# 一种基于 K-Mean 算法的移动应用兼容性测试方法

张涛, 周文强, 李坤, 王海鹏, 成静

(西北工业大学 软件与微电子学院, 陕西 西安 710072)

**摘 要**:随着移动应用市场的快速发展,移动应用兼容性测试问题日显突出和紧迫。本文提出了一种基于 K-Mena 算法的移动应用兼容性测试设备选择方法。该方法首先建立移动应用兼容性测试设备的特征树模型,确定各个基本特征的测试值,定义初始 $K$ 值和中心点。然后给出一种基于设备特征树模型的特征距离计算方法。最后基于 K-Mean 聚类算法,对移动设备进行聚类,从各个聚类中选择适合的移动应用兼容性测试设备。通过实例进行验证分析,结果表明该方法能够帮助测试人员选择适合的测试设备,从而降低测试成本,提高测试效率和测试质量。

**关 键 词**:移动应用;兼容性测试;K-Mean 聚类算法;特征树模型

**中图分类号**:TP311 **文献标志码**:A **文章编号**:1000-2758(2015)03-0512-04

近年来,移动应用取得了快速发展,已广泛应用于新闻、社交、旅游、医疗、金融、商业等各个领域[1-2]。2015 年移动应用市场规模将高达 25 Billion 美元,而到 2017 年,将有 160 Billion 的移动应用被用户下载安装。由于不同手机或者平板电脑的操作系统、硬件特征、驱动程序等的差异性,使得移动应用在某些设备上运行良好,而在另外一些特定设备不能正常运行[3-4]。因此,为了保证移动应用的可靠性和质量,必须在移动设备上进行充分的移动应用兼容性测试。目前,仅中国市场就有 13 000 多种智能手机,考虑到测试费用和时间的限制,难以对所有移动设备进行兼容性测试。于是,如何系统地为移动应用选择合适的兼容性测试设备,既能够保证测试充分性和质量,又能够控制测试成本和测试时间,就成为当前移动应用兼容性测试迫切需要解决难题之一。为此,本文对基于聚类算法的移动设备兼容性测试方法进行了探讨。

## 1 基于 K-Mean 算法的兼容性测试方法

K-Mean 算法是一种常见的聚类算法,其基本思想是:对于给定的样本集合 $n$,首先随机选取 $k$ 个对象作为初始聚类中心;然后将剩余对象根据其与各个初始聚类中心的距离,将其分配到最近的类。最后重新计算每个聚类的中心,再反复迭代计算分组其余对象,直到目标函数最小化,每个聚类均达到较优的效果。在移动应用兼容性测试中,首先基于 K-Mean 算法聚类移动设备,再从各个聚类选择移动设备,保证兼容性测试覆盖率,实现高效、充分的移动应用兼容性测试。

### 1.1 移动应用兼容性测试特征树模型

移动应用兼容性测试用于验证移动应用与其运行环境之间的依赖程度,即移动应用能否在不同运行环境下正常运行。根据移动应用功能特点和运行环境需求,建立如图 1 所示的移动应用兼容性测试特征树模型 MFtr,本文将其定义为一个二元变量 $(N,E)$。其中:$N$ 是特征树模型节点,包括叶子节点、中间节点和根节点,每个节点表示一种移动应用特征,如 CPU、屏幕分辨率、网络连接等;$E$ 是一组节点之间的连接,每个连接包括一个父节点和它的一个子节点,描述父子特征之间归属关系。

