doi 10.3969/j.issn.1671-1122.2014.06.006

基于结构比对的软件同源综合检测工具的设计与实现

刘楠，文伟平

（北京大学软件与微电子学院，北京 102600）

摘 要：文章对基于文本、Token 和抽象语法树的同源性检测技术进行探讨，详细介绍了关于抽象语法树的同源性检测技术。同时，在对实际应用大量研究的基础上，文章着重介绍了源代码同源性检测系统的架构设计，以及引擎比对、比对结果分析和比对结果输出等主要功能模块，并对开发的系统进行了系统测试和分析，验证了算法的可行性。

关键词：软件同源性检测；基于文本的；基于 Token 的；基于树的；抽象语法树

中图分类号：TP309 文献标识码：A 文章编号：1671-1122（2014）06-0031-08

Design and Implementation of Software Plagiarism Detection Tool based on Structural Alignment

LIU Nan, WEN Wei-ping

(*School of Software & Microelectronics, Peking University, Beijing* 102600, *China*)

Abstract: This paper discusses the software plagiarism detection technologies which are based on text, token and abstract syntax tree, and especially discusses the technology based on abstract syntax tree (AST) in detail. At the same time, according to the study of software plagiarism detection applications, this paper mainly introduces the architecture of software source code homologous detection system and some kinds of key functional modules. For a demonstration of the feasibility on the proposed algorithm, this paper makes a deep analysis and evaluation of the source code homologous detection system.

Key words: software plagiarism detection; text-based; Token-based; tree-based; abstract syntax tree

0 引言

针对源代码同源性的检测技术在国内外的研究开展得非常广泛，主要应用于软件知识产权的检测、软件开发过程中的软件构件管理，尤其应用于现代软件工程中的克隆检测等几个方面。一些大型公司和企业已经把源代码同源性检测引入到本单位的所有开发部门中，以进行源代码的检测和管理，规避自身知识产权风险。源代码同源性检测的研究重点当前集中在源代码抄袭检测技术，也就是通常所说的代码克隆检测。对于源代码抄袭检测技术，国内外的相关学者都进行了相应的研究，提出了很多算法。

早期的软件同源性检测算法主要集中于基于文本、Token 和语法树的检测技术。不同的算法各有优势，能检测出不同程度的抄袭现象，不过这些算法都或多或少地存在一些不足之处，不能检测出全部的软件源代码抄袭问题。随着同源性检测技术研究的深入，一些新的研究思路和算法逐渐被引入进来。这些新方法由于考虑了多个相关学科的最新研究进展，使得抄袭检测的准确度较早期相比有了较大的改进。国内对同源性检测技术的研究比国外起步稍晚，然而随着研究力量的不断增大，进展很快。本文将对源代码同源性检测技术中的几种主要检测技术进行介绍。

收稿日期：2014-05-13

基金项目：国家自然科学基金 [61170282]

作者简介：刘楠（1979-），男，北京，工程师，硕士研究生，主要研究方向：信息安全；文伟平（1976-），男，湖南，副教授，博士，主要研究方向：网络攻击与防范、恶意代码研究、信息系统逆向工程和可信计算技术等。

1 源代码同源性检测关键技术介绍

1.1 基于文本的同源检测技术

由于原理简单并且编码容易实现，基于文本的同源检测技术在源代码同源性检测领域是最早研究的一种技术。Baker[1] 和 Johnson[2] 提出了基于文本中字符串和字符指纹的文本匹配比对技术。为了提高比对粒度，Ducasse[3] 针对源代码以行为单位计算其哈希值，再通过二维方法将每行的哈希值与其他哈希值进行比较，以得到相同的代码行。

如果对代码进行简单的变换，基于文本比对的源代码同源性检测技术就难以检测出抄袭行为。因此，对于抄袭检测，基于文本的检测方法漏检率较高，但由于其检测技术是基于文本序列和文本组合，因此其误检率非常低。这也是基于文本的同源性检测技术在抄袭检测中具有非常重要地位的主要原因。

1.2 基于Token的同源检测技术

Token 是一种字符串序列。在基于 Token 的同源检测技术中，最常用的就是 LCS（longest common subsequence）算法，但是 LCS 算法具有局限性，它无法检测变量类型名称修改、代码顺序调整这样的抄袭现象。

