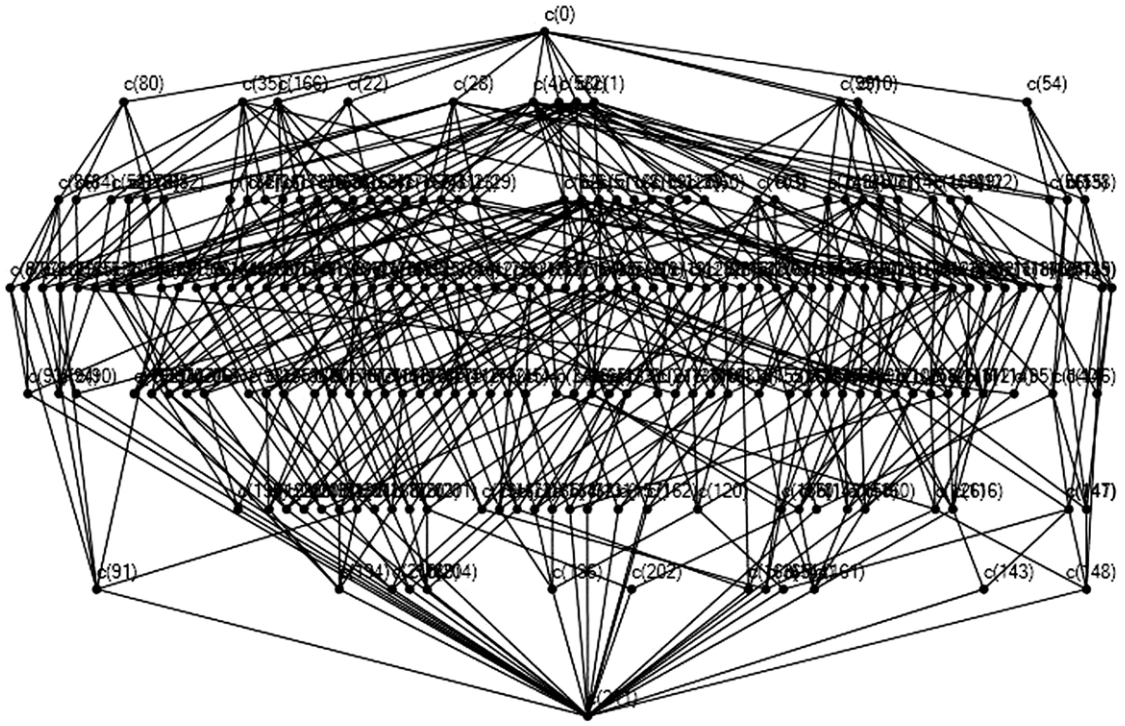

图 4　表 1 的概念格

Fig. 4　The concept lattice of Tab. 1

2.2　形式概念计算

属性拓扑是形式概念分析理论的一种突破,因为该方法既做到了形式背景的表示,同时还可用于计算,其优势就在于背景表示的可视性和概念计算的可视性。

下面将以分析实验的方式分析基于属性拓扑的概念计算方法,并结合实际情况对本组数据进行分析。

分析图 3 可知,该属性拓扑显然是可分解的。采用文献 [2] 中给出的子拓扑构造算法,将图 3 所示拓扑中的 12 个顶层属性进行出度大小排序并以这些顶层属性为中心构造子拓扑。其中这 12 个顶层属性的顺序为: h,a,g,e,j,b,d,c,f,l,m,n,对应做出各顶层属性的子拓扑如图 5 ~ 15 所示,n 的子拓扑只包含 n 一个属性,在此不做画图表示,并依据这些子拓扑进行概念的计算。为表示方便,子图中将各边的权值省略,图中的虚线表示与伴生属性相连的双向边。显然,子图的结构要更简化,更便于属性关系的观察和概念的计算。

下面以属性 h 的子拓扑图 5 为例进行计算和分析,分析子拓扑的概念计算算法。该算法利用了图的全路径搜索原理,首先将子拓扑中中心属性以外的所有属性进行随机排序(本文中采用了字母表顺序) ,再按顺序进行遍历,可实现在搜索过程中省略掉伪概念,只保留概念,因此减小了计算步骤。下图 16 是子图 h 的全路径搜索示意图。

