试论形式概念分析在软件维护中的应用

张献超

作业	分数
得分	

2020 年 11 月 13 日

试论形式概念分析在软件维护中的应用

张献超

(大连海事大学 信息科学技术学院，大连 116026)

摘 要 形式概念分析（ FormalConcept Analysis,FCA ）理论是层次化的形式对象分析方法，通过对数据集中对象和属性之间的二元关系建立概念层次结构，生动简洁地体现了概念之间的泛化和特化关系，再运用格代数理论对数据进行分析．近十几年来，形式概念分析技术已在软件工程领域，特别是软件维护的各项活动中得到了广泛的应用，并取得成功。．本文从软件理解、修改影响分析、重构、调试与测试等软件维护方面总结了形式概念分析在这些领域的研究进展．这些研究成果的分类方法是基于一种软件维护活动框架进行论述，最后文章给出了形式概念分析在软件维护领域的研究趋势与展望。

关键词：形式概念分析；软件维护

中图法分类号：TP18 文献标识码：A

Application of formal Conceptual Analysis in Software maintenance

Zhang Xianchao

(School of Information Science and Technology, Dalian maritime university, Dalian 116026)

Abstract formal conceptual analysis (FormalConcept Analysis,FCA) theory is a hierarchical formal object analysis method. It vividly and succinctly reflects the generalization and specialization of concepts, and then uses lattice algebra theory to analyze data. In the last ten years, formal conceptual analysis technology has been used in the field of software engineering. In particular, software maintenance activities have been widely used, and have been successful. Software dimensions such as software understanding, modifying impact analysis, refactoring, debugging and testing are discussed in this paper The research progress of formal conceptual analysis in these fields is summarized. The classification of these research results is based on a framework of software maintenance activities. Finally, the research trend and prospect of formal concept analysis in software maintenance field are given.

Keywords formal conceptual analysis; software maintenance

1 引言

形式概念分析概述。形式概念分析由德国的著名学者 Wille 首先提出，形式概念分析主要是用于概念的发现和排序，以及显示。形式概念分析时概念的外延可以定义为属于概念所有对象的集合。概念的内涵是指全部对象拥有共同的特征或属性，这样就能实现概念的形式化。所有概念结合它们之间的泛化关系组成概念格。形式概念分析中的概念格模型是其的核心数据结构，它将根据二元关系进行概念层次结构的建立。并能够反映对象和属性之间的泛化关系，通过此能够很好的在概念层次上建立数据之间的依赖和因果关系的模型。形式概念分析时需要首先进行形式背景的熟悉，数据集能够以形式背景给出。

形式概念分析提供了一种较好的层次化(形式)对象的分析方法，它能够识别那些具有共同(形式)属性的一组(形式)对象的组合[1]．一方面，形式概念分析已具备较完善的理论基础，在应用到软件工程，特别是软件维护活动中时，具有形式化方面的表达能力，从而具有较强的理论和技术说服力；

另一方面，形式对象和形式属性这种二元关系经常出现在软件世界中，对这种二元关系处理的方法也推动形式概念分析技术在软件领域应用的不断发展．形式概念分析技术能够有效地对已有代码和文档进行建模和分析，生成一种图形化的格，提高软件代码和文档的抽象层次，从而辅助软件维护和测试人员对已有代码和文档的理解．因此，目前形式概念分析在软件工程中的各种成功的应用主要是在软件维护的诸多活动中，包括逆向工程、重构、程序理解等[2,3]。本文主要对这些研究工作进行介绍。

2 基本概念

形式概念分析提供一种聚类具有共同形式属性的形式对象的方法，其输入是形式背景，输出是具有偏序(层次)关系的概念格．本文为区分形式概念中的对象、属性与软件中对象以及属性，将形式概念分析中的对象称为形式对象，而属性称为形式属性．一些相关定义表示如下[1]：

