《智能信息处理》课程作业

基于形式概念分析的软件工程综述

张锦磊

作业	分数[20]
得分	

2021 年 11 月 29 日

基于形式概念分析的软件工程综述

张锦磊

（大连海事大学 信息科学技术学院，大连 116026）

摘 要 形式概念分析是由 Wille R 于 1982 年首先提出，它提供了一种支持数据分析的有效工具。概念格是一个以概念为元素的偏序集，它可以通过 Hasse 图可视化，其中每个节点是一个概念。概念格结构模型来源于形式概念分析理论，是 FCA 中的核心数据分析工具，它本质上描述了对象（样本）与属性（特征）之间的关联。本文将 FCA 与概念格技术引入软件工程领域，根据软件开发的不同阶段，介绍 FCA 与概念格技术在该环节的具体应用方式及其优缺点，并在此基础上分析软件开发应用中等进一步需要研究的内容。

关键词 形式概念分析；概念格；软件工程

A Survey of Software Engineering Based on Formal Concept Analysis

Zhang Jinlei

(School of Information Science and Technology, Dalian Maritime University, Dalian 116026)

Abstract Formal concept analysis was first proposed by Wille R in 1982. It provides an effective tool to support data analysis. A concept lattice is a partially ordered set with a concept as an element. It can be visualized by a Hasse diagram, where each node is a concept. The concept lattice structure model is derived from the formal concept analysis theory and is the core data analysis tool in FCA. It essentially describes the relationship between objects (samples) and attributes (features). This article introduces FCA and concept lattice technology into the field of software engineering. According to the different stages of software development, introduces the specific application methods and advantages and disadvantages of FCA and concept lattice technology in this link, and analyzes software development and application on this basis. Further research is needed. Content.

Key words formal concept analysis; concept lattice; software engineering

1 引言

最近十年来，软件工程成为国外形式概念分析与概念格应用研究中一个新的热点方向。一方面是由于软件工程更贴近于人们的社会经济生活，另一方面是由于软件本身存在的问题:(1) 软件在适应新环境的过程中，需要不断地进行扩展、修改、更新，从而使编程变得越来越复杂，并偏离原始的设计，使得软件的质量下降;(2) 在系统分析过程中，大多数的方法都存在一个缺陷，即在模块化的区域内不能提供一个适合的指导方案用于识别类和对象;(3) 应用在软件工程中的数据分析方法大多数倾向于通过大大减少给定的信息来获得极少数的“重要参数”。形式概念分析 (Formal Concept Analysis, FCA) 技术通过数据集中对象和属性之间的二元关系建立概念层次结构，再运用格代数理论对数据进行分析，可以保证信息的最大分解，同时保留数据之间的特殊化关系;由形式背景构建的概念格不会人为

减少给定信息的复杂性，且包含了所有的数据细节。因此国际上关于形式概念分析与概念格在这一领域中应用的相关研究发展迅速而突出。

其中具有代表性的应用研究集中在:代码特征定位、类层次再造、模块识别、基于切片技术的程序理解等。系统在开发过程中，由于不断地修改，使得系统越来越难理解，虽然利用逆向工程进行代码特征定位可以摆脱这种恶性循环，但成本巨大。利用 FCA 技术将系统中计算单元与特征建立关联，并基于此二元关系生成概念，不仅可以让用户参与到系统开始阶段代码特征的定义过程，从而使得在程序有问题的情况下，更容易解读需求变更和错误报告，而且整个过程成本明显下降。FCA 为类层次设计和维护提供了天然的理论框架，基于 FCA 产生的类层次不仅在语义上定义明确，而且在使用具体的算法时语义是保持独立的。在自动识别模块的过程中，之前使用的方法提取的模块信息不完整，而使用概念分析可以对模块信息的评判更标准。在对程序理解的过程中，将程序切片技术与概念格结合起来使用，形成新的分解切片图，有利于对程序的深层次理解。本文将结合软件工程的软件开发过程，围绕各个环节的实现目标，以对象建立和属性提取为关键对象，介绍 FCA 在软件工程中的具体应用方式及带来的优越性。[1]

