刻面分类构件的匹配模型*

王渊峰[1+], 薛云皎[1], 张 涌[1], 朱三元[2], 钱乐秋[1]

(复旦大学 计算机科学与工程系 软件工程实验室,上海 200433)
(上海计算机软件中心,上海 200233)

A Matching Model for Software Component Classified in Faceted Scheme

WANG Yuan-Feng[1+], XUE Yun-Jiao[1], ZHANG Yong[1], ZHU San-Yuan[2], QIAN Le-Qiu[1]

[1](Department of Computer Science and Technology, Fudan University, Shanghai 200433, China)
[2](Software Institute of Shanghai, Shanghai 200233, China)
+ Corresponding author: Phn: 86-21-65642826, E-mail: www_yf@sina.com; www_yf@163.com; yfwang@fudan.edu.cn
http://www.fudan.edu.cn
Received 2001-11-08; Accepted 2002-04-18

Wang YF, Xue YJ, Zhang Y, Zhu SY, Qian LQ. A matching model for software component classified in faceted scheme. *Journal of Software*, 2003,14(3):401~408.

Abstract: As the component repositories scaling up and the reuse practice deepening, representing and retrieving software components gains more attention from software engineering researchers. A matching model is proposed in this paper to retrieve reusable components classified in faceted scheme includes three levels and five schemes. A generic matching algorithm is provided and analysis and test are done on its specialized implement and relevant time complexity. The results show that this solution is feasible and effective for the component retrieval.
Key words: component repository; component retrieval; faceted classification; tree matching; software reuse

摘 要: 随着软件复用实践的深入和软件构件库规模的扩大,对软件构件的表示与检索的研究正得到越来越多的重视.针对基于刻面描述的软件构件,结合模式分析中的树匹配思想,并根据构件刻面描述的特点,提出了一个包含3个层次,5种匹配类型的刻面匹配模型.给出了该匹配模型的泛型算法并对具化情况下的算法实现与时间复杂度进行了讨论.同时,通过理论和实践的结果证明了该匹配模型在构件检索上的可行性与有效性.
关键词: 构件库;构件检索;刻面分类方案;树匹配;软件复用
中图法分类号: TP311 **文献标识码**: A

构件的表示与检索技术是可复用软件构件库的两个主要的核心技术.其中,软件构件的刻面表示以及在此基础上的构件检索技术已得到软件复用界的重视与应用.例如,REBOOT,NATO 都提出了各自的可复用软件构件的分类方案[1,2].青鸟构件库所采用的也是以刻面分类为主、多种分类模式相结合的方法,对构件进行分类描述[3].

* Supported by the Development Foundation for Key Disciplines of Shanghai Education Commission of China under Grant No.B990105 (上海市教委重点学科建设项目)

第一作者简介: 王渊峰(1974－),男,上海人,博士,主要研究领域为软件复用,构件库系统.

目前,对于刻面描述的构件检索主要采用的还是以传统的数据库检索技术为主并结合利用同义词词典和刻面术语空间的层次结构来实现构件的松弛匹配[1~4].但是,构件库的检索与一般的数据库或文献库中的检索不同.首先,构件的检索需要一定的松弛匹配的能力,在保证一定查准率的情况下提高查全率.另外,构件的检索需要兼顾对查询的构件的不完全描述,对查询的匹配应有一定的模糊能力,不仅要求能给用户返回匹配的结果,还要求能返回相应的匹配程度,为用户复用构件提供有意义的参考.另外,各个构件库的刻面分类方案可能完全不一样.有时,为了查到合适的构件,用户查询可能需要跨越多个构件库,如何在匹配时有效地屏蔽不同的构件刻面分类方案间的差异,这也是刻面匹配所需要解决的问题.

采用目前的刻面描述匹配技术来解决这些要求还存在许多值得改进的方面.Forbes Gibb 在他们的可复用软件构件研究项目[5]中引进了 XML 作为构件刻面描述的标记语言,并应用 XML 检索语言 XML-SQL 来完成构件检索的任务.但是,在对构件刻面描述的匹配技术的设计上还存在着一定的改进空间.

