技术报告

(2019-2020)

导师, 漆涛 , 姓名, 张浩天, 2018110895

*Abstract*—Neo4j的使用过程

*Index Terms*—图数据库

# NEO4J的CYPHER查询语言使用

考

虑到现软件遍历结果与报告内容较少等因素，同时为了熟悉Cypher查询语言，在图结构相似的情况下，现以某城市地铁数据为基础构建图数据库。

表 I为156条站台数据中的部分站台信息，其中包含所属线路、站台名、站台类型信息。

表 I

地铁站台数据

|  |  |  |
| --- | --- | --- |
| 线路 | 名字 | 类型 |
| 1 | 韦家碾 | 侧式站台 |
| 1 | 升仙湖 | 岛式站台 |
| 1\7 | 火车北站 | 岛式站台 |

表II为部分线路数据源，其中包含上一站台名，下一站台名以及站台之间的距离等信息。

表 II

地铁线路数据

|  |  |  |  |
| --- | --- | --- | --- |
| 站名 | 下一站名 | 上一站名 | 下一站距离 |
| 韦家碾 | 升仙湖 |  | 1514 |
| 升仙湖 | 火车北站 | 韦家碾 | 1593 |
| 火车北站 | 人民北路 | 升仙湖 | 1180 |

## 创建结点

创建结点有多种方法，如下表示创建包含名字属性的站台类型结点的方法。

### 创建单个结点

CREATE (n1:站台{ name: "犀浦" })

CREATE (n2:站台{ name: "天河路" })

### 创建多个结点

CREATE (n1:站台{ name: "犀浦" }) , (n2:站台{ name: "天河路" })

在创建结点的过程中遇到以下问题。

语句：

CREATE (n1:站台{ name: '犀浦' }) return n1; CREATE (n2:站台{ name: '天河路' }) return n2;

错误：

Neo.ClientError.Statement.SyntaxError: Expected exactly one statement per query but got: 2

## 创建关系

一般情况下，使用如下方法创建关系。

match(n:站台{name:"犀浦"}),(m:站台{name:"天河路"}) create (n)-[r1:开往]->(m)

在创建关系的过程中遇到以下问题。

语句：

match(n:站台{name:"犀浦”}),(m:站台{name:”天河路”}) create (n)-[r:”开往”]->(m) create (n)<-[r:”开往”]-(m)

错误：

Neo.ClientError.Statement.SyntaxError: Unexpected end of input: expected '\', ANY or '"

语句：

match(n:站台{name:"犀浦"}),(m:站台{name:"天河路"}) create (n)-[r:开往]->(m) match(n:站台{name:"天河路"}),(m:站台{name:"百草路"}) create (n)-[r:开往]->(m)

错误：

Neo.ClientError.Statement.SyntaxError: WITH is required between CREATE and MATCH

语句：

match(n:站台{name:"犀浦"}),(m:站台{name:"天河路"}) create (n)-[r:开往]->(m) create (n)<-[r:开往]-(m)

错误：

Neo.ClientError.Statement.SyntaxError: Variable `r` already declared

若需要同时创建多个关系，可以使用如下方法。

match(n1:站台{name:"天河路"}),(n2:站台{name:"百草路"}),(n3:站台{name:"金周路"})

create (n1)-[:开往]->(n2) create (n1)<-[:开往]-(n2),create (n2)-[:开往]->(n3) create (n2)<-[:开往]-(n3)

需要注意match和create语句的位置。多个create语句可以合并为同一个。与上方第二条错误语句类似，不可以使用如下方法进行关系的创建，否则会提示需要加入with语句。（with语句的使用方法见网站[1]）

match(n:站台{name:"天河路"}),(m:站台{name:"百草路"}) create (n)-[r1:开往]->(m) create (n)<-[r2:开往]-(m)

match(n2:站台{name:"百草路"}),(m2:站台{name:"金周路"}) create (n2)-[r3:开往]->(m2) create (n2)<-[r4:开往]-(m2)

此时创建的图结构如图 1所示。

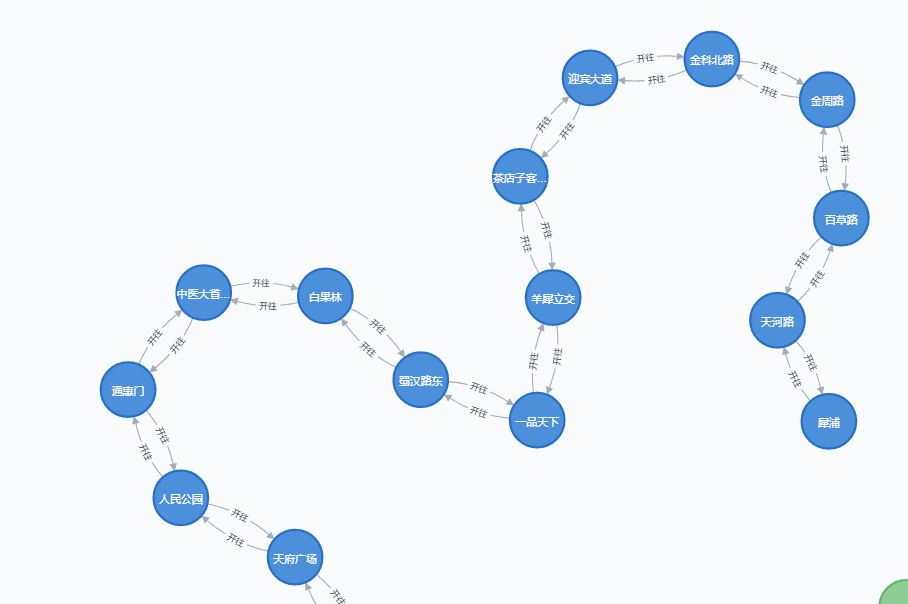


图 1. 初步的结点与关系结构

## 批量导入结点

Neo4j支持批量导入CSV文件，从而实现快速导入大量结点数据。将站台数据以CSV文件格式存储，并删除其中的重复数据。

在导入结点的过程中遇到以下问题。

语句：

LOAD CSV WITH HEADERS FROM 'C:\Users\54251\桌面\2.csv' AS n

CREATE (:站台 { Line: n.Line,Name:n.Name,Type:n.Type,TransLine:n.TransLine})

错误：

Neo.ClientError.Statement.SyntaxError: Invalid input ‘s’: expected four hexadecimal digits specifying a nicode character

语句：

LOAD CSV WITH HEADERS FROM 'C:\\Users\\54251\\桌面\\2.csv' AS n

CREATE (:站台 { Line: n.Line,Name:n.Name,Type:n.Type,TransLine:n.TransLine})

或者

LOAD CSV WITH HEADERS FROM 'C:/Users/54251/桌面/2.csv' AS n

CREATE (:站台 { Line: n.Line,Name:n.Name,Type:n.Type,TransLine:n.TransLine})

错误：

Neo.ClientError.Statement.ExternalResourceFailed: Invalid URL 'C:\Users\54251\桌面\2.csv': unknown protocol: c

语句：

LOAD CSV WITH HEADERS FROM 'file:///C:/Users/54251/桌面/2.csv' AS n

CREATE (:站台 { Line: n.Line,Name:n.Name,Type:n.Type,TransLine:n.TransLine})

错误：

Neo.ClientError.Statement.ExternalResourceFailed: Couldn't load the external resource at: file:/D:/Program%20Files/neo4j-community-3.5.9/import/Users/54251/桌面/2.csv

语句：

LOAD CSV WITH HEADERS FROM 'file:C:/Users/54251/桌面/2.csv' AS n

CREATE (:站台 { Line: n.Line,Name:n.Name,Type:n.Type,TransLine:n.TransLine})

错误：

Neo.DatabaseError.General.UnknownError: URI is not hierarchical

最终将CSV文件存放于Neo4j的import目录下(D:\Program Files\neo4j-community-3.5.9\import)，使用如下方法批量导入结点。注意结点的属性中名字一项需要为小写的name，否则不会在图中显示结点名，也可以选择结点类型后点击Caption中的name来显示名称。

LOAD CSV WITH HEADERS FROM 'file:///2.csv' AS n

CREATE (:站台 { Line: n.Line,name:n.Name,Type:n.Type,TransLine:n.TransLine})

## 批量建立关系

可使用match和create语句通过如下方法对结点建立连接关系。

match (n91:站台{name:"太平园"}),(n153:站台{name:"簇锦"}),(n154:站台{name:"华兴"}),(n155:站台{name:"金花"}),(n156:站台{name:"双流机场1航站楼"}),(n157:站台{name:"双流机场2航站楼"})

create (n91)-[:开往]->(n153) , (n91)<-[:开往]-(n153) , (n153)-[:开往]->(n154) , (n153)<-[:开往]-(n154) , (n154)-[:开往]->(n155) , (n154)<-[:开往]-(n155) , (n155)-[:开往]->(n156) , (n155)<-[:开往]-(n156) , (n156)-[:开往]->(n157) , (n156)<-[:开往]-(n157)

若重复结点过多，或者使用match语句时存在多个匹配结点，则会出现如下错误提示。

Neo.DatabaseError.General.UnknownError: No RelationsipState for added relationship!

或者经过长时间等待后才会提示连接建立成功，如下所示。此时结点与关系如图 2所示。

Created 1280 relationships, completed after 87 ms.



图 2. 错误的结点与关系结构

最终关系建立成功，图结构如图 3所示。

Created 322 relationships, completed after 19 ms

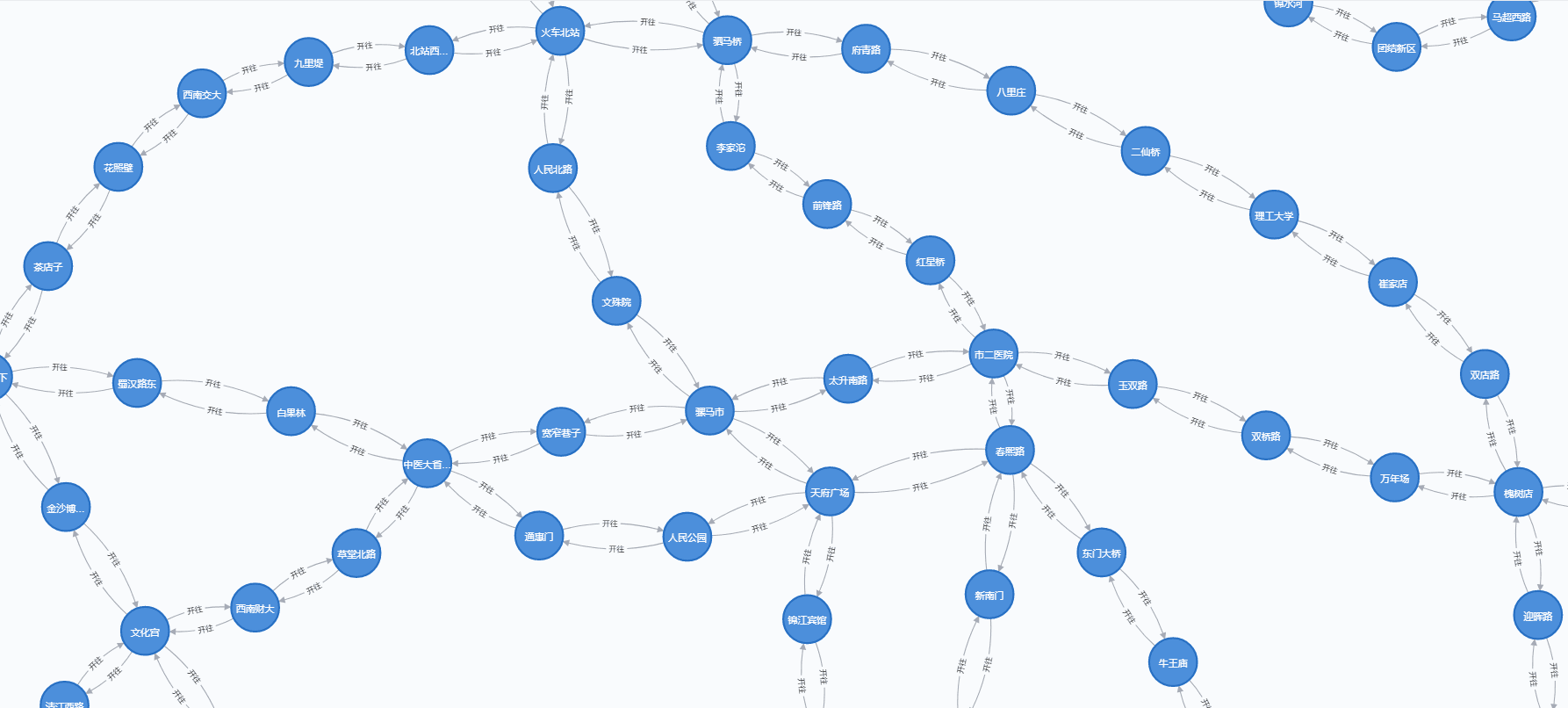


图 3. 最终的结点与关系结构

## 删除语句

对结点进行删除时，需先删除结点间存在的关系。

关系的删除：MATCH (n:站台)-[r:开往]-() DELETE r

结点的删除：MATCH (n:站台) delete n

## 补充

Neo4j Browser浏览器的使用说明可参考https://neo4j.com/developer/neo4j-browser/

# 开题内容考虑

## 毕业设计准备具体实现的内容

1.使用包含结点和关系的图结构，利用Neo4j图数据库对遍历过程进行建模存储。每个结点可表示为一个页面，和测试报告中的每条报告对应。每一个没有出度的结点都代表系统的一个功能。（以首页结点作为最浅层，参考极大值概念，每个极深层的结点都代表系统的一个功能。）（遇到重复页面是建立新的还是连接已有结点。若建立新的结点，则图结构为树，则其为叶子结点。）结点之间的有向连接关系可表示为遍历过程中的跳转关系。

2.结点的属性包含遍历过程中页面控件信息、截图、Log（日志）信息或包含详细内容的文件（避免展示不全）。可输入有关信息，对结点信息进行搜索与展现。

4.针对图结构对测试过程进行分析，包括测试用例、测试步骤、测试记录和测试漏洞。

测试用例包含图结构中至叶子结点经过的所有结点。一个测试用例一般也包含有序的测试子用例，例如登录的测试用例包括输入用户名密码和点击登录按钮等测试用例。

测试步骤包括图结构中至叶子结点经过的每个结点。测试步骤按照顺序构成了测试用例。

（测试记录包括测试时点击控件前后的截图、DOM（文档对象模型）、相关提示（密码错误）等已有信息。

测试漏洞是指记录下的崩溃日志、程序警告内容、界面错乱等现象，需要通过相关算法对一些漏洞进行识别。）

3.根据图结构计算测试用例的数量。针对多个测试入口（如多个到达登录页面的方法），在保证遍历覆盖范围的情况下得到深层结点的最短遍历路径。或者生成覆盖最大遍历范围的遍历路径、最短遍历路径、深度最深的遍历路径等多个路径。对每个页面生成单元测试用例。

4.对不同版本的软件进行测试时，在不同版本生成的图结构不同的情况下，可显示出不同的结点或连接关系，通过diff对比等方法，展示增删的软件功能。支持两次或者多次。

5.将多次遍历后的图结构进行合并，各自图结构的相同部分保持不变；不同部分增加到现有图结构中。在记录人工测试的过程与相关数据情况下，可将人工遍历的结果合并到图结构中，同时可查看测试的覆盖范围。

6. （~~神经网络~~）对已有测试用例进行扩展。根据页面下未遍历控件扩展。如在遍历测试时使用通过微信登录方式，可得到同一页面下包含的通过手机号登录的方式作为新的测试用例。

对下一步的测试遍历路径进行预测。 如在进行遍历测试时，

## 绘制完原型新增的需求们

* 展示结点和关系功能

可以直接点击结点或关系，查看结点简单信息，查看关系简单信息。

* 搜索与展示图结构

搜索可以使用正则匹配等高级搜索办法。

可以选择只显示搜索结果，或者查看（搜索结果）在图（结构）中的位置。

* 分析测试用例

可以点击图结构中的结点，分析测试步骤。

* 分析测试步骤

显示可对结点进行的操作具体为：可以点击查看reqDom文件、DOM文件、点击控件前后的截图、日志文件。

去掉生成单元测试用例功能。

* 分析测试记录和缺陷

每个suite对应的activity页面都可以点击生成单元测试用例。

* 计算测试用例数量

可以（以图结构方式）展示所有的用例。

可以点击每个用例从而分析测试用例，可以点击每个用例中的结点从而分析测试步骤。

* 生成遍历路径

可以保存作为测试用例

* 合并图结构

可以选择查看原本图结构、第二张图结构、合并后的图结构。如有人工部分，可以查看原本图结构、人工遍历图结构、合并后的图结构。

合并后新增加的结点和关系用其他颜色标识。如有人工部分，原本结点、人工遍历图结构结点、合并后新增结点、合并前关系、合并后关系需要以不同颜色标识。

可以将合并后的图结构保存到原本图结构中。

* 获取未遍历控件功能

生成测试用例后可以保存到图结构中。

* 获取下一步遍历路径功能

获取成功后显示下一步要点击的不同控件的概率，和路径与结点的相关信息。

点击预测出的控件可以分析测试步骤。

## 需要改进的地方

没有表现出测试报告读取解析功能的过程。

图结构的展示和操作方法。

对比的效果不明显。

分析测试步骤中，生成遍历路径为包含该结点的遍历路径，不是该结点为端点的遍历路径。

计算测试用例数量中测试用例的展示方法。

分析测试用例中路径与结点的相关信息的展现方法。

单元测试用例的展现形式。

未遍历控件的展现形式。

获取预测的方法。

## 需要改进的地方2

√说明图数据库的必要性，定义。与关系型数据库的对比。

体现系统的用户、其他系统，表明与本系统的输入输出关系，如显示输入项是从AppCrawler中得到。

写出所用到的算法，可行性分析和预期的解决方案、主要解决的技术问题。

## 关于对原型的反馈

测试用例数量计算页面，可以显示页面名称和控件名称作为整体显示并且可以点击。每一个用例右侧有可以查看详细内容的按钮。

分析测试用例的信息有，点击前后的元素，（输入、点击、（滑动））动作、截图。

版本的对比很关键，合并和对比的功能类似，都可以用不同的颜色进行标注，对比部分也使用颜色标注。需要读取elements.yml文件。对于人工测试的范围，是取合集进行计算。未遍历的可以在图结构中以不同颜色显示。

遍历的测试用例和未遍历的测试用例、找到的元素还需进行测试，推导测试用例。可以根据未遍历控件生成新的，或者结果中标记未Cancel中的结果。

生成遍历路径可以选择多个结点。可以考虑生成将单元测试用例和生成测试用例的功能放在同一个页面。可以选择经过的结点或者所在的Activity来进行生成，或者都选择来进行生成。

测试用例构成了suite，目前不考虑suite的问题。

获取element:url: "com.xueqiu.android.LoginOptionActivity"

## 部分参考文献

* **移动应用程序测试**

Android应用程序GUI自动化遍历技术研究\_赵耀宗：

赵耀宗针对现有测试过程中脚本自动化程度低、需要静态分析构建图等问题，提出了通过解析配置文件获取组件信息，从而生成用户动作来模拟交互的过程，实现了对移动应用程序进行图形用户界面自动遍历的方法。

（提到了monkey、monkeyRunner、Robotium、Uiautomator）

移动终端应用程序自动化覆盖测试方法研究\_张硕：

张硕首先通过研究有向图，构建了页面逻辑跳转关系的模型，接着根据页面和控件与图结构的映射关系建立了有向图，并针对页面覆盖提出了最小树形图的遍历方法；针对控件覆盖提出了状态图路径生成方法，实现了页面和控件的高效覆盖。

* **基于UML的软件测试**

一种基于UML 活动图的测试用例生成方法：

李庆首先对活动图进行了形式化定义，并给出了覆盖准则，接着基于UML活动图对并发模块进行了处理和实例化，从而解决路径过多的问题，引入约束条件后，得到有效的测试场景和数据，最终生成了可行的测试用例。

基于UML活动图模型的测试用例自动生成方法的研究及实现

孙瑶为实现活动图中并发模块的有序生成，提出了双层遍历方式，又在保证有效覆盖的情况下限定了循环次数，缩短了遍历时间，解决了测试用例繁杂的问题，最后介绍了场景优化的方法避免了测试用例较多的情形。

* **图数据库的使用：**

基于Neo4j的煤矿领域知识图谱构建及查询方法研究\_叶帅

叶帅**则**基于已有学科知识对其领域构建了知识图谱，并利用图数据库Neo4j提出并改进了适用于该领域的知识图谱存储方法，利用并行朴素贝叶斯知识查询方法，设计实现了领域的知识图谱，对该领域的智能化进行了有效探索。

* **智能测试/国外测试平台的了解：**

mabl、testai、katalon studio、applitools。

https://applitools.com/features

美国的Applitools是一款使用人工智能算法进行功能测试和用户界面测试的可视化管理软件。在捕获应用程序界面后，由人工智能驱动的视觉网络对比算法可以快速找出与基准图像有差异的界面元素。但Applitools无法高效比较不同分辨率和屏幕尺寸下相同的界面元素。

A Novel Tool for Automatic GUI Layout Testing ieee

Kristian则会收集页面的布局信息，对重叠、溢出和未对齐等布局错误进行识别和基于规则与统计的分析，同时可以检查不同屏幕尺寸和分辨率下的应用程序是否会发生布局错误。与Applitools相比，其自动化程度更高且所需的人工干预更少。

人工智能在软件测试上的应用与挑战\_任子恒

任子恒指出，人工智能可运用到软件测试中，通过大量的数据训练，使其能够根据特定需求生成测试用例，执行测试并对结果进行分析。

AI在软件测试领域中的应用与发展\_颜媚

颜媚提到，大部分的测试都是重复已有的测试内容，而人工智能可以高效完成重复性步骤，节省测试人员时间。

（提到了King 采用AI 测试工具测试Candy Crush Saga、APPdiff推出测试机器人、Bugdojo创建基于AI的测试平台、DiffBlue发布三款AI软件测试产品、微软发布“AI安全风险检测”工具、Facebook发布名为Sapienz的工具、阿里Ripper、中国信息通信研究院泰尔终端实验室的智测云测试平台引入AI技术）

https://www.katalon.com/testops/

美国的Katalon TestOps作为自动化测试生成的解决方案，可通过基于视觉的测试，能在更改测试用例的情况下保证测试脚本稳定运行。使用机器学习来选择最重要的测试用例，从而尽早发现可能的缺陷。利用聚类分析，快速发现测试失败的原因，最终节省调查所需的时间与精力。

测试用例自动生成算法设计及自动化测试平台构建\_梁卓杰

提出了UML活动图和遗传算法相结合的测试用例生成方法

基于UML与智能优化算法的测试用例生成方法研究\_张旭良

利用UML活动图模型和改进的蚁群算法来生成测试用例

* ~~图论的研究：~~
* ~~人工智能算法的介绍：~~

## 题目的选取

学长的：

移动应用的UI自动化测试系统的设计与实现

移动应用性能自动化测试系统的设计与实现

移动应用Neo4j图数据库可视化测试系统的设计与实现

基于Neo4j图数据库的移动应用程序遍历过程可视化分析

基于Neo4j图数据库的移动应用程序可视化遍历过程的设计与实现

基于Neo4j图数据库的移动应用可视化遍历过程的设计与实现

基于Neo4j图数据库的自动化测试过程可视化的设计与实现

答辩前：基于Neo4j图数据库可视化的移动应用自动化测试系统的设计与实现

修改后：（基于图数据库的）

移动应用自动化测试系统的可视化研究与实现

移动应用可视化测试系统的研究与实现

最终：面向移动应用的测试可视化系统的研究与实现

zjf的：基于Jenkins集成服务的自动化测试平台的设计与实现

## 片段

路径可视化，参考网站[3]

可以从下面几个方面来阐述为什么自己是解决该科学问题的合适人选：

你提出的解决方案是可行的；

自己的过往研究是适合这个项目；

自己已经对问题有了深入了解，知道潜在的障碍和挑战；

自己的知识结构和技能足以处理潜在的问题。

## 开题报告

## 开题答辩大纲

随着移动应用数量的不断增多与移动应用开发技术的不断发展，诞生了种类齐全的移动应用程序。由于开发者水平、操作系统特性等的问题，应用程序的可靠性不能完全保证，软件质量的重要性也在逐渐显露，开发人员对移动应用测试的重视程度也在进一步加深。

在传统PC端软件测试中，人工测试一般占有很大比重。由于移动应用主要通过用户对图形界面进行点击、滑动等进行交互，不能完全使用已有的测试方法。因此，可以使用自动化测试的方法，一定程度上替代传统的人工测试方法。同时，使用建立模型的方法对系统和测试过程进行抽象表示，从而使测试过程更加清晰、便于理解，提高测试效率。

目前，针对移动应用的自动化测试方法，主要可以通过获取移动应用页面组件信息，并根据相应控件生成对应的用户动作，来模拟用户行为，完成用户图形界面的可视化自动遍历。还可以构建有向图，映射应用程序和图之间的关系，进行动态遍历路径的探索，做到页面和控件的高效覆盖。

同时，面向对象的概念也运用到了软件测试技术中。根据活动图的并发特性进行优化改进，引入约束条件减少无效的测试场景和循环场景，可以为测试用例的自动生成提供了可行性的方案，从而可以减少测试用例的编写时间。

另一方面，随着大量测试数据的积累，使用传统关系型数据库进行关系查询会导致大量开销，因此采用了更容易表示和对移动应用页面进行建模的图数据库，他是以图结构进行语义查询，以结点、边和属性来表示存储数据的数据库。使用图数据库可以有效提高某一领域内信息的展示效率和数据的处理能力。

其次，人工智能技术逐渐成熟，多个国内外测试平台实现了一定程度上的智能化测试，提高了测试效率。

本系统旨在研究一套基于图数据库可视化的自动化测试系统。主要包括以下五点目标，系统能够解析已有测试报告，从而生成可视化图结构，并对图结构进行操作管理；同时可以对测试用例、测试步骤、测试记录和缺陷等测试过程进行分析和处理；并能够生成和计算满足不同特征的遍历路径，来表示测试用例，和各个移动应用页面单元测试用例；针对相同或不同版本移动应用的测试结果，能够对比合并包括人工测试在内的图结构；而且能获取测试过程中页面内还未遍历的控件，并可以预测下一步的遍历路径。

