《智能信息处理》课程作业

基于形式概念分析的通用数据结构操作

邓杰进

作业	分数[20]
得分	

2020 年 12 月 6 日

基于形式概念分析的通用数据结构操作

邓杰进

(大连海事大学计算机科学与技术辽宁省大连市中国 116026)

摘 要 概念格理论也叫形式概念分析，由德国 Wille 教授基于由外延和内涵两部分所组成的思想单元这一哲学理解首先提出，它作为形式化的数学方法为数据分析提供了一种有效的工具。概念格本质上描述了对象和属性之间的联系，表明了概念之间的泛化和例化关系，其相应的 Hasse 图则实现了对数据的可视化。目前，形式概念分析已被广泛地研究，并应用到机器学习、软件工程和信息获取等领域。因此，研究概念格的基本理论以及将其应用于通用数据结构操作等知识发现有着非常重要的意义。

关键词 形式概念；概念格；Hasse 图，通用数据结构

中图法分类号 TP311.20 **DOI 号** 10.3969/j.issn.1001-3695.2014.01.030

General data structure operation based on formal concept analysis

Dengjiejin

(Computerscienceandtechnology，Dalianmaritimeuniversity,LiaoningDalian,116026,China)

Abstract Conceptual lattice theory, also known as formal concept analysis, is based on the thought unit composed of extension and connotation by Professor Wille from Germany, which is the first exit of philosophical understanding. As a formal mathematical method, it provides an effective tool for data analysis. Concept lattices essentially describe the relationship between objects and attributes, indicate the generalization and instantiation relationship between concepts, and their corresponding Hasse plans realize the visualization of data. At present, formal concept analysis has been widely studied and applied to machine learning, software engineering and information acquisition. Therefore, it is of great significance to study the basic theory of concept lattice and apply it to the knowledge discovery of general data structure operation.

Keywords formal concepts; concept lattice; Hasse diagram; universal data structure

1 引言

数据结构是一种特殊的组织和存储数据的方式，可以使我们可以更高效地对存储的数据执行操作。数据结构在计算机科学和软件工程领域具有广泛而多样的用途。

几乎所有已开发的程序或软件系统都使用数据结构。此外，数据结构属于计算机科学和软件工程的基础。当涉及软件工程面试问题时，这是一个关键主题。因此，作为开发人员，我们必须对数据结构有充分的了解。

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关系；由形

=式背景构建的概念格对通用数据结构的操作分析是非常有必要的。

1 基本概念

定义 1 一个形式背景 K 是一个三元组：K=(G,M,I)，其中 G 为所有对象的集合，M 为所有属性的集合，I 是 G 与 M 之间的二元关系。

对于 A⊆G,定义 A'={m| m∈ M， ∀ g∈A，gIm}，对于 B⊆M，定义 B'={g| g∈G， ∀ m∈B，gIm}。

定义 2 设（G，M，I）为形式背景，如果一个二元组（A，B）满足 $A'=B$ 且 $B'=A'$，称（A，B）是一个概念。其中，A 称为概念的外延，B 称为概念的内涵。

概念格

概念格的每个节点是一个形式概念，由两部分组成：外延，即概念所覆盖的实例；内涵，即概念的描述，该概念覆盖实例的共同特征。另外，概念格通过 Hasse 图生动和简洁地体现了这些概念之间的泛化和特化关系。从数据集中(概念格中称为形式背景)中生成概念格的过程实质上是一种概念聚类过程;目前，已经有了一些建造概念格的算法，并且概念格在信息检索、数字图书馆、软件工程和知识发现等方面得到应用。概念格是一种具有完备性的结构，它作为知识表示的一种形式在表现概念之间关系的规则方面有其独特的优势。

定义 3 C1 =(A1，B1)和 C2=(A2，B2)是形式 背景（G，MI）上的任意两个概念，定义二元关 系≦：C2≦C1 ⇔ B1 ⊆ B2 ⇔ A2 ⊆ A1， 称 C1 是 C2 的父概念，C2 是 C1 的子概念，如果不 存在另外一个概念 C3，使得 C2≦C3≦C1，称 C1 是 C2 的直接父概念，C2 是 C1 的直接子概念，记 为 C2＜C1。显然，关系“≤”是集合 β(K)上的一个偏序，它可诱导出 β(K)上的一个格结构，可以证明，它 是一个完备格，相应的下确界和上确界定义为：∧(At， Bt)=(∩At， (∪Bt) ’) ∨(At， Bt) =((∪At)’, ∩Bt) 其中(At ，Bt)∈β(K)，T 是指标集，此完备 格称为形式背景 K 的概念格。

定义 4 对于形式背景 K=(O，A，R)，存在 唯一的一个偏序集与之对应，并且该偏 序集存在一个唯一的下确界和一个唯一的上确 界，这个偏序集产生的格结构称为概念格(concept lattice)，记为 L(O，A，R)。

