图形用户界面测试：系统映射和存储库

Information and Software Technology 55 (2013) 1679–1694

Contents lists available at SciVerse ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Graphical user interface (GUI) testing: Systematic mapping and repository

Ishan Banerjee [a], Bao Nguyen [a], Vahid Garousi [b,c,*], Atif Memon [a]

[a] Department of Computer Science, University of Maryland, College Park, MD 20742, USA
[b] Electrical and Computer Engineering, University of Calgary, Calgary, Canada
[c] Informatics Institute, Middle East Technical University, Ankara, Turkey

1.背景及介绍：

GUI 测试是对具有图形用户界面（graphical user interface）前端的软件进行的系统测试。因系统测试要求整个软件系统 （包括用户界面）作为一个整体进行测试，在 GUI 测试期间，测试用例（建模为用户输入事件序列）通过运行 GUI 的小部件在软件上生成和执行(例如：文本框和可点击的按钮)。自 1991 年以来，在 GUI 测试领域出现 了 230 多篇文章。

在本文中，我们使用系统映射 systematic mapping (SM)来研究现有的知识体。使用 Petersen 等人提出的指南进行 SM，我们提出了三组研究问题。我们定义了选择和排除标准，从最初的 230 篇文章（1991—2011 年）中，我们的文章总共包含 136 篇文章。我们系统地开发了一种分类方案，并将所选文章映射到该方案。我们提出两种类型的结果，首先，我们报告该领域的人口统计和文献计量趋势。包括：本研究领域中被引用最多的文章，活跃的研究人员，顶级文献发表期刊和活跃的国家。此外，我们从文章类型，生成测试用例的方法，文章中使用的评估类型等方面推导出趋势。我们第二个重要结果是一个可公开访问的存储库，其中包含我们所有的映射数据。我们计划定期更新此存储库，使其成为所有研究人员的“实时”资源。我们的 SM 提供了现有 GUI 测试方法的概述，并帮助发现了需要研究人员关注更多的领域。例如，需要做很多工作才能将基于学术模型的技术与商业化工具联系起来。为此，需要研究比较学术技术和工业工具中 GUI 测试的最新状态。该研究的系统架构图如下图所示。

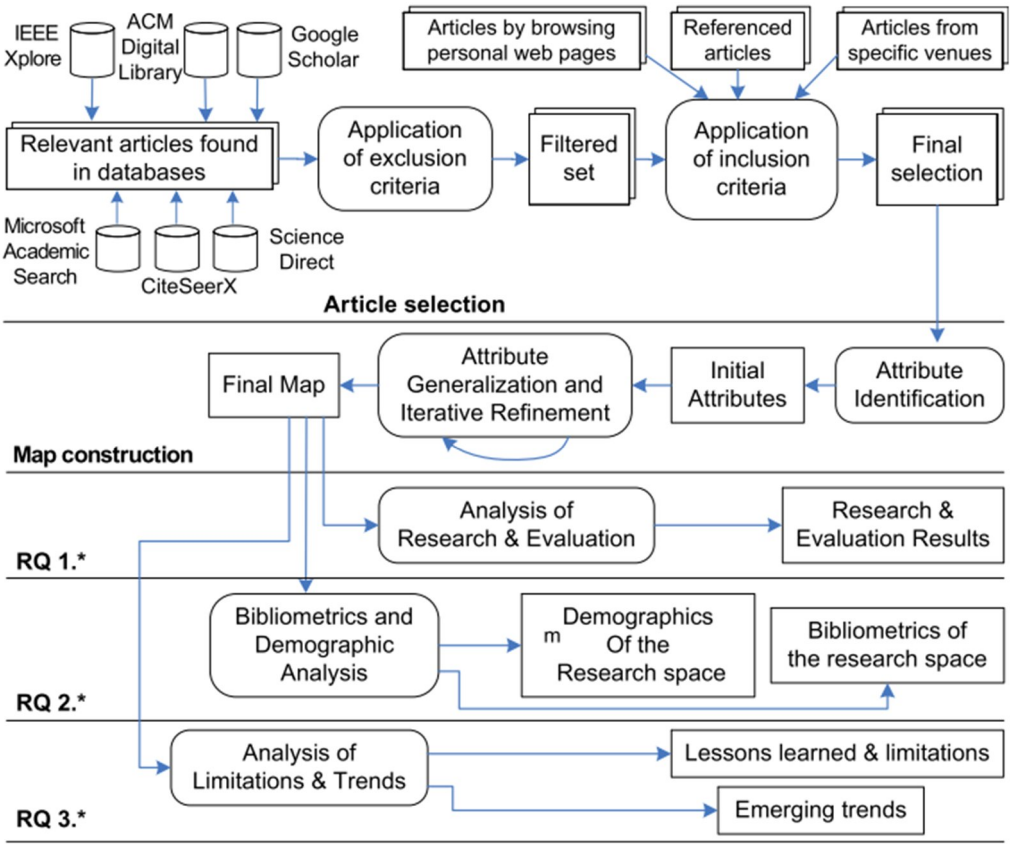


Fig. 1. Protocol guiding this SM. The five distinct phases are **Article selection**, **Map construction**, **RQ 1.∗**, **RQ 2.∗**, **RQ 3.∗**.

2.目标、问题、度量标准

我们使用目标—问题—度量标准（GQM）形成这个 SM 的研究，提出有意义的研究问题，并仔细确定从主要研究中收集的指标以及我们如何使用它们来创建我们的映射图。这项研究的目标如下：

G1：对 GUI 测试领域的文章的性质进行分类，是否正在开发新技术，是否支持工具，优势和劣势，以及突出和总结所面临的挑战和经验教训。

G2：了解研究人员正在研究的 GUI 测试的各个方面（例如，测试用例生成，测试覆盖率）。

G3：研究正在进行的评估的性质，正在使用的工具和目标应用。

G4：确定该领域最活跃的研究人员及其所属，并确定该地区最具影响力的文章。

G5：确定该领域的最新趋势和未来研究方向。

目标中 G1，G2 和 G3 都与理解文章中报告的 GUI 测试研究和评估趋势有关。这些目标导致了我们的第一组研究问题。

RQ 1.1：该地区出现了哪些类型的文章？

RQ 1.2：提出了哪些测试数据生成方法？

RQ 1.3：使用了哪种类型的测试预言？

RQ 1.4：使用/开发了哪些工具？

RQ 1.5：使用了哪些类型的被测系统（SUT）？

RQ 1.6：使用了哪些类型的评估方法？

RQ 1.7：评估机制是自动还是手动？

目标 G4 以及 G1 和 G5 的部分内容涉及文章和作者的人员统计和文献统计这些目标导致了我们的第二组研究问题。

RQ 2.1：每年的文章数量是多少？

RQ 2.2：按场地类型划分的文章数量是多少？

RQ 2.3：按场地类型的引用次数是多少？

RQ 2.4：依据引用次数最具影响力的文章是什么？

RQ 2.5：文章数量最多的场地是什么？

RQ 2.6：引用次数最多的场地是什么？

RQ 2.7：文章数量最多的作者是谁？

RQ 2.8：作者的所属，即他们属于学术界还是工业界？

RQ 2.9：文章数量最多的国家有哪些？

目标 G5 和 G1 的部分内容涉及 GUI 测试领域的最新趋势、局限性和未来研究方向，我们提出了第三组研究问题。

RQ 3.1：报告了哪些限制？

RQ 3.2：报告了哪些经验教训？

RQ 3.3：该领域的趋势是什么？

RQ 3.4：未来的研究方向是什么？

3. 文章选择

第 1 步：文章识别，我们通过进行基于关键字的搜索来启动该过程，从以下数字图书库和搜索引擎提取文章列表。IEEE Xplore，ACM Digital Library，Google Scholar，Microsoft Academic Search，Science Direct，and CiteSeerX。用以下关键字搜索：GUI testing，graphical user interface testing，UI testing，and user interface testing，我们在文章标题和摘要中查找了这些关键字。该步骤产生了 198 篇文章，形成了初始文章库。

