基于结构和语义相似度的 SQL 程序评分模型①

陈 洁

(中华女子学院 计算机系, 北京 100101)

摘 要: 针对 SQL 查询程序实现多样性的问题, 提出一种用于精确评估 SQL 程序的评分模型. 首先基于通用标准的 SQL 语法规范标识符和命令子句, 基于同义词链和抽象语法树规范表达式, 将 SQL 程序转换成统一的中间形式, 充分消除 SQL 程序句法和语义表达多样性带来的差异; 然后, 模拟人工评分思想, 对标准化后的程序按评分点组成评估单元序列, 采用改进的最长公共子序列算法评估代码相似度, 按评分点权重计算成绩, 并给出错误定位; 最后, 通过样例测试和分析说明了评分模型的有效性.

关键词: SQL 查询; 程序标准化; 抽象语法树; 最长公共子序列; 自动评分

SQL Program Grading Model Based on Structure and Semantic Similarity

CHEN Jie

(Department of Computer Science, China Women's University, Beijing 100101, China)

Abstract: In view of the diversity of SQL query program, an accurate scoring model is presented. First, based on the common standard SQL syntax specification identifier, command clause, the synonym chain and the abstract syntax tree, SQL program is converted into a kind of unified intermediate form, fully eliminating the SQL program syntax and semantic differences. Then, referring to the artificial grading thought, the standardized code is transformed into the token sequence according to grading points, and the improved algorithm for Longest Common Subsequence(LCS) is used to grade the program similarity. The scores are calculated according to the weight of the scoring points, and the error location is given as well. Finally, samples are tested and anglicized to illustrate the effectiveness of the grading model.

Key words: SQL query; program standardization; abstract syntax tree; LCS; auto-grading

SQL 查询设计是数据库技术的一个重要应用, 其设计方法灵活, 表达方式多样化, 为实现同一要求, 可以使用不同的命令式子; 而在同一个命令式子中, 同一个数据项可以有不同的表示形式, 同一个表达式也可以有多种表示方式. 这种多样性给机器的自动评阅带来了很大困难, 成为影响评阅精度的重要因素.

字符串的相似度查询和语句相似度计算已成为研究热点[1-3]. 文献[4]给出了一个基于结构相似匹配的评估模型, 该模型借鉴 Petro vskiy[5]提出的通过提取 SQL 查询语句框架进行异常检测的方法, 重点评估 SQL 查询程序的结构, 即命令关键字和运算符, 对标识符(如字段名、表名等)和表达式则不做检测. 但在实际应用中, 标识符和表达式具有灵活的设置方法和多样化的表达方式, 可变性范围很大. 为了更全面和准确地考查学习者对 SQL 查询程序的设计能力, 本文借鉴文献[6-8]中提出的综合句法结构及语义相似度的评估模型, 根据句法和语义两个层次来评估学生程序的正确程度.

为此, 针对 SQL 程序语义表达多样性的问题, 首先采用多维度的语义消歧方法, 对命令子句、标识符、计算表达式和逻辑表达式进行标准化处理; 然后将基于动态规划方法的最长公共子序列算法(Longest Common Subsequence, LCS)[9,10]应用于 SQL 程序相似匹配中, 并考虑不同评分点的分值权重, 计算学生程

① 基金项目:中华女子学院科研基金(KY2016-03011)

收稿时间:2016-08-13;收到修改稿时间:2016-09-27 [doi:10.15888/j.cnki.csa.005750]

序和模板程序的相似度, 使评分过程更加客观公正, 评分结果更加准确.

1 设计思想

基于结构和语义相似度的SQL程序评分模型如图1所示, 分为标准化处理和相似度评估两个部分.

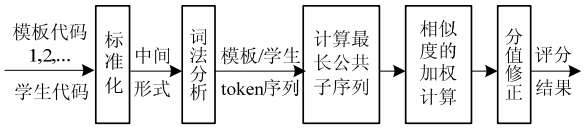

图 1　SQL 程序评分模型

在 SQL 代码中, 除语法规定的关键字, 其他的标识符和表达式具有很大的可变性, 且有多种符合语义的书写方式, 必须进行标准化处理. 通过采用格式规范化、同义词链和基于后缀式的抽象语法树等方法, 将代码中的标识符、表达式和命令子句按一定的语法语义规则转换成统一的中间形式, 以提高词法分析阶段输出的标记串(token)的标准化程度, 从而提高评估结果的准确性.

