

基于搜索的软件工程研究进展与趋势

CCF 软件工程专业委员会

李 征¹ 巩敦卫² 聂长海³ 江 贺⁴

¹北京化工大学信息科学与技术学院, 北京

²中国矿业大学信息与电气工程学院, 徐州

³南京大学计算机科学与技术系, 南京

⁴大连理工大学软件学院, 大连

摘 要

软件工程自从 1968 年提出以来, 一直是提高软件开发效率, 保障软件质量的有效手段。基于搜索的软件工程 (Search Based Software Engineering, SBSE) 是传统软件工程和智能计算 (Intelligent Computing) 交叉的新兴研究领域, 它采用智能计算领域的现代启发式搜索优化算法解决软件工程相关问题, 核心是实现智能化和自动化的软件工程相关问题求解, 被 2007 年度 IEEE 国际软件工程大会 (ICSE) 确立为软件工程的未来发展方向之一。基于搜索的软件工程已经在软件测试数据自动生成、程序错误自动修复等方面取得显著的研究成果, 有效地促进了软件工程学科的发展。报告将从两个方面论述国内外发展现状: 一方面是智能计算领域相关技术综述, 特别是已经应用到软件工程领域的相关演化算法的综述; 另一方面, 从软件工程的各个阶段 (包括需求分析、设计、测试、维护等) 综述基于搜索的软件工程发展现状。

关键词: 软件工程, 智能计算

Abstract

Software Engineering was introduced in 1968, and it has been an effective way to improve the software development process and assure the software quality. Search based software engineering (SBSE) is a promising combination research area of the traditional software engineering and intelligent computing, which using meta-heuristic search algorithms to solve different optimization problems in diverse software engineering areas. SBSE approaches can be implemented automatically and intelligently to obtain solutions for complex tasks and it has been recognized as a promising future of the software engineering. There have been many exciting research achievements in SBSE, such as automatic test case generation, automatic bugs fixing. This report will introduce the SBSE in two aspects, one is the survey of the search algorithms in intelligent computation, which have been applied in SBSE; the other is the art of state of the SBSE, including requirement, software design, testing and maintenance.

Keywords: software engineering, intelligent computation

1 引言

软件危机的出现,促进了软件工程学的形成和发展,并不断进化。从软件工程开发方法来看,人们先是从面向 01 代码的原始数字信息到面向过程 (Procedure Oriented),再从面向过程提升到面向对象 (Object Oriented),再从面向对象进化到面向服务 (Service Oriented)、面向方面 (Aspect Oriented)、面向领域 (Domain Oriented) 等。软件工程的发展历史本身就是一个不断提出问题又不断解决问题的过程。

基于搜索的软件工程 (Search-Based Software Engineering, SBSE[⊖]) 是将传统的软件工程问题转化为基于搜索的优化问题,并使用现代启发式搜索算法[⊖] (meta-heuristic search algorithms, 也称元启发式算法) 解决问题的研究和实践方法。相比启发式搜索算法,现代启发式搜索算法定义智能搜索策略,增强启发式算法搜索性能,以一种智能形式在问题的解空间中搜索最优解或近似最优解。

SBSE 最早可以追溯到 1976 年,Webb Miller 和 David Spooner 尝试把优化算法用于浮点测试数据的生成^[1]。1992 年,Xanthakis 和他的同事首次将搜索算法用于解决软件工程问题^[2]。2001 年,Harman 和 Jones 正式提出将软件工程问题转化为基于搜索的优化问题,并采用以遗传算法 (Genetic Algorithm)、模拟退火算法 (Simulated Annealing Algorithm)、禁忌搜索算法 (Tabu Search Algorithm) 等为代表的现代启发式搜索算法来求解^[3],这可以说是奠定了基于搜索的软件工程的里程碑。面对规模日益庞大和复杂的软件,传统的软件工程方法已经不能有效解决软件开发过程中的问题。基于搜索的软件工程的提出,为解决这些问题提供了新的解决思路,因此被 2007 年的 IEEE 国际软件工程大会正式确立为软件工程领域未来发展的新方向^[4]。

传统的软件工程的解决问题方法是在问题空间通过算法来构造一个解,而基于搜索的软件工程是在解空间 (所有可能的解) 中使用启发式搜索算法以具体问题的适应值函数作为向导搜索最优解。通常,使用基于搜索的优化算法解决问题,需要满足以下两个条件^[3,4]:

- 1) 设计出问题解决方案表达方式 (Solution Representation): 对所需解决问题的结果,必须能通过相应的编码表示出来,以构成搜索算法中的染色体,进行相应的运算。
- 2) 设计出相应的适应度函数 (Fitness Function): 对解进行评价,比较不同解之间的优劣。在搜索解空间内,适应度函数可以指引搜索的方向,寻找满足条件的区域。

Harman 在文献[4]和[5]中曾表述,软件工程师对他们的问题已经有一个合适的呈现,并且许多软件工程师在软件度量方面所取得的成果,也为构造适应度函数打下了基础。

⊖ 后续内容将“基于搜索的软件工程”简称为“SBSE”。

⊖ 后续内容中提到的“搜索算法”或“搜索技术”通常指“现代启发式搜索算法或技术”。

由于基于搜索的软件工程解决问题的方法主要由以上两步组成,针对任何问题,只要能够设计出问题解的表示方式和适应度函数,就可以应用该方法,因此具有很强的普适性,可以方便地应用到不同领域问题上。另外,针对同一问题的不同规模,基于搜索的方法求解方式是不变的,因此容易扩展到大规模的工程问题求解上去。基于搜索的软件工程方法避免了传统方法的一些不足,能在尽可能降低成本的前提下尽可能高效地自动化完成软件开发任务,是软件工程领域中一种高效实用的新方法。

经过十几年的发展,基于搜索的软件工程研究已经取得了显著的成绩。该领域的创始人 Harman 教授所在的研究团队创建并维护了一个 SBSE 研究领域的文献库,收录主要国际期刊和国际会议关于 SBSE 的发表文献。根据该文献库统计数据,截至 2013 年,从事 SBSE 研究团体已超过 270 个,遍布全球 40 多个国家,超过 800 名研究人员,直接相关的发表文章超过 1000 篇[⊖]。

图 1 显示了 1976~2011 年期间基于搜索的软件工程领域发表文章情况,可以看出近 10 年发表文章数量呈现显著增长趋势。

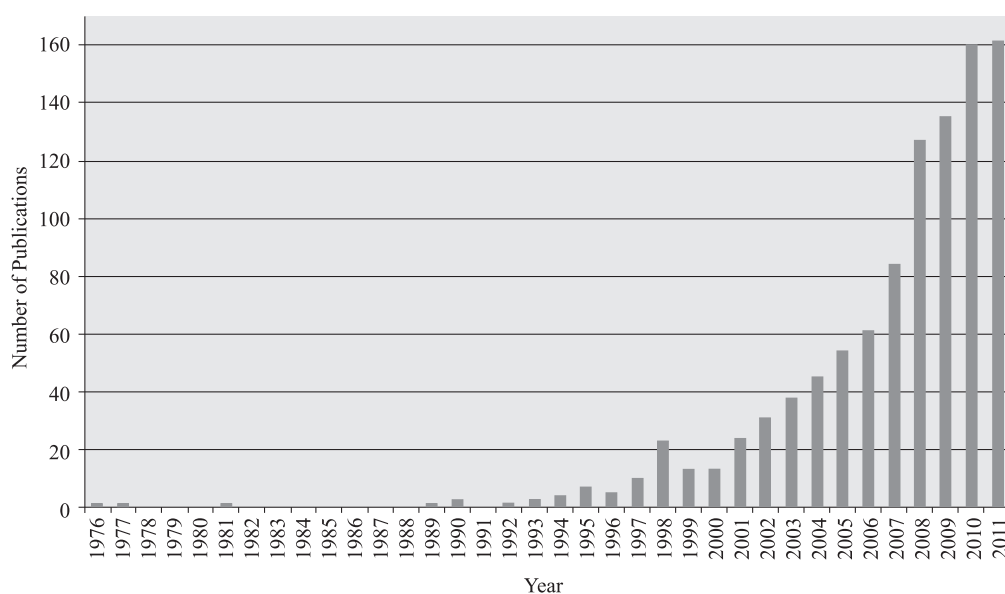


图 1 SBSE 领域发表文献趋势

基于搜索的软件工程已经形成一个独立的研究方向,软件工程领域主流学术期刊,包括顶级 IEEE Transactions on Software Engineering (TSE) 在内,都相继出版了该领域的专刊,具体如表 1 所列。

⊖ The repository of publications in SBSE (<http://www.sebase.orgsbsepublications/>) .

表 1 基于搜索的软件工程国际期刊专刊

年度	期刊名称
2007	Journal of Software Maintenance and Evolution (JSME)
2008	Computers and Operations Research (COR)
2010	IEEE Transactions on Software Engineering (TSE)
2010	The journals Information and Software Technology (IST)
2011	Software: Practice and Experience (SPE)
2011	EMpirical Software Engineering (EMSE)
2013	Journal of Systems and Software (JSS)

从研究内容上看，基于搜索的软件工程已经应用到软件开发生命周期的各个阶段，从测试需求和工程规划，到软件的维护和重建。本文将在后续的章节中详细介绍基于搜索的软件工程在生命周期不同阶段的具体应用。

2 国际研究现状

SBSE 是智能计算与软件工程的交叉，报告将从两个方面论述国内外发展现状：一方面是智能计算领域的相关技术综述，特别是已经应用到软件工程领域相关智能优化方法的综述；另一方面重点从软件工程的各个阶段（包括需求分析、设计、测试、维护等）综述 SBSE 的发展现状并结合具体应用（包括应用场景、应用效果等）论述 SBSE 在工业界的应用现状。

2.1 智能优化方法

图 2 统计了在 SBSE 领域发表文献中涉及的各种智能优化算法使用的频率，可以看出遗传算法（GA）是最广泛使用的优化算法，被 300 多篇 SBSE 领域文献的相关研究使用。

本节主要介绍一些广泛使用的智能优化算法，包括遗传算法、爬山算法、模拟退火算法、蚁群算法和粒子群算法等。

2.1.1 遗传算法

在现有的智能优化算法中，遗传算法（Genetic Algorithm，GA）是软件工程领域中使用最为广泛的一种算法^[6]。

遗传算法是一种模拟自然界生物进化过程的启发式搜索算法。这种启发式算法能够产生一个针对问题解的群体并对其进行进化和优化。遗传算法属于演化算法的一种，这类算法通过模拟自然选择的过程对产生的解进行优化。优化操作包括选择、变异和交叉。

在遗传算法中,最优化问题产生的一系列候选解组成了种群,种群的不断演化能够得到更好的解。每个候选解都有不同的特征,演化过程是通过候选解特征间的变异和交叉实现的。

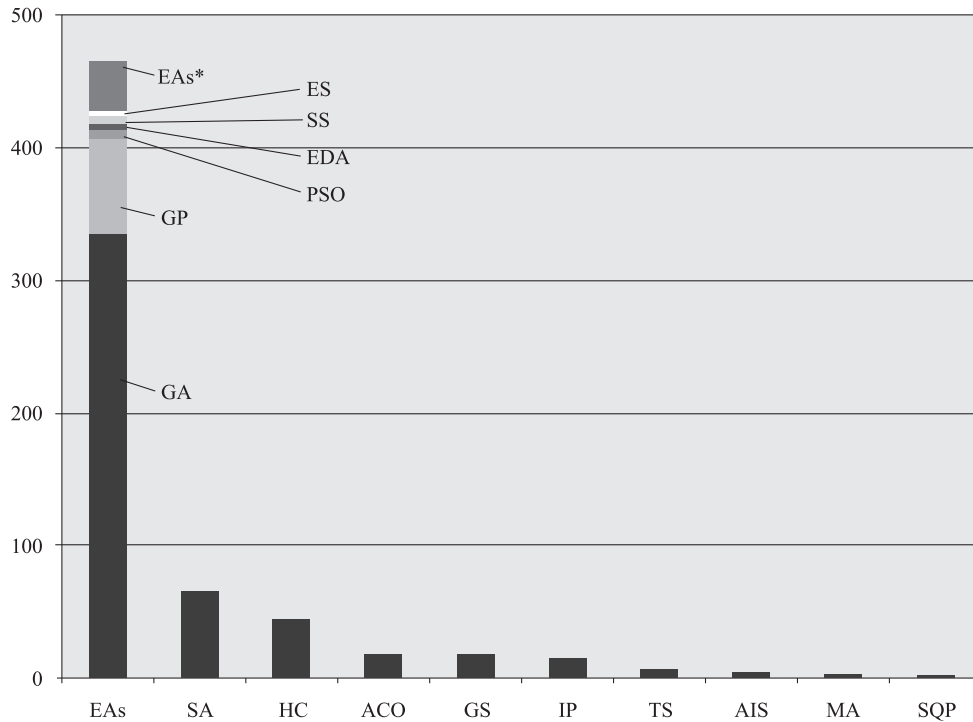


图2 SBSE研究领域使用的主要搜索算法

演化过程通常由一组随机产生的个体构成初始种群,然后对种群不断迭代,每一次迭代过程中的种群被称作一代群体。在每一代中,对群体的每个个体进行评价,度量函数为解的个体的适应度函数。经过选择、交叉、变异等操作,具有较好评估值的个体会被选择至下一代种群中,这样每一代不断优化产生下一代种群。迭代的终止条件是达到迭代次数或者是达到评估函数的目标值。

遗传算法在整个演化过程中有许多变化,关键在于目标函数引导搜索过程的方式、重组和基于种群的搜索过程。另一种独立于遗传算法的演化计算形式为演化策略(evolution strategy),研究者证明其在测试数据生成方面优于遗传算法^[7]。其他遗传算法的变种包括遗传编程(genetic programming)、粒子群优化算法(particle swarm optimization)、进化规划(evolutionary programming)、进化策略(evolutionary solution)等。各种演化算法已成功应用于基于搜索的软件工程中,包括制定能够捕捉软件项目的预测模型^[8,9]和软件测试中的应用^[10]。

2.1.2 爬山算法

爬山算法(Hill Climbing, HC)是一种局部搜索算法。它从问题的某个可行解出发,

通过每次更改解的一个决策变量以寻找更好的解。根据更改决策变量方式的不同,爬山算法可以分为两种,即近邻爬山算法(next ascent hill climbing)和最陡爬山算法(steepest ascent hill climbing)。在近邻爬山算法中,更改首位决策变量以寻找更优解;在最陡爬山算法中,搜索所有决策变量的临近解以寻找更优解。如果对决策变量的更改能够得到一个更好的解,那么就以更改的决策变量进行再次搜索,重复此过程直至解质量无法再提高。

爬山算法从搜索空间中的一个解出发,通过不断迭代,最终可达到一个局部最优解。算法停止时得到的解的质量依赖于算法的初始解的选取、邻域选点的规则和算法的终止条件等。爬山算法作为一个简单高效的搜索算法已广泛应用于基于搜索的软件工程领域中^[11,12]。

2.1.3 模拟退火算法

模拟退火(Simulated Annealing, SA),也叫做蒙特卡罗退火,是源于对热力学中退火过程的模拟。在给定某一温度下,通过缓慢下降温度参数,从而增加退火强度。模拟退火算法可以视为爬山算法的一个变种,通过允许当前解移动到非最优个体来解决爬山算法容易陷入局部最优问题。这种算法首先选取搜索空间中的一个可行解作为搜索起始点,迭代过程中每一步先选择一个邻域点,然后计算从现有位置到达邻域居的概率。

模拟退火算法新解的产生和接受可分为以下步骤:

- 1) 当前解经过简单地变换产生新的解,其中变换包括对构成新解的全部或部分元素进行置换、互换等。
- 2) 计算当前解的目标函数值与新解所对应的目标函数值的差。
- 3) 判断新解是否被接受,判断的依据是一个接受准则,例如最常用的接受准则是Metropolis 准则。
- 4) 当新解被确定接受时,就用新解代替当前解同时修正目标函数值。此时,实现了对当前解的一次迭代。

模拟退火算法具有渐近收敛性,在理论上已被证明它是一种以概率 1 收敛于全局最优解的全局优化算法。该算法已被广泛应用于基于搜索的软件工程的领域中^[12,13,14,15]。

2.1.4 蚁群算法

蚁群算法(Ant Colony Optimization, ACO),又称蚂蚁算法,是一种用来在图中寻找优化路径的机率型算法。

蚁群算法的提出借鉴和吸收了现实世界蚂蚁集体寻径的行为特征。蚂蚁觅食过程中分泌一种信息素的物质,该物质随时间不断挥发。蚂蚁利用信息素作为媒介进行信息沟通,一条路径上留下的信息素浓度的大小与这条路径上通过的蚂蚁数成正比。当通过的蚂蚁越多,留下的信息素越多,导致后来蚂蚁选择该条路径的概率提高,从而建立最短的移动路径。这些规则综合起来具有两个方面的特点:多样性和正反馈。其

中多样性保证了蚂蚁在觅食的过程不会走进死胡同而无限循环；正反馈机制则保证了相对优良的信息能够保存下来。蚁群算法已成功应用于基于搜索软件工程领域的软件测试中^[16~18]。

2.1.5 粒子群算法

粒子群算法 (Particle Swarm Optimization, PSO) 是一种进化计算技术, 是通过模拟鸟群觅食过程中的迁徙和群聚行为而提出的一种基于群体智能的全局随机搜索算法。粒子群算法将群体中的个体看做是在搜索空间中具有质量和体积的粒子, 每个粒子以一定的速度在解空间运动, 并向自身历史最佳位置和邻域历史最佳位置聚集, 实现对候选解的优化。

粒子群算法随机选择一群粒子作为初始种群, 然后通过迭代找到最优解。所有的粒子都有一个适应值, 每个粒子具有运动方向和距离。在每一次迭代过程中, 粒子通过跟踪两个极值来更新自身: 第一个极值是粒子本身所找到的最优解, 这个极值称为个体极值; 第二个极值是整个种群目前找到的最优解, 这个极值称为全局极值。迭代此过程直至达到全局最优解。

粒子群算法通过粒子间的竞争和协作以实现在复杂搜索空间中寻找全局最优解的目的, 它具有易理解、易实现、全局搜索能力强等特点, 已广泛应用于基于搜索的软件工程领域中^[19~21]。

2.1.6 遗传编程

遗传编程 (Genetic Programming, GP) 又称基因编程, 是一种进化计算技术, 它能够从较高层次状态自动解决问题, 自动地进行问题求解, 不需要预知或指定解的形式及结构, 是一种从生物演化过程得到灵感的自动化生成和选择计算机程序来完成用户定义的任务的技术。从理论上讲, 用户使用遗传编程只需要告诉计算机“需要完成什么”, 而不用告诉它“如何去完成”, 最终可能实现真正意义上的人工智能。

遗传编程在 1992 年由美国 John Koza 正式提出^[22], 文中用层次化的结构性语言描述问题, 利用树形结构来表达计算机程序, 其中包括函数集 F (节点) 和终止符集 T (叶子)。函数集 F 包含若干个函数, 终止符集 T 包含若干变量或常量, 利用基本的遗传操作 (如选择和交叉) 产生新个体, 并采用个体适应度来评价个体的优劣。

由于遗传编程所需的计算量非常之大 (处理大量候选的计算机程序), 因此在 20 世纪 90 年代, 人们只能用它来解决一些简单的问题。2000 年以来, 随着遗传编程技术自身的发展和中央处理器计算能力的指数级提升, GP 开始产生了一大批显著的成果, 在多个领域 (如量子计算、电子设计、游戏比赛、排序、搜索等) 均取得了重大发展, 且使用范围也在不断扩展^[23~25]。

2.2 基于搜索的软件工程

图 3 显示了基于搜索的软件工程技术在软件工程生命周期各个阶段发表文章数量的

分布，可以看出超过 50% 的文章是基于搜索的软件测试与调错方向。本章将重点介绍基于搜索的软件测试、需求分析、软件设计、软件重构与维护 and 软件项目开发管理。

