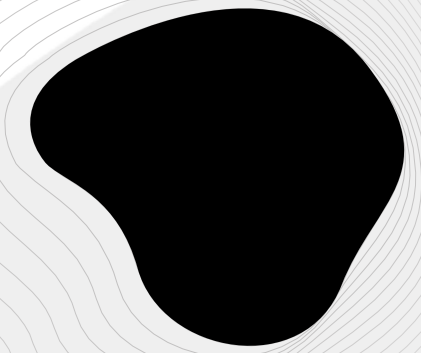
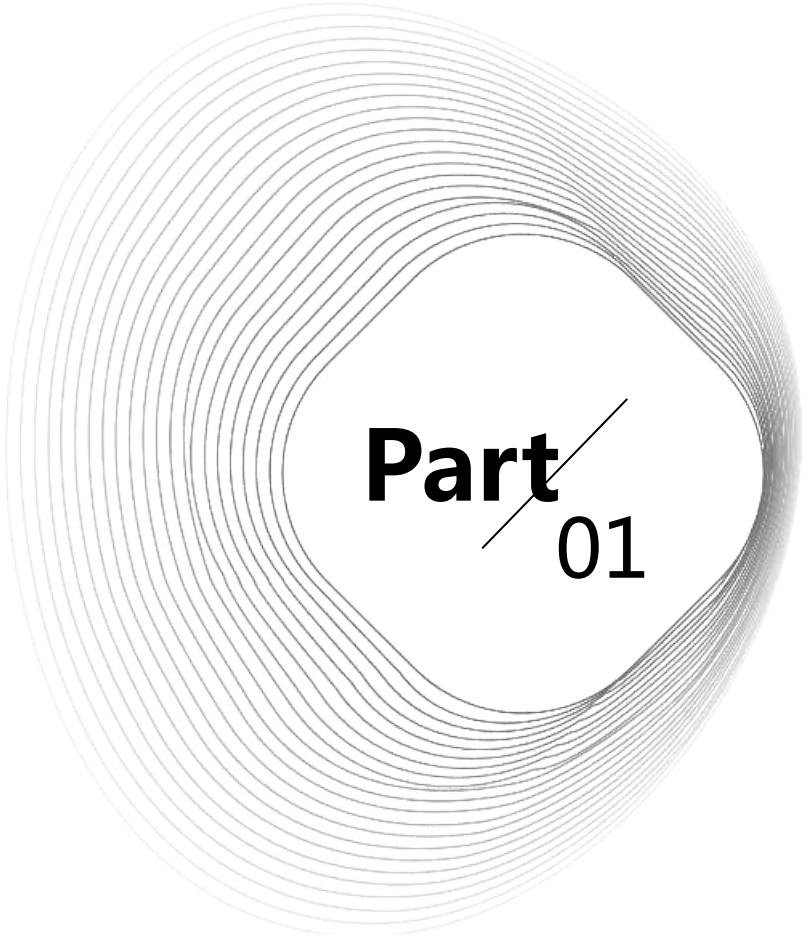