第 2 步：排除标准，在该过程的第 2 步中，定义了以下一组排除标准以从上述初始库中排除文章。C1：英语以外的语言，C2：与主题无关，C3：未出现在公开会议、学术报告会和研讨会，或未出现在期刊或杂志中。对于标准 C2，我们使用投票机制来评估文章与 GUI 测试的相关性。应用上述排除标准产生了 107 篇文章。

第 3 步：纳入标准，由于搜索引擎可能会遗漏与我们研究相关的文章，我们

通过手动检查以下三个信息来补充我们的文章集:(1)活跃研究人员的网页,(2)我们库中文章的参考书目部分，(3）特定场所。最终文章库共有 136 篇文章。

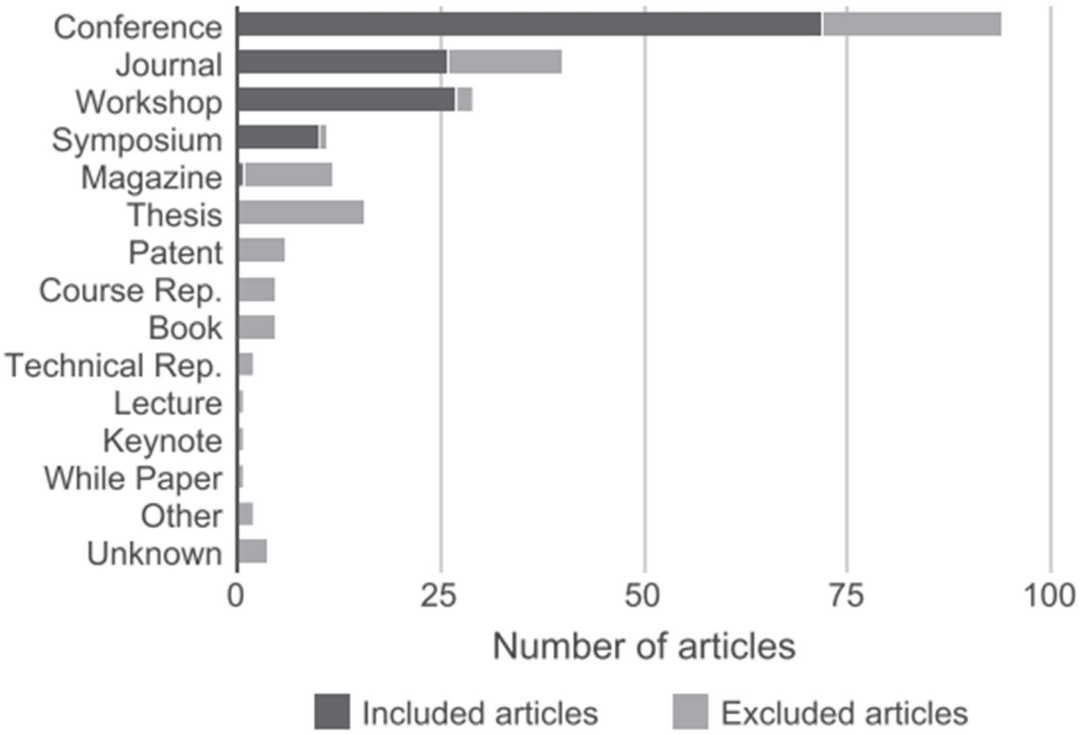


Fig. 2. Total articles studied = 230; final included = 136.

4. 系统图的迭代开发

系统图是用于对所选文章进行分类的工具。系统图的开发是一个复杂且耗时的过程。我们使用 GQM 标准作为系统图构建指南。对于 RQ1.*我们需要收集指标：“文章类型”，“测试数据生成方法”，“测试预言的类型”，“工具”“被测系统的类型”，“评估方法的类型”，“评估机制”。这个列表实际上形成了文章的一组属性。我们定义这些属性并呈现系统图结构,使用此映射（也称为属性框架）所研究的文章可以进行全面的表征。

系统图是以迭代方式开发的，在第一次迭代中分析所有文章并列出似乎对特定方面感兴趣或相关的术语。这个分项任务是由我们所有人执行的，为了减少个人偏见，我们没有假设任何属性或关键字的先验知识，分析所有文章后得到的结果是一大组初始属性。在识别出初始属性之后对其进行推广，这是通过一系列会议实现的。为文章类型定义属性相当复杂，我们定义了两个方面即贡献和研究方面。记录每个方面的结果属性，每个方面内的属性以及每个属性要简要描述系统图，该系统图构成了回答研究问题 RQ 1.*的基础。同样对于 RQ 2.*我们需要以下指标：“年度文章数量”，“按地点类型计算文章数量”，“按地点类型计算文章

引用”“引用数量”，“具有最高文章数量的地点”，“文章最多的作者”等。这些指标引导我们开发第二张系统图。最后，对于 RQ 3.*我们需要收集指标：“局限性”，“经验教训”，“趋势”和“未来研究方向”。这使得我们开发了第三张系统图。