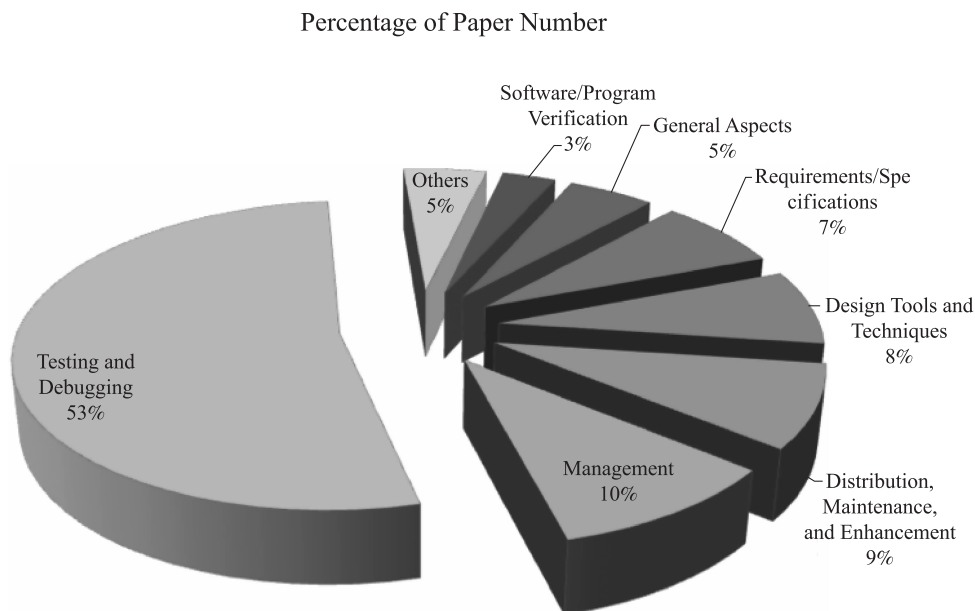


图3 SBSE 在软件工程中研究领域的分布

2.2.1 软件测试

软件测试是软件质量的保证，为了在软件发布之前及时发现和修正软件中可能存在的缺陷，人们提出了很多软件测试方法。例如，结构测试可以检测程序中的各个语句、条件或分支等引发的错误；基于性质的软件测试技术可以有效检测软件系统是否被正确实现；基于组合覆盖的软件测试可以有效检测软件系统中各种因素相互作用引发的故障；变异测试不仅能够检测软件可能潜在的各种错误，还可以对已有的测试质量进行评估，并为进一步测试提供依据。

软件测试的目的是度量和提高软件质量，通过对待测软件（Software Under Test, SUT）及其相关文档、测试标准进行分析，进而设计并执行一系列的测试用例，测试人员往往期望能检测出软件中尽可能多的故障。在理想情况下，为了对测试的有效性和测试后的软件质量有一个较高的信心，软件测试应该是尽可能穷尽整个待测输入空间的，但在实际工程中并没有足够多的资源来执行穷尽测试所需的测试用例。因此，传统意义上的软件测试通常强调测试用例的精心设计，以期达到用最小的代价、最科学的方法，实现对待测试软件最为系统有效的测试。然而，在实践中，人们发现那些精心设计的测试用例不仅不容易自动化，而且有时恰恰是精心地绕过了软件中存在的故障。另一方面，高度自动化的随机测试在很多测试场景下表现得并不逊色，甚至有时比那些理论上很有效的测试技术还更加有效，此外随机测试在可靠性度量和统计估计上也具有天然的优势。但是，纯粹的随机测试却难以实现更深层次的测试，例如要实现语句覆盖、分支覆盖、

路径覆盖等各种覆盖充分性准则测试。

基于搜索的软件测试是基于搜索的软件工程领域的一个重要分支,是一种利用搜索技术来解决软件测试中各项问题的测试方法。从基于搜索的软件工程的角度来看,软件测试的核心过程就是利用各种方法搜索软件中存在的潜在错误的过程,这为搜索技术的应用提供了一个非常好的舞台。目前,基于搜索的软件工程中超过 50% 的研究关注的正是软件测试这一领域,尤其是如何利用搜索技术高效且自动化地为待测系统生成测试数据。McMinn^[26]和 Ali 等人^[27]曾分别对基于搜索的测试数据生成技术和实证研究进行了总结, Afzal 等人^[28]则总结了搜索技术在软件非功能性测试上的应用。最近, McMinn^[29]进一步讨论了基于搜索的软件测试的研究进展和发展趋势。

基于搜索的软件测试最早可以追溯到 1976 年,美国学者 Miller 和 Spooner 提出了一种利用遗传算法生成浮点测试数据的测试方法^[1]。与当时广泛使用的基于符号执行或约束求解的测试数据生成方法不同,这个新方法使用一个评价函数来评估当前测试数据的执行是否接近于测试人员所期望的预期路径,从而使得测试数据的生成问题也就转化为评价函数的优化问题。在遗传算法的优化作用下,那些与预期路径“距离”较远的测试数据将被丢弃,而更加符合预期路径的测试数据将被最终用于测试执行。

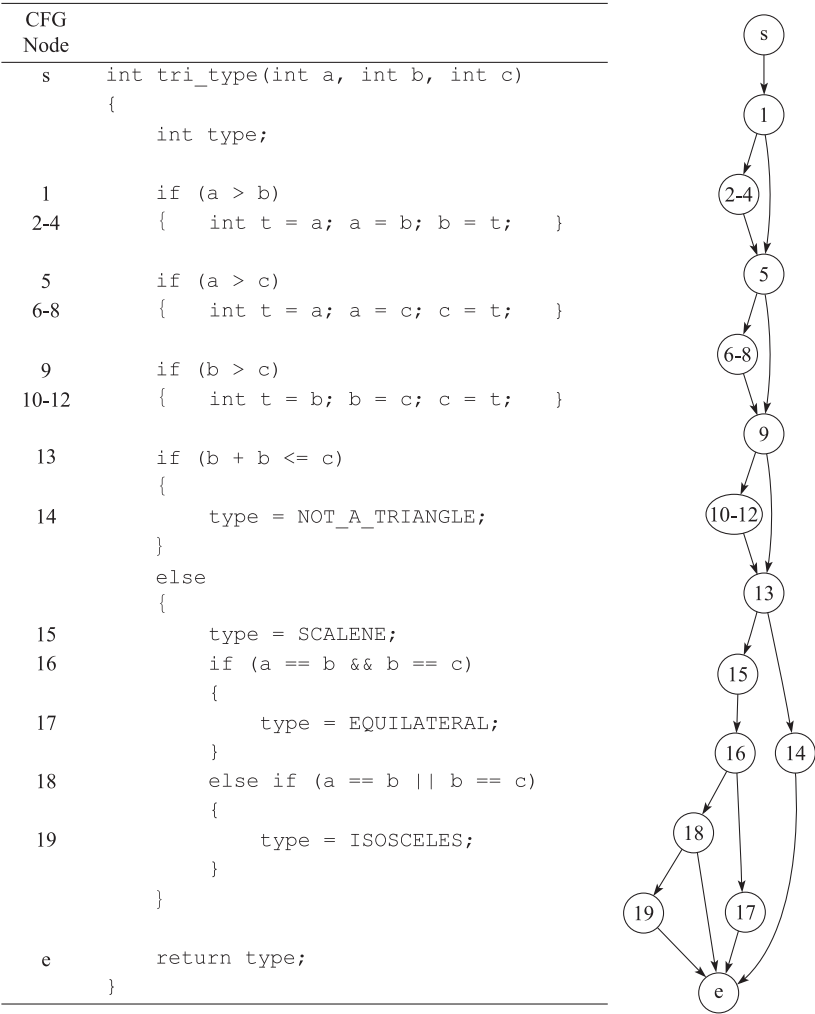
Miller 和 Spooner 并没有在这一领域延续他们的工作,搜索技术也直到 1990 年才由 Korel 应用于测试数据的生成^[31]。随后的 1992 年, Xanthakis 进一步使用遗传算法来生成测试数据^[2]。在此之后,这一研究领域产生了爆炸式的发展,搜索技术被频繁且大范围地应用到软件测试中,成功地解决了诸如功能测试^[32,33]、执行时间测试^[34~40]、集成测试^[41~47]、压力测试^[48~53]、变异测试^[54~62]、组合测试^[63~75]、回归测试^[76~83]等各种测试方法中的各种问题。

在各种测试方法中,测试数据生成是搜索技术在软件测试领域应用最为广泛的场景。例如,在结构测试中,当使用搜索技术生成覆盖特定分支或条件的测试数据时,待测程序将首先被插桩,并执行一些随机产生的测试数据。随后,这些测试数据的质量将依据适应值函数进行评估,在搜索机制的作用下,更加接近测试目标的测试数据将被不断生成,最终达到预期的测试目的。在这一过程中,适应值函数的设计是应用搜索技术求解软件测试问题的关键。分支距离和层接近度,以及两者之间的相互结合^[84]是最为常用的几种方法。

分支距离度量最初由 Korel 使用,其通过改变变量的方法来生成测试数据。在执行测试数据时,如果一条非指定路径被执行,那么就会产生与指定路径间的偏差,这一偏差也称为分支距离,而搜索的目标,就是尽可能地减少这一偏差。例如,我们考虑如图 4 的一个三角形程序^[26],并令预期的测试路径为 $\langle s, 1, 5, 9, 10, 11, 12, 13, 14, e \rangle$ 。如果程序执行了输入 ($a = 10, b = 20, c = 30$),在节点 1 和 5 处将通过相应的预期分支,但在节点 9 处却不能按预期通过正确分支。

为了度量分支距离,我们假定分支的谓词均是“ $a \text{ op } b$ ”的形式,这里 a 和 b 是算术表达式, op 是关系运算符,而相应的目标函数为“ $f_{rel} 0$ ”的形式,表 2 给出了谓词表达式与目标函数之间的关系。当谓词取正确分支时,目标函数取负值,反之则取正值。如

果我们想要执行谓词的正确分支，就需要尽可能降低目标函数的值，对应地，测试数据生成问题也就转化为目标函数的最小化问题。例如，在节点 9 处，针对输入 ($a = 10$, $b = 20$, $c = 30$)，有 $f = c - b = 10$ 。如果我们想要执行该节点处的正确分支，就需要尽可能地设法降低 f 的值。



为了在节点 9 处执行正确分支，可以通过改变变量的方法进行局部搜索，这一搜索过程旨在依次改变每个变量的值，同时保持其他变量值不变。改变变量值的第一个阶段称为探索阶段，通过增加或减少变量的初始值来对变量的邻域进行探索。如果在某个方向上的改变可以降低目标函数值，就可以以此得到某个模式。随后，在下一步模式阶段，就可以根据这个模式进行进一步的相似变换，直到找到目标函数的最小值为止。每个变量都将依据上述过程逐一进行优化，以搜索得到新的测试数据。

在上述例子中，程序执行过程中在节点 9 偏离预期路径，此时改变 a 的值并不会影响目标函数，所以首先选择变量 b 。可以发现，减少 b 的值会使目标函数更差，所以需要增加 b 的值，直到 $b > c$ 。例如，当 $b = 31$ 时新的输入 ($a = 10, b = 31, c = 30$) 就可以实现在节点 9 处按照预期路径执行。但是，这一输入在节点 13 处仍有偏离，此时需再次调用局部搜索。此时，为了在调整输入变量时维持前面的执行路径，新的目标函数将变为 $(a + b) - c$ 。由于减少 b 的值会违反已有的路径约束，而增加 b 的值可以改善目标函数，因此我们最后可以得到测试数据 ($a = 10, b = 40, c = 30$) 以执行预期路径。

然而，上述局部搜索的效果依赖于初始搜索的结果。例如，在图 5 所示的例子中^[26]，如果初始输入选择的是 ($a = 10, b = 10, c = 10$)，控制流将直接进入最后一个节点。此时若想要执行到目标语句，变量 c 的值就需要小于 0，但这样的话就会违反前面路径的执行条件。在这种情况下，局部搜索就失败了。在搜索技术的应用中，如果进入到类似的一个无法提高目标函数的变量值域，就会造成很多无效的搜索和浪费。为了提高搜索效率，可以利用一些来自程序的附加信息，特别是那些影响当前分支节点的变量信息。例如在图 4 中的节点 5 处，由于改变变量 a 和 b 的值可能会改变已成功通过节点 1 处的路径条件，因此搜索变量 c 比搜索变量 a 和 b 更好。

```
void nested_example(int a, int b, int c)
{
    if (a == b)
        if (b == c)
            if (c < 0)
                // target
}
```

图 5 局部搜索失效的例子

除了分支度量外，另一种计算适应值的方法是层接近度。假设当前的执行路径为 t ，预期的执行路径为 t^* ，那么就可以定义层接近度为 $a(t)/|t^*|$ ，其中 $a(t)$ 代表路径 t 没有经过 t^* 中的节点的个数， $|t^*|$ 代表预期路径的节点总数。例如，对于预期路径 $\langle s, 1, 5, 9, 10, 11, 12, 13, 14, e \rangle$ 来说，由于输入 ($a = 10, b = 20, c = 30$) 将执行路径 $\langle s, 1, 5, 9, 13, 14, e \rangle$ ，因此有层接近度为 $2/10 = 0.2$ 。显然，层接近度越小代表执行路径越接近预期情况，而测试数据的生成问题，也同样转化为目标函数的最小值问题。

适应值函数的设计是使用搜索技术生成测试数据的关键，不合适的适应值函数将不能有效地指导搜索的进行。例如，在上述测试数据生成中，对于仅包含一个布尔型变量

(也称为 flag 变量)的分支语句,其真假两个分支将使搜索空间形成两个高原,其中一个非常符合搜索目标,而另一个则非常不符合搜索目标。在这样的情况下,由于缺少相应的梯度信息来指导搜索的进行,搜索技术的应用将变得随机和低效^[85,86]。为了解决这个问题,Harman 等人^[87~89]提出了一种可测试性的转化方法。该方法将通过替换分支条件来为待测软件产生一个适合进行搜索的临时版本,并以此来生成合适的测试数据。对于其他类似的情况,例如基于状态的程序或者字节代码程序等,相应的转化方法^[90~92]也被相继提出以提高软件的可测试性。

在测试数据生成中,除了覆盖程序的某些特定结构外,搜索技术同样可以依据其他不同的测试标准来为不同的测试方法生成测试数据。例如,在功能测试中,搜索技术的一个非常成功的应用是测试 DaimlerChrysler 的停车控制系统的一个早期版本^[32,33]。软件功能测试的目标是检测系统的逻辑行为是否与预期的规格说明相一致,对于停车控制系统来说,它将自动地沿侧方向将汽车移入车位,并在这个过程中依赖不同的传感器来追踪汽车的运动轨迹,从而在汽车与其他实体间距离过近时做出响应。图 6 给出了停车场景的示意图^[32],其中 P0 至 P5 六个点划分了停车系统可以驾驶和碰撞区域的边界,而车辆与边界的初始距离 distance to space 以及两个角度变量 gap 和 psi 共同构成了搜索空间的候选解。由于测试的目标是尽可能多地发现该系统中的故障,因此一个更容易使停车系统产生故障的候选解将被分配更高的适应值。在适应值的计算上,Wegener 等人提出的一种方法是衡量停车过程中汽车轨迹与碰撞边界间的最短距离,另一种方法则是衡量汽车轨迹与碰撞边界间的面积。这项研究大约模拟了 900 个停车场景,并从中发现了 25 个会导致碰撞的情况,其中典型的情形包括车位与汽车距离过远或者汽车初始位置过于接近边界。

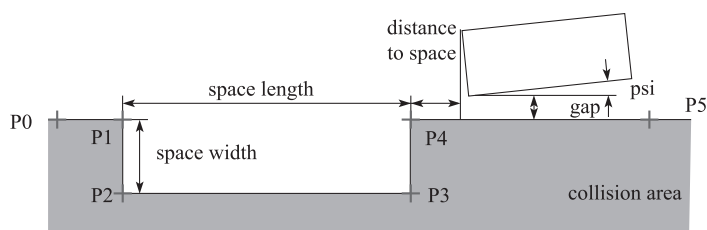


图 6 停车场景的示意图

除了逻辑功能外,软件系统的正确性还依赖于系统的时间属性,尤其在实时系统中,输出产生的过早或过晚都会在某程度上导致系统的故障。为了确保系统满足指定的时间约束条件,在执行时间测试中,测试人员需要生成测试数据来估计系统的最长执行时间(WCET)和最短执行时间(BCET)。然而,软件系统的复杂性使得这一时间很难估计,并且执行时间还与当前的硬件环境息息相关。传统的静态分析方法需要对系统的可能路径进行分析,从而对时间行为进行建模。但这一过程依赖于程序员的辅助信息,并且容易高估最坏执行时间以及低估最好执行时间。为了对系统的真实执行时间进行更准确的估计,Wegener 等人^[34~36]使用遗传算法来进行测试数据生成。其中,搜索的适应值

函数是系统的执行时间，而搜索的目标是尽可能地最大化系统的最长执行时间，或最小化系统的最短执行时间。上述研究讨论了一个绘图程序的控制流图，图 7 给出了其最短执行时间和最长执行时间对应的路径。该图中包括循环在内的所有可能的执行路径构成了整个搜索空间，传统的路径覆盖策略或者随机测试都不能很好地找到满足条件的测试数据，而搜索技术在这一问题上则表现得十分有效。此外，Tracey 等人^[37,38]比较了模拟退火和遗传算法在 WCET 测试数据生成上的性能，结论显示遗传算法要比其他算法更加高效。

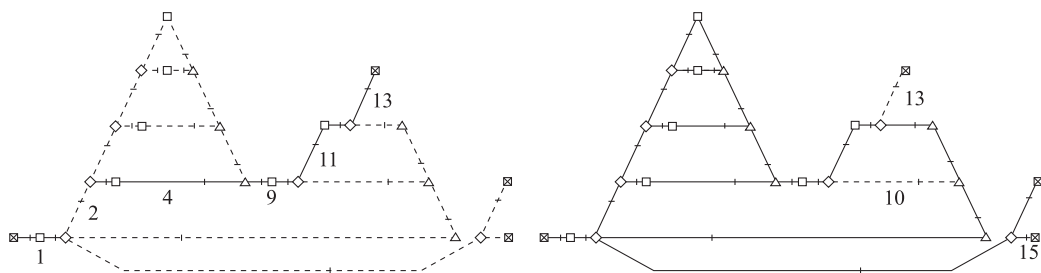


图 7 最短和最长执行时间对应的控制流图

搜索技术也可以为压力测试生成测试数据。压力测试的目标是在持续不断地给待测系统增加压力的情形下，确定系统所能承受的最大压力，相关的测试目标包括最大处理的信息量、最大存储范围以及最大持续时间等。例如，对于实时系统来说，一个触发的事件必须在规定的截止时间内执行完成。因此，在压力测试中，测试人员需要尽可能地构造使系统总是在接近截止时间才能完成任务的测试数据。然而，当系统中存在多种具有不同优先级的周期性或非周期性事件时，测试人员往往难于手工构造合适的测试数据，因此 Briand 等人^[48]尝试使用遗传算法来处理这一问题。假设在测试时间 T 内每个事件 A_i 将被执行 k_i 次，其第 j 次执行的到达时间为 $a_{i,j}$ ，执行的结束时间为 $e_{i,j}$ ，且截止时间为 $d_{i,j}$ ，则测试数据生成的目标是尽可能地减小截止时间和实际结束时间之间的差值 $e_{i,j} - d_{i,j}$ 。这里，遗传算法的每个候选解将包含 $\sum_{i=1}^n [T/\min]$ 个基因，而每个基因都使用二元组 $(A_i, a_{i,j})$ 来表示，其中 n 为事件的数量， \min 为最短事件到达间隔。适应值函数将被用于衡量每个候选解的质量，即事件能在截止时间前完成时，候选解将被赋予一个较小的适应值，而一个较大的值则代表一个质量更高的解。Briand 等人^[50]对几个实例进行了研究，发现即使对于理论上可以准确调度的事件，遗传算法也能发现一些会导致系统无法正常执行的情况，因此其建议使用搜索技术来在设计早期对系统进行验证，并在理论上能正确调度的情况下进一步对关键部分进行测试。类似地，Garousi 等人^[51-53]考虑了分布式实时系统的压力测试，并使用搜索技术来生成达到最大压力的测试数据。此外，Del Grosso 等人^[49]考虑了缓存泄露这一安全相关问题，并使用搜索技术来生成压力测试数据。

