挖掘IETF专家社区的原型系统

叶小榕[1],邵晴[2]

1. 中国科学技术信息研究所,北京 100038
2. 北龙中网(北京)科技有限责任公司,北京 100190

摘要 IETF是全球互联网较权威的技术标准化组织。为分析IETF各个专业领域内的标准化专家,更好地掌握当前互联网标准的研究热点,设计了挖掘IETF专家社区的原型系统。该系统可对IETF的多种数据来源进行针对性的解析和存储,对数据进行清洗和合并,利用优化和改进后的Louvain算法进行社区发现,从而将专家划分到多个不同的互联网专业领域方向,随后挖掘出各个专业领域内的活跃专家,并采用TF-IDF算法挖掘出研究热点,能为制定互联网国际标准提供准确的动态信息。

关键词 IETF;社区发现;Louvain算法

经济全球化的今天,标准化是国家治理和国家间开展经济、贸易、文化与技术等交流的重要规则和手段。在互联网领域,近年来全球化竞争越发激烈。标准的制定,离不开相应领域的专家。通过对制定标准的专家进行分析和挖掘,可以深入掌握当前互联网标准的制定情况和最新动态,进而促进培养中国互联网领域的标准化专家,增强中国在国际标准立项和制定方面的主导权。

目前,对专家进行分析时主要是通过对其发表的论文、论文所属的学术期刊以及论文和期刊彼此的关系进行各种算法分析,识别出不同的专业领域方向,从而挖掘出各个专业领域内的活跃专家。比如刘东信等[1]提出利用特征因子分值和论文影响分值筛选SCI中排名高的期刊,进而根据在这些期刊上发表论文的频次得到专家排名;蒲姗姗[2]通过分析专家发表的论文、知识结构和影响力,提出一种专家推荐模型;李江等[3]提出了利用专家的专长吻合度、学术影响力与社会关联值构建专家遴选、回避与推荐模型。然而以上方法不太适用于分析和挖掘互联网领域的标准化专家。以制订了当前绝大多数国际互联网技术标准的国际互联网工程任务组(The Internet Engineering Task Force,简称IETF[4])为例,标准化专家通过提交RFC(request for comments)、发送电子邮件与参加IETF会议等方式进行讨论交流,来推动标准的制订[5],一方面没有涉及期刊,另一方面电子邮件、会议纪要等内容简短零散,难以进行内容分析并设定影响力,并且标准化专家之间没有回避机制,所以围绕期刊、论文等的各种分析和挖掘方法来进行专家推荐并不适用,需要根据IETF的自身特点,应用新的分析和

收稿日期:2019-10-30;修回日期:2019-11-26

作者简介:叶小榕,高级工程师,研究方向为计算机软件、数字图书馆,电子信箱:yeelfine@sina.com

引用格式:叶小榕, 邵晴. 挖掘IETF专家社区的原型系统[J]. 科技导报, 2019, 37(24): 100-110; doi: 10.3981/j.issn.1000-7857.2019.24.012

挖掘方法。

经过30多年的发展,IETF标准化专家人数迅速增加,专家间的联系愈加频繁,形成了复杂的大型网络结构体。专家研究范围相对集中,因此比较适宜采用社区发现算法,即将IETF复杂的社区结构映射为不同的互联网技术领域方向,然后获取每个领域内的活跃专家及其研究热点。

社区发现算法的研究从21世纪初开始。2002年,Girvan和Newman[6]开创性地提出了经典社区发现算法——GN算法。在此基础上,Vincent、Jean-Loup和Renaud等[7]提出了Louvain算法,为围绕Louvain算法的研究打下了基础。近年来Louvain算法的优化研究取得了很大的进展,例如吴祖峰等[8]提出了叶子社区剪枝策略,提高了Louvain算法的运行效率;Traag[9]提出了优化模块度增益的随机邻居Louvain算法,但也使结果的稳定性有所降低;吴卫江等[10]提出了加快计算速度的Louvain并行社区划分算法,但此算法会受到计算中延迟的影响;李贤和许大卫[11]提出了基于网络数据中心度的优化算法,促使小社区的合并且抑制大社区间的过度合并;陈启伟[12]针对IETF专家的社交数据提出了多种挖掘和分析方法,并对多种社区发现算法进行了分析。

在上述研究的基础上,为了更好地分析挖掘IETF中的互联网领域的标准化专家,本研究组设计开发了挖掘IETF专家社区的原型系统,通过IETF公开的RFC标准文件、电子邮件和会议纪要,分析专家信息、技术内容、讨论时间地点等数据,采用社区发现算法将专家划分到多个不同互联网专业领域方向的社区,每个技术社区代表一个专业领域方向,同时挖掘出每个技术社区内的活跃专家。通过此系统可以深入了解各个专业领域的活跃专家和其研究热点,掌握当前互联网标准的最新成果。