2 形式概念分析的相关定义

2.1 形势背景

形式背景（formal context）可以表示为三元组 T=(O，D，R)，其中 O 是事例（对象）集合，D 是描述符（属性）集合，R 是 O 和 D 之间的一个二元关系，则存在唯一的一个偏序集与之对应，并且这个偏序集产生一种格结构，这种由背景（O，D，R）所诱导的格 L 称为概念格。格 L 中的每个节点是一个序偶(称为概念)，记为(X，Y)，其中 X∈P(O)称为概念的外延；Y∈P(D)称为概念的内涵。每一个序偶关于关系 R 是完备的，即有性质：1)X={x∈O|∀y∈Y，xRy}，2)Y={y∈D|∀x∈X，xRy}。

在概念格节点间能够建立起一种偏序关系。具体地，给定概念 H1=(X1,Y1)和 H2=(X2,Y2)，则 H1＜H2<==>Y1⊂Y2，领先次序意味着 H1 是 H2 的父节点或称直接泛化。根据偏序关系可生成格的 Hasse 图：如果 H1＜H2 并且不存在另一个元素 H3 使得 H1＜H3＜H2，则 H1 从到 H2 就存在一条边。

表 1 给出一个形式背景。

表 1 举例形式背景

att / obj	F	R	E	M	S
1	f1	r1	e1	m1	s1
2	f2	r1	e3	m1	s1
3	f2	r1	e3	m1	s3
4	f3	r2	e1	m2	s1
5	f3	r2	e1	m2	s2
6	f1	r1	e2	m1	s1

2.2 概念格

概念格的表示形式是 Hasse 图，概念格的构建的基础是形式背景，形式背景描述了多个形式概念之间的关系，单个形式概念描述了形式对象以及形式对象所具有的形式属性之间的关系。所以，概念格的构建必须明确不同形式对象以及不同形式对象所具有的形式属性。[2]

概念格的构建包含以下几个步骤：生成形式背景，约简形式背景，生成单值形式背景，确定父子关系，绘制 Hasse 图，补充

形式概念的上确界和下确界，最后获得概念格。建格的过程实际上是概念类聚的过程。因此，在概念格中，建格算法具有很重要的地位对于同一批数据，所生成的格是唯一的，即不受数据或属性排列次序的影响，这也是概念格的优点之一。概念格的建格算法可以分为两类：批处理算法和增量算法。概念格可以添加背景知识，这些知识以 if…then 的规则形式出现。概念格甚至可以只用背景知识建造。

表 1 所给的形势背景对应的概念格如图 1 所示。

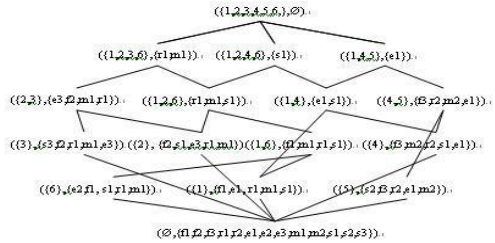

图 1 表 1 对应概念格

3 软件开发和 FCA

根据传统的软件生命周期模型来看，软件生命周期由软件定义、软件开发和软件维护 (也称为运行维护) 三个时期组成，每一个时期又进一步划分成若干个阶段。软件定义时期又可以划分为软件计划、需求分析、软件设计三个阶段。这一时期主要是确定要做的软件“是什么”，对软件进行顶层设计，描绘出软件架构，并对目标软件系统提出完整、准确、清晰和具体的要求。软件开发时期主要包括程序编码和软件测试两个阶段. 根据实际情况选择合适的编程语言，并编写测试计划、测试分析报告等文档，用合理的测试方法 (黑盒、白盒等) 来逐步测试并改正系统中的错误.软件维护是软件生命周期的最后一个阶段，也是持续周期最长、花费代价最高的一个阶段。虽然是软件投入运行之后需要进行的工作，但软件维护的工作通常会占用软件开发机构 60%以上的精力。软件维护一般是指对用户开发的软件系统进行修改，一般包括四个方面：(1) 矫正-通过修改系统更好的满足需求。(2) 适应性-通过修改系统使其能够在一个新环境中正常运转。(3) 完善-为系统增加新的函数功能。(4) 预防-通过提高设计和实现，以便未来可以进行更好的维护。

