# 第五章 系统测试与评估

## 5.1 系统测试

根据第三章功能需求与非功能性需求分析，系统需要保证可用性和并发性。因此，本节将对系统的基础功能进行单元测试、集成测试和功能测试，对边界输入情况进行边界测试，最后会对系统的查询和提交接口进行性能测试。

### 5.1.1 测试环境准备

本部分将根据设计章节所描述的系统物理部署视图，介绍系统测试环境。如表 5.1 所示，系统测试环境的准备包含运行 chrome 浏览器的用户计算机，运行 Angular2 程序的前端服务器，运行 SpringBoot Docker 以及 Redis Docker 的后端服务器，运行 MongoDB 应用程序的数据库服务器。

表 5.1 测试环境准备

| 设备名称 | 运行程序 | 程序版本信息 |
|---|---|---|
| 用户计算机(Mac OS 10.14) | Chrome 浏览器 | Chrome 72.0(64 位) |
| ECS 服务器(Ubuntu 18.04) | Angular 2 | Angular 2.4 |
| ECS 服务器<br>(4G 内存，50M 带宽) | SpringBoot Docker<br>Redis Docker | Docker 17.09<br>SpringBoot 2.0.2<br>Redis 4.0.7 |
| ECS 服务器(Debian 8) | MongoDB | MongoDB 3.2.16 |

### 5.1.2 单元测试

单元测试是指对软件中最小可测单元进行的功能检查和验证。在本系统中，单元测试主要使用 Junit 针对 Java 中的重要接口类进行。

如表 5.2 所示，该单元测试针对系统中根据 Bug 所处页面进行相似报告推荐的函数进行。利用 Junit 创建 MockHttpServletRequest 对象，发送 Post 请求进行测试，测试结果符合预期。

表 5.2 相似报告推荐页面选择接口测试

| 用例编号 | TC-SR-01 |
|---|---|
| 测试目标 | 验证根据 Bug 所处页面进行相似报告推荐接口的有效性 |

| 接口函数 | RecommendController 中的 RecByPage 函数 |
|---|---|
| 前置条件 | 系统中在”搜索”页面下存在报告 |
| 输入 | 发送 Post 请求，在 data 中传入{page:”搜索”} |
| 预期输出 | 在 Response 的 Body 中获得 JSONArray 类型的推荐列表 |

如表 5.3 所示，该单元测试针对系统中根据 Bug 描述进行相似报告推荐的函数进行。利用 Junit 创建 MockHttpServletRequest 对象，发送 Post 请求进行测试，测试结果符合预期。

表 5.3 相似报告推荐描述接口测试

| 用例编号 | TC-SR-02 |
|---|---|
| 测试目标 | 验证根据 Bug 描述进行相似报告推荐接口的有效性 |
| 接口函数 | RecommendController 中的 RecByDiscription 函数 |
| 前置条件 | 系统中已存在部分报告，且包含”自动退出”相似描述 |
| 输入 | 发送 Post 请求，在 data 中传入{description:”自动退出”} |
| 预期输出 | 在 Response 的 Body 中获得 JSONArray 类型的推荐列表 |

### 5.1.3 集成测试

集成测试是单元测试的扩展，将多个已完成的单元组合成一个组件，测试其接口，确保单元组合之后能够达到预期目标。

如表 5.4 所示，该集成测试需在 TC-SR-01 与 TC-SR-02 单元测试通过后，进行对于相似报告推荐的集成测试。该接口会根据多个信息输入进行相似报告推荐。测试结果符合预期。

表 5.4 相似报告推荐集成测试

| 用例编号 | TC-SR-03 |
|---|---|
| 测试目标 | 根据 Bug 所处页面和描述，进行相似报告推荐 |
| 接口函数 | RecommendController 中的 RecSimilar 函数 |
| 前置条件 | TC-SR-01 与 TC-SR-02 测试通过 |
| 输入 | 发送 Post 请求，在 data 中传入{description:”自动退出”,Page:”搜索”} |
| 预期输出 | 在 Response 的 Body 中获得 JSONArray 类型的推荐列表 |