1　系统架构

本系统为了挖掘出IETF专业领域的活跃专家和其研究热点,首先需要对IETF的3类数据进行解析、提取和存储,并进行适当的清洗、修复、查重与合并处理,通过社区发现算法和TF-IDF算法,获取IETF中各专业领域的活跃专家及其研究热点。根据以上所需功能,本系统包括了如下4个模块。(1)数据解析和存储模块:针对IETF的3种数据来源,包括RFC标准文件、电子邮件和会议纪要,分别进行解析,提取专家信息,并建立起专家之间的联系,同时存储在系统的数据库中;(2)数据预处理模块:对数据进行清洗、规范和修复等处理,对于同名专家的数据进行查重和合并,并合并专家之间的关联关系;(3)社区发现模块:结合IETF的实际情况,优化和改进社区发现算法中的Louvain算法[13],计算出IETF当前的专业领域技术社区和该专业领域中活跃的专家;(4)热点发现模块:发现各个专业领域方向的活跃专家,并挖掘出这些专家的研究热点。系统整体架构如图1所示。

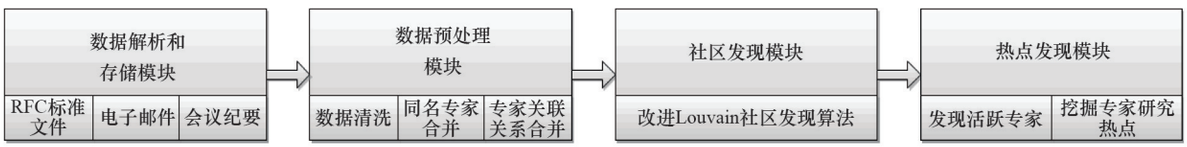

图1　系统整体架构

Fig. 1　Overall system architecture

2　数据解析和存储模块

本模块根据不同的IETF数据来源,分别进行针对性的解析,来提取专家信息,并建立起专家之间的联系,同时存储在专家库中,为后续模块提供数据源。

IETF的标准化专家是通过制定RFC标准、电子邮件和会议讨论等3种方式,在专家之间建立关

联关系。(1) RFC标准文件。RFC标准的发布要求十分严格,一般是由多个专家之间经过反复讨论和校对后共同提交,提交专家之间建立的联系最为紧密,本系统设定其权重最高。RFC标准文件包括提交标准专家的姓名、单位、地址、邮箱和提交时间等。(2) 电子邮件。作为IETF专家之间的交流手段,电子邮件使用十分频繁,对同一个技术内容感兴趣的专家会组成工作组并建立公共的邮件列表,专家通过公共邮件列表对技术内容进行讨论,这样专家之间就建立起了联系,本系统设定其权重低于RFC标准文件。邮件内容包括发件专家的姓名和邮箱、收件专家的姓名和邮箱、邮件列表的公共邮箱以及邮件发送时间等。(3) 会议纪要。IETF大会每年举行3次,各领域的专家根据兴趣参加其中的专题研讨会。会议文件包括两类,一类是总的参会专家名单,包括专家姓名、单位、国家或地区与开会时间等,另一类是各个专题研讨会的会议纪要,包括发言的专家姓名、发言记录等。在专题研讨会上发言的专家之间建立联系的紧密程度最低,本系统设定其权重也最低。但是专题研讨会的会议纪要有时记录不准,需要使用总的参会专家名单来进行修正。

IETF提供了所有RFC标准文件和电子邮件的批量下载,并且每次IETF会议都有专门的网址及公开详细的会议文件。本系统通过网页解析工具Jsoup,对RFC标准文件、电子邮件和会议文件进行解析,分别提取所需的数据,存储在专家库中。本系统采用MySQL数据库作为专家库。

2.1 RFC标准文件解析和存储

当前RFC标准文件有近9000个,涉及的专家有6000多人。下载的RFC标准文件的网页格式是固定的,以RFC7540(Hypertext Transfer Protocol Version 2)为例,网页正文显示了RFC标准的主体内容,结尾显示了提交专家的姓名、单位、地址、邮箱与提交时间等信息。网页源代码如图2所示。

Jsoup根据网页源代码中的关键字“Authors´ Addresses”进行解析,提取出专家信息,伪代码为:

```
Document doc = Jsoup.parse(htmlBody);
Element element = doc.select(".newpage").last();
List experts= GetAllExperts(string.indexOf(element.text(),"Authors´ Addresses") ); //得到所有专家的数据集合
for ( oneExpert: experts) {
    String name = oneExpert.GetName(); //提取专家姓名
    String company = oneExpert.GetCompay(); //提取单位名
    String email = oneExpert.GetEmail(); //提取邮箱
    String address = oneExpert.GetAddress(); //提取地址
}
```

```
<pre class="newpage"><span id="page-96"></span>
<span class="grey"><a href="./rfc7540">RFC 7540</a>HTTP/2  May 2015</span>
Authors' Addresses
  Mike Belshe
  BitGo
  EMail: mike@belshe.com

  Roberto Peon
  Google, Inc
  EMail: fenix@google.com

  Martin Thomson (editor)
  Mozilla
  331 E Evelyn Street
  Mountain View, CA  94041
  United States
  EMail: martin.thomson@gmail.com

Belshe, et al.          Standards Track          [Page 96]
</pre>
```

图2 RFC标准文件的网页源代码

Fig. 2 Web source code of RFC standard

Jsoup采用类似的方法可以提取出RFC标准文件的正文数据。RFC标准文件的专家信息、专家之间的关联关系和RFC标准文件的正文数据均保存在MySQL数据库中。RFC标准文件的专家信息表见表1。专家之间的关联关系数据和RFC标准文件的正文数据见表2。

