正则表达式在编程题自动阅卷中的应用

余石泉,周肆清
(中南大学,湖南 长沙 410083)

摘 要:分析了编程题的自动阅卷的现状及存在的不足。为了让计算机能够更加合理和有效地进行编程题的阅卷,提出了一种用正则表达式来分析程序代码,模拟人工阅卷的方法。介绍了此方法的原理和主要功能,给出了方法实现的具体步骤,对方法的关键部分如Java处理正则表达式的各个类以及方法的具体实施等都进行了详细的解释并举例进行了说明。其核心思想是利用正则表达式来抽象标准答案,再利用正则表达式强大的文本匹配功能来进行程序代码的分析,把不变的文本转化为具有一般性的表达式,大幅度增加了匹配的灵活性,从而达到简化阅卷的目的。

关键词:正则表达式;抽象;文本匹配
中图分类号:G434 **文献标识码**:A **文章编号**:1673-629X(2007)07-0244-03

Application of Regex in Auto-Checking Paper of Programs

SHE Shi-quan, ZHOU Si-qing
(Central South University, Changsha 410083, China)

Abstract: Analyzed the status quo and the shortages of auto-checking answer sheet of programs. In order to look over the examination papers by computer more reasonably and effectively, a method was given which used regular expressions to analyze the codes of program and simulate the artificial model. This paper also introduced the elements and chief function of the method, and the implementary process was given too; moreover, the pivotal parts of the method like the classes of Java that used to manage regular expressions and concretely actualization of the method were explained amply and examples were additional. The central idea is to changeing for complicated standard solution to a series of regular expressions, and utilize their powerful function in text matching to analyze the programming codes, thereby the aim of predigesting for looking over answer papers is achieved.

Key words: regular expressions; abstract; text matching

1 问题的提出

进入21世纪以来,随着计算机的普及,计算机自动阅卷技术的应用已经越来越广泛。客观题的自动阅卷技术已经发展成熟,但主观题乃至编程题的阅卷技术却停滞难前,一直在束缚着计算机自动阅卷技术的发展。目前对编程题阅卷普遍采用的办法是查看考生程序的运行结果,对则满分,错则零分。这种做法与人工阅卷是背道而驰的,很不公平,甚至还给了学生投机舞弊的机会。而有些阅卷系统中虽然加入了对考生结果代码进行关键词的比较机制,但只是进行简单的字词比较,况且存在如何选择关键词的问题。

基于以上所述,有人提出了一种对考生所编写的源程序进行分析检测的办法[1]。也即对考生的源程序和预先做好的一个尽可能全面的标准答案进行比较,根据比较的结果,得到一个称之为代码可信度的数据,把这个数据列入到评分模式之中。代码可信度越高,则得分越高。这种做法比之前者有了较大的改进,但该方法还只是对文本的一种简单比较,效果当然不是那么理想,所以在实现方法上还可以进一步去研究。

文中提出了一种基于正则表达式来分析代码可信度的方法,即利用正则表达式来进行文本的检测,此想法是首先把程序的代码按照功能分块,看考生的源程序是否含有类似的功能模块,得到代码可信度。

2 人工阅卷模式分析

众所周知,任何一种编程语言实现某种功能的方法都不唯一,就拿C语言来说,实现一个循环便有while,for,do-while三种方法,稍微复杂的程序其代码便可能有很多种,因此很难以一个简单的文本作为标准答案。在人工阅卷的时候,如果未对题目解题方法做明确规定的话,各种可能的答案都应是正确的,因此

收稿日期:2006-09-14
作者简介:余石泉(1980-),男,湖南邵阳人,硕士研究生,研究方向为数据库技术、人工智能;周肆清,副教授,硕士生导师,研究方向为图像处理与数据库技术。

阅卷的教师要考生按照题意完成了题目的要求,就可以算是正确的,并不会计较考生以何种方法完成。在给分的时候,阅卷老师也是按照考生的代码对题目直接或间接要求的一些功能的实现情况来给分,而并不仅仅依靠程序的运行结果。下面举例进行说明。

题目:读入一个4位正整数,然后把它逆序输出。

分析:这个题目的关键点是把这个4位正整数进行分解,然后把各个数位逆转过来,再重新组成一个新的正整数。考生如果能够实现这一段代码,应该要获得较高的分数。另外输入和输出以及运行结果也占有一定分值。假如这个题目共占8分,整数分解重组的代码块占5分,记做块A,输入、输出、运行结果各占1分,分别记做块B,C,D。阅卷教师即可按照得分点来进行评分了。比如,某考生只做了输入和输出,其C语言代码如下所示,则得2分,即获得B和C块的得分,D块的得分必须依赖于A块,否则所做出的运行结果即使正确也不能得分。

```
void main()
{
  int n;
  canf("%d",&n);
  printf("%d",n);
}
```