形式概念分析和概念格在软件编码之前的应用主要包括:需求分析、组件重用、形式规范等;形式概念分析和概念格在软件编码之后的应用包括:需求矫正、环境适应、完善功能等。[3]根据上面所介绍的软件工程生命周期，结合 FCA 技术, 我们在下面介绍到的应用如图 2 所示。

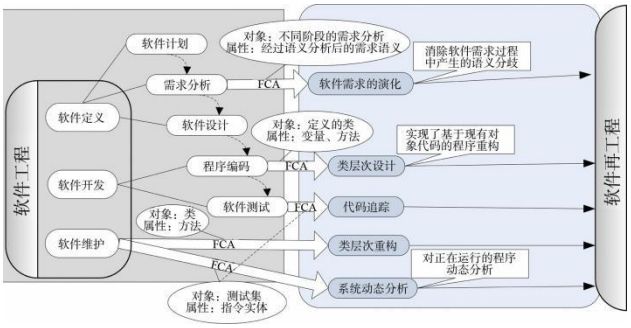

图 2 FCA 在软件工程方面的应用简图

3.1 软件定义阶段

首先根据 Cimiano 等提出的“基于 FCA 从文本集中了解概念层次”方法，捕获在需求分析中出现的动名词语义信息。现在做出如下假设:为了保持语义信息在不同需求阶段的一致性， 将之前在各个需求阶段得到的关系保存下来。现在的工作就是使用形式概念技术对不同阶段的需求分析进行对比分析，消除语义分歧和矛盾。

3.2 软件开发阶段

Godin 等认为形式概念分析为设计和维

护面向对象软件的类层次提供了天然的理论框架，他们基于形式概念分析的类层次的设计与维护保持类的特化关系，在类层次结构的生命周期开发中十分有效，实现了基于现有对象代码的程序重构。

这个阶段的任务是根据详细设计的结果，选择一种适合的程序设计语言，把详细设计的结果翻译成正确的、容易理解并易于维护的程序源代码。结合 FCA 和面向对象编程的特点，介绍基于 FCA 的类层次设计。现在有四个类，分别是 Class1, Class2, Class3, Class4.用 UML 图表示它们的属性如图 3 所示。则它们对应的形式背景如表 2 所示 (x 表示类和属性的关系)。构造概念格后，并将对应的类表示出来，如图 4 所示。

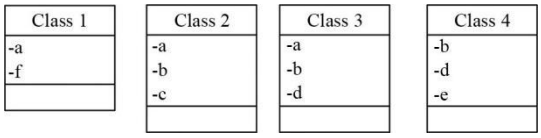

图 3 类和它对应的属性

表 2 类对应的形式背景

	a	b	c	d	e	f
Class1	x					x
Class2	x	x	x			
Class3	x	x		x		
Class4		x		x	x	

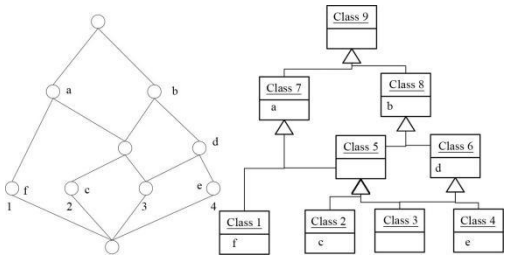

图 4 每一个节点对应的类

从图 4 中我们可以看出最初的四个类到底层的时候都是各自只有一个属性，新增加的类 (从 Class5 到 Class9) 将公共的属性提取出来。总体看来，所有的子类在经过特殊处理后，无一例外地继承了父类的属性。概念格的这种存储结构保证了每一个节点只存储一个类的属性，为类的扩展提供了很好的层次性。