**收稿日期**:2015-01-08 **基金项目**:国家自然科学基金(61103003)与航天科技支撑计划(2014HTXGO)资助

**作者简介**:张涛(1976—),西北工业大学副教授、博士,主要从事移动云测试、软件安全性技术以嵌入式软件技术研究。

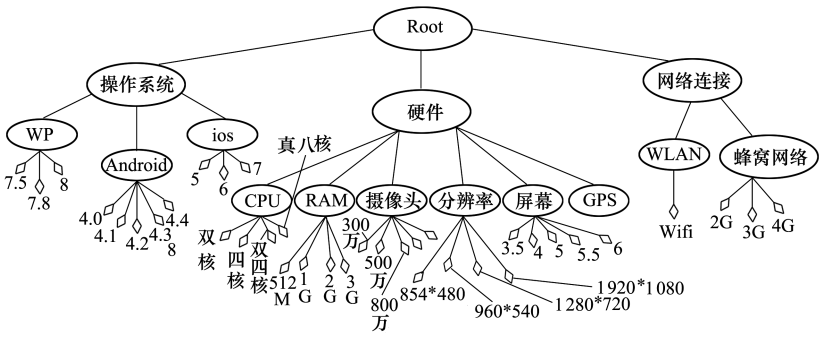



图 1　移动应用兼容性测试特征树模型

### 1.2　初始 K 值及聚类中心点的定义

在移动应用特征树模型中,每个叶子节点表示一种最基本的设备特征,例如分辨率、屏幕大小、内存大小等。首先根据移动应用的特点,市场用户占有率和兼容性测试经验等,选择若干个特征测试数据值,形成该特征的待测数据集 $A_i$。

这样,由所有的叶子节点特征 $A_i$ 的笛卡儿积集,构成了初始的 K-Mean 聚类中心点,包含了各个叶子特征的所有测试组合情况。于是,可将初始中心点集合 $\boldsymbol{B}$ 定义为:

$$\boldsymbol{B} = \boldsymbol{A}_1 \times \boldsymbol{A}_2 \times \cdots \times \boldsymbol{A}_n \tag{1}$$

这样,初始聚类初值 $K$ 则是集合 $\boldsymbol{B}$ 中元素个数,即集合 $\boldsymbol{B}$ 的大小,也就是各个叶子节点待测集大小的乘积,记作:

$$K = |\boldsymbol{B}| = |\boldsymbol{A}_1 \times \boldsymbol{A}_2 \times \cdots \times \boldsymbol{A}_n| = |\boldsymbol{A}_1| \times |\boldsymbol{A}_2| \times \cdots \times |\boldsymbol{A}_n| \tag{2}$$

例如,以屏幕尺寸为例,主要包括 {3.5、4.0、4.5、5.0、5.5、6.0} 等,根据测试资源情况,假定分辨率特征待测数据集 $A_i$ = {3.5、4.5、5.0、6.0},其中个数为 4。分别设定所有叶子节点的待测集合后,便可根据公式(1) 和公式(2),计算获得初始 $K$ 值和中心点。

### 1.3　移动设备特征距离计算

在基于 K-MEAN 算法中,需要计算各个移动设备与中心点之间的特征距离。由于移动设备具有多个特征,且数据类型可能各异。基于特征树模型,这里提出一个递归,分层的计算方法。先定义 2 个设备 $d_x$、$d_y$ 在某个非叶子节点特征 $F$ 上的距离为 $D(d_x,d_y)_F$,可以通过特征 $F$ 的子特征距离,加权和递归计算,其计算公式定义如下:

$$D(d_x,d_y)_F = \sum_{i=1}^{n} \alpha_i D(d_x,d_y)_{S_i} \tag{3}$$

式中,$S_i$ 表示特征 $F$ 的第 $i$ 个子特征,$\alpha_i$ 表示其第 $i$ 个子特征权重因子。对于同一父节点,其所有子节点的权重因子之和为 1,即满足:

$$\alpha_1 + \alpha_2 + \cdots + \alpha_n = 1 \tag{4}$$

这样,2 个设备的特征距离,将最终归结为对其所有叶子节点特征距离的计算。对于叶子节点,应根据其数据值不同,分别计算。对于非数值型特征,如网络类型,其计算公式定义为:

$$D(d_x,d_y)_F = \begin{cases} 0 & d_{xF} = d_{YF} \\ 1 & d_{xF} \neq d_{YF} \end{cases} \tag{5}$$

即如果特征距离值相同为 0,否则为 1。

对于数值型特征,首先基于 Min-Max 方法进行,将其值域限制在[0,1],然后按下列公式计算:

$$D(d_x,d_y)_F = |d_{xF} - d_{yF}| \tag{6}$$

### 1.4　移动测试设备聚类与选择

根据 K-Mean 算法,通过迭代计算各个设备到中心点距离,将其分配到最近的中心点,并将所有设备分配到不同聚类。在最终聚类中,可能有的聚类会没有设备或者只有少量设备,属于无效聚类,应该被去除。然后,根据各个聚类中设备型号数量、总的设备市场份额对其排序。最后根据测试资源限制,优先从设备型号多、市场份额大的聚类中选择市场份额大的移动设备,形成最终的移动应用兼容性测试设备集。