基于 Token 的源代码同源性比对技术，其算法研究较多，CP-Miner[4]、CCFinder[5] 等众多比对工具都采用了这种比对技术。在基于 Token 的源代码同源性比对技术基础上，Muddu[6] 等学者最新提出了一种 Robust 技术，这种技术基于一种 language aware token 序列，同时又融合了一些抽象语法树技术。它首先遍历由 JDT 解析获得代码的语法树来获取代码的 Token 序列，然后通过 K-gram 技术把得到的 Token 序列分割成一系列的 K-gram 并将它们存储在索引中，最后通过比对 K-gram 来定位源代码抄袭。这种技术在一定程度上弥补了 LCS 算法的缺陷。

王伟 [7] 等在基于 Token 的技术研究方面提出了 Token 流分析和序列挖掘相结合的方法，将克隆代码和由克隆代码引起的缺陷进行联合检测，降低克隆代码误检和漏检对标识符重命名不一致缺陷检测的影响。于世英 [8] 把聚类引入到代码同源性检测之中，提出了一种基于序列聚类的检测算法。该算法首先将源代码按照其自身结构进行分段提取，然后对各个分段进行部分代码变换，再将带权重的编辑距离作为相似度量标准对这些符号进行序列聚类，得到相似的程序代码片段，以达到对源程序进行相似功能检测的目的。

1.3 基于语法结构的同源检测技术

在基于语法结构的同源检测技术中，基于语法树的同源检测方法应用非常广泛。Baxter[9] 较早提出了将代码段的语法结构进行比对分析，从而判定是否是抄袭代码的基于抽象语法树（abstract syntax tree，AST）的代码同源性分析技术。在多种实用性的同源性检测技术中，基于抽象语法树的代码同源性比对技术与其他技术相比具有最高的准确性 [10]。

此外，Khaustov[11] 提出了一种基于 Trie-tree 的 NCP（no crib plus）算法，算法使用源代码的标记化 token 表示，并使用 Trie 树存储令牌和 Levenshtein 距离以计算评估两个序列的相似性。Narayanan[12] 在 2012 年第 7 届 ICCSE 会议上提出了一种基于距离的算法，这种算法通过检查文本的语法属性以确定抄袭情况。

刘丽霞 [13] 提出了基于 Tire 树的 New-Trie-Stack 改进算法，它是对 Trie-DPStack 算法的优化。New-Trie-Stack 算法利用编辑距离阈值的限制来研究字符串的相似性。李虎 [14] 针对 Java 提出了一种适用于源代码和二进制代码抄袭的检测方法，该方法以 Java 类和 class 文件作为基础单元进行比较，抽取了 5 种代表语法及语义特征的向量，通过多重计算得出比对的类文件之间的相似程度，可用于判断存在的多种抄袭现象。

1.4 基于语义的同源检测技术

基于语义的同源检测方法起步较晚，这种技术能够从语义方面判断代码结构，而不受代码形式的影响，具有很高的准确度。目前主流的基于语义的检测算法都是采用程序依赖图（PDG）相似检测方法，它将源代码变换为图形化的 PDG，再基于子图同构的概念对克隆代码进行检测。2012 年，Chan 等 [15] 提出了一种基于堆图（Heap Graph）的抄袭检测方法，它通过获取软件的 birthmark 来判断抄袭。这种算法能够有效检测出针对 JavaScript 程序的抄袭。Cosma[16] 等使用 Latent Semantic Analysis（LSA）方法提高了抄袭检测的准确度。

1.5 基于编码风格的同源检测技术

Arabyarmohamady[17] 提出了一种基于编码风格的抄袭检测方法，这种抄袭检测方法不仅适用于程序设计语言，同时也适用于自然语言。它主要分两个阶段进行：在第一

阶段，提取文件的主要编码风格特征；在第二阶段，用3种不同的模块来检测抄袭的代码以确定抄袭的位置。

1.6 基于数据挖掘的同源检测技术

WANG[18]针对学生程序抄袭现象提出了一种基于数据挖掘的算法。算法使用CloSpan算法挖掘程序中相似的代码，然后计算相似度，最后输出相似列表。在国内，这是数据挖掘首次应用到同源性检测之中。

同源性检测系统的主要目标是结合文本比对、Token比对和语法比对3种比对方式，对软件源代码做出完整、有效的同源性鉴别。本章主要介绍了同源性检测系统的系统架构、设计思路和功能模块。

1.7 基于抽象语法树的同源性检测技术

1.7.1 构建抽象语法树

构建源代码文件的抽象语法树是进行语法比对的基础。抽象语法树是经过语法分析之后生成的一种保存有树节点相关信息的树形存储结构。在构建语法树之前，我们首先需要对源代码文件进行预处理；然后再进行词法和语法分析；最后为了方便节点比对，在执行完语法分析后，我们需要把得到的树形结构的语法树转变为存储结构。