基于合约逐级求解的系统测试用例生成

汇报人：

2024-01-29

目录



Part 01

引言

目的和背景

1

提高测试用例生成效率

基于合约逐级求解的方法能够自动化生成系统测试用例，减少人工编写测试用例的时间和成本。

2

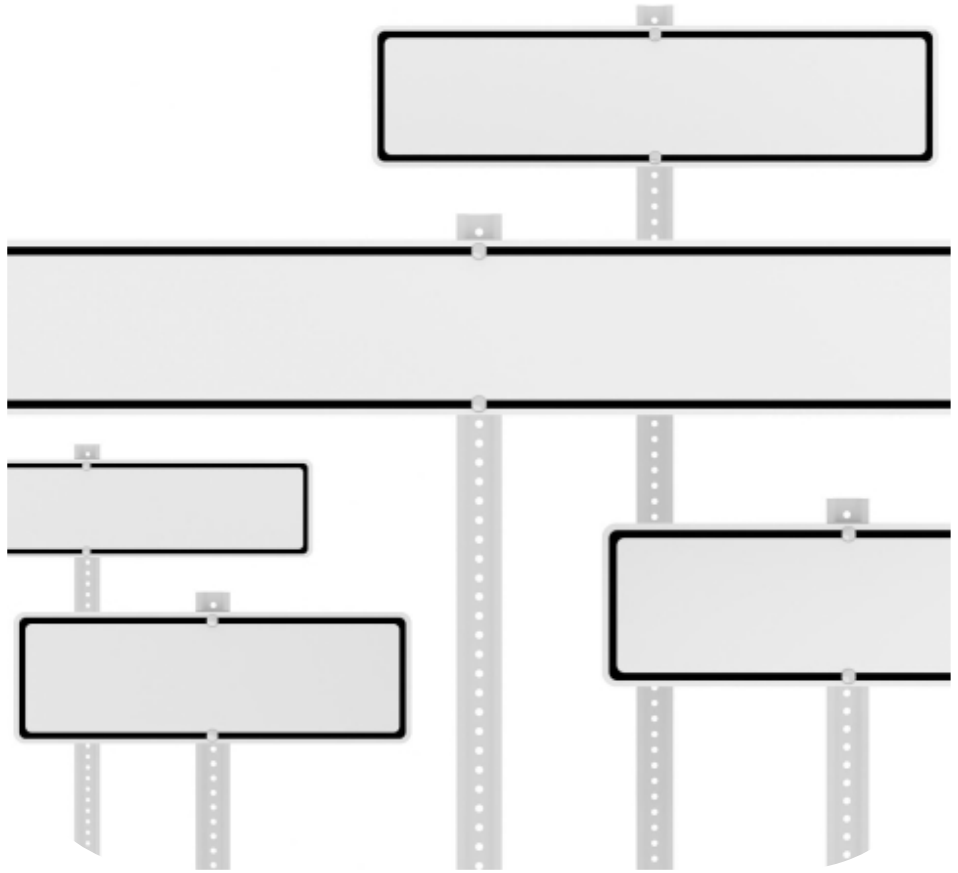
保证系统质量

通过合约对系统行为进行约束和验证，能够发现系统中的潜在问题，提高系统的稳定性和可靠性。

3

支持敏捷开发

在敏捷开发过程中，系统需求经常变更，基于合约逐级求解的方法能够快速响应需求变更，生成相应的测试用例。



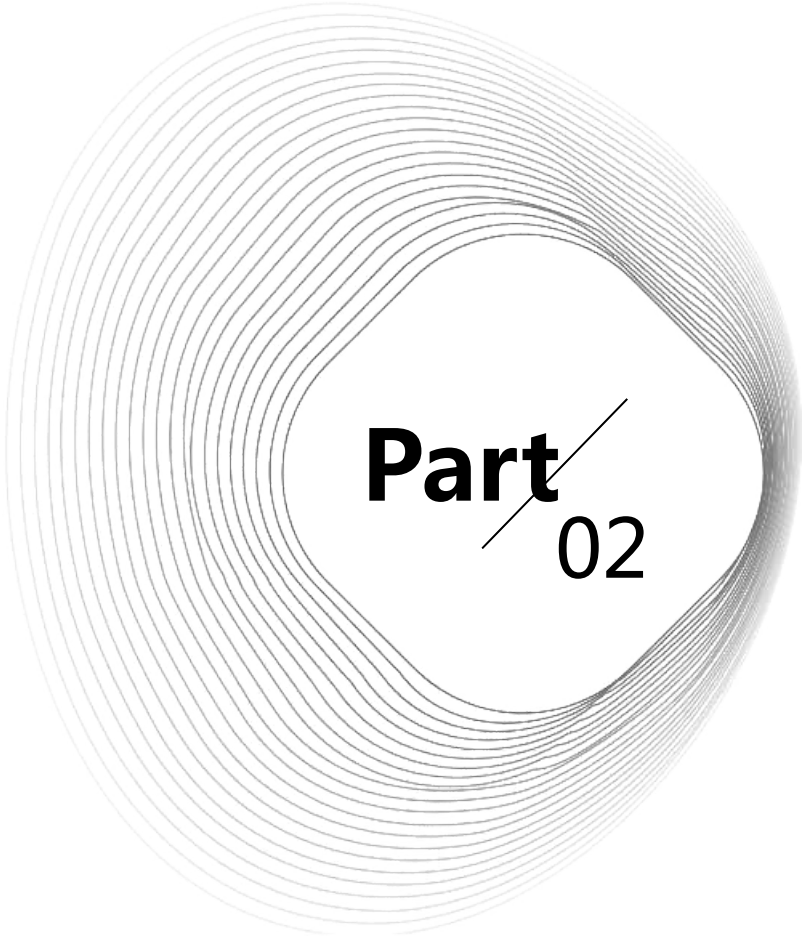
研究现状和意义

研究现状

目前，基于合约的测试用例生成方法已经得到了广泛的研究和应用，包括基于形式化方法、基于模糊测试、基于符号执行等多种方法。这些方法在不同的应用场景下具有不同的优缺点，需要根据实际需求进行选择和改进。

研究意义

随着软件系统的复杂性和规模不断增加，传统的测试用例生成方法已经无法满足需求。基于合约逐级求解的方法能够结合形式化方法和测试技术，提高测试用例生成的准确性和效率，为软件系统的质量保障提供有力支持。同时，该方法还能够促进软件开发过程的规范化和标准化，提高软件开发的效率和质量。



