代码搜索引擎设计文档

二〇一八年七月

目录

1、设计板块

1）数据库设计
2）全文搜索引擎设计
3）代码搜索引擎设计
4）OJ 评测设计

2、总体架构

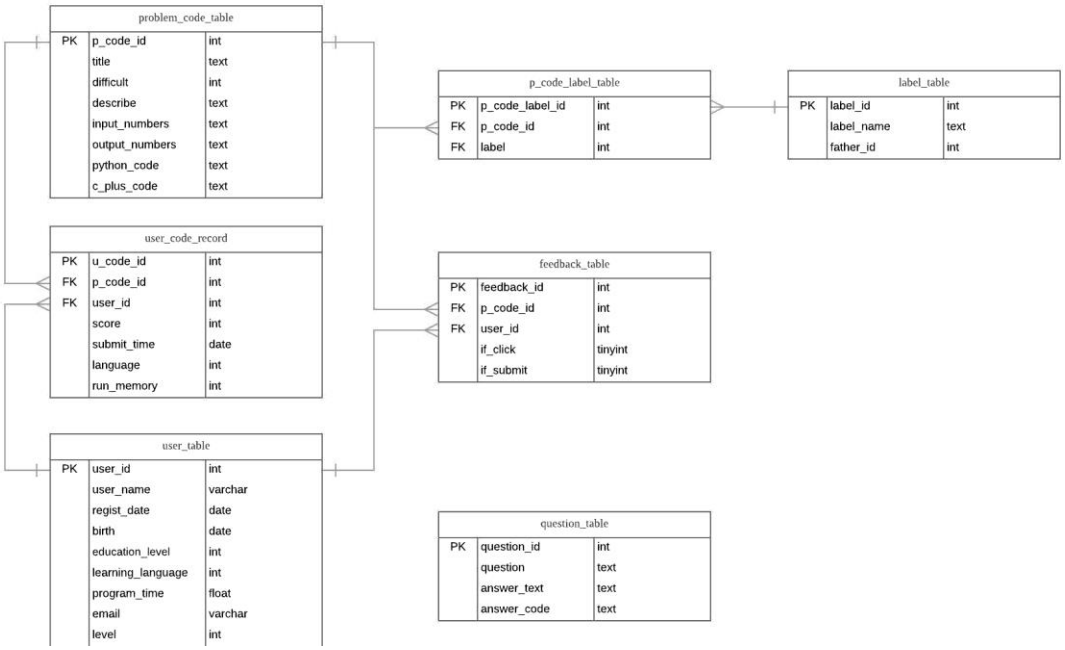
2.1 系统功能表

序号	功能模块		功能描述
1	用户管理	用户注册	用户注册
2		用户登录	用户登录
3		用户注销	用户注销
4		用户信息管理	信息修改
5	搜索功能	文本搜索	普通搜索
6			高级搜索
7		代码搜索	代码搜索
8	OJ 评测	代码评测	Python 代码评测
9			C++代码评测

3、详细设计

3.1 数据库设计

3.1.1 E-R 图



3.1.2 数据表设计

① 表名：problem_code_table

列名	数据类型	长度	主键	外键	允许空
P_code_id	Int		是		否
Title	Text				是
Difficult	Int				是
describe	Text				是
Input_numbers	Text				是
Output_numbers	Text				是
Python_code	Text				是
C_plus_code	Text				是

② 表名：label_table

列名	数据类型	长度	主键	外键	允许空
Label_id	Int		是		否
Label_name	varchar				否
Father_id	int				否

③ 表名：p_code_label_table

列名	数据类型	长度	主键	外键	允许空
Pl_id	Int		是		否
P_code_id	Int			是	否
Label_id	int			是	否

④ 表名：user_table

列名	数据类型	长度	主键	外键	允许空
User_id	int		是		否
User_name	Varchar				否
Regist_date	Date				否
Birth	Date				否
Education_level	Int				否
Learning_language	Int (1,2,3)				否
Programm_time	Float				否
Email	Varchar				是
Level	Int				否

⑤ 表名：user_code_record

列名	数据类型	长度	主键	外键	允许空
Ucode_id	Int		是		否
P_code_id	Int			是	否
User_id	int			是	否
Score	Int				否
Submit_time	Date				否
Language	Int				否

⑥ 表名：feedback_table

列名	数据类型	长度	主键	外键	允许空
Feedback_id	Int		是		否
User_id	Int			是	否
P_code_id	Int			是	否
If_click	Tinyint				否
If_submit	Tinyint				否

⑦ 表名：question_table

列名	数据类型	长度	主键	外键	允许空
Question_id	int		是		否
Question	text				否
Answer_text	text				是
Answer_code	text				是