本文将树匹配的方法结合进构件刻面描述的匹配,提出了一个包含 3 个层次、5 种匹配类型的可复用构件刻面描述的匹配模型,同时给出了该模型的泛型匹配算法,并对其各种匹配类型下的具化匹配算法的复杂度进行了讨论.本文第 1 节介绍树匹配的一些基本概念和主要思想.第 2 节提出一个层次丰富的构件刻面描述的匹配模型.第 3 节给出上述模型的泛型匹配算法,并对其在各种具化情况下的匹配算法的实现以及时间复杂度作了讨论.第 4 节和第 5 节给出相关的实验数据、下一步的工作以及对全文的总结.

1 树匹配的基本概念

1.1 基本概念

一个无环连通图称为树,用 T 表示:$T=(V,E,root(T))$.其中 V 表示一个有限的节点集,$root(T)\in V$ 表示树的根节点,E 表示边集,它是 V 上的一个二元关系,它满足反自反、反对称、可传递性.如果$(u,v)\in E$,则称 u 节点是 v 节点的父节点,记为 $u=parent(v)$或 $v=child(u)$,u 节点的所有儿子节点组成的集合记为 $C(u)$.一棵树必须满足以下 3 个条件:

(1) 根节点不存在父节点;

(2) 除根节点以外,V 中其他节点都有且仅有一个父节点;

(3) 树中每一个非根节点 $v\in V$,存在$(root(V),v)\in E^*$,其中 E^*是 E 的传递闭包.

如果两个节点 v_1,v_2,有$(v_1,v_2)\in E^*$,则称 v_1 是 v_2 的祖先节点,记为 $v_1=ancestor(v_2)$或 $v_2=descendan(v_1)$.将所有 u 节点的祖先节点组成的节点集合记为 $A(u)$,所有 u 节点的子孙节点组成的节点集合记为 $D(u)$.设 $u\in V$,则 T 中以 u 为根节点的子树记为 $T[u]=(V',E',u)$, 其中 $V'=\{u\}\cup D(u)$, $E'=E\cap(V'\times V')$.

如果两个节点 v_1,v_2 有 $(v_1,v_2)\notin E^*\wedge(v_2,v_1)\notin E^*$, 则用下面的亲和度的概念来描述 v_1,v_2 之间的关系.

定义 1.1(宗祖). 设 $v_1,v_2\in T\wedge(v_1,v_2)\notin E^*\wedge(v_2,v_1)\notin E^*$, 则定义集合 A_{v_1,v_2} 为 v_1,v_2 的宗祖, $A_{v_1,v_2}=\{a\mid a=ancestor(v_1)\wedge a=ancestor(v_2)\}$.

定义 1.2(亲和度). 设 $v_1,v_2\in T\wedge(v_1,v_2)\notin E^*\wedge(v_2,v_1)\notin E^*$, 则定义 R_{v_1,v_2} 为 v_1,v_2 的亲和度: $R_{v_1,v_2}=|A_{v_1,v_2}|$, 其中||表示集合的势.

值得注意的是:亲和度只是两个无祖先后辈关系的节点之间关系的一个简单的度量,它的值只有在存在共同祖先节点时,比较才有意义,否则是无意义的.例如:如果 $R_{v_1,v_2}>R_{v_1,v_3}$,可以认为 v_2 与 v_1 的关系比 v_3 与 v_1 的关系要近,但如果 $R_{v_1,v_2}>R_{v_3,v_4}$,则不能认为 v_2 与 v_1 的关系比 v_3 与 v_4 的关系要近,因为这时还要考虑节点的深度,单纯地依靠亲和度值是不能下结论的.

1.2 编辑操作

为了对构件的刻面描述以及刻面匹配进行建模,先介绍树匹配中的有关概念.首先是树的编辑操作的定义,树的编辑操作主要有删除(deletion)、插入(insertion)和改变标签(relabel)3 种,依次介绍如下,其中 T 表示编辑操作以前的树 $T=(V,E,Root(T))$,T'表示编辑操作以后的结果树 $T'=(V',E',Root(T'))$.

- 删除节点 v(记为 delete(v)):

当 $v\neq root(T)$时: $V'=V-\{v\}$, $E'=E-\{(u,v)|u=parent(v)\}+\{(u,v_c)|u=parent(v)\wedge v_c\in C(v)\}$.

当 $v=root(T)$时:删除 v 节点后的结果是一个森林 $\{T[v_c]|v_c\in C(v)\}$.