图 16 中的虚线表示遍历过程中产生的伪概念直接忽略掉不做存储,由此遍历过程即可直接得到全部的概念并且避免了伪概念的产生,提高了计算的效率,并且遍历过程实现了可视化,便于观察和分析。

图 6 ~ 15 子拓扑的概念计算过程与属性 h 的子拓扑计算过程类似,同样是将与其直接相关联的属性进行排序,再按顺序进行全路径搜索,在伪概念产生条件下,遍历路径用虚线表示,并不做存储,直至路径遍历完全。由于过程类似,文中将不再做详细计算分析。

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

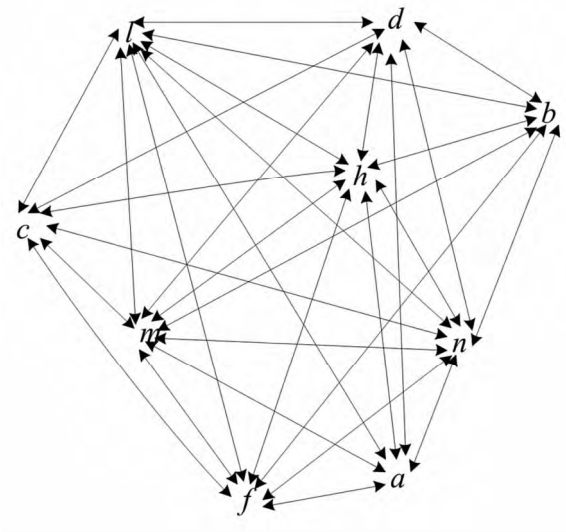

图 5 顶层属性 h 的子拓扑

Fig. 5 The sub-topology of top-attribute h

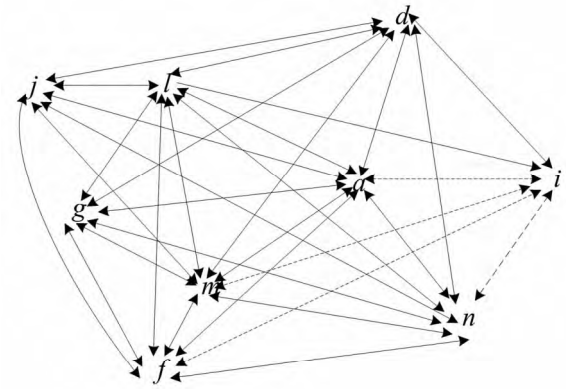

图 6 顶层属性 a 的子拓扑

Fig. 6 The sub-topology of top-attribute a

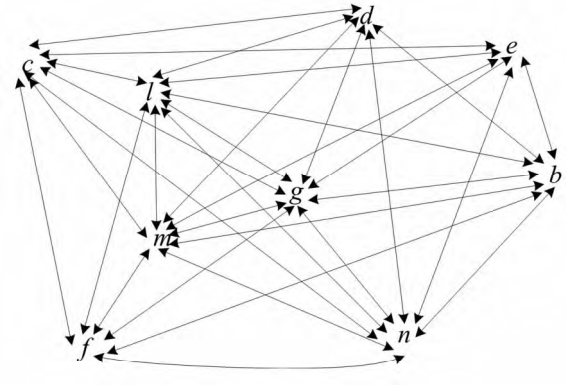

图 7 顶层属性 g 的子拓扑

Fig. 7 The sub-topology of top-attribute g

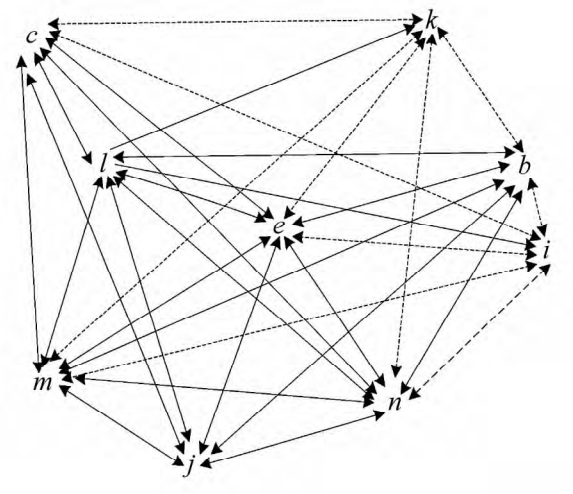

图 8 顶层属性 e 的子拓扑

Fig. 8 The sub-topology of top-attribute e

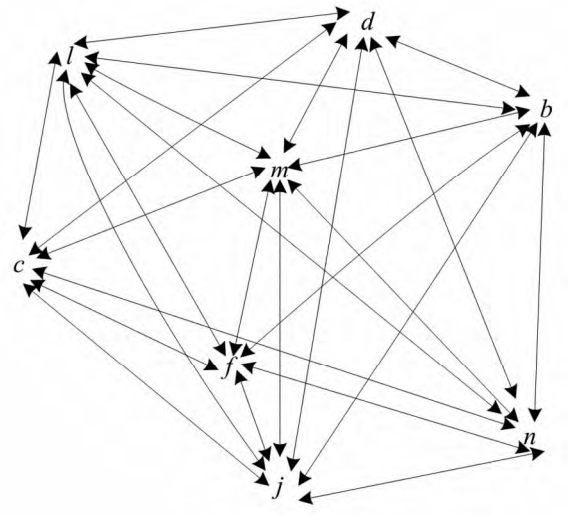

图 9 顶层属性 j 的子拓扑