当然,还要考虑考生代码存在错误书写或语法错误等情况,这也是无法避免的,但只要错误不多,阅卷老师一般都会不扣分或者少扣分。

自动阅卷应当以人工阅卷模式为参考,尽量模拟人工阅卷,只有这样,才可能做到公平合理。因此,文中提出一种利用正则表达式来对考生代码进行检测的方法。当然此方法只是整个阅卷系统中的一部分,它的最终结果只是要得到一个代码可信度,并不能单独来充当阅卷的工具。具体的步骤如下:

1)按照题意,分析出程序的得分点;

2)制定每个得分点的正则表达式;

3)扫描考生程序,利用正则表达式与考生程序进行匹配;

4)根据匹配情况得到代码可信度;

此方法的关键步骤是2)和3),因此文中将重点分析2)和3)的实现。

3　Java处理正则表达式简介

正则表达式的英文名是Regular Expression,缩写为Regex,其起源最早可以追溯到1956年,后来被应用到信息技术领域。正则表达式第一个实用应用程序是Unix中的qed编辑器,而现在,正则表达式引擎已经被Unix中的许多工具所实现。另外,正则表达式在许多脚本语言中也有着非常重要的地位。它的主要的功能是处理文本和匹配模式,常用于在文本中进行查找和替换,或者检验文本是否符合某一模式的规则。比如可以用正则表达式查找某文本中的所有的字符串"hi",并把查到的全部字符串替换成"hello";也可以用正则表达式来检验某个输入字符串是否符合Email地址的格式[2]。

下面简单介绍一下Java处理正则表达式的两个重要的类:Pattern类及Matcher类。Pattern类提供了一个compile方法,这是一个静态的方法,它接收一个正则表达式作为参数,返回此正则表达式的一个Pattern类对象;然后再利用该对象来调用Matcher方法。Matcher方法接收一个字符序列作为参数,即应用中的待匹配的文本,返回一个Matcher类对象。

Matcher类提供诸如matches, find, lookingAt, group, replaceAll等方法。其中的matches方法与Pattern类中的matches方法完成相同的任务,但它不接收任何参数,因为匹配模式(正则表达式)和匹配对象(待匹配的文本)都已经封装在了Matcher类的对象中[3]。

这里,笔者用一个简单的例子来介绍一下Java中如何使用正则表达式进行文本匹配工作的。正则表达式为"^\w+@\w+\.[a-zA-Z]{2,3}$",用来检测某个Email地址的书写格式。\w+表示一个或多个词字符,词字符包括大小写字母、数字和下划线;\.匹配字面意义符号".",而并非正则表达式中".";[a-zA-Z]{2,3}表示2个或者3个字母的组合。在Java中,有些字符必须经过转义,而在正则表达式中也有转义,因此正则表达式"\w"在Java中会被写成"\\w"。如下代码将产生一次成功的匹配[4]:

```
Pattern p=Pattern.compile("^\\w+@\\w+\\.[a-zA-Z]{2,3}$");
Matcher m=p.matcher("HelloRegex@126.com");
m.matches();
```

4　方法具体实现

再来分析前面所举的例子,得分点不再赘述,正则表达式则可以这样来构造:

1)对于代码块B和C,要检测考生是否做了正确的输入和输出,正则表达式如下。

输入:"scanf\(\"%d\s*\",&[a-zA-Z_]\w*\)";

输出:"printf\(\"%d\s*\",[a-zA-Z_]\w*\)"。

2)代码块A的检测:对于这一代码块,主要检测

考生是否对输入的整数做了分解和重组，因此主要检测考生程序里面是否有以下类似的代码：其中 a 表示输入整数的个位，b 表示十位，c 为百位，d 为千位，newInt 表示重组的逆序整数。

a=n%10; b=n/10%10; c=n/100%10; d=n/1000; newInt=a*1000+b*100+c*10+d;

当然，这并不是唯一的正确答案，比如说求百位上的数字 c 也可以这样：c=n%1000/100。

正则表达式并不能分析语义，使用它只能造一个尽可能通用的模式来和源程序进行匹配。通过构造正则表达式能有效地判断程序中是否具有符合某个模式的代码块，这要求正则表达式应该尽可能地描述出代码块的特点，并应该尽量减少歧义。正则表达式设计得越好，阅卷效果也就越好。比如上述代码块的特点如下：