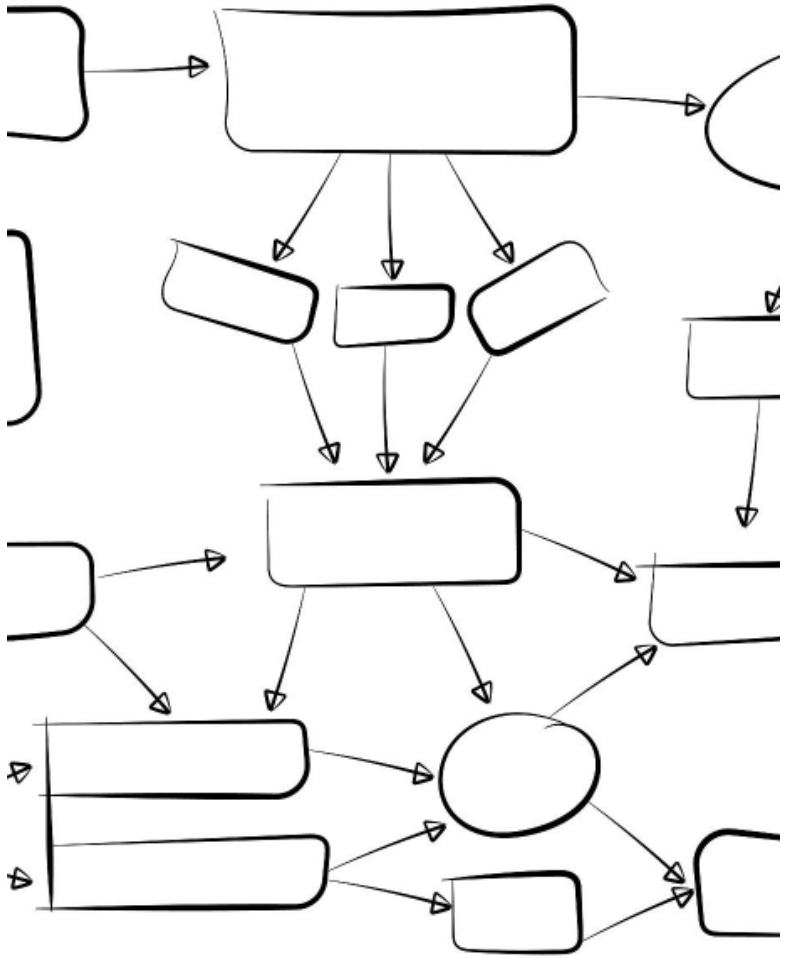
Part 02

合约逐级求解概述

合约定义及分类

合约定义

合约是一种具有法律效力的协议，规定了双方或多方之间的权利和义务。在软件测试领域，合约通常指系统或软件组件之间的交互协议。



合约分类

根据合约的复杂度和涉及方的数量，合约可分为简单合约和复杂合约。简单合约涉及两个参与方，而复杂合约可能涉及多个参与方和复杂的交互逻辑。

逐级求解原理及流程

逐级求解原理

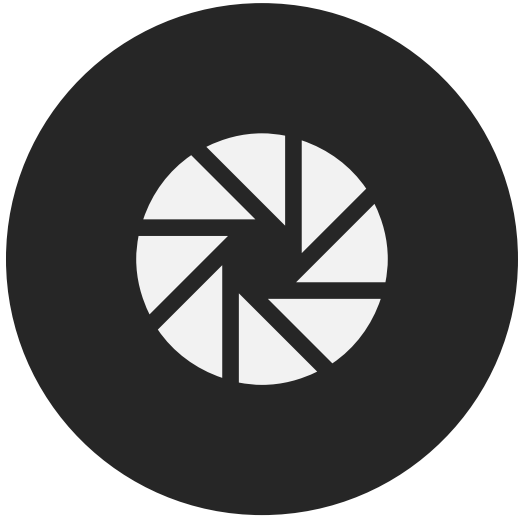
逐级求解是一种分而治之的策略，将复杂问题分解为一系列相对简单的子问题，通过逐步解决子问题来最终解决整个问题。在合约测试中，逐级求解将复杂的合约拆分为多个层级，每个层级对应一个或多个测试用例。