- 在节点 u 下插入节点 v(记为 $\text{insert}_u(v)$):

$$V'=V+\{v\},\ E'=E+\{(u,v)\}+\{(v,u_c)|u_c\in C'(u)\}-\{(u,u_c)|u_c\in C'(u)\},$$

其中 $C'(u)\subseteq C(u)$. $C'(u)$ 具体包含了 $C(u)$ 的哪些元素并不重要,它的选择是由在执行操作时的具体上下文环境决定的,这在后文中定义编辑操作序列中可以看出.

- 改变 v 节点的标签(relabel(v)):

这个编辑操作是标签树(labeled tree)所特有的.标签树是一个四元组(V,E,$root(T)$,f),其中 f 将一棵树中的节点集单射到一个标签集(label set),一个标签相当于一个字符串.例如,在后文中的构件的刻面描述树就是一棵标签树,标签集为刻面名集合和刻面术语集合的并.树中某个节点 v 的标签用 $label(v)$表示.改变 v 节点的标签即是对 v 节点的标签值赋予新的标签值.

树 T 的每一个编辑操作 OP 有一个对应的实数值,定义为编辑代价,用 $\gamma(OP)$表示,编辑代价是编辑操作的操作类型和操作节点的函数.如果 $OP=Op_1,\ldots,Op_k$ 为一个编辑操作序列,则定义该序列的操作代价为 $\gamma(OP)=\sum_{i=1}^{i=|OP|}\gamma(Op_i)$. 如果对一棵树 T_1 执行一个编辑操作序列 OP,将其转化成另一棵树 T_2,则称 OP 为 T_1 到 T_2 的一个编辑操作序列,定义两棵树之间的编辑距离为

$$\gamma(T_1,T_2)=\min\{\gamma(OP)|OP\text{ 是一个 }T_1\text{ 到 }T_2\text{ 的编辑操作序列}\}.$$

2 刻面描述的匹配模型

2.1 刻面描述及其匹配的树建模

对一个刻面分类方案,我们将其中的刻面、子刻面分别映射为树中对应的父节点、子节点,对采用某个刻面分类方案描述的构件,可以将其对应的描述术语映射为对应的叶子节点.例如,Prieto-Diaz 最早所提出的刻面分类方案[4]为两个主刻面:“功能”和“环境”,其下各有 3 个子刻面(作用、对象、媒介)和(系统类型、应用领域、客户类型),用在该刻面分类方案下描述的某个构件可以用如下刻面树来表示(其中根节点是一个虚拟节点):

对于构件的查询也可相应地表示为一棵查询树,将查询中出现的刻面名、子刻面名转化为相应层次的父子节点,并将待查询的刻面术语值映射为叶节点(我们将在其他文章中介绍加入布尔查询符后扩展的查询树的表示方法及相关算法的改动).

于是,构件的检索可以转化为查询树与库中每个构件的刻面描述树之间的匹配,两棵树之间的匹配本质上是两棵树的节点之间的一个映射.根据对该映射所施加的约束条件的不同,可以得到不同类型的匹配.如图 1 所示,其中根节点是一个虚拟节点.

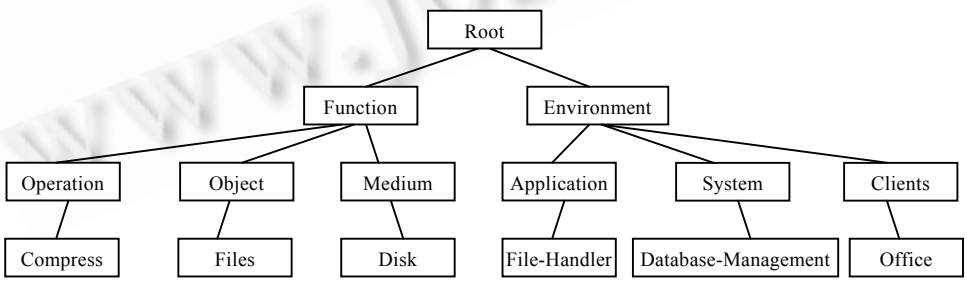

Fig.1 An example of faceted tree

图 1 一例构件刻面描述树

2.2 构件刻面描述的树匹配模型

在下面的论述中,假设 Q 为一棵查询树,Q_{sub} 是 Q 的节点集的一个子集.T 为一棵构件的刻面描述树,T_{sub} 是 T 的节点集的一个子集.