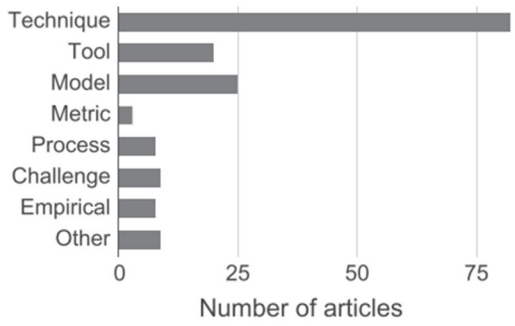
RQ 1.1: Type of article	*Contribution Facet*	Test method/technique (B)	Article describes new technique or improves upon an existing one
		Test tool (B)	Article focuses on testing tool and evaluates its applicability.
		Test model (B)	Article introduces new modeling technique or is based on use of model
		Metric (B)	Article describes new metric for evaluating testing techniques
		Process (B)	Article describes software testing process or life-cycle
		Challenge (B)	Article discusses challenges in certain areas of GUI testing
		Empirical study (B)	Article is an empirical study of technique
	Research Facet	Solution proposal (B)	New solution; applicability shown via example or line of argument
		Validation research (B)	Novel technique demonstrated in lab with experiment
		Evaluation research (B)	Comprehensive experimental evaluation of technique
		Experience article (B)	Personal experience with GUI testing of authors
		Philosophic article (B)	Sketch new way of looking at existing things.
		Opinion article (B)	Opinion of authors on the goodness of techniques.
RQ 1.2	Test data generation	Capture/replay (B)	Capture/replay was used to generate test cases
		Model based (B)	GUI model was used to generate test cases
		Model name (S)	Name of model used (if model-based)
		Random testing (B)	Test cases were generated randomly
RQ 1.3	Test oracle	State reference (B)	GUI state information was used as oracle
		Crash testing (B)	SUT crash was used to identify faults
		Formal specification (B)	Formal specification of SUT was used as oracle
		Manual verification (B)	Result of test case execution was manually verified
		Multiple oracles (B)	More than one oracle was used in same test run
RQ 1.4	Testing tools	Tool proposed (S)	Name of new tool introduced in an article
		Tool used (S)	Name of existing or third party tool used in an article
		Programming language (S)	Programming language used in developing the tool
RQ 1.5	System under test	Number of SUT(s) (N)	Number of SUT(s) used in the article
		Size (LOC) (N)	Number of lines of code in SUT
		Programming language (S)	Programming language of the SUT
		GUI technology (S)	GUI SDK or library used in SUT
		Small/large scale (S)	Qualitative assesment of the size of SUT
RQ 1.7	Evaluation Automation	Automated (B)	Automated test case execution was used in article
		Manual (B)	Manual test case execution was used in article
		None (B)	Test cases were not executed
RQ 2.*	Demographic Information	Authors (S)	Name of all contributing authors
		Authors' country (S)	Country from which author published the article
		Authors' affiliations (E)	Are the authors from academia, industry or a mix of both
		Venue (S)	Where it was published
		Year (N)	Year of publication
		Citation count (N)	Number of times this work has been cited, per year, as of 2011
RQ 3.*	Limitations & Future	Limitations (S)	Limitations noted by article authors
		Lessons learned (S)	Lessons learned
		Future research (S)	Future research directions

* B = Boolean, E = Enumerated, N = Numeric, S = String

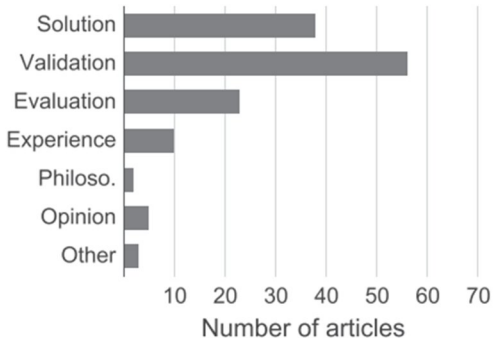
Fig. 3. The final map produced by and used in this research.

5. 映射研究和评估

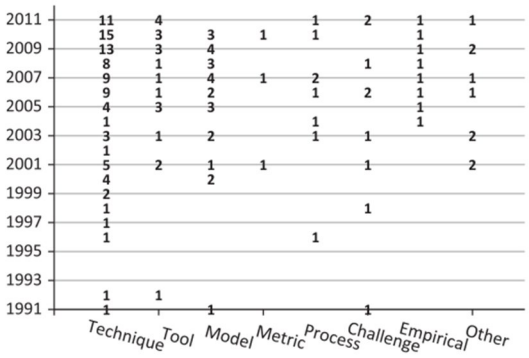
RQ 1.1：图 4a 显示所有 136 篇文章的贡献方面。y 轴列举类别，x 轴显示每个类别中的文章数，大多数文章（90 篇文章）都有助于开发新的或改进测试技术。很少有文章探讨过 GUI 测试指标或开发测试流程。图 4c 显示了贡献方面的年度分布。图 4b 显示了所有 136 篇文章的研究方面，大多数文章提出解决方案，进行各种类型的实验来验证技术。图 4d 显示了研究方面的年度分布，该图显示了近年来的文章数量不断增加，其中大多数文章在研究方案解决建议、验证和评估方面。在 2011 年，最多的文章是关于验证研究，这是一个有希望的方向，表明研究人员不仅要提出新技术，而且他们也要通过实验来支持它们的新技术。图 4e 列举了研究方面和贡献方面的文章。



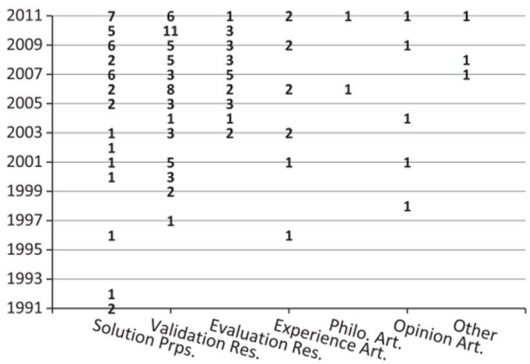
(a) Contribution facet distribution.



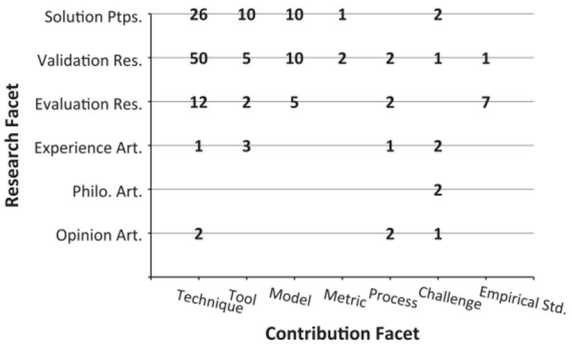
(b) Research facet distribution.



(c) Contribution facet – annual trend



(d) Research facet – annual trend



(e) Contribution vs. Research facet

Fig. 4. Data for **RQ 1.1**.

RQ 1.2：提出了哪些测试数据生成方法。在 136 篇文章中，有 123 篇报道了。例如：测试用例，测试输入数据，预期输出，测试预言，测试需求，测试工具，测试代码等。图 5a 显示了测试数据生成方法的分布。到目前为止，大多数文章（72 篇文章）依赖于测试生成的模型。 图 5b 显示了这 72 篇文章的组成。

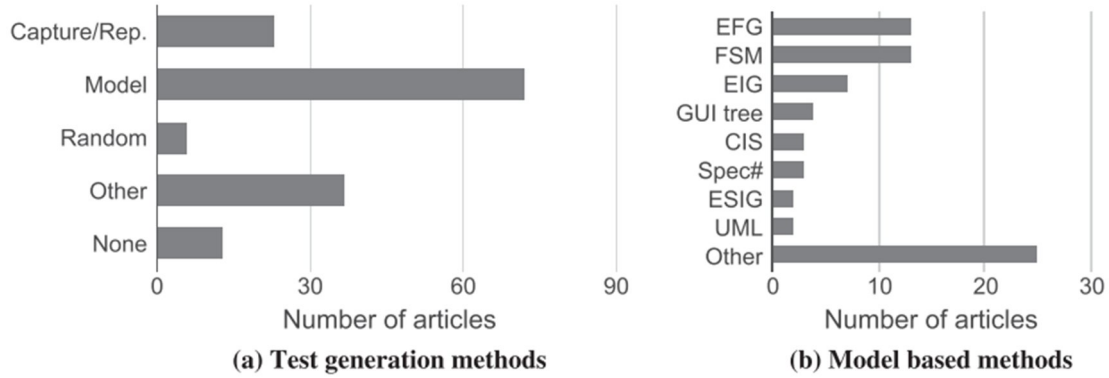


Fig. 5. Data for **RQ 1.2**.