1）预处理

对源代码文件内容进行预处理，最主要是对软件源代码中的文件包含指令、宏定义指令、条件编译指令进行处理。对于源代码文件中的包含指令的处理是直接删除；对于宏定义指令的处理是使用原始字符串替换回被宏定义所替换的字符串；对于条件编译指令的处理是根据条件编译指令中的执行条件选择保留符合执行条件的代码段和删除不满足编译条件的代码段。

预处理阶段主要是为源代码程序进行词法分析和语法分析做准备。对于一个大型工程，每个源代码文件中都包含有很多预处理命令。包含这些预处理命令的源代码在进行词法分析和语法分析时，会大大增加编译程序的处理难度，甚至可能使得词法分析和语法分析处理器因报错而终止运行。对源代码文件中的相关内容进行预处理和替换，才能在生成源代码抽象语法树的扫描和分析中，得到更加准确的分析结果。

2）词法分析

词法分析是一种将输入的源代码文本按照构词规则转化成为一系列单词（Token）的分析工具。Token是词法分析产生的具有独立意义的最小单位。词法分析阶段是编译过程的第一个阶段，是编译的基础。这个阶段的主要任务是从左到右逐个字符地读入源代码，即对构成源代码文件的字符流进行扫描，然后根据构词规则识别单词并返回相应的Token信息。这里的构词规则指的就是匹配输入词法分析器的字符串的正则表达式。

3）语法树存储

经过源代码语法分析后，系统会建立一棵抽象语法树。该语法树为树形结构，如果在比对过程中直接使用该树形结构，将大大降低程序的运行效率，因为对树中节点的查找只能通过遍历语法树来实现，当树节点较多时，遍历查找将会增加系统的时间复杂度。综合后面要讲的比对流程，把树形结构转变为结构体数组类型的结构有利于对节点进行操作。在转变存储结构时，对数组中元素的定义非常重要，下面给出了数组中每个元素的结构定义。

```
StoreNode{
  int beginLine;//节点对应源文件开始位置
  int endLine;//节点对应源文件结束位置
  int id;//节点的类型，语法分析时生成的唯一数值
  long hashValue;//当前节点的赋值
  StoreNode parent;//当前节点的父节点地址
  StoreNode children;//当前节点的子节点地址
  int nodeNum;//节点数目
  int mark;//标记变量，用于标记已经比对节点
}
```

1.7.2 节点比对

节点比对是判断源代码是否同源的主要方法。节点比对包含两个子模块：节点值处理和节点赋值比较。

1）节点值处理

在基于抽象语法树的同源性检测技术中，节点比对需要用到其节点赋值，因此对节点赋值是基于抽象语法树的同源性检测技术的核心。根据节点赋值的定义，一个节点的值为其节点类型的初始值与其孩子节点的值之和。赋值

过程主要包括节点的赋值和节点赋值叠加。

（1）节点赋值

在经过语法分析程序生成的抽象语法树节点中，有一个关于节点类型的字段，定义为节点的 ID。节点的 ID 值根据其在语法正则表达式中定义的位置而唯一确定，即每个确定类型的节点的 ID 都是唯一的。由于定义了节点的 ID 值，在赋值过程中，首先调用随机数生成随机数组，从数组中选取下标值等于节点 ID 的随机数作为该节点的初始值。

（2）节点赋值叠加

经过赋值操作后，我们得到了各个节点的初始赋值，然后通过累加节点的各个孩子节点来计算节点的最终值。首先要找到语法树的叶子节点，叶子节点的最终值一定等于其初始值 ；然后通过叶子节点向上运算得到高层节点的最终赋值。

2）节点赋值比较

节点比对是根据节点的赋值进行比较来判断两个节点是否具有同源性。如果两个节点的赋值完全相同，可以确定两个节点同源，那么根据节点中的信息能够发现对应的源文件内容是相同的。

（1）节点排序

抽象语法树经过赋值处理后，每个节点都有一个唯一的值，但是此时节点并没有一个固定的次序，因此在比较过程中会增加时间复杂度，造成大量的时间浪费。为了更好地提高比对效率，在进行赋值比对之前，首先根据节点的子节点数目和节点的赋值对节点进行排序，在排序中之所以考虑节点的子节点数目是为了让抽象语法树中层次较高的节点尽量排在数组的前面，这样比对时就能够减少比较次数。排序后，我们就能得到一个节点层次由高到低、节点赋值由大到小的节点数组。

