软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2018,29(1):69−88 [doi: 10.13328/j.cnki.jos.005377]

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

众包软件测试技术研究进展*

章晓芳[1,2], 冯 洋[2], 刘 峘[1], 陈振宇[2], 徐宝文[2]

[1](苏州大学 计算机科学与技术学院,江苏 苏州 215006)

[2](计算机软件新技术国家重点实验室(南京大学),江苏 南京 210023)

通讯作者: 章晓芳, E-mail: xfzhang@suda.edu.cn

摘 要: 众包测试是一种新兴的软件测试方式,得到了学术界和工业界的广泛关注.系统地总结了近年来众包软件测试研究的学术文献以及工业界实践进展:首先,从学术文献涉及的研究主题演变、涵盖的软件测试问题和众包测试流程、采用的实验对象及测试人员规模等多个角度对相关文献中提出的技术和方法进行了汇总;然后,从测试领域、测试对象、工人召集方式、绩效考核方式等方面分析对比了当前应用最广泛的 20 个众包测试商业平台;最后,探讨了众包软件测试的未来发展趋势、机遇和挑战.

关键词: 软件工程;软件测试;众包;众包测试;众包平台

中图法分类号: TP311

中文引用格式: 章晓芳,冯洋,刘峘,陈振宇,徐宝文.众包软件测试技术研究进展.软件学报,2018,29(1):69−88. http://www.jos.org.cn/1000-9825/5377.htm

英文引用格式: Zhang XF, Feng Y, Liu D, Chen ZY, Xu BW. Research progress of crowdsourced software testing. Ruan Jian Xue Bao/Journal of Software, 2018,29(1):69−88 (in Chinese). http://www.jos.org.cn/1000-9825/5377.htm

Research Progress of Crowdsourced Software Testing

ZHANG Xiao-Fang[1,2], FENG Yang[2], LIU Di[1], CHEN Zhen-Yu[2], XU Bao-Wen[2]

[1](School of Computer Science and Technology, Soochow University, Suzhou 215006, China)

[2](State Key Laboratory for Novel Software Technology (Nanjing University), Nanjing 210023, China)

Abstract: Crowdsourced software testing is an emerging testing method which has drawn extensive attention in both industrial and academic community. This paper systematically summarizes the academic literatures and industry practice in recent years. This article first summarizes the related literatures from the perspectives of the research topics including software testing, crowdsourcing test process, experimental subjects and scale of crowdsourcing. It also compares total of 20 widely used crowdsourced software testing commercial platforms, and discusses their task domains, subjects, open call forms and performance evaluation forms. Finally, the paper presents the future trends, issues and opportunities for crowdsourced software testing.

Key words: software engineering; software testing; crowdsourcing; crowdsourced testing; crowdsourced platform

正如著名的Linus定律所述,“只要足够多的眼球关注,就可让所有软件缺陷浮现”[1].在软件产品的测试过程中,软件产品管理者希望能够快速获得大量反馈,以尽快修复软件产品缺陷并改进软件产品质量.然而,招募/训练测试人员的巨大成本往往使得召集大量专业测试人员变得困难.此外,由于软件产品的快速更迭,特别是移动应用产品紧凑的生命周期,导致软件测试的周期被急剧压缩.因此,如何快速获得测试反馈,特别是大量真实用

* 基金项目: 国家自然科学基金(61772263, 61772014, 61572375); 软件新技术与产业化协同创新中心资助

Foundation item: National Natural Science Foundation of China (61772263, 61772014, 61572375); Collaborative Innovation Center of Novel Software Technology and Industrialization

收稿时间: 2017-06-23; 修改时间: 2017-08-01; 采用时间: 2017-08-30; jos 在线出版时间: 2017-10-09

CNKI 网络优先出版: 2017-10-09 16:20:53, http://kns.cnki.net/kcms/detail/11.2560.TP.20171009.1620.004.html

户的反馈以帮助产品的改进,同时以较低成本高效地完成测试任务,是当前软件测试领域的困难之一.

众包技术(crowdsourcing)可以在极大程度上有效解决软件测试领域的这一难题.众包是互联网带来的一种分布式问题解决和生产组织模式[2],通过整合互联网上未知大众和机器来解决机器难问题.众包这一概念自2006 年提出后,已经成功应用在人机交互、数据库、自然语言处理、机器学习和人工智能、信息检索、计算机理论等学科领域[3−9].在软件工程领域,众包技术也逐步应用于软件工程的各个环节中[10−13],特别是在软件测试方面,大量在线工人参与完成测试任务,可以提供对真实应用场景和真实用户表现的良好模拟,测试周期短且测试成本相对较低[14].这些优点使得众包测试技术得到了学术界和工业界的广泛关注,广大研究人员密切关注众包软件测试的理论和方法,同时也涌现了许多众包软件测试的商业平台.

目前,已有一些研究人员从不同角度对众包的相关研究工作进行了总结[3,15−20],重点讨论了众包的工作流程以及众包技术的应用,特别地,Mao 等人对众包技术在软件工程中的应用进行了全面的概述[11].在此基础上,本文则重点关注众包技术在软件测试中的研究与实践进展:一方面,系统地收集了众包软件测试的相关学术文献,并从研究主题演变、涵盖的软件测试关键问题和众包测试流程、采用的实验对象及测试人员规模等多个角度对相关技术和方法进行了汇总;另一方面,针对当前应用最广泛的 20 个众包测试平台,从平台的测试领域、测试对象、工人召集方式、绩效考核方式等方面展开了分析和对比.本文力求从学术研究进展和工业实践进展两方面对众包测试技术进行全面的总结,并探讨众包测试技术的未来发展方向.

本文第 1 节描述众包及众包测试的背景知识.第 2 节系统地总结众包软件测试的相关文献.第 3 节则分析主流的 20 个众包软件测试平台.第 4 节讨论众包软件测试的发展趋势、机遇和挑战.第 5 节对全文进行总结.

1 背 景

1.1 众 包

2006 年,Howe 首次提出众包的概念,定义“众包”是:“一个公司或机构把过去由专职员工执行的工作任务,通过公开的 Web 平台,以自由自愿的形式外包给非特定的解决方案提供者群体,共同来完成的分布式问题求解模式”[2].随后,众多学者从不同的角度给出了众包的定义[21],总体而言,众包的基本特征包括:采用公开的方式召集互联网大众;众包任务通常是计算机单独很难处理的问题;大众通过协作或独立的方式完成任务;众包是一种分布式的问题解决机制.

众包的主要参与者包括任务请求者和任务完成者(也称为众包工人),他们通过任务(task)联系到一起.众包的工作流程包括任务准备、任务执行和任务答案整合这 3 个阶段.其中,

- 任务准备阶段包括任务请求者设计任务、发布任务,工人选择任务;
- 任务执行阶段包括工人接收任务、解答任务、提交答案;
- 任务答案整合阶段则包括任务请求者接收或拒绝答案、整合答案[3].

在众包工作流程中,众包任务的发布和答案的收集通常借助众包平台来完成.众包平台主要分为两大类:一类是商用的众包平台;另一类是社交网络、论坛等社交平台.已有研究数据表明:商用众包平台的应用明显多于社交平台[3].早期的众包平台通常是指“问答系统”平台,如维基百科、百度知道等.近年来,由于早期众包平台所支持的任务类型较为单一,已经无法适应当前数据类型的多样化与任务的复杂化,因此涌现了一批大型的在线工作招募与任务分包管理平台,例如 Amazon Mechanical Turk(Mturk)、CrowdFlower 等.这些商用众包平台根据任务请求者和任务完成者的不同需求提供相应的服务,不但带来了新的技术革命,更创造了巨大的市场经济价值.

1.2 众包软件测试

在众包软件工程领域,研究人员已经提出了大量的相关应用技术和应用场景.近期,Mao 等人对众包软件工程领域进行了全面的概述,涵盖了大量众包软件工程的文献和相关平台介绍,并给出了众包软件工程的定义如下[11]:“众包软件工程是指由大量潜在的、未定义的在线工人以公开召集的形式,承担外部软件工程任务的行

为.”基于 Mao 等人的综述工作,本文以众包软件工程的定义派生出众包软件测试的定义.我们使用术语“众包软件测试”来表示“支持软件测试的所有众包活动”,也就是说,所有支持软件测试的众包方法、技术、工具和平台都属于众包软件测试领域.

在众包软件测试活动中,主要参与者包括任务请求者(requester)、众包工人(crowd worker)和众包平台方(platform)[11].众包测试平台作为第三方,为任务请求者和众包工人提供在线系统.任务请求者首先提交待测软件和测试任务至众测平台;众测平台将测试任务分发给合适的众包工人,或者众包工人通过众测平台选择感兴趣的任务来完成;众包工人完成测试任务后,将测试结果以测试报告的形式提交至众测平台.与传统的软件测试报告类似,众包测试报告通常已事先定义好格式,包括状态、报告者、测试环境、测试输入、预期输出、错误描述、建立时间、优先级、严重程度等字段.此外,众包测试报告往往还要求众包工人提供相应的使用截图以帮助后期错误定位和调试.众测平台的工作人员(质量审核人员)将对收集到的大量测试报告进行审查和整理,并将测试结论反馈给任务请求者.通常,将由任务请求者对测试报告进行最终确认,并决定是否支付给相应的众包工人一定的酬金.

较之其他众包任务,众包软件测试任务对三方参与者都提出了新的要求:测试任务请求者需要精心设计测试任务以吸引众包工人并获得理想的测试结果;众包工人通常需要对待测对象有一定的了解,并具备相应的软件操作和测试技能;众包平台则面临着大量测试报告的质量审核和汇总的困难.众包软件测试作为一种新兴的软件测试手段,在很大程度上解决了传统软件测试所面临的用户多样性不足、反馈较少、应用场景不够真实等问题,但也对传统的测试技术提出了新的挑战.

自 2009 年众包技术首次被应用于 QoE(quality of experience)测试以来,如何使用众包技术来解决传统软件测试技术面临的困难并提高现有软件测试技术的效率,逐渐成为软件测试领域的研究热点之一.众包技术先后在 QoE 测试、可用性(usability)测试、GUI 测试、性能测试、测试用例生成等测试子领域中得到普遍应用.同时,针对众包软件测试技术所带来的新问题和相应解决方案也引发了众多研究学者的持续关注.鉴于众包技术已经广泛应用于软件测试任务,并引起了学术界和工业界的共同关注,迫切需要对当前众包软件测试研究过程中所使用的方法、技术、工具和平台进行系统的总结.

2 众包测试文献总结

2.1 文献收集与汇总

本文采用以下流程完成了文献的收集、筛选和分类汇总,如图 1 所示.

首先定义文献的纳入标准:该文献采用众包的方式解决了软件测试中的问题或开展了关于众包测试过程的研究.此外,该文献应公开发表在会议、期刊、技术报告或书籍中.针对 2016 年 12 月之前公开发表的有关文献,我们进行了以下 3 个方面的搜索.