研究内容主要分为以下几个方面。实现能够将测试报告进行读取解析的功能，生成可视化图结构，并且在图结构中可以显示并查看结点与关系的信息，通过输入关键字对图结构进行搜索，展示搜索结果在图结构中位置。

可以查看并分析测试过程中的测试用例、测试步骤、测试记录与缺陷，查看其中展示出的结点和关系，显示每个页面测试过程中出现的提示信息、测试耗时和测试成功的数量，在存在缺陷的情况下，还能够显示其中存在的崩溃日志、程序警告内容和界面错乱情况等信息。

可以根据图结构计算测试用例的数量并展示所有测试用例，并对这些测试用例进行管理。够使生成覆盖范围大、图结构中路径最短或深度最深的测试用例，同时也够生成每个页面的单元测试用例，并将新生成的测试用例进行保存。

可以将不同版本的移动应用测试结果的图结构进行对比；或将相同版本的移动应用测试结果的图结构进行合并，同时也可实现将人工测试过程的图结构与已有图结构进行对比和合并，计算人工测试过程的覆盖范围等信息。

最终可以获取测试过程中移动应用页面内还未遍历的控件，并可以预测下一步遍历的路径。

移动应用页面逻辑跳转关系的图结构的表示方法与操作管理方案。本系统将移动应用中的页面和其中的数据转化为图结构中的结点和相关信息；页面的跳转关系转化为图结构中的有向边。从移动应用相关信息到图结构的转换过程中，需要充分考虑移动应用不同页面包含的控件与布局信息，并且需要准确展现出移动应用内部复杂的页面跳转关系。由于使用了Neo4j图数据库，因此需要在已有可视化的图数据库基础上实现结点和关系的便捷的操作管理功能，分析其中的测试步骤，了解遍历路径，提升测试人员效率。

使用智能算法，在管理已有测试数据和用例的基础上，对已点击过的控件元素进行分析，获取到移动应用同一页面中类似的控件与元素，供测试人员参考，并针对新获取的控件生成新的测试用例；系统还可在已有测试用例覆盖范围不足的条件下，生成覆盖不同页面的测试用例，作为新的可能的遍历路径，完善已有测试方案。实现智能化的测试流程。

## 开题答辩笔记

项目来源为企业合作项目。

修改意见：题目、拟研究的科学问题、时间安排、预期成果等。

具体包括学硕、专硕论文的内容要求、新型的要求、工作量的度量、阶段答辩的要求等。

学硕、专硕论文内容：学硕需要做理论和算法研究，需和已有算法数量上时间性能效率准确度对比，注意对比条件。已有算法较多的话需要实现一些已有算法，例如推荐系统原有算法要做四五种，第一种是在算法上的改进。第二种是已有技术运用到新的场景中，调整原有算法，也需要和新场景下新旧方法区别、已有方法在新场景和类似场景的区别两类来源对比。否则阶段报告通不过。（对算法的改进、对新场景的应用）。

尽量不做大而全的，要针对具体深入，做表层的工作和没有涉及困难的没有任何好处。研究算法时需要结合具体的场景。研究型课题最糟糕的是只做了一点，尽可能做深入。失败作为成果时需要展示出工作量，解释根本原理原因。

研究内容不包括对现有知识的学习、文献阅读。

创新性，需要有大量参考文献，要有三十篇论文，知网的不算，一半都在sci索引库中找到。sci+ei引文。一般整个博士生涯要读三百篇，因此研究生要读一百篇，精读三十篇。根据调研的背景信息进行改进。科学问题和创新点，完成系统后需要与之对比。一般写一个就够了，保持1-2个。

阶段答辩时要清楚为什么答辩，告知已经做完而且效果不错，能做而且已经做完了。

专硕纯做工程的，工作量看代码量，java超过五千行核心代码，不算框架和get等。

发现 研究和开发陷入瓶颈时，找老师联系。

工作计划与时间点：需求11-12月份、做完系统6月底之前、7-8月份可能有实习，学硕小论文或专利7月来不及了，解决关键问题的时候争取在5-6月份给出初稿，8月底之前论文要投出去，专利要准备。都做完了阶段报告10月，大论文初稿12月初。

论文要求：小论文投完顶级，拿不到录用通知后才考虑使用科技论文在线。

2020.1.2补充：毕业前实验室要求每位同学都要提交成果，工学硕士要求提交SCI论文，工程硕士要求提交专利，根据自身能力也可提交SCI论文，从现在开始所有同学都要开始准备了，成果提交之后，到学校要求的答辩时间，方可申请答辩。

预期成果：具体模型，指标功能和验证、完成了系统，功能，通过测试部署，具体的任务。

# 软件需求书摘录

## 第一章

**软件需求包括三个不同的层次**—业务需求、用户需求和功能需求—也包括非功能需求。业务需求（business）反映了组织机构或客户对系统、产品高层次的目标要求，它们在项目视图与范围文档中予以说明。用户需求(user) 文档描述了用户使用产品必须要完成的任务，这在使用实例（use case）文档或方案脚本（scenario）说明中予以说明。功能需求(functional)定义了开发人员必须实现的软件功能，使得用户能完成他们的任务，从而满足了业务需求。

**不适当的需求过程所引起的一些风险**

• 模棱两可的需求说明可能导致时间的浪费和返工。

• 用户增加一些不必要的特性和开发人员画蛇添足

• 过分简略的需求说明以致遗漏某些关键需求。

• 忽略某类用户的需求将导致众多客户的不满。

• 不完善的需求说明使得项目计划和跟踪无法准确进行

# 需求分析文档

完成需求分析文档的内容，包括用例图和各个用例的活动图，如图 4所示，详情间需求分析文档。

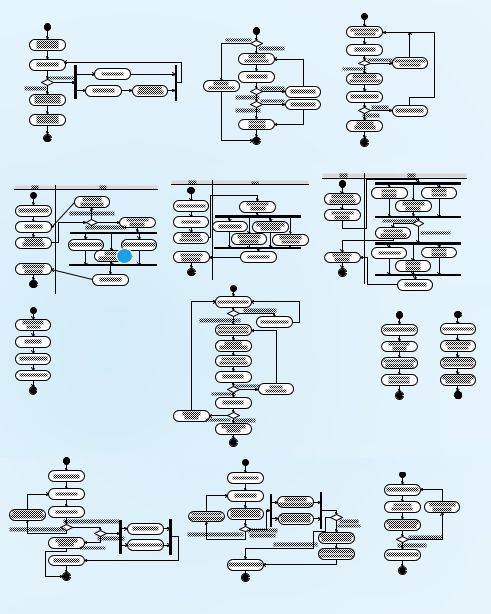
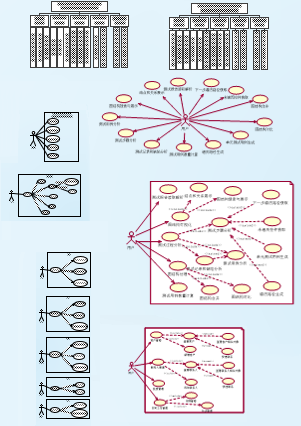


图 4 用例图和活动图

包括可视化管理功能、测试过程分析功能、测试用例管理功能、图结构管理功能、待遍历管理功能。

可视化管理功能包括读取与解析测试报告功能、展示（图结构的）结点和关系功能、搜索与展示图结构功能。

测试分析功能包括分析（展示）测试用例功能、分析（展示）测试步骤功能、分析（展示）测试记录与缺陷功能。

测试用例管理包括计算测试用例数量功能、生成（不同）遍历路径（的测试用例）功能、生成单元测试用例功能。

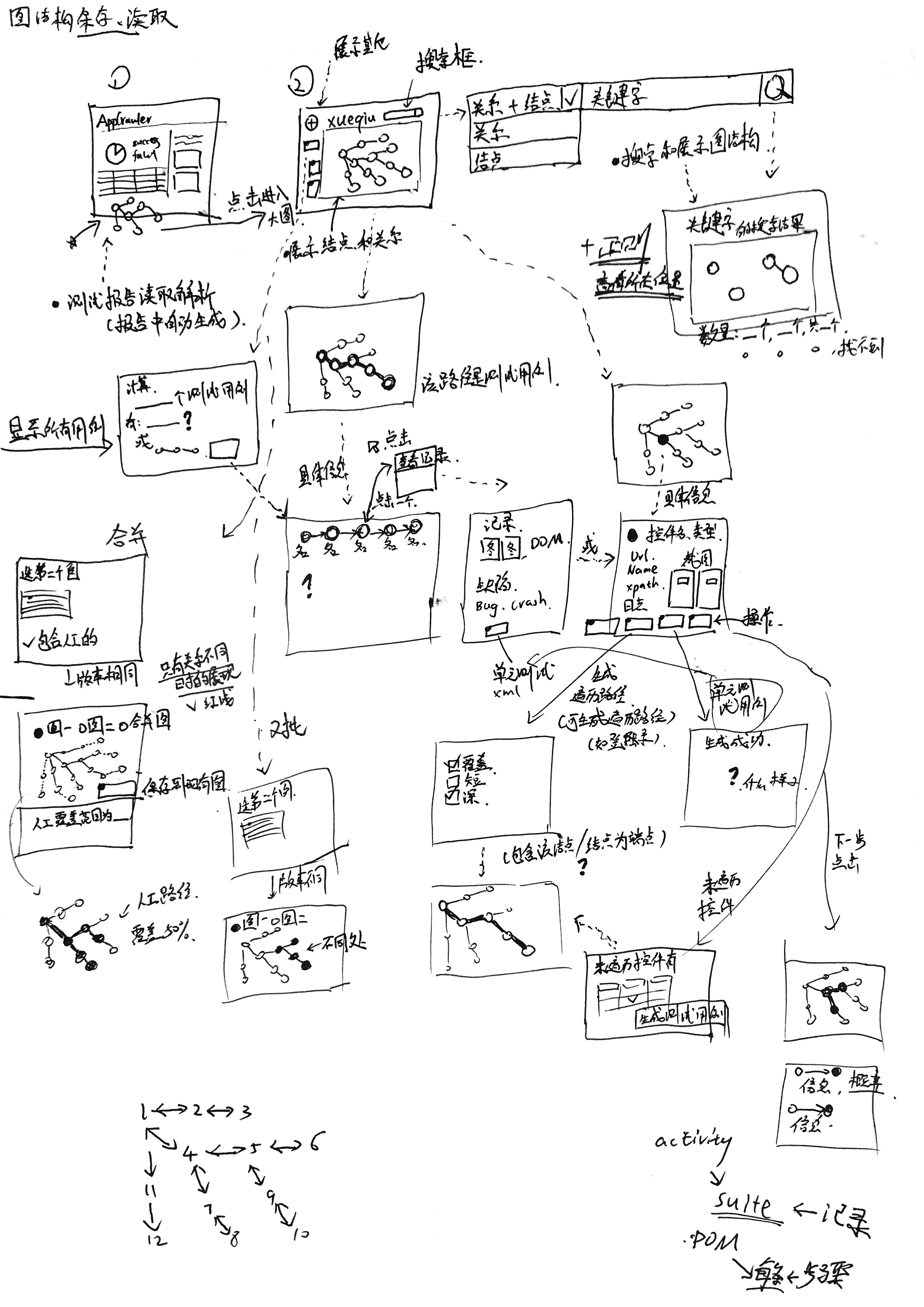
图结构管理包括对比（不同移动应用版本的）图结构功能、合并（相同移动应用版本的图结构和合并人工测试的）图结构功能。

待遍历管理包括获取（移动应用页面内）未遍历控件功能、获取（测试过程中）下一步遍历路径功能。

# 原型文档

为了更清楚系统所需要完成的功能，绘制纸质草稿后，使用Axure制作系统原型。

## 原型规格说明书



## Axure可实现的功能

为解决原型在预览时不显示图片的问题，更新软件版本为Axre RP 9。除了系统之外实现的几个功能和实现方法如下。

点击按钮显示或隐藏元件：选择某一元件，设置交互为鼠标点击时，项目选择显示或隐藏某元件。

全局变量的设置和读取：选择某一元件，设置交互为鼠标点击时，项目选择设置变量值，在此添加一个新的全局变量。接着变量值选择新建的全局变量，可设为1。读取全局变量时，可在元件的交互部分选择载入时，打开详细页面，鼠标右键选择新增用例，双击IF设置用例为变量值=值1，接着再添加一个新的用例，双击IF设置用例为变量值≠1，在两条用例下面添加具体的交互内容，如图 5所示。

图 5. 全局变量交互内容 图 6. 动态面板交互内容

单选框的实现：单选框的交互中，在自身的选中状态下可设置其他所有单选框的选择/选中为“假”。也可以设置自身为选中时的状态。

斜线的绘制：在新版Axure中，可以选择横线或垂直线，拖动一个端点即可以绘制斜线。也可以在样式中指定旋转的角度。

动态面板：可实现轮播图，为某个元件创建动态面板，并设置两个状态。在组织交互中选择载入时切换面板状态，如上方 图 6所示。其他用法可参考网站[2]。

点击后变色：实现复杂的元件变化，可以放置新的控件组成变化前和变化后的元件，在发生交互后设置需要变化的前后元件的隐藏和显示，或者设置变化后的元件置于顶层。

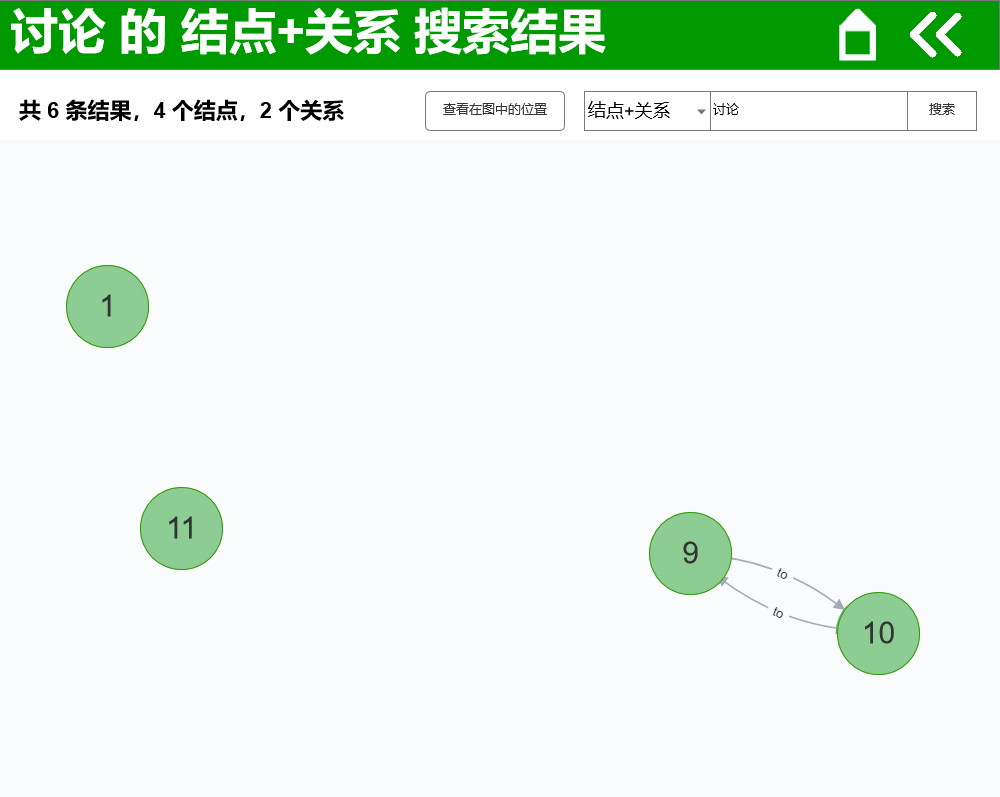
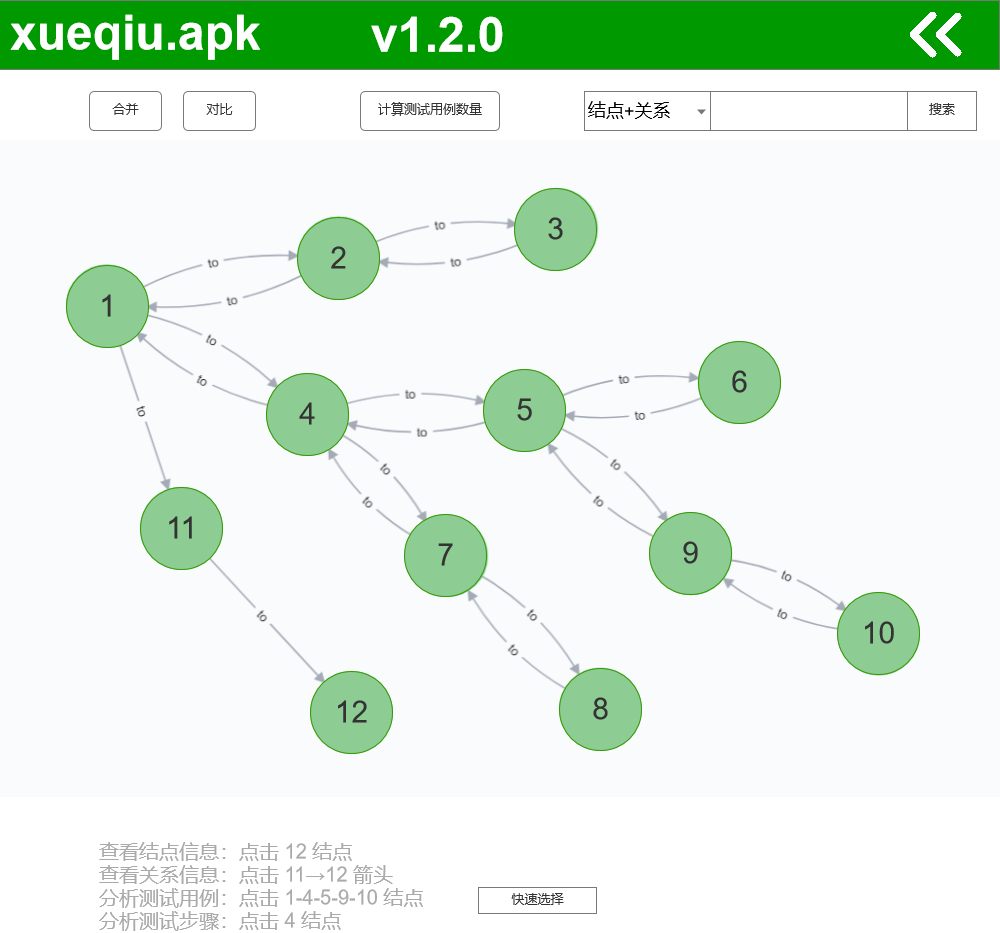
## 原型规格说明书

原型的页面树为：

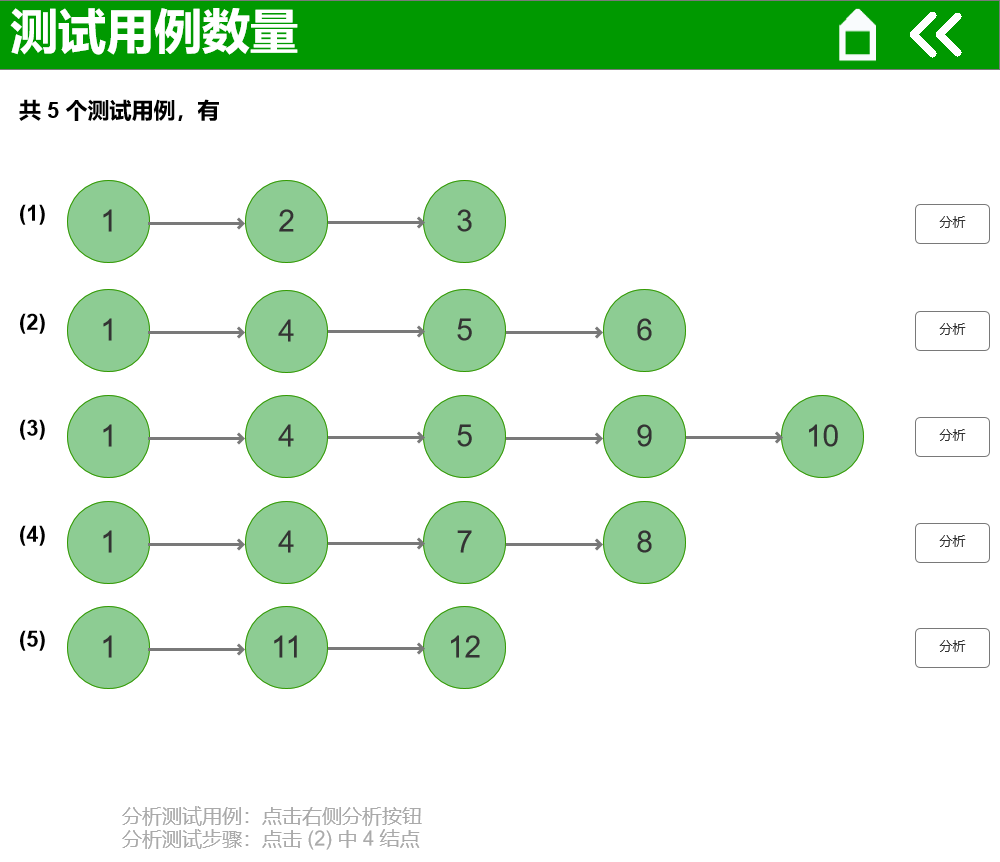
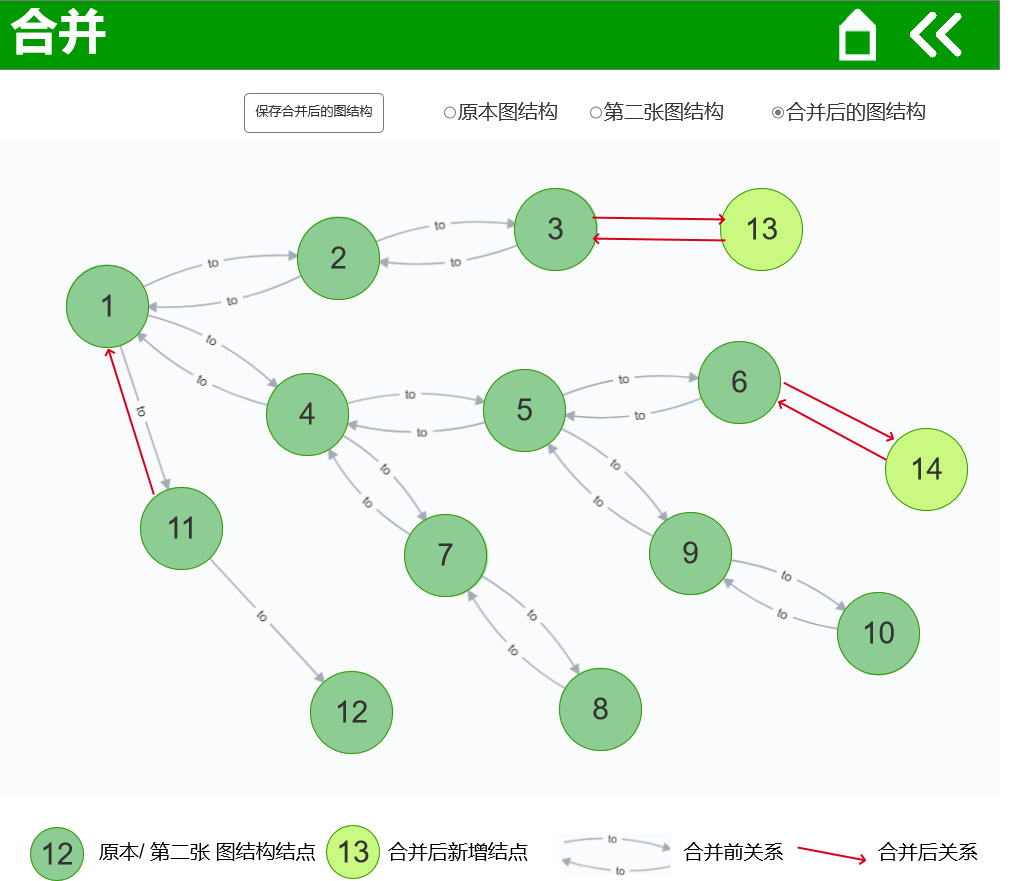
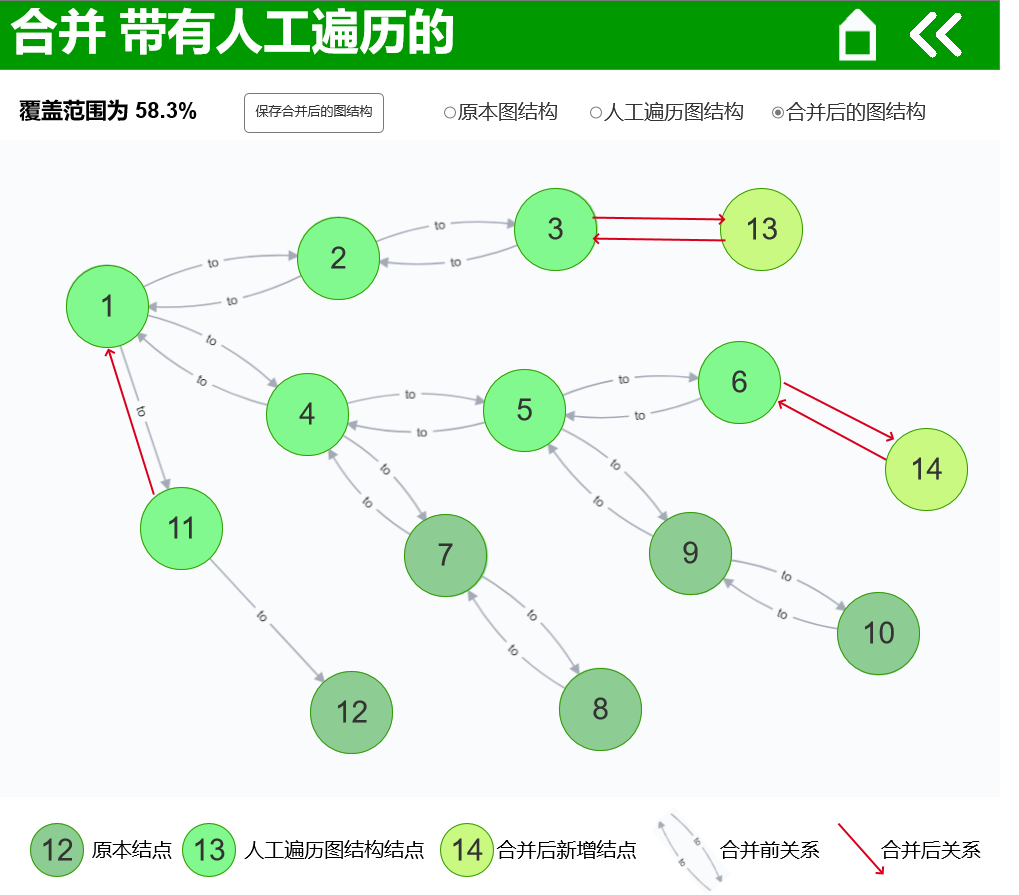
主页、报告{测试报告内容}、图结构主页{搜索结果}、{测试用例数量计算}、{合并，选第二张图}、{合并}、{合并，带人工}、{对比，选第二张图}、{对比}、{分析测试步骤[生成遍历路径、获取未遍历控件(生成测试用例)、[获取下一步遍历路径]}、{分析测试用例}、{分析测试记录和缺陷[生成单元测试用例]}