(1)五个相连的赋值表达式(理论上考生也可能把它们分开写，但这种可能性很小)。

(2)最后一个表达式可以较精确的匹配，因为它的代码模式较为单一，只需考虑考生把诸如 a*1000 写成 1000*a 即可，当然还需考虑括号的存在。

按照以上所述，对于前面四个赋值语句构造正则表达式如下：

```
(\w+\s*=\s*\(?\w+\s*([\+\-\*/%]\s*\w+\)?){1,3};\s*){4}        (1)
```

最后一个赋值语句可以这样构造：

```
\w+\s*=(\s*\(?((\w+(\s*\*\d{0,4})?)|((\d{1,4}\s*\*\s*)?\w+))\)?\s*\+){3}|\(?((\w+(\s*\*\d{0,4})?)|((\d{1,4}\s*\*\s*)?\w+))\)?;        (2)
```

为了方便直观，文中略去 Java 代码，利用一个正则表达式测试器来测试，如图 1 所示[5]。这里只重点地说明一下如何查找块 A。Source 栏输入的代码是一段最普通的代码，如下所示：

```
void main()
{
  int n,a,b,c,d; scanf("%d",&n);
  a=n%10; b=n/10%10; c=n/100%10; d=n/1000;
  newInt=a*1000+b*100+c*10+d;
  printf("%d",newInt);
}
```

Pattern 栏里输入的是正则表达式(1)和(2)。这里没有考虑考生在代码中做注释的情况，实际上这种情况通过稍微改写上述正则表达式便可达到目的，这里就不再赘述了。Results 栏里面是匹配的结果，如果匹配不成功，Results 栏中显示 No matches，匹配成功则显示 Source 栏里符合 Pattern 栏中正则表达式模式的字符串。根据各个模块匹配的情况来得到整个程序的代码可信度。

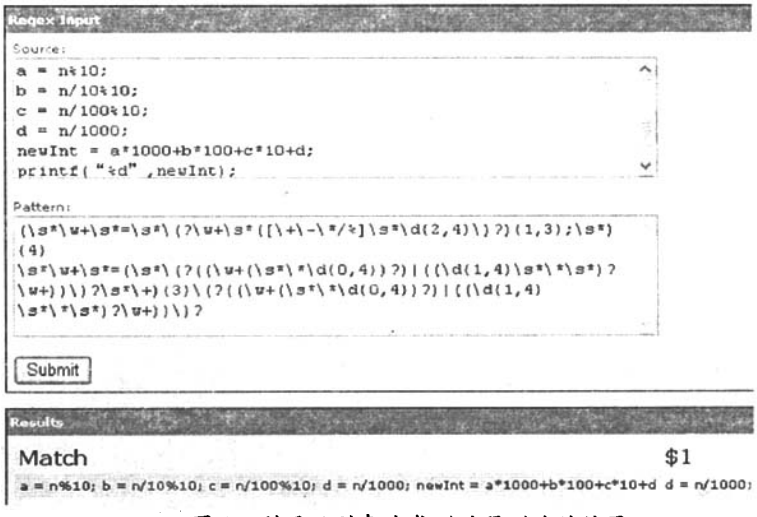

图 1 利用正则表达式测时器测试的结果

5 结束语

通过上述的例子可以看出，利用正则表达式来分析代码，具有较大的灵活性，它能够较好地解决当前阅卷模式呆板的问题。笔者应用此方法进行了大量实验，效果较好。但由于正则表达式无法分析表达式的语义，也就无法判断整个程序是否正确，所以此方法还必须依赖编译器和其他模块。另外，应用此算法比较麻烦的原因在于必须事先手工构造正确的正则表达式，但一旦构造出来恰当的表达式，阅卷便会变得简单，较少的付出能收到巨大的回报，付出的代价还是值得的。

参考文献：

[1] 乔善平，刘 振，朱 波. 编程题的计算机阅卷[J]. 计算机应用研究，2004(8)：229－231.

[2] 钟京旭. Java 中的正则表达式及其应用[J]. 电脑编程技巧及维护，2005(6)：50－57.

[3] Sun 公司. The Java Tutorial[EB/OL]. 2005. Http://java.sun.com/docs/books/tutorial/index.html.

[4] Regular－Expressions info. How to Find or Validate an Email Address[EB/OL]. 2006－03. http://www.regular－expressions.info/email.html.

[5] Regex Expression Library. The Regex Tester[EB/OL]. 2001. http://regexlib.com/RETester.aspx.