(1) 在线搜索以下 9 个数据库:ACM Digital Library、IEEE Xplore Digital Library、Springer Link Online Library、Wiley Online Library、Elsevier ScienceDirect、ProQuest Research Library、Google Scholar、中国知网和万方数据库.使用的搜索关键词为 crowd testing、crowdsourcing testing、crowdsourced testing 以及众包测试,同时在文献的标题、摘要、关键词和索引中进行搜索;

(2) 在线搜索软件工程的主要期刊和会议,具体包括 TOSEM、TSE、ICSE、ESEC/FSE、ASE、IEEE SW、IET、IST、JSS、SPE、SQJ、ISSTA、ICST,此外还包括了众包软件工程的两个专题会议 ISSC(Int’l Symp. on Software Crowdsourcing)以及 CSI-SE(Int’l Workshop on Crowdsourcing in Software Engineering).搜索时间范围从 2006 年 1 月~2016 年 12 月,以查找上一步无法检索到的相关文献;

(3) 基于上述两个步骤所获得的文献集合,逐一查看文献的参考文献中是否存在与众包测试相关的文献,即,从已收集文献的引用文献中识别是否有遗漏的文献.

其次,针对前一阶段获得的共计 123 篇文献进行人工筛选和过滤.通过分组阅读文献,再次应用纳入标准,手动过滤与众包测试不相关的文献.我们采用分组交叉确认的方式,每一篇文献的剔除都需要获得所有参与者的

一致认可.通过筛选后,共计 52 篇文献纳入本文后续的文献总结中.

最后,采用分组交叉确认的方式对这些文献进行了分类汇总,确认每篇文献所讨论的主题.在本文总结的共计 52 篇文献中,包含了 43 篇研究型论文[22−64]、7 篇平台/工具型论文[65−71]、2 篇综述/简介型论文[72,73].我们建立了一个包含所收集文献的元数据和文献全文的数据集.元数据包括作者、标题、出版年份、出版类型、文献中涉及的测试对象、测试人员规模、测试平台以及文献分类汇总的类别标签.基于这个数据集,我们对所收集的文献进行了分析.

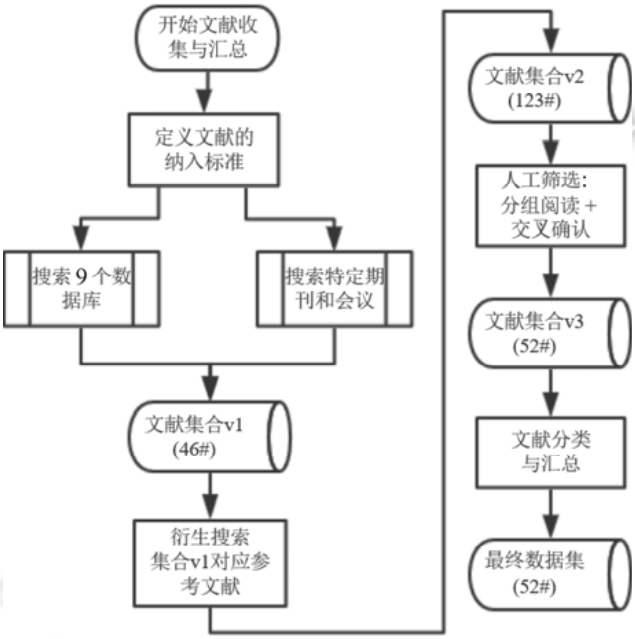


Fig.1 The workflow of collecting literature

图 1 文献收集汇总的流程图

本文总结的 52 篇文献的发表类型分布如图 2 所示,会议和期刊论文占到了绝大多数,特别地,作为一个新兴的研究领域,目前以会议论文为主(67%),其中不乏在软件工程领域的国际顶级会议上发表的研究论文.

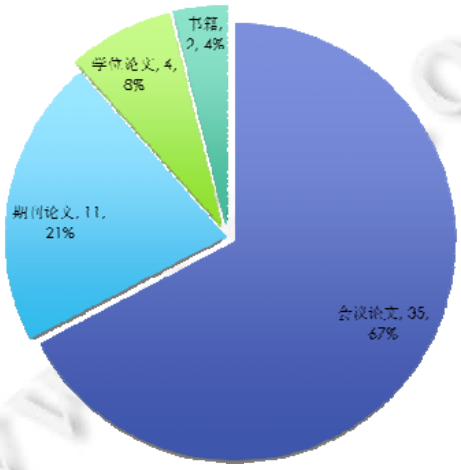


Fig.2 Publication types of surveyed literature

图 2 文献发表类型分布图

针对其中的43篇研究型论文,文献分类总览图如图3所示,我们将文献分为两大类:基于众包的测试优化研究,即,应用众包的方式解决测试问题或改进已有测试方法的研究[22−54];众包测试的自身优化研究,即,针对众包测试自身流程和相关机制的研究[55−64].具体而言,在基于众包的测试优化方面,涉及到以下7个主题:QoE测试、可用性测试、GUI测试、性能测试、测试用例生成、程序调试与修复、软件评估.在众包测试的自身优化方面,则主要考虑以下4个主题:测试工人的召集与管理、测试任务的分解与设计、报告的整合与处理以及其他.此外,我们还汇总了文献中实验验证过程中涉及的测试对象、测试人员规模、测试平台等信息.

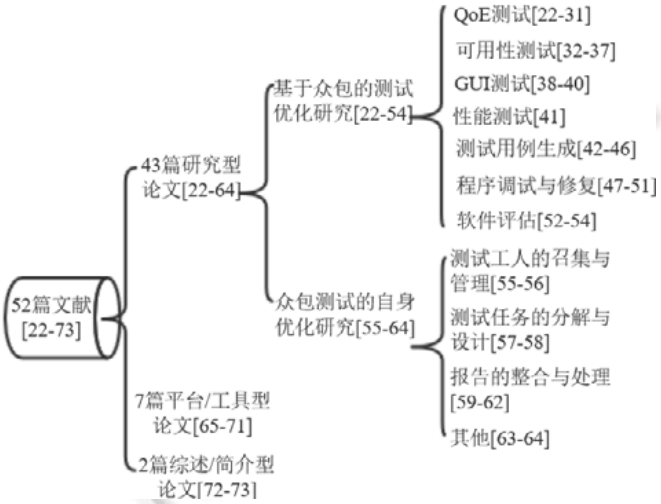


Fig.3 Classification scheme of surveyed literature

图3 文献分类总览图

根据论文主题的分类标签,图4分别从基于众包的测试优化研究(显示为正向纵轴)、众包测试的自身优化研究(显示为负向纵轴)这两个维度给出了文献的主题演化示意图.

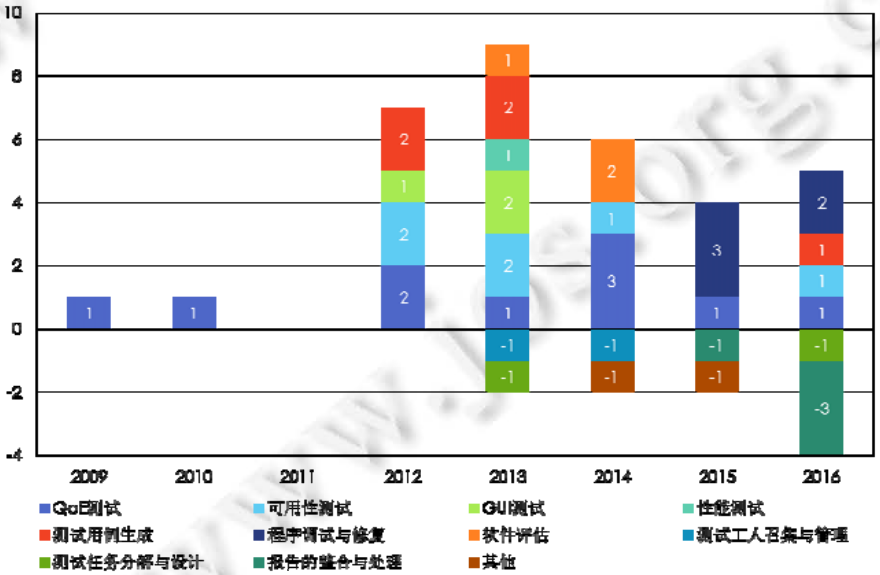


Fig.4 The evolution of research subjects

图4 研究型论文主题演化示意图

由图 4 可知:2009 年起,众包测试技术开始逐渐引起研究人员的关注.研究人员首先关注如何将众包技术应用于已有的测试方法中,即,希望借助众包手段来解决测试方法或测试过程中的困难.其中,QoE 众包测试和可用性众包测试一直都是研究的关注热点.此外,近年来,针对众包测试自身优化的研究论文有明显增长的趋势.自 2013 年起,开始有论文关注众包测试过程中测试工人召集与管理、测试任务的分解与设计以及报告的整合与处理.然而,对于测试任务的搜索或分发、结果的验证与重现等众包测试环节的研究仍然存在着空缺.

2.2 文献总结

众包测试作为一种新兴的测试模式,一方面可以用于有效改进已有的测试方法或测试流程;另一方面,对众包测试模式自身的研究也逐渐成为研究人员的关注重点.因此,本节将根据 43 篇研究型论文涉及的研究主题,从基于众包的测试优化、众包测试的自身优化这两方面,逐一给出相关文献的总结.

2.2.1 基于众包的测试优化

基于众包的测试优化主要包括针对特定测试类型的优化,如应用于 QoE 测试、可用性测试和 GUI 测试,以及针对测试流程的优化,如应用于测试用例生成、程序调试与修复等.

(1) QoE 测试

传统的 QoE 测试方法以人工测试为主,成本较高且耗时较长,因此,众包技术最先被应用于 QoE 测试领域,并引发了广泛、持久的研究[22−31].

众包 QoE 测试重点关注了多媒体研究中的 QoE 评估,先后出现了基于众包技术的框架[22]和基于 Web 的平台(quadrant of euphoria)[23,24].为了解决众包 QoE 测试中存在的问题,Hossfeld 等人详尽分析了对应解决方案,并对众包竞争和最佳实践方式进行了完善的设计[25,26].

QoE 测试结果受到不同平台下用户的地域分布、期望、使用环境以及所接受的训练课程等因素的影响.研究人员通过比较在付费众包平台(Microworkers.com)和非付费众包平台(Facebook)上执行的两个相同的高清视频质量实验的 MOS(mean opnion score)分数来检查不同类型的众包平台对 QoE 结果的影响,结果初步证明了区分平台的重要性[27].在此基础上,Gardlo 开发了一个与 Facebook 社交网络环境紧密相连的示例应用程序,并通过这个应用程序开展了多项研究[28].研究表明:目前,使用众包来评估网络多媒体服务的体验质量的方法在一定程度上受到测试结果的可信度和可靠性的影响.为此,研究人员提出了“Crowdsourcing 2.0”,力求显著提高测试结果的可靠性和活动的执行效率[29].

此外,研究人员尝试利用众包来收集网络服务 QoS(quality of service)数据:一方面,通过使用 Planetlab 节点作为工人来设计模拟众包方法;另一方面设计了 iTest 移动众包框架,使用 Android 手机测试 Web 服务.实验结果表明,这种众包方法可以获得更高的 QoS 预测精度.但在具体应用过程中,该方法需重点考虑用户真实环境带来的影响[30].