表1 RFC标准文件的专家信息

Table 1 Expert information for RFC standard documents

字段名称	字段类型	字段说明
EXPERT_ID	int	专家的id
NAME	varchar(32)	专家姓名
COMPANY	varchar(64)	单位
EMAIL	varchar(32)	电子邮箱
ADDRESS	varchar(128)	地址

表2　RFC标准文件的专家关联数据和RFC文件的正文数据

Table 2　Expert association data for RFC standard documents

字段名称	字段类型	字段说明
RFC_ID	int	RFC的id
EXPERT_IDS	json	以json形式存储此RFC中所有的专家id
RFC_TIME	datetime	RFC标准文件的提交时间
RFC_CONTENT	text	RFC标准文件的全文内容

2.2　电子邮件解析和存储

IETF专家通过公共邮件列表讨论感兴趣的技术内容,从而形成一组邮件线索(称为email thread)。通过分析从IETF下载的邮件的网页源代码,将相同邮件线索的邮件合并到一起,使邮件线索中所有专家建立起关联关系。以某封邮件为例,其源代码如图3所示。

根据IETF中的RFC822中关于电子邮件标准格式的规定,From表示发件专家的姓名和邮箱地址,To表示收件专家的姓名和电子邮箱地址,Cc表示抄送专家的姓名和电子邮箱地址,Date表示邮件发送时间,Subject表示邮件主题,Message-ID是标识本封邮件的ID,References是被当前邮件回复过的其他所有邮件的Message-ID。通过Message-ID和References可以确定同一组邮件线索,使发送邮件进行讨论的专家之间建立起关联关系。与前文类似,电子邮件也通过Jsoup实现解析,提取出电子邮件的专家信息、专家之间的关联关系和电子邮件的正文数据,保存在MySQL数据库中。电子邮件的专家信息表见表3,电子邮件的专家关联数据和电子邮件的正文数据见表4。

```
<div id="msg-header" class="msg-header">
 <p>
   Return-Path: &lt;wjhns1@hardakers.net&gt;<br/>
   X-Original-To: dnsop@ietfa.amsl.com<br/>
   Delivered-To: dnsop@ietfa.amsl.com<br/>
   From: Wes Hardaker <wjhns1@hardakers.net><br/>
   To: Eric Orth <ericorth=40google.com@dmarc.ietf.org><br/>
   Cc: dnsop@ietf.org<br/>
   References: <CAMOjQcEtDBR29yKmOTvnx-7B7SmC9pox_kzOCKs4jBMQr1VSTA@mail.gmail.com><br/>
   Date: Mon, 30 Sep 2019 13:53:55 -0700<br/>
   In-Reply-To: <CAMOjQcEtDBR29yKmOTvnx-7B7SmC9pox_kzOCKs4jBMQr1VSTA@mail.gmail.com><br/>
   Message-ID: <yblblv15wv0.fsf@w7.hardakers.net><br/>
   Subject: Re: [DNSOP] Processing error codes in draft-ietf-dnsop-extended-error-10<br/>
 </p>
```

图3　电子邮件的网页源代码

Fig. 3　Web source code of email

表3　电子邮件的专家信息

Table 3　Expert information for email

字段名称	字段类型	字段说明
EXPERT_ID	int	专家的id
NAME	varchar(32)	专家姓名
EMAIL	varchar(32)	电子邮箱

2.3　会议纪要解析和存储

IETF大会每年举行3次。从2006年开始,每次大会总的参会专家名单和大会中的各个专题研讨会的会议纪要均在网上公开发布。参会专家名单包括专家姓名、单位、国家或地区等。专题研讨会的会议纪要包括了发言的专家姓名、发言记录与会议时间等。专题研讨会的会议纪要有时不是特别准确,可能会出现专家姓名拼写错误等问题,因此使用参会专家名单作为修正参考。修正过的会议纪要,将发言的专家彼此建立起关联关系。会议纪要和参会专家名单的网页源代码见图4,左侧为会议纪要,右侧为参会专家名单。

会议纪要也通过Jsoup实现解析,提取出会议纪要的专家信息、专家之间的关联关系和会议纪要的正文数据,保存在MySQL数据库中。会议纪要的专家信息表见表5,会议纪要的专家关联数据和会议纪要的正文数据见表6。

表4 电子邮件的专家关联数据和电子邮件的正文数据

Table 4 Expert association data for email

字段名称	字段类型	字段说明
MESSAGE_ID	int	邮件的id
REFERENCES	json	以json形式存储被当前邮件回复过的其他所有邮件的id
FROM_EXPERT_IDS	int	此邮件的发件专家id
TO_EXPERT_IDS	json	以json形式存储此邮件的收件专家id
CC_EXPERT_IDS	json	以json形式存储此邮件的抄送专家id
EMAIL_TIME	datetime	邮件发送时间
EMAIL_CONTENT	text	邮件正文内容