以下为各个页面的用户界面：

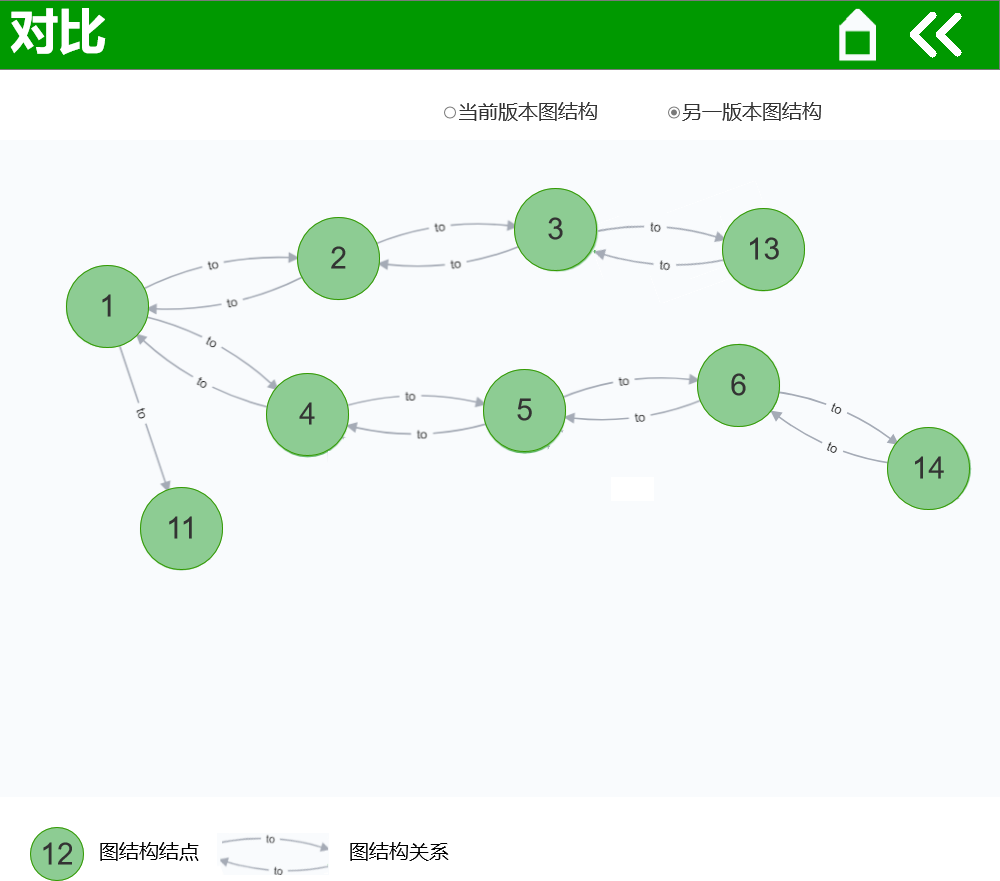
报告、图结构主页、搜索结果



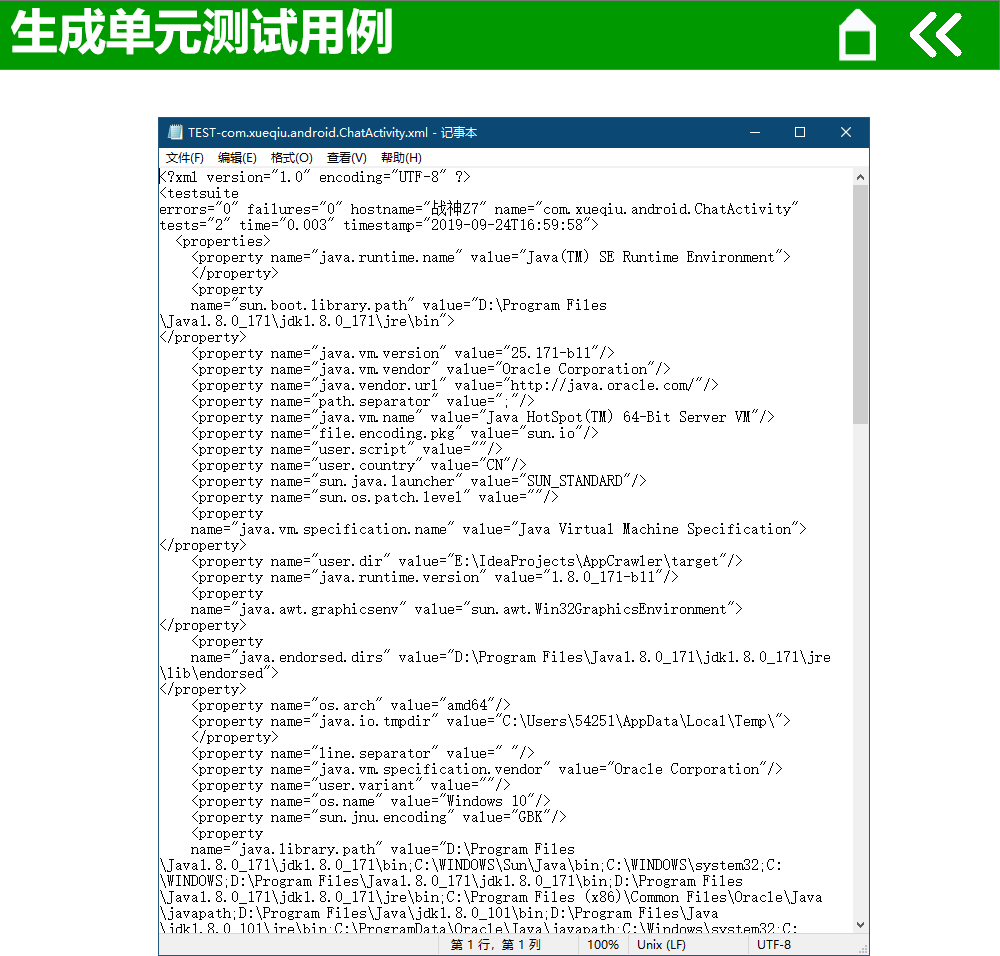
测试用例数量计算、合并、合并，带人工

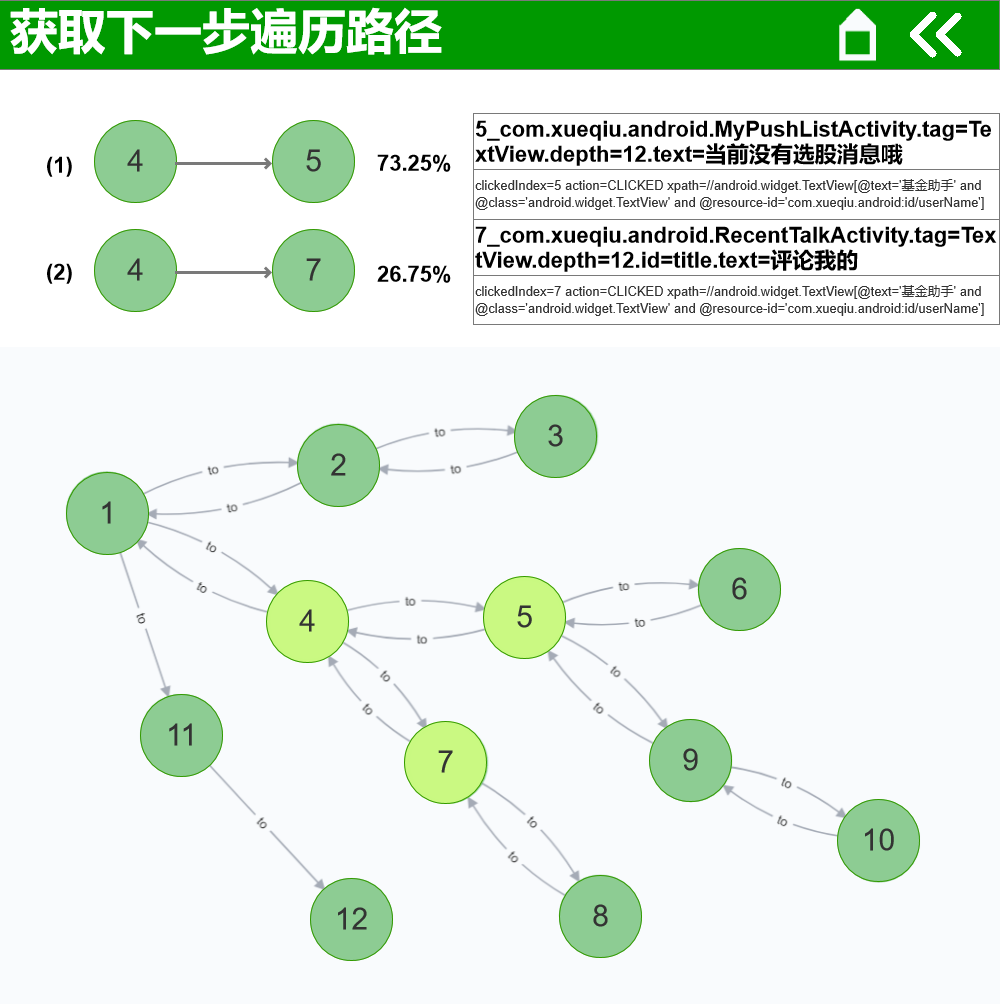
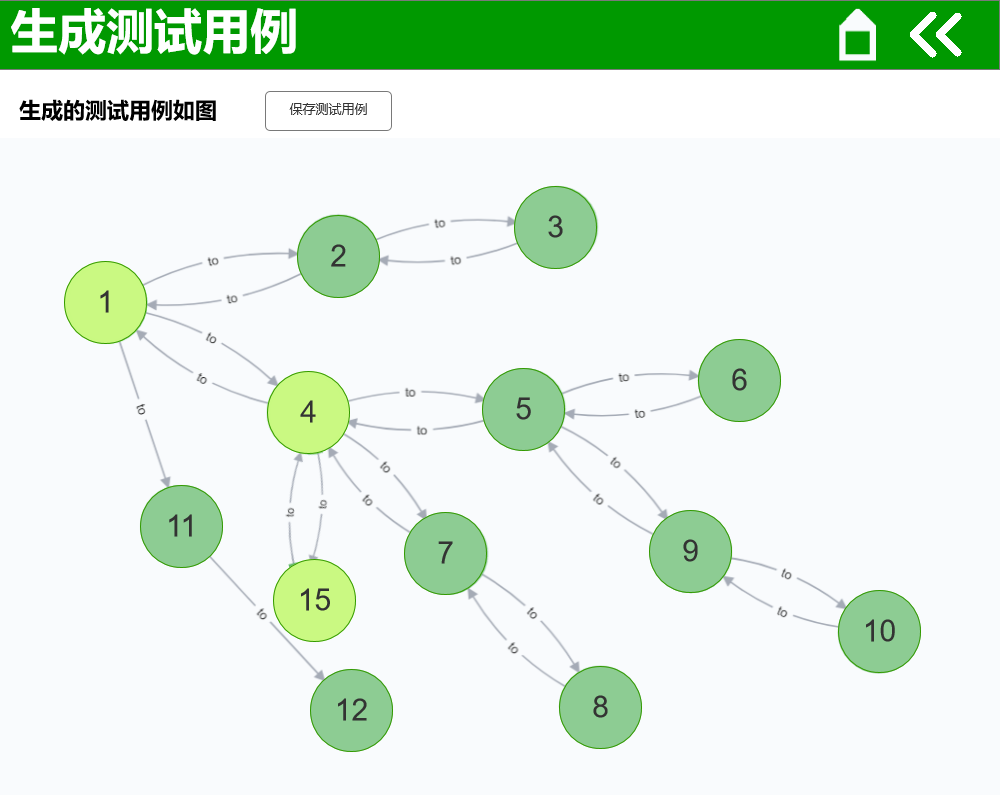
分析测试记录和缺陷、合并/ 对比（选第二张图）（中上）、对比（中下）、分析测试步骤



生成遍历路径、获取未遍历控件、生成单元测试用例

生成测试用例、获取下一步遍历路径、分析测试用例



# 测吧高新文档

测吧为了登记软件著作权信息，特制作代码说明文档。负责对log和report部分进行说明，称为测吧日志和报告系统。

结合之前针对AppCrawler的总结。已完成对AppCrawler类的getGlobalEncoding、setGlobalEncoding、

parseParams、parsePath和addLogFile方法，和对Crawler类的loadPlugins、loadConf、start、crawl、restart、firstRefresh、crawlWithRetry、runSteps、setupAppium和getUri、needBackToApp、needExitApp、needBackToPage、isSmall、getAvailableElement、sortByAttribute、refreshPage、parsePageContext、beforeElementAction、afterElementAction、getPredictBackNodes、getBackButton、crawl、doElementAcion、saveLog、saveScreen、back、getElementByTriggerActions、handleCtrlC方法中关于log部分的简单描述说明。

接着完成了对report包中AllureDiffTestet类、AllureTest类、MvnReplace类、ReadYaml类，和对diff包中的CrawlerDiff类、diffSuite、getURLs、getSuites、drawSquare、removeDuplicate、List方法，DataObject 类、formatElementInfo、formatElement方法，Report类，XPathUtil类、checkDom、getListFromDom、getAttributesFromNode、getParent、getAncestorFromAttributes、getXPathFromAttributes方法的简单描述说明。

最后描述了当前测试报告页面的样式，包括测试报告的页面结构等，如图 7和图 8。并简单叙述了报告文件及文件夹内包含的内容。

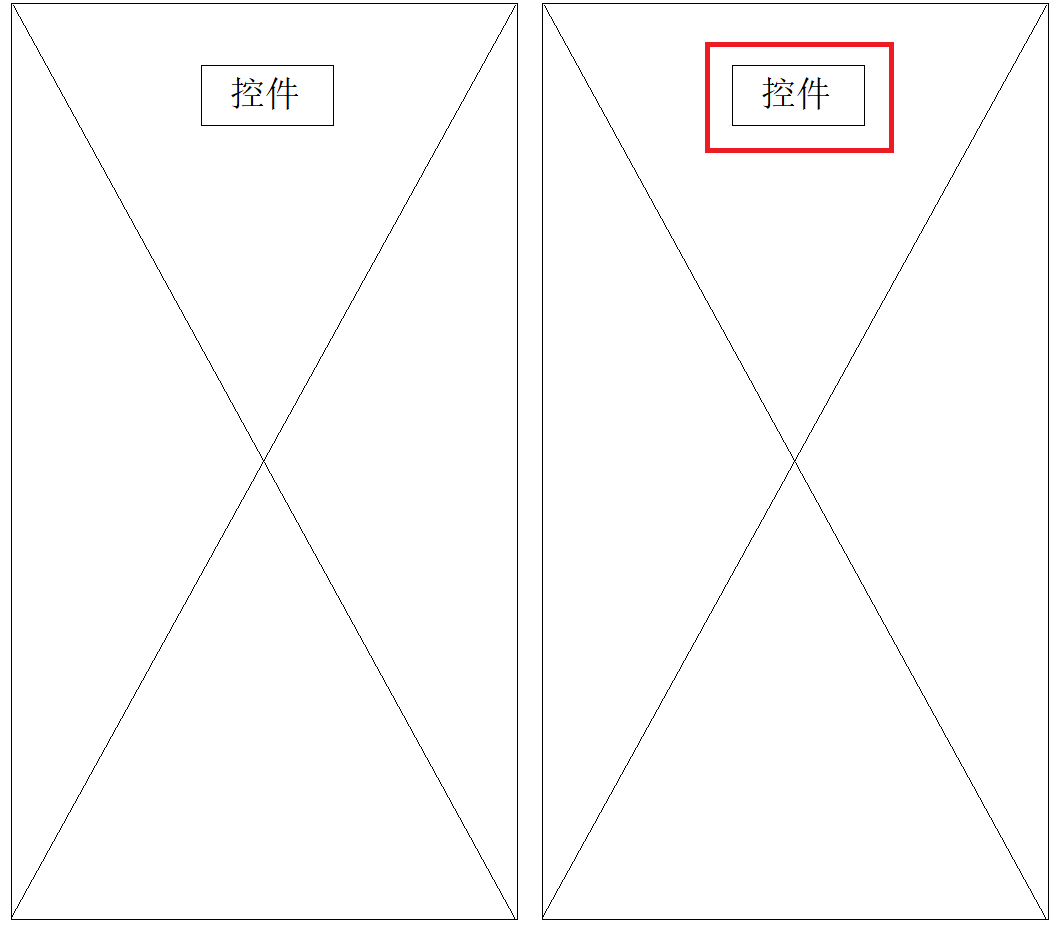


图 7. 测试报告的截图对比



图 8. 测试报告结构

# 图数据库

## 概念

图，是结点和边的集合。在一张图中，一个结点代表一个实体，例如某个人“A”。边，就是关联这些结点的关系 ，例如“A”是“B”的同学。

结点拥有一个（或多个）标签（label），例如植物、动物等；结点可以拥有很多属性（property），例如姓名、年龄等等，以键值对（key-value pair）的形式表示。

边也有标签，标签可以是“住在”等；边也有属性，例如某人“住在”某地的关系上存在“原因”的属性。

子图是图中的特定结点、关系、标签、属性。路径是一组结点和他们关系的集合。

连通图、非连通图。未加权图、加权图。有向图、无向图。非循环图、循环图（有向无循环图DAG，通常有叶子结点，在调度、版本控制问题中很常见，相对于树没有只有一个根的限制）。

## 图数据库的必要性

在包含1m个人，每人约有50个朋友的社交网络中找到最大深度为5的朋友的朋友，表 I为性能对比。可见，数据量越大，查询需要涉及的关系越复杂，图数据库的性能优势越大。

表 I

性能比较

|  |  |  |  |
| --- | --- | --- | --- |
| 深度 | RDBMS时间 | Neo4j时间 | 返回结果 |
| 2 | 0.016 | 0.01 | ~2500 |
| 3 | 30.267 | 0.168 | ~11,000 |
| 4 | 1,543.505 | 1.359 | ~600,000 |
| 5 | ~ | 2.132 | ~800,000 |

对于关系型数据库来说，朋友的朋友意味着两张表的 join， 易产生笛卡尔积的中间数据，数据量随着深度呈幂增长。同理，“A的朋友是谁”、“谁的朋友是A”和“某个顾客买了哪些商品”、“哪些顾客买了某个商品”等不同的查询方式开销一般也不同。

图数据库的好处：更容易表示与建模现实世界的事物、可高效插入大量数据、可高效进行关联查询、专业分析(PageRank、ShortestPath)算法和工具(Neo4j Desktop、**Neo4j Bloom**)。

图数据库的应用：社交网络、实施推荐、知识图谱、反欺诈风控。

## 图计算引擎/架构概述

图数据库的发展历程参考网站[4]。

## 图数据系统

图数据系统应包括：图存储和查询、图处理和计算、数据导入导出、可视化、容灾备份等。

**图存储&数据模型**：主要有属性图、资源描述框架（RDF）、超图。属性图是带有标签属性的图，其是主流的数据模型；资源描述框架中，增加的信息已单独结点表示，结点表示实体/资源或小户型，边表示实体和实体的关系以及实体和属性的关系，其较多运用在知识图谱中。

为“人”结点A添加“name”属性，在属性图中，在A中的name属性添加名字，在RDF中，通过“hasName”的边指向名字。

**图查询&图计算**：图查询指增删改查；OLTP遍历方式为实时返回，涉及少量数据，随机的数据访问，串行运行，用于查询，偏向深度优先的计算引擎，不需要太大的内存；OLAP遍历方式为长时间运行，涉及几乎整个图，串行地访问数据，并行运行，批量处理，偏向广度优先的计算引擎，需要更大的内存。

图计算引擎技术，偏重于全局查询，一般都对与批处理大规模数据做过优化。图计算引擎，或者说图的并行计算框架，包括 Google 的 Pregel，基于 Spark 的 GraphX，Apache 下的 Giraph / HAMA 以及 GraphLab，其中Giraph 是 Pregel 的开源实现。

另一款图数据库产品：JanusGraph。

本章内容至此参考自网站[5]。

**图算法&图分析**：图分析可以查询图数据，使用基本统计信息，可视化地探索图、展示图，或者将图信息预处理后合并到。机器学习任务中。图的查询通常用于局部数据分析，而图计算通常涉及整张图和迭代分析。

图算法主要包括：路径搜索、中心性计算、社群发现。

图搜索算法包括深度优先搜索和广度优先搜索。路径搜索算法建立在图搜索算法基础上，探索结点之间路径，用于物流规划、最低成本乎交、IP路由、游戏模拟等。

**路径搜索**算法有最短路径（A到C）、所有结点对最短路径、单源（A）最短路径、最小生成树。如图 9所示。最短路径应用：地图导航、社交共同好友。所有结点对最短路径应用：优化城市设施位置、货物分配、快递线路设计、作为数据中心设计算法的一部分，查找具有最大带宽和最小延迟的网络。最小生成树应用：旅行计划、传染病追踪。还有随机游走算法，可作为 node2vec 和 graph2vec 算法的一部分，用于结点向量的生成，作为后续深度学习模型的输入；作为 Walktrap 和 Infomap 算法的一部分，用于社群发现，如随机游走总是返回同一组结点，表明这些结点可能在同一个社群；其他机器学习模型的一部分，用于随机产生相关联的结点数据。



图 9. 路径搜索算法

**图评估类算法：**包括**结点**评估类：**中心性算法**，识别图中特定结点的角色及其对网络的影响。识别最重要的结点，帮助我们了解组动态，例如可信度、可访问性、事物传播的速度以及组与组之间的连接。

包括Degree Centrality 度中心性，计算一个结点直接相连的边的数量，包括出度和入度。图中还存在平均度，是边的数量除以结点的数量。平均度易被具有极大度的结点影响skewed。因此，一般使用度的分布表征网络特征。应用：区分在线拍卖的合法用户和欺诈者，欺诈者经常出更高价，拥有更高的加权中心性。

Closeness Centrality紧密性中心性，是检测能够通过子图有效传播信息的结点的方法。对于一个结点来说，紧密性中心性是结点到所有其他结点的最小距离和的倒数。紧密性中心性，其中是要计算的结点，是总结点数，是结点之间的最短路径距离。非连通图时，使用Harmonic Centrality调和中心性，。包含多个子图并且子图间不相连接的非连通图的一种紧密性中心性，是总结点数，n是一个子图中结点数。应用：关注网络中传播信息最快的结点。

Betweenness Centrality中介中心性，检测结点对图中信息或资源流的影响程度的方法。通常用于寻找连接图的两个部分的桥梁结点。首先计算连接图中每对结点之间的最短（最小权重和）路径。每个结点都会根据这些通过结点的最短路径的数量得到一个分数。计算，是结点和之间最短路径数量，是其中经过结点的数量。图 4图 10为结点D的计算过程。避免在大图中计算时昂贵，Randomized-Approximate Brandes通过随机和度的筛选对其改进。应用：发现瓶颈、控制点和漏洞；识别不同组织的影响者，其往往是各组织的媒介，如电网的关键点，从而提高整体鲁棒性。



图 10. 计算过程

HITS，该算法会给每个结点两个属性：Hub（枢纽性）和Authority性（权威性），和PageRank一样，是链接分析中非常基础且重要的算法。在链接分析领域中，“Authority”页面，是指与某个领域或者某个话题相关的高质量网页，比如搜索引擎领域首页、视频网站首页。“Hub”页面指包含了很多指向高质量“Authority”页面链接的网页，比如导航网站首页。

PageRank见之间总结，上述四种算法总结如图 11所示。Degree:连接数，A有高degree；Closeness:最易到达图或子图中其他结点的结点，B在其子图中有最少跳数hops，是最紧密的；Betweenes:结点和图中有对流量的最强控制的结点，C是桥；PageRank:最重要结点，D由于入度数量或权重是最重要的，E由于与D连接的影响次重要。