(2) 可用性测试

众包可用性测试的早期研究主要集中在对比传统实验室测试和众包测试的效果[32,33]上.有研究表明:众包可用性测试比传统方式更容易获取来自全球不同背景的数据,且众包可用性测试可以平行进行,能够显著降低测试成本;但是众包可用性测试的反馈数量和质量略低于传统的实验室测试方法[32].然而,也有研究人员通过在 MTurk 上使用众包可用性测试来评估基于 Web 的工具,获得了与传统实验室测试方法近似的结果[33].

在进行众包可用性测试时,研究人员希望借助更多的工具,以获取并记录更多的信息.例如:通过记录鼠标移动数据来分析用户遇到的潜在错误[34],或是通过捕捉摄像头下众包员工的犹豫特征来验证软件的可用性并分析软件中存在错误的可能性[35].此外,一个名为 CrowdStudy 的工具包可以全面支持众测模式下的网页测试:使用 CrowdStudy 可以更有效地招募众包工人,且能够在不同的条件下测试并评估网站,从而帮助实现基于众包的系统可用性测试[36].

自然地,众包可用性测试也广泛地应用于移动应用测试领域,广大的最终用户使用自有移动终端设备,在使用软件的过程中,自然地完成对该软件的可用性测试.研究指出:尽管众包测试能够有效降低测试时间和成本,但最终用户的测试往往集中在待测对象的功能和可用性方面,且测试结果与用户的实际使用环境紧密相关[37].

(3) GUI 测试

在 GUI 的测试过程中,自动化地生成 GUI 测试用例往往较为困难,而人工的 GUI 测试又费时费力[74],因此,如何开展持续的 GUI 测试,成为当前面临的挑战之一.众包技术被认为是一种有效开展持续 GUI 测试的手段.

为完成众包 GUI 测试,研究人员通过实例化多个运行着待测系统的虚拟机,让测试人员使用浏览器访问这些虚拟机以进行测试,从而把 GUI 测试众包给网络上的海量众包工人.尽管这种众包方法可能会产生较多不准确的测试结果,但研究人员相信,该问题可以通过改进众包任务的设计和限定众包工人得到良好的解决[38,39].

为缓解对于在线收集的数据质量的普遍担忧,Komarov 等人在 MTurk 上进行了 GUI 众包测试,通过和实验室测试结果的比对,证明了使用众包进行 GUI 测试的可行性[40].研究结果表明:在两种不同的设置下所收集到的数据不存在任何显著或实质性的差异,两组数据在数据差异性分布、任务完成时间、错误率、一致性等方面都非常相似.这些结果表明,GUI 众包测试是对传统实验室测试方法的有效补充.

(4) 性能测试

由于用户行为和执行环境的多样性,性能测试一直是软件测试中的难点之一.众包方式的兴起,为有效开展性能测试提供了一种新的可能.

微软的通信工具 Lync[75]被用作研究对象,探讨如何利用众包方式来完成软件产品的性能测试.相对于模拟各种用户操作、应用场景和测试环境的传统测试方法,研究人员将待测软件以受控的方式部署给一组用户,以收集性能数据并通过动态的代码插桩来检查感兴趣的场景.与实验室或模拟测试相比,众包方式收集的数据来源于用户的真实使用场景并反映了用户的实际经验,因此,数据更加多样化并具有代表性.研究表明:众包的方法可以有效地发现软件中的性能问题,并协助开发团队做出相应的决策[41].类似的众包性能测试研究还包括 Chrome 和 Firefox 通过内置遥测装置来实现的性能测试框架[76,77].

(5) 测试用例生成

测试用例生成是软件测试过程中的重要环节之一,众包技术也已成功地应用于测试用例生成方面.尽管已有许多自动测试用例生成技术,但这些技术仍面临着一些技术挑战,而这些挑战由人工来完成却并不困难.基于谜题的自动测试环境是一个易扩展的众包测试平台,该平台将对象突变和复杂约束求解问题分解为一些小的谜题,然后分发给众包工人求解.实验结果表明:这些小谜题可以由人工在短时间内高效地解决,同时获得比当前自动测试用例生成技术(jCUTE[78]、Randoop[79]、Pex[80])更高的代码覆盖率[42].

类似地,如何在社交编码网站,如 GitHub 上构建一个高质量的、充分的测试用例集,也是当前急需解决的问题之一.基于 GitHub 上的 Drive-By Commits(DBC)现象,一种基于 DBC 的测试用例构造方法通过使用众包机制,让有能力的用户(即便他们并不参与当前项目)来完成测试用例的设计和维护任务.这种方式极大地降低了项目所有者在测试用例构造和维护方面的开销[43].

此外,众包数据也可以帮助开发人员生成测试用例以重现缺陷.在移动应用程序中,系统崩溃(crash)的出现将极大地影响用户体验.然而,硬件的异构性、移动开发版本的碎片化以及用户的多样性,都对如何尽快地重现崩溃提出了挑战.文献[44]提出了一种基于众包的方法来支持应用程序开发人员自动再现最终用户面临的上下文相关的崩溃.该方法从用户的移动设备中获取众包数据,识别移动应用执行中的崩溃模式,最终生成能够再现崩溃的测试用例.

在测试用例的生成过程中,部分待测程序的预期输出难以获取,即,存在着测试预期输出问题(oracle problem)[81].为此,研究人员试图使用众包技术在一定程度上解决 Oracle 问题[45,46].Pastore 等人提出了 CrowdOracles,将反映当前程序的行为组织成断言,并将这些断言作为众测任务发布到众测平台上,由众包工人来评估这些断言的正确性.实验结果表明:CrowdOracles 可以有效缓解 Oracle 问题,并推进软件测试的自动化.但该方法高度依赖于清晰易懂的众包测试任务的设计,同时,该方法也面临着众包反馈质量的问题[45].

(6) 程序调试与修复

在程序调试过程中,开发人员希望能够利用众包测试信息以辅助开展程序调试.然而,众包数据量往往较大且存在着冗余,难以直接利用这些信息进行程序调试[82].为了解决这一问题,Crowd Debugging 方法基于 Stack

Overflow 社区中的 Q&A 信息,首先通过克隆检测与匹配分析来识别源码中错误的潜在位置,再通过无关语句识别、相似度分析等手段过滤误报结果,由此有效去除大量错误无关信息,最终生成一个包含源文件名称、错误潜在范围、错误修复说明等信息的 bug 报告来辅助开发人员进行程序调试[47].

针对程序调试这一“觅食”行为[83],Petrillo 等人则提出了 SDI(swarm debug infrastructure)框架,更关注实时信息对程序调试的帮助.该框架支持程序调试信息的收集、存储、共享及可视化,从而使得多个开发人员可以更方便地参加到同一个程序的调试任务中,利用群体智能更有效地完成程序理解、错误定位及修复[48,49].

在软件测试过程中,检测到缺陷后需要进行修复者推荐,即,寻找合适的开发人员来修复缺陷.该问题将研究如何从大量的开发人员中找到与缺陷相关的最合适的修复人员.已有研究人员结合推荐系统、文本挖掘以及社会化网络挖掘来进一步提高预测准确率[84,85].显然,各类众包平台上参与用户的信息及其对应的活动将有助于完成修复者推荐.研究人员利用开发人员在 Q&A 众包平台,例如 Stack Overflow 上的贡献,并结合其错误分配历史来决定修复错误的最佳候选开发人员.与已有方法相比,这种基于众包平台信息来评价开发人员专业技能的方法具备了更好的预测准确率[50,51].

(7) 软件评估

传统的软件评估方法很大程度上依赖于开发人员,且通常仅招募一批精英用户群体作为用户代表.同时,目前的方法在预测和模拟实际使用环境方面受到很大限制,特别是对于移动应用和云计算等频繁动态变化、交互的系统.因此,研究人员开始探讨如何由以开发者为主导的传统评估方式向以用户参与为主导的评估方式转变,众包模式为快速获得多用户的反馈提供了一种有效的方式.

众包方式尤其适用于评估复杂且可变的软件系统,因为这些系统将会在不同的、甚至不可预知的环境中工作.通过众包工人的迭代反馈,可以丰富并保持开发者对软件系统的及时认知[52,53].具体地,针对搜索系统,众包技术具备了完成可扩展、可靠且可重复的搜索系统评测活动的可行性.基于众包的评测活动可以重复开展并仍然保持可靠的结果.同时,众包评测的结果可以与专家判定结果相媲美,在多个不同的评估和相关性指标方面表现良好[54].

2.2.2 众包测试的自身优化

除了将众包技术应用于特定的测试类型或用于解决特定的测试问题,部分研究学者开始着眼于众包测试中特有的流程和机制问题,包括如何召集和管理测试工人、如何分解和设计测试任务、如何整合和处理测试报告等.

(1) 测试工人的召集与管理

大量优质众包测试工人的召集和有效管理是开展众包测试任务的前提.已有文献研究了在众包测试中,众包参与者的规模及其测试时间约束对于测试效果的影响[55].研究结果表明:有时间压力的个人与无时间压力的个人相比,可以获得更好的缺陷检测有效性.在软件测试任务中,众包测试人员的规模应根据人群生成的重复和无效报告的份额以及重复处理机制的有效性来确定.

Chen 等人开展了众包测试在教学项目中的实践研究,通过引进 3 项商业软件产品作为教学测试项目,让在校学生作为众包工人来完成测试任务.由于参与者是具有一定社会关系的学生,所以称其为“准众包测试”.文献探讨了学生作为众包测试工人的可行性,尝试了一种新型的众包工人召集与管理的方式,并通过单元测试、Web 测试、准众包测试这 3 种测试方式的对比,体现了准众包测试的优势[56].

(2) 测试任务的分解与设计

如何对复杂的测试任务进行分解和设计,是众包任务准备阶段的重要工作之一.在众包测试的已有相关研究中,研究人员将众包环境中的协作测试问题看作一个关于众包工人工作分配的 NP 完全问题,并将该问题转化为一个整数线性规划问题来解决[57].

在此基础上,基于划分的多任务匹配的协同测试方法则试图使用众包技术来解决大规模的协同测试.该方法由任务划分算法、基于贪心的任务匹配算法和众包测试结果集成这 3 个环节组成,动态地从测试用例中选择任务组,并将测试用例或任务组分配给适当的测试人员.该方法从质量、效率、可靠性和可扩展性这 4 个方面

进行了验证,并讨论了如何平衡测试人员的数量与测试结果的质量问题[58].

(3) 测试报告的整合与处理

在众包测试过程中,众包工人将提交大量质量参差不齐的测试报告,众包测试平台审核人员或任务请求者将面临着如何有效整合并处理众包测试报告的困难.

众包测试报告排序是一种有效提高报告处理效率的方法之一:一方面,可以综合运用多样性策略和风险策略动态选择测试报告进行检查[59];另一方面,鉴于许多移动应用的众包测试报告中包含有大量应用截图,通过使用空间金字塔(SPM)方法来比较包含屏幕截图的测试报告间的距离,从而结合测试报告中的文本和图片信息,对测试报告进行综合的优先级排序[60].

此外,如何实现大量测试报告的分类并得到真正含有缺陷的测试报告,也是报告整合与处理中需要解决的问题之一[61,62].研究人员提出了一种基于局部的主动学习方法,能够对众包测试报告进行分类.该方法解决了众包测试报告分类中存在的局部偏见以及缺少历史训练数据的问题[61].