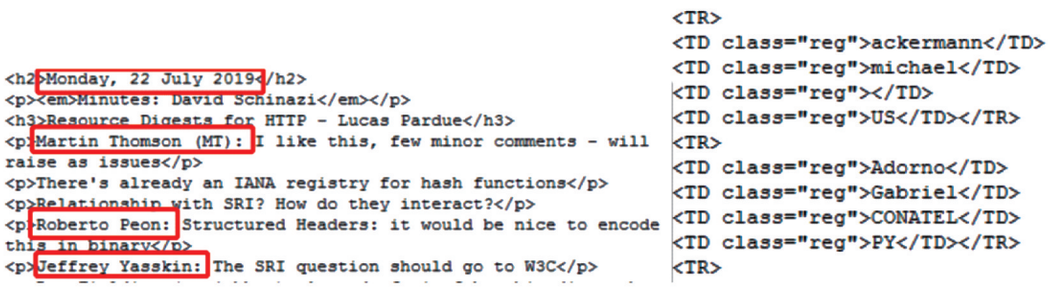

图4 会议纪要和的参会专家名单网页源代码

Fig. 4 Web source code of meeting minutes and list of experts

表5 会议纪要的专家信息

Table 5 Expert information for meeting minutes

字段名称	字段类型	字段说明
EXPERT_ID	int	专家的id
NAME	varchar(32)	专家姓名
COMPANY	varchar(32)	单位
AREA	varchar(128)	国家或地区缩写

表6 会议纪要的专家关联数据和会议纪要的正文数据

Table 6 Expert association and body data for meeting minutes

字段名称	字段类型	字段说明
MINUTES_ID	int	会议纪要的id
EXPERT_ID	json	以json形式存储此会议纪要中所有专家的id
MINUTES_TIME	datetime	会议时间
MINUTES _CONTENT	text	会议纪要正文内容

3 数据预处理模块

数据预处理模块首先对数据进行清洗、规范和修复等处理,再对同名专家的数据进行查重和合并,最后将专家间的关联关系进行合并,为后续进行数据挖掘和分析提供可靠和有价值的数据。

3.1 数据清洗

数据清洗包括对数据的格式进行规范、对错误

数据进行修复,本模块按以下步骤进行处理。

对专家的数据格式进行规范,包括统一大小写、统一换行符、规范邮箱格式、规范日期格式、规范标点符号等。

数据中存在因专家姓名是缩写而无法确认、会议纪要中部分专家记录错误等情况,本模块负责对这些错误数据进行修复。比如在IETF的第104次大会的某份会议纪要中,一位名为"Dough M"的专家只出现了一次,而"Doug M"的专家出现了多次,在总的参会专家名单中没有"Dough M",只有"Doug M",所以判断"Dough M"为错误数据,本模块将其修复为"Doug M"。对于实在无法修复的,本模块将做丢弃处理。

3.2　同名专家合并

对数据进行清洗后,需要进一步对同名专家进行处理。同一位专家可能会在RFC标准文件、电子邮件和会议纪要中被多次记录,需要对同名专家进行查重和合并。本模块按如下流程进行:(1) 根据专家姓名、电子邮箱、单位等字段,本模块将完全相同的专家合并;(2) 电子邮件、会议纪要中的部分数据,专家姓和名的顺序可能互换,本模块会将专家的姓和名顺序调换,同时配合专家的电子邮件、单位等字段进行判断,以合并同名专家;(3) 电子邮件、会议纪要中的部分数据,存在专家姓名缩写的情况,本模块将对照RFC标准文件和总的参会专家名单等数据进行判断,以合并同名专家。

流程(1)和流程(2)采用了Hadoop的MapReduce框架[14]实现。MapReduce是一个高性能的分布式计算框架,适合对海量数据进行并行处理。框架分为Map和Reduce两个阶段[15-16]:Map负责海量数据输入、分解和并行计算,输出的结果为<Key,Value>键值对;Reduce根据Key值对Map结果进行汇总合并。流程(3)由于情况复杂,所以采用SQL脚本结合人工分析进行处理。,

3.3　专家关联关系合并

专家之间的关联关系也需要合并,本模块将RFC标准文件、电子邮件和会议纪要中专家的关联关系加权合并为一个关联关系,本系统中为这3种方式分别设定不同的调整参数:

$$A_{ij}=u*\sum \mathrm{R}_{ij}+v*\sum E_{ij}+w*\sum \mathrm{M}_{ij} \quad (1)$$

式中,A_{ij}代表专家i和j之间的总的关联关系,即总的权重,$\sum \mathrm{R}_{ij}$、$\sum E_{ij}$和$\sum \mathrm{M}_{ij}$分别是RFC标准文件、电子邮件和会议纪要的关联关系之和,并设定RFC标准文件的调整参数u最高,电子邮件的调整参数v其次,会议纪要的调整参数w最低。

4　社区发现模块

社区发现模块是根据IETF的实际情况,优化并实现了社区发现算法中的Louvain算法。

Louvain算法计算速度快、社区划分比较准确,适合大规模数据处理,是性能优异的社区发现算法之一[17]。结合IETF的实际情况,此算法需要进行优化和改进:首先,Louvain算法在迭代循环时,第1轮会遍历所有专家进行计算,实际上IETF中有部分专家和其他专家没有联系,或者只与单一专家有联系,导致了第1轮迭代效率较低,本模块对这类专家数据进行提前处理,从而提高迭代时间。其次,随着互联网技术的不断发展和专家的不断出现,IETF中一些早期数据的重要性有所降低,但Louvain算法并未考虑时间因素,因此本模块在Louvain算法中引入了时间衰减函数,降低早期数据的权重,从而能更好地计算出最新的专业领域社区与该专业领域中活跃的专家。