Eisenbarth 等将形式概念分析用于系统需求识别的计算单元特征定位.通过形式概念分析，将由用户及用户行为触发的代码特征映射集合进行重构，然后在给定特征集中识别出全局和局部计算单元，通过动态与静态分析结合的方法迅速聚焦于相关的特定特征集，其效果优于早期的特征与场景的一对一的对应关系。将概念分析用于软件结构，把源代码分析器产生的信息存储在知识库中，然后使用系统中的新关系和类型、变量、成员、程序包等拓展知识库，而这些正是建立在形式概念分析的基础上。

3.3 软件维护阶段

软件维护包括矫正、适应性、完善、预防等方面， FCA 在软件维护方面的应用覆盖了上面所提到的四个方面，而且这些应用有一个共同点，即通过组织软件系统的构件，提取可以理解的结构。这些应用的不同则取决于输入、构造的概念格以及将概念格应用在哪里。

Dekel 使用概念格技术对 JAVA 类中的代码进行检查，根据类中成员作用的领域，构造概念格，然后通过将“方法调用图”附加到之前的类层次概念格上，得到一个“嵌入式调用图”，它可以提供更多关于类的详细信息，从而实现类结构的可视化，更易于对单个类的理解。

Poshyvanyk 等借助形式概念分析，结合潜在语义索引 (Latent Semantic Indexing) 对开发者在识别源代码的过程中进行概念

定位。这种基于概念定位技术的信息检索，是使用搜索引擎并结合潜在语义索引，允许使用者搜寻源代码及与之相关的文件，并根据搜索到的元素自动建立概念格。[4]

由于很多软件在设计的时候不能很好地遵循面向对象的特点和设计原则，使得软件缺乏模块化，从而造成软件维护困难，不易对软件进行功能扩展。另外， Hakik 等人基于形式概念分析将软件体系结构重新模块化，通过测量耦合和内聚来估计被重新模块化后的系统性能。Cellier 则将形式概念分析用到故障跟踪方面。排除故障是软件开发过程中比较费时的一项工作，而将形式概念分析与关联规则相结合，就可以用来分析执行代码中的错误，提高开发效率。[5]

4 总结

在软件工程领域，形式概念分析与概念格还被应用于组件重构、设计帮助等诸多方面。这些应用使得在软件开发过程中，客观成分逐渐加大，形式化的特征逐步增强，标准化趋势更加明显。另一方面，这些应用使得软件工程对个体认知的依赖程度逐步减弱，受研发人员个体素质的影响逐渐消失，进而大大提高了软件设计、开发、维护、重用的效率和效益。目前对概念格的应用还是一些概念分析中初步应用，并没有涉及到概念格中更丰富的理论和代数应用;其次，在软件完整性、合格检测、接受支持或编码方面的应用较少，这些方面将是 FCA 要研究的领域。

参 考 文 献

[1]智慧来,李逸楠.形式概念分析中的面向对象概念约简[J].海南热带海洋学院学报,2021,28(05):66-71.

[2]易利. 基于概念格的面向对象程序回归测试[D].湖南大学,2006.

[3]臧国轻,李瑞光,郑珂.形式概念分析在软件工程中的应用综述[J].河南大学学报(自然科学版),2018,48(03):309-317.

[4]畅鹏.形式概念分析在软件工程中的应用[J].通讯世界,2017(19):273-274.

[5]孙小兵,李云,李必信,文万志.形式概念分析在软件维护中的应用综述[J].电子学报,2015,43(07):1399-1406.