Fig. 9 The sub-topology of top-attribute j

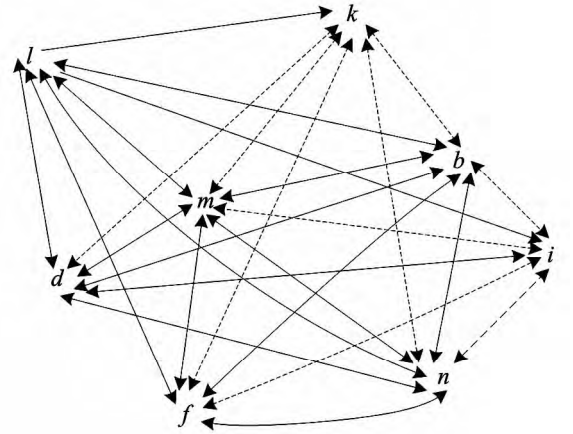

图 10 顶层属性 b 的子拓扑

Fig. 10 The sub-topology of top-attribute b

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

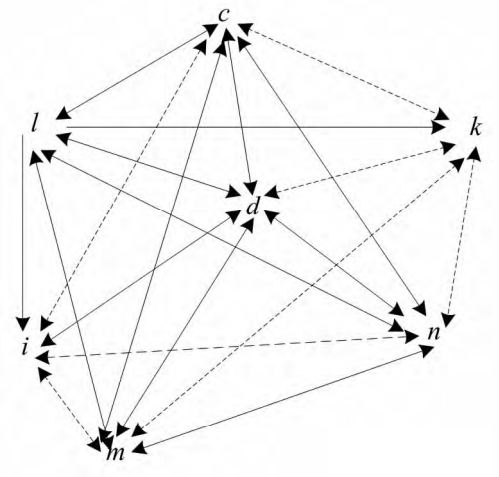

图 11　顶层属性 d 的子拓扑

Fig. 11　The sub-topology of top-attribute d

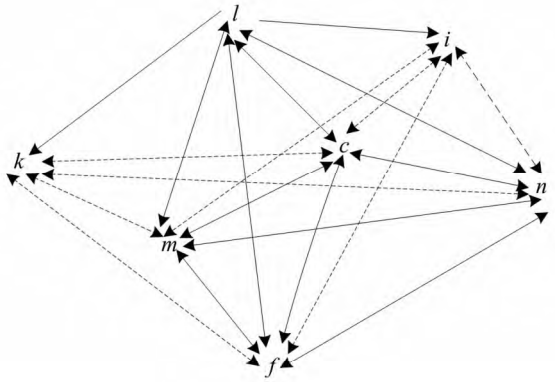

图 12　顶层属性 c 的子拓扑

Fig. 12　The sub-topology of top-attribute c

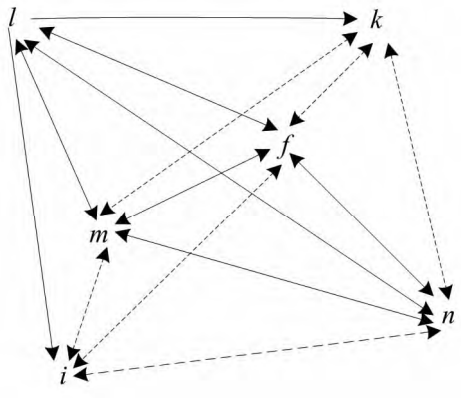

图 13　顶层属性 f 的子拓扑

Fig. 13　The sub-topology of top-attribute f

2.3　数据分析

前面分析了属性拓扑关于概念计算的算法，而概念的计算目的则是为了分析数据中属性之间实际存在的联系。在属性拓扑计算形式概念的同时，可直观的看到每个顶层属性与其他所有属性的关联与关联强度，而根据这个关联强度的大小就可以提取其对应的类别，从而达到属性规整的效果。

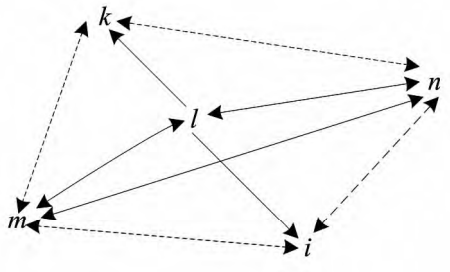

图 14　顶层属性 l 的子拓扑