定义 2.1(子树匹配(*Ms*)). 如果存在 Q_{sub} 到 T_{sub} 的一个映射 f,满足以下 3 个条件,则称该映射 f 是 Q 到 T 的一个子树匹配.

(1) $v_1=v_2$, iff $f(v_1)=f(v_2)$ v_1,v_2 是 Q_{sub} 中的节点(表示 f 是一个单射);

(2) $label(v_1)\approx label(f(v_1))$ ≈表示两个标签之间的近似距离在阈值范围之内;

(3) $v_1=parent(v_2)$, iff $f(v_1)=parent(f(v_2))$;

(4) $|C(v_1)|=|C(f(v_1))|$.

子树匹配相当于 Q_{sub} 到 T 的子树的一个同构关系.如图 2 所示的 Q 到 T_1 的匹配(图中节点内字母相同的节点表示一个映射对).

定义 2.2(区域匹配(*Mr*)). 如果存在 Q_{sub} 到 T_{sub} 的一个映射 f 满足以下 3 个条件,则称该映射 f 是 Q 到 T 的一个区域匹配.

(1) $v_1=v_2$, iff $f(v_1)=f(v_2)$;

(2) $label(v_1)\approx label(f(v_1))$;

(3) $v_1=parent(v_2)$, iff $f(v_1)=parent(f(v_2))$.

区域匹配相当于 Q_{sub} 到 T_{sub} 的一个同构关系.如图 2 所示的 Q 到 T_2 的匹配.

定义 2.3(包容匹配(*Me*)). 如果存在 Q_{sub} 到 T_{sub} 的一个映射 f 满足以下 3 个条件,则称该映射 f 是 Q 到 T 的包容匹配.

(1) $v_1=v_2$, iff $f(v_1)=f(v_2)$;

(2) $label(v_1)\approx label(f(v_1))$;

(3) $v_1=ancestor(v_2)$, iff $f(v_1)=ancestor(f(v_2))$.

包容匹配的例子可见图 2 中的 Q 到 T_5 的匹配.在包容匹配的基础上,我们可以定义以下带约束条件的包容匹配形式.

定义 2.4(强约束的包容匹配(*Mse*)). 如果 f 是一个 Q 到 T 的包容匹配,并且满足:如果 v_1,v_2,v_3 彼此之间不存在祖先、子孙关系,有 $R_{v_1,v_2}=R_{v_1,v_3}$ iff $R_{f(v_1),f(v_2)}=R_{f(v_1),f(v_3)}$,则称 f 是 Q 到 T 的一个强约束的包容匹配(如图 2 所示的 Q 到 T_3 的匹配.)

定义 2.5(弱约束的包容匹配(*Mle*)). 如果 f 是一个 Q 到 T 的包容匹配,并且满足:如果 v_1,v_2,v_3 彼此之间不存在祖先、子孙关系,有 $R_{v_1,v_2}<R_{v_1,v_3}\Rightarrow R_{f(v_1),f(v_2)}\le R_{f(v_1),f(v_3)}$,则称 f 是 Q 到 T 的一个弱约束的包容匹配(如图 2 所示的 Q 到 T_4 的匹配).