RQ 1.3：使用了哪种类型的测试预言。测试预言是一种确定测试用例是通过还是失败的机制。图 6 显示，状态参考（37 篇文章）是常用的预言，在该方中，在 SUT 执行时提取 GUI 的状态并将其存储，然后可以将该状态与另一个执行实例进行比较以进行验证。验证方法（13 篇文章）使用模型或规范来验证测试用例输出的正确性。我们观察到大量文章（49 篇文章）没有使用测试预言。

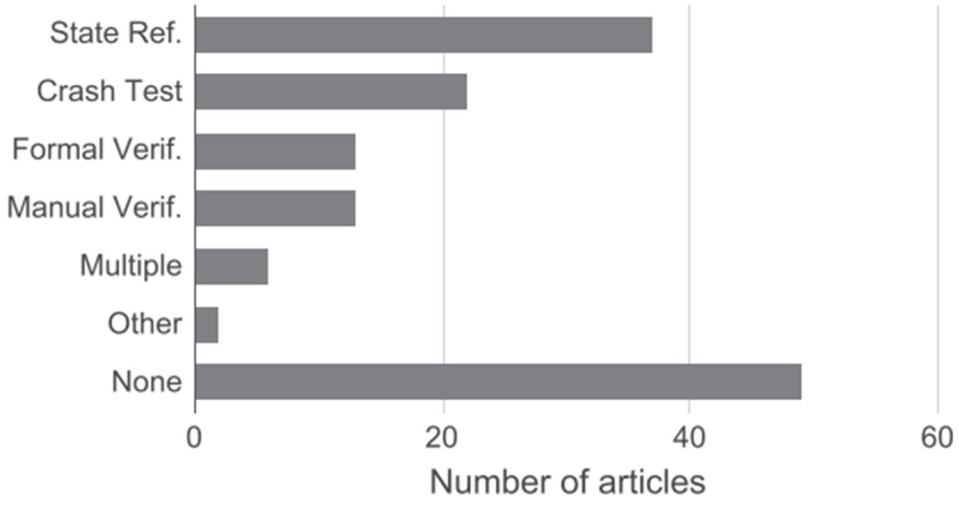


Fig. 6. Data for **RQ 1.3**: Oracle type.

RQ 1.4：使用/开发了哪些工具。如果工具是专门为文章中第一次使用而开发的，则该工具被视为一种新工具。如果工具是在以前工作中开发的，或者是由第三方开发的（商业的、开源的）等则该工具被视为现有工具。图 7a 显示了所有 136 篇文章中新工具和现有工具的组成，可以看出，32 篇文章（23.52%）仅引入了新工具，48 篇文章（35.29%）仅使用现有工具，29 篇文章（21.32%）使

用新工具和现有工具，而 27 篇文章（19.85%）没有使用工具。从这一点可以看出，大多数文章（109 篇文章）使用了一种或多种工具。图 7b 显示了十种最受欢迎的工具及其使用次数。GUITAR 排名最高，已被用于 22 篇文章中。图 7c 显示了开发的工具使用的编程语言的分布，这通常也意味着它在 SUT 方面支持的编程语言，Java 预言是迄今为止最受欢迎的编程语言，在 23 篇文章中使用。

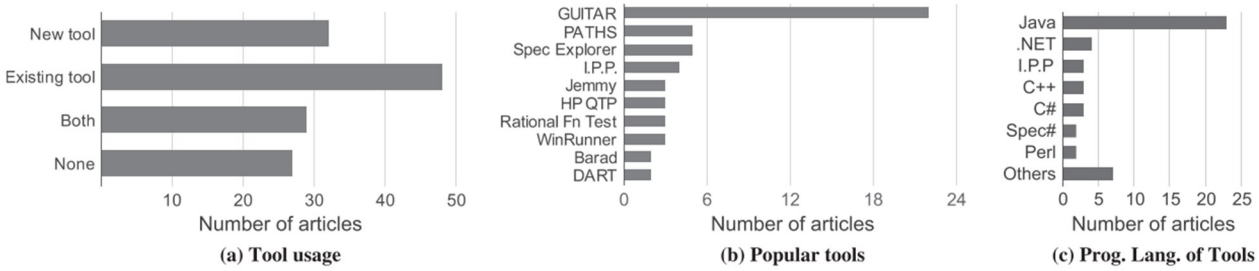


Fig. 7. Data for **RQ 1.4**.

RQ 1.5：使用了哪些类型的被测系统。在 136 篇文章中有 118 篇报道了使用了一种或多种 SUT，图 8a 显示了每篇文章中使用的 SUT 数。图 8b 显示了 71 篇文章报道的 SUT 的编程语言，我们看到 Java 应用程序是迄今为止最常见 SUT 有 48 篇文章使用基于 Java 的 SUT。图 8c 显示了每篇文章中 SUT 使用的累积 LOC，136 篇文章中只有 28 篇报道了这一信息，17 篇文章中 SUT 的 LOC 在 10000-100000 范围内，只有 5 篇文章中 SUT 的 LOC 总数超过 100000 行。

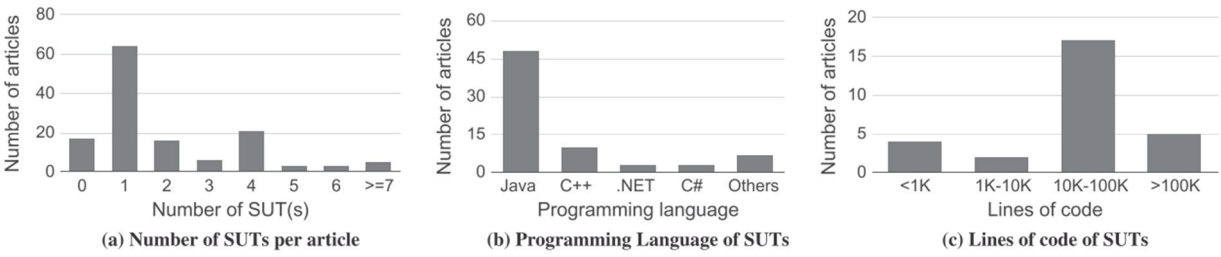
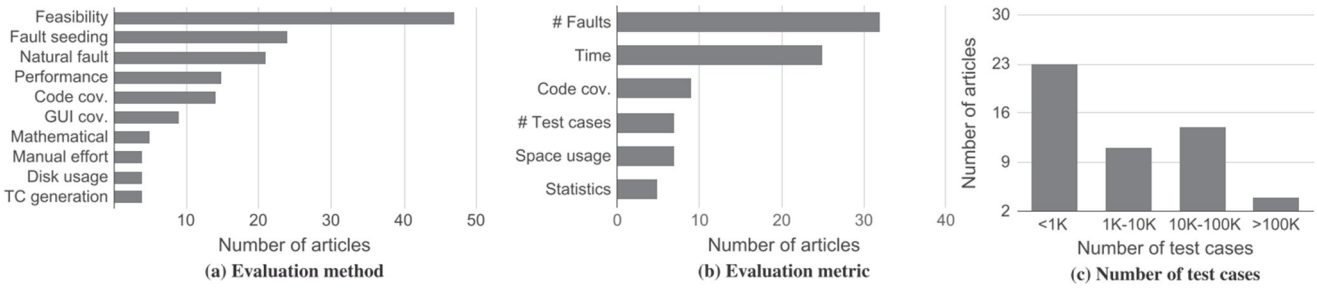


Fig. 8. Data for **RQ 1.5**.

