计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.191200164

众包协作流程的恢复方法

王 扩 王忠杰
哈尔滨工业大学计算机学院企业与服务智能计算研究中心 哈尔滨 150001
(hitwangkuo@sina.com)

摘 要 众包是一种应用群体智慧的分布式问题求解机制,目前广泛存在于以人工智力活动为基础的互联网应用场景中,利用互联网上大量用户的群体协作来解决单人无法解决的复杂问题。众包协作机制对开源领域的发展起到了很大的作用。以开源软件的开发维护过程为例,参与人员通过特定平台共同完成代码编写、bug修复等关键任务。与传统业务过程管理(Business Process Management,BPM)不同,众包场景下的协作流程存在流程结构无法预先确定、协作参与者数量无法预知、协作时间与结果无法提前预测等挑战,这给众包协作的效率与质量控制带来了极大的困难。针对众包协作过程中多个参与者按时间次序产生的一系列协作行为(体现为自然语言形式的文本),利用自然语言处理和人工智能等方法,提出了众包协作过程恢复算法,并以开源软件开发领域bug修复过程中的人员合作为案例进行了实证研究,尝试用3种方法对协作流程进行恢复,分别是文本近似度、关键词汇匹配以及神经网络意图理解恢复算法;然后定量对比了各个流程恢复算法的准确度,得出应用关键词匹配算法进行协作流程恢复的准确度最高、效果最好的结论;最后实现将需要分析的协作流程进行协作流程恢复以及可视化的工作。该研究有助于众包流程的协调者(例如开源项目管理者)更直观地理解众包协作中的问题求解过程,从中发现协作的典型模式,从而可为新的众包任务的协作过程的性质作出准确预测。

关键词:众包;协作流程;开源软件开发;Bug修复;流程恢复
中图法分类号 TP315

Crowdsourcing Collaboration Process Recovery Method

WANG Kuo and WANG Zhong-jie
Research Center on Intelligent Computing for Enterprises & Services,Harbin Institute of Technology,Harbin 150001,China

Abstract Crowdsourcing is a distributed problem solving mechanism using group intelligence. It is widely used in Internet application scenarios based on artificial intelligence activities,using large groups of users on the Internet to work together to solve complex problems that cannot be solved by one person. Taking the development and maintenance process of open source software as an example,participants jointly complete key tasks such as code writing and bug repair through specific platforms. Different from traditional business process management (BPM),collaborative processes in the crowdsourcing scenario face challenges such as undetermined process structure,and unpredictable timing and results,which bring great difficulties to the efficiency and quality control of crowdsourcing collaboration. In this paper,aiming at a series of collaborative behaviors produced by multiple participants according to the time sequence (embodied as text in the form of natural language),natural language processing and artificial intelligence are used to propose a restoration algorithm for the process of crowdsourcing collaboration. An empirical study is carried out on the case of personnel cooperation in the process of bug repair in the field of open source software development. The collaborative process of recovery is visualized,and the accuracy of process recovery algorithm is quantitatively compared. This research can help coordinators of crowdsourcing process (such as open source project managers) to understand the problem solving process more intuitively,and find the typical patterns of collaboration,so as to make an accurate prediction for the nature of the collaborative process of the new crowdsourcing task.

Keywords Crowdsourcing,Collaborative process,Open source software development,Bug fix,Process restore

到稿日期:2019-12-27 返修日期:2020-05-08 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金(61772155)
This work was supported by the National Natural Science Foundation of China (61772155).
通信作者:王忠杰 (rainy.wang@gmail.com)

1 引言

近年来，随着互联网的发展而产生的众包模式在各个领域都起到了越来越重要的作用。众包这一概念是 Howe 于 2006 年提出的[1]。在众包兴起前流行的合作模式是外包，传统外包常常以合同形式把任务委派给合适的人或者机构来完成，并且外包的业务往往是计算机能够完成的，例如软件开发等。随着时间的推移，部分任务使用外包处理不仅效率低，并且质量差，因而诞生了众包合作模式。众包是直接把任务发布到互联网上，通过开放式集合互联网上未知的大众共同来解决任务的新兴合作模式[2]，由于和真实存在的现实场景类似，因此其具备广阔的应用潜景和市场。目前来说，众包的主要研究领域包括：人机交互[3-4]、自然语言处理[5-6]、数据库[7]、信息检索[8-9]和计算机理论[10]等。