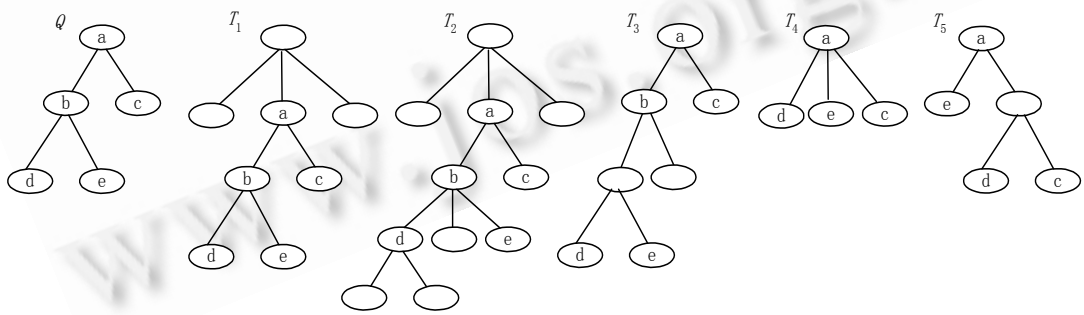

Fig.2 Examples for matching model

图 2 匹配模型示例

上述 5 种匹配之间的关系为 *Ms*→*Mr*→*Mse*→*Mle*→*Me*.箭头指向的是匹配条件更为宽松的匹配.这些匹配的映射约束条件纵向从父子关系放松到祖先子孙关系,横向从孩子数的约束放宽到亲和度的约束直至放宽到无约束.这 5 种匹配构成了一个层次丰富("子树"、"区域"和"包含"3 个层次)的构件刻面描述的匹配模型.其中,*Ms* 体现的是查询信息及其结构的基本完整包含,即相当于"专著"(即不多信息也不少信息)的匹配.*Mr* 则只是体现了查询信息及其结构基本完整包含但可能不是"专著"的匹配.*Me* 则只是匹配查询中的信息部分,对其结

构作了一定的放松,即在横向上允许刻面间的信息的交错或刻面的信息的分解,而在纵向上只要保持节点间的祖先、子孙关系.*Mse* 相对于 *Me* 在结构上作了稍微严格的要求,即要求查询中同一刻面下的信息只能与构件中同一刻面下的信息进行匹配,不再允许刻面间信息的交错以及刻面间信息的分解.而 *Mle* 放宽了 *Mse* 的要求,相当于允许在匹配中出现刻面分解的情况,但仍旧不允许刻面之间信息的交错.

2.3 匹配代价

对上述任何一种类型的匹配都存在一个匹配代价的计算问题.从查询树 Q 到构件刻面描述树 T 的一个匹配过程可以看成是一个从树 Q 到树 T 的编辑操作序列.该序列先将未出现在匹配映射的定义域 Q_{sub} 中的 Q 中的节点从 Q 中删除,再对 Q_{sub} 中的节点进行改变标签的编辑操作,使之变换到 T_{sub} 中相应的节点的标签.最后将 T 中未出现在 T_{sub} 中的节点依次插入到 T_{sub} 中的相应位置.设该编辑序列为 OP_M,根据编辑操作序列的代价的定义,有

$$\gamma_{Q\to T}(OP_M)=\sum_{i\in Q-Q_{\text{sub}}}\gamma(delete(i))+\sum_{i\in Q_{\text{sub}}}\gamma(relabel(i))+\sum_{j\in T-T_{\text{sub}}}\gamma(insert_{parent(j)}(j)). \tag{1}$$

定义上式中的 $\gamma_{Q\to T}(OP_M)$为匹配 M 的匹配代价.

对于 Q 到 T 的每一种类型的匹配,定义相应的匹配代价为

$$\gamma_{Q-T}(M_i)=\min\{\gamma(M)\mid M \text{ 是 } Q \text{ 到 } T \text{ 的一个属于类型 } i \text{ 的匹配}\}.$$

根据第 2.2 节的定义,i 代表子树匹配、区域匹配、包容匹配、强包容匹配、弱包容匹配 5 种匹配类型中的一种.

综上所述,第 2.2 节中介绍的 5 个匹配条件可以微调的匹配类型构成了一个层次丰富的构件刻面描述的匹配模型,并且针对每一种匹配类型还可以通过计算匹配代价在一定程度上定量地比较匹配的相似程度.通过对匹配类型的选择和匹配代价的计算可以在一定程度上满足查询用户对刻面描述的软件构件的检索需求.

3 刻面树的匹配算法

3.1 泛型匹配算法

本节将给出一个计算各种类型的匹配代价的泛型算法.该算法由以下的两个主要步骤组成.图 3 是该泛型算法的一个概要.

算法 1. GenericMatching 计算匹配代价的泛型算法.

输入:$Q=(V,E,root(Q)),T=(W,F,root(T)),M_i$.

输出:$\gamma_{Q\to T}(M_i)$.

```
1  for(p=1; p≤|V|; p++)
2  {
3     M(p)=∅;
4     for(t=1; t≤|F|; t++)
5     {
6        if(label(p)≈label(t))
7        {
8           put t into M(p);
9        }
10    }
11 }
12 minCost=+∞;
13 do
14 {
15    Select a subSet Q_sub of Q;
16    do
17    {
18       Select a subSet T_sub of T from ∏_{q_i∈Q_sub} M(q_i);
19       if(Validate(Q_sub→T_sub, M_i))
20       {
21          Counting γ_{Q→T}(Q_sub→T_sub);
22          minCost=min(minCost, γ_{Q→T}(E_{M_i}))
23       }
24    }while(Try out all possible T_sub)
25 }while(Try out all possible Q_sub of Q)
```

首先,该算法将树 T 和树 Q 中的节点按自左向右的后序顺序编号,并按编号值的升序来遍历这两棵树中的所有节点(1~11 行).在遍历的过程中将每一个与 p 节点($p\in Q$)的标签具有近似关系的 t 节点($t\in T$)记录进集合 $M[p]$中.