4.1　Louvain社区发现算法

Louvain算法是基于模块度的社区发现算法,是用模块度对社区划分的质量进行衡量[18-20]。模块度值在-1到1之间,模块度值越高说明社区划分的质量越高,社区划分的效果越好,本系统的目标就是使社区的模块度最大化。在社区算法中,专家是节点,专家间的联系是边。模块度的计算公式为

$$Q=\frac{1}{2m}\sum_{ij}\left[A_{ij}-\frac{k_i k_j}{2m}\right]\delta\left(C_i,C_j\right) \quad (2)$$

$$\delta(u,v)=\begin{cases}1, & \text{if } u=v\\ 0, & \text{else}\end{cases}$$

式中,$m=\frac{1}{2}\sum_{ij}A_{ij}$表示所有边的权重之和;$A_{ij}$表示

节点i与j之间边的权重；$k_i=\sum_j A_{ij}$表示所有与节点i相连的边的权重之和，即节点i的度数；C_i表示节点i所属的社区；$\delta(u, v)$的值代表u与v是否为同一个社区，如果同一个社区则值为1，否则为0。

Louvain算法的具体步骤如下：

（1）初始时，将每个节点设定为一个单独的社区，此时节点数目等于社区数目。

（2）遍历每个节点i，尝试将节点i加入其相邻节点j所在的社区，计算加入前和加入后模块度的变化量ΔQ，选择ΔQ最大的相邻节点，如果$\max\Delta Q>0$，则把节点i加入到相邻节点j所在的社区，否则保持不变，如式(3)所示：

$$\Delta Q=\left[\frac{\sum_{in}+k_{i,in}}{2m}-\left(\frac{\sum_{tot}+k_i}{2m}\right)^2\right]-\left[\frac{\sum_{in}}{2m}-\left(\frac{\sum_{tot}}{2m}\right)^2-\left(\frac{k_i}{2m}\right)^2\right] \tag{3}$$

式中，$\left[\frac{\sum_{in}+k_{i,in}}{2m}-\left(\frac{\sum_{tot}+k_i}{2m}\right)^2\right]$代表把节点$i$加入节点$j$所在的社区后的模块度，$\left[\frac{\sum_{in}}{2m}-\left(\frac{\sum_{tot}}{2m}\right)^2-\left(\frac{k_i}{2m}\right)^2\right]$标识节点$i$作为单独社区时的模块度，两者之差$\Delta Q$表示模块度的变化量。$\sum_{in}$表示社区内部所有边的权重之和；$\sum_{tot}$表示其他社区节点与社区内部节点相连的边的权重之和；$k_{i,in}$表示节点i与社区内所有节点的权重之和。

（3）重复步骤（2），反复迭代直到所有节点的变动都不能使各社区的模块度增加为止。

（4）对社区进行压缩，将每个社区看作一个新的节点，原节点之间跨社区的联系转为新节点之间的联系，其权重也转换为新节点之间边的权重。

（5）在新节点上重复执行步骤（2）（3）和（4），直至模块度也不再增加为止，算法结束。

每一轮迭代都会把社区压缩为新节点，这样边和节点的数量会越来越少，除了第1次迭代慢以外，后面的历次迭代速度会越来越快，从而实现快速收敛。

4.2 优化专家数据

通过分析IETF数据，发现有些专家向公共邮件列表发过邮件，但没有得到回应，本文称为孤立专家；也有些专家只与某个固定专家有过联系，与其他专家没有联系，本文称为边缘专家。本系统是用来计算出活跃专家，所以孤立专家没有计算的意义，提前删除；对于边缘专家，系统将其合并到其有联系的专家的名下，作为一个节点。通过这两个方法减少了Louvain算法的迭代时间[21]。

4.3 增加时间衰减函数

当前互联网发展日新月异，新专家和新技术不断涌现，而IETF中一些早期的RFC、邮件和会议纪要所涉及的数据已经不是当前最新的，在Louvain算法中应考虑时间因素。专家之间通过RFC标准文件、电子邮件和会议纪要建立的联系，其紧密程度随着时间而逐步降低，相应的权重也随之降低。本系统在改进Louvain算法时，引入了权重的时间衰减函数。当前广泛采用的是指数形式的时间衰减函数，即权重随时间的衰减速度和当前的权重值成正比[22-23]。

由式(4)可以推导出到权重时间衰减函数，权重会随着时间呈现指数形式逐渐放缓的衰减[24]，即式(5)。

$$\frac{\mathrm{d}A}{\mathrm{d}t}=-kA_0 \tag{4}$$

$$A(t,k)=A_0e^{-kt};t\in(0,\infty) \tag{5}$$

式中，A_0表示初始权重；k为遗忘速率，表征遗忘曲线变化快慢的参数；t表示时间参数，本系统为从建立联系到现在的年数。

将式(5)代入式(1)得到式(6)：

$$A'_{ij}=u\sum R_{ij}\mathrm{e}^{-kt}+v\sum E_{ij}\mathrm{e}^{-kt}+w\sum M_{ij}\mathrm{e}^{-kt} \tag{6}$$

A'_{ij}表示了考虑时间因素时的权重值。然后将式(6)带入式(3)的各项有权重的参数中，即$\sum'_{in}$、$\sum'_{tot}$、$k'_{i,in}$、k'_i，最终得到新的ΔQ，见式(7)：

$$\Delta Q'=\left[\frac{\sum'_{in}+k'_{i,in}}{2m}-\left(\frac{\sum'_{tot}+k'_i}{2m}\right)^2\right]-\left[\frac{\sum'_{in}}{2m}-\left(\frac{\sum'_{tot}}{2m}\right)^2-\left(\frac{k'_i}{2m}\right)^2\right] \tag{7}$$