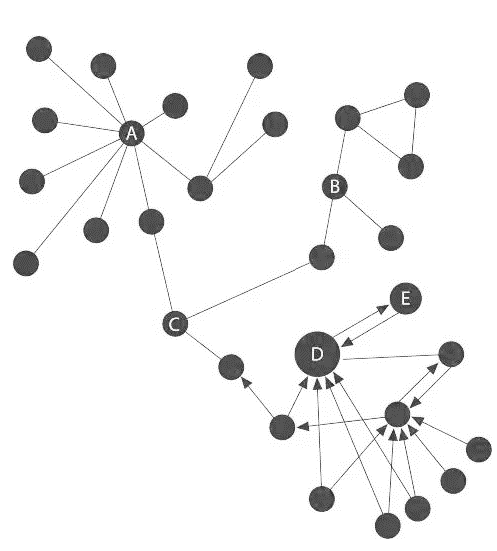


图 11. 中心性算法

Eigenvector centrality，测量给定结点在网络中的重要性。Neighborhood Connectivity，这个度量值是邻居结点的度的均值。Vertex Embeddedness，这个度量方式是指，对于一个结点，它的所有邻居的邻居和自己的邻居的重合个数，除以邻居的邻居和自己的邻居的并集的个数，最后求和后，再除以自己邻居的个数。来衡量我认识的人是不是我邻居都认识。值越大，说明该结点位于一个互关联越强的网络中。Local Clustering Coefficient，和Vertex Embeddedness类似，也是衡量邻居之间的紧密程度。

**边**评估类，**链路预测**问题：Common Neighbours，两个结点的邻居重合个数。Adamic/Adar，衡量共同邻居数量的另外一个版本，给予越孤僻的邻居更高的权重，下方有详细介绍。

链路预测问题。假设网络时刻变化，现存在某个时点的社交网络，推断当前没有链接成员之间在未来发生的新的链接。应用：社交网络的朋友推荐和购物网站的商品推荐；可根据学术界学者们过往合作论文的关系来预测或推荐未来的合作者；可寻找到网络中隐藏或缺失的链接，如用来完善恐怖组织人员网络。Common Neighbors算法，计算两个结点共有的相邻结点数，越多越容易产生链接。Preferential Attachment算法，计算两个结点度的乘积，越大越容易产生链接。Adamic-Adar算法，计算两个结点共有的结点的度，取倒数之后相加。一个共有结点的度越大，倒数越小，可理解为如果一个共同好友的朋友越多，则这个好友介绍给两个人认识的概率越低。搭配监督学习，将指标作为输入变量用于模型训练。本段参考自网站[6]。

**聚类**评估类：模块度评估Modularity：通常值位于-1到1之中，靠近1，说明图存在较强的社群特性，否则近似于没有社群特性。Freeman’s network centrality：该度量方式则是衡量一个网络中，结点的出入度有多不同。在一个星型网络中，这种评估方式得到的值为1。

结点、边、聚类评估类部分内容出自网站[7]。

**社区发现**Community Detection算法，有助于发现社群中群体行为或者偏好，寻找嵌套关系，或者成为其他分析的前序步骤。可评估群体行为或突发事件，也常用于网络可视化。

包括Measuring Algorithm，有三角计数Triangle Count和聚类系数Clustering Coefficient。三角计数计算图中由结点组成的三角形的数量，要求任意两个结点间有边连接。聚类系数算法的目标是测量一个组的聚类紧密程度，计算网络中三角形的数量与可能的关系的比率。为 1时 表示这个组内任意两个结点之间有边相连。包括局部聚类系数和全局聚类系数，局部聚类系数计算一个结点的邻居之间的紧密程度，计算时需要三角计数。，代表需要计算聚类系数的结点，代表经过结点和它邻居的三角形个数，表示结点的度。图 12是结点u在几张图中的计算过程。全局聚类系数是局部聚类系数的归一化求和。三角计数可计算一个组的稳定性或者聚类系数，在社交网络分析中可检测社区。聚类系数可以快速评估特定组或整个网络的内聚性，这些算法可以共同用于特定网络结构的寻找，如探索网页的主题结构，基于网页之间的相互联系，检测拥有共同主题的 网页社区。



图 12. 计算过程

Components Algorithm，有强关联部件Strongly Connected Components算法和关联部件算法。SCC算法寻找有向图内的一组一组结点，每组结点可以通过关系互相访问，不需要直接相连。关联部件算法中，组内的结点对只需通过一个方向访问即可。关联类算法用以了解图的结构，或确定可能需要独立调查的紧密集群。对于推荐引擎也可用来描述组中的类似行为。算法一般被用于查找集群并将其折叠成单个结点，以便进一步进行集群间分析。

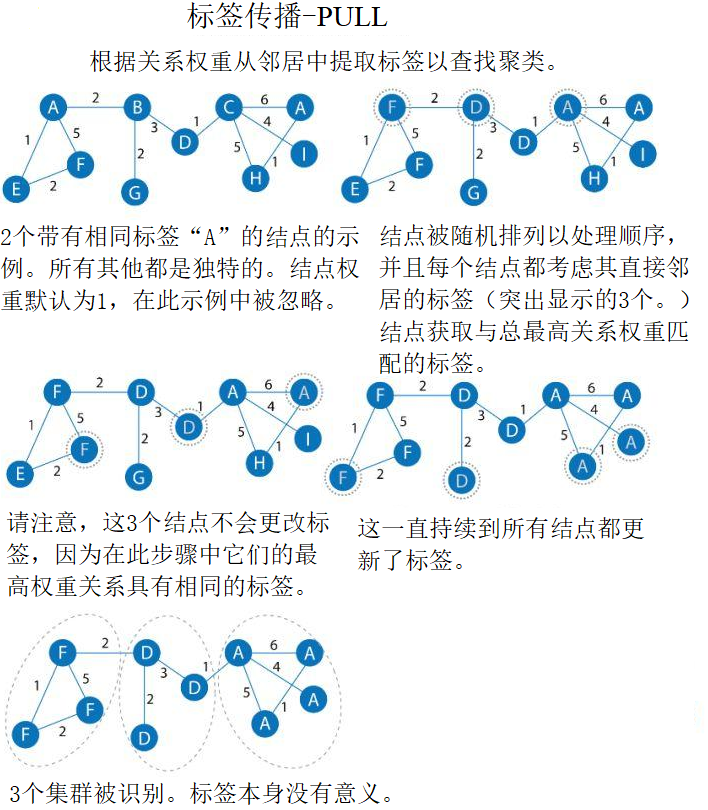
Label Propagation Algorithm标签传播算法可在图中快速发现社区。结点的标签完全由它的直接邻居决定。算法非常适合于半监督学习，可使用已有标签的结点来种子化传播进程。当标签在紧密联系的区域，传播非常快，但到了稀疏连接的区域，传播速度就会下降。当出现一个结点属于多个社群时，算法会使用该结点邻居的标签与权重，决定最终的标签。传播结束后，拥有同样标签的结点被视为在同一群组中。算法的两个变种Push 和 Pull。其中 Pull 算法更为典型，并且可以很好地并行计算。标签传播算法可称作图像分割算法的变种，Push 算法是区域生长法Region Growing的简化版， Pull算法更像是分割和合并divide-and-merge或split-merge。LPA可以与种子标签加未标记的结点一起运行，也可以与每个结点一起以唯一标签开始运行。标签越独特，使用的冲突解决方案就越多。



Pushes Labels TO Neighbors to Find Clusters

* Example of 2 nodes given seed labels.
* They look for immediate neighbors as targets to spread their label to.
* Where there is no conflict the label spreads.
* The recently labeled nodes now activate like a new seed.
* Conflicts are resolved based on a set measure such a relationship weights.
* The process continues until all nodes are updated. 2 clusters are identified.

LPA can be run with seed labels plus unlabeled nodes or each node starting with a unique label. The more unique labels, the more conflict resolution used.

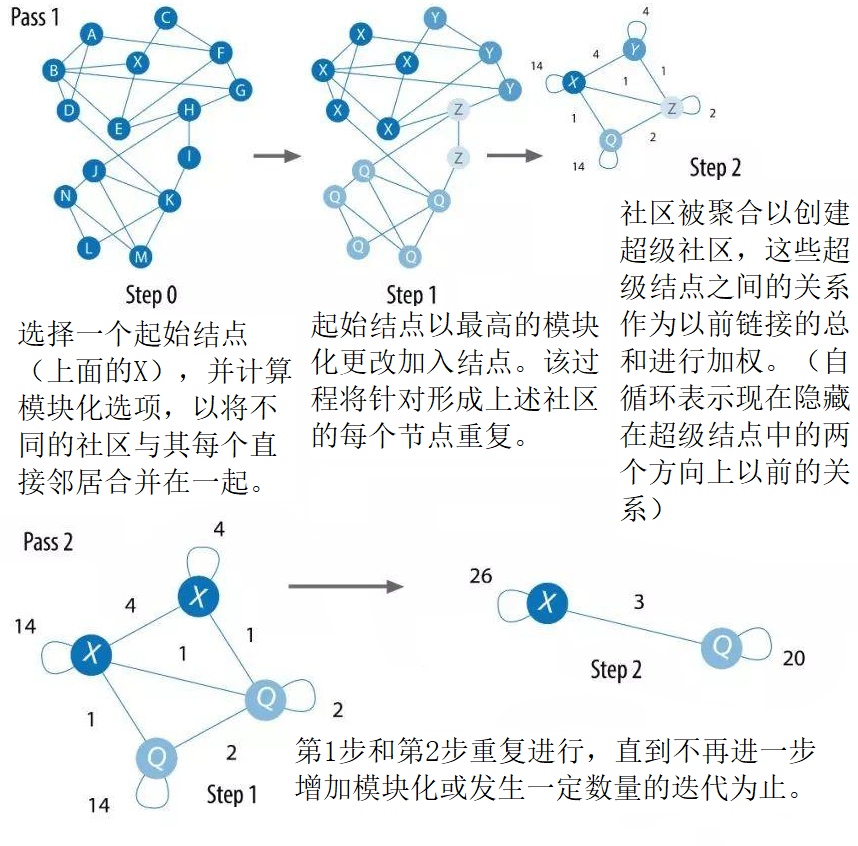


Pulls Labels FROM Neighbors based on Relationship Weights to Find Clusters.

* Example with 2 nodes with same label "A". All others are unique. Node weights default to 1 and in this example are ignored.
* Nodes are shuffled for processing order and each node considers its direct neighbors's labels. (3 highlighted above.) Nodes acquire the label matching the total highest relationship weights.
* Note that these 3 nodes don't change labels because their highest weight relationships in this step have the same label.
* This continues until all nodes have updated their labels.
* 3 Clusters are identified. The labels themselves have no meaning.

PUSH和PULL算法图片和说明修改自[8]。

Louvain Modularity Algorithm算法在给结点分配社群时，会比较社区的密度，而不仅仅是比较结点与社区的紧密程度。算法通过比较结点与社区内关系的密度与平均关系密度，来量化决定一个结点是否属于社区。算法不但可以发现社区，更可以给出不同尺度不同规模的社区层次，对于理解不同粒度界别的网络结构有极大的帮助。算法非常适合庞大网络的社区发现，算法采用启发式方式从而能够克服传统 Modularity 类算法的局限。应用：检测网络攻击，可以应用于大规模网络安全领域中的快速社区发现。一旦这些社区被发现，就可以用来预防网络攻击。主题建模，从在线社交平台中提取主题，基于文档中共同出现的术语，作为主题建模过程的一部分。



Pass 1

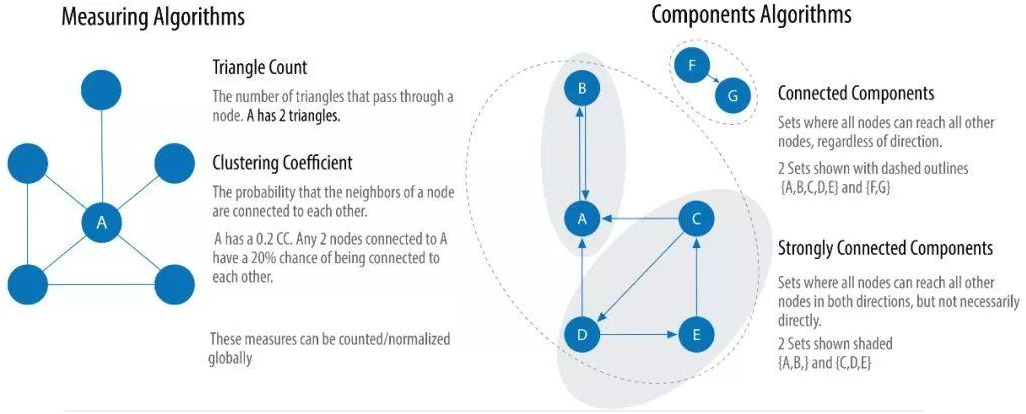
* Choose a start node (X above) and calculate the modularity options for joining different communities with each of its immediate neighbors.
* The start node joins the node with the highest modularity change. The process is repeated for each node with the above communities formed.
* Communities are aggregated to create super communities and the relationship between these super nodes are weighted as a sum of previous links. (Self-loops represent the previous relationship in both directions now hidden in the super node)

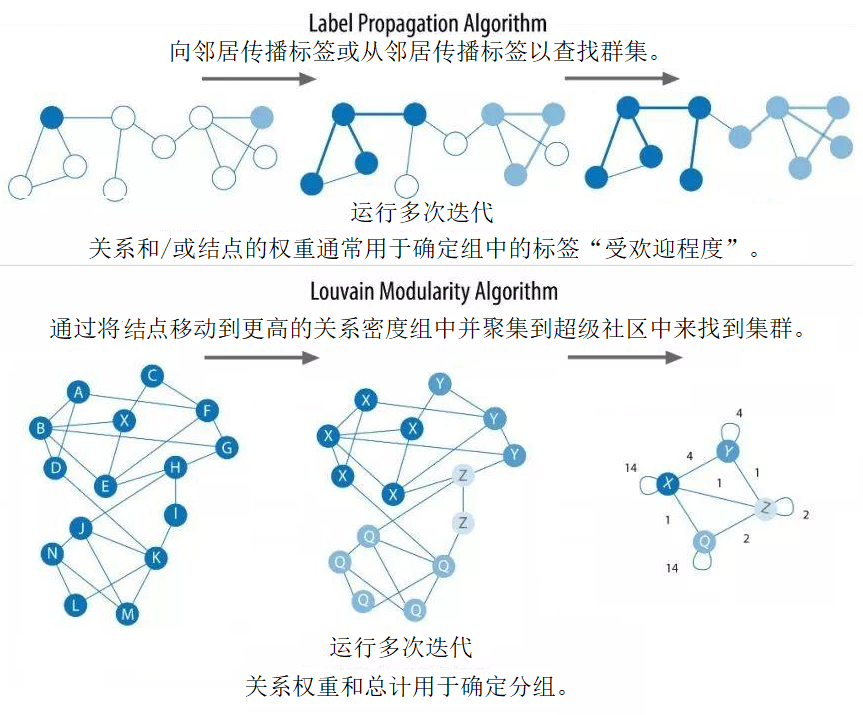
Pass 2

Step 1 and 2 repeat in passes until there is not further increase in modularity or a set number of iterations have occured.

图片和说明修改自[9][8]。

上述四种算法总结如图所示。





Spread Labels To or From Neighbors to Find Clusters.

Run over multiple iterations.

Weights of relationship and/or nodes are often used to determine label "popularity" in a group.

Find clusters by moving nodes into higher ralationship density groups and aggregating into super communities.

Relationship weights and totals are used to determine grouping.

本章D节部分内容至此参考自网站[10]和[11]。

**可视化工具**：原生可视化工具、扩展工具Neo4j Bloom。对其他可视化工具的支持分为三类，分别为内置连接的可视化库（如Neovis.js）、没有直接连接的可视化库（如Sigma.js）以及单独的第三方产品（如GraphXR）。此外，Gephi主要实现网络可视化探索[6]。

详细介绍可参考https://neo4j.com/developer/tools-graph-visualization/ ，例如Neovis.js的页面为https://github.com/neo4j-contrib/neovis.js/

**知识图谱**，揭示实体之间关系的语义网络。信息和知识。知识图谱在逻辑结构上分为模式层与数据层，数据层主要由一系列事实组成，知识将以事实为单位进行存储。如果用（实体1，关系，实体2）、（实体、属性，属性值）这样的三元组来表达事实，可选择图数据库作为存储介质。模式层构建在数据层之上，主要是通过本体库来规范数据层的一系列事实表达。本体是结构化知识库的概念模板，通过本体库而形成的知识库不仅层次结构较强，并且冗余程度较小。体系架构图、知识抽取（实体、关系、属性抽取）、知识表示和知识融合、应用见网站[12]。

关系抽取deepdive：具有语言识别能力的信息抽取工具。通过弱监督学习，从非结构化的文本中抽取结构化的关系数据[13]。

## 算法的使用

Label Propagation算法

语句：

MERGE (nAlice:User {id:'Alice'}) SET nAlice.seed\_label=52

MERGE (nBridget:User {id:'Bridget'}) SET nBridget.seed\_label=21

MERGE (nCharles:User {id:'Charles'}) SET nCharles.seed\_label=43

MERGE (nDoug:User {id:'Doug'}) SET nDoug.seed\_label=21

MERGE (nMark:User {id:'Mark'}) SET nMark.seed\_label=19

MERGE (nMichael:User {id:'Michael'}) SET nMichael.seed\_label=52

MERGE (nAlice)-[:FOLLOW]->(nBridget)

MERGE (nAlice)-[:FOLLOW]->(nCharles)

MERGE (nMark)-[:FOLLOW]->(nDoug)

MERGE (nBridget)-[:FOLLOW]->(nMichael)

MERGE (nDoug)-[:FOLLOW]->(nMark)

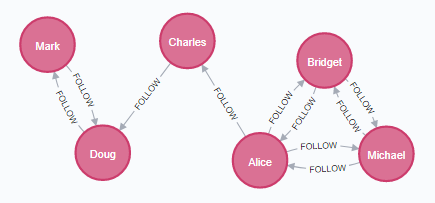
MERGE (nMichael)-[:FOLLOW]->(nAlice)

MERGE (nAlice)-[:FOLLOW]->(nMichael)

MERGE (nBridget)-[:FOLLOW]->(nAlice)

MERGE (nMichael)-[:FOLLOW]->(nBridget)

MERGE (nCharles)-[:FOLLOW]->(nDoug);



运行LPA并在用户中查找社区。执行以下查询：CALL algo.labelPropagation.stream("User", "FOLLOW",{direction: "OUTGOING", iterations: 10})

运行代码后提示Neo.ClientError.Procedure.ProcedureNotFound: There is no procedure with the name `algo.labelPropagation.stream` registered for this database instance. Please ensure you've spelled the procedure name correctly and that the procedure is properly deployed.

查看本地Neo4j版本，Neo4j Browser version: 3.2.20和Neo4j Server version: 3.5.9 (community)。在网站[14]中的releases中下载对应的graph-algorithms-algo-3.5.4.0.jar。移动到neo4j-community-3.5.9\plugins目录下，即$NEO4J\_HOME/plugins。接着配置$NEO4J\_HOME/conf/neo4j.conf文件，最后一行添加dbms.security.procedures.unrestricted=algo.\*后重启neo4j。可在浏览器中输入CALL algo.list()来验证是否安装成功。



运行代码后显示结果如表 II所示。算法找到了两个社区，每个社区三个成员。

表 II

运行结果

|  |  |
| --- | --- |
| nodeId | label |
| 253 | 5 |
| 254 | 5 |
| 255 | 4 |
| 256 | 4 |
| 257 | 4 |
| 258 | 5 |

目前问题：只能根据nodeId查看，可转化为id或name等属性进行显示。

解决方法：可使用如下语句CALL algo.labelPropagation('User', 'FOLLOW', {iterations: 10, writeProperty: 'community', write: true, direction: 'OUTGOING'}) YIELD nodes, iterations, loadMillis, computeMillis, writeMillis, write, writeProperty;此语句过程将结果作为结点属性写回到图形，并报告统计信息。



解决方法2：使用如下语句

CALL algo.labelPropagation.stream("User", "FOLLOW",{direction: "OUTGOING", iterations: 10})

YIELD nodeId,label

RETURN label,

collect(algo.getNodeById(nodeId).id) AS libraries

ORDER BY size(libraries) DESC

运行代码后显示结果如表 III所示。算法找到了两个社区，每个社区三个成员，并且可以清楚看到每个成员所属的社区。

表 III

运行结果

|  |  |
| --- | --- |
| label | libraries |
| 1 | ["Alice", "Bridget", "Michael"] |
| 3 | ["Charles", "Doug", "Mark"] |

对第I章中的数据进行测试，

CALL algo.labelPropagation.stream("站台", "开往",{direction: "BOTH", iterations: 10})

YIELD nodeId,label

RETURN label,

collect(algo.getNodeById(nodeId).name) AS libraries

ORDER BY size(libraries) DESC

运行代码后部分结果如表 IV所示。算法找到了八个社区，每个社区多个成员，并且可以清楚看到每个成员所属的社区。

表 IV

运行结果

|  |  |
| --- | --- |
| label | libraries |
| 17 | ["四河", "华阳", "海昌路", "广福", "红石公园", "麓湖", "武汉路", "天府公园", "西博城", "广州路", "兴隆湖", "科学城", "广都", "五根松"] |
| 103 | ["万盛", "杨柳河", "凤溪河", "南熏大道", "光华公园", "涌泉", "凤凰大街", "马厂坝", "非遗博览园", "蔡桥", "中坝", "成都西站", "清江西路"] |
| … | ………… |

**示例数据出自[15]。**

安装过程出自网站[16]和[17]。

详细算法说明见网站 [18]。

## Neo4j Bloom

**http://s3-eu-west-1.amazonaws.com/ux.neotechnology.com/assets/neo4j-insight-latest/index.html**

**https://blog.csdn.net/superman\_xxx/article/details/81320171**

**http://neo4j.com.cn/topic/5aee4af351bad0a10b198cf4**

**https://github.com/Maicius/QQZoneMood**

## 图算法系列

如何通过neo4j做图计算

来自：https://www.jianshu.com/p/71db5b7354e8

* spark中graphx对neo4j的数据进行读取，然后通过graphx的相关算法进行计算，最后把结果会写到neo4j
* python中igraph进行图计算，回写到neo4j(from igraph import Graph as IGraph pg = ig.pagerank())
* 利用neo4j本身提供的apoc算法，目前已经不推荐使用(apoc-3.4.0.1-all.jar apoc.algo.pageRank)
* 利用neo4j的增强算法，官方推荐(graph-algorithms-algo-3.4.7.0.jar algo.pageRank)

**http://neo4j.com.cn/?tab=good**

**neo4j-graph-algorithms文档见网站[14]。**

Graph Algorithms, Practical Examples in Apache Spark & Neo4j参考书见网站[19]。

## 其他

D3.js

**https://blog.csdn.net/qq\_35318838/article/details/79820923 用来读取页面的所有控件等**

# Appcrawler参数表

## Appcrawler运行参数

用法：java -jar appcrawler.jar [options]，具体内容见表 V。

表 V

参数表

|  |  |  |
| --- | --- | --- |
| 参数 | 说明 | 示例 [options] |
| -a | --app，Android或者iOS的文件地址，可以是网络地址，赋值给appium的app选项 | -a APIDemos.apk |
| -c | --conf，配置文件的地址，支持YAML和JSON格式，不指定即使用默认配置 |  |
| -a <value> | --app，Android或者iOS的文件地址，可以是网络地址，赋值给appium的app选项 | -a APIDemos.apk |
| -c <value> | --conf，配置文件的地址，支持YAML和JSON格式，不指定即使用默认配置 | -c APIDemosConfig.yml |
| -e <value> | --encoding，设置编码格式，例如UTF-8 GBK | -a APIDemos.apk -e UTF-8/GBK |
| -p <value> | --platform，平台类型，Android或者IOS，默认会根据App后缀名自动判断 | -a APIDemos.apk -p Android/IOS |
| -t <value> | --maxTime，最大运行秒数，超过此值会退出，默认最长运行3个小时 | -a APIDemos.apk -t 10800 |
| -u <value> | --appium，appium的url地址 | -c APIDemosConfig.yaml -a APIDemos.app -u 4730 |
| -a APIDemos.apk -u http://127.0.0.1:4730/wd/hub | |
| -o <value> | --output，遍历结果的保存目录，用来存放遍历生成的截图和日志 | -a APIDemos.apk -o result/ |
| -c conf/ APIDemosJson.json -p android -o result/ | |
| --capability k1=v1,k2=v2... | 使用这个参数会覆盖-c指定的配置模板里的参数，用于模板配置里的capability参数微调 | -a APIDemos.apk --capability noReset=true |
| -c APIDemosConfig.yaml --capability udid=[你的udid] -a APIDemos.app | |
| --capability "appPackage=com.Demo.android,appActivity=.view.WelcomeActivityAlias" #启动已经安装过的app | |
| -r <value> | --report，输出html和xml报告的目录 | -a APIDemos.apk -r result/report/ |
| 从已经结束的结果中重新生成报告 | --report result/ |
| -y <value> | --yaml，代表配置的yaml语法，用于不使用配置文件的情况下添加约束 | -a APIDemos.apk -y "blackList: [ {xpath: action\_n} ]" |
| -a APIDemos.apk -y "blackList: [ {xpath: action\_n},{xpath: '.\*[0-9\\.]{2}.\*'}]" | |
| --template <value> | 输出代码模板 | -a APIDemos.apk --template "xx" |
| --master <value> | master的elements.yml文件地址，与--candidate参数一起使用，生成两者的Diff报告 |  |
| --candidate <value> | candidate环境的elements.yml文件，同上 | --master master.yml --candidate candidate.yml -r result/diff/ |
| -v | --verbose-debug，是否展示更多debug信息 | -a APIDemos.apk -v |
| -vv | --verbose-trace，是否展示更多trace信息 | -a APIDemos.apk -vv |
| --demo | 生成demo配置文件的示例到当前目录，学习使用方法 | --demo |
| --help | 输出帮助文档 | --help |