定义 1 形式背景 通常定义为一个三元组（O，A，R），其中 O 是形式对象集合，A 是形式属性集合，R 是 O 和 A 之间的二元关系，满足 $R \subseteq \mathrm{O} \times A$

形式概念分析中另外一个重要的概念是形式概念．给定某个形式背景后，只需满足一定的条件就可得到该形式背景上的形式概念，定义如下：

定义 2 形式概念 形式概念表示具有共同形式属性的形式对象的最大集合，定义为二元组(x，y)，并且满足这样的关系：

$\tau(A) = X \wedge \sigma(O) = Y$ 其中：

$$\tau(A) = \{o \in \mathrm{O} \mid \forall a \in Y : (o,a) \in R\}$$

$$\sigma(O) = \{a \in A \mid \forall o \in X : (o,a) \in R\}$$

x 称为形式概念的外延(Extent)；而 y 称为形式概念的内涵(Intent)．

形式概念之间通常存在着一种层次(偏序)关系，我们通常将这种层次关系定义为概念与子概念的关系．子概念的定义如下：

定义 3 子概念 一个概念 $C_1(X_1, Y_1)$是另一个概念 $C_2(X_2, Y_2)$的子概念($C_1 \leqslant C_2$)，满足：

$$C_1 \le C_2 \Leftrightarrow X_1 \subseteq X_2 \Leftrightarrow Y_2 \subseteq Y_1$$

子概念的关系为构造概念格提供了一种自然的偏序关系．Birkhoff 早在 1940 年就证明这样的概念格是一种完全格[2]，定义如下：

定义 4 概念格(Concept Lattice)记为 L(C)：

$$L(C) = \{(O,A) \in 2^{\mathrm{O}} \times 2^{A} \mid O = X \wedge A = Y\}$$

根据这些基本定义，可以得到形式概念分析的一些特征，可根据这些特征进行实际应用，这些特征包括：①形式概念决定了具有共同形式属性的一组最大的形式对象的集合；②概念格显示了一种形式对象以及形式属性的层次分类；③概念格中的某个格与子格关系可以说明一些属性是某抽象属性的不同实例．

3 一个简单的软件维护活动框架

软件维护在软件演化过程中必不可少．软件维护通常表示在系统部署之后对系统所进行的修改．基本的软件维护过程应包括如下几个活动：与修改请求相关的软件理解，修改影响分析，修改实施(包括重构以及修改传播分析)，修改后系统的调试与测试．图 1 给出软件维护活动的一个简单框架模型，它包含了基本的软件维护活动．在该框架中，首先是用户提出修改请求；然后进行特征定位，即把自然语言的修改请求规约成维护人员可理解的修改请求，并进一步理解与这次修改请求相关的软件系统以及系统内部各组件间的依赖关系；下一步就是进行修改影响分析，通过影响分析，管理人员可评估出这次修改所带来的影响范围，从而决定是否接受这次更改请求，若接受，则进入修改实施阶段，这个过程包括重构以及修改传播分析；当完成这次修改之后，需重新测试修改后的系统，确保修改的正确性与一致性，以及修改没有给系统其它部分带来副作用．本文总结的一些基于形式概念分析的应用主要从图 1 中的软件维护活动框架出发，讨论近十几年来形式概念分析在软件维

护这些应用中的一些相关研究工作.

本文针对图1软件维护活动框架中的四种软件维护活动分别进行介绍：软件理解、修改影响分析、重构、调试与测试. 图l软件维护活动框架要是因为概念格本身就是一种中间表示，可直接用于软件理解；其次，还有很大一部分工作在重构方面，这是因为概念格是一种层次化的表示，能够有效地识别一些可重构的代码；而在修改影响分析，调试和测试方面可查的研究工作还处于起步阶段，这也为广大软件工程学者提供一些好的关注点和研究方向.