RQ 1.6：使用了哪些类型的评估方法。共有 119 篇文章报道了一种或多种评估方法。图 9a 显示评估方法的分布，图 9b 显示评估中使用的指标，图 9c 显示了使用的测试用例数，大多数文章使用不到 1000 个测试用例。



RQ 1.7：评估机制是自动还是手动。在 136 篇文章中，86 篇文章报道了评估测试用例的执行情况，其中 72 篇报道了自动化测试用例执行，11 篇文章是手动测试用例执行，3 篇文章报道了自动和手动测试用例共同执行。

6. 映射人员统计

我们现在讨论 RQ 2.*研究问题集，这与理解文章和作者的人口统计有关。

RQ 2.1：每年的文章数量是多少？从 1991 年至 2011 年的出版趋势如图 10 所示，该库中最早的两篇文章由 Yip 和 Robson 于 1991 年出版，2010 年和 2011 年的 GUI 测试文章总数都为 19。

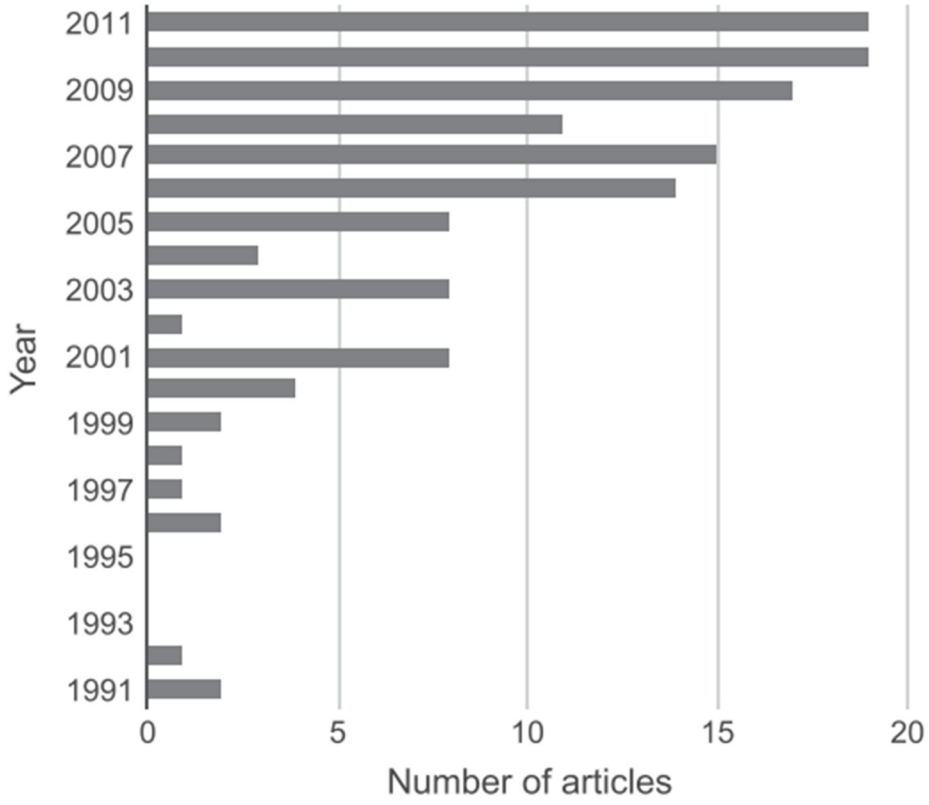


Fig. 10. RQ 2.1: Annual counts.

RQ 2.2：按场地类型计算文章数量是多少。我们将场地按会议、期刊、研讨会、学术报告会或杂志对文章进行分类。图 12 显示会议文章的数量（72 篇）超过其他四个场地类型文章数量的总和。

RQ 2.3：场地类型的引用次数是多少。针对每种场地类型提取和汇总每篇文章的引用次数。图 12 显示来自不同场地类型的引用次数。会议文章有 1544 次的最高引用次数。

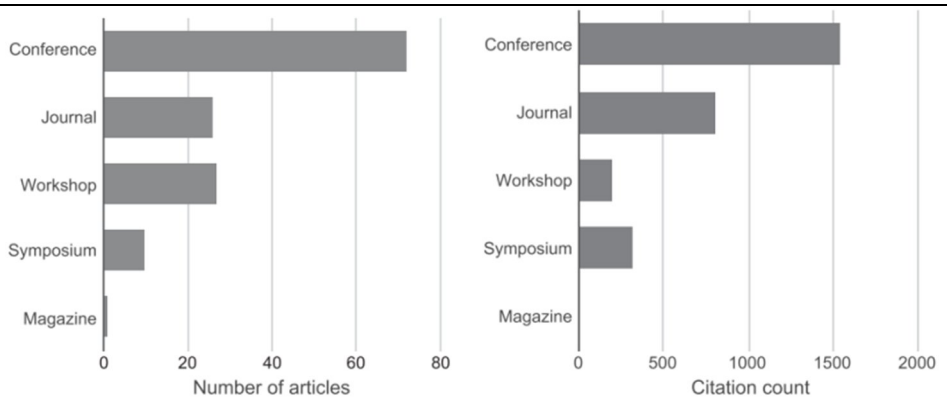


Fig. 12. RQ 2.2 and 2.3: Venue types.

RQ 2.4：引用次数最具影响力的文章是什么。该问题研究分析了每篇文章的引文与其出版年份之间的关系，图 11 显示此数据。最近几篇文章（从 2006 年到 2011 年）的引用彼此更接近，表明大多数最近几篇文章由于篇幅较短，文章在该领域产生影响并被引用需要时间，所以收到的引文数量相对相同。最早的三篇文章的引用率相对较低，引用次数最多的文章是 Memon 等人 2001 年在 IEEE TSE 发表的文章。标题为“Hierarchical GUI Test Case Generation Using Automated Planning”收到 204 次引用。

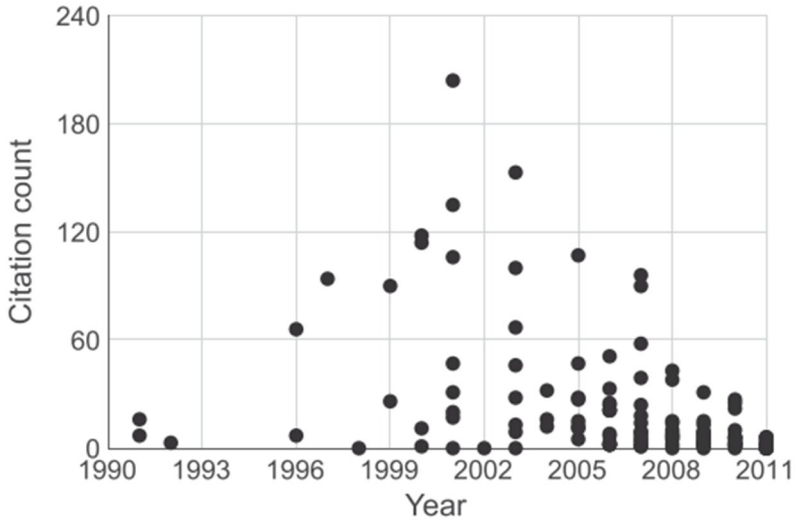


Fig. 11. RQ 2.4: Citations vs. year.