Fig. 14　The sub-topology of top-attribute l

图 15　顶层属性 m 的子拓扑

Fig. 15　The sub-topology of top-attribute m

依旧以图 16 为例进行说明。图 16 是以属性 h 为中心的子拓扑的遍历过程，其中虚线代表过程中产生了伪概念，计算时该二元组不做存储，例如二元组$(\{2,10\},\{h,a,d,l\})$与二元组$(\{2,10\},\{h,a,d,l,m\})$，而存储直线后的二元组$(\{2,10\},\{h,a,d,l,m\})$，这是因为相同的对象集$\{2,10\}$下，属性集达到最大的$\{h,a,d,l,m\}$时，该二元组才可称为概念，结合图 16 的遍历过程反映到数据中即为，在同时满足了博客的主题为政治和博主为高学历前提下的 2 和 10 这两篇博客中，该博客政治立场为左派以及博客被当地媒体转载和该博客反映了地方，政治和社会空间这 3 条属性将并列同时出现，可以称这 3 个属性为属性集$\{h,a\}$下的绑定属性，即可以认为当一篇博客满足博客政治立场为左派以及博客被当地媒体转载两个条件时，这篇博客很有可能也反映了地方，政治和社会空间。而对于连接实线的属性 h 和属性 a 而言，表示二元组$(\{2,5,6,10,14,17\},\{h,a\})$即为一个概念，运算过程中可直接存储，结合属性代表的含义分析可知，2，5，6，10，14 和 17 这几篇博客同时满足了博客的主题为政治和博主为高学历两个条件。特别的，如图 16 所示，当前路径中若存在伴生属性，则必然存在其父属性，在实验中表现为，当博客主题为旅游或者科学时，则该博客均会被当地媒体转载。