## 2　实例验证与分析

为了验证上述方法的有效性,本文针对“微信”和“携程”2 个移动应用进行了兼容性测试。首先从“中关村在线网站”收集了约 3 000 种近 5 年的 Android 智能手机,作为候选兼容性测试设备。

“微信”是一款即时通信与社交类移动应用,其兼容性测试主要关注 Android 操作系统版本、前置摄像头分辨率、和屏幕大小、分辨率、网络连接等特征。“携程”则是一款机票、酒店、旅游门票订购移动应用,更多关注于屏幕大小、屏幕分辨率、网络连接等特征的兼容性问题。如图 2 和图 3 所示,首先分别对上述的 2 个应用建立兼容性测试特征模型。

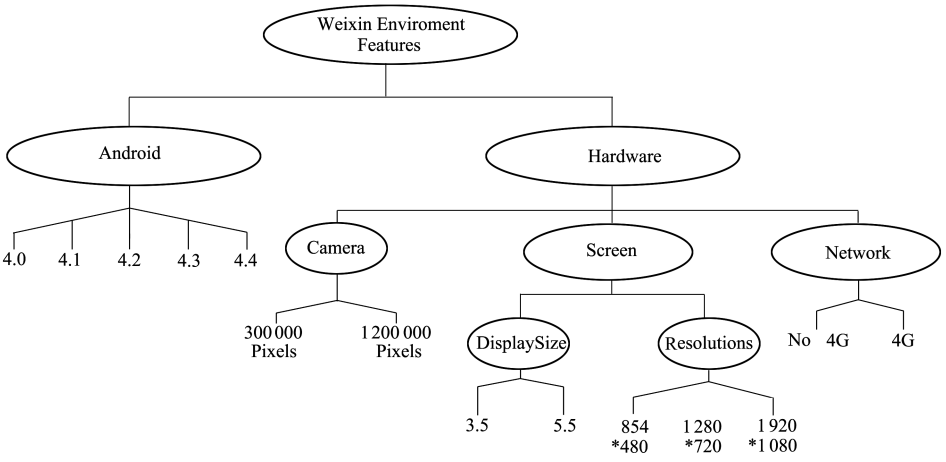



图 2 微信兼容性测试特征树

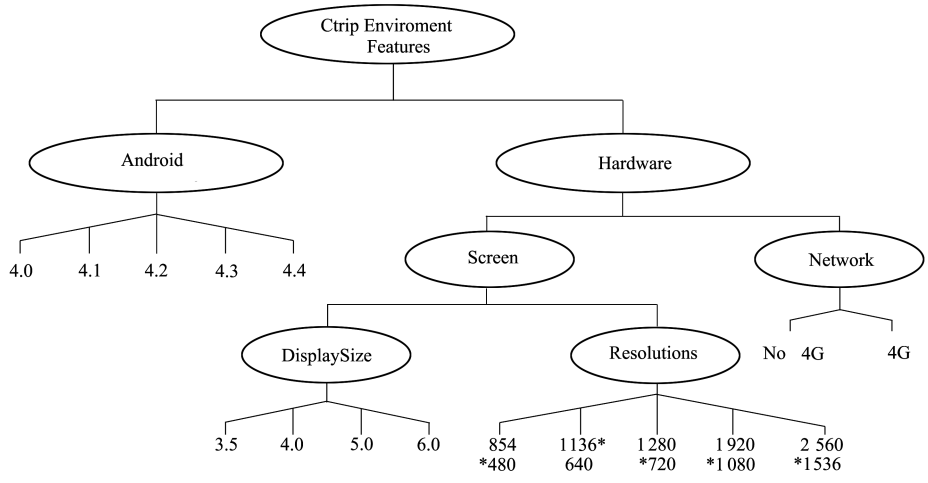



图 3 携程兼容性测试特征树

然后根据本文提出的算法,分别针对“微信”和“携程”应用,对 Android 移动设备进行 K-Mean 聚类,从每个有效聚类中选择 1 个设备,构成其最终兼容性测试设备集。为了进行对比分析,根据测试工程师经验,选择相同数量的移动设备,作为经验测试设备集。分别对 K-Mean 算法选择的测试设备集和经验选择的设备集,执行相同的测试用例集。最后统计 2 种设备集的测试特征覆盖率,并将测试发现的故障区分为“与测试环境无关的非兼容性故障”,及“相关的兼容性故障”,测试结果参见表 1。

**表 1 移动应用兼容性测试结果表**

| 移动应用 | 聚类数量 | 有效聚类数量 | 测试覆盖率 | | 非兼容性故障 | 兼容性故障 | |
|---|---|---|---|---|---|---|---|
| | | | K-Mean 测试设备集 | 经验测试设备集 | | K-Mean 测试设备集 | 经验测试设备集 |
| 微信 | 40 | 31 | 87% | 64% | 1 | 6 | 2 |
| 携程 | 60 | 34 | 89% | 71% | 3 | 8 | 3 |

测试结果表明,基于本文的算法选择的测试设备集,具有更高的设备特征覆盖率,有助于发现更多的兼容性故障,对降低测试成本、减少测试时间、提高测试效率和测试质量,有明细效果。

## 3　结　论

目前,对移动应用兼容性测试研究仍处于初期阶段,缺乏系统有效的方法。本文探讨了基于 K-Mean 的移动应用兼容性设备选择方法。该方法的有效性主要受兼容性测试特征树模型影响。首先需要根据测试资源,移动应用特点、兼容性测试需求等因素,建立适合的兼容性测试特征树模型。兼容性测试特征树模型过大,则测试成本和时间越大;过小,则测试覆盖率较低。

## 参考文献:

[1] Adrian Holzera, Jan Ondrusb. Mobile Application Market: A Developer's Perspective[J]. Telematics and Informatics, 2014, 28(1): 22-31

[2] Kakkar K, SNMC Kakkar M. Risk Analysis in Mobile Application Development[C]//Proceedings of 4th International Conference on the Next Generation Information Technology, Noida, 2013: 26-27

[3] Gao J, Bai X, Tsai W T, Uehara T. Mobile Application Testing: a Tutorial[J]. IEEE Computer, FEBRUARY, 2014: 4-5