1. 分析合约

对合约进行详细分析，识别出关键交互点和潜在风险。

2. 设计测试用例

针对每个关键交互点和潜在风险，设计相应的测试用例。



3. 构建测试环境

搭建符合合约要求的测试环境，包括所需的硬件、软件和网络配置。

4. 执行测试用例

按照设计好的测试用例，逐一执行测试，并记录测试结果。

5. 分析测试结果

对测试结果进行分析，确定是否满足合约要求，并针对不满足要求的部分进行改进。

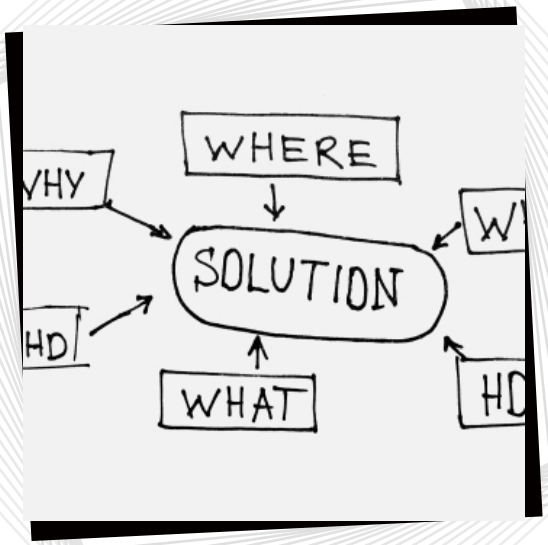
关键技术分析

合约建模技术