（2）节点赋值比较

节点赋值比较是根据节点的值逐个比对源代码抽象语法树的各个节点，由于之前进行了节点排序，所以节点赋值比较会变得更加容易。

1.7.3 相似度计算

由于基于抽象语法树的同源性检测技术是通过比较节点的值来判断源代码文件的同源性质，因此根据语法树的特性，我们可以把相似度定义如下：

$$S = \frac{N_s}{N} \quad \cdots\cdots(1)$$

其中，N_s 代表节点值相同的节点个数，N 代表比对源的代码文件所生成的抽象语法树的节点总数。

2 总体设计

2.1 业务流程设计

本系统采用 B/S 架构，用户首先使用已注册的用户名和密码登录系统，然后才能对系统进行操作。具备比对权限的用户通过浏览器登录比对服务器后，选择比对功能，将目标分析文件提交至服务器，服务器收到任务请求后，选择相应的比对模块进行比对。比对模块从基准样本库中取出样本数据与之进行比对，并反馈比对结果。比对服务器将比对结果页面返回给客户端浏览器。如果用户是管理员权限，则可以对系统中的用户组和样本库进行管理和修改。

系统支持对源代码的比对算法包括文本比对、Token 比对和语法比对。其中文本比对是对源文件的字符序列或行序列进行比对。Token 比对是先将代码源文件中的字符串进行词法分析，解析为 Token 序列，然后再进行比对。语法比对首先对文本形式的代码源文件进行预处理分析，再对预处理分析后的源代码进行词法和语法分析，最终将整个代码文本的所有数据结构模块形成语法树，按照本文设计的算法对语法树进行比较，从而得到语法层面的鉴别结果。

通过 3 种比对方式对软件源代码进行比对之后，系统还设计实现了有效的综合度量算法，将文本、Token 和语法 3 种比对结果进行合理的综合分析，得到一个同源性综合度量结果，以此更加合理地判断软件源代码的同源性。

同源性检测系统的检测处理流程如图 1 所示。

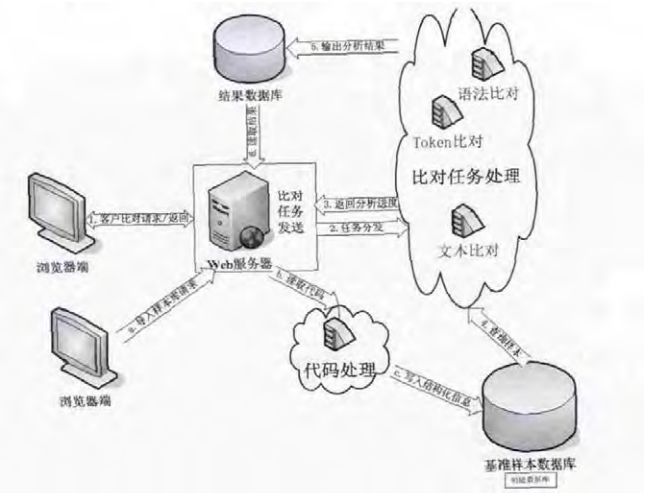

图1 系统检测处理流程

2.2 系统架构设计

本系统采用 B/S 架构，用户通过浏览器登录 Web 服务器后，选择比对功能，将目标分析文件提交至 Web 服务器，Web 服务器收到任务请求后，根据用户选择的比对功能将任务分配给文本、Token、语法比对模块，比对模块从基准样本库取出样本数据与之进行比对。比对任务完成后将分析结果发送给结果数据库，并通知 Web 服务器比对任务完成。Web 服务器将比对结果页面返回给客户端浏览器。

系统同时支持样本库导入功能，用户通过浏览器登录后选择样本库导入功能，将样本库文件提交至 Web 服务器，Web 服务器收到导入样本库请求后，将任务交给样本库导入模块，该模块从 Web 服务器读取代码，并对代码进行处理后，将处理后的代码写入基准样本数据库。

系统从层次上可分为 3 层，分别为表示层、业务逻辑层和数据层（如图 2 所示）。其中表示层通过 struts 组件向业务逻辑层传递表单数据，并控制逻辑操作；业务逻辑层通过 hibernate 组件操作数据库实现数据持久化处理。

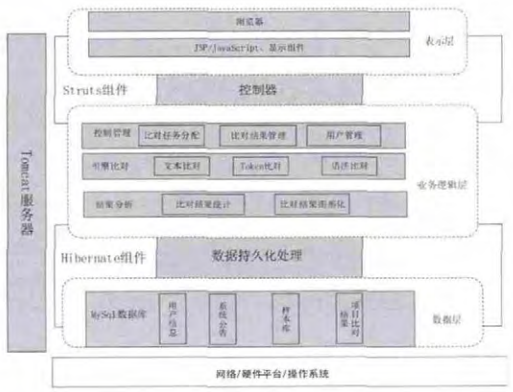

图2 系统分层架构图

总体框架主要包含 3 个层次：

1）表示层

系统表示层包括浏览器、JSP 以及显示组件，主要负责为用户提供操作界面以及向用户展示结果。用户在 JSP 展现的操作界面通过 struts 组件向后台提交数据请求，struts 控制逻辑转发请求，后台业务逻辑处理完请求之后再经 struts 返回给前台展示。