## Appcrawler的yaml配置文件参数

路径下APIDemosConfig.yml文件，见表 VI。

表 VI

配置文件中参数

|  |  |
| --- | --- |
| pluginList: #插件列表，暂时禁用  useNewData: false #是否使用新数据格式  logLevel: "TRACE" #设置log的层级  saveScreen: true #是否截图  screenshot: true  showCancel: true #是否展示取消的操作  reportTitle: "" #结果目录  resultDir: ""  maxTime: 10800 #最大运行时间  maxDepth: 10 #默认的最大深度10, 结合baseUrl可很好的控制遍历的范围  resultDir: "" #结果目录  findBy: "xpath" #可选 default|android|id|xpath，默认状态会自动判断是否使用android定位或者ios定位  capability: #appium的capability通用配置，需要指定appPackage和appActivity  noReset: "true"  fullReset: "false"  appium: "http://127.0.0.1:4723/wd/hub"  baseUrl: #设置一个起始url，指定遍历的初始状态  urlBlackList: #url黑名单，用于排除某些页面  blackList: #黑名单列表 matches风格, 默认排除内容是2个数字以上的控件.  - xpath: ".\*[0-9]{2}.\*"  - xpath: //android.widget.ImageView[@resource-id="com.zol.android:id/multi\_image\_add\_image"]  urlWhiteList: #url白名单  appWhiteList: #app白名单  backButton: #后退按钮标记，目前具备了自动判断返回按钮的能力  - xpath: "Navigate up"  firstList: #基于selectedList定位到的元素，优先遍历的元素列表  selectedList: #使用xpath定位期望的遍历列表  - xpath: "//\*[contains(name(), 'Button')]"  - xpath: "//\*[@clickable="true"]//android.widget.TextView[string-length(@text)>0 and string-length(@text)<20]"  lastList: #基于selectedList定位到的元素，最后遍历的元素列表  - xpath: "//\*[contains(@resource-id, 'header')]//\*"  - xpath: "//\*[contains(@resource-id, 'indicator')]//\*"  beforeStartWait: 6000 #等待启动，待废除  beforeRestart： #在重启session之前做的事情  500 #执行action之后的等待时间  beforeElement: #在执行action之前默认执行的动作，比如等待 | afterElement: #在执行action之后默认执行的动作，比如等待  afterElementWait：afterAll: #所有动作执行完后，是否需要刷新或者滑动  afterAllMax： #afterAll执行多少次后才不执行，比如连续滑动2次都没新元素即取消  triggerActions: #引导规则，action动作，xpath定位，times执行次数  - given: []  when: null  then: []  xpath: "permission\_allow\_button"  action: ""  actions: []  times: 3  - given: []  when: null  then: []  xpath: "允许"  action: ""  actions: []  times: 3  tagLimitMax: 2 #相似控件最多点击几次  tagLimit: #特殊的按钮，可以一直被遍历  - xpath: "确定"  times: 1000  testcase: #可以自定义测试用例  name: "TesterHome AppCrawler"  steps:  - given: []  when: null  then: []  xpath: "/\*/\*"  action: "Thread.sleep(1000)"  actions: []  times: 0xpathAttributes： #自动生成的xpath表达式里可以包含的匹配属性  - "name()"  - "resource-id"  sortByAttribute: #先按照深度depth排序，再按照list排序，最后按照selected排序。后排序是优先级别最高的  - "depth"  - "list"  - "selected"  suiteName: #用来确定url的元素定位xpath 他的text会被取出当作url因素  - "//\*[@selected='true']//android.widget.TextView/@text"  assertGlobal: #断言，只需要写given与then即可 |

## 配置文件参数类型

配置文件中参数有不同类型，见表 VII。

表 VII

配置文件中参数类型

|  |  |
| --- | --- |
| 参数类型是true或false的有  * useNewData * screenshot * showCancel  参数类型是一个字符串的有  * resultDir * findBy可以是"default", "android", "id", "xpath"中的一个  参数类型是整数数字的有  * maxTime * maxDepth * beforeStartWait * afterElementWait * afterAllMax * tagLimitMax  参数类型是字符串列表的有，长度不限  * xpathAttributes可以包含"name()","name", "label", "value", "resource-id", "content-desc", "text", "id", "name", "innerText", "tag", "class" * sortByAttribute可以包含"depth", "list", "selected" * suiteName * baseUrl * pluginList   以下长度可以改变   * appWhiteList * urlBlackList * urlWhiteList * beforeRestart  参数类型是Step构成的列表的有，长度不限  * assertGlobal 只需要写given与then即可   以下长度可以改变 | * triggerActions包括action、xpath、times * selectedList包括xpath * firstList包括xpath * lastList包括xpath * backButton * beforeElement * afterElement * afterAll * tagLimit包括xpath、times  blackList包括xpath capability的参数是appium的启动参数，由[一个字符串:任意类型]的map组成，长度不限。 noReset: "true"  fullReset: "false"  appium: "http://127.0.0.1:4723/wd/hub"  appPackage: "com.Demo.android"  appActivity: ".view.WelcomeActivityAlias"  newCommandTimeout: 120  launchTimeout: 120000  platformVersion: ""  platformName: ""  autoWebview: "false"  autoLaunch: "true" ReactTestCase(name, List[Step]) testcase的参数由name: "字符串"，和steps构成的列表组成。其中的Step包括xpath、action。  name: "TesterHome AppCrawler"  steps:  -xpath: //android.widget.TextView[@text="动态"]  -xpath: //android.widget.TextView[@text="讨论"] 附：Step 包括given（一般为空）、when（一般为null）、then（一般为空）、xpath、action、actions（一般为空）、times（整数数字）。为空时可以省略。 |

## 使用实例

具体的使用实例参考网站[20]、[21]和[22]。

## aitest系统集成appcrawler流程初步设计

**[项目管理页面]**的列表新增两列，分别为是否生成 AppCrawler 用例(ACC)和是否生成单元测试用例(UTC)，至少勾选一个； [](https://gitlab.com/testerhomee/bupt/aitest/uploads/f063f1212caaf015680a7a6bf8ab05f9/image.png)

当用勾选生成 AppCrawler 用例(ACC)时，在生成用例时跳转到[AppCrawler 配置页面]，配置 AppCrawler 相关参数，其中在指定 App 时跳转到[**应用管理页面]**，用户可在 App 管理页面上传 App 或者选择已有 App ，完成后跳回[AppCrawler 配置页面]，配置完成点击该配置页面的生成用例按钮，进行 AppCrawler 用例生成(如果生成单元测试用例(UTC)也勾选，则同时生成生成单元测试用例)，并跳转到[**用例管理页面]**；

如果没有勾选生成 AppCrawler 用例(ACC)，则直接生成用例，并跳转[到**用例管理页面]**；

在[**用例管理页面]**，AppCrawler 用例(ACC)项始终处于单元测试用例列表(UTC)顶部，用户可以同时选择两种类型的测试用例生成同一任务并执行，也可以分别生成任务并执行；点击执行测试任务时，弹窗显示两种测试用例的配置及用例信息按钮，用户可以修改 AppCrawler 用例(ACT)或单元测试用例(UTC)的数据；

执行测试任务完毕，在[**任务管理页面]**，根据用户选择的用例类型，可以查看 AppCrawler 用例测试报告或单元测试用例报告，或者保持现有查看报告不变，用户点击后跳转到一个完整的[**报告页面]**。

图 13为该过程的程序流程图。



图 13 平台集成AppCrawler过程

# 模拟器

## 模拟器问题

在更新完毕Android Studio和其内的Android Virtual Device Manager后，在Device Manager窗口内启动已有模拟器或者新建模拟器并启动后，会卡死在进度条部分。而在命令行中启动模拟器则会提示Incompatible HAX module version 3,requires minimum version 4。

在网站[23]下载最新HAXM，或者在本地找到Android\sdk\extras\intel\Hardware\_Accelerated\_Execution\_Manager文件夹，打开intelhaxm-android.exe进行安装。或者通过解压软件打开exe程序，点击其内部的setup.exe进行安装。

然而，由于安装或卸载残留问题，此时会提示“安装的产品和安装源不匹配，在提供了匹配的源或安装的产品和源同步之前，不能执行此操作”类似的语句。在下方选择相关的hax64.msi文件，在解压的文件夹内或者User\AppData\Local\Temp\Intel\HAXM\7.5.4中选中该文件都不可以。

最终使用MicrosoftProgram\_Install\_and\_Uninstall.meta.diagcab等卸载工具卸载Intel Hardware Accelerated Execution Manager，重新安装新版即可。

打开模拟器提示The ADB binary found at XX is obsolete and has seriousperformance problems with the Android Emulator，解决方法：双击Shift打开搜索框，输入SDK管理器。在显示的结果中，单击“ SDK管理器”，在打开的窗口中，单击第二个选项卡“ SDK工具”。在第一行看到SDK Build-Tools有可用的更新，选中右下角的“显示包详细信息”，滚动到“ SDK构建工具”下的最后一项，然后选中该框，点击“应用”。

## 目前一些问题

* 控制台中启动时会提示ANGLE: D3D11: rx::Renderer11::generateConfigs和Failed to open /usr/local/google/home/joshuaduong/emu/master/prebuilts/android-emulator-build/qemu-android-deps/windows\_msvc-x86\_64/qemu.conf, err: 2。
* 打开Andorid Studio后会自动下载gradle，在setting中搜索gradle。选择Use local Gradle distribution，Gradle home填写D:/Android/.gradle/wrapper/dists/gradle-4.10.1-all，下方选择Offline wordk，Service directory path填写D:/Android/.gradle。会提示gradle location is incorrect，待修改。
* 模拟器内安装一些app会提示应用未安装，或者拖动安装时弹窗提示The APK failed to install. Error: INSTALL\_FAILED\_NO\_MATCHING\_ABIS: Failed to extract native libraries, res=-113。是由于使用了native libraries 。该native libraries 不支持当前的cpu的体系结构。安卓模拟器的CPU/ABI一般有三种类型，INTEL X86,ARM,MIPS。

为此，下载使用mumu模拟器，其支持ARM架构。进行测试前可以使用adb命令输入adb connect 127.0.0.1:7555进行连接。

## Jenkins安装

将jenkins.war放到D:\Program Files\Jenkins目录下，设置系统环境变量JENKINS\_HOME。

控制台中输入命令java -jar jenkins.war进行安装，记录途中显示的密码为53f0f50b40aa434a9dff637661b5c83d。浏览器中打开http://localhost:8080/，输入密码后进入新手入门指导。选择推荐的插件依赖进行安装。需要保持该控制台开启。

表 VIII

部分插件列表

|  |  |  |
| --- | --- | --- |
| Folders | Ant | Subversion |
| Pipeline | Mailer | Git |
| Build Timeout | Gradle | SSH Slaves |
| Credentials Binding | Build Timeout |  |
| PAM Authentication | OWASP Markup Formatter | PAM Authentication |
| Pipeline: GitHub Groovy Libraries | GitHub Branch Source |  |
| LDAP | Timestamper | LDAP |
| Workspace Cleanup | Pipeline: Stage Vie | Email Extension |
| Localization: Chinese (Simplified) | Matrix Authorization Strategy |  |

在控制台中将其转换为外网，供其他人员使用。运行messenger-1.0-SNAPSHOT.jar后显示yCxok5PPO3LB8G6M317SkJ4fgHss7J0J4EN9gRF5xB2y8guAnhrMSD0vNJFIv5Zu和IP地址信息47.105.117.0、端口号 12121等。

点击已有的一项任务，如图 14。点击左侧配置按钮，在下方构建部分中的Execute Windows batch command部分输入相关命令。

java -version

mvn -v

adb -v

点击保存后选择左侧Build with Parameters，输入参数后点击开始构建。

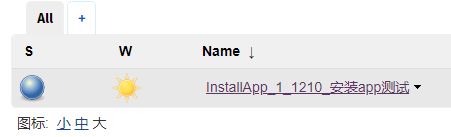


图 14 Jenkins任务

在左侧Build History中可以看到本次构建的编号，进入后选择控制台输出即可以看到输出内容。

问题：输入以上命令后控制台输出最后显示Finished: SUCCESS，但看不到adb相关的版本输出，查看后发现是mvn命令出错。暂时删掉mvn相关命令。

在配置中改为以下命令，在本地开启Android模拟器的情况下，可以实现构建成功时自动安装选定的App。

java -version

adb -v

adb install E:\test-evosuite\xueqiu.apk

使用以下命令，将Apk文件移到`D:\Program Files\Jenkins\workspace\InstallApp\_1\_1210\_安装app测试`中，同时可以在本任务的工作区中看到该文件。构建成功后同时可以安装选定的App。

java -version

adb -v

adb install .\xueqiu.apk

部分输入内容如下：

D:\Program Files\Jenkins\workspace\InstallApp\_1\_1210\_安装app测试>adb install .\xueqiu.apk

Success

D:\Program Files\Jenkins\workspace\InstallApp\_1\_1210\_安装app测试>exit 0

Finished: SUCCESS

## 测试过程：

将appcrawler-2.6.0-jar-with-dependencies.jar文件和demoxq.yml文件等移动到工作区中，配置中添加一下测试代码java -jar appcrawler-2.6.0-jar-with-dependencies.jar --capability "appPackage=com.xueqiu.android,appActivity=.view.WelcomeActivityAlias" -c "demoxq.yml"后可在控制台输出中和本地的Appium查看到测试的过程。需要注意的是首次打开时会有系统权限的提示弹窗。测试完成后可以在工作区中看到新的文件夹20190909090909\_com.xueqiu.android，作为本次的测试报告。

## 使用同一Wifi无线连接真机

手机与电脑连接同一个Wifi并使用数据线连接，使用adb devices命令查看设备是否已经连接。输入adb tcpip 5555命令，断开数据线连接。在手机设置中点击关于本机-状态信息中查看ip地址，使用adb connect 192.168.X.XXX进行连接，最后使用adb devices命令查看设备是否已经连接。需要断开时输入adb disconnect即可，本段参考网站[24]。

## 使用uiautomatorviewer查看App控件相关属性的办法

除了在Appium中新建Session窗口进行查看外，还可以使用uiautomatorviewer。命令行中跳转到D:\Android\sdk\tools\bin路径，输入uiautomatorviewer.bat即可以启动查看器。

点击上方第二个或第三个图标既可以查看移动应用当前页面的控件属性信息等内容，如图 15所示。



图 15 UI Automator Viewer图

## 使用真机测试海关App某长页

使用aapt dump badging命令获取海关App的appPackage为gov.hgm.hgmeap，appActivity为gov.hgm.hgmeap.newHgmm.activity.LoadingActivity。

使用数据线连接安卓真机，打开手机的开发人员选项和USB调试，命令行中输入adb devices后会看到连接的机器4QXDU19624000383 device。

打开appium后点击Start Server v1.13.0，接着在File中选择New Session Window，设置Desired Capabilities配置为

{ "platformName": "android",

"deviceName": "Google",

"appPackage": "gov.hgm.hgmeap",

"appActivity": "gov.hgm.hgmeap.newHgmm.activity.LoadingActivity",

"automationName": "appium"}

点击Start Session，在Appium和真机中都会启动海关App。在页面中获取所需控件的xpath或resource-id，用来编写配置文件。

在cmd中输入以下命令运行AppCrawler。

java -jar appcrawler-2.6.0-jar-with-dependencies.jar --capability "appPackage=gov.hgm.hgmeap,appActivity=gov.hgm.hgmeap.newHgmm.activity.LoadingActivity" -c "demohg1.yml"

其中，使用到了配置文件。在新机上进行测试时或者首次打开App、清理App后重新打开时可以使用demohg0.yml配置文件。其余情况下一般可以使用demohg1.yml，节省单次测试时输入用户名和下载iOS测试插件的时间。

**配置文件demohg0.yml**说明：进入App后开始测试用例，点击弹窗中的我知道了按钮，选择综合服务登录。输入用户名和密码后点击登录按钮。

主页加载完成后点击主页中我的应用下的加号按钮，在应用商店页中下滑两次，找到iOS接口测试并点击安装，等待安装完成后点击打开按钮，进入iOS接口测试页面。选择页面右上角的下拉菜单，选择成为DMZ后向下滑动页面，AppCrawler的测试开始。

在测试过程中，会点击Button和EditText的控件类型；当页面中出现确定、完成、OK等弹窗按钮时会进行点击。测试报告参考20191225162609\_gov.hgm.hgmeap文件。

**配置文件demohg1.yml**说明：进入App后开始测试用例，输入密码后点击登录按钮。

主页加载完成后点击主页中我的应用下的iOS接口测试按钮，进入iOS接口测试页面。选择页面右上角的下拉菜单，选择成为DMZ后向下滑动页面，AppCrawler的测试开始。

在测试过程中，会点击Button和EditText的控件类型；当页面中出现确定、完成、OK等弹窗按钮时会进行点击。

**遇到的问题：**在配置文件编写和遍历时，遇到以下等问题。

* 黑条：在进行遍历时使用mumu模拟器时，进去iOS接口测试页面后会点击页面最上方系统时间和状态栏，引出机器的通知信息栏。导致AppCrawler判断当前已经跳出海关App，导致海关App重启不能够继续进行遍历。
* 横屏：使用mumu模拟器时，搭配uiautomatorviewer，在uiautomatorviewer的界面左侧中能够显示模拟器当前页面，但方框引出的布局是横屏状态下的布局，导致页面和控件错位。考虑到mumu模拟器的首页为横屏显示，改为竖屏显示或者模拟器的分辨率不能解决问题。
* 更新：使用mumu模拟器时，进入海关App并且海关App有新的更新弹窗提示下载时，mumu模拟器无法下载新版本apk，提示下载中断，无法安装更新。由于mumu模拟器不支持模拟器内下载安装，需要随时获取海关App的最新apk文件来避免该问题。
* 更名：海关App更新时，部分控件的text、resource-id等属性会被清空或者更改，导致使用旧版配置文件时无法找到指定控件。需要随着App更新确认控件属性是否更改。
* 自返：进入海关App的iOS接口测试页面后，AppCrawler不会进行任何操作而返回上一页面。为此，在配置文件中添加selectedList和相关内容，要求点击Button和EditText的控件类型。
* 需返：进入海关App的iOS接口测试页面后，点击进入拍照测试等页面，需要进行进一步的拍照操作从而实现测试。拍照完成后，需要返回iOS接口测试页面。**需要对此完善配置文件。**
* 确定：进入海关App的iOS接口测试页面后，某些测试项目会出现确定、完成、OK等带有按钮的弹窗。为此，在配置文件中添加afterAll和相关内容，*xpath: //\*[@text="确定"]*并添加*action: "click"*和*times: 50*。在此之前添加到了triggerAction中，而在其下时点击只会进行一次。需要区分selectedList、triggerAction和afterAll。
* 下滑：进入海关App的iOS接口测试页面后，由于该页面为长页，需要不断向下滑动页面从而使其显示出更多的接口测试选项。为此，在配置文件中的afterAll中添加*xpath: //\**和*action: driver.swipe(0.2,0.5,0.2,0.2)*，作用是在执行完测试用例后在任意界面进行下滑操作。如只需在iOS接口测试页面进行下滑，可以指定为该页面的xpath为//*\*[@resource-id='content']*等。
* 点击：设置完自动下滑后，可能会遇到还没有滑动到需要点击的按钮、已经滑动过需要点击过的按钮，导致按钮还未出现在或已经不在当前模拟器页面中。**需要进一步复现。**
* 格式：输入命令运行AppCrawler时，出现错误提示，包含“错误引用的文字... 应该为双引号”等内容，可根据本地工具或者网站[25]进行格式化校验。也可以在编写配置文件时随时检验。
* 用例：输入命令运行AppCrawler后，不会按照配置文件中指定动作进行交互。查询配置文件格式无误，而发现了不同步骤间的断行和yaml的list分割错误，写到了一起。检查即可解决。

## 测试海关App其他页

对海关App的其他页面进行测试，可以使用demohg2.yml配置文件。

**配置文件demohg2.yml**说明：进入App后开始测试用例，点击弹窗中的我知道了按钮，选择综合服务登录。输入用户名和密码后点击登录按钮。主页加载完成后点击考勤打卡图标，等待安装完成后打开考勤打卡图标，进入页面后点击底栏主页按钮返回；点击海关e学习图标，等待安装完成后打开海关e学习图标，进入页面后点击底栏主页按钮返回。

点击主页底部的消息栏，进入消息页面后选择一条消息打开，进入通知详情页面后返回。

点击底部的应用栏，进入应用商店页后点击安装按钮，等待安装完成后点击打开按钮，进入相应页面后返回应用商店页，下滑两次。点击安装按钮，等待安装完成后点击打开按钮，进入相应页面后返回应用商店页。

点击底部的“我的”栏，进入“我的”页面，点击个人名字，进入个人信息页面，点击左上角图标返回我的页面。点击账号与安全，进入账号与安全页面，点击修改密码，输入原密码、新密码和再次输入新密码后点击提交按钮，返回到账号与安全页面，点击左上角图标返回我的页面。点击新消息通知，进入新消息通知页面，点击声音右侧的开关按钮，点击左上角图标返回我的页面。点击意见反馈，进入意见反馈页面，点击左上角图标返回我的页面。点击网络检测，进入网络检测页面，等待检测后点击左上角图标返回我的页面。点击关于移动门口，进入关于移动门户页面，点击门户介绍，进入门户介绍页面，点击左上角图标返回关于移动门户页面，点击左上角图标返回我的页面。

点击底部的首页栏，进入首页，AppCrawler的测试开始。测试报告参考20201231104305\_gov.hgm.hgmeap文件。

**遇到的问题：**在配置文件编写和遍历时，遇到以下等问题。

* 返回：在海关App的各个页面中，如需要返回上一页面而当前页面没有返回按钮或者返回按钮的xpath等控件属性无法有效获取时，可调用系统的返回按键，如*driver.pressKeyCode(KEYCODE\_BACK)*。
* 双击：进入海关App后，在考勤打卡、海关e学习、应用商店中打开插件页面如ios接口测试页面下方带有页面的底栏，包括返回按钮、主页按钮等。如需点击返回按钮（或系统的返回按键**？需验证**）需要进行双击，而AppCrawler默认有500毫秒的延迟，导致双击时不能被识别为双击。可以暂时改为点击页面的底栏中的主页按钮实现返回动作。
* 解压：进入海关App中，安装海关e学习插件时提示解压失败和安装失败。经测试，需要在下载安装的过程中静止等待，不进行其他操作后即可安装成功。

## 相关网站

* 测试平台网站：进行自动遍历管理和项目配置，执行测试后会在远程Mac Mini中进行相关命令的执行和测试。用户名密码ttt123。

http://aitest.testing-studio.com:8084/

* Jenkins网站；平台中自动遍历管理中执行测试后，在Jenkins/AppCrawlerStart\_8\_aitestDemo2/6dbbcaab7d03,mac-mini-local中查看控制台输出从而查看测试任务的执行。用户名密码hogwarts。

http://192.168.31.51:8081/

* STF网站：可以控制手机模拟器或者连接到Mac Mini上面的手机。可以拖动安装移动应用、查看设备信息等。用户名密码任意。

http://192.168.31.51:7100/

* 可在STF网站中选择设备并查看设备信息，将info下Hardware的Serial中的6dbbcaab7d03填写到测试平台中的项目配置下设备列表中。

## 通过测试平台和Jenkins进行测试并搭配STL

在远程Mac Mini机器中开启Appium、连接安卓真机，同时可以在STF网站中查看该手机。

在测试平台中填写雪球App的包名、Activity名等后选择执行测试，在Jenkins网站中看到执行的命令为java -jar appcrawler-2.6.0-jar-with-dependencies.jar --capability '"appPackage=com.xueqiu.android,appActivity=.view.WelcomeActivityAlias"'

**引号错误：**在Jenkins的控制台输出中有以下错误提示

Exception in thread "main" scala.MatchError: ["appPackage, app, appActivity, appium, deviceName, fullReset, newCommandTimeout, noReset] (of class java.util.Collections$UnmodifiableSet)

可以看到代码中关于包名等项存在单引号和双引号的问题，可能从而会造成MatchError错误。因此使用Team Viewer打开Mac Mini机器，在机器内直接运行带有双引号和单引号的命令，可以看到相同的错误提示。

**解决：**测试在Mac机器上运行只带有双引号的命令，AppCrawler可以正常运行。修改测试平台的后台代码，把带有引号的相关代码删除，此时的命令无引号。首先测试在Mac机器上运行无任何引号的命令，AppCrawler可以正常运行。接着在测试平台中设置并执行了测试无引号的命令后，AppCrawler可以正常运行并对雪球App进行测试。

**闪退错误：**此时打开STF网站查看手机运行情况，发现雪球App打开后即闪退。接着测试在Mac机器上运行无任何引号的命令，AppCrawler可以正常运行，但是雪球App打开后也闪退。最后直接在STF中控制手机打开雪球App仍然闪退。

下载新版雪球App后使用测试平台设置并执行命令，AppCrawler可以正常运行且雪球App可以正常运行。

**配置文件错误：**在测试平台中设置添加配置文件的选项并上传配置文件，在Jenkins的控制台输出中有以下错误提示

Exception in thread "main" org.openqa.selenium.WebDriverException: It is impossible to create a new session because 'createSession' which takes HttpClient, InputStream and long was not found or it is not accessible。

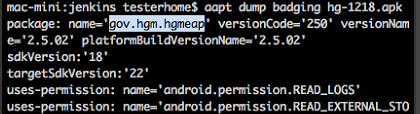
remote stacktrace: UnknownError: An unknown server-side error occurred while processing the command. Original error: Cannot start the 'com.xueqiu.android' application……. Original error: Activity name '.view.WelcomeActivityAlias' used to start the app doesn't exist or cannot be launched! Make sure it exists and is a launchable activity

**解决：**在Mac中重新启动Appium后问题没有解决。此时在Mac机器上输入无引号的capability和双引号的demo.yml、单独有引号和无引号的capability等命令都是同样的错误。查看错误提示，推测是由于雪球App更新更换了appActivity。考虑到获取新的appActivity需要时间，因此在测试平台中添加关于海关App的设置和包名、配置文件等，在Jenkins可以看到命令为无引号的capability和demo.yml，此时AppCrawler可以正常运行。

下一步：获取雪球App的新Activity，在测试平台设置并执行测试。

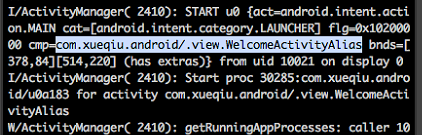
## 补充：获取APK的appPackage和appActivity

方法一： cmd中输入命令cd D:\Android\sdk\build-tools\29.0.2和aapt dump badging hg.apk 。但查找某些App如雪球时无ActivityName的信息。