在变异测试中，测试人员使用某些变异算子来向原程序中植入一些错误并通过测试来检测这些错误是否能被检出。其中，被植入错误的程序也称作变异体，而当测试用例

集中的某条测试用例检测出原程序和变异体间的不同时,我们就称该测试用例杀死了一个变异体,否则称之为变异体存活。通过分析为原程序设计的测试用例集所能杀死和最终存活的变异体数量,我们就能对原测试用例集的测试充分程度进行评估,如果一个测试用例集能杀死所有植入的变异体,那么这个测试用例集就有能力检测出软件中所有可能潜在的错误^[61]。然而,对每一个变异体都执行一遍测试用例集是一件开销很大的工作,同时还存在一些任何测试用例都无法进行区分的等价变异体,为此人们提出了很多方法来改进变异测试的过程。其中,与搜索技术相关的一项工作是 Adamopoulos 等人^[93]提出的基于遗传算法的变异体和测试用例协同演化策略。在他们的方法中,为了从一系列变异体中选择一个规模较小且较难杀死的变异体子集,首先需要对每个变异体在 0.0 和 1.0 的范围内进行评分,其中一个较高的评分代表该变异体较难杀死。随后,一些随机选择的变异体子集将作为演化的初始候选解,当某个变异体集合 S 中任意一个变异体的

评分都不为 1.0 时,则适应值函数为 $f(S) = \sum_{i=1}^n S_i / S$, 其中 S_i 代表变异体 i 的评分;

否则该候选解 S 的适应值为 0。显然,当一个变异体的评分为 1.0 时,该变异体无法被测试用例集中的任一条杀死,因此这样的适应值定义方法能在一定程度上避免 S 中包含等价变异体。类似地,在测试用例选择上,如果一条测试用例能杀死更多的变异体,则该测试用例将被赋予较高的评分,而遗传算法将用于从候选测试用例集中选择一个规模尽可能少且评分尽可能高的测试用例子集。上述两个过程将同时并行地演化,从而不断改进变异体和测试用例集合,最终产生一个较难杀死且不含等价变异体的变异体集合,以及一个能尽可能多地杀死变异体的高质量测试用例集。这里,如果我们分开考虑测试数据和变异体的生成,诸如文献[54~56]等研究尝试使用搜索技术生成能杀死更多变异体的测试数据来提高测试用例集的质量,而 Dominguez-Jimenez 等人^[60]则用搜索技术约简所需变异体的数目来减小变异测试的开销。此外,在变异测试中运用多次变异算子产生的高阶变异体更能模拟真实的软件错误,但传统研究认为构造这样的变异体需要巨大的开销。基于此,Harman 等人^[57~59]提出了基于搜索的技术来为变异测试生成合适的一阶或高阶变异体,Omar 等人^[62]则将类似的方法应用到 Java 程序测试中。

在组合测试中,我们假设系统的故障是由某几个参数间的交互作用所引起的。因此,对于一个包含 n 个参数且每个参数有 l_i 个不同取值的待测系统,测试数据生成的任务即生成一个测试用例集,使得任意 t 个参数间所有可能的组合都在该测试用例集中至少出现一次,而这样的测试用例集也被称作 t -way 覆盖表。例如,图 8 给出了一个包含 4 个参数,且每个参数都有 2 个取值 $\{0, 1\}$ 的 2-way 覆盖表。在覆盖表生成上,当参数和对应取值个数满足某些特定的要求时,一种称为正交表的数学结构可以被快速地构造。在正交表中,任意 t 个参数间的每个可能组合都恰好出现一次,因而该测试用例集具有理论上最小的规模。然而,对于任意的 n 和 l_i ,我们并不知道满足覆盖要求的最小测试用例集大小,因而研究者们提出了很多方法来尝试生成规模尽可能小的覆盖表^[94]。在这一问题上,搜索技术同样

f_1	f_2	f_3	f_4
0	0	0	0
0	1	0	0
1	0	0	0
1	1	1	0
1	1	0	1
0	0	1	1

图 8 一个组合测试覆盖表

表现出了巨大的潜力。例如,在Cohen等人^[63]的研究中,测试人员首先指定一个规模 N 并随机初始化一个 $N \times n$ 的候选数组,这里的适应值定义为该数组中未覆盖的组合个数。随后,搜索技术将不断优化这个候选数组以尽可能地降低其适应值,如果能找到一个适应值为0的候选数组,则该数据即为所求覆盖表;否则就递增 N 的值并进行新一轮的搜索。在这一框架下,类似的研究还包括文献^[72~74]等。此外,Bryce等人^[65,67,68]则使用每次生成一条测试用例的方法构造覆盖表。其中,测试用例是搜索的候选解,而适应值定义为其所能覆盖的新的组合个数,因此搜索技术将用于在每次迭代中生成一条能覆盖最多组合的测试用例,直到覆盖所有组合为止。在这一框架下,类似的研究还包括文献^[64,71,73,75]等。从目前的研究来看,搜索技术在覆盖表生成上尽管需要花费比贪心搜索更长的执行时间,但其能极大地减少所需测试用例的数目。此外,由于待测系统中某些参数间的取值组合并不能同时出现,搜索技术同样能在生成时高效地避免生成具有约束条件的测试用例^[69,70]。目前,在组合测试数据的生成领域,搜索技术是主流应用的方法。

除了生成测试数据外,搜索技术同样被成功地应用到其他的软件测试活动中。例如,在集成测试中,测试人员需要依据各构件间的依赖关系来确定构件被集成的顺序。基于层次结构图,传统的集成方法包括自底向上集成(即先集成叶子节点所代表的构件),或自顶向下集成(即从高层逐步向下集成)。对于后一种方法来说,测试人员需要开发桩程序来模拟底层尚未集成模块的对应功能。然而,很多软件构件间会存在依赖关系,从而使得层次结构图中包含一些环,这样传统的集成方法就不能很好地确定合适的集成顺序。因此,集成测试人员需要找到一种合适的构件集成顺序,从而能尽可能地减少所需的桩程序数目,以及尽可能地减少集成所需的步骤或时间,而这两个问题均已知为NP完全问题。对于面向对象软件,Hanh等人^[41]使用遗传算法来解决前一个维度的问题,即最小化所需的桩程序数目。在他们的方法中,某个集成顺序即对应为一个候选解,例如按顺序{B,D,F,A,H}集成五个构件,对应的适应值就等于在集成过程中所需的桩程序数。随后,通过求解适应值函数的最小化问题,我们就可以找到优化的集成顺序。Hanh等人在6个真实的实例上比较了包括确定性分析等不同集成方法的性能,验证了搜索技术的有效性。此外,Briand等人^[42,43]进一步将类间的耦合度纳入集成顺序的考虑因素,并使用遗传算法来寻找面向对象软件的最优集成顺序;da Veiga Cabral等人^[44]在此基础上使用多目标蚁群算法来对不同的排序标准进行权衡;Colanzi等人^[45,47]和DelamareK等人^[46]则使用搜索算法来解决面向方面的软件的集成顺序问题。

对一个给定的序列按某种准则进行排序是搜索技术应用的典型场景,除了确定集成测试的构件集成顺序外,另一个类似的应用是回归测试中的测试用例集排序^[79,95~106]。在软件开发过程中,当软件产生更改时,测试人员就需要进行回归测试来确保新的更改并不会对原有软件的功能造成不良的影响。然而,随着软件功能的不断开发,回归测试用例集的规模也将逐步增长,而可供测试的资源往往又十分有限,因此如何从中选择合适测试用例进行执行,以及如何安排测试用例的执行顺序来更快地检测出软件中的故障,是回归测试中需要考虑的关键问题。对于测试用例排序来说,给定一个测试用例集 T 以

及评价函数 f ，我们期望找到 T 的某个执行顺序 P ，使得对任意的 $P' \neq P$ 都有 $f(P) \geq f(P')$ 。例如，表3给出了四条测试用例的语句覆盖情况^[79]，当按 $\langle A, B, C, D \rangle$ 的顺序来执行时，测试完前两条测试用例后仅覆盖了其中6条语句。而如果我们按 $\langle C, D, A, B \rangle$ 的顺序来执行，测试完前两条测试后所有8条语句都得到了覆盖。在排序问题上，由于某个执行顺序 P 可以很容易地编码为适合搜索技术应用的形式（例如字符串），排序目标也可以很容易地转换为适应值函数，因此搜索技术非常适合于解决排序问题。目前，基于覆盖率、测试执行时间或者测试需求等排序场景中都有搜索技术的应用。此外，在回归测试中，如何从测试用例集中选择一个子集，从而对软件更改的部分进行测试也是搜索技术应用的一个场景^[76-78]，另一个回归测试的应用场景则是对已有测试用例中冗余的部分进行约简，从而减少所需测试用例的规模^[82]。在排序算法的研究中，Li等人^[79]在2007年首次比较了贪心算法、额外贪心算法、遗传算法等五种不同的搜索算法在测试用例优先排序方面的效率差异，随后又继续开展了基于多目标的启发式搜索算法在测试用例优先排序上的研究^[104]，使用了基于非支配排序的遗传算法改进算法（Non-dominated Sorting Genetic Algorithm II, NSGA-II），并结合异构并行技术GPGPU（General-Purpose computation on GPU）技术，提出了粗粒度和细粒度两种并行计算策略，极大地提高了搜索算法的效率，为测试用例优先排序技术在产业界的实际应用提供了支持。

表3 测试用例的语句覆盖情况

测试用例	语句覆盖							
	1	2	3	4	5	6	7	8
A	×	×	×			×	×	×
B	×	×	×				×	×
C	×	×	×	×				
D					×	×	×	×

此外，测试用例的预期输出也是软件测试中的一个关键问题。McMinn曾提出一种基于搜索的测试预期生成方法来对浮点数进行测试^[81]。在这一方法中，为了检测原始程序输出的正确性，可以应用一种类似 N -版本编程的方法来生成待测程序的另一个版本，进而通过比较两者间输出的差异来判断测试结果是否满足预期的测试输出。然而，大多数情况下测试用例的预期输出还是需要人工来进行确定，但搜索技术仍可以在这一问题上减少测试人员的人工成本开销。从数量上看，合适的适应值函数可以使搜索在最大化测试目标的前提下最小化所需执行的测试用例数目^[108,109]，从而减少需要人工评估的测试结果。而从质量上看，搜索技术也可以通过增强测试场景的可辨别性来帮助测试人员更加容易地判断所执行的测试数据是否通过测试^[110]。

总之，面对规模日益庞大和复杂的软件，基于搜索的软件测试避免了传统测试和分析方法的不足，能在尽可能降低测试成本的前提下尽可能高效地完成不同的测试任务，是软件测试领域中一种高效实用的新方法。

2.2.2 程序错误自动修复与错误定位

程序错误自动修复技术综合利用错误定位、修复规则以及补丁验证技术对可复现的

错误进行修复。20 世纪 80 年代软件生产自动化被广泛关注，但软件维护阶段的自动化技术受阻于程序逻辑的复杂性一直没能达到工业级应用要求。

随着 21 世纪软件资源挖掘和搜索技术的流行，自 2009 年以来以测试为基础的逻辑无关或弱逻辑的软件错误自动修复技术开始出现工业级应用的前景。（图 9 给出了软件错误自动修复技术框架）。随后，研究者发现了程序错误自动修复中量变和质变现象^[112]，利用面向自动修复的错误定位^[113]，提高了定位的准确度，推动了软件维护自动化进程。2012 年，针对百万行软件错误修复代价高昂的问题，研究者从软件自动修复三阶段（即错误定位、补丁生成和补丁验证阶段）入手，建立了大型开源软件历史错误分析数据集和错误定位方法有效性分析比较平台，进行错误定位有效性分析，形成错误定位方法的改进完善实验床^[114,115]；分析程序错误修复过程中可能的复用信息和过程，在补丁验证中引入基于编译现场缓存的弱编译技术，提升了验证过程中占据主要耗时的编译的效率^[114,115]；在补丁验证的回归测试用例集合中，引入测试用例动态序概念，加快了无效补丁的发现效率，降低了验证时间^[116]。

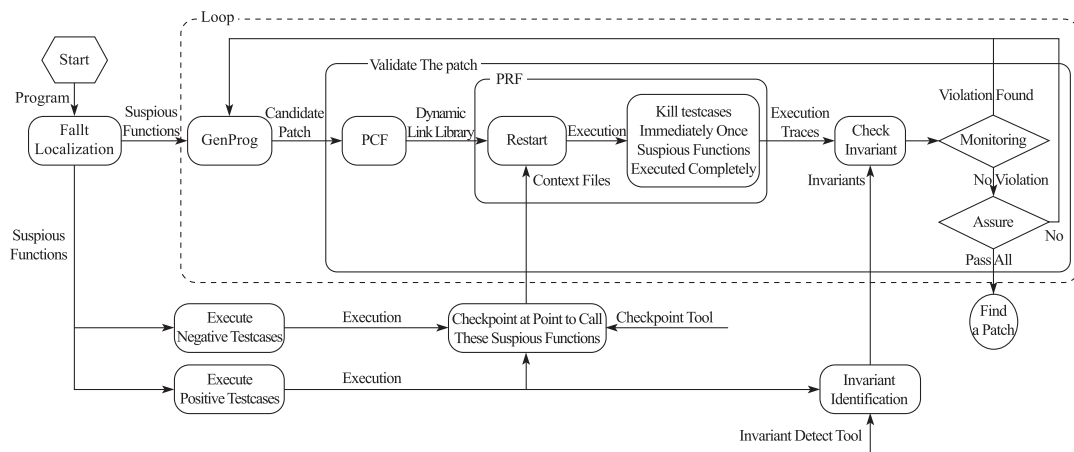


图 9 软件错误自动修复技术框架

错误定位通过对程序失效或者异常行为进行分析，寻找导致失效或者异常的根源所在，即定位到错误在程序中的位置。软件错误定位技术的先进性主要体现在错误定位的准确性和实用性。现有的错误定位技术主要关注于两个方面：一是如何找出尽量小且包含缺陷语句的可疑语句集合；二是如何赋予缺陷语句更高的可疑值，提升缺陷语句在可疑语句排序表的位置。主要定位技术涉及基于切片、基于统计、基于程序状态、基于机器学习等四个方面。

研究者通过构造面向错误定位的轻量级近似动态后向切片算法，提出了程序逻辑与统计相结合的软件错误定位方法 SSFL，构造出基于程序逻辑和差异信息的度量公式，通过控制流依赖和数据流依赖分析，融合差异性比对获得的统计信息，提高了定位准确度^[117,118]。图 9 即为软件错误自动修复技术框架示意图。

2.2.3 需求分析

需求分析作为软件生命周期的第一个阶段，并贯穿于整个软件生命周期。随着工程规模的扩大，需求工程的重要性越来越突出，因为在初期阶段的频繁和细微的变化都会对后期带来大量的修改工作。如何在工程初期做好需求分析将直接决定整个工程的成败。

事实上，需求工程常被划分为需求开发和需求管理两部分贯穿于软件工程的整个生命周期之中。随着软件规模的日益增长和复杂性增加，每个客户都有自己的优先需求目标（见图 10），不同的需求之间又通常会存在利益冲突。此时，问题就是一个典型的 NP 问题。为了在现有资源中权衡满足每个用户的需求做出最优的选择，需求工程师们选择启发式优化算法解决复杂的、多目标的、条件约束的需求分析问题。

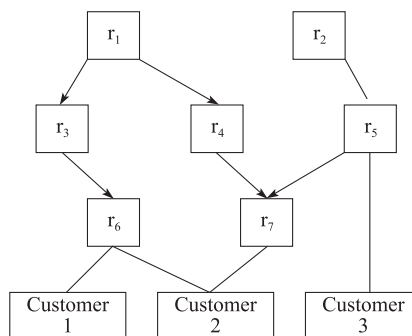


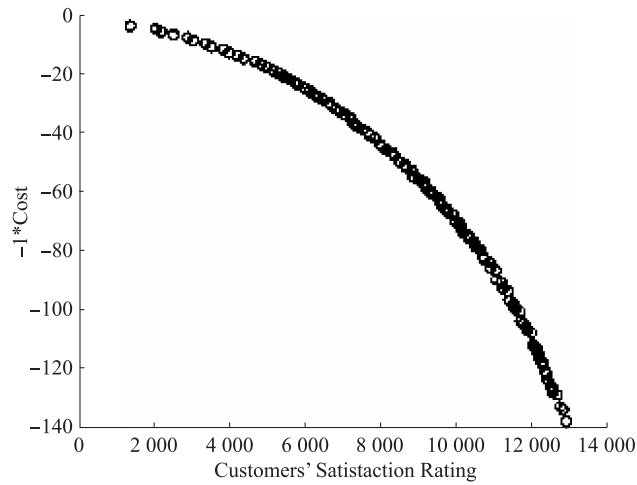
图 10 客户需求结构

在需求工程中，以下一版本问题为例（Next Release Problem, NRP）^[120]，通常采用启发式算法解决探索成本消耗和利润之间的平衡关系。在 NRP 中，唯一的目的是挑选出最优的需求解，在满足约束条件下使客户满意度达到最大，最大化客户利润的同时最小化成本花销。优化结果能给决策者在需求选择的问题上提供充足的理论支撑和证据。Bagnall 等人提出 NRP 模型解决单目标优化问题。假定有一个软件项目，存在客户群 $C = \{c_1, \dots, c_m\}$ ；提出了一系列软件需求表示为 $R = \{r_1, \dots, r_n\}$ ；相对应需求项目消耗 $\text{Cost} = \{\text{cost}_1, \dots, \text{cost}_n\}$ ；每个客户对公司的重要性不同，所占权值表示为 $\text{Weight}_i = \{w_1, \dots, w_m\}$ ($w_j \in [0, 1]$ 且 $\sum_{j=1}^m w_j = 1$)；将评价需求重要性的分数值表示为 $\text{Score}_i = \sum_{j=1}^m w_j * \text{value}(r_i, c_j)$ ；决策向量 $x = \{x_1, \dots, x_n\} \in \{0, 1\}$ ，当 $x_i = 1$ ，表示需求 i 将选入下一优化版本中，否则 $x_i = 0$ 。

随着解决需求的复杂化，传统的单目标优化问题逐步发展成为多目标优化问题（Multi-Objective Next Release Problem, MONRP）^[121,122]。基于多目标的帕累托最优优化策略（Multi-objective Pareto Optimal），通过启发式算法挑选出不受其他变量支配的非支配子集（non-domination population），每一个非支配集合表示了一种可能的资源分配，并且在已有资源限制条件下使客户满意度达到最优。需求目标的适应度计算公式如下^[123]：

$$\text{Maximize } \sum_{i=1}^n \text{score}_i * x_i \text{ Minimize } \sum_{i=1}^n \text{cost}_i * x_i$$

Zhang 等人^[121]使用 Motorola 需求数据集，采用了 NSGA-II 算法进行了多目标需求优化的实验。如图 11 中所示的帕累托前沿（Pareto Front），每个圆圈表示当需求消耗不变时所能构成的最优解决方案。实验证明随着客户满意度的增加，执行客户需求的消耗值也随之增加。

图 11 基于 Motorola 数据集的实验结果^[121]

随之，越来越多的基于多目标的优化算法应用到需求分析中^[124~132]。Tonella 等人提出了一种交互式的遗传算法优化需求顺序^[125,126]，Kumari 等人采用了精英量子演化算法 (Quantum-inspired Elitist Multi-objective Evolutionary Algorithm, QEMEA)^[127]，Jifeng 等人使用了骨架分析与启发式算法^[128]等。此外，还包括多目标的混合式量子差异进化算法 (Multi-objective Quantum-inspired Hybrid Differential Evolution, MQHDE)^[129]、双归档算法 (Two-Archive algorithm)^[131]、聚类的方法^[132]等应用到多目标到需求选择优化问题。Antônio Mauricio Pitangueira 等人^[133]分析了这一领域的 30 多篇文献，系统地评价了不同的优化算法，指出由于不同的方法具有不同的优缺点，研究者们可以根据问题的特征选择较优的算法解决需求优化问题。

在众多的算法研究中，基于搜索的需求工程同时还研究需求分析中的公平性分析^[134]和敏感性分析^[136]等。

(1) 公平性分析 (Fairness Analysis)

公平性分析的动机在于平衡客户的需求执行度，尽可能满足每个人的首要需求。然而，不同的人对待公平的定义有不同的标准，Finkelstein 等人^[135]首次在不同的公平定义中寻求公平在不同人标准的平衡，采用了基于搜索的技术揭露了客户需求间的依赖关系。非支配排序的遗传算法 (Non-dominated Sorting Genetic Algorithm-II, NSGA-II) 是目前最流行的多目标进化算法之一，它降低了非劣排序遗传算法的复杂性，具有运行速度快，解集的收敛性好的优点，成为其他多目标优化算法性能的基准。在需求工程中的公平性分析问题中可以看成是多目标优化问题。