2）业务逻辑层

系统的业务逻辑层实现了系统的主要核心功能，主要由控制管理、引擎比对和结果分析 3 大组件组成。控制管理组件包括比对任务分配、比对结果管理和用户管理；引擎比对组件包括文本、Token、语法 3 种比对算法；结果分析组件包括比对结果统计和比对结果图形化处理。其中各个组件具体设计如下：

（1）比对任务分配模块。根据前台传递过来的不同用户请求将任务分配给文本比对、Token 比对和语法比对模块。

（2）比对结果管理模块。主要负责处理前台对比对结果的各种操作，如查询、添加、删除等。业务逻辑在收到前台传来的请求之后根据请求调用 hibernate 组件对数据层的项目比对结果数据库进行相应操作，然后将相应的结果返回给业务逻辑，再由业务逻辑经 struts 返回前台展示。

（3）用户管理模块。实现对用户的权限划分和登录控制。当用户进行登录、信息修改、用户删除以及注册申请时，都由业务逻辑的用户管理模块调用 hibernate 组件对用户信息数据库进行修改，并根据用户权限确认操作是否可执行。

（4）引擎比对模块。主要通过 Java 实现了文本、Token 和语法结构 3 种比对算法。

（5）比对结果统计模块。利用 AHP 层次分析法将由文本、Token 和语法结构 3 种比对算法得到的相似度计算出一个综合相似度，并将检测项目中文件的相似度按相似度分布进行统计合并，得到检测项目中每个文件与样本文件比较时的最大相似度值。

（6）比对结果图形化处理模块。将统计出的比对结果以柱状图的形式返回给前台显示。

3）数据层

数据层采用了传统的关系型数据库，包括了系统中所有需要持久化存储的数据，如用户信息、样本库、项目比对结果以及系统公告。

2.3 基准库设计

系统的基准数据库核心主要分为 3 个部分，分别是数据获取模块、数据分析模块和数据存储模块，如图 3 所示。

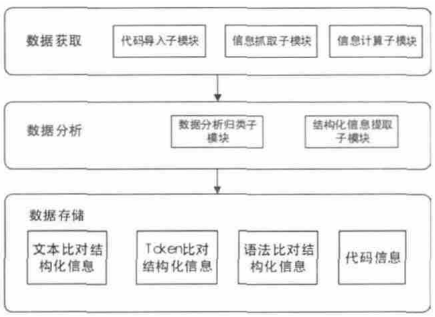

图3 代码同源基准库设计

1）数据获取模块

数据获取模块包括 3 个子模块，分别是代码导入子模块、信息抓取子模块和信息计算子模块。

（1）代码导入子模块

代码导入子模块主要用于自研代码的导入功能，通过对外接口，支持对特定格式的源代码进行导入。

（2）信息抓取子模块

信息抓取子模块主要针对一些知名站点，如 Google Code Search、Csdn、SourceForge 等进行深入分析，采用爬虫技术设计爬虫搜索工具，收集镶嵌在站点中的开源代码。

（3）信息计算子模块

爬取的代码信息不仅包括源代码，还包括用于标识开源代码的多种外在的关键信息。信息计算子模块的作用是对代码的外在信息进行提取，这些信息包括项目名称、版本号、文件名称、开发语言、下载网址、遵循的协议、开发者信息等。这类代码的属性特征包含在下载的代码数据中，采用相应的匹配识别技术就可以从代码数据中抽取得到。

2）数据分析模块

数据分析模块主要用于对获取的代码信息进行分析和分类存储，包括数据分析归类子模块和结构化信息提取子模块。

（1）数据分析归类子模块

主要根据提取的开发语言、功能类型等信息对获取的代码信息进行分类，便于后续的存储。

（2）结构化信息提取子模块

以代码文件为单位，分别生成用于文本比对、Token 比对和语法树比对的关键信息。针对文本比对算法，将源代码的文本文件按行进行 hash 计算，生成摘要 ；针对 Token 比对算法，先将代码的文本文件进行 Token 处理，再按行进行 hash 计算，生成摘要 ；针对抽象语法树比对算法，将经过词法和语法分析形成的语法树，按照数组的形式转化为字符串进行存储。