### 5.1.4 功能测试

本小节的功能测试将设计章节的系统需求保持一致，从用户视角出发，模拟用户正常使用流程，对报告填写模块、测试页面推荐模块和审核报告推荐模块进行功能测试。

如表 5.5 所示，在进行该测试之前，需要在”搜索”页面下提交部分与本次测试内容不同的报告，以便在用户填写过程中验证相似报告推荐是否有效。该测试用例模拟了用户填写报告时的选择、文字输入以及截图标记上传等过程。在填写过程中，推荐列表不断刷新。但由于相关度较低，因此用户选择直接上传该报告。测试结果与测试目标相符。

表 5.5 报告提交测试用例

| 用例编号 | TC-RS-01 |
| --- | --- |
| 测试目标 | 用户填写并提交所发现的 Bug，并且填写过程中，有相似报告推荐。 |
| 前置条件 | 系统中已在”搜索”页面下有部分报告提交 |
| 测试步骤 | 1. 点击创建 Bug 按钮<br>2. 选择 Bug 所处页面(搜索)<br>3. 选择 Bug(必现、严重、不正常退出)<br>4. 填写 Bug 题目(输入小数点，程序闪退)<br>5. 填写 Bug 详细描述(点击搜索，用键盘输入小数点时，程序崩溃，闪退。)<br>6. 上传 Bug 截图，并进行标注<br>7. 点击提交 |
| 预期结果 | 在用户选择所处页面、Bug 属性，填写 Bug 描述时，右侧推荐列表不断刷新相似报告。系统提示用户提交成功。 |

### 5.1.5 边界测试

本部分将对模块进行边界测试。其目的是保证系统在极端输入或条件下，依旧可以正确做出响应。

报告填写模块的测试用例，如表 5.6 所示。其边界条件主要是用户进行报告填写时可能不提供完整的描述信息。

表 5.6 相似报告推荐边界测试用例

| 用例 ID | 用户操作 | 预期结果 | 实际结果 |
| --- | --- | --- | --- |
| 1 | 用户未选择 Bug 所在页面，直接进行填写描述 | 在所有报告中进行检索，返回推荐列表 | 与预期相符 |
| 2 | 用户描述与当前所提交的报告无匹配结果 | 推荐列表显示当前无相似报告 | 与预期相符 |
| 3 | 当前系统无报告提交 | 显示当前无相似报告 | 与预期相符 |
| 4 | 用户未选择 Bug 属性信息进行提交 | 提示用户报告内容缺失，不予提交 | 与预期相符 |

### 5.1.6 性能测试

性能测试为了保证系统在高并发情况下，对于大部分请求能够在有效时间内做出响应。本部分将使用 Jmeter 对系统的查询接口和报告上传接口进行性能测试。整个 Jmeter 的运行过程如下所示：

1) 在开始运行 Jmeter.bat 之后，首先在 Test Plan 下新建 Tread Group 线程组。线程组中每个线程都可以看成一个虚拟用户，其线程数量在整个执行过程中不会发生改变。
2) 设置线程组中 Number of Treads 为 100，Ramp-Up Period 为 2，LoopCount 为 2。该设置模拟了 100 个用户，在两秒内同时对服务器发送请求，并且会循环请求两次。