Finkelstein 等人分别使用不同的数据集 (Motorola data set, Greer data set, Random data set) 进行测试^[136]，实验发现执行的需求的数量与公平性成反比。当完成的需求越多，客户之间的公平性就逐渐降低。总的来说，基于搜索技术将逐步应用在现实生活中，揭示出暗藏在数据集中的潜在关系。但是，需求工程作为新兴学科方向，还有较大的探索空间等着我们去发掘。

(2) 敏感性分析 (Sensitivity Analysis)

敏感性分析是研究与分析一个系统（或模型）的状态或输出变化对系统参数或周围条件变化的敏感程度的方法。在最优化方法中经常利用灵敏度分析来研究原始数据不准确或发生变化时最优解的稳定性。通常，当一个系统变得越来越复杂，就越来越难获取输入输出之间的关系。灵敏度分析可以基于参数输入和观察到的输出结果分析出它们之间的关系，通过灵敏度分析还可以决定哪些参数对系统或模型有较大的影响^[136,137]。

2.2.4 软件设计

在软件开发流程中，软件设计是从需求分析向软件具体设计进行转化的一个步骤。通常来说，设计人员首先需要依据规格说明来确定软件的一些基本结构，随后通过进一步分析并进行相应地修改，设计人员就可以确定软件的最终设计。除此之外，对软件质量进行增强和预测的相关研究，也通常被视为软件设计研究的组成部分。近年来，基于搜索的软件设计也发展得较为迅速，从软件的高层体系结构设计到软件聚类 and 软件重构，搜索技术在很多方面都取得了成功的应用。Raiha^[138]曾对搜索技术在软件设计领域的应用进行了总结，尤其强调了在不同问题下的编码和适应值函数的选择问题，并对已有方法的性能进行了评价和讨论。

软件的体系结构设计是软件设计的核心，这一工作需要设计人员有丰富的经验来从高层的需求中确定具体的细节设计。例如，在面向对象的软件设计中，设计人员首先需要从使用案例中抽取信息来确定每个类的方法和属性，并在此基础上确定接口和继承关系，从而得到类似 UML 类图的表现形式。其中，使用案例是用户为了达到某种目的而执行的一系列时序步骤，描述了用户与系统间的交互场景。使用案例通常以文本形式来记录相关操作和对应数据，例如“系统扣除某账户的若干余额”就描述了系统中的某一具体行为。从使用案例中可以确定系统所包含的行为和数据，然后通过为每个类分配合适的方法和属性，就可以得到具体的一种软件设计，而所有可能的方法和属性的集合就构成了面向对象设计问题的求解空间。图 12 给出了面向对象软件底层设计的流程^[138]。

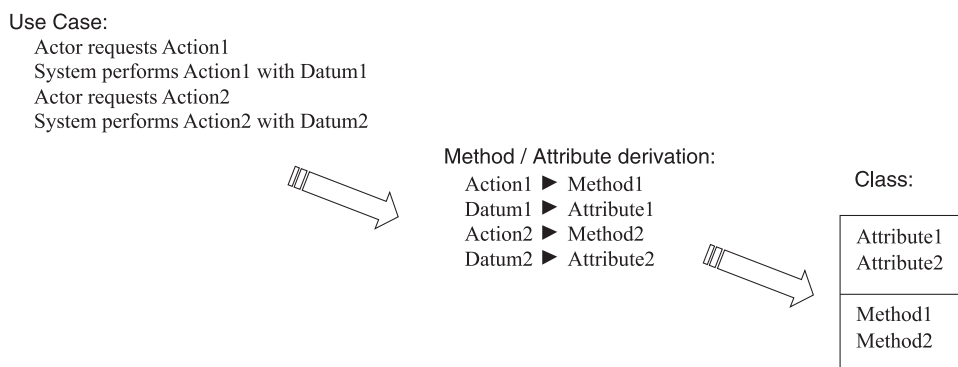


图 12 面向对象软件底层设计的流程

由设计人员人工来确定最优的面向对象软件设计往往是很困难的,而搜索技术不仅能产生与人工设计质量相当的结果,并且能在一定程度上生成许多人工设计未曾考虑过的更优结果,因而是这一领域的一种较有潜力的方法。当应用搜索技术时,一个候选类图解可以被编码为一个整形数组,其中某一位代表该系统的一种方法或属性,而对应的整数值代表该方法或属性被分配到的类。这里的适应值函数通常是基于方法或属性的内聚度 (cohesion) 或耦合度 (coupling) 来进行设计。例如,对于类 C 来说,其方法的内聚度 (COM) 定义为 $f(c) = 1 / (|Ac| + |Mc|) * \sum \Delta_{ij}$, 其中 Ac 和 Mc 分别代表了类中属性和方法的个数,且当方法 i 使用了属性 j 时有 $\Delta_{ij} = 1$, 否则 $\Delta_{ij} = 0$ 。此外,由于对软件设计的某种性质进行优化往往会削弱其他性质,因此这一领域所使用的方法大多是基于多目标优化的搜索策略。

例如, Simons 和 Parmee^[139~141] 使用多目标优化的遗传算法来从使用案例设计面向对象软件,他们所使用的适应值函数包括三个部分:类的内聚度、类间的耦合度以及类的数量。类似地, O'Keeffe 和 O'Cinneide^[142,143] 使用模拟退火方法来求解这一问题,并对一系列的适应值函数进行了比较。Bowman 等人^[144] 则针对已有软件设计进行优化的问题,提出了基于多目标遗传算法的决策支持系统来优化类中方法和属性的分配。此外,上述设计方法主要考虑的是面向对象的底层设计,搜索技术同样可以进一步用于软件体系结构或设计模式的实现。为了提高软件的可重用性, Amoui 等人^[145] 使用遗传算法来寻找最佳的模式实现顺序,从而构建设计模式的最优转化步骤。而通过将上述研究工作与设计模式的相关研究进行结合, Raiha 等人^[146,147] 提出了基于遗传算法的搜索技术来自动化地对包含多种设计模式的软件体系结构进行合成。

在面向对象软件设计中,搜索技术的另一个热门应用是软件行为模型的设计。高可信软件通常需要依赖自动系统对外界计算资源、组件和物理环境的变化做出合适的响应,然而自动系统的行为在部署前往往很难预测,因此自动化地生成软件行为模型能极大地帮助软件工程师来理解系统的行为。在这一问题上, Goldsby 等人^[148~150] 开发了基于搜索的软件行为模型生成工具 Avida-MDE。这一演化工具使用描述系统行为的 UML 状态图作为候选解,并通过选择和变异操作来不断演化。其中,每个个体模型的适应值将依据开发人员所定义的一个任务集合来评估,能成功执行的每个任务都将增加候选模型的适应度。类似地, Lucas 和 Reynolds^[151] 同样使用搜索技术来对确定性状态机进行学习,从而更好地对软件系统的状态标签进行分配。

搜索技术同样可以用于面向服务的软件设计。通过将应用程序的功能作为服务发送给最终用户或者其他服务,面向服务的体系结构可以非常灵活地结合不同的服务以提供一些复杂的功能。其中,服务的后绑定机制是面向服务软件的核心,这就使得面向服务系统可在运行时选择所需的具体服务。例如,在图 13 所示的例子中^[152], 圆形所代表的是设计中需要使用的服务 (抽象服务),而正方形所代表的是具体实现该功能的可选服务 (具体服务)。与同一抽象服务相关的具体服务在功能上是等价的,因此人们通常按照服务质量 (Quality of Service, QoS) 这一非功能性标准来在其中进行选择,例如考虑服务的价格、响应时间、可用性以及名声和用户满意度等指标。

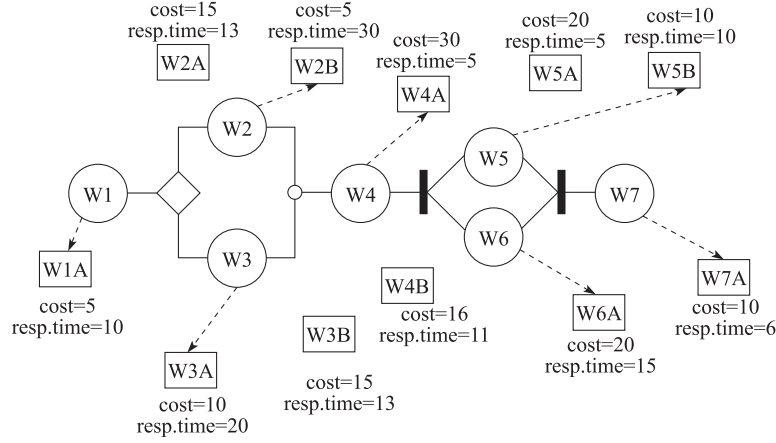


图 13 抽象服务与具体服务间的绑定示例

给定需要的抽象服务和可选的具体服务，如何进行对应的绑定以使得系统在满足一定约束条件的前提下尽可能地优化某些特定目标，是面向服务软件中的关键问题。这一问题也称作 QoS 的服务组合问题，而其最优解的寻找已被证明是 NP-hard 的。线性整数规划可以用于寻找该问题的解，但其求解时间并不能满足当前系统的快速构建需求。基于此，Canfora 等人^[152]提出了一种优化服务选择的遗传算法。在他们的方法中，一个组合服务的候选解 S 用包含 n 个服务的整型数组 $[s_1, s_2, \dots, s_n]$ 来表示，其中 s_i 代表与第 i 个抽象服务进行绑定的具体服务。由于需要考虑的 QoS 指标包括时间、开销、可用性和可靠性，因此适应值函数定义为：

$$f = (w_1 \text{Cost} + w_2 \text{Time}) / w_3 \text{Availability} + w_4 \text{Reliability}) + w_5 D$$

其中， w 代表了不同指标的权重， D 则代表了候选解偏离满足约束条件的距离，从而对无效的候选解进行惩罚。虽然上述适应值的计算看起来较为简单，但其中 Cost 和 Time 尤其是 Availability 和 Reliability 等指标的衡量往往需要复杂的计算，并且需要针对具体系统进行具体设计。显然，一个较小的适应值代表该候选组合的质量越高。除了考虑对新的服务进行组合优化外，Canfora 等人^[153]随后继续将遗传算法用于服务运行时的重绑定问题，从而使得原先确定的服务在服务质量与预测值产生偏差时进行服务的重新组合。

此外，Jaeger 和 Muhl^[154]考虑了在不同的 QoS 特征下的服务组合问题，并使用遗传算法来进行优化。Khoshgoftaar 等人^[155,156]则使用多目标优化的搜索技术来平衡不同的 QoS 服务。在算法设计上，通过对遗传算法的编码、初始化方法和演化机制进行改进，Zhang 等人^[157~160]提高了遗传算法在优化服务选择上的性能，从而更高效地解决了约束条件下的 QoS 服务选择问题。而与传统的多目标优化不同，Cao 等人^[161]指出很多商业流程中开销是需要首要考虑的因素，并使用单目标优化的遗传算法来优化服务选择流程。

软件设计中的构件选择问题同样可以用搜索技术来优化。由于软件开发大多是迭代进行的，基于构件的方法可以不断地在当前状态下为软件加入新的构件以使得软件不断演化。然而，为软件的下一个版本选择构件并不是一项简单的工作，项目管理人员通常需要对开发开销、用户期望、开发时间、预期回报等指标进行综合考虑来做出最合理的

选择。同时,现实中这些指标间往往存在一定的权衡,这就使得最优选择的确定变得十分困难。此外,对于选择好的一些构件,如何安排它们的实现顺序以最大化项目收益也需要谨慎考虑。类似于服务选择问题,假设当前共有 n 个构件 x_1, \dots, x_n 可供选择,可用的开发资源为 K ,其中开发每个构件所需的成本为 c_i ,且对应的预期的收益为 w_i ,则构件选择问题就等同于背包问题,即选择某 m 个构件以使得在满足约束条件的前提下最大化收益。由于这一问题已知为 NP-hard 的,为了寻找尽可能好的解,Baker 等人^[95]、Vijayalakshmi 等人^[162]以及 Yang 等人^[163]分别使用模拟退火和遗传算法来对其进行优化。传统的运筹学方法在这一问题上同样被广泛使用,例如 Desnos 等人^[164]曾尝试将回溯法和分支限界法进行结合。此外,Cortellessa 等人^[165]提出了一个旨在最小化系统构件成本的构件选择框架,Kuperberg 等人^[166]提出了一个基于遗传规划的重工程参数行为模型来进行黑盒构件性能的预测。

在软件设计领域,Feldt^[167~169]还尝试对容错进行考虑,其使用遗传规划方法来对软件进行演化以得到软件的多个版本,从而帮助开发人员在 N -版本容错中降低开发不同软件版本的开销,同时提高整个软件系统的鲁棒性。在其他的软件设计问题上,Barlas 和 RI-Fakih^[170]使用遗传算法来优化多客户端和多服务器系统中的应用交付问题;Chardigny 等人^[171]使用搜索技术来解决面向对象体系结构中基于组件的抽取问题;Sharma 和 Jalote^[172]使用搜索技术来优化所部署软件构件的性能。

2.2.5 软件重构与维护

软件维护费用占软件开发总费用的 40% 以上。程序开发者通过软件重构、程序分析等手段提高软件的灵活性、可重用性等方面进而降低软件维护开销。基于搜索的软件工程利用搜索算法寻找有价值的软件重构方式或程序片组合模式,进而提高软件维护过程的效率,最终达到自动化或半自动化软件维护的目的。利用 SBSE 进行软件维护通常包括三个步骤:1) 建模软件维护问题;2) 设定合适的目标函数;3) 选择合适的搜索算法。基于搜索的软件工程在软件重构与维护问题上的主要研究内容包括软件重构与程序分析两部分。

软件重构是 SBSE 在软件维护中的主要研究方向。早期的基于搜索的软件重构技术集中在利用搜索技术提高程序执行效率、减少程序规模,该过程主要是通过启发式算法搜索并优化程序中的循环语句、冗余语句等,寻找更高效的代码表现形式^[173~175]。

随着面向对象语言的成熟,研究者尝试结合面向对象语言的特性利用基于搜索的软件重构方式进行自动化或半自动化的软件重构工作^[176~178]。在自动化软件重构过程中,研究者首先对软件重构问题进行建模,如从代码级别、方法级别甚至是模块级别^[179]进行问题建模,寻找符合搜索算法的域编码方式,并在此阶段确定不同域的重构规则,如降低域的继承层次、增加子类等规则。除针对软件源代码进行建模外,也有研究者对软件中说明性语言(declarative languages)进行建模,研究软件说明性语言的自动化重构^[180,181]。其次 SBSE 需要设定合适的目标函数(fitness function)。在目前的研究中,软件重构目标函数以 QMOOD 度量为主,QMOOD 度量从软件的灵活性、可重用性、可理解

性等方面评价软件重构结果的优劣。也有研究者从其他角度,如重构软件的可测试性等方面,进行软件重构结果的度量^[182,183]。由于软件重构结果可以从多个角度进行度量,因此研究者尝试联合多个目标函数进行多目标的软件重构优化^[184]。最后 SBSE 需要选择合适的搜索算法,大量的搜索算法被尝试用于进行软件重构,并且研究者通过经验学习的方式在多个数据集上结合多种软件重构规则来对比不同搜索算法在软件重构的可用性^[185,186],所对比的搜索算法如遗传算法、模拟退火算法、爬山算法等。

由于自动化软件重构可能会产生大量无意义的重构模式,研究者又尝试引入人机交互进行半自动化软件重构,通过对重构中间结果的人工干预,使得最终的重构软件符合预期。常见的方式如交互式遗传算法(interactive genetic algorithm)、可视化的 Pareto 最优曲线的引入^[178,184]等。

SBSE 在软件维护中的另一个研究方向是程序分析。程序分析这里指依赖分析(dependence analysis)与概念分配(concept assignment)。自动化程序分析能够帮助软件维护人员快速理解整个待维护的软件系统。在依赖分析方面,研究者利用遗传算法、贪心算法等搜索能够覆盖所有程序功能点的程序片组合,帮助维护人员进行程序依赖结构分析^[187]。在概念分配方面,研究者利用搜索算法寻找可能的程序片组合方式,挖掘便于理解的高层概念^[188]。

2.2.6 软件项目开发管理

软件项目开发管理是整个软件开发过程的基础,它通过分析软件项目规模、开发人员专业度等因素来辅助项目经理对软件项目的人员、开发任务进行合理分配,达到减少软件开发时间、减少人员时间碎片的目的。SBSE 把软件项目开发管理建模为对软件项目中多种资源的组合优化的问题,利用搜索算法进行高效的资源分配和项目评估。软件项目开发管理涵盖软件项目开发中的时间管理、花销管理、质量管理、人力资源管理、风险管理等,它可以分为软件项目资源优化和软件评估两方面。

在软件项目资源优化方面,研究者利用分散搜索(scatter search)、遗传算法并结合软件项目中已知的可更新资源、人员等级等进行软件项目中的人员分配优化和工作模块分配优化,并研究如何根据人员的数量提出不同的分配意见^[189,190]。另外,人员交流情况、人员专业度等特征也可作为影响软件项目优化的因素^[191,192]。除进行单目标的资源优化外,软件项目中往往需要同时对诸如项目完成时间、项目花销等多目标进行优化。研究者提出利用多目标搜索的方式完成该任务,如利用多目标遗传算法、NSGA-II、SPEA2 等演化算法^[193,194]。也有研究者从迭代的角度对多个优化目标进行迭代交叉优化,或以基于事件的方式在某个资源发生变化时对其相对应的目标进行优化^[189,195]。考虑到多目标优化结果的多样性,研究者利用可视化的方式让决策者从多个优化方案中进行选择,如 Pareto 最优或对每一个目标利用仿真模拟器生成其相对应的结果供决策者选择^[191,196]。

软件评估是软件项目开发管理中的重要活动,利用软件评估结果可以进行任务分配、人员分配等任务,典型的软件评估工作如软件规模评估、软件开发开销评估等。研究者

对已有的机器学习软件评估方法与基于搜索的方法进行对比,证明后者能够完成软件评估任务,并可能获得更好的准确性,但是基于搜索的方法在程序设置和运行开销上优势并不明显^[197,8]。因此研究者提出利用混合策略把已有的软件评估方法和启发式算法相结合,进行联合的软件评估,提高评估的准确性,例如把基于类比(Analogy-Based Estimation, ABE)的评估方式与粒子群算法相结合^[198]。大量的搜索算法(如遗传算法、爬山算法、模拟退火算法等)被用于进行软件评估工作,同时有研究者对不同算法、不同项目规模、不同建模方式进行对比,评估各种因素在软件评估方面的优劣^[199,200]。除了常见的软件规模评估、软件开销评估外,也有研究者利用SBSE进行软件复杂度、耦合度等影响其质量的因素的评估^[201]。

在软件项目开发管理中,与上述利用软件项目资源优化结果和软件评估结果来间接辅助项目经理决策不同,SBSE也可以直接基于已有的项目决策数据,搜索并生成优秀的项目决策组合方案。研究者可以根据项目经理对已有项目决策的标注结果,结合软件项目模拟器(software project simulator)的模拟数据,利用搜索算法搜索出大量“GOOD”决策,或直接把搜索算法应用到范例推理系统中(Case-Based Reasoning system, CBR)^[202,203]。

2.3 基于搜索的软件工程在工业界的应用

戴姆勒-克莱斯勒公司(DaimlerChrysler AG)较早就对基于搜索的软件工程的工业运用进行了尝试。为了满足单元测试的需要,J. Wegener等人设计并搭建了完整的演化测试环境^[204],其核心思想是利用遗传算法完成测试数据的自动生成。文中将不同的测试准则分为四类,并提出了针对性的适应度函数。A. Windisch等人^[205]于2007年提出运用完全学习粒子群算法(Comprehensive Learning Particle Swarm Optimization, CL-PSO)来代替遗传算法。实验比较两种算法在13个工业程序上的测试数据生成效率,发现了CL-PSO在各个被测程序上的测试用例生成效率上不劣于GA,并且在更复杂的被测程序上要明显优于GA算法。Harman等人分析了输入域约减对基于搜索的测试数据生成的影响^[206],通过实验验证了随机测试、爬山算法以及演化测试对搜索空间的敏感程度,得出了随机测试不受输入域约减的影响以及演化测试较爬山算法受影响程度更大的结论。在以上研究与工业实践的基础上,O. Buhler和J. Wegener通过模拟具体场景的方法对演化测试的应用范围做了进一步的延伸^[207],设计了针对两个特定的系统(自动停车系统和辅助刹车系统)的功能性演化测试方法。