(4) 其他

除了探讨众包测试具体环节的优化方法外,还有少数文献着重讨论了众包测试面临的机遇和挑战[63,64],主要包括如何管理众包工人、如何管理测试流程、如何管理相应的测试技术.目前的众包测试过程仍缺乏相应的标准,关于关键功能覆盖面的信息较为有限,仍存在重复的缺陷管理、不当的奖励机制等问题.因此,研究人员希望将众包测试与传统的实验室测试相结合,做到取长补短.

2.3 实验验证分析

随着众包测试技术的发展,研究人员也构建了一些众包测试的实验平台和工具,但尚未得到大规模的商业应用.在本文总结的 52 篇文献中,有 7 篇文献主要介绍了相应的平台和工具[65−71],具体包括:

(1) 针对 Web 应用的测试平台.如开源的用户测试工具(UTT)结合了自动化测试和众包测试,其客户端架构同样适合于众包测试,能够减少测试活动对带宽和服务器容量的需求[65].CrowdStudy 在客户端拥有一个用户活动跟踪组件,在服务器端拥有一个数据记录和众包组件,通过集成 Web 应用程序的 C/S 基础架构,CrowdStudy 可以通过单行代码嵌入到网站中[66];

(2) 针对移动应用的测试平台.如移动系统云测试框架 CTOMS 能够对 Android 系统应用程序的功能和 UI 进行全面的检测,很好地适应于移动应用的众包测试,并整合了移动开发中的多种测试技术[67].iTest 使用移动应用众包测试技术来完成对移动应用程序和 Web 服务的测试任务.iTest 能够帮助任务请求者在提交待测软件后轻松地获得测试结果[69].CrowdBlaze 系统首先使用静态分析和自动测试探索应用程序,然后招募众包工人为复杂案例提供输入,使得自动测试能够进一步探索应用程序.通过在自动测试和众包测试之间切换,该方法有效地提高了测试覆盖率[70].可扩展移动应用程序测试的云服务平台 Caiipa 采用上下文模糊方法来扩展移动应用运行的上下文空间,并利用了众包工人的输入数据以及众包测度(如多种网络条件、多个运营商网络和不同的地理位置等)[71];

(3) 综合众包测试服务平台.如 TestCloud 的众包业务模式能够利用互联网上的群体智能来实现高效的测试,为企业提供各种众包测试服务[68].

这些实验平台和工具有效地促进了众包测试技术的发展.在前文汇总的 43 篇研究型论文中,在相应实验环节也涉及了不少众包测试的平台和工具.因此,本节将总结 43 篇研究型论文以及 7 篇平台/工具型论文中所使用的测试对象、测试人员规模和测试平台等信息,为众包测试技术的实验验证环节的发展提供一些有益的参考.

由表 1 可知,众包测试的实验对象主要集中在多媒体、Web 网站和移动/PC 应用上.其中,多媒体内容是 QoE 测试的主要实验对象,Web 网站则主要用于可用性测试研究.此外,部分数学模型和开源程序也被用作众包测试的实验对象.

从实验验证环节中的测试人员规模来看,如表 2 所示:在明确提及测试人员规模的 33 篇文献中,有 16 篇文献仅采用了 1~100 个众包工人,12 篇文献采用了 101~500 人的测试规模,仅有 5 篇文献采用了超过 500 人的测试规模.由此可见,现有的众包测试人员规模主要集中在 500 人以下.通常来说,参与测试的众包工人越多,越有

利于获得可信的实验结果.然而在众包测试的实验过程中,测试规模的大小一方面取决于召集众包工人的困难程度,另一方面也和具体的测试任务类型相关.

在使用的实验平台方面,表 3 总结了部分应用较为广泛的实验平台情况.其中,Mturk、Microworkers 作为通用的众包商业平台,广泛地应用于众包测试的实验验证中.GitHub、Stack Overflow 等开源社区、Planetlab 开放平台[30]也被用于有效获取众包数据.国内的众包测试商业平台,如百度 MTC、慕测 MoocTest[60]等也开始引起研究学者的关注.此外,研究人员还提出了不少众包测试系统平台原型,例如 CrowdStudy[66]、CTOMS[67]、iTest[69]、CrowdBlaze[70]、Caiipa[71]等.

Table 1 The summary of testing subjects

表 1 测试对象信息汇总

测试对象	文献编号	计数
多媒体内容	[22−24,31,38]	5
Web 网站	[29,30,32−34,36,57,58]	8
移动/PC 应用	[35,39−41,44,52,59−62,66,67,69−71]	15
其他	[27,28,42,43,45−47,49−51,54,55]	12
未提及	[25,26,37,48,53,56,63−65,68]	10

Table 2 The participant scales

表 2 测试人员规模信息汇总

参与人数	文献编号	计数
1~100	[27,28,32−36,38,44,46,49,52,54,64,66,69]	16
101~500	[22,24,30,39,40,42,45,51,55,57,59,60]	12
>500	[23,26,29,31,41]	5
未提及	[25,37,43,47,48,50,53,56,58,61−63,65,67,68,70,71]	17

Table 3 The usage of testing platforms

表 3 测试平台部分信息汇总

平台	文献编号	计数
MTurk	[22,24,32−34,36,38−40,45,46,52,54,57,65]	15
Microworkers	[27−29,31]	4
Github/Stack overflow	[43,47,50,51]	4
Baidu MTC	[59,61,62]	3
SDI	[48,49]	2
TestCloud	[63,68]	2
未提及	[23,25,26,35,37,41,42,44,53,55,56,58,64]	13

通过上述对相关文献的分类总结及其实验验证环节的讨论,我们可以看到:尽管众包测试技术已经开始引起国内外研究学者的广泛关注,然而目前大多数研究仍停留在将众包技术应用于软件测试过程中,即,验证众包测试的可行性方面.针对众包这种新型工作方式所引发的软件测试中特定问题的相关研究还较少,仍然存在着众多研究领域有待深入探索.

3 众包测试平台分析

平台和工具始终是软件测试领域的关键部分,近年来,在众包测试领域出现了大量商业众包测试平台.我们使用关键词“crowdsourcing/crowdsourced testing+platform”及“众包/众包测试+平台”在 Google 中进行搜索,选择了前 5 页的平台作为调研对象.此外,我们还涵盖了一些学术研究人员在研究文献中所提及的平台.同时,还人工调研了这些众包测试平台,受篇幅限制,本节选取了 20 个主流众包测试平台**,从平台分类、测试领域、测试对

** Baidu MTC (test.baidu.com), Tencent Test (task.qq.com), Aliyun (mqc.aliyun.com), TestFlight (testflightapp.com), MoocTest (mooctest. net), Applause (applause.com), Crowd Testing (crowdsourced testing.com), uTest (utest.com), Testin (mtestin.com), 99tests (99tests.com), Pay4bugs (pay4bugs.com), MyCrowd (mycrowd.com), BugFinders (bugfinders.com), Testing Army (testingarmy.com), Passbrains (passbrains.com), TestBirds (testbirds.com), Testbats (testbats.com), WooYun (ce.wooyun.org), Bugcrowd (bugcrowd.com), Sobug (sobug.com).

象、工人召集方式、绩效考核方式等方面进行分析对比.

3.1 平台的分类

不同平台在众包测试过程中的参与程度不同,这将对众包工人的召集方式、绩效考核方式等产生直接的影响.因此,依据平台参与度的强弱不同,可将 20 个平台大致分为以下 3 类.

- 弱参与(weak-involved)平台,简称 W 平台(5 个:Baidu MTC,Tencent Test,Aliyun,TestFlight,MoocTest),参与度较弱.这类平台仅仅为任务请求方和众包工人提供技术平台,扮演市场的角色,较少参与众包测试任务的设计,通常不对众包测试任务的内容、形式和奖励机制等做出限制;
- 强参与(strong-involved)平台,简称 S 平台(12 个:Applause,uTest,Testin 等),参与度较强.这类平台不仅会参与众包测试任务的设计,控制测试任务的内容和形式,还对绩效考核方式、奖励方式等给出明确的规则;
- 强参与-安全测试(strong-involved and security testing)平台,简称 S-S 平台(3 个:WooYun,Bugcrowd,Sobug),是参与度较强的安全性测试平台.这些平台的技术专业性较强,重点关注安全性测试领域.在强参与平台的基础上,对于绩效考核方式及奖励方式有着更为严格的规定.显然,S-S 平台是一类特殊的 S 平台.

在调研的 20 个众测平台中,强参与的平台(含强参与-安全测试平台)占到了 75%.可见,大多数的商用众包测试平台还是深入地参与到众包测试过程中,为众包测试的顺利开展提供了监管和帮助.

3.2 测试领域

众包测试平台涉及的测试领域,即提供的测试服务类型主要包括:(a) 功能测试;(b) 可用性测试;(c) GUI 测试;(d) 本地化测试;(e) 性能测试;(f) 安全性测试;(g) 用户体验性测试.

平台涉及的测试领域与平台的类型有着紧密关联.W 平台几乎包含了所有的测试领域,这是因为 W 平台提供的测试任务通常由任务请求方独立完成测试任务的分解与设计,测试任务类型的自由度较高,即测试领域不受平台限制,因此,W 平台涵盖的测试范围也最广.相对而言,S 类平台的专注领域少于 W 类,这是因为此类平台将对测试任务的类型、范围等加以限制,通常由任务请求方提供测试的整体目标,而测试平台将完成对测试任务的分解、设计和分派.S-S 类平台只涉及安全性测试,显然,此类平台对众包测试人员的技术水平要求较高.

针对 20 个众测平台所涉及的测试领域进行频次统计可知:在所有测试类型中,功能测试、安全性测试和可用性测试出现次数较多,分别出现了 16/15/14 次.较为广泛地使用众包方式来完成这几类测试的原因在于:众包工人可以在使用软件的过程中较为自然地完成功能测试;众包所蕴含的群体智能可以帮助安全性测试的开展;众包带来的用户多样性有利于完成可用性测试.因此,大部分任务提供方和众包测试平台都认可利用众包工人去完成上述 3 种类型的测试.

3.3 测试对象

众包测试平台的测试对象主要包括:(a) 移动应用;(b) Web 系统;(c) 桌面系统;(d) 其他.大部分 W 和 S 平台涉及了所有种类的测试对象,S-S 平台则完整地涵盖了所有种类的测试对象,这说明安全性测试对于各类测试对象都尤为重要.

此外,所有的平台都将移动应用作为自己的测试对象,这一方面说明众多软件的开发都面向移动端,移动应用测试已经成为软件测试的关注重点;另一方面,也充分说明了使用众包方式完成移动应用测试的可行性.特别地,MoocTest 和 TestFlight 等众测平台将移动应用测试作为核心业务,研发了配套自动化工具友好地收集用户的设备概要、操作日志等信息,提供屏幕截图、测试报告上传等功能,进一步帮助众包工人以高效的方式完成移动应用的测试任务.众包方式带来的用户多样性、用户真实场景下真实体验的及时反馈,可以有效地帮助移动应用的开发团队缩短测试周期,并减少测试投入.