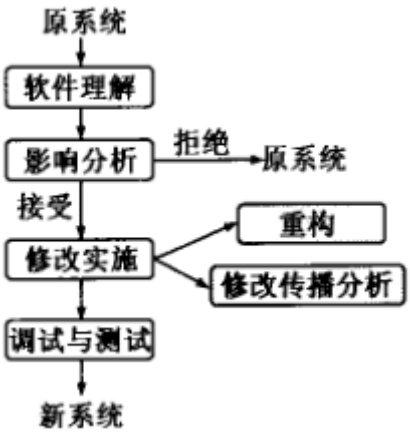

图 1 软件维护活动框架

4 形式概念分析在软件维护中的应用

近十几年来形式概念分析在软件维护中得到广泛与成功的应用，其应用过程通常比较简单，包括三步：首先构造形式背景，也就是一个二元关系表，确定形式对象与形式属性以及两者之间的二元关系；然后根据概念格构造算法生成概念格…；最后根据概念格上的特征，与具体应用相结合. 本文介绍了形式概念分析技术在软件理解、修改影响分析、重构、调试与测试4种软件维护活动方面的应用.

4．1 形式概念分析应用于修改影响分析

当对软件进行修改时，肯定会对软件的其他部分造成一些潜在影响，从而导致软件的不一致性. 软件修改影响分析是用来识别软件修改对软件其他部分所带来的潜在影响[4] ToneHa 将概念分析与分解切片结合起来，提出一种分解切片概念格的概念【5】，该方法主要适用于过程内针对某语句或者变量修改的影响分析.

孙小兵等人在利用形式概念分析来进行修改影响分析方面的研究做了大量的工作【6-9】，主要是将形式概念分析技术应用于面向对象语言程序的影响分析，其所分析的是类和成员方法之间的关系，得到一种新面向对象程序的中间表示，类与方法依赖关系格. 在该格上进行影响分析比较容易实现，且结果也是一种层次化的结果，可更好地指导维护人员准确定位修改所带来的影响.

4.2 形式概念分析应用于软件理解

软件系统在实施修改前，维护人员首先要对维护的软件进行软件理解以决定如何对系统进行修改. 软件理解是软件维护与演化过程中的首要活动，也是最难的活动之一，它包括了对软件文档和程序中元素及各种依赖关系的理解.

目前，形式概念分析在软件理解中的应用主要包括7个方面的活动：Web服务挖掘、代码特征分析、可重用性分析、类接口与类层次识别、特征定位、需求分析、演化分析. 图2给出当前形式概念分析在软件理解7个方面活动的统计结果，从该图可知，当前形式概念分析在特征定位应用方面的工作最多，其次是代码特征的提取与分析；而由于 Web 服务技术出现得比较晚，形式概念分析在 Web 服务挖掘方面的工作还较少.

Web 服务计算作为一种新出现的规范在近年来得到快速发展，形式概念分析在 Web 服务挖掘方面也有一些应用，主要是 Web 服务的挖掘. Aversano 等人利用概念格来挖掘服务在演化过程中发生变化的服务以及没有变化的服务【10】. Jiang 等人提出利用基于输入和输出参数的 Web 服务集的扩展概念格模型，来有效地组织服务，为 Web 服务的选择、优化和组合提供智能支持【11】. 形式概念分析在代码特征分析方面的工作得到很多人的关注. Cole 和 Dekel 等人利用

概念格提供一种“导航”视图，通过概念格上层次化的视图，指导维护人员应该从哪个方法开始着手去理解源代码[12,13]．Kuipers 将类型推理与概念分析结合起来在 COBOL 语言程序中识别对象[14]．Men 和 Ar6vMo 等人则利用形式概念分析去挖掘、分类程序代码中的关系、模式、编程风格，辅助维护人员理解要维护的软件[15,16]．而 Lienhard 等人从类层次中提取特征进行软件理解[17]。Pflatz 等人对程序运行轨迹进行分析，并根据概念格的层次关系去识别程序运行轨迹中各个元素的前后因果关系[18].