众包模式下参与者多是自发自愿加入到开发过程中的，因而和传统开发模式完全不同。众包通过公开的方式来召集互联网中愿意加入的自愿者，是一种典型的分布式的解决机制。传统开发过程中的组织形式在众包领域并不适用，因此针对众包的开发协作过程的研究对于提高众包的合作效率十分重要。

众包广泛的应用背景使其成为了近年来的研究热点。许多重要会议都包含众包相关的主题。目前已有的相关研究从不同角度对众包进行了探索，如 Kittur 等分析了众包在实时响应、同步协作等方面面临的挑战[11]，Doan 等总结了众包系统在万维网上的应用[12]。新兴的众包开发模式还对开源的发展起到了极大的促进作用。许多开源开发平台都是以协作式众包模式进行组织开发的。整个开发过程由多人自发合作完成，每次合作都会有相应的协作流程。不同合作过程中的协作流程是不一样的，但目前现有的基于众包的合作平台忽视了对协作流程的分析和展现。以开源软件的开发和维护为例，其基本上都是以协作式众包为合作模式的。然而，无论是开源的开发协作平台还是维护平台，都没有明确地展现人员协作流程的相关内容。人员协作基本上都是以时间为次序进行顺序展示，这种展现方式对于分析了解不同参与者如何共同完成众包模式的开发十分不便，更不利于针对协作流程进行众包开发效率提升的研究。因此，本研究选取以协作式众包方式组织合作的开源领域中 bug 修复过程为实例，尝试对协作式众包的协作过程进行研究，通过对协作过程中人员交流的文字内容进行分析，从而恢复出真实的协作流程并进行可视化。

Bugzilla 是目前众多有关开源项目 Bug 修复追踪平台中较为知名的，许多针对开源领域 Bug 修复方面的研究都是在这个平台上进行的。例如，有关 Bug 不可重现的问题研究[13]、Bug 报告分配的统一框架提出[14]以及 Bug 跟踪系统中测试人员的排名分析[15]等。这些最新的研究都是基于 Bugzilla 提供的数据进行的。

本研究进行实例分析的数据选取自目前热度很高的 bug 跟踪平台 Bugzilla[16]，这类缺陷追踪平台由于时间累积形成了大量的历史数据[17]。目前针对这类数据分析的研究越来越多，例如工业界的微软率先依据长期累积的数据挖掘，从重复的缺陷报告中获取信息[18]。

本文的研究目标定位在以协作式众包模式来组织合作的 bug 修复的协作过程上，并且按照以下次序针对 bug 修复过程中的人员协作流程进行分析和研究。首先应用爬虫从 Bugzilla 平台获取要进行分析的评论内容历史数据，然后针对这些历史数据进行人工标注，建立起真实的协作过程。由于 Bugzilla 平台上 bug 修复历史数据数以千万计，全部实现人工标注是不现实的。因此需要设计协作流程的恢复算法，其能够通过读入协作过程中人员交流的内容信息直接将协作流程恢复出来。本文按照通常判定文本关联的方法设计并实现了 3 种算法来进行协作流程恢复，分别是文本近似度恢复算法、关键词汇匹配恢复算法和神经网络意图理解恢复算法。这 3 种算法都结合自然语言技术，通过对评论内容中包含的自然语言进行分析，从而生成 bug 修复过程中人员的真实协作流程。然后将恢复结果与人工标注样本进行比对，分析各个算法的优劣。为了更加清楚地展示修复过程中的人员协作流程，还采用了可视化方法对恢复出来的协作流程进行直观展示。这样就能够将隐藏在文本自然语言中的协作信息挖掘出来，为更好地理解 bug 修复中的协作方式提供指导。具体的实现流程如图 1 所示。

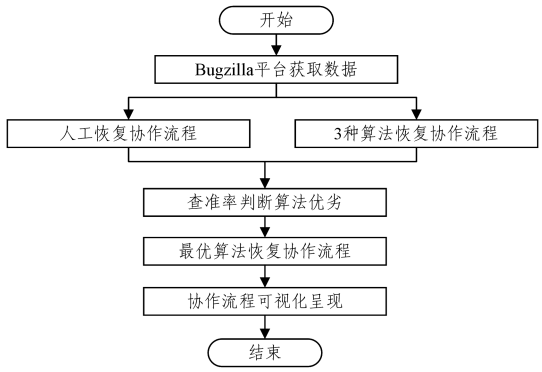


图 1 众包协作过程恢复

Fig. 1 Process recovery of crowdsourcing collaboration

2 众包协作流程恢复算法

依据合作过程中的人员交流信息可以对众包的协作过程进行分析。人员交流基本上都是通过自然语言进行的，例如，在开源项目 bug 修复平台 Bugzilla 上修复参与者就是通过评论进行交流的。目前来说，针对众包协作过程进行的研究并不多。本研究开创性地结合自然语言处理技术针对协作过程中的评论对话内容进行分析，尝试恢复出真实的协作流程。结合现有的自然语言处理技术以及进行人工标注真实协作逻辑的过程分析，本文提出了 3 种协作流程恢复算法，分别是通过文本相似度、关键词汇匹配以及神经网络意图理解协作流程恢复算法。