微软公司的PEX是工业应用广泛的软件测试工具。PEX与其他许多工业级的软件测试工具相同,也使用了符号执行(Symbolic Execution)技术^[208]。一方面T. Xie等人^[209]将基于搜索的软件工程技术与PEX中的符号执行技术相结合,通过利用适应度函数引导的搜索策略改进传统的符号执行的路径搜索策略,使得新的测试方法能够更早地达到较高的覆盖率。另一方面,K. Lakhotla^[210]将基于搜索的软件测试引入动态符号执行(Dynamic Symbolic Execution),分别利用AVM(Alternating Variable Method)以及ES(Evolution Strategies)两

种搜索算法来改进 PEX 对浮点数约束处理能力的不足。实验比较了 PEX 以及两个应用搜索算法的 PEX，结果显示后者确实能提高 PEX 对包含浮点数计算的程序的覆盖率，但是大量的适应度函数计算也带来了更多的执行时间开销。

谷歌公司使用大量的并行设备来执行大规模的回归测试。但测试用例集过大以及高频率的代码变化都限制了回归测试的效率。因此，谷歌希望在代码被提交到并行平台上进行大规模的回归测试前，开发者就能够在本地对部分模块先进行小规模测试。针对这一需求，S. Yoo 等人于 2011 年利用回归测试优化技术设计了新的测试框架^[211]。这一框架包含本地测试以及全局测试两部分。通过使用 Two-Archive 多目标演化算法（Multi-Objective Evolutionary Algorithm, MOEA）对四个优化目标：依赖覆盖、故障历史、执行时间以及误报测试用例过滤进行优化，选择出测试用例子集用于提交前的本地测试。提交前的测试优化有以下三个优点：1）避免了开发者人工确定测试用例与被修改的模块之间是否关联的开销；2）能够给予开发者更早的测试反馈信息；更早地对错误进行修复提高了整个测试过程效率。基于保持已有测试框架完整性的考虑，在并行设备上进行的全局测试需执行所有与改动关联的测试用例。这样保证了测试用例集能够发挥全部错误检测能力。实验结果显示，优化的测试用例子集相比较测试用例全集可以减少 33% ~ 82% 的测试时间。

IBM 公司将多目标优化算法应用于大规模测试用例最小化中，并使用 GPGPU 提升了算法效率。S. Yoo 等^[211]指出频繁的适应度计算耗费了大量的时间，这使得许多基于搜索软件工程的研究不能直接应用到工业上。然而，GPGPU 并行构架的出现为这一问题提供了可行的途径。文章提出了一种基于搜索的优化算法来解决测试用例最小化问题，并且用 GPU 加速了该算法。除了实验室级的标准测试程序，工业界的程序同样被用来评估文章提出的方法。实验表明，通过使用 GPGPU 技术能够获得超过 25 倍的加速比。

IBM 现有的测试用例选择方案能够从测试用例集中选择出多个测试用例子集。但是实验发现，有一些测试用例频繁出现在各个测试用例子集中。为了避免每次回归测试频繁地执行同样的测试用例，每一次回归测试的时候不选择那些之前已经执行过的测试用例。这种策略会使得测试用例池的规模慢慢减小，同样降低了测试用例子集的代码覆盖率。S. YOO 等人^[212]提出了一种测试用例约简和生成相结合的方法，在从测试用例集中选出测试用例子集后，重新生成一些测试用例来保证测试用例集的完整性。针对测试用例的再生成，文献[212]提出了几种策略：爬山算法和分布估计算法（Estimation of Distribution Algorithm, EDA）。实验将结合测试用例生成的方法与原有方法对比，通过记录回归测试的每次迭代选择出的测试用例子集能够达到的最大代码覆盖率，验证了文章提出的方法能够在减少测试用例重用的基础上保持测试用例子集的代码覆盖率。

摩托罗拉公司运用基于搜索的软件工程技术来解决软件最优发布问题。P. Baker 等人^[213]在文中指出，随着基于组件的软件开发的广泛应用，管理者更多地面临困难的决策：在综合考虑众多影响因素时，如何从大量的候选组件中选择出合适组件以及确定组件的优先级。这些因素可能包括：获取开销、用户意愿、开发时间、预期回报、组件间的依赖关系和不同用户的优先级。通过将这一决策问题转化为最优化 0-1 背包问题

(optimization 0-1 knapsack problem), 文章提出使用模拟退火算法以及贪心算法的解决方法。实验对比以上两种算法以及专家人工评估的方法在一个包含 40 个组件的软件库上的表现, 结论显示两种自动的方法都优于人工的方法, 并且模拟退火算法的效率以及稳定性都优于贪心算法。

爱立信公司运用基于搜索的软件工程技术解决需求分析和优化的问题。Y. Zhang 等人指出虽然需求提取以及分析是软件开发过程中的第一个行为, 但是在实际开发过程中需求却经常发生变化^[214]。因此, 文章提出了现在/未来重要性分析 (Today/Future Importance Analysis, T/FIA)。需求的变化可以分为可预知的以及不可预知的两类。文章 [214] 利用可预知的需求变化以及不可预知中已知很可能发生的需求变化来构造未来的需求。针对当前需求、未来需求以及开销三个目标, 使用 NSGA-II 算法对原始需求集进行优化。实验数据使用了来自爱立信公司 14 个不同软件测试子部门对同一测试管理软件的需求。结果显示 NSGA-II 算法产生了具有良好分布的 Pareto Front 最优解集。

尽管基于搜索的软件测试已经有了大量的实际应用, 但是依然缺乏相应的开放工具。K. Lakhotia 等人^[215]设计并实现了针对 C 程序的基于搜索的开源测试数据生成工具 AUSTIN (Augmented Search-based TestINg)。AUSTIN 可以生成大多数类型的测试数据, 除了字符串、空指针、函数指针和结构体。AUSTIN 支持三种搜索算法: 随机搜索、爬山法和结合符号执行的爬山法。实验选择 8 个 C 函数作为评估的基准程序。这 8 个程序选自汽车的自动控制系统, 包括自适应车前灯控制、门锁控制以及电动窗控制系统。通过与先进的演化测试工具 ETF 进行比较, AUSTIN 在生成满足分支覆盖准则的测试数据上具有更高的效率。

3 国内研究进展

在国内, SBSE 的研究发展也很有起色。目前, 已经有很多高等院校、研究所的教师和学生从事该方向的研究工作, 并组织了国内基于搜索的软件工程研讨会 (Chinese Search Based Software Engineering Workshop)[⊖]。第一届于 2012 年 7 月在北京化工大学召开, 第二届于 2013 年 6 月在大连理工大学召开, 第三届于 2014 年 7 月 4 日在中国矿业大学召开。研讨会把国内从事 SBSE 相关的学者组织在一起, 逐步发展并形成了一定规模的研究人员社区。

国内研究人员紧跟 SBSE 研究热点, 很多研究成果都发表在重要国际会议和国际期刊上, 在上一章节的综述中已经包括。本章我们仅收集了《软件学报》、《计算机学报》、《计算机研究与发展》和《电子学报》近五年刊发的与 SBSE 相关的文献, 历年相关的文献发表情况如图 14 所示。可以看出, 从 2009 年至今, 每年发表的有关 SBSE 的文献数量在稳步上升, 而软件测试在其中占据了很大的比重。

⊖ <http://www.csbse.org>.

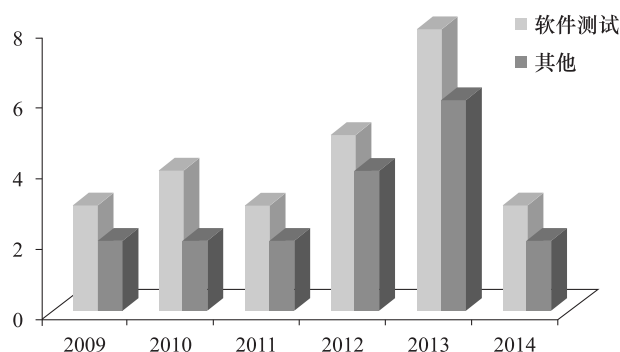


图 14 SBSE 方向论文发表情况年度表

软件测试的主要工作集中在测试用例（测试数据）的生成^[216~230]，约简^[231]和优化^[232~235]。其中，利用进化算法进行测试用例生成是目前研究工作的热点，例如：毛澄映等^[216]在讨论基于搜索的测试数据生成基本框架的基础上，以分支覆盖作为测试覆盖准则，给出了基于粒子群优化的测试数据生成算法，并通过分析分支谓词的结构特征提出了一种新的适应函数构造形式；史娇娇等^[217]针对粒子群算法易陷入局部最优解及搜索精度低的问题，提出一种约简的自适应粒子群优化算法应用于测试数据自动生成；姚香娟等^[218]利用语句占有关系，把含有标记变量的测试数据问题转化为语句覆盖问题，然后再利用遗传算法进行求解；张岩等^[219]提出一种基于搜索空间自动缩减的路径覆盖测试数据进化生成方法，使种群在不断缩小的空间里寻找测试数据，以提高测试数据生成的效率；巩敦卫等^[220]采用遗传算法生成回归测试数据以覆盖目标路径时，利用已有测试数据，提出一种新的回归测试数据进化生成方法；侯可佳等^[228]建立了基于服务接口语义契约模型（Interface Semantic Contract, ISC）并探讨了基于 ISC 的测试数据生成技术，给出了分区生成算法以及测试数据生成的模拟退火算法。测试用例约减可有效降低测试用例集规模，顾庆等^[231]针对选择性回归测试中测试需求集部分覆盖要求，提出采用启发式贪婪搜索算法解决多目标部分覆盖测试用例集约简问题。测试用例优化问题是回归测试研究中的一个热点，例如，聂建平等^[234]提出了一种基于 I/O 的 ART 方法，其发现失效的效率较之前方法有了极大的提高，同时可以一次发现多个失效。

国内学者还研究了 SBSE 在 Web 服务的组合优化和网络服务方面的应用^[236~241]，例如：温涛等^[237]提出的改进离散粒子群算法用于解决 Web 服务组合优化问题；谢晓芹等^[239]针对在开放、动态环境下现有的服务发现研究中存在的搜索效率不高、负载不均衡和语义欠缺等问题，提出了一种基于推荐网络和蚁群算法的服务发现方法；黄发良等^[240]将网络社区发现问题形式化为多目标优化问题，提出了一种基于多目标粒子群优化的网络社区发现算法。

此外，基于搜索的方法也在其他方面得到了应用^[242~244]，梁亚澜等^[242]针对已有工作未能系统探索遗传算法生成覆盖表的性能，对遗传算法的配置参数进行了更为深入的探索；严秋玲等^[244]比较了列存储系统中查询优化与行存储系统的不同，在此基础上提出适合于列存储的启发式查询优化机制。

4 国内外研究进展比较

基于搜索的软件工程概念最先由英国 Harman 教授等几位学者提出，并且在英国 EPSRC 的资助下，率先于英国和欧洲召开关于 SBSE 领域研究的研讨会。

图 15 统计了目前已经收录 SBSE 领域文章的作者国家分布，可以看出英国的作者占到四分之一。中国的软件工程学科起步较晚，2012 年才被正式认定为国家一级学科，但中国的学者一直跟踪国际软件工程领域的研究热点，在国际会议和国际期刊发表的文献中作者数量已占到 5% 左右。

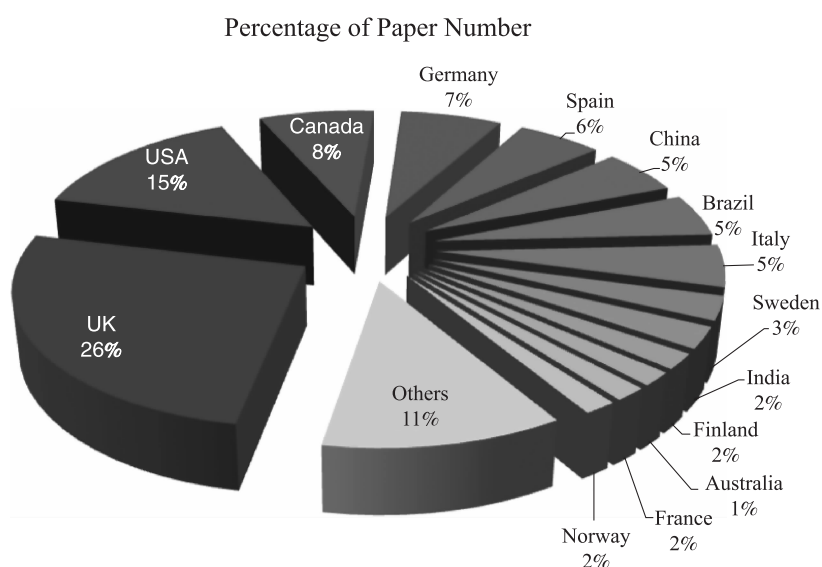


图 15 SBSE 研究人员的国家分布

此外，SBSE 是智能计算与软件工程的交叉，而国内从事智能计算、演化计算、系统优化的人员主要集中在控制科学与工程学科，从事软件工程的研究人员则集中在计算机科学与技术学科，所以目前国内从事 SBSE 领域的研究人员部分是具有计算机学科背景的，部分是具有控制学科背景的。SBSE 真正把两个学科的人交叉凝聚到了一起。

从研究内容上看，由于国内学者密切关注国际上的新热点，同步开展相关研究，在软件测试的自动化和智能化、测试数据自动生成、组合测试、程序自动修复、基于 GPGPU 的并行多目标演化算法等领域有较深入的研究，部分研究成果已经处于世界领先水平。

5 发展趋势与展望

2012 年 Harman 等人在 ACM Computing Surveys 上发表了 SBSE 综述文章“Search

Based Software Engineering: Trends, Techniques and Applications”^[245]，分析报告了 SBSE 的成果、应用和发展趋势。报告从 SBSE 本质提出四个基础研究方向：

(1) 普适性和可应用性 (Generality and Applicability)

基于搜索的软件工程研究目前主要集中在软件测试领域，但大部分软件工程过程中的行为都可以运用基于搜索的优化方法进行优化。为了达到这一目的，需要解决两方面问题。1) 解决方案表达方式 (Solution Representation)：解决方案的表达方式在最优化问题中扮演一个重要的角色，它部分决定了解决方案的质量，并且帮助软件工程师更好地理解分析结果。不同于其他工程学科，软件工程领域内的问题往往具有较好的可描述性。规范化描述软件工程中更多问题的解决方案，能够使得软件工程更多地受益于基于搜索的优化算法。2) 适应度函数的定义 (Fitness Function Definition)：适应度函数作为评估解决方案的方法，引导了对较优解决方案的搜索。因此，适应度函数是 SBSE 的核心。在软件工程的领域中，评估与度量作为一个重要的研究领域已引起了大量研究人员的研究兴趣，并产生了大量的研究成果。因此，如何对已有的评估方法进行改造，使之转化为可测量的参数，是基于搜索的优化方法在软件工程领域内更广泛应用的重要挑战。

(2) 可扩展性 (Scalability)

软件工程师所面临的一个重要问题就是解决方案有可扩展性，导致许多表现优异的实验室级方法在工业界无法被广泛应用。但幸运的是，基于搜索的优化方法具有天然的并行性。在如今软件高度并行化的趋势、软件工程可扩展性的需要以及基于搜索软件工程方法天然并行性的共同作用下，针对并行的基于搜索的软件工程研究必将得到显著的发展。

(3) 鲁棒性 (Robustness)

在软件工程的一些领域中，问题解决方案的鲁棒性可能与解的优劣同样重要。具体体现为，在搜索空间中找到一个适应度较为稳定的区域要优于找到一个被较差适应度的区域包围的较优解。目前，基于搜索的软件工程的研究关注于构造适应度值更高的解。但是，当目标发生一些微小的变化时，解的适应度可能会显著下降。因此，为了应对这种变化的需要，未来的研究应更多关注于解的鲁棒性。

(4) 反馈与洞察力 (Feedback and Insight)

不同于人为设计，自动化的搜索方法能够避免一些人类直觉可能导致的偏差，这使得基于搜索的方法擅于构造一些意想不到的解，有助于洞察问题解决方案的核心。这一核心优势目前已在工业设计中得到了一定的证实。可以预期，将人为决策与自动化搜索串联结合的迭代过程会产生更好的解决方案。这必将成为未来研究的重要考量。

由于基于搜索的软件测试是 SBSE 研究领域中所占份额最大的部分，下面分 6 个方面给出基于搜索的软件测试的未来研究方向。1) 执行环境的处理，通常基于搜索的测试数据生成技术，缺少对操作系统、文件系统、网络访问及数据库的处理，即软件执行环境的处理；但是执行环境的不同，往往会影响测试的结果，因此有必要对执行环境进行相应的处理。2) 提高改善可测试性的方法，由于标志变量只有两种取值情况，对搜索过程起不到向导的作用，因此通过可测试性转换解决该问题；针对该问题还需要提出更多的

可测试转换方法。3) 通过可测试性转换的自动 Oracles 问题, 通过转换生成另一个版本的程序可能不同于原始程序, 这就可能引起错误的结果; 因此, 如何设计适应度函数以区分一个程序的不同版本是值得研究的问题。4) 搜索代码迁徙和重建是否成功, 对于不同两个版本的程序, 根据它们的行为要采取不同的适应值函数作为向导, 因此搜索两个不同元素之间的不同是需要的, 其中包括代码的迁徙步骤是否执行正确, 及重建是否维持原系统的行为等。5) 最小化人工 Oracle 花费, 人工花费往往在测试中占据不小的比例, 但是对这方面的关注仍很少; 其中一个好的解决方案就是, 通过降低生成的测试用例的缺陷检测能力, 以降低人工的花费, 这便需要二者之间的一个平衡。6) 多个测试目标, 在基于搜索的方法中, 一次优化多个适应值函数的方法往往应用很少; 该方法可以生成同时满足最大覆盖率和最小运行时间的测试数据, 对于那些潜在的多个优化目标的问题, 都有待于多目标优化算法的应用。

通过对国内外研究的综述与比较, 基于搜索的软件工程将呈现以下几方面的趋势:

(1) 大数据环境下的基于搜索的软件工程

伴随着计算机软硬件技术、网络技术、移动通信技术、信息处理技术等蓬勃发展, 物联网、云计算、大数据等新技术被业界提出, 从不同层面拓展了软件等外延和内涵, 也对软件开发提出更多的挑战。2012 年 Harman 等就提出基于云工程也是基于搜索的软件工程^[246], 并通过分析云计算的特点总结出基于搜索的云计算工程所面临的 5 个挑战。在 2014 年的国际基于搜索软件工程研讨会 (Symposium on Search- Based Software Engineering, SSBSE 2014)[⊖]上, 提出了 SBSE 在大数据领域广泛使用的 Hadoop 系统应用的专题征文。可以看出大数据环境下的基于搜索的软件工程将成为未来的发展趋势之一。

(2) 基于搜索的动态自适应软件工程

SBSE 在软件开发生命周期的每个阶段都有应用, 基本涵盖了软件工程的方方面面, 但同时也可以看出, 现有 SBSE 的应用主要集中在软件生命周期中具体的一个阶段中的个案, 并没有面向整个生命周期体系。智能计算及优化在其他工程领域已经有很多系统层次的解决方案, 但在软件工程领域还没有, 因此我们提出软件工程自动化是 SBSE 的最终目标, 基于搜索的动态自适应软件工程将成为 SBSE 未来发展的重要方向之一。

(3) 知识自动化是软件工程自动化的核心

基于搜索的软件工程的最终目标是实现软件的自动化, 而软件工程自动化的核心就是知识自动化。知识自动化可以分为两个方面, 一方面是已知或已约定的知识自动化, 另一方面是未知或无法规定的模式的表示及处理, 需要融入机器学习和人机交互等方法和技术, 间接地改变行为模式, 从以“知你为何”为基础实现自动化, 转化到以“望你为何”为依据争取智能化, 促使希望的测试结果或者目标得以实现。

当前, 正值科技革命的重要变革时期, 网络空间正以巨大的冲击力影响着我们的生活和工作的各个方面。互联网、物联网、云计算、大数据等理念和技术的到来, 预示并已经开拓了人类向人工世界进军, 深度开发数据和智力资源, 深化农业和工业革命

⊖ <http://ssbse.org/2014/>.