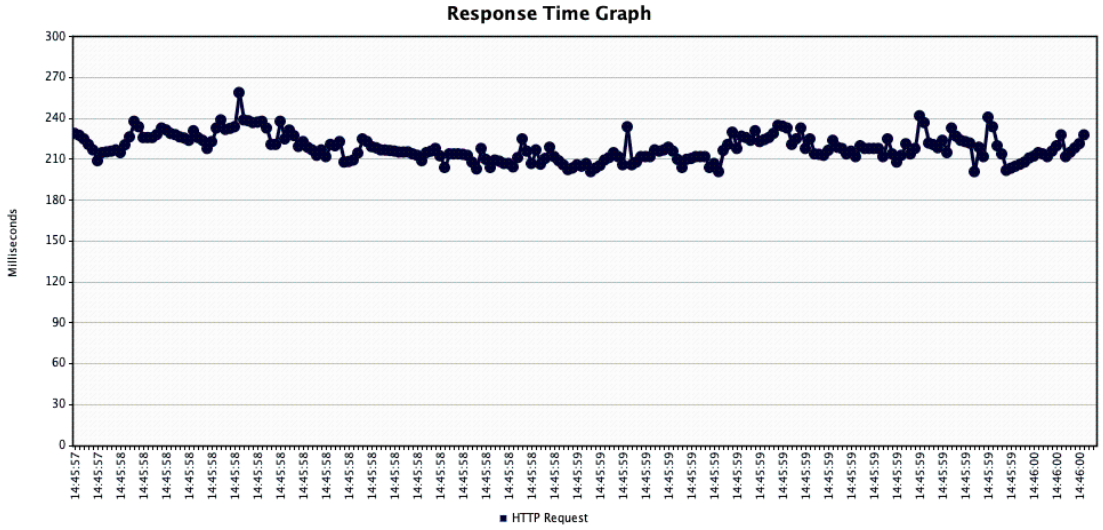


图 5.1 接口响应所需时间图

3) 在线程组中添加三组 HTTP GET 请求，分别对三种推荐算法进行性能测试。另外，创建一组 HTTP POST 请求，对报告提交进行测试。
4) 在线程组中再添加一个用户监听结果的聚合报告。这里以测试页面推荐为例。如图 5.1 所示，以 10ms 为间隔记录了响应时间，从图中可知接口响应所需时间较为平稳。具体测试结果如图 5.2 所示。在四秒内，共进行了 200 次访问请求，平均响应时间为 218ms，响应时间中位数为 217ms，其中 99%的用户请求都在 242ms 内得到了相应，没有错误请求的发生，系统通过该性能测试。

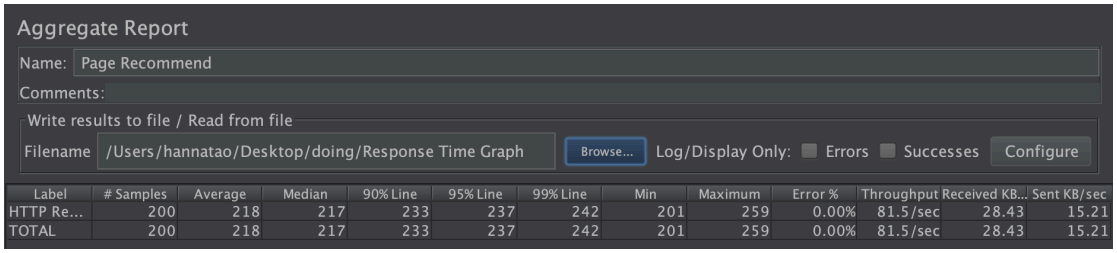


图 5.2 页面推荐性能测试结果

## 5.2 系统评估

系统评估首先将介绍实验的设计与测试目标，然后将就实验结果，从多个角度来进行独立竞争与协作式众包测试推荐方式对比。同时，会就面向协作式众包测试推荐系统得到的结果进行独立分析。

### 5.2.1 A/B 测试

本系统的 A/B 测试将 40 个用户分成两组，其中 B 组采用如图 1.1 所示的独立竞争式的众包测试报告提交系统上传测试报告，A 组采用本系统进行报告提交。两组人员对同一个移动应用 APP 进行功能测试，实验时间为 2 个小时，对比用户所提交测试报告的重复率、报告的质量以及最终报告页面覆盖的情况。报告质量的评价由该移动应用 APP 的开发专家完成。每份报告根据提供截图，描述质量，得分分布在[0,10]的区间内，得分大于 0 分的即被认定为有效 Bug。重复提交的报告不得分。