标准化处理后, 再模拟人工评分的思想, 从程序结构和设计内容两方面进行考查, 分别将 SQL 代码中的命令关键字、标识符、表达式、子查询设置为评分点, 在词法分析时, 以评分点为单元抽取相应的标记, 组成一个评估单元(token)序列. 同时, 还考虑到不同的评估单元在信息体现的重要性上的差别, 通过赋予不同的权重, 对得到的最长公共子序列进行加权处理, 计算模板代码和学生代码的相似度, 使评分过程更加客观公正.

一题多解的情况在 SQL 程序设计中很常见, 为此需要建立多个 SQL 模板程序, 并用每个模板代码依次评估学生代码, 该题的最终得分即为每个结果中的最大值, 这也体现了评分过程的完备性.

2 SQL代码的标准化

在基本的代码格式标准化(如将大写字母转化为小写等)基础上, 通过运用符合语法语义的标准化规则, 可以消除 SQL 代码语义表达的多样性, 提高学生程序和模板程序之间相似度匹配的准确率, 同时也能有效地减少模板程序的数量.

2.1 命令子句和标识符的标准化

2.1.1 命令子句的标准化[11]

对多表的连接查询, 将 Where 子句中的连接条件移入 From 子句, Where 中只保留筛选条件. From 子句可形式化表示为“from <table1>, <table2>, … on <table1>.<field1>=<table2>.<field2>,…”.

2.1.2 标识符的标准化

基于查询使用的基表, 通过分析 From 子句, 可以得到表名、表的别名、连接条件等信息.

① 别名处理: 由于表的别名和字段别名不影响代码执行结果, 所以可以忽略字段别名, 表的别名统一用表名表示.

② 更新标识符: 将所有字段名均表示为“表名.字段名”的形式. 若输出列中使用了“*”号, 则将“*”扩展为基表中的所有字段. 排序项为列的别名或序号时, 均用标准化后的输出列标识符表示, 没有指定排序方向的默认加 asc 关键字.

③ 重排标识符顺序: Select 子句中输出列的顺序和From子句中表的排列顺序不影响查询结果, 可以统一为排序后的顺序. 对连接条件中的同名字段, 也统一为排序后的顺序.

2.2 表达式的标准化

语义等价的表达式可以有多种表达方式, 表达式的标准化处理基于同义词链和抽象语法树.

2.2.1 同义词替换

语义等价的表达式采用不同的实现方法. 如, left(学号, 2)和 substring(学号, 1, 2)语义等价. 对此, 可以建立同义词链, 进行同义词替换. 同义链的形式为“目标表达式:备选表达式 1|备选表达式 2|…”, 如, “left(学号, 2): substring(学号, 1, 2)”, 统一表示为 left(学号, 2).

2.2.2 运算符转换

语义等价的表达式中采用不同的运算符. 如, “in”与“or”运算符语义等价, “between …and”与“>=… and …<=”语义等价. 将包含 in 或 between…and 运算符的表达式转换为逻辑表达式或关系表达式.

2.2.3 表达式中运算符和运算对象位置的标准化

针对运算符的优先级、结合性及所满足的运算律方面的不同而导致的表达多样性, 可以按一定的语法语义规则将表达式转换成中间形式, 从而达到标准化的效果.

(1) 算术表达式的标准化

文献[12]中介绍了 31 条算术表达式转换规则和 22

条布尔表达式转换规则对表达式进行标准化处理. 本文根据 SQL 查询程序中表达式的应用情况, 基于后缀式和抽象语法树, 对表达式进行标准化处理.

第一步, 将中缀式转换为后缀式, 统一表达式中不同优先级运算符的顺序. 后缀式也称为逆波兰式[13], 是一种表达式中间代码形式, 它将运算符写在运算对象的后面, 其形式化表示为“arg1 arg2 op”, 这种表示方式除去了原表达式中的括号, 且运算符的顺序与表达式的实际运算次序相同.

文献[14]中介绍了中缀式转换为后缀式的算法.

第二步, 建立抽象语法树, 规范同级运算符和运算对象的顺序. 本文规定, 同级运算符的排列优先级为: +、- 运算符按先“+”后“-”的顺序, *、/、%(取模)运算符按“*”、“/”、“%”的顺序. 按以下方法调整运算符节点(op)和运算对象节点在树中的位置:

① 依次交换左右子节点、分解右子树、交换左侧上下子节点和右侧上下子节点, 使排序优先级高的 op 节点位于树的左下方, 排序优先级低的 op 节点位于树的右上方.