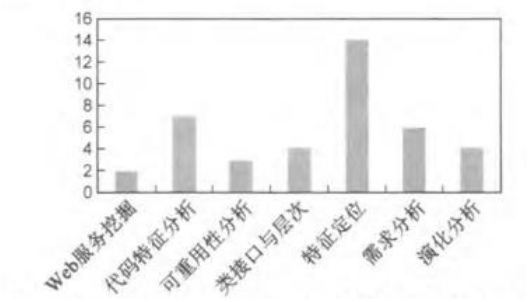

图 2 形式概念分析应用于软件理解各个的技术统计

还有一些工作用于理解软件组件或者接口的可重用性以便在软件维护过程中可重用这些组件或者接口．Tonella 等人通过从源代码中提取设计模式去理解程序[19]，这些设计模式不是预先知道在程序中存在的，而是直接从源代码中提取出来去指导程序的下一步演化．而 Viljamaa 等人利用形式概念分析从软件框架中提取可重用性接口的规约和代码[20]．另外，Zhang 等人将形式概念分析和程序切片技术组合起来从面向对象程序代码中挖掘可重用性代码[21].

形式概念分析在类接口与层次识别方面也有一些研究工作，对类层次和类接口的识别有助于维护人员整体地去理解软件结构[22]。Arevalo 等人通过形式概念分析技术将类层次区分为好的层次设计，坏的层次设计以及不正常的层次设计，这样可直接指导维护人员去理解相关的类层次[23]．另外，SuRon 等人通过形式概念分析自动识别出 UML 设计层次的类模型[24]．Dekel 等人则利用概念格对 Java 源代码中的类和方法之间的关系显示出源代码的类接口[25].

形式概念分析在软件理解中应用较多地集中在特征定位方面．特征定位用于将自然语言形式的修改请求映射到代码层次的修改，识别哪些代码实现需求中的哪些功能(或者非功能)特性[27]．Poshyvanyk 将信息检索与形式概念分析结合起来进行特征定位，找出与某些特征最相关的类或者方法，并以排序后的列表返回给维护人员，从而方便维护人员有效地进行特征定位[28]．另外，Xue 等人则通过形式概念分析从软件代码演化过程中不同产品的变体中挖掘出共同的特征，这样有利于后期相似特征的代码重用[29]．上面的特征定位工作都是形式概念分析用于源程序的静态分析，但对于大规模程序，如果仅采用静态分析进行程序理解，往往精确性不高，且代价较大．而动态分析可有效降低大规模程序的理解度．Eisenbarth 将形式概念分析应用于动态分析进行特征定位[30,31].

最后，形式概念分析在软件理解中的应用不仅包括对软件各元素依赖关系的理解，还包括软件演化过程中演化性的分析．Xu 等人提出一种基于模糊形式概念分析的程序聚类方法[32]．还有一些工作从代码演化过程中提取开发过程和演化过程中的一些模式去理解源程序[33]．另外，协同修改模式也可通过概念格提取出来[34]，这为后续软件演化以及维护过程中所需的维护活动提供一个有效的度量方法.

4.3 形式概念分析应用于重构

目前相关的形式概念分析应用于软件修改实施主要集中在重构领域．重构可在不改变软件原有功能的前提下，通过调整程序代码内部结构改善软件质量和性能，使其程序的设计模式和架构更趋合理，提高软件的可扩展性和可维护性[34]．形式概念分析很自然地为人们提供一个层次聚集的视图，这就为识别程序中的层次与聚集打下好的基础，目前将形式概念分析应用重构主要包括类层次结构重构，模块结构，“坏味道”设计和代码重构，由低层次语言规范向高层次的语言规范进行重构等．图 3 给出形式概念分析应用于重构各个活动的技术统计，从图中

可见，每种类型的重构活动都有 2～3 个技术支持.