RQ 2.5：文章数量最多的场地是什么。图 13 显示了来自前二十个场地的文章数量，这些文章贡献了 80 篇文章。International Workshop on TESTing Techniques & Experimentation Benchmarks for Event-Driven Software (TESTBEDS) 是一个相对较新的期刊，始于 2009 年。由于场地特别注重测试 GUI 和事件驱动软件，因此在 2009 年至 2011 年期间发布了较多的文章（16 篇）。International Conference on Software Maintenance (ICSM)有 8 篇，IEEE Transactions on Software Engineering

(TSE)有 6 篇。

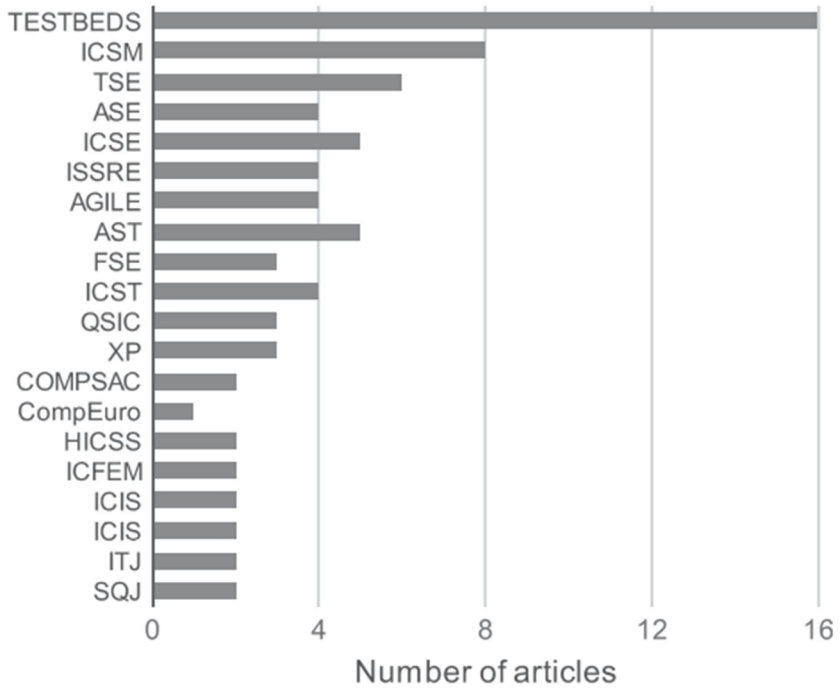


Fig. 13. Data for **RQ 2.5:** Top 20 venues.

RQ 2.6：引用次数最多的场地是什么。图 14 表明前三个引用的场所是 IEEE TSE、ACM SIGSOFT Symposium on the Foundations of Software (FSE)、International Symposium on Software Reliability Engineering (ISSRE)。

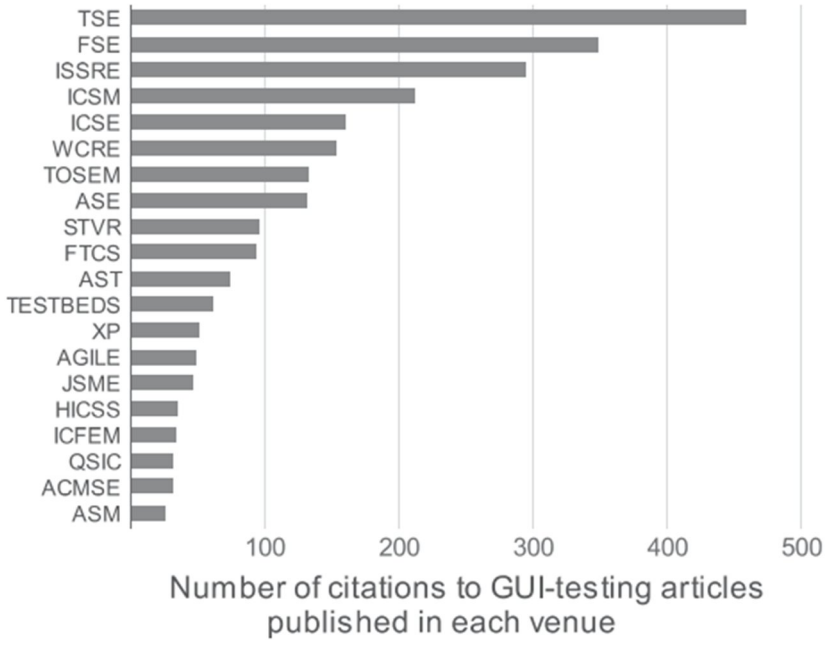


Fig. 14. Data for **RQ 2.6**: Venues most cited.

RQ 2.7：文章最多的作者是谁。如图 15 所示，Atif Memon（马里兰大学）排在首位有 32 篇文章。 排名第二和第三的作者是 Qing Xie (埃森哲技术实验室)和 Mary Lou Soffa（维尔吉尼亚大学）分别有 13 篇和 7 篇文章。

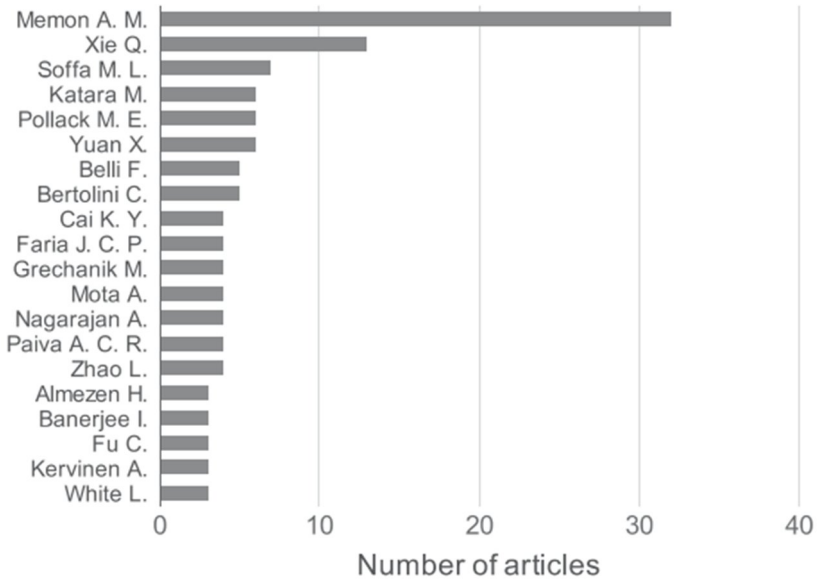


Fig. 15. Data for **RQ 2.7**: Top 20 authors.