② 重复上述过程, 直到没有一项操作要做为止.

③ 运算对象按字符串的排序规则排序, 值小的放在运算符的左边. 先比较每个 op 节点下的 2 个叶子节点; 再比较每个 op 节点下的 2 个非叶子节点(或 1 个叶子节点和 1 个非叶子节点), 对非叶子节点比较其左叶子节点的值; 最后比较 2 个或多个相邻且同名的 op 节点下的子节点.

例如, “d*c-a+b”的后缀式为“dc*a-b+”, 依此建立的树结构, 按同级运算符排列优先级调整的结果以及对运算对象排序后的结果如图 2 所示, 最后的标准式为“bcd*+a-”.

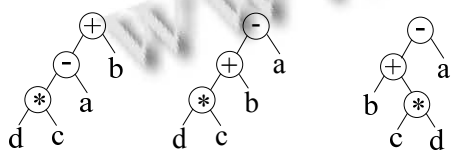

图 2 同级运算符和运算对象的标准化过程

(2) 逻辑表达式的标准化

对于逻辑表达式, 首先分离出其中的算术表达式成分, 然后再对逻辑表达式进行处理, 方法与算术表达式的处理类似. 表达式中包含 Not 运算符且运算对象是关系表达式时, 先执行取反操作, 再将表达式转换为后缀式. 例如, “y>30 and x not between 20 and 50”的标准式为“x<20 x>50 or y>30 and”.

3 基于LCS算法的程序相似度评估

评分模型针对 SQL 语言本身所具有的特征, 模拟人工评分的思路, 划分评分点, 并采用 LCS 算法, 评估学生程序和模板程序的相似度, 同时按评分点的难易度计算分值, 充分体现评分过程的公正性和合理性.

3.1 最长公共子序列

SQL 程序本质上就是一个句子, 本文采用 LCS 算法, 并根据 SQL 程序的结构特征进行适当改进后, 用于 SQL 程序的相似度评估.

$$c[i][j]=\begin{cases}0, & \text{当 } i=0,\ j=0\\ c[i-1][j-1]+1, & \text{当 } i,j>0 \text{ 且 } x_i=y_j\\ \max(c[i][j-1],\ c[i-1][j]), & \text{当 } i,j>0 \text{ 且 } x_i\neq y_j\end{cases} \quad (1)$$

最长公共子序列的长度即为 c[n], 依据该数组回溯, 便可找出最长公共子序列. 对 SQL 语句相似度评估时, 序列中的每一项对应一个评估单元, 最长公共子序列即代表学生程序中正确的部分.

本文的评分模型要同时从结构和语义两方面评估代码相似度, 为适合本文的问题域, 对式(1)做适当修正, 将 2 个数据项相同的概念定义为“值相等并且来自同一个命令子句”, 因此递归式中的条件修改为:

当 $i,j>0$ 且 $x_i=y_j$ 时, 如果 x_i, y_j 来自同一个命令子句, 则 c[i] = c[i-1][j-1] + 1; 如果是来自不同的命令子句, 则 c[i] = max(c[i][j-1], c[i-1]).

不同的评估单元在信息体现的重要性上是有差别的, 因此各评估点的分值是不同的, 评分模型中通过将每个评估单元划分为不同的单元类型, 并赋予不同的权重, 来体现这种差别. 单元类型分为关键字、表达式、标识符、子查询标识符四种, 其中, 关键字是指用来描述程序结构的命令关键字, 如 select、from、where 等; 表达式包括算术表达式和关系表达式, 逻辑表达式拆分为关系表达式和逻辑运算符, 以兼顾评估粒度的大小和信息表达的完整性; 标识符包括表名、字段名、逻辑运算符和程序中的其他关键字(如 distinct、

asc、desc 等); 子查询标识符代表子查询在父查询中的占位符, 用“[?]”表示, 子查询从父查询中分离出来单独评估. 每个类型的权重分别为 w1、w2、w3 和 0.

基于权重的 X 和 Y 两个序列的相似度 sim(X,Y), 可通过式(2)计算得到.

$$sim(X,Y)=(\sum_{i=1}^{3}n_i\times w_i)/(\sum_{i=1}^{3}N_i\times w_i) \quad (2)$$

其中, n_i 和 N_i 分别表示最长公共子序列和模板程序的 token 序列中各单元类型的数量(计算相似度时, 子查询和父查询的结果合并在一起), w 为对应的权重.