3.2 爬取技术

针对 python、c++代码 Scrapy，Python 开发的一个快速、高层次的屏幕抓取和 web 抓取框架，用于抓取 web 站点并从页面中提取结构化的数据。

3.2.1 爬取网页

3.2.1.1 南阳理工学院 OJ

题目、一个测试样例、难度、分类（考察知识点）
1340 道题
只判断 c/c++/java
http://acm.nyist.edu.cn/JudgeOnline/problemset.php

3.2.1.2 Python2.7 100 例

题目、源代码

http://www.runoob.com/python/python-100-examples.html

3.2.1.3 c + + 经典编程题汇总

题目、源代码

100 道

https://blog.csdn.net/qq_36864672/article/details/76037595

3.2.1.4 杭州电子科技大学 online judge

题目、一个测试数据、相关题 id 推荐
5200 道题

3.2.1.5 蓝桥杯习题

题目、程序代码（90%C,部分 C++和 java）、测试数据（多组）
200 道
90%习题使用 C 语言解答，部分使用 C++或者 Java
https://blog.csdn.net/rodestillfaraway/article/details/50529597
题目在网页上，测试样例在压缩包中

3.2.1.6 CSDN C + + 经典编程题汇总

题目、程序代码
100 道题
https://blog.csdn.net/qq_36864672/article/details/76037595

3.2.1.7 CSDN 刷题汇总 python 版

题目，代码，知识点(字符串、动态规划)，分析

35 道

https://blog.csdn.net/dongrixinyu/article/details/78775057

是为了能够提升自己的数据结构和算法的水平，以及码代码的速度和熟练度。

所有的代码都提交到了我的 github 上面，并且不定期更新和优化：冬日新雨的github：数据结构和算法刷题代码下载

1、字符串：

- 求数组中两个字符串的最小距离 Python 版
- KMP 算法 Python 版
- 分解调整字符串中的字符 Python 版
- 将字符串中的空字符全部替换为别的字符串 Python 版
- 在有序但含有None的数组中查找字符串 Python 版
- 判断字符数组中是否所有的字符都只出现过一次 Python 版
- 获取字符串的统计字符串 Python 版
- 将整数字符串转成整数值 Python版
- 判断两个字符串是否互为旋转词 Python版
- 去掉字符串当中的连续k个0，Python版
- 计算字符串中所有数字之和Python版
- 判断两字符串是否互为变形词Python版
- Anagrams 归类 的 python 版本代码
- 括号字符串的相关问题 Python 版