3）数据存储模块

数据存储模块主要用于对生成的各种信息进行存储，包括代码源文件名、代码作者等关键水印信息，代码的存放路径等用于标识文件的信息，以及用于代码比对的信息。

3 系统功能及实现

3.1 系统功能概述

本系统总共包括访问控制、服务器配置、比对任务分配、引擎比对、比对结果分析和比对结果输出 6 个模块。

其中，访问控制功能主要用于对不同角色的用户进行权限控制管理 ；服务器配置功能主要包括用户管理和样本库导入功能 ；比对任务分配、引擎比对功能实现具体的文本、Token、语法比对功能 ；比对结果分析功能对由比对得到的各种结果进行分析统计 ；比对结果输出功能将由比对结果分析得到的数据在浏览器端进行显示以及进行比对结果导出。系统主界面如图 4 所示。

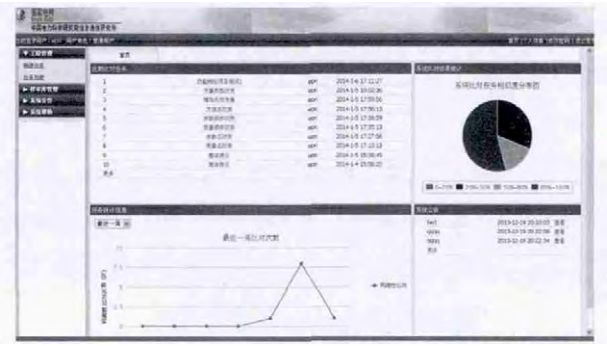

图4 同源性检测系统界面

按照以上对系统功能的划分，将 6 大功能模块组织为如图 5 所示的功能结构图。

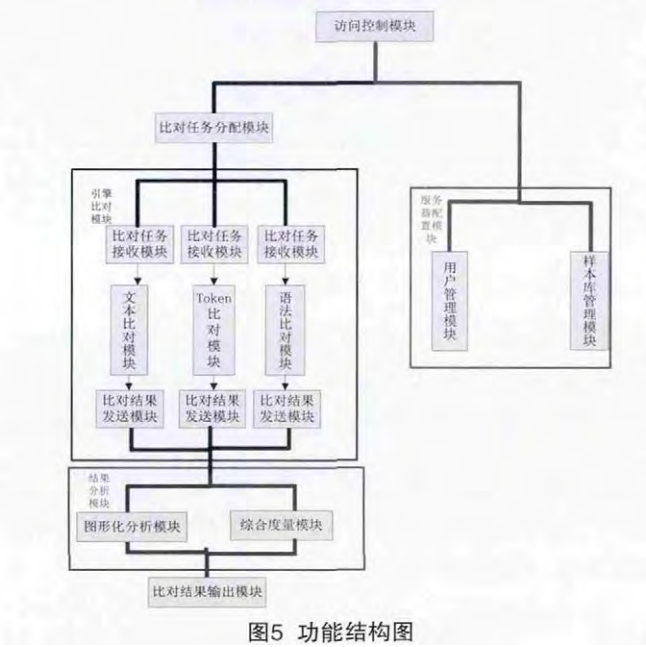

图5 功能结构图

3.2 系统主要功能模块

1）访问控制模块

此模块负责同源性综合分析系统的入口管理，包括新

用户注册和已注册用户的登录操作。

2）用户管理模块

将用户权限分为 3 类，不同的用户自动进入不同的页面：

（1）管理员。能够删除用户、通过用户申请、为用户添加权限，还能进行系统管理，包括引擎配置、样本库管理。

（2）普通用户。能使用系统进行文本、Token、语法比对，并查看比对结果，导出报告。

（3）受限用户。仅能查看以往比对结果，不能进行任何比对操作。

3）样本库管理模块

同源性综合分析的一个重点就是样本库，样本库生成模块负责样本库的管理，包括样本库的删除和添加。管理员可以选择生成样本库的源代码，然后服务器对源代码进行分析之后，存为样本库。

4）引擎比对模块

该模块首先接收从服务器分配出来的比对任务，然后根据比对任务的需要，调用相应的模块进行比对，再由比对结果发送模块将比对之后的结果返回到服务器。

5）比对任务接收模块

比对任务接收模块负责接收服务器的任务下发。当有新的任务从服务器分配到比对引擎时，该模块负责将比对任务存储到磁盘，然后遍历相关文件夹，取得需要的文件路径和比对相关参数信息。