的时代使命。在这一历史进程中,以数据驱动的自动化知识将是关键的核心支撑科学和技术。

知识自动化绝对不是知识本身的自动产生,但可以诱发知识的传播、获取、分析、影响、产生等方面的重要变革。知识自动化必将在今后的软件工程中起到关键作用。我们必须从面向物理世界的工业自动化,走向面向数据的知识自动化。

6 结束语

基于搜索的软件工程最终目标是软件工程的自动化和智能化。虽然国内外在该领域的研究已经形成一定规模,但从目前的成果看,离目标还有很大距离,还需要开展更广泛的、更深入的研究,并进一步结合工业界的实际问题,推动发展和提高水平。

基于搜索的软件工程是智能计算与传统软件工程的交叉与结合。一方面,智能计算、机器学习等技术等发展不断为基于搜索的软件工程领域研究提供新的技术支撑;另一方面,软件开发技术的发展也将影响软件工程的发展,势必为该领域的研究提出更多、更新的挑战,提供更广阔的发展空间。

参考文献

- [1] W Miller, D L. Spooner, Automatic Generation of Floating-Point Test Data [J]. IEEE Transactions on Software Engineering, 1926, 2(3): 223-226.
- [2] S Xanthakis, C Ellis, C Skourlas, A Le Gall, S Katsikas, K Karapoulios. Application of Genetic Algorithms to Software Testing [C]. Proceedings of the 5th International Conference on Software Engineering and Applications, 1992: 625-636.
- [3] M Harman, B Jones. Search-based software engineering Information and Software Technology, 2001, 43(14): 833-839.
- [4] M Harman. The current state and future of search based software engineering, in Future of Software Engineering 2007(FOSE 2007) [J]. IEEE Computer Society, 2007: 342-357.
- [5] M Harman, J Clark. Metrics are fitness functions too, in International Software Metrics Symposium (METRICS 2004) [J]. IEEE Computer Society, 2004: 58-69.
- [6] Harman M. The current state and future of search based software engineering [C]. 2007 Future of Software Engineering. IEEE Computer Society, 2007: 342-357.
- [7] E Alba, J F Chicano. Observations in using parallel and sequential evolutionary algorithms for automatic software testing [C]. Computers and Operations Research (COR) focused issue on Search Based Software Engineering.
- [8] J J Dolado. A validation of the component-based method for software size estimation [J]. IEEE Transactions on Software Engineering, 2000, 26(10): 1006-1021.
- [9] J J Dolado. On the problem of the software cost function [J]. Information and Software Technology, 2001, 43

- (1): 61-72.
- [10] S Wappler J Wegener. Evolutionary unit testing of object-oriented software using strongly-typed genetic programming[C]. In GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation, Seattle, USA, 8-12 July 2006. ACM Press, 2006, 2: 1925-1932.
- [11] M Harman, R Hierons, M Proctor. A new representation and crossover operator for search-based optimization of software modularization[C]. In GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, San Francisco, USA, 9-13 July 2002. Morgan Kaufmann Publishers, 2002: 1351-1358.
- [12] B S Mitchell, S Mancoridis. Using heuristic search techniques to extract design abstractions from source code [C]. In GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, San Francisco, USA, 9-13 July 2002. Morgan Kaufmann Publishers, 2002: 1375-1382.
- [13] S Bouktif, H Sahraoui, G Antoniol. Simulated annealing for improving software quality prediction[C]. In GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation, Seattle, USA, 8-12 July 2006. ACM Press, 2006, 2: 1893-1900.
- [14] M Harman, K Steinhöfel, A Skaliotis. Search based approaches to component selection and prioritization for the next release problem[C]. In 22nd International Conference on Software Maintenance (ICSM 06), Philadelphia, USA, Sept. 2006.
- [15] N Tracey, J Clark, K Mander. Automated program flaw finding using simulated annealing [C]. In International Symposium on Software Testing and Analysis (ISSTA 98). 1998: 73-81.
- [16] Li H, Lam C P. Software Test Data Generation using Ant Colony Optimization[C]. International Conference on Computational Intelligence. 2004: 1-4.
- [17] Li H, Lam C P. An ant colony optimization approach to test sequence generation for state based software testing[C]. Quality Software, 2005. (QSIC 2005). Fifth International Conference on. IEEE, 2005: 255-262.
- [18] Ayari K, Bouktif S, Antoniol G. Automatic mutation test input data generation via ant colony [C]. Proceedings of the 9th annual conference on Genetic and evolutionary computation. ACM, 2007: 1074-1081.
- [19] Windisch A, Wappler S, Wegener J Applying particle swarm optimization to software testing [C]. Proceedings of the 9th annual conference on Genetic and evolutionary computation. ACM, 2007: 1121-1128.
- [20] Li A, Zhang Y. Automatic Generation Method of Test Data for Software Structure Based on PSO [J]. Computer Engineering, 2008, 6: 036.
- [21] Nie P, Geng J, Qin Z. Self-adaptive inertia weight PSO test case generation algorithm considering prematurity restraining[J]. International Journal of Digital Content Technology and its Applications, 2011, 5(9): 125-133.
- [22] John Koza, James Rice. Genetic Generation of Both the Weights and Architecture for a Neural Network[C]. International Joint Conference On Neural Networks, 1991.
- [23] Jia X, Tang C, Zuo J, Chen A, Duan Lei, Wang R. Mining Frequent Function Set Based on Gene Expression Programming[J]. Chinese Journal of Computers, 2005, 28(8).
- [24] Xuan S, Liu Y. GEP Evolution Algorithm Based on Control of Mixed Diversity Degree [J]. Pattern Recognition and Artificial Intelligence, 2012, 25(2).
- [25] Li L, Qu L. Fault Detection Based on Genetic Programming and Support Vector Machines[J]. Journal of

- XI'AN Jiaotong University, 2004, 38(3).
- [26] Phil McMinn. Search-based Software Test Data Generation: A survey[J]. *Software Testing, Verification and Reliability*, 2004, 14(2): 105-156.
 - [27] Shaukat Ali, Lionel C Briand, Hadi Hemmati, Rajwinder Kaur Panesar-Walawege. A systematic review of the application and empirical investigation of search-based test-case generation [J]. *IEEE Transactions on Software Engineering*, 2010, 36(6): 742-762.
 - [28] Wasif Afzal, Richard Torkar, Robert Feldt. A systematic review of search-based testing for non-functional system properties[J]. *Information and Software Technology*, 2009.
 - [29] Phil McMinn. Search-based software testing: Past, present and future[J]. *Proceedings of the 4th International Workshop on Search-Based Software Testing*, Berlin, Germany, 21-21 March 2011. 2011: 153-163.
 - [30] Webb Miller, David L Spooner. Automatic generation of floating point test data [J]. *IEEE Transactions on Software Engineering*, 1976, 2(3): 223-226.
 - [31] Bogdan Korel. Automated software test data generation [J]. *IEEE Transactions on Software Engineering*, 1990, 16(8): 870-879.
 - [32] Joachim Wegener, Oliver B. Evaluation of different fitness functions for the evolutionary testing of an autonomous parking system [C]. *Proceedings of the 2004 Conference on Genetic and Evolutionary Computation*, Seattle, USA, 26-30 June 2004. Springer Berlin / Heidelberg, 2004, 3103: 1400-1412.
 - [33] Oliver B, Joachim Wegener. Evolutionary functional testing[J]. *Computers; Operations Research*, 2008, 35(10): 3144-3160.
 - [34] Joachim Wegener, Harmen-Hinrich Sthamer, Bryan F Jones, David E Eyres. Testing real-time systems using genetic algorithms[J]. *Software Quality Journal*, 1997, 6(2): 127-135.
 - [35] Joachim Wegener, Matthias Grochtmann. Verifying timing constraints of real-time systems by means of evolutionary testing[J]. *Real-Time Systems*, 1998, 15(3): 275-298.
 - [36] Joachim Wegener, Frank Mueller. A comparison of static analysis and evolutionary testing for the verification of timing constraints[J]. *Real-Time Systems*, 2001, 21(3): 241-268.
 - [37] Nigel Tracey, John A Clark, Keith Mander. The way forward for unifying dynamic test-case generation: the optimisation-based approach [J]. *Proceedings of the IFIP International Workshop on Dependable Computing and Its Applications*, Johannesburg, South Africa, 12-14 January 1998. 1998: 169-180.
 - [38] Nigel Tracey. A Search-based Automated Test-Data Generation Framework for Safety-Critical Software [D]. University of York, UK, 2000.
 - [39] Hans-Gerhard Groszlig. A prediction system for evolutionary testability applied to dynamic execution time analysis[J]. *Information and Software Technology*, 2001, 43(14): 855-862.
 - [40] Eileen Dillon. Hybrid approach for the automatic determination of worst case execution time for embedded systems written in c [D]. Institute of Technology, Carlow, 2005.
 - [41] Vu Le Hanh, Kamel Akif, Yves Le Traon, Jean-Marc J. Selecting an efficient oo integration testing strategy: An experimental comparison of actual strategies [C]. *Proceedings of the 15th European Conference on Object-Oriented Programming*, Budapest, Hungary, 18-22 June 2001. Springer, 2001, 2072/2001: 381-401.
 - [42] Lionel C Briand, Jie Feng, Yvan Labiche. Using genetic algorithms and coupling measures to devise optimal integration test orders [C]. *Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering*, Ischia, Italy, 15-19 July 2002. 2002: 43-50.
 - [43] Lionel C Briand, Jie Feng, Yvan Labiche. Experimenting with genetic algorithms and coupling measures to

- devise optimal integration test orders[R]. Technical report, 2003.
- [44] Rafael da Veiga Cabral, Aurora Trinidad Ramirez Pozo, Silvia Regina Vergilio. A pareto ant colony algorithm applied to the class integration and test order problem[C]. Proceedings of the 22nd IFIP International Conference on Testing Software and Systems, Natal, Brazil, 8-12 November 2010. 2010, 6435: 16-29.
- [45] Thelma Elita Colanzi, Wesley Klewerton Guez Assuncao, Silvia Regina Vergilio, Aurora Trinidad Ramirez Pozo. Integration test of classes and aspects with a multi-evolutionary and coupling-based approach[C]. Proceedings of the 3rd International Symposium on Search Based Software Engineering, Szeged, Hungary, 10-12 September 2011. 2011, 6956.
- [46] Romain Delamare, Nicholas A Kraft. A genetic algorithm for computing class integration test orders for aspect-oriented systems[C]. Proceedings of the 5th International Workshop on Search-Based Software Testing, Montreal, Canada, 21-21 April 2012. 2012: 804-813.
- [47] Wesley Klewerton Guez Assuncao Thelma Elita Colanzi, Silvia Regina Vergilio, Aurora Pozo. On the application of the multi-evolutionary and coupling-based approach with different aspect-class integration testing strategies[C]. Proceedings of the 5th International Symposium on Search Based Software Engineering, St. Petersburg, Russia, 24-26 August 2013. 2013, 8084: 19-33.
- [48] Lionel C Briand, Yvan Labiche, Marwa Shousha. Stress testing real-time systems with genetic algorithms[C]. Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, Washington, D C, USA, 25-29 June 2005. 2005: 1021-1028.
- [49] Concettina Del Grosso, Giuliano Antoniol, Massimiliano Di Penta, Philippe Galinier, Ettore Merlo. Improving network applications security: A new heuristic to generate stress testing data[C]. Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, Washington, D C, USA, 25-29 June 2005. 2005: 1037-1043.
- [50] Lionel C Briand, Yvan Labiche, Marwa Shousha. Using genetic algorithms for early schedulability analysis and stress testing in real-time systems[J]. Genetic Programming and Evolvable Machines, 2006, 7(2): 145-170.
- [51] Vahid Garousi. Traffic-aware Stress Testing of Distributed Real-Time Systems based on UML Models using Genetic Algorithms[D]. Department of Systems and Computer Engineering, Carleton University, August 2006.
- [52] Vahid Garousi, Lionel C Briand, Yvan Labiche. Traffic-aware stress testing of distributed real-time systems based on uml models using genetic algorithms[J]. Journal of Systems and Software, 2008, 81(2): 161-185.
- [53] Vahid Garousi. A genetic algorithm-based stress test requirements generator tool and its empirical evaluation[J]. IEEE Transactions on Software Engineering, 2010, 36(6): 778-797.
- [54] Md. Mehedi Masud, Amiya Nayak, Marzia Zaman, Nita Bansal. Strategy for mutation testing using genetic algorithms[C]. Proceedings of 2005 Canadian Conference on Electrical and Computer Engineering, Saskatoon, Saskatchewan Canada, 1-4 May 2005. 2005: 1049-1052.
- [55] Yuan Zhan, John A Clark. Search-based mutation testing for Simulink models[C]. Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, Washington, D C, USA, 25-29 June 2005. 2005: 1061-1068.
- [56] Pete May, Jon Timmis, Keith Mander. Immune and evolutionary approaches to software mutation testing[C]. Proceedings of the 6th International Conference on Artificial Immune Systems, Santos, Brazil, 26-29 August 2007. 2007: 336-347.

-
- [57] Yue Jia, Mark Harman. Constructing subtle faults using higher order mutation testing[C]. Proceedings of the 8th International Working Conference on Source Code Analysis and Manipulation, (Best Paper Award), Beijing, China, 28-29 September 2008. 2008: 249-258.
 - [58] Mark Harman, Yue Jia, William B Langdon. A manifesto for higher order mutation testing[C]. Proceedings of the 5th International Workshop on Mutation Analysis, Paris, France, 6 April 2010. 2010: 80-89.
 - [59] William B Langdon, Mark Harman, Yue Jia. Multi objective higher order mutation testing with gp[C]. Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, Montreal, Canada, 8-12 July 2009. 2009: 1945-1946.
 - [60] JJ Dominguez-Jimenez, A Estero-Botaro, A Garcia-Domnguez, I Medina-Bulo. Evolutionary mutation testing[J]. Information and Software Technology, 2011.
 - [61] Yue Jia, Mark Harman. An analysis and survey of the development of mutation testing[J]. IEEE Transactions on Software Engineering, 2011, 37(5): 649-678.
 - [62] Elmahdi Omar, Sudipto Ghosh, Darrell Whitley. Constructing subtle higher order mutants for java and aspectj programs[C]. Proceedings of IEEE 24th International Symposium on Software Reliability Engineering, Pasadena, CA, USA, 4-7 November 2013. 2013: 340-349.
 - [63] Myra B Cohen, Charles Joseph Colbourn, Alan C H Ling. Augmenting simulated annealing to build interaction test suites[C]. Proceedings of the 14th International Symposium on Software Reliability Engineering, Denver, Colorado, USA, 17-21 November 2003. 2003: 394-405.
 - [64] S Ghazi, Moataz A Ahmed. Pair-wise test coverage using genetic algorithms[C]. Proceedings of the IEEE Congress on Evolutionary Computation, Canberra, Australia, 8-12 December 2003. 2013: 1420-1424.
 - [65] Renee C Bryce, Charles Joseph Colbourn, Myra B Cohen. A framework of greedy methods for constructing interaction test suites[C]. Proceedings of the 27th International Conference on Software Engineering, St. Louis, MO, USA, 15-21 May 2005. 2005: 146-155.
 - [66] Renee C Bryce, Charles Joseph Colbourn. Constructing interaction test suites with greedy algorithms[C]. Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering, Long Beach, CA, USA, 7-11 November 2005. 2005: 440-443.
 - [67] Renee C Bryce, Charles Joseph Colbourn. One-test-at-a-time heuristic search for interaction test suites[C]. Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, London, England, 7-11 July 2007. 2007: 1082-1089.
 - [68] Renee C Bryce, Charles Joseph Colbourn. The density algorithm for pairwise interaction testing[J]. Software Testing, Verification and Reliability, 2007, 17(3): 159-182.
 - [69] Myra B Cohen, Matthew B Dwyer, Jiangfan Shi. Interaction testing of highly-configurable systems in the presence of constraints[C]. Proceedings of the 2007 International Symposium on Software Testing and Analysis, London, United Kingdom, 9-12 July 2007. 2007: 129-139.
 - [70] Myra B Cohen, Matthew B Dwyer, Jiangfan Shi. Constructing interaction test suites for highly-configurable systems in the presence of constraints: A greedy approach[J]. IEEE Transactions on Software Engineering, 2008, 34(5): 633-650.
 - [71] Changhai Nie, Hareton Leung, Baowen Xu. Using computational search to generate 2-way covering array[C]. Proceedings of the 1st International Symposium on Search Based Software Engineering, Cumberland Lodge, Windsor, UK, 13-15 May 2009.
 - [72] Brady J Garvin, Myra B Cohen, Matthew B Dwyer. An improved meta-heuristic search for constrained

- interaction testing [C]. Proceedings of the 1st International Symposium on Search Based Software Engineering, Cumberland Lodge, Windsor, UK, 13-15 May 2009. 2009: 13-22.
- [73] Xiang Chen, Qing Gu, Jingxian Qi, Daoxu Chen. Applying particleswarm optimization to pairwise testing [C]. Proceedings of the 34th Annual Computer Software and Applications Conference, Seoul, South Korea, 19-23 July 2010. IEEE, 2010: 107-116.
- [74] Brady J Garvin, Myra B Cohen, Matthew B Dwyer. Evaluating improvements to a meta-heuristic search for constrained interaction testing[J]. Empirical Software Engineering, 2011, 16(1): 61-102.
- [75] Liang Yalan, Changhai Nie, Jonathan M Kauffman, Gregory M Kapfhammer, Hareton Leung. Empirically identifying the best genetic algorithm for covering array generation [C]. Proceedings of the 3rd International Symposium on Search Based Software Engineering, Szeged, Hungary, 10-12 September 2011. 2011, 6956.
- [76] Nashat Mansour, Rami Bahsoon. Reduction-based methods and metrics for selective regression testing [J]. Information and Software Technology, 2002, 44(7): 431-443.
- [77] Ghinwa Baradhi, Nashat Mansour. A comparative study of five regression testing algorithms [C]. Proceedings of the Australian Software Engineering Conference, Sydney, NSW, Australia, 28 September-2 October 1997. 1997: 174-182.
- [78] Nashat Mansour, Khalid El-Fakih. Simulated annealing and genetic algorithms for optimal regression testing [J]. Journal of Software Maintenance: Research and Practice, 1999, 11(1): 19-34.
- [79] Zheng Li, Mark Harman, Robert M Hierons. Search algorithms for regression test case prioritization [J]. IEEE Transactions on Software Engineering, 2007, 33(4): 225-237.
- [80] Shin Yoo. Extending the Boundaries in Regression Testing: Complexity, Latency, and Expertise [D]. PhD thesis, King's College London, UK, 2009.
- [81] Shin Yoo, Mark Harman. Regression testing minimisation, selection and prioritisation: A survey [J]. Journal of Software Testing, Verification and Reliability, 2012, 22(2): 67-120.
- [82] Liang You, Yansheng Lu. A genetic algorithm for the time-aware regression testing reduction problem [C]. Proceedings of the 8th International Conference on Natural Computation, Chongqing, China, 29-31 May 2012. 2012: 596-599.
- [83] Jeffery Shelburg, Marouane Kessentini, Daniel R. Tauritz. Regression testing for model transformations: A multi-objective approach [C]. Proceedings of the 5th International Symposium on Search Based Software Engineering, St. Petersburg, Russia, 24-26 August 2013. 2013, 8084: 209-223.
- [84] Joachim Wegener, Andre Baresel, Harmen-Hinrich Sthamer. Evolutionary test environment for automatic structural testing [J]. Information and Software Technology Special Issue on Software Engineering using Meta heuristic Innovative Algorithms, 2001, 43(14): 841-854.
- [85] Leonardo Bottaci. Instrumenting programs with flag variables for test data search by genetic algorithms [C]. Proceedings of the 2002 Conference on Genetic and Evolutionary Computation, New York, USA, 9-13 July 2002. 2002: 1337-1342.
- [86] Andre Baresel, Harmen-Hinrich Sthamer. Evolutionary testing of flag conditions [C]. Proceedings of the 2003 Conference on Genetic and Evolutionary Computation, Chicago, Illinois, USA, 12-16 July 2003. 2003, 2724: 2442-2454.
- [87] Mark Harman, Lin Hu, Robert M Hierons, Andre Baresel, Harmen-Hinrich Sthamer. Improving evolutionary testing by flag removal [C]. Proceedings of the 2002 Conference on Genetic and Evolutionary Computation, (Best Paper Award), New York, USA, 9-13 July 2002. 2002: 1359-1366.