重构活动最早开始关注的是对“坏味道”设计和代码进行重构. Moha 等人通过形式概念分析识别的方法去识别程序中的“坏味道”设计[35,36]. 而 Arevalo 等人则通过形式概念分析技术将类层次区分为好的层次设计、坏的层次设计以及不正常的层次设计，这样可直接指导维护人员去理解相关的类层次以及修改类层次中存在的“坏味道”. Joshi 则提出一种内聚格的表示方法，该格可用于指导维护人员对哪些类进行提取，转移哪些方法，以及确定需要的属性和删除没有使用到的属性等等[37].

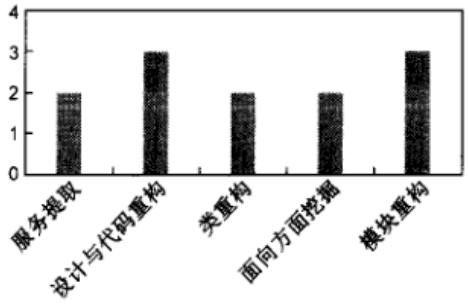

图 3 形式概念分析应用于重构各个活动的技术统计

另外，还有一些学者利用形式概念分析去挖掘识别程序代码中的方面或者服务，这是一种从低层次语言规范向高层次语言规范进行重构. Tourw6 和 ToneHa 等人通过概念格将那些分散在不同类中被不同测试用例覆盖的方法挖掘出来，作为面向方面语言程序中的方面使用. Del Grosso 和 Chen 等人利用形式概念分析技术对功能和组件进行分析，识别出可重构的服务应用在面向服务的软件中[38,39].

5 研究趋势与展望

形式概念分析在软件维护领域的应用在近十几年来取得长足的发展，尽管如此，我们认为如下一些方面仍值得国内外学者关注：

（1）应用领域的深入. 形式概念分析技术在程序理解和重构方面已取得大量的研究成果，而形式概念分析在修改影响分析、调试以及测试方面的研究工作目前还处于起步阶段，甚至在修改后的系统验证方面还没有找到相关研究文献，因而形式概念分析技术在这些方面的研究还有待深入.

（2）应用层次之间的可跟踪化. 虽然有部分研究人员将形式概念分析应用于需求层次或者设计层次，但各个层次使用形式概念分析技术所提取出来的概念是不同的，但这些概念之间从纵向的角度看是存在一定的可跟踪性，但目前还没有相关的从需求层次到设计层次以及规约层次之间的概念跟踪性研究.

（3）应用工具的支撑. 尽管形式概念分析技术已有一些工具支撑，但这些工具主要还是用于概念格的构造，当这些工具应用到某个具体的领域中还有一些不适应，需进一步修改和扩展，而这方面的工具目前还相当缺乏，所以开发适合某个具体领域形式概念分析的工具很有必要.

（4）形式概念分析支持软件维护的统一化. 形式概念分析可用于各种软件维护活动中，形式概念分析具有较完善的理论基础，研究如何利用该技术统一软件维护的各种活动：从修改(软件)理解到修改实施，修改后测试以及最后的系统验证，这样的研究将会很有意义.

6 结语

形式概念分析技术本身经过几十年的发展和完善，已成为具有较强形式化理论支撑的软件分析技术，该技术近年来被广泛应用到软件维护各种活动中，包括软件理解，修改影响分析，修改实施，调试与测试. 尽管形式概念分析在软件维护中有着广泛的成功应用，但其在应用领域深化、工具支撑等方面还需进行深入的研究.

参考文献：

[1]Ganter B，Wille R. Formal Concept Analysis：Mathematical Foundations [M]. Springer-Verlag，1996.

[2]Birkhoff G. Lattice Theory[M]. USA：American Mathematical Society，1940.

[3]Tilley T，Cole R，Becker P，eta1．A survey of formal concept analysis support for software engineering activities[A]．LNCS 3626：Formal Concept Analysis [C]．Berlin：Springer，2005．250—271.

[4]Li B，Sun XB，Leung H，Zhang S．A survey of code-based change impact analysis techniques【J】. Journal of Software Testing，Verification and Reliabiblity，2013，23(8)：613—646.