[4] Gao Jerry, Tsai Weitek, Paul Ray, Bai Xiaoying, Uehara Tadahiro. Mobile Testing-as-a-Service (MTaaS)——Infrastructures, Issues, Solutions and Needs[C]//Proceeding of 2014 IEEE 15th International Symposium on High-Assurance Systems Engineering (HASE), Miami Beach, FL, 2014: 158-167

# Selecting Mobile Devices for Mobile Compatibility Testing Using K-Mean Algorithm

Zhang Tao, Zhou Wenqiang, Li Kun, Wang Haipeng, Cheng Jing

(Department of Software Engineering, Northwestern Polytechnical University, Xi'an 710072, China)

**Abstract**: With the rapid development of mobile application's market, mobile application compatibility testing has become a most prominent and urgent problem. We propose a systematic method of selecting adaptive mobile devices for mobile application compatibility testing; it is based on K-mean algorithm. Firstly, the compatibility feature tree model of mobile application is constructed, and every leaf node of feature tree is set some testing values by experience. So initial value of K and cluster centers are set. Secondly, a layered and recursive algorithm is described for calculating feature distance between mobile devices. Thirdly, all mobile devices can be clustered automatically with K-mean algorithm, and testing devices can be selected from different clusters according to popularity market share of devices. Finally we test two selected mobile applications with proposed method, proving that the method is effective in selecting mobile devices for compatibility testing of mobile applications.

**Key words**: algorithms, calculations, cameras, clustering algorithms, global positioning system, mathematical models, mobile devices, optical resolving power, pixels, wireless local area networks (WLAN); compatibility testing, K-mean clustering algorithm, mobile application, mobile devices feature tree, test coverage.