方法二：通过 adb logcat |grep ActivityManager命令。首先打开模拟器或者连接真机，新建一控制台输入adb shell后输入logcat | grep ActivityManager，再打开某App后查询控制台内cmp字段内容。可忽略/符号。



多台设备连接时，输入adb shell 时则会提示error: more than one device and emulator。此时通过adb devices命令确定需要选择的设备，接着输入adb -s 6dbbcaad7d03 shell即可。

## 并行测试

在进一步完善平台功能后，在测试平台网站中的自动遍历管理中添加新的任务，可以实现多台真机的并行测试。

**问题：**真机运行出现错误，在Jenkins出现关于AppCrawler的错误内容remote stacktrace: UnknownError: An unknown server-side error occurred while processing the command. Original error: Error executing adbExec. Original error: 'Command '/Users/testerhome/Library/Android/sdk/platform-tools/adb -P 5037 -s 4QXDU19624000383 forward tcp\:652 tcp\:6790' exited with code 1'; Stderr: 'adb: error: cannot bind listener: Permission denied'; Code: '1'。相似问题https://testerhome.com/topics/18915。详细错误内容参照构建#52。

可以查看到运行中的所有机器的编号。如DEVICE\_SERIAL=6dbbcaab7d03和DEVICE\_SERIAL=4QXDU19624000383。同时可以查看到每台机器执行的AppCrawler命令，如+ port=51957和java -jar appcrawler-2.6.0-jar-with-dependencies.jar -c demohg0.yml --capability appPackage=gov.hgm.hgmeap,appActivity=gov.hgm.hgmeap.newHgmm.activity.LoadingActivity,udid=6dbbcaab7d03,systemPort=51957 -o 6dbbcaab7d03\_20200102160257\_gov\_hgm\_hgmeap。另外一台机器为+ port=55187和+ java -jar appcrawler-2.6.0-jar-with-dependencies.jar -c demohg0.yml --capability appPackage=gov.hgm.hgmeap,appActivity=gov.hgm.hgmeap.newHgmm.activity.LoadingActivity,udid=4QXDU19624000383,systemPort=55187 -o 4QXDU19624000383\_20200102160257\_gov\_hgm\_hgmeap。

**问题：**在没有添加配置文件的情况下进行两台真机的平行测试。结果两台机器使用的是同一个端口，后启动的机器无法正常运行。错误内容为Caused by: org.openqa.selenium.WebDriverException: An unknown server-side error occurred while processing the command. Original error: Could not proxy command to remote server. Original error: Error: connect ECONNREFUSED 127.0.0.1:54954。

经过多次测试，在无配置文件时启动海关App，可以出现端口相同与不同的情况；在无配置文件时启动雪球App，也会遇到端口相同的情况。

**问题：**多次连续测试时，会出现以下错误：org.openqa.selenium.WebDriverException: An unknown server-side error occurred while processing the command. Original error: Could not proxy. Proxy error: Could not proxy command to remote server. Original error: Error: socket hang up.详细错误内容参照构建#62。是由于appium的session删除需要时间，启动新的测试时需要一段时间等待。

**问题：**执行海关测试时，搭配配置文件demohg2.yml，一台机器的初屏弹窗没有被关掉，日志中显示org.openqa.selenium.NoSuchElementException: An element could not be located on the page using the given search parameters.是由于在多台机器进行并行测试时，需要完成海关App的用户登录，然而配置文件中只有一份用户名和密码，较晚登录海关App的机器会将较早登录海关App的踢下线，需要进一步改进appcrawler。

## 多用户并行测试

经过完善相关代码后，APPcrawler的yaml文件中可对需要参数化的字段使用${}进行处理，接着在测试平台的自动遍历管理中添加Json格式的设备特有配置信息。如在配置文件中添加用户名和密码的参数化字段，即可以实现海关App平台多用户登录情况下的的平行测试。

**配置文件demohg10.yml**说明：与**配置文件demohg0.yml**操作相同，其中对用户名进行操作时action为${username}；对密码进行操作时action为${password}。

**配置文件demohg11.yml**说明：对用户名进行操作时action为${username}；对密码进行操作时action为${password}，登录进入首页后无指定操作。

**配置文件demohg20.yml**说明：与**配置文件demohg2.yml**操作相同，其中对用户名进行操作时action为${username}；对密码进行操作时action为${password}。

**配置文件demohg21.yml**说明：与**配置文件demohg2.yml**操作相同，其中对用户名进行操作时action为${username}；对密码进行操作时action为${password}，修改密码时没有点击确定键避免对密码进行修改。

使用新的配置文件后可以实现在Jenkins中的多用户并行测试。



问题：Jenkins中关于两台设备都出现错误提示：json格式的参数：ac\_device\_conf\_1578454641797.json、设备名称：6dbbcaab7d03、APPcrawler配置文件路径：demohg10.yml、解析结果文件所在路径：/Users/testerhome/home/jenkins/Exception in thread "main" java.lang.NullPointerException。是由于在测试平台的自动遍历管理中没有添加设备特有信息。

# 测测试报告读取

## YAML文件说明

AppCrawler在对移动应用测试完成后，生成了包括elements.yml在内的测试报告文件。其中，elements.yml文件是以YAML标记语言为格式的文件。

YAML标记语言适合描述程序所使用的数据结构，使用空格缩进表示层级关系。以下为YAML常用的数据结构和格式说明。

“---”：在同一个文件中表示多个文档，以三个横线作为分隔符。

“#”：用来表示注释说明的内容。

对象：冒号后有一个空格。

key: value

层级关系：使用缩进表示。

key:

child-key: value1

child-key2: value2

map结构：使用冒号（:）表示键值对，同一缩进的键值对属于同一个map。

age: 12

name: zhang

list结构：使用横线（-）表示。

- key1

- key2

或者

companies:

-

id: 1

name: company1

price: 200W

-

id: 2

name: company2

price: 500W

数据结构的嵌套，如表 IX所示。

表 IX

嵌套表示

|  |  |  |
| --- | --- | --- |
| 内\外 | map | list |
| map | # YAML表示  websites:  YAML: theyaml  Python: thepython  # Json为{ websites: { YAML: 'theyaml', Python: 'thepython' }} | # YAML表示  -  id: 1  name: li  -  id: 2  name: liao  # Json为[ { id: 1, name: 'li' }, { id: 2, name: 'liao' } ] |
| list | # YAML表示  languages:  - YAML  - Python  # Json为{ languages: [ 'YAML', 'Python' ] } | # YAML表示  -  - Ruby  - Perl  -  - c  - java  # Json为[ [ 'Ruby', 'Perl' ], [ 'c', 'java' ] ] |

## 转义处理

一般情况下不加引号。双引号""：不会转义字符串里面的特殊字符，特殊字符作为本身想表示的意思。

name: "123\n123" ，输出 123 换行 123

单引号''：会将字符串里面的特殊字符转义为字符串处理，和不加引号一样。

name: '123\n123'，输出 123\n123

Elements.yml文件中reqDom等项目包含了许多转义字符。可将其还原到字符原本代表的意思。

reqDom: "<hierarchy rotation=\"0\">\r\n <android.widget bounds=\"\

[0,0][10,1706]\" checkable=\"false\" checked=\"false\" click=\"false\"

</hierarchy>\r\n"

首先通过readFileByLines函数按行读取Elements.yml文件，并添加一个特殊符号用来表示换行符，并保存为ElementsCache.yml文件。读取带有特殊符号的ElementsCache.yml文件，将YAML文件中集中多余的空格缩进删除，同时将\"字符替换为"、\r\n字符替换为回车，最终将特殊符号替换为换行符。此时生成的为ElementsOut.yml文件。该文件可以清楚看到其内部的xml格式层次关系。



## Elements.yml文件格式和解析

---

elementStoreMap:

com.zol.android.首页.tag=Start.depth=2.id=Start.name=Start:

reqDom: "<hierarchy rotation=\"0\">\""

resDom: "<hierarchy rotation=\"0\">\""

reqHash: "8e9b61a540855aa3a20bdb4f917616a9"

resHash: "3fd1c4478f2f4b3d3802e7ac52fda096"

reqImg: ""

resImg: "20190926163217\_ =Start.clicked.png"

reqTime: ""

resTime: ""

clickedIndex: 0

action: "CLICKED"

element:

url: "com.zol.android.首页"

tag: "Start"

id: "Start"

name: "Start"

text: ""

instance: "0"

depth: "2"

valid: "true"

selected: "false"

xpath: "//android.widget.FrameLayout.FrameLayout']"

ancestor: "\_hierarchy/android.widget.FrameLayout"

x: 0

y: 0

width: 1440

height: 2392

action: "\_Start"

clickedElementsList:

- url: "com.zol.android.首页"

tag: "Start"

id: "Start"

name: "Start"

text: ""

instance: "0"

depth: "2"

valid: "true"

selected: "false"

xpath: "//android.widget.FrameLayot.FrameLayout']"

ancestor: "\_hierarchy/androi.FrameLayout"

x: 0

y: 0

width: 1440

height: 2392

action: "\_Start"

其中，elementStoreMap和clickedElementsList下各有多组数据信息，以上各展示了其中的一组数据信息。每组包含了其下各子项的信息和内容，其中内容部分可以为空。

导入org.yaml.snakeyaml.Yaml包对yaml文件进行解析，使用yaml.load(new FileInputStream(f))进行解析即可。

遇到的yaml格式问题：while parsing a block mapping、expected <block end>, but found '<scalar>'、can not read a block mapping entry; a multiline key may not be an implicit key at line 6, column 12。是由于yaml文件内缩进格式或符号表示有误，修改后即可，

## 读取保存 clickedElementsList

clickedElementsList下的数据结构较为简单，由list嵌套着map数据类型。编写getYamlclickedElementsListWithClass函数对clickedElementsList下的多组数据信息进行读取。定义Map<String, List<Map<String,Object>>>类型的map变量对Elements.yml文件进行解析，其包含了yml文件的所有数据。

定义List<Map<String,Object>>类型的list变量获取map下clickedElementsList部分的内容，此时list为多组clickedElementsList信息，如*[{url=com.zol.android.首页, tag=Start , action=\_Start}, {url=Steps, tag=android.widget.TextView, action=click}]*。

最后定义Map<String,Object>类型的map2对list进行遍历，通过map2.get("url").toString()等方法获取每组数据信息下各个子项的信息和内容。

最终数据保存在Map<Integer, Elements>类型的map中，其中的Elements是用来保存clickedElementsList各个子项信息的类。

## 读取保存 elementStoreMap

elementStoreMap下的数据结构较为复杂，由多个map嵌套而成。编写getYamlelementStoreMapWithClass函数对elementStoreMap下的多组数据信息进行读取。定义Map<String, Map<String,Map<String,Object>>>类型的map变量对Elements.yml文件进行解析，其包含了yml文件的所有数据。

定义Map<String,Map<String,Object>>类型的map2变量获取map下elementStoreMap部分的内容，此时map2为多组elementStoreMap信息，如*{com.android.首页={ reqHash=8e9b61a, action=CLICKED, element={url=com.android.首页, tag=Start, action=\_Start}}, com.BBSAsk={reqHash=, action=READY, element={url=com.BBSAsk, tag=android.widget.EditText,action=click}}}*。

将map2集合转换为Set集合，获取到Iterator对象，赋值给Iterator<Map.Entry<String ,Map<String,Object>>>类型的it2变量，再以Map.Entry<String,Map<String ,Object>>类型的map3变量接收每个it2变量的内容，此时每个map3为一组elementStoreMap信息，如*com.android.首页={ reqHash=8e9b61a, action=CLICKED, element={url=com.android.首页, tag=Start, action=\_Start}*。

通过map3.getKey()获取到ElementStoreName内容，如*com.android.首页*，接着通过map3.getValue().get("reqDom").toString()获取到每组下除element外其他各个子项的内容。最后将elements中内容暂时以字符串形式切割从而获取各个element下各个子项的信息和内容，将element下子项的内容保存到Elements类中。

最终数据保存在Map<Integer, ElementStore>类型的map中，其中的ElementStore是用来保存elementStoreMap各个子项信息的类，也包含着Element类。

## 补：使用Java API实现简单的增删改查

重要：操作时路径为D:\\Program Files\\neo4j-community-3.5.9\\data\\databases\\graph.db，而不是D:\\Program Files\\neo4j-community-3.5.9\\data\\databases。

代码运行方法：①运行java代码②cmd在neo4j目录下（D:\Program Files\neo4j-community-3.5.9\bin）输入neo4j start③浏览器打开http://localhost:7474/输入match(n) return n查看图④如需再次运行代码则cmd中先输入neo4j stop再运行。

可依次运行E:\IdeaProjects\NeoTest下addData()、queryAndUpdate()和delete()方法，进行添加结点和关系，修改与删除操作。

## 原本图结构建立说明

使用getclickedElementsListYaml函数生成了Map<Integer,String>类型的map变量，此时map为clickedElementsList的部分信息，如*{0=com.zol.首页, 1=Steps知识, 2=Steps问答, 3=Steps课堂, 4=Steps问答, 5=Steps我的问答, 6=Steps用户登录, 7=Steps我的问答, 8=Steps问答, 9=Steps待你回答, 10=Steps我来回答, 11=Steps待你回答, 12=Steps问答, 13=Steps提问, 14=com.zol.BBSAsk1, 15=com.zol.BBSAsk2, 16=com.zol.BBSAsk3}*。

调用ToGraph的addNode函数。相同结点数SameNum初始为零，遍历map中数据，定义真实结点数urlNum1为map中序号减SameNum。调用creatNodeFromMap函数判断是否需要创建结点。将Map的序号和该结点的名字传递给creatNodeFromMap函数，创建事务，在已有图数据库中搜索label为Page，UrlName为该结点名字的结点，如果有则会提示结点已存在并返回false；否则会创建一个Number为序号、UrlName为该结点名字的结点，提交事务后返回true。此时接着在addNode函数中判断结点是否存在，如果已存在则SameNum加一，代表不计数。此时图数据库中已经创建了结点，如*{0=com.zol.首页,1=Steps知识,2=Steps问答,3=Steps课堂,4=Steps问答,已经存在.4=Steps我的问答,5=Steps用户登录,6=Steps我的问答,已经存在.6=Steps问答,已经存在.6=Steps待你回答,7=Steps我来回答,8=Steps待你回答,已经存在.8=Steps问答,已经存在.8=Steps提问,9=com.zol.BBSAsk1,10=com.zol.BBSAsk2,11=com.zol.BBSAsk3}*。addNode函数结尾调用creatRalationByNode(map)函数用来创建结点间关系。

creatRalationByNode(map)函数首先创建事务，进入循环，获取UrlName为0和1的两个结点，获取前者结点的所有出度关系和所有后者结点的入度关系，依次进行判断，是否存在相同的关系，如相同则说明两个结点之间已存在关系；否则创建在这两个结点之间创建一个从前者结点到后者结点的关系。UrlName的结点数字增加，直至获取完所有的结点后提交事务。该函数的运行过程如*{com.zol.首页→Steps知识已连接,Steps知识→Steps问答已连接,Steps问答→Steps课堂已连接,Steps课堂→Steps问答已连接,Steps问答→Steps我的问答已连接,Steps我的问答→Steps用户登录已连接,Steps用户登录→Steps我的问答已连接,Steps我的问答→Steps问答已连接,Steps问答→Steps待你回答已连接,Steps待你回答→Steps我来回答已连接,Steps我来回答→Steps待你回答已连接,Steps待你回答→Steps问答已连接,Steps问答→Steps提问已连接,Steps提问→com.zol.BBSAsk1已连接,com.zol.BBSAsk1→com.zol.BBSAsk2已连接,com.zol.BBSAsk2→com.zol.BBSAsk3已连接}*。如需建立其他关系且图数据库中已存在时，则会提示{*Steps知识→Steps问答的连接已经存在}*。

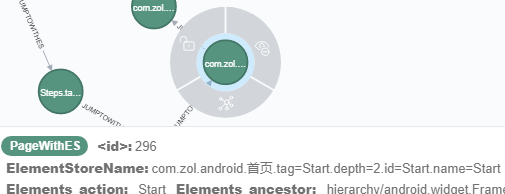
## 添加带有elementStoreMap包含信息的图结构

在编码过程中，遇到以下问题：

[null] is not a supported property value原因：getYamlelementStoreMapWithClass中忘记为Elements中的id赋值，导致为null。

结点已经存在总是识别不出来，原因：graphDb.findNodes中写成了ElementsStoreName。

对element1.yml文件进行运行并测试，可以看到成功创建了17个结点，每个结点中包含了完整的详细信息。

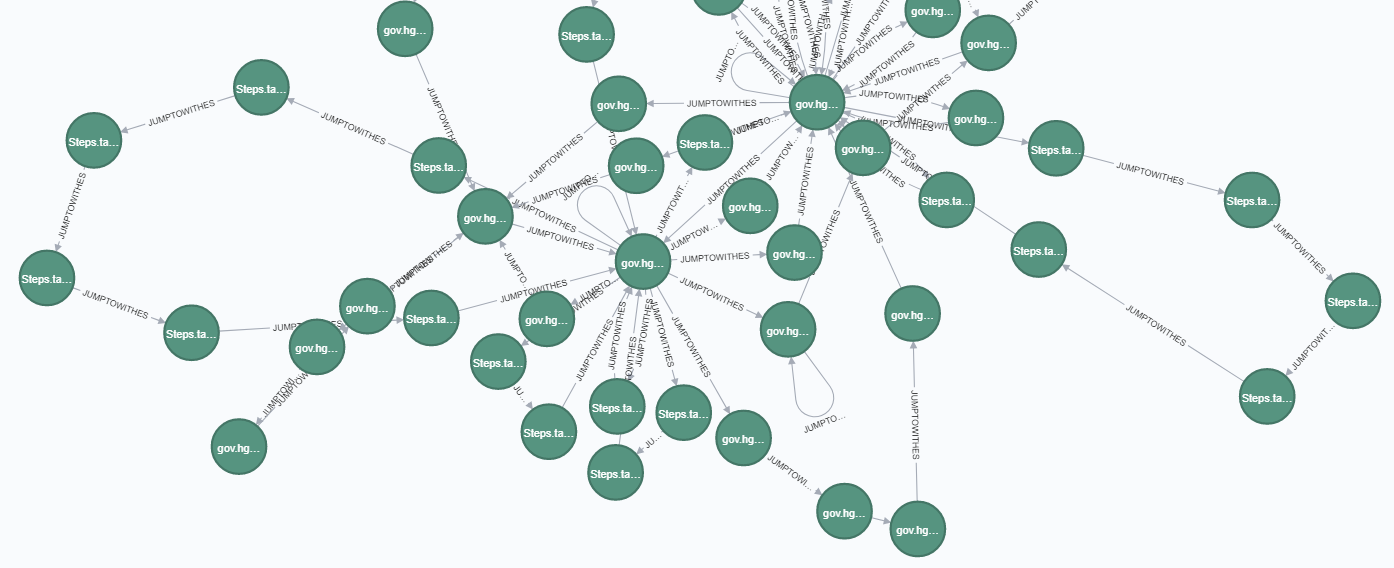


java.nio.charset.MalformedInputException: Input length = 1，原因：yml文件编码为ANSI，修改为UTF-8即可。

对20191225162609\_gov.hgm.hgmeap测试报告中的element.yml文件进行运行并测试，可以看到成功创建了42个结点，每个结点中包含了完整的详细信息。但是报告中有66条数据，经过对比，发现element.yml中的结点不是按照顺序的，而且不包含重复的项目。将报告中的66条数据去重后（不包含序号名）后也和42个结点不对应。获取clickedElements后发现也包含66条数据。

ElementStore中的element内的Ancestor与clickedElements中的Ancestor是一对多对应关系（疑），所以可以根据clickedElements的顺序建立图结构的连接顺序，并根据Ancestor在ElementStore中匹配相同的Ancestor，从而获取每个结点的完整信息。

通过y20201231104305进行测试时，可以得到AppCrawler遍历过程中的图结构。



通过y20191225162609进行测试时，发现很多结点都连接到了gov.hgm.hgmeap.LoadingActivity.tag=Start.depth=2.id=Start.name=Start，即第一个结点。

发现有自己连接自己的情况。

带有step的前缀可以删掉前缀，因为是同一个页面？。实际上在y20201231104305中的第56和57条数据在图结构中自己连接到了自己，是因为56\_Steps.tag=TextView.depth=9.id=main\_tv\_home.text=首页和57\_gov.hgm.hgmeap.MainActivity.tag=TextView.depth=9.id=main\_tv\_home.text=首页两个的ancestor是相同的。

经过排查发现，ElementStore中的ancestor、xpath等信息都存在重复现象，但是ElementStore的name是唯一的。因此，可以避免直接比对ancestor，而通过从clickedElements中的信息拼接出ElementStore的name，最终得到每个结点的详细信息。经过测试可以发现，每一条从clickedElements拼接出的信息都可以在ElementStore找到对应的name。即ElementStore中的name与clickedElements中拼接出的name是一对多对应关系。

在拼接过程中，需要包含ElementStore中的url、tag、depth、id、text和name的属性信息，并在每个属性后添加说明信息。如包含tag信息，则在tag内容前添加".tag="的说明字段。如果某项属性为空则跳过该属性。特别的，对于tag信息可以删除其包含的"android.widget."字段。同时也需要将text信息中包含的特殊字符删除。拼接过程参考SetCombinedInfo函数。

拼接完成后会将完整的信息保存到map中当前ElementStore中的combinedInfo字段中。进行匹配时可调用getOrderFromClickElmentByCombinedInfoAndgetInfoFromElementStore函数实现相应的功能。

## 不再查询数据库，使用list和ArrayList构建图结点间的关系（aitest）

在AiTest项目中的AcToYamlUtil.java中，通过调用parseElemetsYaml获取所分析的测试报告文件的结点构成的Map；调用parsetoGetJump获取结点之间的关系构成的ArrayList。

在获取结点之间的关系时，使用新的creatRalationByNodeWithElementStoreMap函数创建结点之间的关系，避免NeoTest项目中的ToGraph.java中的同名方法频繁开启事务查询图数据库。

在函数运行过程中，首先创建ArrayList<AcJumpToWithES>类型的Jumpto作为存储关系的列表。接着对结点构成的Map进行分析，依次获取结点Map中的第一个和第二个结点作为startNode和endNode，最终构造成AcJumpToWithES类型的AcTes关系对象，通过Search函数该对象判断两者之间是否已经存在关系。在Search函数中，会在Jumpto这一个存储关系的列表中寻找是否存在一条关系，满足该对象中AcElementStore类型的startNode和endNode中的ElementStoreName属性是否与刚构造出的AcTes中的相同。如果相同则返回其在关系列表中的索引位置，如果索引大于-1则代表这条关系已经存在， 否则不存在则建立这一条关系。

在建立关系的过程中，需设置AcTes类中包含的其他属性，设定访问次数CallTime为1，增加一条访问顺序CallNum属性为当前的循环自增值。属性设定完毕后将其添加到Jumpto关系列表中。

如果关系已经存在，则在之前CallNum的基础上继续增加访问顺序的数据，同时其访问次数CallTime的变量值加一，最终通过set函数覆盖掉其自身的旧数据。

另外，没有用到的addNodeWithElementStoreMap函数可以实现从从Map中读取结点，创建结点和关系。最终生成无重复、带有序号num的AcElementStore list。该函数是根据NeoTest项目中的ToGraph.java中的同名方法改写的。

## 问题

文件读取和预处理：

readFileByLines和readFileByLines2函数在目前运行过程中还没有调用到，直接通过了getYamlclickedElementsListWithClass方法进行了读取。同时，其中间文件ElementsCache和ElementsOut文件也没有创建和删除，后续可以设定每次读取完文件后进行删除，也可以测试每次生成中间文件时其内容否会自动清空。

运行时若提示Input length = 1，需要将yml文件编码修改为UTF-8。曾尝试直接通过代码判断文件编码，将ANSI自动修改为UTF-8。

ancestor信息的匹配：

解析报告时只根据日志文件对ancestor信息进行了匹配处理，但也添加完善了对指定属性合并处理的方法SetCombinedInfo。

也就是说，SetCombinedInfo在目前运行过程中还没有调用到，如果调用后则需要运行getOrderFromClickElmentByCombinedInfoAndgetInfoFromElementStore函数判断combinedInfo是否相同；否则只通过调用getOrderFromClickElmentAndgetInfoFromElementStore函数判断ancestor属性是否相同。

目前NeoTest和aitest中合并的代码的对比：

NeoTest多了 Ancestor等属性合并信息的比对。

Aitest多了通过 arraylist获取构建关系的部分，不再调用AcToGraphUtil中的代码。

NOTFOUND结点：

有些测试报告中的结点名称可能为NOTFOUND，具体可参考看田利彬的报告文件。

临时调试用：

MATCH (n:PageWithES)-[r:JUMPTOWITHES]-() delete r

MATCH (n:PageWithES) delete n

## **图比较对比**

考虑新建图结构，以不同的类来表示不同之处，在neo4j中显示的颜色也不同

## **移植系统**

1.D3js-neo4j系统的移植

拷贝neo4j-community-3.5.9，在idea中的neo4j通过maven自动下载。maven版本为3.5.4。

bin目录下安装服务，neo4j.bat install-service

错误：neo4j start启动服务后打开浏览器。

输入密码后提示Neo.ClientError.Security.AuthenticationRateLimit:The client has provided incorrect authentication details too many times in a row.

输入密码为neo4j后提示Neo.ClientError.Security.Unauthorized:The client is unauthorized due to authentication failure.