6）文本比对模块

文本比对模块可以对软件的源代码进行文本层次的比对。首先对源代码文件的各行计算出 hash 值，然后建立此源代码文件的行链表，进行行比对和分块比对，得出文本相似度。

7）Token 比对模块

Token 比对模块可以对软件的源代码进行 Token 级别的比对。首先对软件源代码进行词法分析，得到对应源代码的 Token 序列，然后再根据此 Token 序列进行比对。Token 比对可以有效检测出更改变量名的代码抄袭行为。

8）语法比对模块

通过对源程序进行预处理、词法分析、语法分析，得到抽象语法树，再使用特定的比对算法对抽象语法树进行比对，得出文件的相似度。

9）比对结果发送模块

各个比对模块分析完源代码文件后，将比对相关结果存储到数据库文件中，比对结果发送模块的任务就是将比对结果数据库发送回服务器，供服务器做进一步的分析。

10）比对结果分析模块

服务器在接收到比对引擎返回的比对结果数据库之后，对结果进行综合分析，并反馈到用户界面。

11）图形化分析模块

图形化分析模块对比对结果进行图形化分析，得到饼状图、柱状图等直观形式的表示。用户可以通过直观图例看到比对结果的一个统计情况。

12）综合度量模块

综合度量模块也是同源性综合分析系统中的一个重要模块。此模块根据合理的综合度量算法，对软件源代码通过文本比对、Token 比对和语法比对 3 种比对方式得到的比对结果进行综合度量，得到综合相似度，从而得到全面合理的分析结果。

13）对结果输出模块

比对结果分析完成之后，用户通过比对结果输出模块得到相应的比对结果分析报告，包括 Word、PDF 和 Html 3 种格式的文档。

4 系统测试结果及分析

本次测试中将使用从 sourceforge 上下载的开源代码进行功能相似的软件之间的检测和同款软件不同版本之间的检测。测试证明本文开发的同源性综合比对系统具有实际的使用价值。

1）功能相似开源项目验证

本实验从 sourceforge 上下载具有开源代码的项目，将相同功能的两个项目分为检测项目和样本项目。首先人工分析两个项目的真实合理相似度，然后使用本文开发的同源性综合比对系统进行检测，最后对检测结果进行分析。实验选择 P2P 软件、播放器、牌类游戏、邮箱客户端和图片处理软件共 5 组开源项目，检测结果如表 1 表示。其中，相似度误差公式如下：

相似度误差 =（综合相似度 - 人工判断相似度）/ 人工判断相似度……(2)

表1 本文同源性系统对功能相似项目的检测结果

项目功能	检测项目	样本项目	综合相似度	人工判断相似度	相似度误差
P2P 软件	p2pradio-2.0	Stream-2-Stream_1.0	81.78%	80%	2.22%
播放器	Music Box_1.07	aTunes_3.09	54.01%	50%	8.02%
牌类游戏	Scala 40	FourRow Solitaire_v.63	48.62%	45%	8.04%
邮箱客户端	columba-1.4	falcon-mailer0.8.0	59.32%	60%	1.13%
图片处理	FastPhotoTagger.v2.2	JavaPEG_2.4	56.12%	55%	2.03%

从表 1 可以看出，被测的 5 组功能相似的开源项目中均存在相似代码，本文开发的同源性检测系统能够检测出其中的相似代码，并且检测的误差都在可接受的范围之内，由此可以证明本文开发的同源性检测系统可以应用到实际中。

2) 电科院项目与开源代码检测验证

本实验室将电科院开发的 5 个项目的代码与开源样本库进行比对，其中样本库为 a2gameserver,aardvarkjava,basiliskgl,behavior, bookprinter,boox, camera3d, dynamicdispatch, easycook, eclipse-study,freetts, Gamesroom，共12个开源项目。得到的检测结果如表 2 所示。

表2 本文同源性系统对实际项目的检测结果

项目名称	文本相似度	Token 相似度	语法相似度	综合相似度
公司产****系统	24.04%	71.42%	39.11%	39.58%
公司全面***8 信息系统	24.04%	71.66%	47.06%	44.74%
运营****系统	24.21%	71.48%	45.02%	43.74%
合同****管理	26.24%	69.87%	47.05%	44.43%
安监****应用系统	24.47%	73.43%	45.74%	43.43%

通过表 2 可以发现，被测的 5 个应用系统开发过程中全部存在引用开源代码的情况，部分系统引用开源代码的比例较高。这些开源代码不但会给公司造成潜在的知识产权纠纷，而且开源代码的违规使用导致缺少必要的技术支持，这会带来额外的风险。