然后,算法枚举所有可能的 Q_{sub}(Q_{sub} 表示 Q 的一个节点子集)和 T_{sub}($T_{sub} \in \prod_{q_i \in Q_{sub}} M(q_i)$)(12~23 行).

其中,Q_{sub} 和 T_{sub} 的枚举具有一定的规律(15,18 行),具体规律将在后面加以介绍.

对枚举出来的一对 Q_{sub} 和 T_{sub},首先判断 $Q_{sub} \rightarrow T_{sub}$ 的映射是否符合 M_i(M 表示 5 种匹配类型的某一种)的条件(19 行)(根据定义 2.1~定义 2.5).

对匹配条件成立的映射再计算相应的匹配代价(21 行,根据公式(1)).

在整个处理过程中,min*Cost* 记录了符合 M_i 匹配条件的最小的匹配代价.

3.2 具化匹配算法的讨论

在考虑某种具体的匹配类型的匹配算法的设计时,如果对 Q_{sub} 的枚举策略和对基于 $\prod_{q_i \in Q_{sub}} M(q_i)$ 的 T_{sub} 的枚举策略(15,18 行)根据具体的匹配类型进行具体设计,将得到很好的效果,改善算法的时间复杂度.讨论如下:

对于子树匹配和区域匹配,Pekka Kilpelainen 在其博士论文[6]中提出了可行的枚举策略.方法的主要思想是:如果一个节点对(p,t)属于匹配,并且 p 的儿子节点也参与了匹配,那么 p 的儿子节点必然匹配到 t 的儿子节点.利用这一性质可以大大减少算法的搜索空间.对于子树匹配,算法的时间复杂度目前达到了 $O(m \cdot n)$.对于区域匹配,算法的时间复杂度目前达到了 $O(m^2 \cdot n)$.其中,m 为 Q 中节点的个数,n 为 T 中节点的个数.算法的具体设计可参见文献[6],此不赘述.

包容匹配是树匹配研究领域的经典问题之一.目前已经证明,它的算法难度为 NP 完全问题[7,8].对其具化算法的研究可以参见文献[9],算法的主要思想是:如果一个节点对(p,t)属于匹配,并且 p 的儿子节点也参与了匹配,那么 p 的儿子节点则必定匹配到 t 的子孙节点.根据这一性质,文献[10]提出了一种自顶而下的改进匹配算法,时间的复杂度为 $O(m \cdot n^{degree(Q)} \cdot degree(Q)^2)$.

强约束包容匹配是针对刻面分类方案的特性提出的包容匹配的一种约束匹配形式.它要求查询树中一个刻面域下的内容在匹配时只能作为整体映射到构件刻面描述树对应的某个刻面域下,反之亦然.换句话说,即不同的刻面域的内容在匹配时不能交错重叠.也不允许在匹配时出现刻面域分解的情况,即一个刻面域下的内容在匹配时不能分解成两个或两个以上的刻面域的内容.

弱约束包容匹配是使用较多的一种刻面查询模式,它在强约束的包容匹配的基础上放宽了对匹配时刻面域的内容不能分解的限制.可以发现,强约束包容匹配算法比较类似于区域匹配算法,而弱约束包容匹配算法则更靠近包容匹配算法.这从算法的复杂度上也可以看出,弱约束包容匹配已被证明为属于 NP 完全问题,而强约束包容匹配算法的复杂度证明为 $O(m \cdot n \cdot (degree(Q)+degree(T)) \cdot (\log_2(degree(Q)+degree(T))))$.在算法的实现中,它们都利用了树的层次结构和匹配映射的约束的特性,进行了自底向上的最小匹配代价所对应的匹配映射的构造.算法的思想分别和区域匹配及包容匹配类似,但是增添了更多的内部的数据结构来记录匹配中间阶段的数据,以指导下一步匹配空间的搜索,避免不必要的枚举,用空间来换时间.整个证明过程和算法的具体实现可参见文献[11].