2、数学计算：

- 求两数的最大公约数 Python 版
- 超级素数幂 Python 版
- 斐波那契数列的计算方法

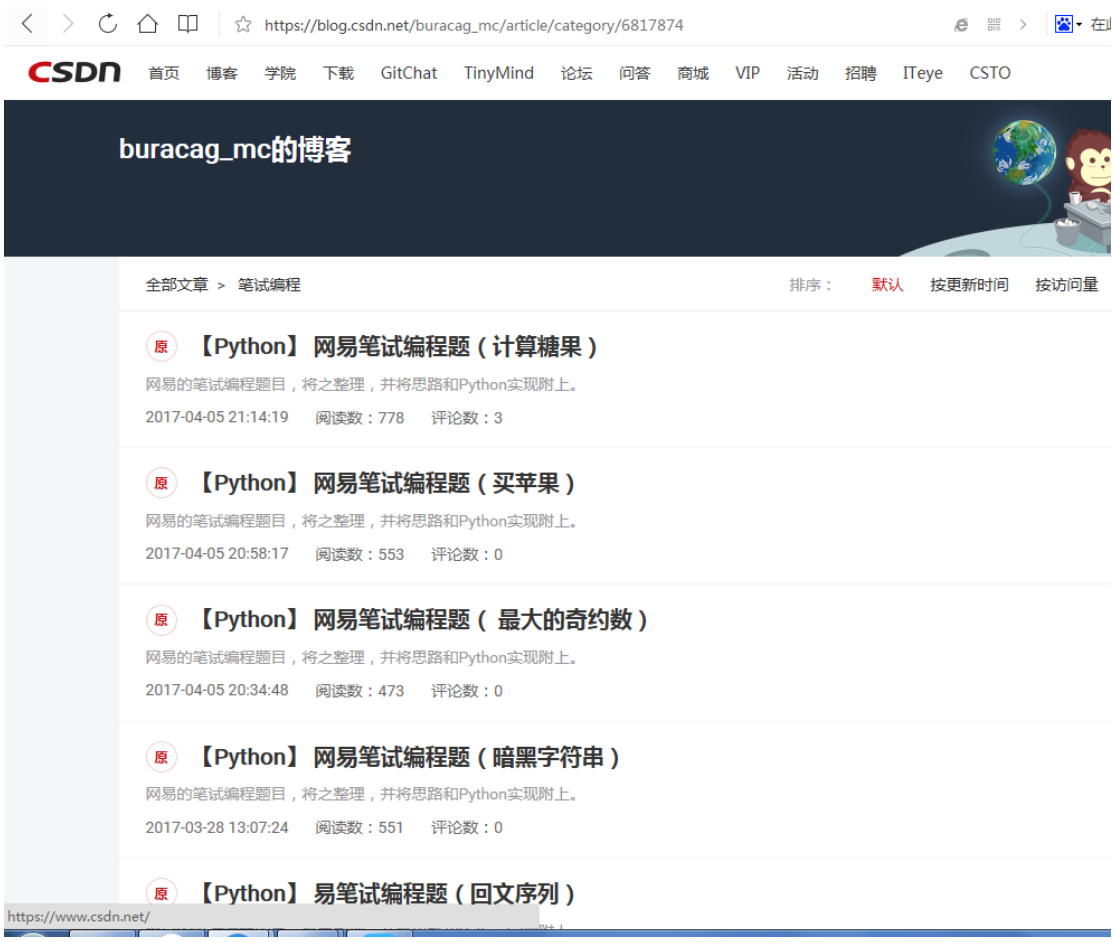
3、动态规划：

3.2.1.8 CSDN 网易笔试编程题 python 实现

题目、输入/输出例子、源代码

9 道

https://blog.csdn.net/buracag_mc/article/category/6817874



3.2.1.9 NOIP 历年题目(全国青少年信息学奥林匹克联赛

题目、输入/输出格式、一个测试样例、答案代码（C）
https://www.cnblogs.com/shenben/category/840423.html

3.3 正文搜索

正文搜索模块的主要功能是在用户提交字符串题面信息，完成相关题的搜索。系统会通过搜索引擎查找服务器上的相关题，传给用户进行选择。

3.3.1 功能描述

普通查询：通过向搜索框输入字符串(题面)完成相关题搜索，并可通过事物的这些属性（例如：题的难度、语言类型、知识点）不断筛选过滤搜索得到的题目，按照相关性大小排序。

高级查询：按照表单要求输入标题、题面、知识点、语言、难度(可为空)进行检索,按照相关性大小排序。

3.3.2 搜索引擎选型:

Elasticsearch (基于 Apache Lucene(TM)的开源搜索引擎)

特点：　分布式的实时文件存储，每个字段都被索引并可被搜索;

分布式的实时分析搜索引擎;(近实时)

可以扩展到上百台服务器，处理 PB 级结构化或非结构化数据;(集群处理)

ES 作为搜索引擎常用查询:

全文本查询：针对文本类型的数据;

字段级别查询(结构化查询): 针对结构化数据，如数字、日期等

3.3.3 ES 索引类比

在 Elasticsearch 中，文档归属于一种类型(type),而这些类型存在于索引(index)中，简单的对比图来类比传统关系型数据库:

Relational DB -> Databases -> Tables -> Rows -> Columns

Elasticsearch -> Indices -> Types -> Documents -> Fields

默认情况下，文档中的所有字段都会被索引（拥有一个倒排索引），可被搜索。

3.3.4 ES 检索文档

我们只要执行 HTTP GET 请求并指出文档的“地址”——索引、类型和 ID 既可。根据这

三部分信息，我们就可以返回原始 JSON 文档。

简单搜索：GET　/megacorp/employee/_search?q=last_name:Smith

megacorp 索引和 employee 类型，在结尾使用关键字_search 来搜索姓氏中包含‘Smith’的数据。

使用 DSL 语句查询:DSL(Domain Specific Language 特定领域语言)以 JSON 请求体的形式出现。

```
GET　/megacorp/employee/_search
{
    "query" : {
        "match" : {
            "last_name" : "Smith"
        }
    }
}
```

全文搜索:　GET　/megacorp/employee/_search

```
{
    "query" : {
        "match" : {
            "about" : "rock climbing"
        }
    }
}
```

短语搜索：确切的匹配若干个单词或者短语。

例子:

```
GET /megacorp/employee/_search
{
    "query" : {
        "match_phrase" : {
            "about" : "rock climbing"
        }
```

```
        }
    }
```