除了传统的 Web 端和桌面端的应用程序以外,一些刚刚涌现的智能技术,如智能硬件和可穿戴设备等,也被一些平台(如 TestBirds,WooYun)作为众包测试的对象.由此可见,众包测试可被广泛运用于最新一代的科技产品

中.不难想象,在将来的 AR 及 VR 技术的测试过程中,众包测试也能发挥重要的作用.

3.4 众包工人召集方式

如何召集到更多的众包工人来参与完成测试任务,是当前众包测试平台需要解决的关键问题之一.目前,众包工人召集方式(open call forms)主要包括:(a) 按需匹配(on-demand matching);(b) 在线竞标(online bidding);(c) 自由市场(free market);(d) 游戏/竞赛(game/competition)[11].

这些工人召集方式各有特点,其中,按需匹配方式将把测试任务的需求和众包工人的信息相匹配,例如性别、年龄、持有移动设备型号、经验、任务历史等.如果两者匹配,则对应的众包工人将收到测试任务的邀请.在线竞标方式的特点在于:当任务请求者发布测试任务后,众包工人可以通过竞价的方式争取这一测试任务.任务请求者可以根据众包工人给出的报价、个人信息、任务历史等情况,选择合适的众包工人来完成对应的测试任务.在自由市场的方式下,众测平台将为任务请求者和众包工人提供相互交流的平台,任务请求者将测试任务发布到众包平台上,众包工人可以收到推荐任务列表,也可以自行浏览任务列表并选择感兴趣的任务.游戏/竞赛方式则将完成众包测试任务的过程视为一场游戏/竞赛,仅有部分获胜的众包工人可以得到相应的奖励报酬.

在 20 个众测平台中,W 类平台都采用了自由市场方式,S 类平台都采用了按需匹配的方式,3 个 S-S 类平台则分别根据各自的运行流程采用了不同的众包工人召集方式.此外,在一些涉及复杂技术问题的测试平台中,众包工人只有通过完成基础任务和相应课程才能有机会获得奖励更高的测试任务.这种方式一方面确保了众包工人完成任务的最低成功率;另一方面,众包工人也能获得和自己能力相匹配的测试任务.

总体而言,按需匹配方式作为效率最高的召集方式被绝大多数平台所采用.少量平台采用了自由市场的方式,给予任务请求者和众包工人更高的自由度.在线竞标和游戏/竞赛这两种召集方式的使用目前相对较少.

3.5 绩效考核方式

绩效考核方式反映了任务请求者或众测平台将如何评价众包工人的任务完成情况[10,19].绩效考核方式主要包括:(a) 根据 bug 的检测情况进行考核,并区分 bug 的严重程度;(b) 根据 bug 的检测情况进行考核,但不区分 bug 的严重程度;(c) 根据任务量进行考核,并区分任务的难易程度;(d) 根据任务量进行考核,但不区分任务的难易程度.

平台采用的绩效考核方式与平台在众包测试过程中参与程度的强弱紧密相关.W 类平台均采用(d)方式,即,完成某一份任务的奖励是固定的.例如,TestFlight 是一款基于 Appstore 的众测平台,可以帮助开发者在软件正式上架之前募集众包工人参与测试,测试领域包括 GUI 测试、游戏体验测试等,每一份测试任务都会给众包员工奖励以相同的虚拟奖品.

S 类平台主要通过以下 3 种方式完成绩效考核.

(1) 选项(a)、选项(c),即,按照 bug 和任务的等级进行奖励.这些众测平台会在网站上明确公开自己的收费标准,并允许互联网上的任何潜在测试人员注册成为网站众包工人.此类平台包括 uTest、99tests 等.在 S 类的 12 个平台中,有超过一半的平台采用这种绩效考核方式;

(2) 选项(b)、选项(d),即,不区分 bug 和任务的等级.任务请求方要和平台沟通以确定具体的收费,平台的众包工人作为该平台的自有资源存在,新增测试人员需要通过考核后才能加入.此类平台包括 Applause、Passbrains 和 TestBirds;

(3) 选项(a)、选项(d),是一种混合模式.这类平台既可以按照 bug 的严重程度进行考核,同时也为任务请求者提供了简单的仅根据任务量来实现相同任务相同奖励的考核方式.此类平台包括 Testin 和 Testbats.以 Testin 为例,该平台为不同类型的测试任务设定了不同的考核机制,既提供企业服务,也支持个人测试任务的发布.

S-S 平台则全部采用以 bug 的严重程度来衡量绩效并支付薪酬的方式.Bug 的严重程度和发现难度越高,所支付的薪酬就越高.如 WooYun,每一个 bug 级别都有最低的奖金数,如果发现的 bug 尤为严重,甚至还会有基础奖励之上的额外奖励,直到达到封顶为止.相比固定奖励,这种方式更能调动众包工人寻求安全漏洞的积极性.

针对 20 个平台所涉及的绩效考核方式的选项进行频次统计之后可知:考虑测试特性从而根据 bug 的检测情况并区分 bug 严重程度的考核方式(选项(a),占比 39%)以及较为简单的仅根据完成任务量进行考核的方式(选项(d),占比 32%)都占据了较大的比重.由此可见,众测平台的参与度、服务对象和测试领域都将对绩效考核方式产生影响.

4 未来研究方向

众包软件测试技术作为一个新型的测试技术,尽管学术界和工业界已经开展了不少研究工作,但多数工作集中在利用众包的方式改进现有的测试方法或测试流程,针对众包测试所引发的新问题、众包测试的新特性、众包测试自身流程和机制的研究还较初步.本节立足于现有众包测试技术的发展情况,对未来的研究方向作一些展望.

4.1 测试工具

工具在软件工程实践中具有重大的意义,对于众包软件工程这样一个以实践为驱动的研究方向,工具显得更加重要.在众包软件工程领域,研究者们已经提出了一些工具和策略[86−88],用以辅助工程实践.然而在众包软件测试领域,当前针对辅助工具或技术的研究还较为缺乏.我们认为:通过工具和平台的开发,将有效降低众包测试的技术门槛,提高众包测试的整体效率.众包测试工具的开发可以针对以下几个主要问题来展开.

(1) 自动化测试数据获取工具

在众包测试过程中,专业的测试人员通常希望获得众包测试工人在测试过程中的实际行为,包括输入的测试用例、测试执行剖面、执行日志等.同时,获取这些信息的过程不能对待测对象的行为产生太大的影响.因此,轻量级的测试数据自动获取工具将是未来的一个研究重点.

(2) 协同测试工具的构建

如何帮助众包工人以协同的方式来完成测试任务,是提高众包测试效率的一个重要方向.然而,协同测试的实施过程需要依赖于相应的工具或者平台.因此,协同测试的工具或平台的构建很有必要,将从根本上提高众包测试过程中各个阶段的效率.

(3) 多样性测试环境模拟工具

由于当前操作系统种类、版本较多,同时,各类基础框架也越来越复杂,为了确保操作环境的统一,一些已有研究在其实验中应用了虚拟机技术[38,39].因此,需要研究如何通过模拟以获得各种版本和各种配置的软硬件环境,这对于软件兼容性的测试具有重要意义.

4.2 众包测试工人的召集与管理

如何召集更多的众包工人参与众包任务,一直是众包领域的热点研究课题之一,也是众包技术在实际应用中能否成功的关键因素.已有大量的研究工作关注于如何实现任务请求者和众包工人的双赢,从而最大限度地提高任务完成效率和众包工人的利益[89−93].关于如何做好众包测试工人的召集与管理,我们认为,可以围绕以下几个问题展开.

(1) 提供更好的技能培训机制

众包测试的优势之一在于众包工人的人数众多,然而部分工人在测试专业技能和业务领域知识方面可能还存在着欠缺.因此,任务请求者和众包测试平台应合力建立良好的培训机制,构建相关平台并提供对应资源,这对于众包测试生态环境的维护具有非常积极的意义.

(2) 设计合理的考核与奖励机制

合理的考核与奖励机制直接决定了众包任务对众包工人的吸引力,能够更好地激励众包工人高质量地完成任务.在众包测试领域,当前主要采用的考核机制是根据 bug 的严重程度,然而实际运营中,根据 bug 的严重程度来考核的方式并不能适用于所有的情况.例如,在可用性测试中,界定可用性缺陷的错误等级较为困难,在这种情况下,可以考虑采用动态定价的激励机制来引导众包工人的行为[94].

4.3 测试任务设计和分配

众包软件测试任务的设计和分配,是众包软件测试中的一个核心问题.我们注意到:已有相关工作把测试任务游戏化(gamify)[25,95,96].同时,有部分研究者开始关注如何高效、经济地分配测试任务[57,58,97].我们认为,这一方向的未来工作可以围绕以下几个方面展开.

(1) 提高测试任务的吸引力

测试任务设计的好坏,直接影响了众包测试任务的吸引力.特别是在测试经费预算有限的情况下,无法简单采用加大任务回报的方式来吸引更多众包工人,此时,提高测试任务的趣味性成为一种可行的方式.例如,一种提高众包任务趣味性的方式是将众包任务游戏化或比赛化.由此可见:如何设计测试任务、进一步提高众包任务的趣味性,是一个非常有意思的研究课题.

(2) 协同化的任务分配和完成机制

当前,众包任务的分配和完成主要采用独立任务“发布-执行-完成”的模式,即:任务之间关联性较弱,同时,执行任务的众包工人之间的关联也相对较弱.我们认为,后续的研究可以围绕如何建立协同化的任务分配和完成机制展开.一方面,在众包任务的设计过程中,将任务之间的关联关系纳入考虑范畴,将传统的“独立任务”模式转化为“任务链”模式;另一方面,在任务的分配和完成过程中,不再局限于众包工人个体完成,可以基于众包工人之间的关联关系,以众包团队的形式来完成任务,从而发挥每位众包工人的技能特点和优势.

4.4 测试报告处理

测试报告处理包括重复 bug 报告检测[98−100]、bug 分配[101−106]、bug 报告摘要[107,108]等,是软件测试中的重要问题.针对传统测试报告,研究人员已提出了多种解决方法并研发了配套工具[109−111],然而,众包软件测试的特性使得测试报告的处理面临新的挑战.相对于传统测试报告以文字描述为主,众包测试报告完成过程中会产生多种类型的信息,例如文字、图片、视频、日志等,但目前仅有部分原始信息反映在测试报告中,且信息的呈现方式较为单一,给大量众包测试报告的快速审查带来了困难.同时,由于众包软件测试力求通过群体智能来弥补当前自动化测试技术的不足,众包测试过程中必将产生大量的测试报告.因此,如何高效地处理大量的测试报告变得尤为关键,测试报告的多元化信息,也为软件缺陷的复现提供了更多可能.

基于当前众包软件测试报告处理技术的研究进展[59−62]和传统测试报告处理技术的分析总结[84],我们认为,未来的研究主要将围绕以下几个要点展开.

(1) 众包测试报告的设计

测试报告模板的质量直接关系到众包工人反馈的质量和审核人员对测试报告的处理效率.相对于传统的测试报告,众包测试报告应较为简炼且能够精确地反映问题.这主要是因为:相当一部分众包测试工人不具备软件工程的专业知识,他们期望在较短时间内完成测试报告的填写,而且测试报告的填写应该可以在各类平台上完成.因此,需要开展相关调研以明确针对不同测试任务最有效的测试报告字段,从而完成针对不同测试任务的测试报告模板的设计.