4.4　优化后的Louvain算法

本系统针对IETF的实际情况,对Louvain算法进行了优化:(1)优化专家数据,删除与其他人没有联系的孤立专家,合并只有一个联系人的边缘专家;(2)考虑时间因素,引入权重的时间衰减函数。通过这些改进,初步缩短了Louvain算法的迭代时间,并且考虑了时间因素。改进后的Louvain算法将专家划分到不同的社区,在每个社区内,根据权重高低对专家进行排序。优化后的伪代码如下,伪代码中专家对应Louvain算法中的节点,专家之间的关联关系对应算法中的节点之间的联系。

```
Data origData=GetAllExpertsData();//得到RFC标准文件、电子邮件和会议纪要中的专家信息和专家关联数据
Data optimData = OptimizeExperts(origData);//优化专家数据,删除孤立专家、合并边缘专家
Data communityData = UpdateByTimeDecay(optimData); //根据时间衰减函数的公式(6),修改权重,并初始化社区数据
while(true){
    double Q1 = CalcQ(); //计算当前社区的模块度Q值
    foreach(expertId){  //遍历每一个专家
        List deltaQs=CalcDeltaQs(expertId);//根据公式(7),依次将此专家加入相邻的所有社区,计算每次加入相邻社区前后的模块度变化量 ΔQ' ,最后得到 ΔQ' 的集合列表
        if ((maxDeltaQ=Max(deltaQs) )> 0 ) {//如果 ΔQ' 集合列表中最大的maxDeltaQ大于0
            JoinCommuinty(expertId, maxDeltaQCommunity);//将此专家加入maxDeltaQ对应的社区
    }else{
        MaintainExpert(expertId); //否则此专家所在社区不变
    }
    if (IsAllExpertMaintain()) { //如果所有专家在上一个循环中所在社区都不发生变化,即所有专家的变动都不能使模块度增加
        communityData = Compress(communityData); //对社区进行压缩
        double Q2= CalcQ(); //计算新社区的模块度Q值
        if (IsEqual(Q2,Q1)) { //Q1与Q2值相同时Louvain算法结束
            return SaveResults();//保存发现的社区,算法退出
        }
    }
}
```

经过本模块的计算,就能将专家划分到各个社区中。

5　热点发现模块

当专家划分到社区后,每个社区对应一个专业领域方向,本模块负责发现每个专业领域方向的活跃专家,采用TF-IDF算法挖掘出这些专家当前研究的热点。

5.1　社区内活跃专家

在前一个模块中,已经得到了每个专家的 $k'_{i,\text{in}}$ 值,即此专家 i 与社区内所有专家的权重之和, $k'_{i,\text{in}}$ 代表了这个专家和社区内部其他专家联系的程度,联系越频繁、越是近期的联系,则 $k'_{i,\text{in}}$ 值越高。本模块通过 $k'_{i,\text{in}}$ 值确定社区内专家的排名,将排名靠前的专家设定为活跃专家,下一步将专门分析其研究热点。

5.2　TF-IDF算法

本模块采用TF-IDF(term frequency-inverse document frequency,词频-逆文本频率)算法来挖掘活跃专家所有的RFC标准文件、电子邮件和会议纪要的正文,从而发现活跃专家当前研究热点。

TF-IDF是一种数据检索与挖掘技术,能简单高效地提取文档中的关键词,被广泛用于关键词提取、文本挖掘以及信息检索等领域[25-26]。TF-IDF定

义为

$$tf\text{-}idf_{i,j}=tf_{i,j}*idf_i \tag{8}$$

$tf_{i,j}$ 表示某一个特定词语在文档中出现的频率,见式(9):

$$tf_{i,j}=\frac{n_{i,j}}{\sum_k n_{k,j}} \tag{9}$$

$n_{i,j}$ 为词语 t_i 在文档 d_j 出现的次数,$\sum_k n_{k,j}$ 为文档 d_j 中所有词语出现的次数之和。

idf_i 表示特定词语的普遍程度,当有大量文档包含这个词时,其 idf_i 值越低,反之则 idf_i 值越高,见式(10):

$$idf_i=\log\frac{|D|}{1+\left|\left\{j:t_i\in d_j\right\}\right|} \tag{10}$$

$|D|$ 为文档库中的文档总数,$1+\left|\left\{j:t_i\in d_j\right\}\right|$ 代表包含词语 t_i 的文档数目,为了避免当词语 t_i 不在文档库时分母为零的情况,分母需加1。

tf-idf 值是随着词语在某个文件中出现的频率增加而正比增高,但同时也会随着词语在整个文档库中出现的频率增加而反比降低。

5.3 实现TF-IDF算法

本模块采用了Spark的MLlib接口实现TF-IDF算法。MLlib是基于Spark的可扩展的机器学习库,实现了分类、回归、聚类与协同过滤等多种算法。算法步骤如下:(1)将活跃专家的RFC标准文件、电子邮件组成文档库;(2)对文档库进行分词;(3)过滤掉英文的常用词语,如an、the、there等;(4)计算文档库里的每一个词语的TF-IDF值,当同一个词语在不同文档中有不同的TF-IDF值时,取最大值;(5)TF-IDF值按从大到小排序,取前若干个词语作为此活跃专家当前的研究热点。