2.1　文本近似度流程恢复算法

文本相似度算法来源于真实情况中人们判别文本相关性的思路之一。通常来说,人们在讨论同一件事情的时候会提到较多的相同或相关词汇。因此可以认为,如果两段文本的相似度越高,那么在讨论同一件事情时,其具备关联性的可能性就越大。

算法 1　近似度流程恢复算法(人工标注样本 50 组,原始评论内容 50 组)

输入:人工标注好的 50 组 bug 修复流程,人工标注相同的 50 组 bug 的修复评论信息

输出:恢复好的 50 组 bug 修复流程,近似度恢复效果与人工标注样本相比查准率

1. 将文本内容进行分词,将词汇映射到相应的向量空间中。
2. 判断当前评论是否是问题描述。
3. 如果是,则直接跳出,进行下一条循环。
4. 如果不是,则分别计算该评论与之前评论的近似度。
5. 选取与当前评论近似度最大的评论作为关联评论。
6. 在两个评论间建立联系,组成一条边。
7. 依次执行,直到所有边都循环完毕。
8. 将恢复出的与人工标注得出的协作流程进行对比来确定正确边的数量。
9. 获得恢复出的协作流程以及近似度算法恢复结果的查准率。

算法 1 重点介绍了将文本映射到向量空间,然后通过向量空间来进行余弦近似度计算的过程。

2.2　基于关键词汇的流程恢复算法

由于开源项目的 bug 修复过程中人员的交流协作是通过语言进行的,因此若想要对开源项目 bug 修复协作流程进行恢复,就需要先针对修复参与者协作过程中的语言交流内容进行分析。为了将恢复出的算法进行优劣比对,首先要对修复过程进行人工恢复并且将结果标注记录下来。这个过程中发现,一些词汇对于判定两段文本间的关联关系能够起到直接帮助。通过记录和分析,最终汇总出 3 类词汇能够对文本关联度判定起到明显的分辨作用,分别为:关联词汇集合表、重要词汇表和指示词汇表。这 3 类词汇的区别在于各自建立文本关联的方式不同,定义如下。

定义 1(关联词汇集合表)　关联词汇集合表中包含多组词汇的集合。每组词汇集合中包含多个相关词汇,当两段文本中同时出现属于同一组词汇集合中的词汇时,则这两段文本具备相关性。

定义 2(重要词汇表)　重要词汇表中包含有多个词汇。这些词汇多是具备标志性的实词,当两个文本中同时出现词汇表中的同一个词汇时,则这两段文本具备相关性。

定义 3(指示词汇表)　词汇表中包含多个词汇或词组。这些词汇或词组后面往往带有这段评论所针对的评论标志信息。评论文本中如果包含这些词汇或词组,就能够明确知道该评论是针对之前的具体哪条评论发出的。

这 3 类词汇表中元素的选取是先通过标注过程中的人工选取,然后通过自然语言处理中的相似词汇发掘来进行扩展。最终形成较为完整的 3 类判定文本关联度的词汇表。

然后通过对未知关联关系的文本中包含词汇表中词汇的情况就可以对文本间的联系进行判定,进而能够恢复出符合真实协作情况的协作流程。

算法 2　重点词匹配恢复算法(fw_1List,fw_2Lis t,fw_3List,原始评论内容 50 组)

输入:作为判断文本相关性基础的三类关键词汇表 fw_1List,fw_2List,fw_3List,以及需要进行恢复流程的 50 组 bug 修复评论信息

输出:恢复好的 50 组 bug 修复流程,以及用该方法恢复出的流程与人工标注流程对比获得的查准率

1. 将 3 类关联词汇表内容从 Mysql 中读入到列表集合中。
2. 判定每条评论内容包含 3 类关联词汇表中词汇的情况并且记录到相应的属性中。
3. 针对现有评论与已出现的评论,根据其是否包含相同或相近的词汇内容来判定关联度。
4. 依据关联度建立评论间联系,并且以边的形式记录下来。
5. 依次执行直到所有边都循环完毕。
6. 将恢复出的与人工标注得出的协作流程进行对比来确定正确边的数量。
7. 获得恢复出的协作流程以及近似度算法恢复结果的查准率。