-
- [88] Andre Baresel, David Binkley, Mark Harman, Bogdan Korel. Evolutionary testing in the presence of loop-assigned flags: A testability transformation approach [C]. Proceedings of the 2004 ACM SIGSOFT International Symposium on Software Testing and Analysis, Boston, Massachusetts, USA, 11-14 July 2004. ACM, 2004: 108-118.
- [89] Mark Harman, Lin Hu, Robert M Hierons, Joachim Wegener, Harmen-Hinrich Sthamer, Andre Baresel, Marc Roper. Testability transformation [J]. IEEE Transactions on Software Engineering, 2004, 30(1): 3-16.
- [90] Robert M Hierons, Mark Harman, Chris Fox. Branch-coverage testability transformation for unstructured programs [J]. Computer Journal, 2005, 48(4): 421-436.
- [91] AbdulSalam Kalaji, Robert M Hierons, Stephen Swift. A testability transformation approach for state-based programs [C]. Proceedings of the 1st International Symposium on Search Based Software Engineering, Cumberland Lodge, Windsor, UK, 13-15 May 2009. 2009: 85-88.
- [92] Yanchuan Li, Gordon Fraser. Bytecode testability transformation [C]. Proceedings of the 3rd International Symposium on Search Based Software Engineering, Szeged, Hungary, 10-12 September 2011. 2011, 6956: 237-251.
- [93] Konstantinos Adamopoulos, Mark Harman, Robert M Hierons. How to overcome the equivalent mutant problem and achieve tailored selective mutation using co-evolution [C]. Proceedings of the 2004 Conference on Genetic and Evolutionary Computation, Seattle, Washington, USA, 26-30 June 2004. 2004, 3103/2004: 1338-1349.
- [94] Changhai Nie, Hareton Leung. A survey of combinatorial testing [J]. ACM Computing Surveys, 2011, 43(2): 11: 1-29.
- [95] Paul Baker, Mark Harman, Kathleen Steinhofel, Alexandros Skaliotis. Search based approaches to component selection and prioritization for the next release problem [C]. Proceedings of the 22nd IEEE International Conference on Software Maintenance, Philadelphia, Pennsylvania, 24-27 September 2006. 2006: 176-185.
- [96] Kristen R. Walcott, Mary Lou Soffa, Gregory M Kapfhammer, Robert S Roos. Time-aware test suite prioritization [C]. Proceedings of the 2006 International Symposium on Software Testing and Analysis, Portland, Maine, USA, 17-20 July 2006. 2006: 1-12.
- [97] Praveen Ranjan Srivastava, Aditya Vijay, Bhupesh Barukha, Prashant Singh Sengar, Rajat Sharma. An optimized technique for test case generation and prioritization using tabu search and data clustering [C]. Proceedings of the 4th Indian International Conference on Artificial Intelligence, Tumkur, India, 16-18 December 2009. 2009: 30-46.
- [98] Camila Loiola Brito Maia, Fabricio Gomes de Freitas, Jefferson Teixeira de Souza. Applying search-based techniques for requirements-based test case prioritization [C]. Proceedings of the Brazilian Workshop on Optimization in Software Engineering, Salvador, Brazil, 30-30 September 2010.
- [99] Paolo Tonella, Angelo Susi, Francis Palma. Using interactive genetic algorithms for requirements prioritization [C]. Proceedings of the 2nd International Symposium on Search Based Software Engineering, Benevento, Italy, 7-9 September 2010. 2010: 57-66.
- [100] Dayvison Lima, Fabricio Gomes de Freitas, Gustavo Augusto Lima de Campos, Jefferson Teixeira de Souza. A fuzzy approach to requirements prioritization [C]. Proceedings of the 3rd International Symposium on Search Based Software Engineering, Szeged, Hungary, 10-12 September 2011. Springer, 6956: 64-69.
- [101] Sangeeta Sabharwal, Ritu Sibal, Chayanika Sharma. A genetic algorithm based approach for prioritization of

- test case scenarios in static testing [C]. Proceedings of the 2nd International Conference on Computer and Communication Technology, Allahabad, India, 15-17 September 2011. 2011: 304-309.
- [102] Camila Loiola Brito Maia, Thiago do Nascimento Ferreira, Fabricio Gomes de Freitas, Jefferson Teixeira de Souza. An ant colony based algorithm for test case prioritization with precedence [C]. Proceedings of the 3rd International Symposium on Search Based Software Engineering, Szeged, Hungary, 10-12 September 2011. 2011, 6956.
- [103] Matheus Henrique Esteves Paixao, Ma Maria Albuquerque Brasil, Thiago Gomes Nepomuceno da Silva, Jefferson Teixeira de Souza. Applying the ant-q algorithm on the prioritization of software requirements with precedence [C]. Proceedings of the 3rd Brazilian Workshop on Search-Based Software Engineering, Natal, RN, Brazil, 23 September 2012.
- [104] Zheng Li, Yi Bian, Ruilian Zhao, Jun Cheng. A fine-grained parallel multi-objective test case prioritization on gpu [C]. Proceedings of the 5th International Symposium on Search Based Software Engineering, St. Petersburg, Russia, 24-26 August 2013. 2013, 8084: 111-125.
- [105] Paolo Tonella, Angelo Susi, Francis Palma. Interactive requirements prioritization using a genetic algorithm [J]. Information and Software Technology, 2013, 55(1): 173-187.
- [106] Antonio Mauricio Pitangueira, Rita Suzana P Maciel, Marcio de Oliveira Barros, Aline Santos Andrade. A systematic review of software requirements selection and prioritization using sbse approaches [C]. Proceedings of the 5th International Symposium on Search Based Software Engineering (SSBSE '13), St. Petersburg, Russia, 24-26 August 2013. 2013, 8084: 188-208.
- [107] Phil McMinn. Search-based failure discovery using testability transformations to generate pseudo-oracles [C]. Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, Montreal, Canada, 8-12 July 2009. 2009: 1689-1696.
- [108] Mark Harman, Sung Gon Kim, Kiran Lakhotia, Phil McMinn, Shin Yoo. Optimizing for the number of tests generated in search based test data generation with an application to the oracle cost problem [C]. Proceedings of the 3rd International Workshop on Search-Based Software Testing in conjunction with ICST 2010, Paris, France, 6 April 2010. 2010: 182-191.
- [109] Gordon Fraser, Andrea Arcuri. Whole test suite generation [J]. IEEE Transactions on Software Engineering, 2013, 39: 276-291.
- [110] Phil McMinn, Mark Stevenson, Mark Harman. Reducing qualitative human oracle costs associated with automatically generated test data [C]. Proceedings of the 1st International Workshop on Software Test Output Validation, Trento, Italy, 13 July 2010. 2010: 1-4.
- [111] Reza Matinnejad, Shiva Nejati, Lionel Briand, Thomas Bruckmann, Claude Poull. Search-based automated testing of continuous controllers: Framework, tool support, and case studies [J/OL]. Information and Software Technology, Available online 22 May 2014, ISSN 0950-5849. <http://dx.doi.org/10.1016/j.infsof.2014.05.007>.
- [112] Yuhua Qi, Xiaoguang Mao, Yan Lei, Ziyang Dai, Chengsong Wang. The Strength of Random Search on Automated Program Repair [C]. 36th International Conference on Software Engineering, ICSE 2014, May 31-June 7, Hyderabad, India, 2014.
- [113] Qi Yuhua, Mao Xiaoguang, Lei Yan, Wang Chengsong. Using automated program repair for evaluating the effectiveness of fault localization techniques [C]. 22nd International Symposium on Software Testing and Analysis, ISSTA 2013, July 15-20, pp. 191-201, Lugano, Switzerland, 2013.

-
- [114] Qi YuHua, Mao XiaoGuang, Wen YanJun, Dai ZiYing, Gu Bin. More efficient automatic repair of large-scale programs using weak recompilation [J]. Science China-Information Sciences, 2012, 55 (12): 2785-2799.
 - [115] Qi Yuhua, Mao Xiaoguang, Lei Yan. Making automatic repair for large-scale programs more efficient using weak recompilation[C]. 28th IEEE International Conference on Software Maintenance, ICSM 2012, 2012/9/23-2012/9/28, pp.254-263, Riva del Garda, Trento, Italy, 2012.
 - [116] Qi Yuhua, Mao Xiaoguang, Lei Yan. Efficient automated program repair through fault-recorded testing prioritization [C]. 29th IEEE International Conference on Software Maintenance, ICSM 2013, September 22-28, pp.180-189, Eindhoven, Netherlands, 2013.
 - [117] Mao Xiaoguang, Lei Yan, Dai Ziyang, Qi Yuhua, Wang Chengsong. Slice-based statistical fault localization[J]. Journal of Systems and Software, 2014, 89(1): 51-62.
 - [118] Lei Yan, Mao Xiaoguang, Dai Ziyang, Wang Chengsong. Effective statistical fault localization using program slices [C]. 36th IEEE Annual International Computer Software and Applications Conference, COMPSAC 2012, July 16-20, pp.1-10, Izmir, Turkey, 2012.
 - [119] Lei Yan, Mao Xiaoguang, Chen Tsong Yueh. Backward-slice-based statistical fault localization without test oracles[C]. 13th International Conference on Quality Software, QSIC 2013, July 29-30, pp. 212-221, Nanjing, Jiangsu, China, 2013.
 - [120] Bagnall A J, Rayward-Smith V J, Whitley I M. The Next Release Problem[J]. Information & Software Technology, 2001, 43(14): 883-890.
 - [121] Zhang Y, Harman M, Mansouri S A. The multi-objective next release problem[C]. International Conference on Genetic and Evolutionary Computation(GECCO2007), New York: ACM, 2007: 1129-1136.
 - [122] Saliu M O, Ruhe G. Bi-Objective Release Planning for Evolving Software Systems[C]. 6th European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering, New York: ACM, 2007: 105-114.
 - [123] Y Zhang, M Harman, Search based optimization of requirements interaction management[C]. Proceedings of the 2nd International Symposium on Search Based Software Engineering (SSBSE '10), Benevento, Italy. IEEE, 2010: 47-56.
 - [124] Zhang Y. Multi-Objective Search-based Requirements Selection and Optimisation [D]. University of London, 2010.
 - [125] Tonella P, Susi A, Palma F. Interactive requirements prioritization using a genetic algorithm[J]. Information and Software Technology, 2013, 55(1): 173-187.
 - [126] Tonella P, Susi A, Palma F. Using interactive GA for requirements prioritization [C]. Search Based Software Engineering(SSBSE), 2010 Second International Symposium on. IEEE, 2010: 57-66.
 - [127] Kumari A C, Srinivas K, Gupta M P. Software requirements selection using Quantum-inspired Elitist Multi-objective Evolutionary algorithm[C]. Advances in Engineering, Science and Management(ICAESM), 2012 International Conference on. IEEE, 2012: 782-787.
 - [128] Jifeng Xuan, He Jiang, Zhilei Ren, Zhongxuan Luo. Solving the Large Scale Next Release Problem with a Backbone Based Multilevel Algorithm. IEEE Transactions on Software Engineering, vol. 38, no. 5, Sept. - Oct. 2012, pp. 1195-1212
 - [129] Kumari A C, Srinivas K, Gupta M P. Software Requirements Optimization Using Multi-Objective Quantum-Inspired Hybrid Differential Evolution [M]. EVOLVE-A Bridge between Probability, Set Oriented

-
- Numerics, and Evolutionary Computation II. Springer Berlin Heidelberg, 2013: 107-120.
- [130] Gay G, Menzies T, Jalali O, et al. Finding robust solutions in requirements models [J]. Automated Software Engineering, 2010, 17(1): 87-116.
 - [131] Zhang Y, Harman M, Finkelstein A, et al. Comparing the performance of metaheuristics for the analysis of multi-stakeholder tradeoffs in requirements optimisation [J]. Information and Software Technology, 2011, 53(7): 761-773.
 - [132] Veerappa V. Clustering Methods for Requirements Selection and Optimisation [D]. UCL(University College London), 2013.
 - [133] Pitangueira A M, Maciel R S P, de Oliveira Barros M, et al. A systematic review of software requirements selection and prioritization using SBSE approaches [M]. Search Based Software Engineering. Springer Berlin Heidelberg, 2013: 188-208.
 - [134] RenJ: Sensitivity Analysis in Multi-Objective Next Release Problem and Fairness Analysis in Software Requirements Engineering [D]. Master's thesis, DCS/PSE, King's College London, London, 2007.
 - [135] A Finkelstein, M Harman, A Mansouri, J Ren, Y Zhang. Fairness analysis in requirements assignments [C]. 16th IEEE International Requirements Engineering Conference, Los Alamitos, California, USA. IEEE Computer Society Press, 2008.
 - [136] Y Zhang, A Finkelstein, M Harman. Search based requirements optimisation: Existing work and challenges [C]. International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ '08), Montpellier, France. Springer LNCS, 2008.
 - [137] Yuanyuan Zhang, Mark Harman, Soo Ling Lim, Empirical evaluation of search based requirements interaction management [J]. Information and Software Technology, ISSN 0950-5849. 2013, 55(1): 126-152.
 - [138] Outi Raiha. A survey on search-based software design [J]. Computer Science Review, 2010, 4(4): 203-249.
 - [139] Christopher L. Simons and Ian C Parmee. A cross-disciplinary technology transfer for search-based evolutionary computing: from engineering design to software engineering design [J]. Engineering Optimization, 2007, 39(5): 631-648.
 - [140] Christopher L. Simons, Ian C Parmee. Single and multi-objective genetic operators in object-oriented conceptual software design [C]. Proceedings of the 8th annual Conference on Genetic and Evolutionary Computation, pages 1957-1958, Seattle, Washington, USA, 8-12 July 2006.
 - [141] Christopher L Simons, Ian C Parmee. User-centered, evolutionary search in conceptual software design [C]. Proceedings of the IEEE Congress on Evolutionary Computation, pages 869-876, Hong Kong, China, 1-6 June 2008.
 - [142] Mark O'Keeffe, Mel O Cinneide. A stochastic approach to automated design improvement [C]. Proceedings of the 2nd International Conference on Principles, Practice of Programming in Java, pp. 59-62, Kilkenny City, Ireland, 16-18 June 2003.
 - [143] Mark O'Keeffe, Mel O Cinneide. Towards automated design improvement through combinatorial optimisation [C]. Proceedings of the 26th International Conference on Software Engineering and Workshop on Directions in Software Engineering Environments, pages 75-82, Edinburgh, UK, 23-28 May 2004.
 - [144] Michael Bowman, Lionel C Briand, Yvan Labiche. Solving the class responsibility assignment problem in object-oriented analysis with multiobjective genetic algorithms [R]. Technical Report SCE-07-02,

August 2008.

- [145] Mehdi Amoui, Siavash Mirarab, Sepand Ansari, Caro Lucas. A genetic algorithm approach to design evolution using design pattern transformation [J]. *International Journal of Information Technology and Intelligent Computing*, 2006, 1(2): 235-244.
- [146] Outi Raiha. Applying genetic algorithms in software architecture design [D]. Master's thesis, Department of Computer Sciences, University of Tampere, February 2008.
- [147] Outi Raiha. Genetic Synthesis of Software Architecture [D]. PhD thesis, University of Tampere, Finland, September 2008.
- [148] Heather J Goldsby, Betty H C Cheng. Automatically generating behavioral models of adaptive systems to address uncertainty [C]. *Proceedings of the 11th International Conference on Model Driven Engineering Languages and Systems*, pages 568-583, Toulouse, France, 28 September-3 October 2008.
- [149] Heather J Goldsby, Betty H C Cheng. Avida-mde: a digital evolution approach to generating models of adaptive software behavior [C]. *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, pages 1751-1758, Atlanta, GA, USA, 12-16 July 2008.
- [150] Heather J Goldsby, Betty H C Cheng, Philip K. McKinley, David B Knoester, Charles A Ofria. Digital evolution of behavioral models for autonomic systems [C]. *Proceedings of the 2008 International Conference on Autonomic Computing*, pages 87-96, Chicago, IL, USA, 2-6 June 2008.
- [151] Simon M Lucas, T. Jeff Reynolds. Learning deterministic finite automata with a smart state labeling evolutionary algorithm [J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005, 27(7): 1063-1074.
- [152] Gerardo Canfora, Massimiliano Di Penta, Raffaele Esposito, Maria Luisa Villani. An approach for qos-aware service composition based on genetic algorithms [C]. *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation*, pages 1069-1075, Washington, D C, USA, 25-29 June 2005.
- [153] Gerardo Canfora, Massimiliano Di Penta, Raffaele Esposito, Maria Luisa Villani. Qos-aware replanning of composite web services [C]. *Proceedings of 2005 IEEE International Conference on Web Services*, pages 121-129, Orlando, FL, USA, 11-15 July 2005.
- [154] Michael C Jaeger, Gero Muhl. Qos-based selection of services: The implementation of a genetic algorithm [C]. *Proceedings of Kommunikation in Verteilten Systemen 2007 Workshop: Service-Oriented Architectures and Service-Oriented Computing*, 2007.
- [155] Taghi M Khoshgoftar, Yi Liu, Naeem Seliya. A multi-objective module-order model for software quality enhancement [J]. *IEEE Transactions on Evolutionary Computation*, 2004, 8(6): 593-608.
- [156] Taghi M Khoshgoftar, Yi Liu, Naeem Seliya. Module-order modeling using an evolutionary multi-objective optimization approach [C]. *Proceedings of the 10th IEEE International Symposium on Software Metrics*, pages 159-169, Chicago, USA, 14-16 September 2004.
- [157] Chengwen Zhang, Sen Su, Junliang Chen. A novel genetic algorithm for qos-aware web services selection [C]. *Proceedings of the 2nd International Workshop on Data Engineering Issues in E-Commerce and Services*, volume 4055, pages 224-235, San Francisco, CA, USA, 26 June 2006.
- [158] Chengwen Zhang, Sen Su, Junliang Chen. Diga: Population diversity handling genetic algorithm for qos-aware web services selection [J]. *Computer Communications*, 2007, 30(5): 1082-1090.
- [159] Sen Su, Chengwen Zhang, Junliang Chen. An improved genetic algorithm for web services selection [C]. *Proceedings of the 7th IFIP WG6.1 International Conference on Distributed Applications and Interoperable*

- Systems, Paphos, Cyprus, 6-8 June 2007. 2007, 4531: 284-295.
- [160] Yue Ma, Chengwen Zhang. Quick convergence of genetic algorithm for qos-driven web service selection[J]. Computer Networks, 2008, 52(5): 1093-1104.
- [161] Lei Cao, Minglu Li, Jian Cao. Cost-driven web service selection using genetic algorithm[C]. Proceedings of the 1st International Workshop on Internet and Network Economics, pages 906-915, Hong Kong, China, 15-17 December 2005.
- [162] K Vijayalakshmi, N Ramaraj, R Amuthakkannan. Improvement of component selection process using genetic algorithm for component-based software development[J]. International Journal of Information Systems and Change Management, 2008, 3(1): 63-80.
- [163] Lili Yang, Bryan F Jones, Shuang-Hua Yang. Genetic algorithm based software integration with minimum software risk[J]. Information and Software Technology, 2006, 48(3): 133-141.
- [164] Nicolas Desnos, Marianne Huchard, Guy Tremblay, Christelle Urtado, Sylvain Vauttier. Search-based many-to-one component substitution[J]. Journal of Software Maintenance and Evolution: Research and Practice, 2008, 20(5): 321-344.
- [165] Vittorio Cortellessa, Ivica Crnkovic, Fabrizio Marinelli, Pasqualina Potena. Experimenting the automated selection of cots components based on cost and system requirements[J]. Journal of Universal Computer Science, 2008, 14(8): 1228-1255.
- [166] Michael Kuperberg, Klaus Krogmann, Ralf Reussner. Performance prediction for black-box components using reengineered parametric behavior models[C]. Proceedings of the 11th International Symposium on Component-Based Software Engineering, Karlsruhe, Germany, 14-17 October 2008. 2008, 5282: 48-63.
- [167] Robert Feldt. An experiment on using genetic programming to develop multiple diverse software variants[R]. Technical Report 98-13, Gothenburg, Sweden, September 1998.
- [168] Robert Feldt. Generating multiple diverse software versions with genetic programming[C]. Proceedings of the 24th EUROMICRO Conference, Vasteras, Sweden, 25-27 August 1998. 1998, 1: 387-394.
- [169] Robert Feldt. Generating multiple diverse software versions with genetic programming-an experimental study[C]. IEE Proceedings-Software, 1998, 145(6): 228-236.
- [170] Gerassimos Barlas, Khaled El-Fakih. A ga-based movie-on-demand platform using multiple distributed servers[J]. Multimedia Tools and Applications, 2008, 40(3): 361-383.
- [171] Sylvain Chardigny, Abdelhak Seriai, Dalila Tamzalit, Mourad Oussalah. Quality-driven extraction of a component-based architecture from an object-oriented system[C]. Proceedings of the 12th European Conference on Software Maintenance and Re engineering, pages 269-273, Athens, Greece, 1-4 April 2008.
- [172] Vibhu Saujanya Sharma, Pankaj Jalote. Deploying software components for performance[C]. Proceedings of the 11th International Symposium on Component-Based Software Engineering, pages 32-47, Karlsruhe, Germany, 14-17 October 2008.
- [173] C Ryan. Automatic re-engineering of software using genetic programming[J]. Springer, 1999, 2.
- [174] K P Williams. Evolutionary algorithms for automatic parallelization[D]. Doctoral dissertation, University of Reading, 1998.
- [175] D Fatiregun, M Harman, R M Hierons. Evolving transformation sequences using genetic algorithms[J]. Source Code Analysis and Manipulation, 2004, pages 65-74, Sep. 2004.
- [176] M O'Keeffe, M O'Conneide. Search-based software maintenance[C]. Conference on Software Maintenance