系统调用MLlib算法接口的伪代码如下:

```
val contentList = GetContent(expertId);//根据专家Id获得该专家的文档库
val docList = Filter(Spit(contentList)); //对文档库进行分词,并过滤掉常用词语
for( var doc <- docList){      //按文档库执行遍历
    val tf = (new HashingTF()).transform(doc).cache(); //计算TF值
    val idf = new IDF().fit(tf);   //计算IDF值
    val tfidf = idf.transform(tf);  // 计算TF-IDF值
    wordlist.Add(tfidf) ;//  将TF-IDF值存入结果列表
}
Sort(wordList); //结果列表排序获取研究热点
```

按同样的方式,可以计算得到各个社区内每个活跃专家的研究热点,跟踪和掌握当前互联网标准的最新成果。

6 结果与分析

设计开发了基于IETF的专家社区发现原型系统。系统已经开始在测试平台运行,该系统由8台服务器组成,其中1台安装了Jsoup和MySQL,部署了数据解析和存储模块;3台安装了Hadoop集群,部署了数据预处理模块;1台安装了Java,部署了社区发现模块;3台安装了Spark集群,部署了热点发现模块。服务器的CPU分别为16核到32核的至强处理器,内存为24G到32G,操作系统为CentOS7;软件包括Jsoup1.11.1、MySQL5.7.24、Hadoop2.8.4、Jdk1.8.0_131和Spark2.3.0。本系统的具体环境配置见表7。

试运行期间,系统通过数据解析和存储模块,分析了IETF建立以来的RFC标准文件约8500个,撰写RFC标准文件的专家约6000人,电子邮件约162.8万封,收发邮件的专家约355.1万人次,会议纪要约6000份,发言专家约11.7万人次;而陈启伟论文的实验中获取的IETF标准文件专家、电子邮件和会议纪要的时间是2013—2017年,RFC标准获取了约1000个,邮件约45万封,会议纪要约2000份,相比之下本系统获取数据的时间跨度更久,数据量更大。系统通过数据预处理模块对数据进行清洗、对专家进行合并。系统通过社区发现模块对Louvain算法进行了优化和改进,Louvain算法的平均耗时从72.96 s提升到68.79 s,提升约5.72%,经过此模块计算得到的社区数量约为200

表7　系统的环境配置

Table 7　System environment configuration

服务器ID	测试服务器IP地址	硬件参数	操作系统	部署软件	部署模块
1	192.168.13.7	CPU:16核　内存:32G	Centos7.1	Jsoup1.11.1 MySQL5.7.24	数据解析和存储模块
2-4	192.168.13.8-10	CPU:16核　内存:24G	Centos7.1	Hadoop2.8.4	数据预处理模块
5	192.168.13.11	CPU:32核　内存:32G	Centos7.1	Jdk1.8.0_131	社区发现模块
6-8	192.168.13.12-14	CPU:16核　内存:24G	Centos7.1	Spark2.3.0	热点发现模块

个;吴祖峰基于叶子社区剪枝策略的Louvain算法,其优化提升的时间约为4.1%,两个系统提升幅度近似。系统通过热点发现模块发掘出各个专业领域的活跃专家,并采用TF-IDF算法挖掘出其研究热点。例如,在某个社区比较活跃的专家Russ Housley,通过本系统计算,发现其研究的热点为cryptographic、certificate、security、algorithm与protocols等,均是和网络加密协议、安全算法相关;对照他发表的90篇RFC标准文件、电子邮件和会议纪要,特别是近期发表的RFC,如2018年的RFC 8419“Use of Edwards-Curve Digital Signature Algorithm (EdDSA) Signatures in the Cryptographic Message Syntax (CMS)”、2019年的RFC 8619“Algorithm Identifiers for the HMAC-based Extract-and-Expand Key Derivation Function (HKDF)”和RFC 8649“Hash Of Root Key Certificate Extension”,多数也是围绕相关的技术领域;再分析这个社区内其他活跃专家,最终确认这个社区是围绕网络安全的专业领域,社区内的活跃专家当前以制定发布新的互联网加密协议和算法为主。再例如,某个社区的活跃专家Lou Berger,通过本系统计算其研究热点为IP、GMPLS、MPLS、RSVP与Switching等,均是IP层网络基础架构和基于IP的控制信令协议等技术;对照其发表的42篇RFC标准文件电子邮件和会议纪要,特别是近期的RFC,例如2019年的RFC 8529“YANG Data Model for Network Instances”和RFC 8629“Dynamic Link Exchange Protocol (DLEP) Multi-Hop Forwarding Extension”,也基本围绕相关技术领域;同样的方法再分析这个社区的其他活跃专家,最终确认这是一个围绕互联网基础架构的社区,其活跃专家以研究基础层网络信令协议为主。通过对各个专业领域方向的活跃专家研究热点的分析,可以较好地了解和掌握当前互联网标准化领域的最新动态。

7　结论

本原型系统实现了对IETF专家的社区发现,提供了跟踪和掌握互联网国际标准中最新研究热点的途径,可以深入掌握当前互联网标准的制定情况和最新动态,从而进一步推进中国制定更多的互联网国际标准。下一步,本原型系统将继续完善,如对Louvain算法并行化处理、增加结果的可视化展现等。

参考文献(References)

[1] 刘东信, 朱崇业. 英文版科技期刊审稿和组稿专家的遴选——基于SCI数据库高特征因子(Eigenfactor)期刊的分析[J]. 中国科技期刊研究, 2012, 23(3): 369-372.