查询返回 John Smith 的文档：

```
{
    ...
    "hits": {
        "total":           1,
        "max_score":    0.23013961,
        "hits": [
            {
                ...
                "_score":          0.23013961,
                "_source": {
                    "first_name":    "John",
                    "last_name":     "Smith",
                    "age":           25,
                    "about":         "I love to go rock climbing",
                    "interests": [ "sports", "music" ]
                }
            }
        ]
    }
}
```

3.4 代码搜索引擎

3.4.1 词法分析

一个 Python 程序先由词法分析器生成 token 序列，然后再将 token 序列输入语法分析器，进行语法分析生成抽象语法树。

词法分析：将一段字符序列 （如一段程序代码），转化成为一个 token 序列。

标识符：

由字母 (包含大小写)，数字，下划线 (_) 组成，其中，标识符的首位必须为字母或下划线，不能为数字。

关键字（保留字）：

False	await	else	import	pass	None	break
except	in	raise	True	class	finally	is
return	and	continue	for	lambda	try	as
def	from	nonlocal	while	assert	del	global
not	with	async	elif	if	or	yield

操作符：

+	-	*	**	/	//	%
@	<<	>>	&	|	^	~
<	>	<=	>=	==	!=	

定界符：

(	)	[	]	{	}	,
:	.	;	@	=	->	+=
-=	*=	/=	//=	%=	@=	&=

实值：

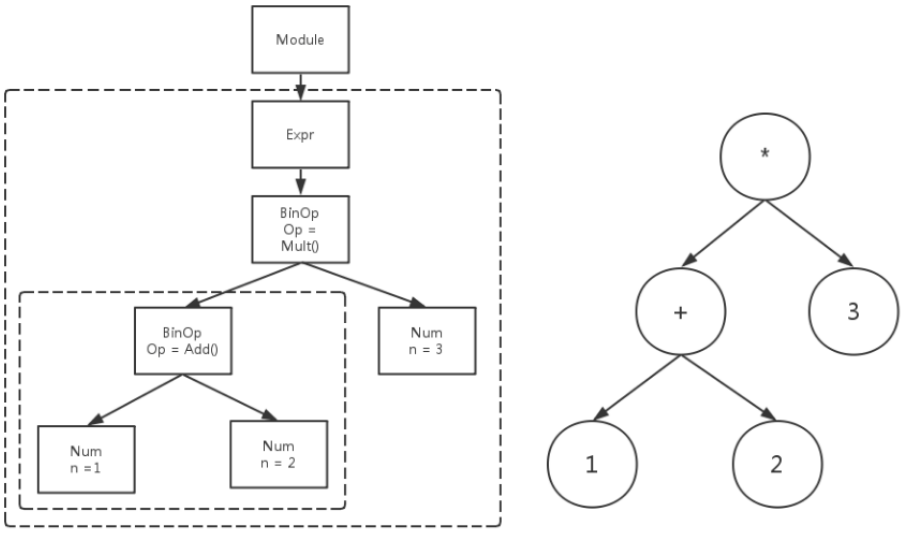
整数型，浮点型，字符串型

3.4.2 语法分析

语法分析器基于特定语言的语法，将 token 序列（由词法分析器生成）转换成一个抽象语法树。

抽象语法树（Abstract Syntax Tree，AST），或简称语法树（Syntax tree），是源代码语法结构的一种抽象表示。它以树状的形式表现编程语言的语法结构，树上的每个节点都表示源代码中的一种结构。之所以说语法是“抽象”的，是因为这里的语法并不会表示出真实语法中出现的每个细节。比如，嵌套括号被隐含在树的结构中，并没有以节点的形式呈现。

例如，（1 + 2）* 3 的语法树如下：



3.4.3 分析工具

Python 语言的抽象语法树的生成工具考虑使用以下两者:

3.4.3.1 PLY package

PLY（python lex yacc）是由 Python 实现的 lex 和 yacc 工具。

PLY 的设计目标是尽可能的沿袭传统 lex 和 yacc 工具的工作方式，提供丰富的输入验证、错误报告和诊断。

3.4.3.2 Python ast module

Python 的 ast 模块，能够通过 parse()内置函数生成抽象语法树，并提供对抽象语法树的遍历。

结点类型有： Num, Str, Name, Expr, keyword, Assign, Print, Import, If, For, While, Return.....(99 种)

3.4.4 代码相似性度量

3.4.4.1 编辑距离

定义：给定两个树，编辑距离就是将 a 树转化成 b 树所需要的最少操作次数。
操作只允许以下三种:

- 删除一个结点

- 插入一个结点
- 替换一个结点

通过对程序代码进行语法分析，得到程序的 AST，然后计算两段代码的 AST 之间的编辑距离，即可度量两段代码的相似性。

注意：

两个树之间的编辑距离确实可以有效捕捉他们之间的相似性，但是这并不是一种高效的算法，因为：

(1) the complexity of computing the editing distance between two trees is expensive.