(2) 众包测试报告的合并

已有针对传统测试报告的研究表明,重复的测试报告对于软件缺陷的定位和修复具有积极意义[98−100].在众包测试过程中,众多众包工人将提交大量的测试报告,其中必将包含较多的重复数据.因此,对于描述了类似或者相同错误的测试报告信息进行合并,将为后续的缺陷定位和修复提供更多有益的信息.该方向主要的研究内容包括:如何自动化地识别相似的测试报告、如何有效地管理描述相似问题的测试报告、如何提取并呈现反映软件缺陷的关键信息.

(3) 众包测试报告的内容摘要

针对信息量较大的测试报告进行内容摘要,是一个必要的工作.内容摘要可以帮助报告审核人员关注测试报告中的关键信息.已有研究学者针对传统的测试报告,给出了针对文本信息的自动摘要技术[107,108].然而与传统的测试报告不同,众包测试报告含有的信息类型更为丰富,可能包含文本、图像、视频、音频等.因此,需要研

究针对多类型信息的内容摘要技术.例如:可将关键帧提取技术应用到测试过程截屏录像中,提取本次测试过程中最具有代表性的帧或错误发现帧;从图片中提取关键错误信息块的技术,则可以帮助审核人员快速地从大量截图信息中确认最有效的错误截图.

4.5 测试行为的自动化回放

测试行为的自动回放对程序员定位错误和修订错误具有重大的意义.在众包测试领域,目前使用最为广泛的回放方法包括:要求众包工人使用录屏或者拍摄工具对操作过程进行全程录制,然后提交所录制的数据;通过客户端记录工具记录众包工人的操作行为并生成相应的脚本,然后将脚本提交至服务器端.然而,上述两种方式都在不同程度上降低了众包任务的执行效率.同时,由于软硬件环境存在着差异,这两种方式都可能会遗漏实际执行过程中的关键信息.研发自动记录后台系统日志的工具,将有效解决测试行为自动回放的难题[112].一方面,日志信息的记录相对于现有的两种主流录制方式,可以获得更加丰富的数据信息;另一方面,日志信息的记录过程更加轻量,对待测对象的行为和执行效率的影响更小.尽管日志记录工具的研究已较为成熟,但是如何从日志中实现测试行为的自动回放,这方面的研究还处于较为初步的阶段.

5 总 结

本文从学术研究现状和工业研究进展两个方面,对众包软件测试技术做出了比较全面的概述.我们收集并最终汇总了众包软件测试领域的 52 篇文献,并简要描述了这些文献的技术、策略和实验验证情况.此外,我们分析对比了当前应用最广泛的 20 个众包测试平台,并从多个角度讨论了众包测试平台的特性.在此基础上,详细讨论了众包软件测试技术的未来研究方向.众包软件测试技术有着良好的应用前景,但仍然存在着大量的研究问题尚待解决,它必将成为软件工程领域的一个新兴研究热点.

References:

[1] Raymond E. The cathedral and the bazaar. Knowledge, Technology & Policy, 1999,12(3):23−49. [doi: 10.1007/s12130-999-1026-0]

[2] Howe J. The rise of crowdsourcing. Wired Magazine, 2006,14(6):1−4.

[3] Feng JH, Li GL, Feng JH. A survey on crowdsourcing. Chinese Journal of Computers, 2015,38(9):1713−1725 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2015.01713]

[4] Heer J, Bostock M. Crowdsourcing graphical perception: Using mechanical Turk to assess visualization design. In: Proc. of the SIGCHI Conf. on Human Factors in Computing Systems. Atlanta, 2010. 203−212. [doi: 10.1145/1753326.1753357]

[5] Franklin MJ, Kossmann D, Kraska T, Ramesh S, Xin R. CrowdDB: Answering queries with crowdsourcing. In: Proc. of the ACM SIGMOD Int'l Conf. on Management of Data. 2011. 61−72. [doi: 10.1145/1989323.1989331]

[6] Negri M, Bentivogli L, Mehdad Y, Giampiccolo D, Marchetti A. Divide and conquer: Crowdsourcing the creation of cross-lingual textual entailment corpora. In: Proc. of the Conf. on Empirical Methods in Natural Language Processing. Edinburgh, 2011. 670−679. http://conferences.inf.ed.ac.uk/emnlp2011/

[7] Yan Y, Rosales R, Fung G, Dy JG. Active learning from crowds. In: Proc. of the 28th Int'l Conf. on Machine Learning. Bellevue, 2011. 1161−1168. http://www.icml-2011.org/

[8] Zuccon G, Leelanupab T, Whiting S, Yilmaz E, Jose JM, Azzopardi L. Crowdsourcing interactions: Using crowdsourcing for evaluating interactive information retrieval systems. Information Retrieval Journal, 2013,16(2):267−305. [doi: 10.1007/s10791-012-9206-z]

[9] Cinalli D, Marti L, Sanchezpi N, Gracia ACB. Using collective intelligence to support multi-objective decisions: Collaborative and online preferences. In: Proc. of the IEEE/ACM Int'l Conf. on Automated Software Engineering Workshops. 2015. 82−85. [doi: 10.1109/ASEW.2015.12]

[10] Wu W, Tsai WT, Li W. An evaluation framework for software crowdsourcing. Frontiers of Computer Science, 2013,7(5):694−709. [doi: 10.1007/s11704-013-2320-2]

[11] Mao K, Capra L, Harman M, Jia Y. A survey of the use of crowdsourcing in software engineering. Journal of Systems and Software, 2017,126:57−84. [doi: 10.1016/j.jss.2016.09.015]

[12] Stolee KT, Elbaum S. Exploring the use of crowdsourcing to support empirical studies in software engineering. In: Proc. of the ACM/IEEE Int'l Symp. on Empirical Software Engineering and Measurement. Bolzano-Bozen: ACM Press, 2010. 1–4. [doi: 10.1145/1852786.1852832]

[13] Bari E, Johnston M, Tsai W, Wu W. Software crowdsourcing practices and research directions. In: Proc. of the IEEE Symp. on Service-Oriented System Engineering. 2016. 372–379. [doi: 10.1109/SOSE.2016.69]

[14] Latoza T, Hoek A. Crowdsourcing in software engineering: Models, motivations, and challenges. IEEE Software, 2016,33(1):74–80. [doi: 10.1109/MS.2016.12]

[15] Zhao Y, Zhu Q. Evaluation on crowdsourcing research: Current status and future direction. Information Systems Frontiers, 2014, 16(3):417–434. [doi: 10.1007/s10796-012-9350-4]

[16] Yuen M, King I, Leung K. A survey of crowdsourcing systems. In: Proc. of the 3rd IEEE Int'l Conf. on Privacy, Security, Risk and Trust, and IEEE Int'l Conf. on Social Computing. Boston, 2011. 766–773. [doi: 10.1109/PASSAT/SocialCom.2011.203]

[17] Kittur A, Nickerson JV, Bernstein MS, Gerber EM, Shaw A, Zimmerman J, Lease M, Horton JJ. The future of crowd work. In: Proc. of the 2013 ACM Conf. on Computer Supported Cooperative Work. San Antonio, 2013. 1301–1318. [doi: 10.1145/2441776.2441923]

[18] Doan A, Ramakrishnan R, Halevy AY. Crowdsourcing systems on the World-Wide Web. Communications of the ACM, 2011,54(4): 86–96. [doi: 10.1145/1924421.1924442]

[19] Chittilappilly AI, Chen L, Amer-Yahia S. A survey of general-purpose crowdsourcing techniques. IEEE Trans. on Knowledge and Data Engineering, 2016,28(9):2246–2266. [doi: 10.1109/TKDE.2016.2555805]

[20] Tong YX, Yuan Y, Cheng YR, Chen L, Wang GR. A survey of spatiotemporal crowdsourced data management techniques. Ruan Jian Xue Bao/Journal of Software, 2017,28(1):35–58 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5140.htm [doi: 10.13328/j.cnki.jos.005140]

[21] Estellésarolas E. Towards an integrated crowdsourcing definition. Journal of Information Science, 2012,38(2):189–200. [doi: 10.1177/0165551512437638]

[22] Chen KT, Wu CC, Chang YC, Lei CL. A crowdsourceable QoE evaluation framework for multimedia content. In: Proc. of the 17th ACM Int'l Conf. on Multimedia. ACM Press, 2009. 491–500. [doi: 10.1145/1631272.1631339]

[23] Chen KT, Chang CJ, Wu CC, Chang YC, Lei CL. Quadrant of euphoria: A crowdsourcing platform for QoE assessment. IEEE Network the Magazine of Global Internet Working, 2010,24(2):28–35. [doi: 10.1109/MNET.2010.5430141]

[24] Wu CC, Chen KT, Chang YC, Lei CL. Crowdsourcing multimedia QoE evaluation: A trusted framework. IEEE Trans. on Multimedia, 2013,15(5):1121–1137. [doi: 10.1109/TMM.2013.2241043]

[25] Hossfeld T, Keimel C, Timmerer C. Crowdsourcing quality-of-experience assessments. Computer, 2014,47(9):98–102. [doi: 10.1109/MC.2014.245]

[26] Hossfeld T, Keimel C, Hirth M, Gardlo B, Habigt J, Diepold K, TranGia P. Best practices for QoE crowdtesting: QoE assessment with crowdsourcing. IEEE Trans. on Multimedia, 2014,16(2):541–558. [doi: 10.1109/TMM.2013.2291663]

[27] Gardlo B, Ries M, Hoßfeld T, Schatz R. Microworkers vs. facebook: The impact of crowdsourcing platform choice on experimental results. In: Proc. of the 4th IEEE Int'l Workshop on Quality of Multimedia Experience. 2012. 35–36. [doi: 10.1109/QoMEX.2012.6263885]

[28] Gardlo B. Quality of experience evaluation methodology via crowdsourcing [Ph.D. Thesis]. Slovakia: University of Zilina, 2012.

[29] Gardlo B, Egger S, Seufert M, Schatz R. Crowdsourcing 2.0: Enhancing execution speed and reliability of Web-based QoE testing. In: Proc. of the IEEE Int'l Conf. on Communications. 2014. 1070–1075. [doi: 10.1109/ICC.2014.6883463]

[30] Sun H, Zhang W, Yan M, Liu X. Recommending Web services using crowdsourced testing data. In: Crowdsourcing. Berlin, Heidelberg: Springer-Verlag, 2015. 219–241. [doi: 10.1007/978-3-662-47011-4_12]

[31] Seufert M, Zach O, Hoßfeld T, Slanina M, TranGia P. Impact of test condition selection in adaptive crowdsourcing studies on subjective quality. In: Proc. of the IEEE 8th Int'l Conf. on Quality of Multimedia Experience. 2016. 1–6. [doi: 10.1109/QoMEX.2016.7498939]

[32] Liu D, Bias RG, Lease M, Kuipers R. Crowdsourcing for usability testing. American Society for Information Science and Technology, 2012,49(1):1–10. [doi: 10.1002/meet.14504901100]

[33] Meier F, Bazo A, Burghardt M, Wolff C. Evaluating a Web-based tool for crowdsourced navigation stress tests. In: Proc. of the Int'l Conf. of Design, User Experience, and Usability. Berlin, Heidelberg: Springer-Verlag, 2013. 248–256. [doi: 10.1007/978-3-642-39253-5_27]

[34] Schneider C, Cheung T. The power of the crowd: Performing usability testing using an on-demand workforce. In: Proc. of the Information Systems Development. New York: Springer-Verlag, 2013. 551−560. [doi: 10.1007/978-1-4614-4951-5_44]

[35] Gomide VHM, Valle PA, Ferreira JO, Barbosa JR G, da Rocha AF, de Barbosa TMGA. Affective crowdsourcing applied to usability testing. Int'l Journal of Computer Scienceand Information Technologies, 2014,5(1):575−579.