RQ 2.8：作者的所属，即他们属于学术界还是工业界。我们根据作者的意见将文章归类为以下三类中的一类：学术界、工业界和学术工业界（即作者的文章来自学术界和工业界）。结果如图 16 所示，73.52%，13.23% 和 13.23%的文章分别来自学术界、工业界实践者，和学术工业界实践者。近年来学术界和工业界发表的文章数量稳步上升，此外学术界和工业界者之间的合作文章数量也在不断增加。

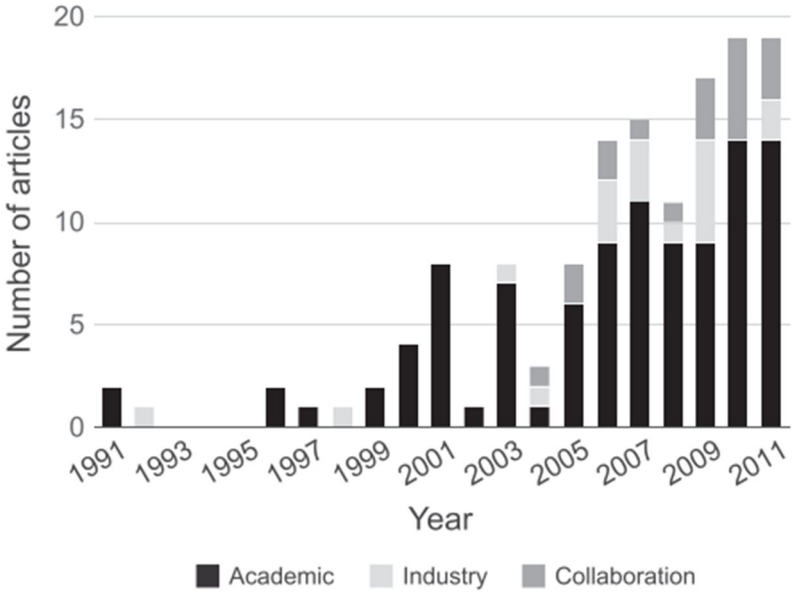


Fig. 16. Data for **RQ 2.8**: Author affiliation trend.

RQ 2.9：哪些国家的文章数量最多。为了根据发表的文章数量对国家进行排名，提取了作者居住国。如果一篇文章有来自多个国家的多位作者，则为该作者的国家记一分。结果如图 17 所示，美国研究人员撰写或共同撰写了 51.47%（136 篇文章中的 70 篇）文章。来自中国和德国的作者（分别有 12 篇 和 9 篇文章）排在第二和第三位。世界上只有 20 个国家为 GUI 测试知识体系做出了贡献，GUI 测试研究人员之间的国际合作尚未得到充分发展，因为 136 篇文章中只有 7 篇是跨越两个或更多国家的合作。其余大多数文章都是由一个国家的研究人员撰写的。

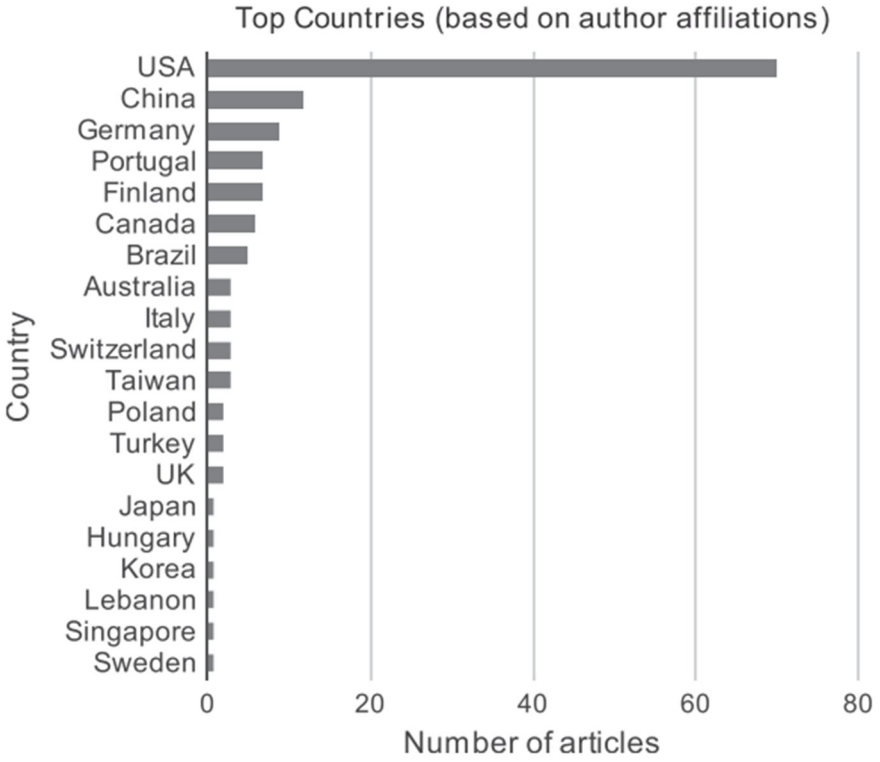


Fig. 17. Data for **RQ 2.9**: Top author countries.

6. 映射局限性和未来方向

研究问题 RQ 3.*通过分类报道文章中的局限性和未来发展方向来解决。

RQ 3.1：提出了哪些局限性。许多文章明确地指出了工作的局限性。这些局限性被大致归类为如下：

算法：所提出的技术或算法具有已知的局限性，例如，算法可能不能很好地处理源代码中的循环。

适用性：不同环境下可用性的限制，例如工具或算法可能是基于 AWT 的特定应用程序。

手动：在实验中使用手动步骤，这可能会限制方法的可用性和可扩展性。手动步骤也可能影响实验或技术的质量。例如可能需要手工努力来维护模型。

测试预言（Oracle）：用于实验的测试预言可能在检测所有故障方面受到限

制，例如，oracle 可能仅限于检测 SUT 的崩溃或异常，而不是比较 GUI 状态。

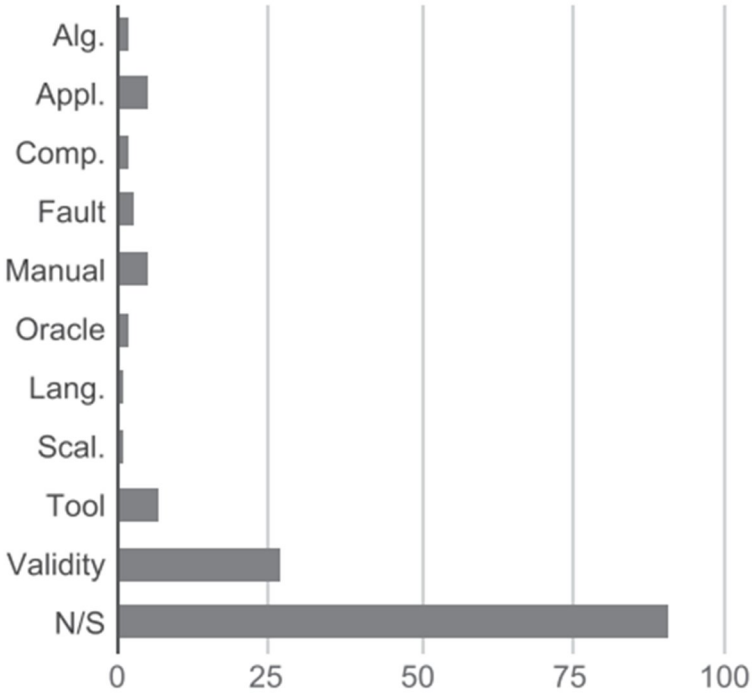
故障：检测各种故障或检测错误缺陷能力的限制，例如，工具可能无法很好地处理意外情况。