算法 2 通过文本内容包含词汇表中词汇的情况来判定文本相关度,然后依据文本相关度来进行协作流程恢复。

2.3　基于意图理解的流程恢复算法

人工神经网络是通过模仿动物神经网络的特征进行信息处理的数学模型。目前神经网络模型已经应用到了多个领域,其在文本识别中也有出色的表现。

目前关于神经网络意图理解已有较为完善的第三方包和程序支持,在应用过程中需要特别注意参数的选取。最优参数的获取往往凭借知觉和多次尝试,其中隐层节点数目的选取尤为关键。目前提出的比较能得到大家认可的隐层节点确定方案主要如式(1)-式(4)所示:

$$s=\sqrt{i(o+2)}+1 \tag{1}$$

$$s=\log_2 i \tag{2}$$

$$s=\sqrt{i*o} \tag{3}$$

$$s=\sqrt{i+o}+a \tag{4}$$

其中,s 表示隐层节点数,i 为输入层节点数,o 为输出层节点数。

本研究先通过这些公认较为有效的公式进行计算,获取可能的隐层节点数,然后在此基础上结合对其他参数的调整来尝试获取最优参数组合。在获取最优参数组合后,通过对已分类样本进行学习,就可以获取网络中各节点的最优取值,然后就能够对新文本类别进行判定。

算法 3　神经网络恢复算法(WLIst,ParaCom,原始评论内容 50 组)

输入:原始分类数据 Wlist,神经网络模型的各个参数 ParaCom,以及需要进行恢复流程的 50 组 bug 修复评论信息

输出:恢复好的 50 组 bug 修复流程,以及用该方法恢复出的流程与人工标注流程对比获得的查准率

1. 将原始分类数据读入到相应的列表集合中。
2. 尝试各个参数的具体取值,选取效果最好的一组作为参数集合 ParaCom。

3. 依据原始数据和神经网络参数取值，运行神经网络意图理解模型，令其取得最优效果。
4. 将当前最优效果的神经网络模型记录到 jason 格式文件中。
5. 对当前的评论文本内容应用神经网络模型来判断其所属分类。
6. 在属于相同分类的评论内容间建立联系，确定边。
7. 依次执行直到所有边都循环完毕。
8. 将恢复出的协作流程与人工标注流程查询正确边的查准率。
9. 获得恢复出的协作流程以及近似度算法恢复结果的查准率。

通过依次实现这 3 种协作逻辑恢复算法，并且与人工标注的协作逻辑进行比对，就能够知道这 3 种算法的优劣。选取最优的算法进行协作逻辑恢复后，再进行可视化处理，就能够使原本只按照时间顺序排列的交流过程恢复出更接近真实情况的协作流程。

3 实验分析

针对众包协作流程进行分析主要是通过参与人员在合作过程中的语言交流。因此，需要应用自然语言处理的技术结合众包项目参与人员交流的具体方式来进行研究。本研究以开源项目 Bug 修复平台 Bugzilla 为研究实例。通过分析得知，修复参与人员在以众包模式对 bug 进行修复的过程中主要是通过评论进行交流的。本文以此为基础设计了针对 Bug 修复评论信息的协作流程恢复实验，并通过这个实例来展现如何对众包模式中的人员协作流程进行分析。

3.1 数据获取

第一步是尝试将 Bugzilla 中的协作过程交流讨论数据收集起来形成数据库，然后就是如何对自然语言内容进行识别，判断两段对话的关联性，从而依据关联性恢复出真实的协作流程。

本研究获取的实验数据包括来自 ASF 基金会的项目 ants，appache，JMter 和来自 eclipse 基金会的 JDTText，PlatformCVS，还有来自 Mozilla 基金会的 Bugzilla，Calendar，Firefox，FirefoxOS，共计 9 个项目，26 万条评论数据。每条评论数据都包含所属 BugID、时间、内容、参与者和位置标识。

由于目前并没有有关流程恢复结果的数据集，因此只能采用人工标注的办法先确定作为判定标准的真实协作流程数据集，然后设计流程恢复算法，并将恢复结果与人工标注结果进行对比，分析其效果。

3.2 近似度恢复算法

文本近似度建立文本间联系的思路是通过计算两个短文本映射到空间向量中的余弦近似度值来判断两个文本的相关度。实现这个算法需要先进行分词以获得词频向量，接下来就是使用余弦相似度计算各个词频向量的相似程度。