[5]ToneUa P．Using a concept lattice of decomposition slices for program understanding and impact analysis [JJ．IEEE Transactions on Software Engineering，2003，29(6)：495—509.

[6]孙小兵，李必信，陶传奇．基于 LoCMD 的软件修改分析技术[J]．软件学报，2012，23(6)：1368—1381.

Sun XB，Li BX，Tao CQ．Using formal concept analysis to support change analysis[J]．Journal of Software，2012，23(6)：1368—1381．(in Chinese)

[7]Sun XB，Li BX，211ang S，Tao CQ．Using lattice of class and method dependence for change impact analysis of object oriented programs l AJ．Proceedings of the Symposium on Applied Computing[C]. TaiChung，Taiwan：ACM，2011．1439—1444.

[8]Li BX，Sun XB，Leung H．Combining concept lattice with call graph for impact analysis [J]．Advances in Engineering Software，2012，53(11)：1—13.

[9] Li BX，Sun XB，Keung J．I℃A-CIA：An approach of using FCA to Support cross-level change impact analysis for object oriented Java programs[J]．Information&Software Technology，2013，55(8)：1437—1449.

[10] Aversano L，Bruno M，Di Pent M，Falanga A，Scognamiglio R．Visualizing the evolution of web services using formal concept analysis [A]. Proceedings of the International Workshop on Principles of Software Evolution[C]. Lisbon，Portugal：IEEE，2005．57—60.

[11]姜峰，范玉顺．基于扩展概念格的 Web 关系挖掘[J]．软件学报，2010，21(10)：2432—2444.

Jiang F，Fan YS．Web relationship mining based on extended concept lattice[J]．Journal of Software，2010，21(10)：2432—2444．(in Chinese)

[12]]Cole R，Becker P．Navigation spaces for the conceptual．1al analysis of software structure[A]．Proceedings of the International Conference On Formal Concept Analysis l C j．Lens，France：Springer，2005．113—128.

[13] Dekel U，Gil Y．Revealing class structure with concept lattices[A]．Proceedings of the l0th Working Conference On Reverse Engineering[C]．Victoria，Canada：IEEE，2003．353—365.

[14]Kuipers T，Moonen L．Types and concept analysis for legacy systems[A]．Proceedings of the International Workshop on Program Comprehension[C]．Limerick，Ireland：IEEE，2000．221—230.

[15]Mens K，Tourw6 T．Delving source code with formal concept analysis [J]．Journal of Computer Languages，Systems and Structures，2005，31(3—4)：183—198.

[16] Arevalo G，Buchli F，Nierstrasz o．Detecling implicit collaboration patterns [A]. Proceedings of the Working Conference on Reverse Engineering [C]．Delft，Netherlands：IEEE，2004．122—131.

[17]Lienhard A，Ducasse S，Ar6valo G. Identifying waits with formal concept analysis[AJ．Proceedings of the 20th IEEE／ACM International Conference on Automated Software Engineering [C]．Long Beach，CA，USA：ACM，2005．66—75.

[18]Pfaltz JL．Using concept lattices to uncover visual dependencies in software[A]．Proceedings of International Conference on Formal Concept

Analysis[C]. Dresden，Germany：Springer，2006．233—247．

[19]ToneUa P，Antoniol G．Inference of object oriented design patterns[J]．Journal of Software Maintenance and Evolution，2001，13(5)：309—330．

[20]Viljamaa J.Reverse engineering framework reuse interfaces[A]. Proceedings of Symposium on the Foundations of Software Engineering[C J．Newport Beach，CA：ACM，2003．217—226．

[21]]Zhang Z，Yang H，Chu WC．Extracting reusable object-oriented legacy code segments with combined formal concept analysis and slicing techniques for service integration[C]．Proceedings of International Conference Oil Quality Software[C]．Beijing，China-,IEEE，2006．385—392．

[22]Sneldng G，Tip F．Understanding class hierarchies using concept analysis[J]．ACM Transactions on programming Languages and Systems，2000，22(3)：540—582．