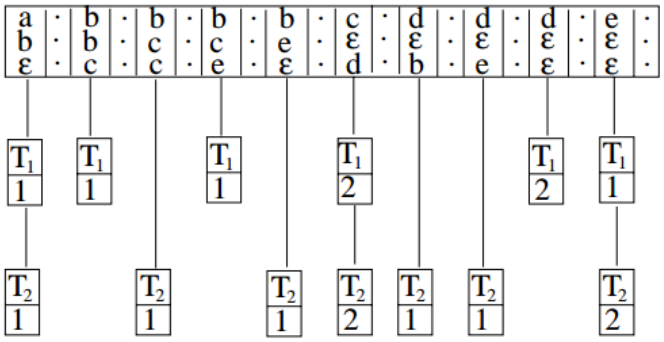
(2) it requires many pairwise comparisons to locate similar code.

3.4.4.2 夹角余弦值 / 欧氏距离

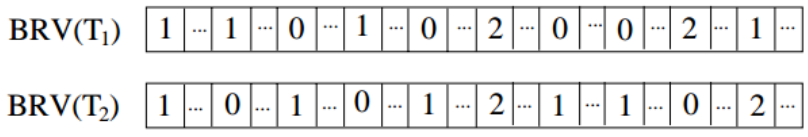
使用数值型向量来近似表达树的结构信息，从而将计算树的相似性问题转化为计算两个向量之间的相似性问题。

3.4.4.2.1 建立索引

对所有的语法树的子分支（子树）建立倒排索引，索引包含两个部分：
(1) 词典：数据库代码中存在的所有子树分支
(2) 倒排表：每一个词项的倒排表记录了该词项在对应的语法树中出现的次数



(a) Inverted File



(b) Binary Branch Vectors

如上图所示，每个语法树的向量表达可以从索引文件中得到，向量的每个维度代表每个子树分支在对应语法树中出现的次数。

3.4.4.2.2 计算相似性

夹角余弦值:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

欧式距离:

$$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

3.5 Online Judge 判题模块

判题模块的主要功能是，编译运行用户提交的代码，测试代码的正确性，并依据一系列的评价标准，给予用户相应的反馈。

3.5.1 判题流程

1. 针对一个题目，用户指定一种编程语言，编写该题目对应的解题代码。
2. 用户点击提交按钮，数据库中写入用户-题目-解题代码相关的数据条目。
3. 将用户提交的代码保存为代码文件，并根据题目获取的测试输入文件和测试输出文件。
4. 编译运行该代码文件，输入测试样例，得到输出结果。
5. 判断用户提交代码输出和测试输出之间的匹配情况，以此为据计算最后得分。

在对用户提交的代码进行评测的过程中，可能会出现如下问题:

1. AC (Accepted): 用户提交的代码运行正常，并通过了所有的测试点。计做满分。
2. WA (Wrong Answer): 用户提交的代码正常，但并没有通过所有的测试点，某些测试点错误。部分计分。
3. TLE (Time Limit Excced): 用户提交的代码异常，运行时间过长，超出了该题目的时间限制。按照错误处理。
4. OLE (Output Limit Excced): 用户提交的代码异常，输出内容超出了某个该题目的输出限制。按照错误处理。
5. MLE (Memory Limit Excced): 用户提交的代码异常，运行使用内存超出了该题目的内存使用限制。按照错误处理。
6. RE (Runtime Error): 用户提交的代码通过了编译，在运行时出现了异常。按照错误处理。
7. PE (Presentation Error): 用户提交的代码运行正常，且输出的结果正确，但是展示格式错误。按照错误处理。
8. CE (Compile Error): 用户提交的代码编译错误。按照错误处理。

以面向的用户群体为标准，以上规则可以进行调整。比如，只展示 AC 和 WA 两种，即全部通过和部分通过。除此之外的评判标准只在后台保留，不加展示。

3.5.2 安全性保证

因为用户提交的代码是不被信任的，所以不应该直接在服务器环境运行，需要限制用户提交代码的运行环境和资源使用情况。主要跟踪用户代码的运行时内存使用情况、CPU 占用情况、文件访问情况，保证用户代码在使用正常资源的情况下，能够返回结果，并不对运行环境造成危害。

假设运行环境为 Linux 操作系统。在实施过程中，需要调用 Linux 系统 API 以达到对资源进行追踪限制的目的；需要使用 seccomp 功能，达到用户代码和系统环境隔离的目的。