[36] Nebeling M, Speicher M, Grossniklaus M, Norrie MC. Crowdsourced Web site evaluation with crowdstudy. In: Proc. of the Int'l Conf. on Web Engineering. Berlin, Heidelberg: Springer-Verlag, 2012. 494−497. [doi: 10.1007/978-3-642-31753-8_52]

[37] Khan AI, Al-khanjari Z, Sarrab M. Crowd sourced testing through end users for mobile learning application in the context of bring your own device. In: Proc. of the IEEE 7th Annual Conf. on Information Technology, Electronics and Mobile Communication. 2016. 1−6. [doi: 10.1109/IEMCON.2016.7746256]

[38] Vliegendhart R, Dolstra E, Pouwelse J. Crowdsourced user interface testing for multimedia applications. In: Proc. of the ACM Multimedia 2012 Workshop on Crowdsourcing for Multimedia. ACM Press, 2012. 21−22. [doi: 10.1145/2390803.2390813]

[39] Dolstra E, Vliegendhart R, Pouwelse J. Crowdsourcing GUI tests. In: Proc. of the 6th IEEE Int'l Conf. on Software Testing, Verification and Validation. 2013. 332−341. [doi: 10.1109/ICST.2013.44]

[40] Komarov S, Reinecke K, Gajos KZ. Crowdsourcing performance evaluations of user interfaces. In: Proc. of the SIGCHI Conf. on Human Factors in Computing Systems. ACM Press, 2013. 207−216. [doi: 10.1145/2470654.2470684]

[41] Musson R, Richards J, Fisher D, Bird C, Bussone B, Ganguly S. Leveraging the crowd: How 48000 users helped improve LYNC performance. IEEE Software, 2013,30(4):38−45. [doi: 10.1109/MS.2013.67]

[42] Chen N, Kim S. Puzzle-Based automatic testing: Bringing humans into the loop by solving puzzles. In: Proc. of the 27th IEEE/ACM Int'l Conf. on Automated Software Engineering. ACM Press, 2012. 140−149. [doi: 10.1145/2351676.2351697]

[43] Pham R, Singer L, Schneider K. Building test suites in social coding sites by leveraging drive-by commits. In: Proc. of the 2013 Int'l Conf. on Software Engineering. IEEE Press, 2013. 1209−1212. [doi: 10.1109/ICSE.2013.6606680]

[44] Gómez M, Rouvoy R, Adams B, Seinturier L. Reproducing context-sensitive crashes of mobile apps using crowdsourced monitoring. In: Proc. of the ACM Int'l Workshop on Mobile Software Engineering and Systems. 2016. 88−99. [doi: 10.1109/MobileSoft.2016.033]

[45] Pastore F, Mariani L, Fraser G. Crowdoracles: Can the crowd solve the oracle problem? In: Proc. of the 6th IEEE Int'l Conf. on Software Testing, Verification and Validation. 2013. 342−351. [doi: 10.1109/ICST.2013.13]

[46] Bachrach Y, Minka T, Guiver J, Graepel T. How to grade a test without knowing the answers—A Bayesian graphical model for adaptive crowdsourcing and aptitude testing. In: Proc. of the 29th Int'l Conf. on Machine Learning. Edinburgh, 2012. 1183−1190. https://icml.cc/2012/

[47] Chen F, Kim S. Crowd debugging. In: Proc. of the ACM Joint Meeting on Foundations of Software Engineering. 2015. 320−332. [doi: 10.1145/2786805.2786819]

[48] Petrillo F, Lacerda G, Pimenta M, Freitas C. Visualizing interactive and shared debugging sessions. In: Proc. of the IEEE Working Conf. on Software Visualization. Bremen, 2015. 140−144. [doi: 10.1109/VISSOFT.2015.7332425]

[49] Petrillo F, Soh Z, Khomh F, Pimenta M, Freitas C, Guéhéneuc YG. Towards understanding interactive debugging. In: Proc. of the IEEE Int'l Conf. on Software Quality, Reliability and Security. 2016. 152−163. [doi: 10.1109/QRS.2016.27]

[50] Badashian AS, Hindle A, Stroulia E. Crowdsourced bug triaging. In: Proc. of the IEEE Int'l Conf. on Software Maintenance and Evolution. 2015. 506−510. [doi: 10.1109/ICSM.2015.7332503]

[51] Badashian AS, Hindle A, Stroulia E. Crowdsourced bug triaging: Leveraging Q&A platforms for bug assignment. In: Proc. of the Int'l Conf. on Fundamental Approaches to Software Engineering. Berlin, Heidelberg: Springer-Verlag, 2016. 231−248. [doi: 10.1007/978-3-662-49665-7_14]

[52] Sherief N, Jiang N, Hosseini M, Phalp K, Ali R. Crowdsourcing software evaluation. In: Proc. of the 18th ACM Int'l Conf. on Evaluation and Assessment in Software Engineering. 2014. 19. [doi: 10.1145/2601248.2601300]

[53] Sherief N. Software evaluation via users' feedback at runtime. In: Proc. of the 18th Int'l Conf. on Evaluation and Assessment in Software Engineering. 2014. 1−4. http://ease2014.org/

[54] Blanco R, Halpin H, Herzig DM, Mika P, Pound J, Thompson HS. Repeatable and reliable search system evaluation using crowdsourcing. In: Proc. of the 34th Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval. 2011. 923−932. [doi: 10.1145/2009916.2010039]

[55] Mäntylä MV, Itkonen J. More testers—The effect of crowd size and time restriction in software testing. Information and Software Technology, 2013,55(6):986−1003. [doi: 10.1016/j.infsof.2012.12.004]

[56] Chen Z, Luo B. Quasi-Crowdsourcing testing for educational projects. In: Proc. of the 36th ACM Int'l Conf. on Software Engineering. 2014. 272−275. [doi: 10.1145/2591062.2591153]

[57] Tung YH, Tseng SS. A novel approach to collaborative testing in a crowdsourcing environment. Journal of Systems and Software, 2013,86(8):2143−2153. [doi: 10.1016/j.jss.2013.03.079]

[58] Guo S, Chen R, Li H. A real-time collaborative testing approach for Web application: Via multi-tasks matching. In: Proc. of the IEEE Int'l Conf. on Software Quality, Reliability and Security Companion. 2016. 61−68. [doi: 10.1109/QRS-C.2016.13]

[59] Feng Y, Chen Z, Jones JA, Fang C, Xu B. Test report prioritization to assist crowdsourced testing. In: Proc. of the 10th ACM Joint Meeting on Foundations of Software Engineering. 2015. 225−236. [doi: 10.1145/2786805.2786862]

[60] Feng Y, Jones JA, Chen Z, Fang C. Multi-Objective test report prioritization using image understanding. In: Proc. of the IEEE/ACM Int'l Conf. on Automated Software Engineering. 2016. 202−213. [doi: 10.1145/2970276.2970367]

[61] Wang J, Wang S, Cui Q, Wang Q, Li M, Zhai J. Local-Based active classification of test report to assist crowdsourced testing. In: Proc. of the IEEE/ACM Int'l Conf. on Automated Software Engineering. 2016. 190−201. [doi: 10.1145/2970276.2970300]

[62] Wang J, Cui Q, Wang Q, Wang S. Towards effectively test report classification to assist crowdsourced testing. In: Proc. of the ACM/IEEE Int'l Symp. on Empirical Software Engineering and Measurement. 2016. 6−16. [doi: 10.1145/2961111.2962584]

[63] Zogaj S, Bretschneider U, Leimeister JM. Managing crowdsourced software testing: A case study based insight on the challenges of a crowdsourcing intermediary. Journal of Business Economics, 2014,84(3):375−405. [doi: 10.1007/s11573-014-0721-9]

[64] Guaiani F, Muccini H. Crowd and laboratory testing, can they co-exist? An exploratory study. In: Proc. of the 2nd IEEE/ACM Int'l Workshop on CrowdSourcing in Software Engineering. 2015. 32−37. [doi: 10.1109/CSI-SE.2015.14]

[65] Teinum A. User testing tool: Towards a tool for crowdsource-enabled accessibility evaluation of Web sites [MS. Thesis]. Agder: University of Agder, 2013.

[66] Nebeling M, Speicher M, Norrie MC. CrowdStudy: General toolkit for crowdsourced evaluation of Web interfaces. In: Proc. of the ACM SIGCHI Symp. on Engineering Interactive Computing Systems. 2013. 255−264. [doi: 10.1145/2480296.2480303]

[67] Starov O. Cloud platform for research crowdsourcing in mobile testing [MS. Thesis]. East Carolina University, 2013.

[68] Zogaj S, Bretschneider U. Crowdtesting with testcloud—Managing the challenges of an intermediary in a crowdsourcing business model. In: Proc. of the European Conf. on Information Systems. 2013. 143−157. [doi: 10.2139/ssrn.2475415]

[69] Yan M, Sun H, Liu X. iTest: Testing software with mobile crowdsourcing. In: Proc. of the 1st Int'l Workshop on Crowd-Based Software Development Methods and Technologies. 2014. 19−24. [doi: 10.1145/2666539.2666569]

[70] Xue H. Using redundancy to improve security and testing [Ph.D. Thesis]. University of Illinois at Urbana-Champaign, 2013.

[71] Liang CJM, Lane ND, Brouwers N, Zhang L, Karlsson BF, Liu H, Liu Y, Tang J, Shan X, Chandra R, Zhao F. Caiipa: Automated large-scale mobile app testing through contextual fuzzing. In: Proc. of the 20th Annual Int'l Conf. on Mobile Computing and Networking. 2014. 519−530. [doi: 10.1145/2639108.2639131]

[72] Rao P, Dubey A, Virdi G. Crowdsourced testing for enterprises: Experiences. In: Proc. of the Workshop on Alternate Workforces for Software Engineering. 2015. 56−57. http://ceur-ws.org/Vol-1519/

[73] Sharma M, Padmanaban R. Leveraging the Wisdom of the Crowd in Software Testing. Boca Raton: CRC Press, 2014.