修改neo4j.conf配置文件，取消dbms.security.auth\_enabled=false行的注释后重启服务即可，也不用再输入密码了。

代码移植的时候注意maven的路径配置也需要进行相应的修改。

maven的pom中language level也需要修改版本到8。

2.测试平台aitset系统移植

代码运行时可能会遇到Error executing Maven.The specified user settings file does not exist: D:\Program Files\Maven\.m2\settings.xml等错误，尝试点击toggle offline mode按钮并修改settings.xml里面的offline配置、删除remote.reposiories文件等操作都不能解决。最终，通过删除\Maven\.m2\repository下所有文件，耐心等待重新下载来解决该问题。

另外，需要手动下载jenkins的dependency，并把其文件拷贝在com.offbytwo目录下。

在编码过程中，maven编译时可能会报com.sun.tools.javac.code.TypeTags错误。可能是因为lombok版本太低，不支持java10以上，Maven中引入高版本lombok即可。对于程序包不存在、找不到符号等错误，可能是lombok版本问题、也可修改pom.xml内java版本和lombok版本，也有可能是代码编写错误。

## **远程服务器**

服务器远程地址 http://aitest.testing-studio.com:7474/browser/

Connect URL为neo4j://39.102.48.202:7687或neo4j://aitest.testing-studio.com:7687，用户名密码为neo4j

遇到问题：SessionExpired: WebSocket connection failure. Due to security constraints in your web browser, the reason for the failure is not available to this Neo4j Driver. Please use your browsers development console to determine the root cause of the failure. Common reasons include the database being unavailable, using the wrong connection URL or temporary network problems. If you have enabled encryption, ensure your browser is configured to trust the certificate Neo4j is configured to use. WebSocket `readyState` is: 3

springboot 集成 neoj参考 https://blog.csdn.net/zt15732625878/article/details/98797467

## **调试过程优化**

经过查看，maven目录下与下载的Neo4j-community内文件基本相同。并不同。

查询资料

Java实现的接口和实例参考，使用了SpringBoot的项目，可以直接启动ConsumerClientApplication，不用放到Tomcat下。https://blog.csdn.net/sunroyi666/article/details/80801859

## **搜索查询功能**

考虑使用Cypher，参考https://blog.csdn.net/yitengtongweishi/article/details/80481099

MATCH (p:Page{UrlName:"Steps待你回答"})-[:JUMPTO]->(p1:Page)-[:JUMPTO]->(p2:Page) return p2

搜索返回Steps待你回答结点指向的结点所指向的结点。

代码中实现方法：查询语句放入result = graphDb.execute，并使用result.hasNext()获取即可。return后加入distinct可以去重。

查询方法：

查询结点的字符串类型的属性，需要输入完全对应的属性名和内容。

查询结点的字符串类型的属性，只需包含查询的属性名和内容即可，如果结点相连则也会展示关系。区分大小写 。

查询结点的整数型的属性，包含输入的属性名、比较符号、数值。

待完善：√错误处理、√显示在原图结构中的位置、在结果上接着查询？。

√跳转关系在建立的时候添加次数的属性、顺序的属性（有了顺序可以查看遍历过程的动画）

同时进行多次单独查询时的处理：每次函数单独开启查询事务，查询结束时会关闭数据库。此时若多次调用此类函数则会在第二次查询起提示数据库已关闭。只需将关闭数据库代码移动到程序末尾。

## **Neo4j添加新的扩展**

安装APOC过程：首先下载jar包，接着放在安装目录plugins目录下，最终修改neo4j配置文件。重新启动neo4j服务，结果提示Neo4j service did not start。首先重新修改了一次neo4j配置文件，但仍然无法启动。

查询log发现提示" Commons Daemon procrun failed with exit value: 5" ception: property [elasticsearch.version] is missing for plugin [head]和传递给系统调用的数据区域太小。查询资料https://issues.apache.org/jira/browse/DAEMON-311、https://blog.csdn.net/lvyuan1234/article/details/77742965考虑是jdk问题。本地搜索elasticsearch无数据，因其和neo4j服务启动时的弹窗图标相同。寻找弹窗所在位置，文件为prunsrv-amd64.exe。

查看http://neo4j.com.cn/topic/5c2040d04ac68d0442ff3b85发现提到版本问题，在github的apoc页面寻找与neo4j版本对应的jar包重新安装，启动时可以正常启动，不再提示Neo4j service did not start。但在浏览器的控制台输入return apoc.version()仍然显示不可用。

多处查询，最后参考https://stackoverflow.com/questions/47634496/neo4j-apoc-functions-doesnt-exist将配置文件中关于apoc白名单一行注释掉即可正常运行。

# 图结构可视化

## **web创建**

1.页面的创建过程

参照https://blog.csdn.net/zhdkong/article/details/78901654

问题：找不到Run/Debug Configuration：https://blog.csdn.net/sinat\_34104446/article/details/82840258

添加时找不到Tomcat Server：https://www.cnblogs.com/roseAT/p/10290221.html和https://blog.csdn.net/sinat\_28530913/article/details/80117186，此处安装的为Smart Tomcat，webapps位置也在项目目录下，参考https://majing.io/posts/10000039201249。

由于Smart Tomcat配置信息不完整，不能用，而idea为Ultimate版本却仍然在Plugins里搜不到tomcat and tomee integration，最后发现已经安装在了installed中了，勾选即可。

2.页面上方导航栏的设置

包含了回到主页按钮、跳转到上一个页面按钮。在图结构的首页还有刷新按钮、打开新的测试报告文件按钮。

回到主页按钮的id为btnHome，点击后通过<a href="query.html">跳转到query.html的主页页面。

跳转到上一个页面按钮的id为btnBack。点击后通过<a href="javascript:history.back(-1)">跳转到上一个页面。其中的aria-hidden起到让该元素对浏览器隐藏的作用，主要在辅助设备中的屏幕识读时抓取相关元素并朗读时发挥作用。

在主页中，刷新按钮的id也为btnBack。点击按钮时调用showDefaultGraph()函数。该函数在“显示在原图中的位置（显示完整图结构）”功能用也会调用。

打开新的测试报告文件按钮的id为btnOpen，其图标为加号的形状。在其附近还包含input type= "file"的组件，该组件的id为fileUpload，还包含webkitdirectory directory multiple的属性，并将该组件设置为display:none隐藏效果。点击btnOpen按钮后调用openfilefloder()函数，在函数中实际上会通过$("#fileUpload").click()点击fileUpload组件。

点击按钮后会弹出上传文件夹的窗口，选择测试报告的文件夹后，fileUpload监测到变动，触发change方法。Fileflodername赋值为$("#fileUpload")[0].files[0].webkitRelativePath，表现为“所选择的文件夹名/文件夹内其中一个文件名.后缀名”（可参考https://blog.csdn.net/leixu1027/article/details/80689872）。通过字符串处理获取到该文件夹名并赋值给fileflodername，同时显示到页面导航栏中id为name的div中。

最后调用generateGraph函数，在该函数中可以解析所选择的测试报告文件，并生成相关图数据库数据保存到图数据库中。待移植ToGraph和ToYaml代码到此。在此，临时手动赋值absopath为某测试报告文件的绝对路径，如D:\\IdeaProjects\\20201231104305\_gov.hgm.hgmeap，通过GlobalConfig函数下的sessionStorage.setItem方法将absopath值保存到abpath中。

（早期描述：添加页面的导航栏、所需实现的功能按钮并优化布局，为后续按钮预留位置。

简单学习bootstrap网格、组件属性的操作。组件的学习参考https://v3.bootcss.com/components/，图标引用自https://www.thinkcmf.com/font/font\_awesome/icons.html。）

3.页面参数跳转和sessionStorage

页面参数的传递可以使用jquery.params.js，源码参考http://www.voycn.com/article/html-yemiantiaozhuanshichuandicanshujqueryparamsjs。跳转前设置url = "b.html?name="+name;并通过window.location.href = url跳转，在新的页面使用$.query.get("name")获取指定内容。在query.html主页中点击某一个结点代表的圆形后单击查看测试步骤时会使用此方法，同时在teststep.html的$(function()初始化函数中获取该结点的id。

另一种方法可以使用sessionStorage。其可以针对一个 session 存储数据，这些数据只在同一个会话页面中可以访问，当会话结束后数据也被销毁。

方法包括sessionStorage.setItem("key", "value")、var value = sessionStorage.getItem("key")等。除了上述的路径参数，在判断是否可以生成测试用例时也使用该方法传递了结点列表nodelist参数。

还可以使用cookie、localStorage的本地存储方法和url传参的方法，具体参考https://blog.csdn.net/weixin\_33910137/article/details/92549792。

## **绘图页面**

D3.js的说明了解https://blog.csdn.net/qq\_34414916/article/details/80026029、可视化的说明了解https://blog.csdn.net/qq\_34414916/article/details/81168275。只想看绘图的源码：https://github.com/mp2930696631/d3.js-demo/blob/master/d3.js-demo/testD3\_chp14\_1.html 。说明：https://blog.csdn.net/qq\_34414916/article/details/80036679、https://www.cnblogs.com/eco-just/p/10035970.html

github中neo4j-vis.js项目参考https://github.com/IsFive/neo4j-vis.js，对应的博客https://blog.csdn.net/qq\_31748587/article/details/84538724 | neo4j-vis.js

使用D3搭配neo4j driver的项目案例https://blog.csdn.net/weixin\_30342639/article/details/86756977，其中java连接neo4j使用了bolt、http和https中的bolt协议

两种d3效果的简单展示https://blog.csdn.net/csdjia11/article/details/100097520和http://neo4j.com.cn/topic/59b7aea3ae1b13da5cba6d7c

springboot2.0+Neo4j+d3.js构建知识图谱https://github.com/MiracleTanC/Neo4j-KGBuilder对应的博客https://blog.csdn.net/poxiaomeng187/article/details/83615972

一个收藏较多的d3项目https://github.com/eisman/neo4jd3

使用http连接的项目https://github.com/micwan88/d3js-neo4j-example。可以直接查看到Neo4j中存储的数据库，并支持简单的查询功能。后续以此为基础添加新的功能。

1.重绘

界面中的绘图和属性显示部分适配了不同分辨率。并在重绘时可以及时更新绘图和属性的内容。

其中绘图<svg id="resultSvg"></svg>部分的重绘监听窗口大小并计算新的绘图部分的宽度，其默认值和重绘时的宽度都为(document.documentElement.clientWidth-15)/12\*11。监听到window.onresize变化时，调用initGraph()函数。

而属性<div id="propertiesBox"></div>部分的重绘直接设置（其父div中的style，且）自身的#propertiesBox的width属性为auto、padding-right为0即可。

3.互斥

绘图时互斥函数的简单了解。首先尝试调整结点和关系的绘制时的图形属性，如减小结点的圆形大小，避免大量关系交叉堆叠显示。

3.显示完整图结构

优化页面逻辑，可在刚进入页面时显示所选测试报告文件的完整图结构。在页面初始化$(function()…)函数中添加showDefaultGraph()函数。

在函数中，设置查询框内容为MATCH (f1:PageWithES) MATCH path = (f1)-[]->(f:PageWithES) RETURN path查询语句，若Cypher查询的可选框未勾选时则进行勾选，接着调用submitQuery()进行查询。最后将查询框中的内容和选框的勾选情况复原。将“显示在原图中的位置”按钮改为不可按的状态。

在任何情况下，当submitQuery()进行查询后若显示commit有415 (Unsupported Media Type)错误，并且Content-Type变成了application/x-www-form-urlencoded（而不是application/json）时。需要注意代码中setupNeo4jLoginForAjax(neo4jLogin, neo4jPassword)语句是否被误删了。

4. 图结构下方的属性栏

相关的显示函数为showProperties，图中若获取到的key为reqDom或resDom，则跳过不显示这两项属性及内容。避免属性栏中文字内容过多。

5.关系无法显示问题的解决

打开页面后可以进行查询，但只能显示出结点信息，虽然代码已经写出但无法显示出箭头表示的关系。是由于D3代码中存在以下问题：

问题一：Relationships字段为空

打开浏览器的调试中的Network，查看commit中发现其json的relationships字段为空。

经过查询https://github.com/micwan88/d3js-neo4j-example/issues/6、https://github.com/neo4j/neo4j/issues/7944和 https://stackoverflow.com/questions/44865150/why-does-neo4j-3-1-x-return-empty-for-relationships等，首先将查询语句进行修改，如查询结点时，MATCH (f1:结点) RETURN f1改为MATCH (f1:结点) MATCH path = (f1)-[]->(f:结点) RETURN path即可返回结点以及结点之间存在的关系。具体一些查询语句查看2020.3.30-4.3和2020.4.7-4.10笔记。

但使用这种查询方法会与本来需要查询的内容不一一对应，考虑以下语句，MATCH (f1:结点) where f1.name='1' MATCH path = (f1)-[]->(f:结点) RETURN path。上述语句对f1结点存在约束限制时，即名字为1；而结点f并不满足对应的限制要求，导致查询结果中的结点不完全满足约束限制。若对f也添加相应的约束限制，则会导致查询结果中的结点少于所需结点。

经过再次查询https://stackoverflow.com/questions/45407672/return-nodes-only-without-relationship-in-graph-view-neo4j，发现在Neo4j Brower中的设置内取消勾选Connect result nodes后，查询结果与D3JS中的查询结果相同。该设置的含义是显示所展示的结点间的每一个关系。

问题二：.关系绘制代码错误

进行搜索时，发现搜索PageWithES类时，返回结点没问题，返回关系有问题。搜索测试结点类时，返回结点或关系都没问题。F12查看控制台，发现存在Error <path> attribute d: Expected number, "MNaN,NaN LNaN,NaN …问题。

考虑是否是[结点名字显示中修改D3中绘图代码]问题导致，修改后仍然出错。考虑是PageWithES结点内属性内容过多的问题，且使用MATCH (f1:Musician) MATCH path = (f1)-[]->(f) RETURN path查询简单结点时并不报错。

在页面中拖动结点，发现拖动结点重绘图形时报错会逐渐变得更多。定位代码的drawLine部分，发现其return部分存在M和L字段。逐个输入函数中各个变量值，发现cos和sin存在空值问题，其值来源于dist变量。具体是因为计算结点位置的变动距离或偏移量的deltaX和deltaY时，没有考虑差值为0的情况。因此为dist添加为0时的判断即可。

6.结点名字显示的项目修改

由于D3代码中显示结点名字时默认显示属性中的name字段，而测试报告中的结点中对应的属性为ElementStoreName，导致绘图时显示的名字为空。尝试以下两种解决办法：

方法一：修改D3中绘图代码

经查看代码，相应的字段为 circleText= ….text(function(d) {return d.properties.name;})，其中name为在绘制时显示的结点属性。将其修改为ElementStoreName后可以正常显示测试报告的结点名，但会导致其他类型的结点名无法正常显示。而且，会导致后续查询语句中关于name属性相关的代码出现错误。因此不再采用该种方法。

方法二：修改测试报告类的属性

为不影响之前代码对测试报告类PageWithES中对ElementStoreName进行的操作处理，在创建结点时同时新建name属性，内容与ElementStoreName相同。

7.结点名字显示过长问题

图结构的可视化展示中，结点名字最开始显示为完整的ElementStoreName如“gov.hgm.hgmeap.MainActivity.tag=TextView.depth=16.id=science\_Msg\_MsgType.text=富文本”。修改相关语句，在circleText的设置部分将text属性设置为d.properties.name属性，在上述修改D3中绘图代码的基础上，继续设置该name的属性。设定索引shortIndex为name字段中“\_”和“=”中更为靠后的一个，并以此取子字符串作为新的name，如“富文本”。

## **搜索查询模块**

1.计数

添加了显示搜索结果中的结点数量和关系数量的内容。在搜索完成时，调用编写的promptAlert函数，显示在resultNumContainer中。首先使用了单独的变量进行计数，但出现了多次搜索时计数不清零和结点数计数错误的问题。

其中结点数结束错误的原因是没有考虑到代码中nodeItm.id in nodeItemMap结点是否已经存在的问题，只有在不存在时才添加计数。

后续使用Object.keys(nodeItemMap).length得到搜索结果中的结点数和关系数。

2.属性的显示

添加了搜索内容为空时下方的属性栏清空的功能。

待续

3.显示在原图中的位置

添加“显示在原图中的位置”按钮，点击后显示完整图结构，并把其中为搜索结果的结点以其他颜色显示。在提交查询请求后按钮变为可点击；在选择显示完整图结构时为不可点击。页面初始化时添加$('#btnPos').click(function() {showPosinFullGraph()} );语句。当按钮点击时，调用showPosinFullGraph()函数。

初期，由于代码中存在结点双击的监听，可以展示于其相连的结点，是调用了submitQuery(d.id)函数，在函数内调用了match (n)-[j]-(k) where id(n) = d.id return n,j,k查询语句。因此可以在显示出查询结果后对结果中的每一个结点进行双击，逐步显示出完整的图结构。

查询http://www.imooc.com/wenda/detail/604109使用circlesinResult.each(function(d, i) { var onClickFunc = d3.select(this).on("dblclick"); onClickFunc.apply(this, [d, i]); });语句可实现双击每个结点，但每次只能出现一层新的结点，若需显示出完成的图结构则需要多次调用该语句。且在进行测试时会造成重绘的过程卡顿，图形崩溃弹出绘图窗口等问题。因此尝试使用新的方法。

首先使用circlesinResult通过d3.select('#circle-group').selectAll('circle')记录当前结果中所有的结点的数据。接着调用显示完整图结构的函数showPosinFullGraph()。最后通过使用类似于双击所有结点的语句，使每个结点都处于鼠标处在上面mouseover的状态，最终使结点都处于鼠标划走的mouseout状态。

在调用显示完整图结构函数showPosinFullGraph()的过程中，其内部也调用了submitQuery()函数。提交查询后会调用更新updateGraph()的函数，在绘制结点时通过.attr('fill', getItemColor)指定结点的颜色。

对于getItemColor(d)函数，首先判断该类有没有存在于itemColorMap中，若不存在则分配一种颜色。如果点击了“显示在原图中的位置”按钮，circlesinResult会记录下结点信息，此时使用circlesinResult.each(function(dd,i)进行遍历，查看其中代表搜索结果的dd中的结点id和完整图结构中的结点d的id是否相同，如果相同则将itemColorMap中已为该中类型结点分配的颜色替换为新的颜色，否则返回之前分配的颜色。

测试发现，点击按钮后只有唯一一个结点颜色改变了。调试输出发现，circlesinResult.each中没有终止循环。在结果中的圆dd与所有的圆d循环比对的过程中，在某次已经相同的情况下仍然进行了该循环内的后续比对，覆盖了已经分配为其他颜色的颜色。唯一的一个颜色改变是某次循环中的最后一次比对的结果相同，后续不会被覆盖。循环不支持break，使用return false也不起作用。查询源码得知d3的selection.each不支持中途停止。最终使用flag标记改变，循环结束后再分配颜色即可。

## **测试步骤页面**

分析测试步骤和分析测试用例按钮的可点击情况：

页面初始化后，分析测试步骤和分析测试用例按钮都不可以点击。

在showProperties函数中，根据传入参数isNode判断是否显示的是结点属性，如是则设置分析测试步骤按钮可点击，否则不可以点击。

在点击结点重绘颜色时，getClickedItemColor函数会调用setTestcaseButton函数，其会根据clickednode变量的长度判断当前已点击的结点数量是否大于一，如果大于一则分析测试用例按钮可以点击；等于一则分析测试用例按钮不可点击；其余情况下分析测试用例按钮也不可点击。不需要在setTestcaseButton函数中设置分析测试步骤按钮的可点击性，只根据属性栏和相关函数showProperties来判断可点击性即可。

1.测试步骤页面布局

在分析测试步骤页面中，上方包含了导航栏；左侧有圆形按钮，和查看reqDom、查看resDom、查看DOM文件、查看点击截图、查看日志文件几个按钮；右侧包含显示这些内容的组件；最下方有生成遍历路径、获取未遍历控件、获取下一步遍历路径按钮。

左侧按钮的id分别为btnid、btnShowreq、btnShowres、btnShowdom、btnShowscreenshot、btnShowlog。在点击这些按钮后，左侧还会弹出id为btnClose的关闭显示窗口的按钮。这些按钮的style还添加了position: fixed属性，在页面窗口向下滚动时依然会维持在同一位置。另外，其父级的div还添加了z-index属性，确保其显示层级一直在最上。

页面右侧有<div id="table-div"包含多个div，根据按钮的点击情况显示其中一个。主要包含：

①able id="table"表格。表格分为项目和内容两列，用来显示所分析的结点的属性信息，如reqTime、Elements\_id等。表格每行的<tr>中包含<td>reqTime</td>的属性名称，和如<td><div id="reqTime"></div></td>形式的属性内容。

② div id="table-div-reqDom"，显示reqDom信息。

③ div id="table-div-resDom"，显示resDom信息。

④div id="table-div-screenshot"，显示截图信息。

⑤div id="table-div-Log"，显示Log信息。

其中②和③下还包含一个domBox和domBox2两个子div；④中包含两个带有<img id="image"src="" />的子div；⑤中包含着LogBox和LogBoxCache两个div，包含word-break:break-all; word-wrap:break-word属性避免内容过长导致显示错位。另外，后者的display设置为none，是不可见的。

2.测试步骤页面初始化

默认情况下，进入测试步骤页面时，通过id = $.query.get("id")获取结点的id并通过设置innerHTML属性，显示在左侧的btnid按钮上，该按钮存在border-radius:50%属性，从而会以圆形显示用来代表该测试步骤页面所分析的结点。

table表格通过$("td,th")..addClass("text-center")添加了文字居中的属性，又通过起始位置的<link rel="stylesheet" href="css/main.css">中的th,td {adding: 0!important;}语句设置了表格tr和td的的padding属性，确保表格内每行的文字紧凑显示。表格还包含了word-break:break-all; word-wrap:break-word属性，使较长的文字可以在必要的位置断行显示。其本身的class包含table-hover属性，在鼠标经过表格时会有高亮显示的效果。

获取完毕id信息后，调用submitQuery(id)函数。使用match (n) where id(n) = ' + nodeID + ' return n语句在图数据库中查询。返回查询结果data，令neo4jDataItmArray值为data.results[0].data的值。再使用forEach语句获取后者data中每项的值，以dataItem表示。因此每次的查询结果只有一个结点，所以可将每次的dataItem.graph.nodes中的第一项dataItem.graph.nodes[0]赋值给neo4jNodeItm，此时的neo4jNodeItm直接包含着结点的属性信息。如表格中的reqTime项目可以通过$("#reqTime").text(neo4jNodeItm.properties.reqTime)语句获取值并显示。

同时，domBox和domBox2也会以相同方式获取到reqDom和resDom信息，使用append($('<p></p>').text(neo4jNodeItm.properties.reqDom)) 语句将完整属性内容添加到对应的div中。在获取到ElementStoreName和通过clickedIndex、action等属性拼接的name2属性后，显示在页面右侧上方的<h4>和<h6>作为该结点的名字和简略信息。另外，也将index和name的值另外赋给nodeIndex和nodeName，便于后期调用。

在调试时，dataItem经常为undefined信息，同时使用了JSON.stringify(nodeItemMap)、JSON.parse(JSON.stringify(nodeItemMap))、Object.keys(nodeItemMap)和(Object.values(nodeItem)等方法尝试获取信息，可以参考fillData函数；也通过console.log输出相关信息但显示为{}；在浏览器的F12控制台中直接输入相关参数内容也无法显示。

（例如，想得到nodeItemMap的值，在代码中console.log输出nodeItemMap和nodeItem的不同形式都不可以，其中一种显示{}，点击展开才有内容。而直接在chrome的console中输入变量却可以看到。最终使用neo4jNodeItm=dataItem.graph.nodes[0]得到内容，但只在当前函数内有效(但是函数外的 console.log(nodeItemMap)仍然可以显示)，在fillData函数中console输出仍为空。）以上几种方法都无法获取到所需的信息，是由于submitQuery(id)函数中的$.post查询语句会异步运行，所以在调用fillData函数时无法获取到相关的信息。

3.测试步骤页面左侧按钮的点击与显示

总体上，点击查看reqDom、查看resDom、查看DOM文件、查看点击截图、查看日志文件中的其中一个按钮时，其自身所代表的右侧div的style.display会设置为""，其余的div的style.display会设置为"none"。而关闭显示窗口的按钮也会进行显示；在点击关闭显示窗口的按钮后，其按钮本身也会隐藏。

查看reqDom、查看resDom：由于reqDom和resDom框中的属性内容已经在初始化过程中获取并赋值，所以点击对应的按钮后只会进行div的显示和隐藏的设置。

查看DOM文件：首先通过sessionStorage.getItem('absopath')获取绝对路径，再添加"\\"+nodeIndex+"\_"+nodeName+".dom"字段在测试报告文件夹中找到当前结点对应的DOM文件，赋值给filename作为绝对路径。

在尝试打开DOM文件时，尝试使用了以下方法①var obj=new ActiveXObject("wscript.shell");obj.Run(filename, 1, false );②参考https://blog.csdn.net/u013970232/article/details/89146426的var xhr = new XMLHttpRequest();xhr.open("get", filename, true); xhr.send();xhr.onreadystatechange = function() {if(xhr.readyState == 4 && xhr.status == 200) {console.log(xhr); console.log(xhr.responseText); } else if(xhr.status == 404) {console.log(xhr.status); } };③参考<https://www.cnblogs.com/qiyongliang/p/3822931.html>的var htmlobj=$.ajax({url:filenameChrome,async:false}); var temp=htmlobj.responseText;④window.open(filenameChrome, null);等，但遇到了Not allowed to load local resource和has been blocked by CORS policy等问题。最终在Chrome浏览器中使用Web Server for Chrome扩展应用，设定D盘为根目录，映射为http://127.0.0.1:8887。通过连接路径+文件路径使用④打开文件。最终点击按钮后弹出新的页面显示当前结点对应的DOM文件。

查看点击截图：同样使用了Web Server for Chrome扩展应用，将当前结点的两张截图分别赋值filenameChrome为"http://127.0.0.1:8887"+"/IdeaProjects/20201231104305\_gov.hgm.hgmeap"+"/"+nodeIndex+"\_"+nodeName+".click.png"，再通过document.getElementById('image').src = filenameChrome将子div的src设置为其路径即可显示。