利用形式化方法或可视化建模工具对合约进行建模，以便更好地理解和分析合约的结构和行为。

结果分析与报告技术

对测试结果进行深入分析，识别潜在问题和改进点，并生成详细的测试报告以供决策参考。

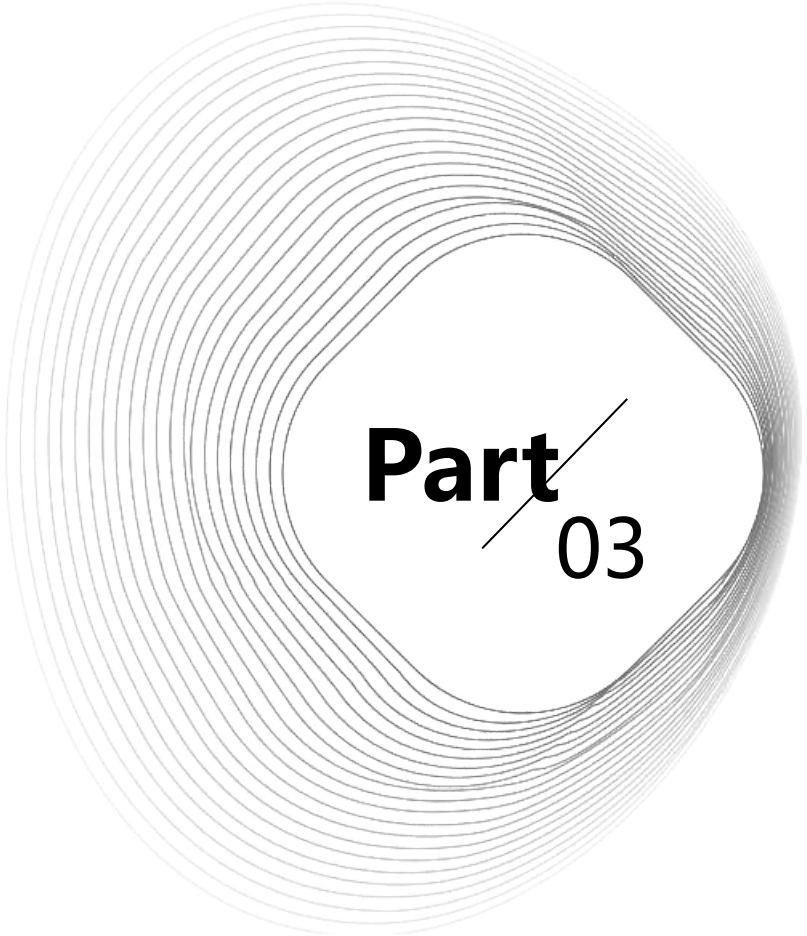


测试用例设计技术

采用等价类划分、边界值分析、因果图等测试用例设计技术，针对合约的特点设计高效且全面的测试用例。

测试执行与监控技术

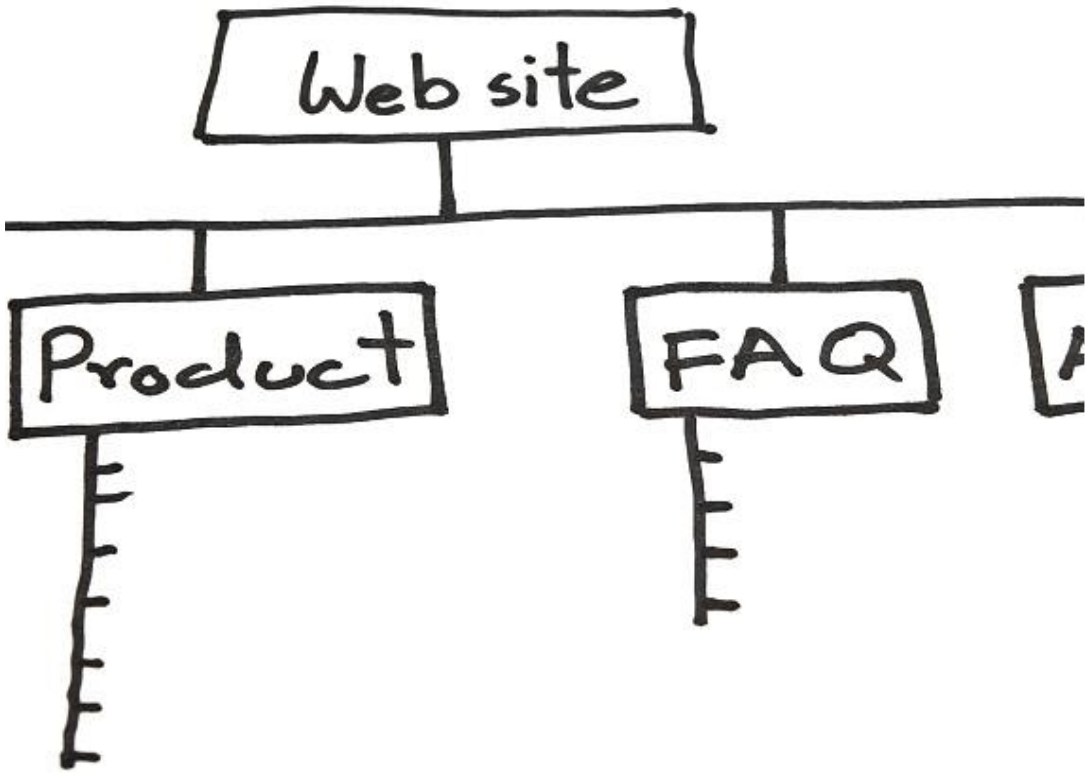
利用自动化测试工具或框架执行测试用例，并实时监控测试过程，确保测试的准确性和效率。