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

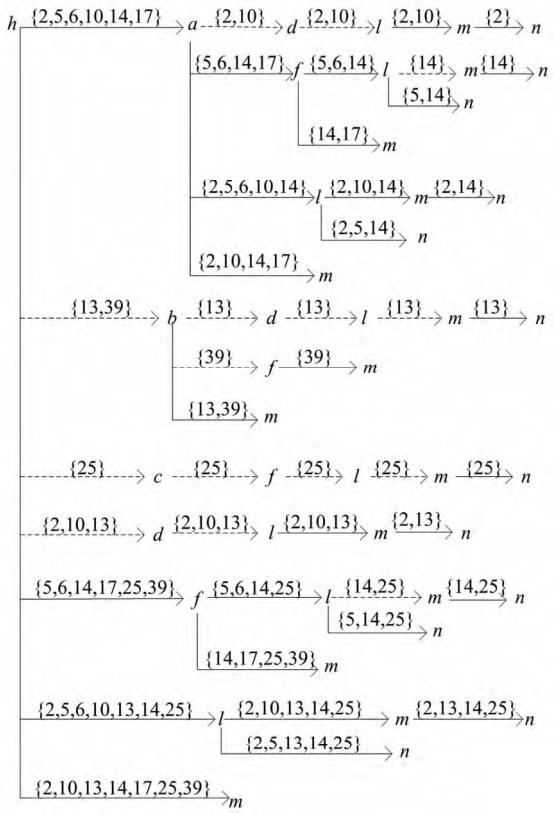

图 16 图 5 的全路径搜索示意图

Fig. 16 The full path searching of Fig. 5

3 结束语

本文在子拓扑计算形式概念的基础上，加入一些约束条件，将博客数据背景下的属性拓扑分解为子拓扑，利用全路径搜索计算博客数据的形式概念。通过博客数据形式概念的计算，深入分析了各个博主相关信息及其博客主题相关信息之间实际存在的联系，有利于博客使用者快速了解博主及其博客的相关主题，摒弃无用信息。

参考文献

[1] Ganter B, Wille R. Formal concept analysis: mathematical foundations [M]. New York: Springer-Verlag, 1999.

[2] Mehdi Kaytoue, Sergei O Kuznetsov, Amedeo Napoli, et al. Mining gene expression data with pattern structures in formal concept analysis [J]. Information Sciences, 2011, 181(10): 1989-2001.

[3] Tarek Abudawood. Improving Predictions of Multiple Binary Models in ILP [J]. The Scientific World Journal, 2014, 739062.

[4] Du Ya-jun, Hai Yu-feng. Semantic ranking of web pages based on formal concept analysis [J]. The Journal of Systems and Software, 2013, 86(1): 187-197.

[5] Anna Formica. Semantic Web search based on rough sets and Fuzzy Formal Concept Analysis [J]. Knowledge-Based Systems, 2012, 26: 40-47.

[6] Li Bi-xin, Sun Xiao-bing, Leung Hareton. Combining concept lattice with call graph for impact analysis [J]. Advances in Engineering Software, 2012, 53: 1-13.

[7] LI Bi-xin, SUN Xiao-bing, Jacky Keung. FCA-CIA: An approach of using FCA to support cross-level change impact analysis for object oriented Java programs [J]. Information and Software Technology, 2013, 55(8): 1437-1449.

[8] Lambrini Seremeti, Ioannis Kougias. Yoneda Philosophy in Engineering [J]. International Journal of Engineering Mathematics, 2013, 758729.

[9] Kang Xiangping, Li Deyu, Wang Suge. Research on domain ontology in different granulations based on concept lattice [J]. Knowledge-Based Systems, 2012, 27: 152-161.

[10] 张涛，任宏雷. 形式背景的属性拓扑表示[J]. 小型微型计算机系统，2014，35(3)：590-593.

[11] Zhang Tao, Ren Hong-lei, Wang Xiao-min. A calculation of formal concept by attribute topology [J]. ICIC Express Letters, Part B: Applications, 2013, 4(3): 793-800.

[12] 张涛，任宏雷，洪文学，等. 基于属性拓扑的可视化形式概念计算[J]. 电子学报，2014，42(5)：925-932.

Attribute topology analysis of blogger data

ZHANG Tao[1], LI Hui[1], REN Hong-lei[2]

(1. College of Information Science and Engineering, Yanshan University, Qinhuangdao, Hebei 066004, China;
2. System Testing Department, Tangshan Qiao Technology Company Limited, Tangshan, Hebei 063000, China)

Abstract: Blog has gradually showed vigorous development trend in recent years, but blog information presents mottled and multifarious, and information garbage emerge in endlessly due to the zero threshold and lack of regulation. Formal concept analysis is a powerful tool for data analysis and knowledge processing, and attribute topology, a new representation method of formal context, is especially effective in the visual representation of context and computational visualization. Based on the theory of computing formal concepts in parallel, this paper presents the reasonable integration and deep-seated mining of the related information of the bloggers and the themes of blog by adding some constrains and computing the formal concepts of blog data. It is beneficial to abandon the useless information and it provides a theoretical basis for the blog users to find useful and interesting bolg content for himself and the related information of bloggers quickly.

Key words: formal concept analysis; attribute topology; full path searching; blog

?1994-2015 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net