对于 2 个给定的字符序列, 最长公共子序列的个数可能有多个, 当基于权重计算相似度时就会导致不同的结果. 但在本文设计的评分模型中, 相似度值是唯一的. select、from、where 等子句中的各项在标准化之后都是按顺序排列的, 如果 group by 和 order by 子句中包含多个数据项, 因数据项未按顺序排列, 这 2 个子句中的匹配结果可能有多组, 但匹配个数是相同的, 且这 2 个子句中的评估单元都可以视作标识符类型, 因此最终得到的相似度值只有一个.

最长公共子序列评估了学生程序中正确的部分, 当学生程序中包含多余子句时, 应该酌情扣除这部分结构性错误的分值. 因此, 每题最后的得分 s, 可通过式(3)计算:

$$s=S*sim(X,Y)-\Delta n*w' \quad (3)$$

其中 S 表示题目的总分值, Δn 为学生程序中多余子句的数量, w'表示扣分权重.

3.2 SQL 程序相似度评估

基于 LCS 算法的评估步骤为:

① 对模板程序进行词法分析, 按语句的自然顺序和句法规定的分隔符划分每个评估单元, 组成一个 token 序列 X, 并统计 X 中各单元类型的数量 N_1, N_2, N_3.

② 对学生程序进行词法分析, 组成评估序列 Y, 并统计关键字类型的数量 $n1'$.

③ 使用修正后的式(1)计算 X 和 Y 的最长公共子序列 Z.

④ 统计 Z 中各单元类型的数量 $n1$, $n2$, $n3$, 通过式(2)计算学生程序的相似度值 sim.

⑤ 计算学生程序中多余的子句数量 $\Delta n=n1'-n1(n1'>n1)$.

⑥ 通过式(3)计算该题的得分 s.

⑦ 若是一题多解, 有多个模板程序, 则重复执行③-⑥, 得到相对于每个模板程序的评分结果, 并取其中的最大值作为实际得分.

假设标准化后的模板程序: select s.name, s.sno from s where s.dept=’CS’order by s.sno asc, 学生程序: select s.sno from s where s.dept=’CS’ order by s. dept asc, 则 X=(select,s.name,s.sno,from,s, where,s. dept = ’CS’, orderby, s.sno,asc), Y=(select,s. sno, from, s, where, s.dept=’CS’,orderby,s.dept,as), Z= (select, s. sno, from, s, where, s.dept=’CS’, orderby, asc) . 若 w1 =2, w2=2, w3=1, S=15, 则 s=13.

对于包含子查询的 SQL 程序, 分别计算父查询和子查询的最长公共子序列, 并将统计的各单元类型数量合并后计算出整个 SQL 代码的相似度.

4 样例测试和分析

为了验证评估模型的实际效果, 选取不同类型的 SQL 程序样例进行测试和分析. 其中, x、y 分别表示模板程序和学生程序, X、Y 分别表示 2 个程序标准化之后的中间形式, sim 表示 X 和 Y 的相似度值. 测试结果如表 1 所示.

表 1　测试分析

示例	相似度	分析	结果说明
例 1	$sim_{x1,y1}$ =1	y1 与 x1 句法结构相同表达方式不同, y1 与 x2 结构不同	From 子句标准化可消除句法表达的不一致
	$sim_{x2,y1}$ =0.43		
	$sim_{x1,y2}$ = 0.56	y2 与 x1 和 x2 的句法结构都不同	x1 使用连接查询, y2 与 x1 结构更相似一些
	$sim_{x2,y2}$= 0.43		
例 2	sim = 0.38	存在句法错误但多表连接正确; 筛选条件缺少一个子查询	①消除 From 子句表达不一致; ②逻辑表达式统一为后缀式; ③分组筛选句法和条件表达是考核重点占较大权重

(1) 例 1: 一题多解的 SQL.

连接查询 x_1: select sno,grade from c inner join sc on c.cno = sc.cno where cname=’java’, X_1: select sc. grade,sc.sno from c,sc on c.cno=sc.cno where c.cname = ’java’

子查询 x_2: select sno, grade from sc where cno=

(select cno from c where cname='java'), X_2: select sc.grade, sc.sno from sc where sc.cno=(select c.cno from c where c.cname='java')

X_2 对应的 token 序列包括父查询和子查询 2 个:

token_1: select, sc.grade, sc.sno, from, sc, where, sc.cno=[?]

token_2: select,c.cno, from,c, where, c.cname= 'java'