Part 03

系统测试用例生成方法

传统测试用例生成方法



等价类划分法

将输入域划分为若干个等价类，从每个等价类中选取一个代表值进行测试。这种方法可以减少测试用例的数量，但可能无法覆盖所有边界情况。

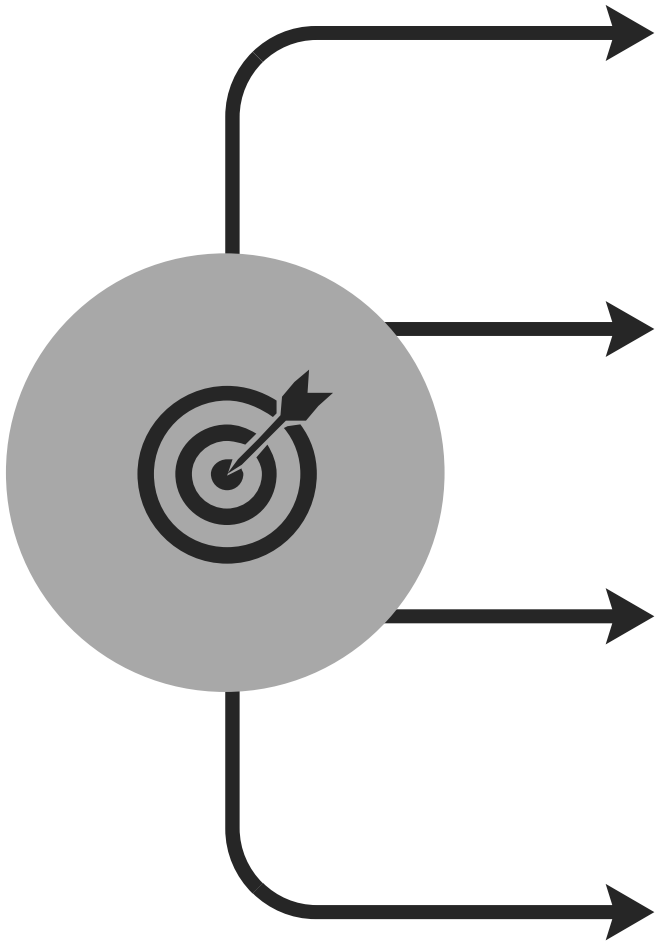
边界值分析法

针对输入域的边界值进行测试，以发现潜在的错误。这种方法对于检查边界条件和处理异常情况非常有效。

因果图法

通过因果图描述输入与输出之间的关系，并根据因果图生成测试用例。这种方法可以系统地分析多个输入条件之间的组合情况。

基于合约逐级求解的测试用例生成方法



合约定义

明确系统或模块的输入、输出及其约束条件，形成合约。合约可以是形式化的规格说明或非形式化的文档描述。

合约分解

将复杂的合约分解为更简单的子合约，以便逐级求解。子合约可以是函数、方法或模块的规格说明。

逐级求解

从最低层级的子合约开始，逐级生成满足合约要求的测试用例。每一级的测试用例都要验证相应层级的子合约是否被满足。

测试用例优化

根据测试覆盖率、执行效率等指标对生成的测试用例进行优化，以提高测试效果。