表 1 是 ID 为 7320 的 bug 各个评论间余弦相似度的计算结果。

表 1 余弦近似度计算结果

Table 1 Results of cosine approximation

ID:7320	D	C1	C2	C3	C4	C5
C6	0.08	0.05	0.33	0	0	0
C5	0.12	0.13	0	0.1	1.0	
C4	0.12	0.13	0	0.1		
C3	0.07	0.1	0			
C2	0.01	0				

依据判定结果将当前评论与之前出现的评论中与其近似度最高的评论建立关联，并且依次进行该过程直到最后一条评论结束。这样就可以将该次 bug 修复的完整协作流程恢复出来了。

3.3 关键词恢复算法

在进行人工标注的过程中，可以发现有些词汇能够明显体现出两段文本是否具备关联性。先将这些词汇记录下来，然后将这些词汇按照建立文本关联的不同方式分为不同的类别。最后通过人工收集各个类别的词汇或短语并进行汇总扩展，就能够得到一个完整的通过评论文本词汇特征识别文本关联的词汇表集合。通过对评论文本内容中包含词汇的识别就可以分析出文本间的关联关系。

其中关联词汇集合需要应用自然语言处理中的 word2vec 先将词汇转换为向量，然后使用 gensim 包从语料库中查找与关联词汇集相近的所有词汇扩展关联词汇集。将最初获得的关联词汇集合整理为扩展后的关联词汇集合表。词汇集合表部分内容如表 2 所列。

表 2 关联词汇集合表

Table 2 Key vocabulary

Type	Group	Content
Associated Vocabulary	1	Provide, feature, functionality, offer, maintain, generate, require, accommodate, providing, feature, theme, genre, anime, hardware, implementation, capabilities, interfaces, processors
	2	Resolved, resolution, reopening, confirmed, terminated, ignored, addressed, fulfilled, protocol, priority, resolutions, procedure, legislation, nawab
	3	Target, targeted, milestone, targets, trajectory, firing, dive, rear, aimed, outraged, exploited, employed, timeline, postseason
Important Vocabulary	无	whiteboard parameter Bug ***
		Attachment ***
Index Vocabulary	无	Related bug Comment on attachment ***
		Duplicated of this bug ***

在进行实证研究的过程中，应用 Python 程序实现了对协作交流过程中出现的这些特殊词汇的识别，然后通过识别结果来判定两文本间是否有存在关联性的可能，并且通过这些联系恢复出 bug 修复的协作流程，最后将结果与人工标注样本进行对比，就可以观察到恢复效果。

3.4　神经网络恢复算法

应用神经网络意图理解来恢复协作流程的关键在于参数的选取，主要方法是通过尝试来获取最优参数配置。经过多次实验，使用神经网络意图理解尝试恢复协作交流过程的最优参数是每个类别取样本数限制为 15，隐含层数目使用式(1)进行计算，下降梯度为 0.035，循环次数限制在 100 000 次以内，Dropout 取 0.3。最终恢复结果的最优查准率为 0.755，达到目前已试参数的最优效果。

3 种不同的协作流程恢复方法在 50 组和 30 组实验数据上将恢复结果与人工标注样本对比得出的查准率如表 3 所列。

表 3　恢复算法效果对比

Table 3　Comparison of accuracy of recovery algorithms

（单位：%）

Three kinds of algorithms	Accuracy of 50 samples	Accuracy of 30 samples
Text approximation	44.5	47.1
Vocabulary matching	74.1	76.1
Text intention comprehension	75.5	71.0

从表 3 可以明显看出，目前最优的恢复流程方案就是文本意图理解和词汇表匹配。两者的准确度不相上下，在 50 组和 30 组样本下分别取得最优查准率。

选取最优的协作流程恢复方案将流程恢复后，再通过可视化就能够直接将流程呈现出来。

3.5　协作流程恢复及可视化实例展示

以 Bugzilla 平台为实例，众包模式下协作流程恢复及可视化大致要经过 7 个步骤：依次是编写爬虫从 Bugzilla 平台获取实验数据，人工对协作流程进行恢复，实现 3 个协作流程恢复算法，与人工标注结果进行比对以确定最优的流程恢复算法，应用最优恢复算法将需要分析的 bug 修复协作流程进行恢复，最后将恢复结果进行可视化呈现。