4 实验与下一步的工作

我们设计了一个构件库检索的原型系统 RCRS(reusable component retrieval system)(VC++6.0,SQL Sever2000),实现了上述的匹配思想.该构件库使用青鸟构件的刻面分类方案,对 MFC,STL 和 VCL 中选取出来的 146 个构件进行了刻面描述.

实验设计为:首先,为参与实验的人员(10 位)提供所要求查找的构件(10 个)的一段文字描述.由他们使用原型系统提供的构件检索手段在库中进行查找,将他们所查找到的构件集合以及查找时所使用的匹配类型进行统计.实验得到以下的结论:

- 无论哪种匹配类型,都能够在查找的结果集的前 3 位(按最小匹配代价升序排序)出现正确的所要查找的构件.其中在第 1 位出现的情况是 99%(仅有一次在包容匹配查找中,所要查找的正确的构件集出现在检索的结果集中的第 2 位).这说明树匹配应用在构件检索时的查准率和查全率是可行的,并说明 $Me \rightarrow Mle \rightarrow Mse \rightarrow Mr \rightarrow$

Ms 是一个匹配精度依次精确的匹配过程.

- 对匹配类型的使用情况的统计发现,如果实验主体熟悉青鸟刻面分类方案,则会选择使用子树匹配;如果实验主体不熟悉青鸟刻面分类方案但是如果是从事软件工作的人员,则会选择使用区域匹配或强约束的包容匹配.如果是非软件工作的人员并且是不熟悉青鸟刻面分类方案的用户,则一般会选择使用弱约束的包容匹配或包容匹配.这说明每一种匹配类型都有它所适用的用户和环境,不存在谁取代谁的问题.
- 对每种匹配类型的响应时间数据统计,如图 3 所示.

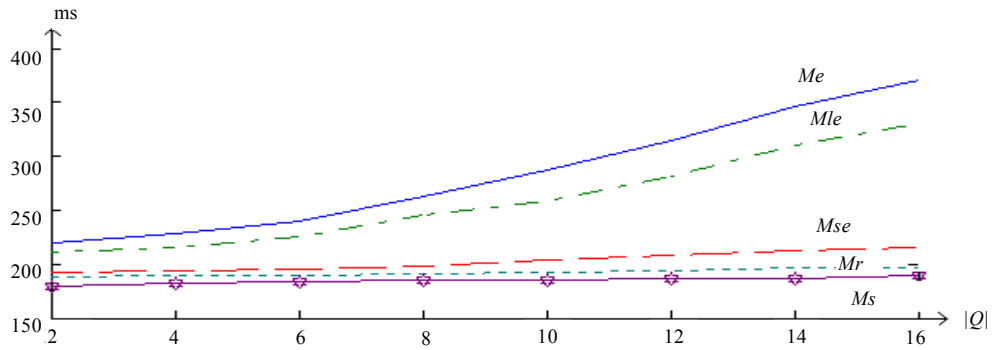

Fig.3 Efficiency comparison of every specific matching algorithm

图 3 各种具化匹配算法的效率比较图

从图中可以看出,完成匹配所需要的时间从 *Ms* 到 *Me* 依次增加,这与第 3.2 节对匹配具化算法的复杂度的分析结论是一致的.同时,我们还发现即使是 *Me*,它所需要的时间最多也只在 380ms 左右(16 个节点),这种新的检索方法在效率上是可行的.

综上所述,*Me*→*Mle*→*Mse*→*Mr*→*Ms* 是一个查准率微调上升,查全率有选择地牺牲的过程.查询用户可以根据自己的需求、知识背景和对查询效率的要求从这 5 种匹配中做出合适的选择来完成构件的检索任务.

目前,对树匹配(*Me* 和 *Mle*)代价的算法还只考虑了枚举算法,为了进一步提高它们的效率,我们正在实验模拟退火和遗传算法等方法.并且,为了更加有效地使用树匹配的检索算法,还可以根据各个构件刻面描述树之间的树匹配的偏序关系来设计构件库中构件存储的逻辑体系结构,以提高检索的效率.另外,对各种树编辑操作代价的确定也需要进行有效的精确化工作,以提高树匹配的精确程度.这些是我们下一步所要研究的 3 个主要方向.