[23]Arevalo G，Ducasse S，Gordillo S，Nierstrasz O．Generating a catalog of unanticipated schema in class hierarchies using formal concept analysis[J]．Information and Software Technology，2010，52(11)：1167—1187．

[24]Sutton P，Maletic I P．Recovering UML class models from C++：A detailed explanation [J]．Information and Software Technology，2007，49(3)：212—229．

[25]Dekel U，Gil J．Visualizing class interfaces with formal concept analysis[A]．Proceedings of the Conference on Object-Oriented Programming，Systems，Languages，and Applications[C] Anaheim，CA，USA：ACM，2003．288—289．

[27]Dit B，Revere M，Gethers M，Poshyvanyk D．Feature location in source code：ataxonomy and survey [J]．Jotmml of Software：Evolution and Process，2013，25(1)：53—95．

Poshyvanyk D，Gethers M，marcus A．Concept location using formal concept analysis and information retrieval l J j．ACM Transactions on Software Engineering And Methodnology，2012，21(4)：23．

Xue YX，Xing ZC，Jarzabek S．Feature location in a collection of product variants[A]．Proceedings of the Working Conference on Reverse Engineering[C]。Kingston，ON，Canada:IEEE，2012．145—154．

[30]Eisenbarth T，Koschke R，Simon D．Locating features in source code[J]．IEEE Tramactions on Software Engineering，2003，29(3)：195—209．

[31]Eisenbarth T，Koschke R，Simon D．Aiding program comprehension by static and dynamic feature analysis[A]．Proceedings of the International Conference on Software Maintenance[C]．Florence，Italy：IEEE，2001．602—611．

[32]许佳卿，彭鑫，赵文耘．一种基于模糊形式概念分析的程序聚类方法[J]．计算机研究与发展，2009，46(9)：1556—1566．

Xu JQ，Peng X，Zhao WY．Program clustering for comprehension based On fuzzy formal concept analysis[J]．Journal of Computer Research and Development，2009，46(9)：1556—1566．(in Chinese)

[32]Glorie M，Zaidman A，Hofland L，Deursen A．Splitting a large software archive for easing future software evolution an industrial experience report using formal concept analysis[A]．Proceedings of European Conference on Software Maintenance and Reengineering[C]．Athens，Greece IEEE，2008．153—162．

[33]Glrba T，Ducasse S，Kuhn A，et a1．Using concept analysis to detect co-change patterns [A]．International Workshop on Principles of Software Evolution [C]．Dubrovnik，Croatia：IEEE．2007．83—89．

[34]Mere T，Tourw Tom．A survey of software refactoring [J]．IEEE Transactions on Software Engineering，2004，30(2)：126—139．

[35]Moha N，Hacene AR，Valtchev P，Gueneuc YG．Refactorings of Design defects using

relational concept analysis [A]. Proceedings of International Conference on Formal Concept Analysis[C]. Montreal，Canada：Springer，2008．289—304．

[36] Moha N，Rezgui J，Gueneneuc Y，Valtchev P，EI-Boussaidi G．Using FCA to suggest refactorings t0 correct design defects [A]．International Conference Concept Lattices and their applications[C]．Tunis，Tunisia：IEEE，2006．269—275．

[37]Joshi P，Joshi RK．Concept analyfis for class cohesion [A]．Proceedings of the 2009 European Conference On Software Maintenance and Reengineering [C]．Bombay Powai，Mumbai：IEEE，2009．237—240．

[38] Del Grosso C，Massimiliano，Di Penta．An approach for mining services in database oriented application[A]．Proceedings of the European Conference on Software MaintenanceandReengineering[C]．Amsterda m，Netherlands：IEEE，2007．287—296．

[39]Chen F，21：zhang ZP，Li JZ，Kang J，Yang HJ.Service identification via ontology mapping[A]．Proceedings of tlae International Computer Software and Applications Conference[C]．Washington，USA：IEEE，2009．486—491．