首先要做的是从 Bugzilla 平台获取数据。在这个平台上的人员交流协作信息是通过评论进行的，原始的网页显示数据如图 2 所示。

图 2　缺陷管理平台评论内容

Fig. 2　Defect management platform comment content

然后编写爬虫程序，获取协作流程分析的数据并存储到 Excle 格式的文件中，如表 4 所列。

表 4　评论内容抓取结果示例

Table 4　Example of comment content fetching results

Comments logo	Reviewers	Content
Comment 1	Frédéric Buclin	Probably a good idea.
Comment 2	Max Kanat-Alexander	Yeah, I think we're all in agreement with this. I'd never thought of it, but it's a good idea.
Comment 3	Max Kanat-Alexander	It's no need to suppress it with a Parameter, because you can set and then disable user preferences.
Comment 4	Eiren Smith	You're right, editsettings. cgi is definitely the way to do this.
Comment 5	Aleksandr Derevianko	Also need this feature. How to vote/increase priority for this one?

然后从中随机选取约 50 个 bug 修复的相关评论信息进行人工标注。将标注结果记录在纸上并将其以树形结构存储到 Mysql 中，以实现数据持久化。标注结果如表 5 所列。

表 5　人工标注恢复协作流程

Table 5　Manually annotate recovery collaboration process

Key	BugID	ID	Name	Layer	Parent_ID
1	244913	0	D	0	
2	244913	1	C1	1	0
3	244913	2	C6	1	0
4	244913	3	C2	2	1
5	244913	4	C3	3	3
6	244913	5	C4	4	4
7	244913	6	U1	5	5
8	244913	7	C5	6	5

其中，BugID 表示评论所示 bug，ID 是评论在该 bug 修复中的唯一标识，用来确定评论间的关联关系；Name 是评论在 bug 中位置标识的简单映射；Layer 是该评论在恢复好的流程中的所在层数；Parent_ID 表示与该评论相关的评论的唯一标识。通过这种方式就能够将协作流程存储到数据库中进行持久化。

需要使用 3 种协作流程恢复算法读入从网页获取的原始数据，应用各自的算法将协作流程恢复出来；这样根据评论关联边的信息就可以将协作流程恢复出来；再将人工标注的协作流程从 Mysql 中读入，存储到与算法恢复出的协作流程同样格式的数组中；这样就可以通过与人工标注的协作流程对比分析 3 个算法的优劣性。通过表 3 可以分析出目前恢复协作流程效果最好的是重点词汇匹配恢复算法。使用这种算法，将假设需要进行协作流程分析的 30 组 bug 修复数据进行协作流程恢复，将恢复结果以与人工标识同样的结构存储到 Mysql 数据库中。

编写可视化算法，从 Mysql 中读取恢复好的协作流程，并进行可视化。可视化算法设计为可以按照两种需求进行，一种是可视化指定 BugID 的 bug 修复协作流程，另一种是可视化评论数目超过指定要求的 bug 修复协作流程。

由获取的数据分析得知，在同一个 bug 修复过程中评论的标识与评论内容是一一对应的，由于可视化过程中要考虑显示效果以及美观等方面，因此将 bug 修复过程中参与人员的合作内容抽象为评论标识、评论时间和评论人这 3 个可以代表一条评论的关键特征信息。然后应用一个矩形表示一条参与者发出的评论，再应用线性连接就可以直观地展现恢复出的协作流程可视化效果。下面随意选取 3 个满足条件、且评论数大于 5 条的 bug 修复协作流程可视化进行展示。如图 3—图 5 所示。