5 总 结

本文针对基于刻面描述的构件,借鉴树匹配中的思想,并结合构件刻面描述方法本身的特征,提出了一个包含 3 个层次、5 种匹配类型的查询树与刻面描述树之间的匹配模型,并给出了该匹配模型的泛型算法及对具化情况下算法的讨论.这些匹配思想的实践与完善为构件的检索提供了新的、有益的思路与探索.

References:

[1] NEC Software Engineering Laboratory. NATO standard for management of a reusable software component library. Volume 2, Tokyo, NATO Communications and Information Systems Agency, 1991. 32~43.

[2] Morel JM, Faget J. The REBOOT Environment. In: Prieto-Diaz R, Frakes WB, eds. Proceedings of the 2nd International Workshop on Software Reusability Advances in Software. Lucca: IEEE Computer Society Press, 1993. 80~88.

[3] Chang JC, Li KQ, Guo LF, Mei H, Yang FQ. Representing and retrieving reusable software components in JB (Jadebird) System. **Electronica Journal,** 2000,28(8):20~24 (in Chinese with English Abstract).

[4] Prieto-Diaz R. Implementing faceted classification for software reuse. Communications of the ACM, 1991,34(5):88~97.

[5] Gibb K, McCartan C, O'Donnell R, Sweeney N, Leon R. The integration of information retrieval techniques within a software reuse environment. Journal of Information Science, 2000,26(4):520~539.

[6] Kilpelainen P. Trees matching problems with applications to structured text database [Ph.D. Thesis]. Department of Computer Science, University of Helsinki, 1992.

[7] Zhong KZ, Tao J. Some MAX SNP-hard results concerning unordered labeled trees. Information Processing Letters, 1994,49(5): 249~254.

[8] Zhong KZ, Statman R, Shassha D. On the editing distance between unorded labeled trees. Information Processing Letters, 1992,42(3):133~139.

[9] Shasha D, Tsong J, Wang L. Exact and approximate algorithm for unordered tree matching. IEEE Transactions on Systems Man and Cybernetics, 1994,24(4):668~678.

[10] Schlieder T. ApproXQL: design and implementation of an approximate pattern matching language for XML. Technical Report, B 01-02, Berlin: Freie Universität, 2001.

[11] Wang YF. Research on retrieving reusable components classified in faceted scheme [Ph.D. Thesis]. Shanghai: Fudan University, 2002 (in Chinese with English Abstract).

附中文参考文献:

[3] 常继传,李克勤,郭立峰,梅宏,杨芙清.青鸟系统中可复用软件构件的表示与查询.电子学报,2000,28(8):20~24.

[11] 王渊峰.基于刻面描述的构件检索算法研究[博士学位论文].上海:复旦大学,2002.

第 20 届全国数据库学术会议(NDBC2003)

征 文 通 知

由中国计算机学会数据库专业委员会(www.ccf-dbs.org.cn)主办、中国人民解放军国防科学技术大学(www.nudt.edu.cn)承办的第 20 届全国数据库学术会议将于 2003 年 10 月 10 日~12 日在湖南召开。会议网址为 http://ndbc2003.nudt.edu.cn/。

一、征文范围(但不限于这些领域)

A. Web 与数据库
B. 移动计算和数据库
C. 数据库实现技术
D. 数据库安全性
E. 数据仓库和 OLAP
F. 数据挖掘和知识发现
G. 文本数据库与信息检索
H. 查询处理与用户界面
I. 生物与基因信息系统
J. 数字图书馆
K. 数据集成和迁移
L. 面向对象与对象-关系数据库系统
M. 中间件和应用服务器技术
N. 实时数据库系统
O. 科学与统计数据库
P. XML 和半结构化数据库系统
Q. 空间和时态数据库系统
R. 内容管理/知识管理
S. 工作流/数据库应用
T. 并行和分布式数据库系统
U. 多媒体数据库技术
V. 电子商务/电子政务

二、重要日期

论文提交截止时间：2003 年 5 月 10 日
论文录用通知时间：2003 年 7 月 20 日
排版稿件截止时间：2003 年 8 月 10 日

三、会议咨询

会议信息可以通过访问网站 http://ndbc2003.nudt.edu.cn/得到，也可以与会务组联系。

E-mail: ndbc2003@nudt.edu.cn

电话：86–731–4552710（NDBC2003 会务组：梁妍女士、隋品波女士）

传真：86–731–4512504

通讯地址：湖南省长沙市国防科学技术大学计算机学院 NDBC2003 会务组（邮编 410073）