- and Reengineering(CSMR '06) , pages 249-260 , Mar. 2006.
- [177] S Bouktif, G. Antoniol, E Merlo, M Neteler. A novel approach to optimize clone refactoring activity[C]. Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, pages 1885-1892, July. 2006.
 - [178] O Seng, M Bauer, M Biehl, G. Pache(2005, June). Search-based improvement of subsystem decompositions [C]. Proceedings of the 2005 conference on Genetic and evolutionary computation . ACM, 2005; 1045-1051.
 - [179] A Ghannem, G El Boussaidi, M Kessentini. Model refactoring using interactive genetic algorithm[C]. Search Based Software Engineering(SBSE '2013) , pages 96-110, Springer Berlin Heidelberg.
 - [180] R Lämmel. Towards generic refactoring[C]. Proceedings of the 2002 ACM SIGPLAN workshop on Rule-based programming . 2002; 15-28.
 - [181] H Li, S Thompson. Comparative study of refactoring haskell and erlang programs[C]. Source Code Analysis and Manipulation,(SCAM '2006) , pages 197-206, Sep. 2006.
 - [182] M Harman. Open problems in testability transformation[C]. Software Testing Verification and Validation Workshop, (ICSTW'2008) , pages 196-209) , April 2008.
 - [183] M Harman. Refactoring as testability transformation[C]. Software Testing, Verification and Validation Workshops(ICSTW '2011) , pages 414-421 , Mar. 2011.
 - [184] M Harman, L Tratt. Pareto optimal search based refactoring at the design level[C]. Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO' 2007) , pages 1106-1113, July 2007.
 - [185] M K O'Keeffe, M O Cinneide. Getting the most from search-based refactoring[C]. Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO' 2007) , pages 1114-1120, July 2007.
 - [186] E Koc, N Ersoy, A Andac, Z S Camlidere, I Cereci, H Kilic. An empirical study about search-based refactoring using alternative multiple and population-based search techniques [J]. Computer and Information Sciences II, London: Springer, 2012; 59-66.
 - [187] J Tao, N Gold, M Harman, Z Li. Locating dependence structures using search based slicing [J]. Information and Software Technology.
 - [188] N Gold, M Harman, Z Li, K Mahdavi. A search based approach to overlapping concept boundaries[C]. 22nd International Conference on Software Maintenance (ICSM 06) , pages 310-319, Philadelphia, Pennsylvania, USA, Sept. 2006.
 - [189] Alvarez-Valdes R, Crespo E, Tamarit JM, Villa F. A scatter search algorithm for project scheduling under partially renewable resources[J]. Journal of Heuristics, 2006, 12(1-2) : 95-113.
 - [190] G. Antoniol, M Di Penta, M Harman. A robust search-based approach to project management in the presence of abandonment, rework, error and uncertainty [C]. 10th International Software Metrics Symposium (METRICS 2004) , Los Alamitos, California, USA, Sept. 2004. IEEE Computer Society Press, 2004; 172-183.
 - [191] Di Penta M, HarmanM, Antoniol G. The use of search-based optimization techniques to schedule and staff software projects: an approach and an empirical study[J]. Software: Practice and Experience, 2011, 41 (5) : 495-519.
 - [192] Jifeng Xuan, He Jiang, Zhilei Ren, Weiqin Zou. Developer Prioritization in Bug Repositories [C].

- Proceedings of 34th International Conference on Software Engineering(ICSE 2012), Zurich, Switzerland. June 2-9, 2012. 2012: 25-35.
- [193] Di Penta M, Harman M, Antoniol G, Qureshi F. The effect of communication overhead on software maintenance project staffing: a search-based approach[C]. In Software Maintenance, 2007. ICSM 2007. IEEE, 2007: 315-324.
- [194] Stylianou C, Andreou A S. A multi-objective genetic algorithm for intelligent software project scheduling and team staffing[J]. Intelligent Decision Technologies, 2013, 7(1): 59-80.
- [195] Chicano F, Luna F, Nebro A J, Alba E. Using multi-objective metaheuristics to solve the software project scheduling problem [C]. Proceedings of the 13th annual conference on Genetic and evolutionary computation . ACM, 2012, 1915-1922.
- [196] Chen W N, Zhang J. Ant colony optimization for software project scheduling and staffing with an event-based scheduler[J]. Software Engineering, IEEE Transactions on, 2013, 39(1): 1-17.
- [197] Burgess C J, Lefley M. Can genetic programming improve software effort estimation? A comparative evaluation[J]. SERIES ON SOFTWARE ENGINEERING AND KNOWLEDGE ENGINEERING, 2005, 16, 95.
- [198] Bardsiri V K, Jawawi D N A, Hashim S ZM, Khatibi E. A PSO-based model to increase the accuracy of software development effort estimation[J]. Software Quality Journal, 2013, 21(3): 501-526.
- [199] GAntoniol, M D Penta, M Harman. Search-based techniques applied to optimization of project planning for a massive maintenance project [C]. 21st IEEE International Conference on Software Maintenance, Los Alamitos, California, USA, 2005. IEEE Computer Society Press, 2005: 240-249.
- [200] Dolado J J. On the problem of the software cost function[J]. Information and Software Technology, 2001, 43(1): 61-72.
- [201] Azar D. A Genetic Algorithm for Improving Accuracy of Software Quality Predictive Models: A Search-based Software Engineering Approach [J]. International Journal of Computational Intelligence and Applications, 2010, 9(02): 125-136.
- [202] J Aguilar-Ruiz, I Ramos, J C Riquelme, M Toro. An evolutionary approach to estimating software development projects[J]. Information and Software Technology, 2001, 43(14): 875-882.
- [203] C Kirsopp, M Shepperd, J Hart. Search heuristics, case-based reasoning and software project effort prediction[C]. GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, San Francisco, CA 94104, USA, 9-13 July 2002. Morgan Kaufmann Publishers, 2002: 1367-1374.
- [204] J Wegener A Baresel, H Sthamer. Evolutionary Test Environment for Automatic Structural Testing[J]. Information and Software Technology, 2001, 43(14): 841-854.
- [205] Windisch A Wappler S, Wegener J. Applying particle swarm optimization to software testing [C]. Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation. 2007.
- [206] Harman M, Hassoun Y, Lakhota K, McMin P, Wegener J. The impact of input domain reduction on search-based test data generation [C]. Proceedings of the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering. New York: ACM Press, 2007: 1121-1128.
- [207] Buhler O, Wegener J. Evolutionary functional testing[J]. Comput. Oper. Res. 35, 10, 3144-3160. 2008.
- [208] C Cadar, P Godefroid, S Khurshid, C S Pasareanu, K Sen, N Tillmann, W Visser. Symbolic Execution for Software Testing in Practice: Preliminary Assessment [C]. Proceedings of the 33rd International

- Conference on Software Engineering, pp. 1066- 1071, Hawaii, USA, 21-28 May 2011.
- [209] Xie T, Tillmann N, Dehalleux P, Schulte W. Fitness-Guided path exploration in dynamic symbolic execution[R]. Tech. rep. MSR-TR-2008-123, Microsoft Research. September. 2008.
- [210] K Lakhotia, N Tillmann, M Harman, J de Halleux. FloPSy Search-Based Floating Point Constraint Solving for Symbolic Execution[C]. Proceedings of the 22nd IFIP International Conference on Testing Software and Systems, Natal, Brazil, November 2010. Springer, 2010, 6435(of LNCS): 142- 157.
- [211] S Yoo, R Nilsson M Harman. Faster Fault Finding at Google using Multi Objective Regression Test Optimisation[C]. Proceedings of the 8th European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, Szeged, Hungary, 5-9 September 2011.
- [212] Yoo S, Harman M, Ur S. Measuring and improving latency to avoid test suite wear out[C]. Proceedings of the IEEE International Conference on Software Testing, Verification, and Validation Workshops. IEEE, 2009: 101-110.
- [213] P Baker, M Harman K Steinhofel, A Skaliotis. Search Based Approaches to Component Selection and Prioritization for the Next Release Problem[C]. Proceedings of the 22nd International Conference on Software Maintenance, pp. 176-185, Philadelphia, Pennsylvania, USA, 24-27 September 2006.
- [214] Y Zhang, E Alba, J J Durillo, S Eldh, M Harman. Today/Future Importance Analysis[C]. Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, pp. 1357-1364, Portland, Oregon, USA, 7-11 July 2010.
- [215] Kiran Lakhotia, Mark Harman, Hamilton Gross. AUSTIN: An open source tool for search based software testing of C programs[J]. Information and Software Technology, 2013, 55(1): 112-125.
- [216] 毛澄映, 喻新欣, 薛云志. 基于粒子群优化的测试数据生成及其实证分析[J]. 计算机研究与发展, 2014, 04: 824-837.
- [217] 史娇娇, 姜淑娟, 韩寒, 王令赛. 自适应粒子群优化算法及其在测试数据生成中的应用研究[J]. 电子学报, 2013, 08: 1555-1559.
- [218] 姚香娟, 巩敦卫. 基于目标语句占优关系的软件可测试性转化[J]. 电子学报, 2013, 12: 2523-2528.
- [219] 张岩, 巩敦卫. 基于搜索空间自动缩减的路径覆盖测试数据进化生成[J]. 电子学报, 2012, 05: 1011-1016.
- [220] 巩敦卫, 任丽娜. 回归测试数据进化生成[J]. 计算机学报, 2014, 03: 489-499.
- [221] 巩敦卫, 张岩. 一种新的多路径覆盖测试数据进化生成方法[J]. 电子学报, 2010, 06: 1299-1304.
- [222] 舒挺, 刘良桂, 徐伟强, 李文书. 自适应 EFSM 可执行测试序列生成[J]. 计算机研究与发展, 2012, 06: 1211-1219.
- [223] 刘新忠, 徐高潮, 胡亮, 付晓东, 董玉双. 一种基于约束的变异测试数据生成方法[J]. 计算机研究与发展, 2011, 04: 617-626.
- [224] 侯可佳, 白晓颖, 陆皓, 李树芳, 周立柱. 基于接口语义契约的 Web 服务测试数据生成[J]. 软件学报, 2013, 09: 2020-2041.
- [225] 谢晓园, 徐宝文, 史亮, 聂长海. 面向路径覆盖的演化测试用例生成技术(英文)[J]. 软件学报, 2009, 12: 3117-3136.
- [226] 王建民, 蔡媛. 基于维持种群多样性的测试数据生成算法的研究[J]. 计算机研究与发展, 2012, 05: 1039-1048.
- [227] 张岩, 巩敦卫. 基于稀有数据扑捉的路径覆盖测试数据进化生成方法[J]. 计算机学报, 2013, 12: 2429-2440.

- [228] 田甜, 巩敦卫. 消息传递并行程序路径覆盖测试数据生成问题的模型及其进化求解方法[J]. 计算机学报, 2013, 11: 2212-2223.
- [229] 查日军, 张德平, 聂长海, 徐宝文. 组合测试数据生成的交叉熵与粒子群算法及比较[J]. 计算机学报, 2010, 10: 1896-1908.
- [230] 董国伟, 聂长海, 徐宝文. 基于程序路径分析的有效蜕变测试[J]. 计算机学报, 2009, 05: 1002-1013.
- [231] 顾庆, 唐宝, 陈道蓄. 一种面向测试需求部分覆盖的测试用例集约简技术[J]. 计算机学报, 2011, 05: 879-888.
- [232] 陈翔, 陈继红, 鞠小林, 顾庆. 回归测试中的测试用例优先排序技术述评[J]. 软件学报, 2013, 08: 1695-1712.
- [233] 张智轶, 陈振宇, 徐宝文, 杨瑞. 测试用例演化研究进展[J]. 软件学报, 2013, 04: 663-674.
- [234] 聂剑平, 曹旭, 钱越英, 陈昱松. 基于 I/O 关系的适应性随机测试[J]. 计算机研究与发展, 2010, S1: 56-63.
- [235] 夏亚梅, 程渤, 陈俊亮, 孟祥武, 刘栋. 基于改进蚁群算法的服务组合优化[J]. 计算机学报, 2012, 02: 2270-2281.
- [236] 王尚广, 孙其博, 杨放春. 基于全局 QoS 约束分解的 Web 服务动态选择[J]. 软件学报, 2011, 07: 1426-1439.
- [237] 温涛, 盛国军, 郭权, 李迎秋. 基于改进粒子群算法的 Web 服务组合[J]. 计算机学报, 2013, 05: 1031-1046.
- [238] 邓亮, 赵进, 王新. 基于遗传算法的网络编码优化[J]. 软件学报, 2009, 08: 2269-2279.
- [239] 谢晓芹, 宋超臣, 张志强. 一种基于推荐网络和蚁群算法的服务发现方法[J]. 计算机学报, 2010, 11: 2093-2103.
- [240] 黄发良, 张师超, 朱晓峰. 基于多目标优化的网络社区发现方法[J]. 软件学报, 2013, 09: 2062-2077.
- [241] 曾明霏, 余顺争. P2P 网络服务器部署方案及其启发式优化算法[J]. 软件学报, 2013, 09: 2226-2237.
- [242] 梁亚澜, 聂长海. 覆盖表生成的遗传算法配置参数优化[J]. 计算机学报, 2012, 07: 1522-1538.
- [243] 聂长海, 蒋静. 覆盖表生成的可配置贪心算法优化[J]. 软件学报, 2013, 07: 1469-1483.
- [244] 严秋玲, 孙莉, 王梅, 乐嘉锦, 刘国华. 列存储数据仓库中启发式查询优化机制[J]. 计算机学报, 2011, 10: 2018-2026.
- [245] Mark Harman, Afshin Mansouri, Yuanyuan Zhang. Search Based Software Engineering: Trends, Techniques and Applications[J]. ACM Computing Surveys, 2012, 45(1): Article 11.
- [246] Mark Harman, Kiran Lakhotia, Jeremy Singer, David R White, Shin Yoo. Cloud engineering is Search Based Software Engineering too[J]. Journal of Systems and Software, 2013, 86(9): 2225-2241.

附录：基于搜索的软件工程相关资源

1. 基于软件工程的文献库: <http://www.sebase.orgsbsepublications/>
2. 基于搜索的软件工程领域的相关综述文章 (Survey)

- 测试数据生成

Phil McMinn. Search- based software test data generation: a survey [J]. *Software Testing, Verification and Reliability*, 2004, 14 (2): 105-156.

Shaukat Ali, Lionel Briand, Hadi Hemmati and Rajwinder Panesar- Walawege. A Systematic Review of the Application and Empirical Investigation of Search-Based Test-Case Generation [J]. *IEEE Transactions on Software Engineering*, 2010, 36 (6): 742-762.

Wasif Afzal, Richard Torkar and Robert Feldt. A Systematic Review of Search- based Testing for Non- Functional System Properties [J]. *Information and Software Technology*, 2009, 51 (6): 957-976.

- 软件设计

Outi Räihä. A Survey on Search- Based Software Design [J]. *Computer Science Review*, 2010, 4 (4): 203-249.

- 发展趋势

Mark Harman. The Current State and Future of Search Based Software Engineering [C]. *ICSE Future of Software Engineering*, 2007: 342-357.

Mark Harman, Afshin Mansouri and Yuanyuan Zhang. Search Based Software Engineering: Trends, Techniques and Applications [J]. *ACM Computing Surveys*, 2012, 45 (1): Article 11.

3. 基于搜索的软件工程相关工具

- Austin 针对 C 程序的，基于搜索的软件测试数据生成工具。

<http://code.google.com/p/austin-sbst/>

- Cocotest 利用搜索的方法对连续控制器进行自动化的 MIL (Model-In-the-Loop) 测试的工具。<https://sites.google.com/site/cocotesttool/>

- PEX 微软针对 .NET 的白盒测试开源框架。结合了符号执行与搜索技术。可以作为 Visual Studio 的插件，自动生成高覆盖率的测试数据。

<http://research.microsoft.com/en-us/projects/pex/>

- GenProg 利用遗传编程技术对 C 语言程序进行故障自动修复的工具。

<http://dijkstra.cs.virginia.edu/genprog/#problem>

- EvoSuite 针对 java 程序的测试数据生成工具。

<http://www.evosuite.org/>

- JavaPathFinder 对 java 程序进行执行、检验的工具，可用于探测多线程程序所有可能的执行路径。

<http://javapathfinder.sourceforge.net/>

- Milu 基于搜索的高阶变异测试 (Higher Order Mutation Testing) 工具。

<http://www0.cs.ucl.ac.uk/staff/y.jiaMilu>

作者简介

李 征 博士，北京化工大学信息科学与技术学院教授、博士生导师、教育部新世纪优秀人才计划获得者。中国计算机学会（CCF）高级会员，软件工程专委会和容错计算专委会委员，担任 STVR、JSS、JSEP 等国际期刊客座编辑，IEEE SCAM 2012 程序委员会主席，ICSM、WCRE、GECCO 和 RT 等多个国际会议的程序委员会委员，是中国基于搜索的软件工程研讨会（CSBSE）发起人。主要研究领域：基于搜索的软件工程、程序源代码分析，目前已发表文章被引次数超过 500 次。



巩敦卫 博士，中国矿业大学教授，博士生导师，教育部新世纪优秀人才支持计划入选者，江苏省“六大人才高峰”高层次人才，江苏省“333 高层次人才培养工程”培养对象，全国煤炭青年科技奖获得者，IEEE 会员，中国计算机学会会员。研究方向为：基于搜索的软件工程、智能优化与控制，主持国家自然科学基金 5 项、国家“973”计划子课题 1 项，研究成果获省部级科技奖励 6 项。



聂长海 博士，南京大学计算机科学与技术系教授，博士生导师。中国计算机学会（CCF）高级会员，软件工程专委会委员。一直从事软件工程，特别是软件测试领域的教学和科研工作。发表软件测试学术论文 70 多篇，包括软件工程领域 A 类期刊 TOSEM 及计算机领域最具影响力的期刊 ACM Computing Surveys。主讲软件测试等 10 多门课程，独立编写并正式出版了《软件测试的概念与方法》，获得过发明专利和多项软件著作权，主持完成过国家自然科学基金项目、科技部 863 专题项目和江苏省自然科学基金项目，参与的项目多次获得过省部级科研奖。



江 贺 博士，大连理工大学软件学院教授、博士生导师、教育部新世纪优秀人才计划获得者。Applied Intelligence Journal 客座编辑，Frontiers of Computer Science 青年编委，中国计算机学会软件工程专委会，中国计算机学会计算机应用专委会委员。主要研究领域：基于搜索的软件工程、软件仓库挖掘。先后在 IEEE Transaction 系列汇刊（TSE，TKDE，TSMCB，TCYB）、EC、中国科学等期刊及 ICSE、GECCO 等国际会议发表论文 60 余篇。