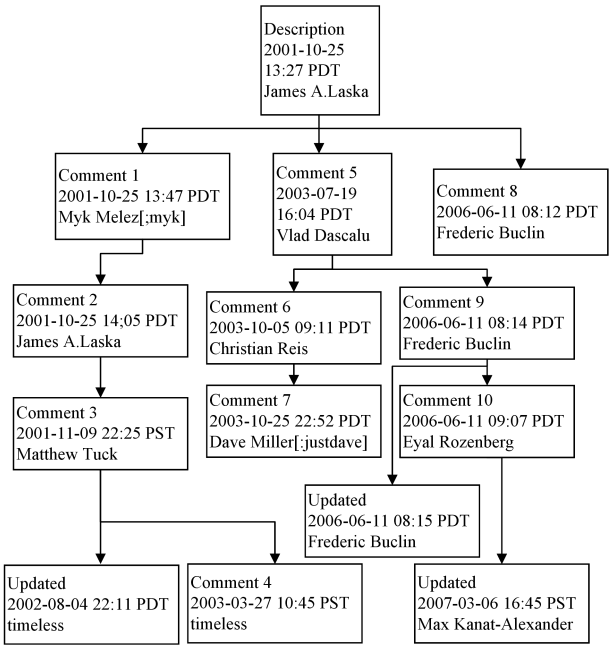


图 3 协作流程可视化效果-BugID106778

Fig. 3 Visualization of collaborative processes-BugID106778

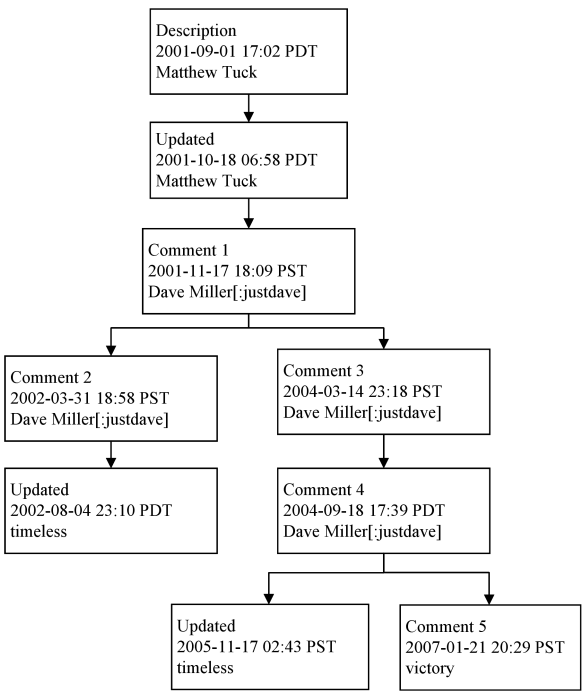


图 4 协作流程可视化效果-BugID97955

Fig. 4 Visualization of collaborative processes-BugID97955

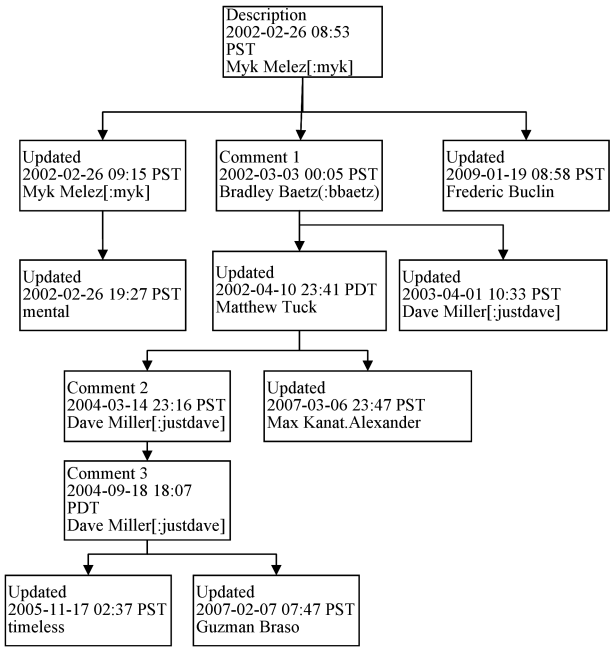


图 5 协作流程可视化效果-BugID127871

Fig. 5 Visualization of collaborative processes-BugID127871

如图 3—图 5 所示，通过协作流程恢复就实现了将原本按照时间顺序排列的交流过程恢复成更能够体现真实协作中人员交互过程的树结构，并且能够将需要观察分析的协作流程通过可视化的方式展现出来。和原始的 bug 追踪平台下的评论展示效果相比，流程恢复并且可视化之后能够清晰地展示人员在修复 bug 过程中的交互过程。

通过这个实例可以看出，对协作流程恢复并且进行可视化对于众包模式下合作过程的分析起到了更为直观的指导作用，为众包模式的进一步分析奠定了基础。

结束语 本文重点对以众包模式组织的合作过程中的人员协作流程进行了分析，并采用以协作式众包合作方式组织的 bug 修复追踪平台 Bugzilla 为研究实例。通过获取 Bugzilla 上存在的真实数据并进行分析，可以看出，在进行 bug 修复的过程中，人员之间的交互信息能够体现协作特征。针对这部分内容进行研究，对于增加人员协作的默契以及提高 bug 修复效率具有价值。

本文在众包领域针对协作流程的分析目前仍较为薄弱的前提下，提出了以开源领域 bug 协作过程恢复为实例的众包协作过程恢复算法。本文研究方法结合当前已有的自然语言处理技术尝试设计算法将协作流程恢复出来，并且通过比对选出其中的最优算法，将恢复出的协作流程进行可视化。为针对人员协作进行的研究提供了思路及相应的成果支持。不论是对于后续进行众包模式下人员交流过程的进一步研究，或者仅想应用可视化观察 bug 修复过程的人员协作的项目管理人员，本文研究方法和结果都能够提供有效帮助。

参 考 文 献

[1] HOWE J. The rise of crowdsourcing [J]. Wired Magazine, 2006, 14(6): 1-4.