3) 实验结果分析

通过采用本文开发的代码同源性检测系统对开源代码进行实验，证明本系统的检测结果是真实可信的，并且由检测得到的相似度与由人工判断得到的相似度仅有较低的误差。本系统能够应用到实际环境中，具有较高的使用价值。

5 结束语

本文从系统架构和功能方面对开发同源性检测系统进行了介绍，并使用该系统进行了实验测试。

同源性检测系统采用 B/S 架构设计实现，系统的主要目标是结合文本比对、Token 比对和语法比对 3 种比对方式，对软件源代码进行同源性鉴别。本系统共包括访问控制、服务器配置、比对任务分配、引擎比对、比对结果分析和比对结果输出 6 个模块，便于对整个比对流程进行管理。

使用同源性检测系统对构造的各种抄袭类型进行检测并与工具检测结果进行比较，证明本系统能够检测更多的抄袭类型。使用同源性检测系统对真实的开源代码进行检测，证明本系统可以应用到真实环境中并具有一定的使用价值。（责编 马珂）

参考文献

[1]Brenda S. Baker. A Program for Identifying Duplicated Code[C]. In Proceedings of Computing Science and Statistics: 24th Symposium on the Interface, Vol. 24:49-57, March 1992.
[2]J Howard Johnson. Identifying Redundancy in Source Code Using Fingerprints [C]. In Proceeding of the 1993 Conference of the Centre for Advanced Studies Conference (CASCON ' 93), pp. 171-183, Toronto, Canada, October 1993.
[3]S. Ducasse, M. Rieger, S. Demeyer, A Language Independent Approach for Detecting Duplicated Code [C]. in:Proceedings of the 15th International Conference on Software Maintenance, ICSM 1999, pp. 109-118.
[4]Li, Z., Lu, S., Myagmar, S., et al. CP-Miner: A tool for finding copy-paste and related bugs in operating system code [C]. In OSDI, 2004:289-302.
[5]Toshihiro Kamiya, Shinji Kusumoto, Katsuro Inoue. CCFinder: A multilinguistic token-based code clone detection system for large scale source code [J]. IEEE Transactions on Software Engineering, 2002,28(7):654-670.
[6]Muddu, B.，Asadullah, A.，Bhat, V. Software Clones (IWSC)[C].2013 7th International Workshop on Digital，2013，Page(s): 39- 45.
[7] 王伟，苏小红，马培军，等 . 标识符重命名不一致性缺陷的检测 [J]. 哈尔滨工业大学学报，2011，43（1）：89-94.
[8] 于世英，袁雪梅，卢海涛，等 . 基于序列聚类的相似代码检测算法 [J]. 智能系统学报 ,2013，8（2）: 52-57.
[9]I. Baxter, A. Yahin, L. Moura，et al. Clone Detection Using Abstract Syntax Trees[C]. In ICSM, pp. 368-377, 1998.
[10]S. Bellon, R. Koschke, G. Antoniol, et al. Comparison and Evaluation of Clone Detection Tools[J]. IEEE TSE,2007,33(9):577-591.
[11]Khaustov Pavel A. Strategic Technology (IFOST)[C].2012 7th International Forum on Digital，2012 , Page(s): 1-3.
[12]Narayanan S.，Simi S. Computer Science & Education (ICCSE)[C]. 2012 7th International Conference on Digital，2012 , Page(s): 1065-1068.
[13] 刘丽霞，张志强 . 基于 Trie 树的相似字符串查找算法 [J]. 计算机应用，2013，33(8) : 2375-2378.
[14] 李虎，刘超，刘楠，等 . Java 源代码字节码剽窃检测方法及支持系统 [J]. 北京航空航天大学学报，2010，36（04）：424-428.
[15]Chan, P.P.F. ，Hui, L.C.K.，Yiu, S.M. Information Forensics and Security[J].IEEE Transactions on Volume: 8 , Issue: 1，2013 , Page(s): 101 -110.
[16]Cosma, G. ，Joy, M. Computers[J].IEEE Transactions on Volume: 61 , Issue: 3，2012 , Page(s): 379-394.
[17]Arabyarmohamady, S.,Moradi, H., Asadpour, M. Interactive Mobile and Computer Aided Learning (IMCL)[C].2012 International Conference on Digital，2012 , Page(s): 180 - 186.
[18]WANG Kechao, WANG Tiantian, ZONG Mingkui, et al. Detection of Plagiarism in Students ' Programs Using a Data Mining Algorithm[C]. Computer Science and Network Technology (ICCSNT), 2012 2nd International Conference，2012 , Page(s): 1318-1321.