有如下 2 个学生程序:

y_1: select sno, grade from sc,c where sc.cno =c.cno and c.cname='java', Y_1: select sc.grade,sc.sno from c,sc on c.cno=sc.cno where c.cname='java'

y_2: select sno, grade from sc where cno='101'

(2) 例 2: 带有连接和嵌套的复杂 SQL.

x: select 职工号, sum(金额) as 销售总金额 from 职工 e join 订购单 o on e.职工号=o.经手人 join 仓库 w on e.仓库号=w.仓库号 where year(订购日期)= year(GETDATE()) or (城市='北京' and 供货方 is not null and 经手人 not in(select distinct 经手人 from 订购单 where 供货方<>'S4')) group by 职工号 having sum(金额)>100000

预处理后, 外层查询语句中忽略列别名"销售总金额", from 子句规范为"仓库,订购,职工", on 子句规范为"订购单.经手人=职工.职工号,仓库.仓库号=职工.仓库号", where 子句规范为"year(getdate())= year(订单.订购日期) 仓库.城市='北京' 订购单.供货方_is_not_null and 订购单.经手人 in(?) not and or".

有如下学生程序:

y: select 职工号, sum(金额) as 总金额 from 职工, 订购单,仓库 where 职工.职工号=订购单.经手人 and 职工.仓库号=仓库.仓库号 and (城市='北京' and 供货方='S4' or year(订购日期)='2016') and sum(金额)> 100000

预处理后, y 中的 where 子句规范为"仓库.城市='北京' 订购单.供货方='S4' and year(订单.订货日期)='2016' or sum(订购单.金额)>100000 and".

从测试结果中可以看出, SQL 程序经过标准化处理后提高了评估结果的准确性; 基于结构和语义的评分方式也充分体现了评分过程的合理性, 如学生程序存在结构性错误时评估的相似度较低, 符合人工评分思想.

5 结语

本文按照一定的语法语义规则, 并结合抽象语法树对 SQL 程序中的命令子句、标识符和表达式进行标准化处理, 消除句法和语义的多样性, 减少模板程序的数量; 采用改进的 LCS 算法, 考查模板程序和学生程序的匹配度, 其中标准化处理是提高评估效率的关键. 该模型能够有效地评估多表查询、子查询、含有复杂条件的查询, 以及一题多解的情况, 贴合人工评分结果, 已在教学中得到实际应用.

参考文献

1 林学民,王炜.集合和字符串的相似度查询.计算机学报, 2011,34(10):1853–1860.

2 吕强,邓薇,宋玲.句子语义相似度计算.计算机工程与应用, 2010,46(36):150–152,229.

3 刘运通,梁燕军.基于分段语义比较的语句相似度计算方法. 计算机工程与设计,2013,34(7):2637–2641.

4 杨鹤标,刘玲,杨立凡.基于结构相似匹配的 SQL 程序自动评估模型研究.计算机工程与科学,2010,32(11):92–96.

5 Petrovskiy M. A data mining approach to learning probabilistic user behavior models from database access log. Proc. of Portuguese Confon Artificial Intelligence, 2003.

6 马培军,王甜甜,苏小红.基于程序理解的编程题自动评分方法.计算机研究与发展,2009,46(7):1136–1142.

7 段利国,陈俊杰.综合句法结构及语义相似度的问题推荐技术.计算机科学,2012,39(1):203–206.

8 屠方博,杨志强.基于语法树和 JavaCC 的程序题自动评分系统.计算机技术与发展,2012,22(1):126–128.

9 Hirschberg DS. Algorithms for the longest common subsequence problem. ACM, 1977, 24(4): 664–675.

10 王红梅.算法设计与分析.北京:清华大学出版社, 2006: 126–128.

11 李海龙,张伟明,肖卫东等.通用标准 SQL 语法分析模型.小型微型计算机系统,2003,24(11):1969–1972.

12 Wang TT, Su XiH, Wang YY, et al. Semantic similarity-based grading of student programs. Information and Software Technolgy, 2007, 49(2): 99–107.

13 张素琴,吕映芝,蒋维杜等.编译原理.北京:清华大学出版社,2005.

14 陈慧南.数据结构——C 语言描述.西安:西安电子科技大学出版社,2009.

15 周松松,马建红.基于 URL 相似度的会话识别方法.计算机系统应用,2014,23(12):191–196.

16 于海英.程序代码相似度度量的研究与实现.计算机工程,2010,36(4):45–46,49.

17 曾波,潘少彬,陆璐.改进的 LCS 方法在测试脚本序列比对中的应用.计算机工程与应用,2011,47(35):71–76.