### 5.2.2 实验结果

如表 5.7 所示，系统在 2 个小时的测试时间内，A 组共收到 201 份报告，覆盖了 142 种不同 Bug。其中由专家判定为有效 Bug 报告的有 182 个，有效比率高达 90.5%。有两组报告判定重复。其中一组原因为两位用户对于该 Bug 描述产生在不同页面上，但实际情况为该 Bug 仅存在某一固定页面，因而导致在进行页面筛选时，不会出现相似报告的推荐。另一组两位用户提交时间相近，因而在提交时没有能够发现重复。测试页面需求共有 24 个，本组报告在测试进行至第 78 分钟时，完成了页面全覆盖。

表 5.7 对比实验结果

| | A 组 | B 组 |
|---|---|---|
| 总提交数 | 201 | 251 |
| Bug 覆盖种类 | 142 | 135 |
| 有效 Bug 数 | 182 | 196 |
| 有效比率 | 90.5% | 78.2% |
| 重复报告数 | 4 | 14 |
| 页面全覆盖所需时间 | 78 分钟 | 未完成 |

B 组虽提交数目达到 251 份，比 A 组更多，但其仅覆盖了 135 种不同 Bug。专家认定的有效 Bug 数为 196，有效比率为 78.2%。其中重复报告有 14 组，这 14 组中甚至有 2 组为同一个人提交了两条完全一致的 Bug。直至测试结束，该组仍未完成测试页面的全覆盖，覆盖页面达到 19 个。仍有 5 个页面由于长尾效应的存在为能完成覆盖。

系统协作效果显著，以如图 5.3 所示的一棵 Bug 树的生成过程为例，展示了用户之间互相协作，对于一个 Bug 的描述层层深入的过程。其中，白色圆圈内的数字表示 Bug 序号，黑色圆圈数字表示其提交顺序。直线起点表示原始报告，指向节点表示由此报告 Fork 之后修改形成的报告。

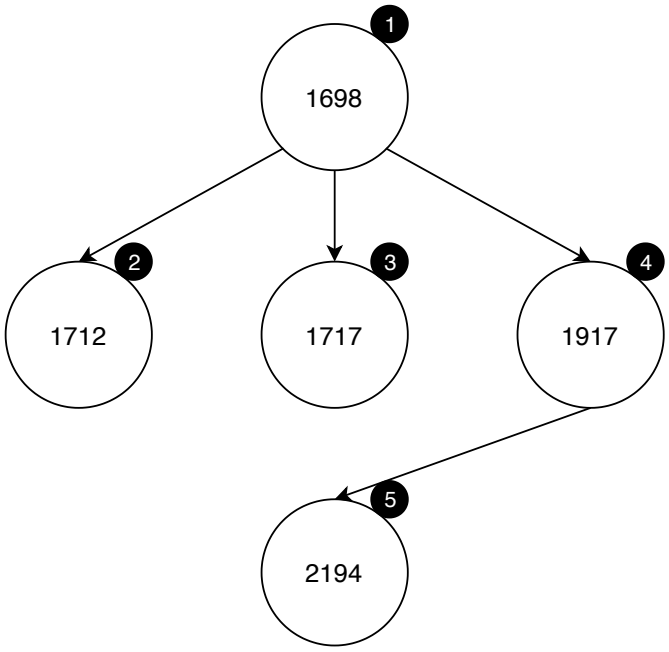


图 5.3 一棵 Bug 树的生成过程

如表 5.8 所示，这 5 份报告由 4 位用户提交。4 位用户通过本系统的相似报告推荐，利用 Fork 操作，对查看更多目的地时 app 出现的问题不断进行深入描述，从而提升了 Bug 的描述质量。

表 5.8　Bug 树相关报告列表

| 报告 ID | 创建时间 | Bug 描述 | 增益 |
|---|---|---|---|
| 1698 | 8:05 | 点击查看更多目的地时 app 停止运行 | 根节点，发现 Bug |
| 1712 | 8:07 | 1. 点击查看更多<br>2. 进行查看更多页面，app 停止运行 | 描述步骤化 |
| 1717 | 8:09 | 当启动 app 第一次点击查看更多时，app 会出现崩溃，重启错误不再发生 | 只有第一次进入时才会崩溃 |
| 1917 | 9:03 | 1. 推荐目的地查看更多时，可能造成 app 停止运行或崩溃<br>2. 推荐目的地查看更多时，可能造成 app 需要加载时间变长 | 第一次提出页面加载时间变长 |
| 2194 | 9:46 | 打开应用进入搜索页面，点击推荐目的地查看更多，一定出现以下情况中的其中一种：<br>1. app 崩溃，停止运行<br>2. 页面卡顿，长时间空白后才进入正确的查看更多页面 | 总结上述情况，描述更加完整 |