[2] 蒲姗姗. 基于知识互补的科研合作专家推荐模型研究[J]. 情报理论与实践, 2018, 41(8): 96-101.

[3] 李江, 李东, 冯培桦, 等. 基于专长吻合度、学术影响力与社会关联值的专家推荐模型研究[J]. 情报学报, 2017, 36(4): 338-345.

[4] IETF. IETF About[EB/OL]. (2009-07-17) [2019-01-20]. https://www.ietf.org/about.

[5] 中国通信标准化协会. Internet 工程任务组[EB/OL]. (2004-02-12) [2019-01-20]. http://www.ccsa.org.cn/organization/intro.php?org=IETF.

[6] Girvan M, Newman M E J. Community structure in social and biological networks[J]. Proceedings of the National

Academy of Sciences, 2002, 99(12): 7821-7826.
[7] Vincent D B, Jean-Loup G, Renaud L, et al. Fast unfolding of communities in large networks[J]. Journal of Statistical Mechanics: Theory and Experiment, 2008, 2008 (10): 155-168.
[8] 吴祖峰, 王鹏飞, 秦志光, 等. 改进的Louvain社团划分算法[J].电子科技大学学报, 2013, 42(1): 105-108.
[9] Traag V A. Faster unfolding of communities: Speeding up the louvain algorithm[J]. Physical Review E, Statistical, Nonlinear and Soft Matter Physics, 2015, 92(3): 032801.
[10] 吴卫江, 李沐南, 李国和. Louvain算法的并行化处理[J]. 计算机与数字工程, 2016, 44(8): 1402-1406.
[11] 李贤, 许大卫. 基于聚类中心度的网络数据划分研究[J]. 自动化技术与应用, 2018, 37(9): 86-90.
[12] 陈启伟. 基于IETF的互联网国际组织社交数据挖掘方法研究[D]. 北京: 中国科学院大学计算机网络信息中心, 2018.
[13] 李沐南. Louvain算法在社区挖掘中的研究与实现[D]. 北京: 中国石油大学, 2016.
[14] 郝树魁. Hadoop HDFS和MapReduce架构浅析[J]. 邮电设计技术, 2012(7): 37-42.
[15] 杨煜, 赵成贵. 基于Hadoop MapReduce并行近似谱聚类算法研究与实现[J]. 计算机应用与软件, 2015, 32 (8): 17-21.
[16] 李玉林, 董晶. 基于Hadoop的MapReduce模型的研究与改进[J]. 计算机工程与设计, 2012, 33(8): 3110-3116.
[17] Newman M E J. Fast algorithm for detecting community structure in networks[J]. Physical Review E, 2004, 69 (6): 066133.
[18] Girvan M, Newman M E J. Community structure in social and biological networks[J]. Proceedings of the National Academy of Sciences of the United States of America, 2002, 99(12): 7821-7826.
[19] Illes J F, Daniel A, Gergely P, et al. Weighted network modules[J]. New Journal of Physics, 2007, 9(6): 180.
[20] Newman M E J. Analysis of weighted networks[J]. Physical Review E, 2004, 70(5): 056131.
[21] 夏玮, 杨鹤标. 改进的Louvain算法及其在推荐领域的研究[J]. 信息技术, 2017(11): 125-128.
[22] 钮瑗瑗, 程国振, 齐超, 等. 基于遗忘曲线的用户影响力时效性度量方法[J]. 计算机应, 2017, 37(S1): 18-22.
[23] 李桃迎, 张鑫, 陈燕. 基于艾宾浩斯遗忘曲线的零售商品模糊关联分析[J]. 计算机应用研究, 2018, 35(2): 462-465.
[24] 于洪, 李转运. 基于遗忘曲线的协同过滤推荐算法[J]. 南京大学学报(自然科学版), 2010, 46(5): 520-527.
[25] 蒋永新, 孙爱莉. 基于tf-idf方法的图情学核心期刊学科特征分析[J]. 情报资料工作, 2009(1): 89-92.
[26] 路永和, 李焰锋. 改进TF-IDF算法的文本特征项权值计算方法[J]. 图书情报工作, 2013,57(3): 90-95.

A prototype for discovering the communities of IETF experts

YE Xiaorong[1], SHAO Qing[2]

1. Institute of Scientific and Technical Information of China, Beijing 100038, China
2. KNET Co., Ltd., Beijing 100190, China

Abstract The IETF is a relatively authoritative technical standardization organization in the global Internet. In order to analyze the standardization experts in various professional fields in the IETF and identify the research hotspots of the current Internet standards, a prototype for discovering the communities of the IETF experts is designed. This system analyzes and stores various data sources of the IETF, cleans and merges the data, and uses the optimized and improved Louvain algorithm for the community discovery. Then the system divides the experts into a number of different Internet professional fields, and mines the active experts in various professional fields, and uses the TF-IDF algorithm to mine its research hotspots. So the system can provide accurate and dynamic information for China to develop the Internet international standards.

Keywords IETF; community discovery; Louvain algorithm

（责任编辑　陈广仁）