[2] ZHAO Y X,ZHU Q H. Evaluation on crowdsourcing research: current status and future direction [J]. Information Systems Frontiers,2012,11(1):1-18.

[3] KOCH G,FULLER J,BRUNSWICKER S. Online crowdsourcing in the public sector: how to design open government platforms [C]// Proceedings of The 4th International Conference on Online Communities and Social Computing. Orlando, USA, 2011: 203-212.

[4] KHEER J,BOSTOCK M. Crowdsourcing graphical perception: Using mechanical turk to assess visualization design[C]// Proceedings of the 28th International Conference on Human Factors in Computing Systems. Atlanta,USA,2010:203-212.

[5] PARAMESWARAN A G,GARCIA-MOLINA H,PARK H,et al. CrowdScreen: Algorithms for filtering data with humans [C]// Proceedings of the ACM SIGMOD International Conference on Management of Data. Scottsdale,USA,2012:361-372.

[6] VENETIS P,GARCIA-MOLINA H,HUANG K,et al. Maxalgorithms in crowdsourcing environments[C]// Proceedings of the 21st World Wide Web Conference. Lyon,France,2012:989-998.

[7] LAWS F,SCHEIBLE C,SCHUTZE H. Active learning with amazon mechanical turk[C]// Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing. Edinburgh,UK,2011:1546-1556.

[8] LIU X,LU M,OOI B,et al. CDAS:A crowdsourcing data analytics system[J]. Proceedings of the VLDB Endowment,2012,5(10):1040-1051.

[9] KAZAI G. In search of quality in crowdsourcing for search engine evaluation[C]// Proceedings of the 33rd European Conference on IR Research. Dublin,Ireland,2011:165-176.

[10] CHAWLA S,HARTLINE J D,SIVAN B. Optimal Crowdsourcing contest[C]// Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms. Kyoto, Japan, 2012: 856-868.

[11] KITTUR A,NICKERSON J V,BERNSTEIN M S,et al. The future of crowd work[C]// Proceedings of the 2013 ACM Conference on Computer Supported Cooperative Work. San Antonio,USA,2013:1301-1318.

[12] DOAN A,RAMAKRICHNAN R,HALEVY A Y. Crowdsourcing systems on the world-wide web [J]. Communications of the ACM,2011,54(4):86-96.

[13] ANJALI G,NEETU S. An empirical study of non-reproducible bugs[J]. International Journal of System Assurance Engineering and Management,2019,10(5):1186-1220.

[14] ZHAO Y,HE T K,CHEN Z Y. A Unified Framework for Bug Report Assignment[J]. International Journal of Software Engineering and Knowledge Engineering,2019,29(4):607-628.

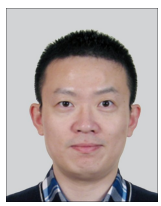
[15] HUI L,GAO G F,CHEN R,et al. The Influence Ranking for Testers in Bug Tracking Systems[J]. International Journal of Software Engineering and Knowledge Engineering,2019,29(1):93-113.

[16] PRESSMAN R S,INCE D. Software engineering : a practitioner ' s approach[M]. New York:McGraw-hill,1992.

[17] XIE T,PEI J,HASSAN A E. Mining software engineering data [C]// Proceedings of the 29th International Conference on Software Engineering. Minnesota. USA,2007:172-173.

[18] ZHOU J,ZHANG H Y,LO D. where should the bugs be fixed? more accurate information retrieval-based bug localization based on bug reports [C]// the 34th International Conference on Software Engineering. Switzerland,2012:14-24.

WANG Kuo, born in 1990, Ph. D, is a member of China Computer Federation. His main research interests include social software engineering and crowdsourcing, software warehouse mining and service recommendation.



WANG Zhong-jie, born in 1978, Ph. D, professor, Ph. D supervisor, is a member of China Computer Federation. His main research interests include service computing, service engineering Internet services and cloud computing, social networking services, social software engineering and crowdsourcing, software warehouse mining.