[74] Memon A, Banerjee I, Nagarajan A. GUI ripping: Reverse engineering of graphical user interfaces for testing. In: Proc. of the 10th Working Conf. on Reverse Engineering. 2003. 260−269. [doi: 10.1109/WCRE.2003.1287256]

[75] Microsoft LYNC. http://office:microsoft:com/lync

[76] Chrome telemetry. http://www:chromium:org/developers/telemetry

[77] Firefox telemetry. https://telemetry:mozilla:org

[78] Sen K, Agha G. CUTE and jCUTE: Concolic unit testing and explicit path model-checking tools. In: Ball T, Jones R, eds. Proc. of the Computer Aided Verification. LNCS 4144, 2006. 419−423. [doi: 10.1007/11817963_38]

[79] Pacheco C, Lahiri SK, Ernst MD, Ball T. Feedback-Directed random test generation. In: Proc. of the 29th Int'l Conf. on Software Engineering. 2007. 75−84. [doi: 10.1109/ICSE.2007.37]

[80] Tillmann N. De Halleux J. Pex-White box test generation for .NET. In: Beckert B, Hhnle R, eds. Proc. of the 2nd Int'l Conf. on Tests and Proofs. LNCS 4966, 2008. 134−153. [doi: 10.1007/978-3-540-79124-9_10]

[81] Barr ET, Harman M, McMinn P, Shahbaz M, Yoo S. The oracle problem in software testing: A survey. IEEE Trans. on Software Engineering, 2015,41(5):507−525. [doi: 10.1109/TSE.2014.2372785]

[82] Allamanis M, Sutton C. Mining idioms from source code. In: Proc. of the ACM Sigsoft Int'l Symp. on Foundations of Software Engineering. 2014. 472−483. [doi: 10.1145/2635868.2635901]

[83] Lawrance J, Bogart C, Burnett M, Bellamy R, Rector K, Fleming SD. How programmers debug, revisited: An information foraging theory perspective. IEEE Trans. on Software Engineering, 2013,39(2):197−215. [doi: 10.1109/TSE.2010.111]

[84] Zhang J, Wang X, Hao D, Bing X, Lu Z, Hong M. A survey on bug-report analysis. Science China Information Sciences, 2015, 58(2):1−24. [doi: 10.1007/s11432-014-5241-2]

[85] Xia X, Wang XY, Yang XH, Lo D. Bug-Report management and ananlysis of open-sourced software sysytms. Communications of the CCF, 2016,2:29−34 (in Chinese with English abstract).

[86] Bruch M. Ide 2.0: Leveraging the wisdom of the software engineering crowds [Ph.D. Thesis]. Technische Universität Darmstadt, 2012.

[87] Ponzanelli L. Exploiting crowd knowledge in the ide [MS. Thesis]. Universita Della Svizzera Italiana, 2012.

[88] Zagalsky A, Barzilay O, Yehudai A. Example overflow: Using social media for code recommendation. In: Proc. of the 3rd IEEE Int'l Workshop on Recommendation Systems for Software Engineering. 2012. 38−42. [doi: 10.1109/RSSE.2012.6233407]

[89] Kittur A, Chi EH, Suh B. Crowdsourcing user studies with mechanical Turk. In: Proc. of the ACM SIGCHI Conf. on Human Factors in Computing Systems. 2008. 453−456. [doi: 10.1145/1357054.1357127]

[90] Zhang ZQ, Pang JS, Xie XQ, Zhou Y. Research on crowdsoucing quality control strategies and evaluation algorithm. Chinese Journal of Computers, 2013,36(8):1636−1649 (in Chinese with English abstract).

[91] Singla A, Krause A. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In: Proc. of the 22nd ACM Int'l Conf. on World Wide Web. 2013. 1167−1178. [doi: 10.1145/2488388.2488490]

[92] Zhao D, Li XY, Ma H. How to crowdsource tasks truthfully without sacrificing utility: Online incentive mechanisms with budget constraint. In: Proc. of the IEEE INFOCOM 2014—IEEE Conf. on Computer Communications. 2014. 1213−1221. [doi: 10.1109/INFOCOM.2014.6848053]

[93] Wu Y, Zeng JR, Peng H, Chen H, Li CP. Survey on incentive mechanisms for crowd sensing. Ruan Jian Xue Bao/Journal of Software, 2016,27(8):2025−2047 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5049.htm [doi: 10.13328/j.cnki.jos.005049]

[94] Singer Y, Mittal M. Pricing mechanisms for crowdsourcing markets. In: Proc. of the 22nd ACM Int'l Conf. on World Wide Web. 2013. 157−1166. [doi: 10.1145/2488388.2488489]

[95] Morschheuser B, Hamari J, Koivisto J. Gamification in crowdsourcing: A review. In: Proc. of the IEEE Hawaii Int'l Conf. on System Sciences. 2016. 4375−4384. [doi: 10.1109/HICSS.2016.543]

[96] Hu Z, Wu W. A game theoretic model of software crowdsourcing. In: Proc. of the IEEE Int'l Symp. on Service Oriented System Engineering. 2014. 446−453. [doi: 10.1109/SOSE.2014.79]

[97] Xie T. Cooperative testing and analysis: Human-Tool, tool-tool, and human-human cooperations to get work done. In: Proc. of the 12th IEEE Int'l Working Conf. on Source Code Analysis and Manipulation (Keynote). 2012. [doi: 10.1109/SCAM.2012.31]

[98] Wang X, Zhang L, Xie T, Anvik J, Sun J. An approach to detecting duplicate bug reports using natural language and execution information. In: Proc. of the Int'l Conf. on Software Engineering. Leipzig, 2008. 461−470. [doi: 10.1145/1368088.1368151]

[99] Sun C, Lo D, Wang X, Jiang J, Khoo S. A discriminative model approach for accurate duplicate bug report retrieval. In: Proc. of the ACM/IEEE Int'l Conf. on Software Engineering. 2010. 45−54. [doi: 10.1145/1806799.1806811]

[100] Bettenburg N, Premraj R, Zimmermann T. Duplicate bug reports considered harmful … really? In: Proc. of the IEEE Int'l Conf. on Software Maintenance. 2008. 337−345. [doi: 10.1109/ICSM.2008.4658082]

[101] Xuan J, Jiang H, Ren Z, Yan J, Luo Z. Automatic bug triage using semi-supervised text classification. In: Proc. of the 22nd Int'l Conf. on Software Engineering and Knowledge Engineering. 2010. 209−214. http://www.ksi.edu/seke/seke10.html

[102] Xuan J, Jiang H, Ren Z, Zou W. Developer prioritization in bug repositories. In: Proc. of the 34th IEEE Int'l Conf. on Software Engineering. 2012. 25−35. [doi: 10.1109/ICSE.2012.6227209]

[103] Hu H, Zhang H, Xuan J, Sun W. Effective bug triage based on historical bug-fix information. In: Proc. of the IEEE Int'l Symp. on Software Reliability Engineering. Naples, 2014. 122−132. [doi: 10.1109/ISSRE.2014.17]

[104] Xia X, Lo D, Wang X, Zhou B. Dual analysis for recommending developers to resolve bugs. Journal of Software Evolution & Process, 2015,27(3):195−220. [doi: 10.1002/smr.1706]

[105] Yang X, Lo D, Xia X, Bao L, Sun J. Combining word embedding with information retrieval to recommend similar bug reports. In: Proc. of the IEEE Int'l Symp. on Software Reliability Engineering. 2016. 127−137. [doi: 10.1109/ISSRE.2016.33]

[106] Xia X, Lo D, Ding Y, AI-Kofahi JM, Nguyen TN, Wang X. Improving automated bug triaging with specialized topic model. IEEE Trans. on Software Engineering, 2017,43(3):272−297. [doi: 10.1109/TSE.2016.2576454]

[107] Mani S, Catherine R, Sinha VS, Dubey A. AUSUM: Approach for unsupervised bug report summarization. In: Proc. of the ACM SIGSOFT, Int'l Symp. on the Foundations of Software Engineering. 2012. 1–11. [doi: 10.1145/2393596.2393607]

[108] Rastkar S, Murphy GC, Murray G. Automatic summarization of bug reports. IEEE Trans. on Software Engineering, 2014,40(4): 366–380. [doi: 10.1109/TSE.2013.2297712]

[109] Bettenburg N, Just S, Schr¨oter A, Weiss C, Premraj R, Zimmermann T. What makes a good bug report? In: Proc. of the ACM SIGSOFT Int'l Symp. on Foundations of Software Engineering. Atlanta, 2008. 308–318. http://dblp.uni-trier.de/db/conf/sigsoft/fse2008.html

[110] Zhou J, Zhang H, Lo D. Where should the bugs be fixed? More accurate information retrieval-based bug localization based on bug reports. In: Proc. of the ACM/IEEE Int'l Conf. on Software Engineering. Zurich, 2012. 14–24. https://files.ifi.uzh.ch/icseweb/

[111] Feng Y, Liu Q, Dou M, Liu J, Chen Z. Mubug: A mobile service for rapid bug tracking. Science China Information Sciences, 2016, 59(1):1–5. [doi: 10.1007/s11432-015-5506-4]

[112] Liao XK, Li SS, Dong W, Jia ZY, Liu XD, Zhou SL. Survey on log research of large scale software system. Ruan Jian Xue Bao/Journal of Software, 2016,27(8):1934–1947 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4936.htm [doi: 10.13328/j.cnki.jos.004936]

附中文参考文献:

[3] 冯剑红,李国良,冯建华.众包技术研究综述.计算机学报,2015,38(9):1713–1725. [doi: 10.11897/SP.J.1016.2015.01713]

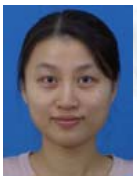
[20] 童咏昕,袁野,成雨蓉,陈雷,王国仁.时空众包数据管理技术研究综述.软件学报,2017,28(1):35–58. http://www.jos.org.cn/1000-9825/5140.htm [doi: 10.13328/j.cnki.jos.005140]

[85] 夏鑫,王新宇,杨小虎,David Lo.开源软件系统缺陷报告管理与分析.计算机学会通讯,2016,2:29–34.

[90] 张志强,逄居升,谢晓芹,周永.众包质量控制策略及评估算法研究.计算机学报,2013,36(8):1636–1649.

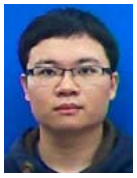
[93] 吴垚,曾菊儒,彭辉,陈红,李翠平.群智感知激励机制研究综述.软件学报,2016,27(8):2025–2047. http://www.jos.org.cn/1000-9825/5049.htm [doi: 10.13328/j.cnki.jos.005049]

[112] 廖湘科,李姗姗,董威,贾周阳,刘晓东,周书林.大规模软件系统日志研究综述.软件学报,2016,27(8):1934–1947. http://www.jos.org.cn/1000-9825/4936.htm [doi: 10.13328/j.cnki.jos.004936]



章晓芳(1980－),女,福建连江人,博士,副教授,CCF 专业会员,主要研究领域为软件分析与测试,众包软件工程,强化学习.

冯洋(1988－),男,硕士,主要研究领域为众包软件工程,软件测试,程序理解,软件仓库挖掘.



刘頔(1994－),男,学士,主要研究领域为众包软件测试,强化学习.

陈振宇(1978－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为测试自动化,群体智能,众包测试,众包数据分析,数据分析,开发者社交网络.

徐宝文(1961－),男,博士,教授,博士生导师,CCF 会士,主要研究领域为程序设计语言,软件工程(软件方法论,软件分析,度量与测试),Web 技术.