报告被点赞数与专家最终评分结果如图 5.4 所示，其中去除了无效报告。每个点表示一份或多份报告的重合结果，曲线表示被点赞数与专家评分的拟合结果。横坐标为被点赞数，纵坐标为专家评分。从结果来看，仍有少部分报告未得到审核。随着点赞数越多，其得分的下限越高。报告的专家评分随着被点赞数的增多，呈现上升趋势。

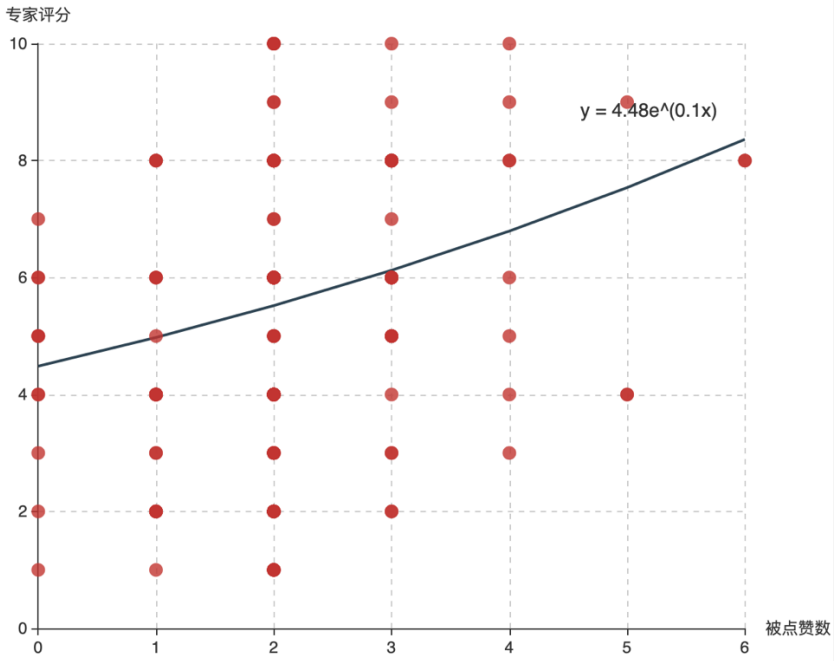


图 5.4　报告被点赞数与专家评分关系图

综上所述，本系统通过相似报告推荐与任务分配，在以下三个方面相较于传

统独立竞争式众包测试系统取得了更好的结果：

1) 减少重复报告：利用实时相似报告推荐，在报告填写过程中，就有效阻止了重复报告的提交。
2) 提升报告质量：利用多人协作，不断完善对于同一个 Bug 的描述，提升报告质量。
3) 加快页面覆盖：测试页面推荐，有效可视化了测试需求，同时，根据用户的历史提交记录，使得用户以最小的代价，进行未知页面的测试。

同时，用户间相互进行审核的点赞、点踩结果也能够更好地辅助专家对报告进行评分。根据 3.3.5 小节设计的用户行为记录，其中，相似报告推荐的召回率为 85.7%，测试页面推荐的召回率为 78.4%，审核报告推荐的召回率为 62.3%。推荐系统的召回率也基本达到预期。

## 5.3 本章小结

本章根据需求，对系统的基础功能进行了测试，同时，设计了边界输入用例对系统进行了边界测试，然后利用 Jmeter 对系统进行了并发性能测试。并使用 A/B 测试，在真实的实验条件下，将本系统与独立竞争式的众包测试系统进行了 A/B 测试，验证了本系统的有效性。