可伸缩性：该方法无法很好地扩展到大型 GUI，例如，执行算法所花费的时间可能会随 GUI 大小呈指数级增长。

工具：在使用或提出的工具中存在一些已知的工具的限制或缺少明显的功能，例如，工具可能只处理某些类型的 GUI 事件。

有效性：实验结果受内部或外部有效性的影响。

作者本身并未自己判断局限性，而是我们提取了每个初步研究中明确提到的局限性。在 136 篇文章中有 45 篇文章报道了研究工作的一个或多个局限性。如图 18 所示。这一点可以帮助我们理解作者所注意到研究工作的局限性。



(a) Limitations

RQ 3.2：报道了哪些经验教训。只有少数作者明确报告了他们从研究中吸取的教训。所有文章中只有 11.76%（16/136）报告了所吸取的经验教训。所吸取的教训因作者而异。它们很大程度上依赖于个人的研究和学习环境。例如在某些情况下，关注于基于模型测试的作者在他们的方法中，大量的工作将花费在模型创建上。在其他一些情况下，使用基于自动化逆向工程模型技术的作者得出结论，由于模型是自动创建的，测试人员的大部分工作将花在测试维护上。

同样实验环境也影响了作者的结论。一些计算资源有限的作者认为，应该在

测试选择、测试优先级划分和测试重构上投入更多的研究工作，以减少要执行的测试用例的数量。但是有些具有丰富计算资源的作者表示，未来的研究应侧重于大规模研究。

RQ 3.3：该领域的趋势是什么。基于 Java 的 SUT 和工具的广泛使用似乎很常见。一个值得注意的发展趋势是与 1991-2007 年期间只有 1 篇文章相比，2008-2011 年期间移动平台上 GUI 测试工作出现了 8 篇文章。另一个显著的趋势是由于自动化工具和廉价计算资源的可用性，从单元脚本测试转移到大规模的自动化系统测试已经进行了若干大规模的实证研究。

RQ 3.4：未来的研究方向是什么。GUI 测试是软件工程中一个相对较新的研究领域。大多数文章为继续研究提供了指导，可以大致分为 以下几类：

算法：扩展现有算法或开发新算法。例如，扩展算法以处理潜在的大量执行路径。

分析：基于给定研究结果的进一步研究。例如，研究具有完整交互序列的不同 GUI 组件的交互。

覆盖率：本文中介绍的基于覆盖率的技术可以进一步改进或评估，覆盖技术可能适用于代码、GUI 或模型覆盖。例如，该研究报告制定了新的覆盖标准。

评估：评估所提出的方法和技术，进一步扩展基于现有结果的研究。例如，进行更多对照实验。

平台：扩展其他平台的实施。例如，Web 端和移动端。

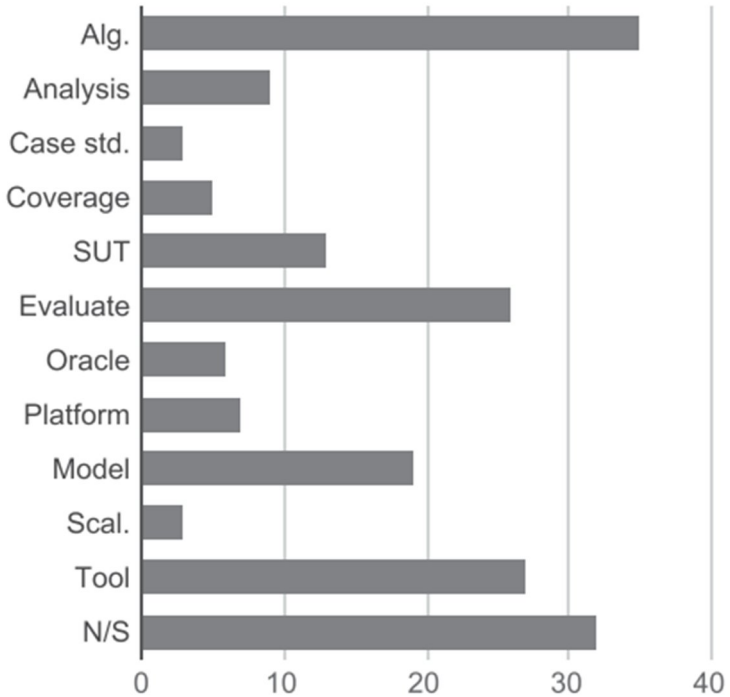
模型：改进或分析文章中给出的模型。例如，模型的自动生成。

可伸缩性：将提出的算法扩展到更大的系统，降低计算成本。例如，扩展算法以处理更大的 GUI，同时提高执行性能。

被测系统（SUT）：使用更多 SUT 评估所提出的技术。例如，使用复杂的 SUT 进行评估。

工具：向本文中讨论的工具扩展或添加新的功能或特性。例如，改进工具以支持更好的模式匹配并更好地从错误中恢复。

图 18b 显示了从文章中提取的未来研究方向，这些数据可以帮助我们了解研究人员提出的指导意见。虽然这些数据包含了可追溯到 1991 年的未来方向，但它有助于我们了解研究人员在此期间的想法，以及他们的成果在执行和发表时所认为的缺失部分。可以看出改进算法（35 篇文章）被认为是最需要它们研究的方向，改进和开发更好的 GUI 测试工具（27 篇文章）也被认为是需要进一步工作的领域。



(b) Future research

7. 总结

从收集的数据可以看出，基于模型的 GUI 测试技术已经引起了研究领域的极大关注。然而，诸如 JFCUnit、Android Monkey、Quick Test Pro、Selenium 等工业工具并不是基于模型的。没有文章将业界流行的 GUI 测试技术、方法和实践与研究领域正在开发的那些进行比较。从业者和研究人员之间普遍缺乏合作。为了与计算技术的最新趋势保持一致，最近在基于移动端和 web 平台上也出现了 GUI 测试文章。

大量文章（136 篇中有 109 篇）使用 GUI 测试工具。使用了多达 112 种工具。这表明虽然在开发用于 GUI 测试活动方面已经付出了大量的努力，但是 GUI 测试工具还没有太多的标准化，不同的研究人员为了特定的研究目的开发自己的定制工具。这些工具通常不能被其他研究人员使用，因为它们不能被广泛应用。而且这些工具通常不能在多年内维护、调试和开发以供其他研究人员使用。

大多数文章讨论新的 GUI 测试技术和工具。目前普遍缺乏能够向研究人员介绍最新技术观点或对未来可能发展和研究方向提供指导的文章。

（本文翻译工作由王川涛本人独立完成）