查看日志文件：由于每份测试报告文件中只包含唯一一个appcrawler.log文件，因此需要在日志文件中截取每个结点测试过程中对应的部分。暂时将日志文件放到当前程序同一目录以$("#LogBoxCache").load(filelog,function(response)加载出，此时整个日志文件内容以response显示出。通过日志文件中的"current index ="字符串截取两个不同index之间的内容，即为当前结点测试过程中对应的部分日志，而index也为结点中的clickedIndex属性，在初始化中已经获取。通过截取字符串，并获得包括当前"current index ="行和不包括下个"current index ="行的内容，作为本结点的日志文件，显示在id为LogBox的div中。同时设置了ifgetLog参数，确保截取和内容添加到LogBox的过程只进行一次。

## **测试用例页面**

1.判断能否构成哈密顿路径

添加ClickedItemnum记为被点击的结点数量，添加clickednode数组保存已经点击过的结点id。

每当在图结构中点击结点，getClickedItemColor函数中首先根据传入参数结点的id判断该结点是否在clickednode中。如果不在则通过push添加到clickednode数组，同时ClickedItemnum加一；如果已经在数组中，则ClickedItemnum减一，同时通过indexOf定位所在位置并通过splice删掉该元素。

值得注意的是，如果点击了一些结点，并全部取消点击之后再点击结点，会导致clickednode的第一项在浏览器的console中显示为empty，如[empty, "187", "191"]，造成其长度和ClickedItemnum不相同。此时会通过shift删除掉为empty的第一项。

在点击分析测试用例按钮时，根据该按钮的可点击情况，此时必定已在图结构中选择两个或两个以上的结点。进入jumptestcase函数，声明两个二维数组outArray和inArray，行数为已点击的结点的数量，列数为某一很大的值。

首先生成关于已点击结点clickednode有关的出度二维数组和入度二维数组。依次获取clickednode中每个结点id，使用Cypher语法查询并返回和该结点的相连的出度结点id，同时为了多次并行查询每个结点时更容易获取自身结点，也返回自身结点id。每次查询返回neo4jDataItmArray内的的多组数据中，都包含自身结点row[0]和出度结点row[1]，分别赋值给selfnodeid和neo4jNodeItm。

通过ifinClickedNode函数判断出度结点neo4jNodeItm是否在在clickednode中，同时通过ifnotselfNode函数判断该结点是否是指向自身结点（参数为出度结点id、当前结点在clickednode中的索引，判断两者是否相等）（为什么当时没直接判断selfnodeid和neo4jNodeItm是否一样？）。只有该结点包含在clickednode中且不是自身结点时，才会保存到outArray中，这样outArray数组中只会包含与已经点击过的结点们有关的数据。在保存时，设定保存的行数为当前结点在clickednode中的位置，即outArray数组的第i行保存的是第i个结点的出度id数据；设定保存的列数newindex为index-less，index指示遍历到了neo4jDataItmArray中的第几组数据；less初始为零，当某组数据没有保存到outArray中后其加一，确保每次保存时的列数从0逐增。

在设置保存到outArray所在的行列数中时，需要注意由于多个结点并行查询，不能直接将查询循环次数的num或者每次返回数据后自增的numberout值作为某次保存的行列值，否则会导致实际取值时与预计值不符（如会出现Cannot set property '1' of undefined等越界错误），如实际取值时num变为循环的总次数，numberout会根据并行运行的实际情况变化。

入度结点inArray与之情况相同，也通过ifinClickedNode判断并去除了数组中不包含在clickednode的结点、通过ifnotselfNode判断并去除了结点连接自身的数据。

接着参考https://blog.csdn.net/enjoying\_science/article/details/48103621编写实际的判断函数（也可参考https://blog.csdn.net/qq\_41587612/article/details/86224564），由于需要符合函数的运行过程，需要通过reIdx函数对结点重新编号。将出入度中的id按照在clickednode中的顺序编号为1，2，3等（不包含0）；同时也要将出入度数组outArray转换成关系数组jdgmap（都jdg函数中的第一个for循环实现）。后续具体判断过程可查看所参考的函数。

如果可以构成路径则会通过sessionStorage保存nodelist（由clickednode构成的有序结点id，每相邻两项都可从左至右连通）并跳转到分析测试用例页面，否则会提示选择的结点不能生成测试用例。

因代码运行至跳转函数前数据库的相关查询语句还未进行完，暂时使用了setTimeout函数延时等待500毫秒后才进行jdg判断函数。每次判断之前还通过arraryset函数清空并初始化vis、jdgmap和jdgpath等数组。

测试过程中，遇到了相同的多个结点在点击顺序不同时产生了不同的判断结果的问题。检查后发现是因为没有及时清空相关的数组，对应参考代码的memset部分。

例如，某次运行时clickednode为["176", "184", "183", "185"]，出度数组的计算生成顺序如下。

outArray[2][0]=184

outArray[1][0]=176

outArray[0][0]=183

outArray[0][1]=185

outArray[3][0]=176

入度数组可以由出度数组推导得出，在此不再给出。关系数组jdgmap的计算生成过程如下。

jdgmap[1][3]=1

jdgmap[1][4]=1

jdgmap[2][1]=1

jdgmap[3][2]=1

jdgmap[4][1]=1

nodelist为 ["183", "184", "176", "185"]。

outArray中的第0行表示clickednode中的第0个结点176的出度结点分别由183和185。对应到jdgmap中的jdgmap[1][3]和jdgmap[1][4]都是连通的。

2.测试用例页面布局

在分析测试步骤页面中，上方包含了导航栏；中间有测试用例图示，以带有数字的圆形按钮和箭头表示，按钮下方包含着属性名字；下方是测试用例信息，以表格的形式表示；页面底部是查看测试记录和缺陷按钮。

测试用例图示的可视化部分包含了nodegraph和nodegraphname两个div；而测试用例信息的显示部分则带有包含id为tableInfo表格的一个id为nodeInfoTable的div。需要具体显示的内容和组件是在加载页面后动态添加的。

3.测试用例页面初始化

进入页面后首先通过sessionStorage获取并保存nodelist所点击过的结点，接着进入setButtons函数进行添加设置测试用例图示的操作。

由于nodelist为字符串，通过split(",")方法赋值到数组nodeList中。进行结点数量次数的循环，通过document.createElement("input")方法并指定type为button创建按钮（参考https://www.cnblogs.com/szy13037-5/articles/9780782.html ），按钮上显示的文字value为数组nodeList第当前循环次数项的值，即结点的id；并为按钮设置带有序号的id，style.cssText中设置border-radius为50%从而显示为圆形，每次循环时设定按钮的位置都会比上一次靠右100px；最终添加click事件监听，该函数通过获取按钮本身的value从而跳转到按钮所代表结点的分析测试步骤页面中。最终通过appendChild方法将按钮添加到nodegraph中。

每次添加完一个按钮后，会使用同样的createElement方法接着添加一个指向右侧的箭头，该箭头实际上也是一个透明的无边框按钮，通过设定cssTeext的中left的值确保每次箭头的两侧能紧靠相应的结点代表的圆形按钮。最终也通过appendChild方法将箭头添加到nodegraph中。

循环结束后会进入另外一次循环，查询每个按钮代表的结点的id，返回结点的相关信息。分别保存到结点id数组nodeidlist、简略名字数组nodebtnamelist、结点简单信息和详细信息的数组nodename1list和nodename2list中，其中nodename2list是通过clickedIndex、action等属性拼接的较为详细的结点信息；name用于后续保存显示圆形按钮中的名称。在保存前，获取当前所查询结点的id，接着使用indexOf得到当前结点在传入结点列表nodelist或nodeList数组中的位置作为index索引，从而确定保存在数组中的位置，避免多次并行查询造成保存混乱。

等待查询结束后进入setTableInfo函数进行添加设置测试用例信息的操作。将带有<table>字符的datainfo字符串作为参数传入，函数会进行结点数量次的循环，每次添加rowspan=2的结点id单元格，接着添加nodename1list和nodename2list单元格。最终添加</table>并以设定tableInfo的innerHTML的方式完成单元格的创建（参考https://www.cnblogs.com/jpfss/p/9510970.html）。

创建完成后会调用setBtnBottomName进行添加设置测试用例图示的中圆形按钮下方名称的操作。同样将名称nodebtnamelist中的信息以createElement方式添加到透明无边框按钮中，最终添加到nodegraph下方id为nodegraphname的div中，并通过设置cssText确保名称和按钮能够对齐。另外，获取通过上述步骤设置的带有序号的圆形按钮id，将名称信息同样添加到按钮到title中，使鼠标悬浮于按钮上方时可以看到相关信息提示。

相关的问题：

其中设置按钮的style时遇到了Assigned expression type string is not assignable to type CSSStyleDeclaration问题，查询https://stackoverflow.com/questions/51030396/cost-benefits-of-using-different-ways-to-style-html-elements-with-javascript将设置o.style改成了设置o.style.cssText。

设置表格的单元格合并效果时，参考了https://www.cnblogs.com/qlqwjy/p/9048317.html和https://blog.csdn.net/qq\_26927285/article/details/79916354内容，最终改为使用rowspan=2直接合并。还可以使用<td colspan=2>。

表格的宽度会随文字长度变化的问题，最终为表格添加word-break:break-all;word-wrap:break-word属性和id为nodeInfoTable的父级div。

## **测试用例数量页面**

1.测试用例数量页面布局

在测试用例数量页面中，上方包含了导航栏；中间有显示测试用例的具体数量的提示和手动刷新结果的按钮；下方有每条测试用例的简单图示和对应的分析测试用例按钮。

2.测试用例数量页面初始化

设定可以查询到的测试用例数量testcasenum为20，初始化行数为testcasenum、列数为20的二维数组TCnum\_nodelist和TCnum\_nodenamelist，其中的每行代表一条测试用例中的多个结点id和结点名称，默认为-1。

载入页面后，首先通过sessionStorage.getItem获取本地存储的TCnum\_nodelist和TCnum\_nodenamelist字符串。如果字符串都为空，说明之前没有保存过，则通过调用submitQuery(1)函数在数据库中进行相应的语句查询并返回结果。

在submitQuery中，设定语句查询返回的类型resultDataContents为row（若为graph类型则可能返回空结果）。通过计数为k的forEach语句获取语句查询结果data.results[0].data中testcaenum个数的信息，再通过两个单独计数为j的forEach循环获得其内部每项的meta[0]和row[0]中的内容，其中meta[0]中是有序的测试用例的结点和关系的id，row[0]中是结点和关系对应的属性信息。在此只保存节点信息，故循环次数若为不偶数时则跳过该次循环，否则保存到二维数组的[k][j]位置中，k代表第k个测试用例；j代表该测试用例中的第j个结点的id或者结点名称。

每进行完一次外部的forEach循环前，即每保存完一个测试用例的信息、或每保存完毕二维数组TCnum\_nodelist和TCnum\_nodenamelist的一行信息后，调用setButtons和setBtnBottomName函数，传递循环次数（或二维数组行数）k和二维数组的第k行，从而设置每条测试用例的图示。

测试用例的图示与测试用例页面中的图示相同，以多个带有结点id的圆形按钮和相连箭头代表一条测试用例，每个圆形按钮下方显示测试用例的名称。

在setButtons函数中，创建div2并设置innerText为传入参数k+1的值作为测试用例数量的序号。接着创建div，并通过div2.appendChild(div)将其添加到其末尾中。接着进行i次循环，直到传入的数组中第i列的值为-1时停止。循环中，设定圆形按钮和箭头的过程大致与测试步骤页面初始化中的过程相同。在nodeList的最后一次循环中，不再添加设置指向右侧的箭头；而是设置分析测试用例的按钮，其按钮id中包含传入参数k作为序号，同时添加了showtestcase的click的监听。设置完毕后将div2添加到id为nodegraph的div中。

在setBtnBottomName函数中，首先创建了div，再将nodenamelist当前行中某列的值为-1结束的循环中创建的按钮op添加到div中，最终将div也添加到id为nodegraph的div中。

对于本页面中分析测试用例按钮的showtestcase函数，通过获取按钮id中的序号并赋值给rownum，得到TCnum\_nodelist中第rownum行的数据。接着将该行数据中不为-1的数据通过push保存newnodelist中，最终使用sessionStorage.setItem方法将newnodelist保存到nodelist字段内，供测试用例分析页面读取并跳转到测试用例分析页面。

当submitQuery查询结束后延时等待500毫秒后，会调用数据保存的函数dataSave。其函数内会更新试用例的具体数量的提示，同时通过sessionStorage.setItem保存TCnum\_nodelist和TCnum\_nodenamelist信息到对应字段内。

载入页面后通过sessionStorage.getItem获取本地存储的TCnum\_nodelist和TCnum\_nodenamelist字符串。如果字符串不为空，说明之前已经保存过相关的信息。通过split(",")方法将字符串转换成一维数组，再通过一维数组长度次数的循环，将数组内的值赋值到TCnum\_nodelist和TCnum\_nodenamelist 二维数组的[parseInt(i/20)][parseInt(i%20)]位置中，20代表二维数组的最大列数；parseInt(i/20)向下取整代表所保存的行数；i%20取余为具体列数。保存完毕后，进行testcasenum次循环，每次循环同样调用setButtons和setBtnBottomName函数，传递循环次数（或二维数组行数）k和二维数组的第k行，从而设置每条测试用例的图示。循环完毕后会即刻更新试用例的具体数量的提示，而不再调用dataSave函数。

当图结构有变动时或者需要刷新时，可以点击手动刷新按钮。在reFresh函数中，会通过sessionStorage.removeItem清空所保存过的TCnum\_nodelist和TCnum\_nodenamelist字段。页面初始化时获取不到TCnum\_nodelist和TCnum\_nodenamelist信息后会重新进行数据库的查询。

说明：

某次系统运行。testcasenum为2，sessionStorage.getItem('TCnum\_nodelist')为字符串"5,6,7,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,5,6,7,8,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,"，TCnum\_nodelist\_cache调用完spllit后为一维数组保存其中每项信息；TCnum\_nodelist中第一行保存了前十条信息；第二行保存了后十条信息。页面显示两个测试用例，第一个为5→6→7；第二个为5→6→7→8。

3. 本页面submitQuery使用的查询语句与分析

尝试过的查询语句：

MATCH (n:PageWithES) WITH collect(n) as nodes UNWIND nodes as n

UNWIND nodes as m WITH \* WHERE id(n) < id(m) / MATCH path = allShortestPaths( (n)-[\*..4]-(m) ) RETURN path RETURN p ORDER BY length(p) limit 20

为了计算所有的用例，需要生成从A结点到B结点的路径，且路径应避免环路和重复的结点。参考https://blog.csdn.net/wry2008wry/article/details/80899860编写相关查询语句。

通过查询其他类型的结点测试语句可用性MATCH p=(a:测试结点)-[\*3..5]->(b:测试结点) RETURN p ORDER BY length(p) limit 20 修改为避免重复结点的MATCH p=(a:测试结点)-[\*3..5]->(b:测试结点) Where a.name='8' and b.name='1' and size(apoc.coll.toSet(NODES(p))) = LENGTH(p) + 1

适用于本系统的查询语句：

写法1，指定两个结点：MATCH p=(a:PageWithES)-[\*3..5]->(b:PageWithES) Where a.ElementStoreName='gov.hgm.hgmeap.MineInfoActivity.tag=ImageView.depth=10.id=scienceTitleBarIvStart' and b.ElementStoreName='gov.hgm.hgmeap.MineInfoActivity.tag=Back.depth=2.id=Back.name=Back' and size(apoc.coll.toSet(NODES(p))) = LENGTH(p) + 1 RETURN p ORDER BY length(p) limit 20

将其中的路径长度范围[\*3..5]改为2-10 Started streaming 6 records after 1272 ms and completed after 1279 ms / 改为2-9 Started streaming 6 records after 425 ms and completed after 430 ms. / 改为2-8 Started streaming 6 records after 127 ms and completed after 132 ms.

写法2，指定两个结点：MATCH p=(a:PageWithES)-[\*3..5]->(b:PageWithES) Where a.clickedIndex=1 and b.ElementStoreName='gov.hgm.hgmeap.MineInfoActivity.tag=Back.depth=2.id=Back.name=Back' and size(apoc.coll.toSet(NODES(p))) = LENGTH(p) + 1 RETURN p ORDER BY length(p) limit 20

写法3，指定一个结点a：MATCH p=(a:PageWithES)-[\*2..8]->(b:PageWithES) Where a.clickedIndex=1 and size(apoc.coll.toSet(NODES(p))) = LENGTH(p) + 1 RETURN p ORDER BY length(p) limit 20 查询结果Started streaming 20 records after 1 ms and completed after 893 ms. / 删掉limit限制后 Started streaming 53 records after 1 ms and completed after 3790 ms.

写法4，将写法3中的函数改为Cypher实现：MATCH p=(a:PageWithES)-[\*2..8]->(b:PageWithES) Where a.clickedIndex=1 UNWIND NODES(p) AS n WITH p, SIZE(COLLECT(DISTINCT n)) AS testLength WHERE testLength = LENGTH(p)+1 RETURN p limit 20 查询结果Started streaming 20 records after 5 ms and completed after 905 ms./ 删掉limit限制后 Started streaming 53 records after 12 ms and completed after 3723 ms

写法4添加order排序：MATCH p=(a:PageWithES)-[\*2..8]->(b:PageWithES) Where a.clickedIndex=1 UNWIND NODES(p) AS n WITH p, SIZE(COLLECT(DISTINCT n)) AS testLength WHERE testLength = LENGTH(p)+1 RETURN p ORDER BY length(p) limit 20

写法4的简化： MATCH p=(:PageWithES{clickedIndex:1})-[\*2..8]->(b:PageWithES) UNWIND NODES(p) AS n WITH p,b, SIZE(COLLECT(DISTINCT n)) AS testLength WHERE testLength = LENGTH(p)+1 RETURN b.ElementStoreName ORDER BY length(p) limit 20

写法3的简化：MATCH p=(:PageWithES{clickedIndex:1})-[\*2..8]->(b:PageWithES) Where size(apoc.coll.toSet(NODES(p))) = LENGTH(p) + 1 RETURN b.ElementStoreName ORDER BY length(p) limit 20

最终使用的查询语句：

MATCH p=(:PageWithES{clickedIndex:' + clickedIndex + '})-[\*2..8]->(b:PageWithES) UNWIND NODES(p) AS n WITH p,b, SIZE(COLLECT(DISTINCT n)) AS testLength WHERE testLength = LENGTH(p)+1 RETURN p ORDER BY length(p) limit '+testcasenum;。clickedIndex默认为1代表图结构中的起始结点，testcasenum默认为20避免查询时长过长。

# 例会记录

## **2020.3.10**

创新点（是不是创新、在哪创新、怎么证明（已有论文没有））、研究的价值，答辩也注重创新点。

ppt的内容：做的事情、结果、问题、下一步的事情。

企业合作的同学：实验的日志记录、界面截图要保存好。等完成论文的需要的时候方便获取。跟公司说一下。

周竞帆：绘制系统体系架构图，把功能放到体系架构图中。从层次和功能上都能展示做的部分。 包括自己。工作汇报开篇就要有。有些ppt页面可以每次都放上去。

毕设：可以将具体的问题简化，成为数学问题。转换成图的术语：例如唯一标示的note等。一定要重视强调的学术研究方向的力度。

可视化的问题中的权重，可考虑为：来源、出度入度、出现的次序、访问的次数。

可视化工具的选择。

可视化时 用户浏览时点击的次数、过程、脑力，是否需要考虑 。

## **2020.4.27**

需要做的：读论文（sci和ei、电子学报、计算机学报、软件学报等一级学报和ccf计算机学会指定的的）、学术期刊网等。

论文进度：11月基本做些什么东西，要差不多完成。剩下12月和1月写小论文、2月起写大论文。

推荐的研究方向：

1、图数据库的可视化研究。写出一套新的展示库，从库的层级角度，体系架构考虑。

2、怎么遍历的问题，可以考虑关联关系，使用尽量少的测试用例来覆盖。可参考一篇论文：基于图数据库生成测试用例。

## **2020.5.22**

文本框结点-输入合法性。

单元测试-白盒测试，函数调用的代码测试。

测试情况的分类是否合理。

告诉用户测试的目的，改成多结点测试。

结点选择后，相邻的结点换一种颜色

论文：方法没实现到的 深度学习的思路

雷老师：通信协议测试，机器学习的方法研究工业上还不是很多。遗传算法、gan网络。不做模糊措施，会有错误的测试用例。gan从已有的生成。测试用例有的说是覆盖路径，有的是发现问题，有不同的标准。工业上运用的还是很初步。

专心考虑理论点和创新点，给出想法思路。工作计划把创新点的部分写进去。提前列出论文和大致思路。

可视化的应用场景，专用。信息的展示方式：图形gui界面、怎么让用户操作更符合直觉等、呈现信息的方式：vr ar融合进来、性能方面的算法加快读取响应速度。

测试用例没过：反向查找功能，好像和可视化没关联了。可以是一个方向。（功能和测试用例两个系统的结点链接）

测试可视化的移动领域的专门的特点、对可视化的要求。

综述：可视化评判方案，测试性的东西的可视化

系统的改进方便查看是否有研究的点。

# 文章自动编号

参考Common\_Template - 2.0.dotm文件， 本文实现了图和表格（如表 X）的自动编号以及通过域进行更新功能。同时修改了主题的默认字体为Times New Roman和宋体。

快速更新域：全选文档后按。F9。

文中部分粗体内容代表还需继续研究学习。

# 背单词，表格示例

表 X

单词及其解释

|  |  |  |
| --- | --- | --- |
| 单词 | 音标 | 解释 |
| circus | [ ˈsɜ:kəs ] | n.马戏团，马戏，竞技场 |
| tow | [ təʊ ] | n.拖，拉，牵引 |
| hazard | [ ˈhæzəd ] | n.危险，危害；vt.冒...的危险，使遭危险 |
| peer | [ pɪə(r) ] | n.同辈，同等地位的人，伙伴，贵族；vi.仔细看，费力地看 |
| dismay | [ dɪsˈmeɪ ] | n.灰心，沮丧，惊愕；vt.使沮丧，使惊愕 |
| notable | [ ˈnəʊtəbl ] | adj.值得注意的，著名的；n.名人，要人 |
| endeavor | [ ɪn'devə ] | vt.／n.努力，尽力，尝试 |
| empirical | [ ɪmˈpɪrɪkl ] | adj.以经验（或观察）为依据的，经验（主义）的 |
| deem | [ di:m ] | vt.认为，相信；vi.想 |
| duplicate | [ ˈdju:plɪkeɪt ] | n.复制品；v.复写，使加倍；adj.复制的，二倍的 |
| pigeon | [ ˈpɪdʒɪn ] | n.鸽子 |
| mechanic | [ məˈkænɪk ] | n.技工，机(械)工，机修工 |
| veteran | [ ˈvetərən ] | n.老兵；经验丰富的人，老手 |
| parachute | [ ˈpærəʃu:t ] | n.降落伞；v.用降落伞投送/降落 |
| exploit | [ ɪkˈsplɔɪt ] | n.英雄行为，辉煌功绩；vt.开拓，开采，剥削 |
| arbitrary | [ ˈɑ:bɪtrəri ] | adj.任意的，任性的，主观的；专断的，武断的 |
| revelation | [ ˌrevəˈleɪʃn ] | n.泄露，揭示，展示，惊人的新发现 |
| terrace | [ ˈterəs ] | n.平台，阳台，梯田；vt.使成梯田，给...建阳台 |
| desirable | [ dɪˈzaɪərəbl ] | adj.值得拥有的；可取的，有利的 |
| utmost | [ ˈʌtməʊst ] | adj.极度的，最大的；n.极限，最大可能 |

Appendix

# 空闲

## A conclusion section is not required. Although a conclusion may review the main points of the paper, do not replicate the abstract as the conclusion. A conclusion might elaborate on the importance of the work or suggest applications and extensions.

Appendix

Appendixes, if needed, appear before the acknowledgment.

Acknowledgment

The preferred spelling of the word “acknowledgment” in American English is without an “e” after the “g.” Use the singular heading even if you have many acknowledgments. Avoid expressions such as “One of us (S.B.A.) would like to thank ... .” Instead, write “F. A. Author thanks ... .” In most cases, sponsor and financial support acknowledgments are placed in the unnumbered footnote on the first page, not here.

References and Footnotes

1. https://stackoverflow.com/questions/55561052/syntaxerror-with-is-required-between-create-and-match
2. https://blog.csdn.net/k7Jz78GeJJ/article/details/80002741
3. https://blog.csdn.net/dukai392/article/details/79365376
4. https://www.secrss.com/articles/12186
5. https://mp.weixin.qq.com/s/-PfhYgZwnRrMxhXSRg\_7UA
6. https://mp.weixin.qq.com/s/vLyeYce2xXgAnxR09J9RhA
7. https://www.secrss.com/articles/12186
8. https://img.chainnews.com/material/images/f3d6d4a1c035797def0a827f42520218.jpg
9. https://img.chainnews.com/material/images/ac0eb1d7fb7a22b23a86ead06f95f219.jpg
10. https://mp.weixin.qq.com/s/7PZ6L1YzPZdiMNK25sUqjg
11. https://www.chainnews.com/zh-hant/articles/318452873849.htm
12. https://mp.weixin.qq.com/s/sOVQGqNfhmSdUD0fLypWqw
13. https://blog.csdn.net/m0\_37565948/article/details/81010419
14. https://github.com/neo4j-contrib/neo4j-graph-algorithms
15. https://neo4j.com/blog/graph-algorithms-neo4j-label-propagation/
16. https://blog.csdn.net/qq\_38737992/article/details/89036406
17. https://www.jianshu.com/p/71db5b7354e8
18. https://neo4j.com/docs/graph-algorithms/current/algorithms/label-propagation/
19. https://neo4j.com/graph-algorithms-book/
20. AppCrawler自动化遍历使用详解(版本2.1.0 ), <https://www.cnblogs.com/alwayswyy/p/7576964.html>
21. Appcrawler配置文件的说明可参考, <https://blog.csdn.net/zhanglu_400/article/details/100586698>
22. Appcrawler 参数实验经验, https://testerhome.com/topics/10574
23. https://github.com/intel/haxm/releases/
24. https://www.cnblogs.com/songzhenhua/p/9312804.html
25. YAML、YML在线格式化校验工具, http://www.bejson.com/validators/yaml/