概念格可以图形化形式表示为有标号的线图，概念格的每个节点表示一个形式概念，由外 延和内涵两部分组成。概念的外延是指此概念所 覆盖的对象的集合；概念的内涵则是外延所具有 的共同属性的集合。这种线图也称为 Hasse 图， 它是概念格的可视化表示。

2 通用数据结构概述

数据结构是带有结构的数据元素的集合，它研究的是数据的逻辑结构和数据的物理结构以及它们之间的相互关系，并对这种结构定义相适应的运算，设计出相应的算法，并确保经过这些运算以后所得到的新结构仍然保持原来的结构类型。简而言之，数据结构是相互之间存在一种或多种特定关系的数据元素的集合，即带“结构”的数据元素的集合，“结构”就是指数据元素之间存在的关系，分为逻辑结构和存储结构。

数据结构的研究内容是构造复杂软件系统的基础，它的核心技术是分解与抽象。通过分解可以划分出数据的三个层次；再通过抽象，舍弃数据元素的具体内容，得到的就是逻辑结构。。类似地，通过分解将处理要求划分成各种功能，再通过抽象舍弃实现细节，就得到运算的定义。上述两个方面的结合就可以将问题变换为数据结构。

3 通用数据结构操作形式概念分析

数据结构是计算机学科的基础部分，也是计算机学科中最为经典的部分，通用的数据结构有很多。本文主要数据结构中的 Array、Stack、Queue、Singly-Linked List、Skip List、Binary Search Tree、B-Tree、Red-Black Tree、AVL Tree、KD Tree 这几种数据结构作为对象集。把对数据结构的操作，例如 Access、Search、Insertion、Deletion 的平均时间复杂度和最坏的时间复杂度作为属性集构建形式背景，如表一.

表一：形式背景

	Access/Ave	Search/Ave	Insertion/Ave	Deletion/Ave	Access/Wo	Search/Wo	sertion/W	Deletion/Wo
Array	O(1)	O(n)	O(n)	O(n)	O(1)	O(n)	O(n)	O(n)
Stack	O(n)	O(n)	O(1)	O(1)	O(n)	O(n)	O(1)	O(1)
Queue	O(n)	O(n)	O(1)	O(1)	O(n)	O(n)	O(1)	O(1)
Singly-Linked List	O(n)	O(n)	O(1)	O(1)	O(n)	O(n)	O(1)	O(1)
Skip List	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)	O(n)	O(n)	O(n)
Binary Search Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)	O(n)	O(n)	O(n)
B-Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))
Red-Black Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))
AVL Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))
KD Tree	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)	O(n)	O(n)	O(n)

此形式背景为多值，为了方便，把最差平均时

间复杂度为 O(n)的操作认为时间复杂度高，低于 O(n)的为低时间复杂度低。并把时间复杂度高的用 0 表示，把时间复杂度低的用 1 表示，简化后的形式背景如表二。

表二：简化的形式背景

	Access/Ave	Search/Ave	Insertion/Ave	Deletion/Ave	Access/Wo	Search/Wo	Insertion/Wo	Deletion/Wo
Array	1	0	0	0	1	0	0	0
Stack	0	0	1	1	0	0	1	1
Queue	0	0	1	1	0	0	1	1
Singly-Linked List	0	0	1	1	0	0	1	1
Skip List	1	1	1	1	0	0	0	0
Binary Search Tree	1	1	1	1	0	0	0	0
B-Tree	1	1	1	1	1	1	1	1
Red-Black Tree	1	1	1	1	1	1	1	1
AVL Tree	1	1	1	1	1	1	1	1
KD Tree	1	1	1	1	0	0	0	0

可以依据基于约简意义的概念格构造方法对其进行简约。把属性值相同的对象进行合并。通过观察，可以看出 Stack、Queue、Singly-Linked List 可以合并，Skip List、Binary Search Tree、KD Tree 可以合并，B-Tree、Red-Black Tree、AVL Tree 可以合并，如表三。

表三：约简后的形式背景

	Access/Ave	Search/Ave	Insertion/Ave	Deletion/Ave	Access/Wo	Search/Wo	Insertion/Wo	Deletion/Wo
Array	1	0	0	0	1	0	0	0
Stack/Queue/Singly-Linked List	0	0	1	1	0	0	1	1
Skip List/Binary Search Tree/KD Tree	1	1	1	1	0	0	0	0
B-Tree/Red-Black Tree/AVL Tree	1	1	1	1	1	1	1	1

把对象的属性值为 1 的作为默认值。把简约的形式背景变为单值的形式背景。并用 1,2,3,4，5，6，7，8 依次代表各属性。用 a, b, c，d, e, f, g, h, i, j 依次代表各个对象。则有对象集{a, b, c, d, e, f, g, h, i, j}，属性集{1，2，3，4,5,6,7,8}，如表四，表五。

表四：单值形式背景

	1	2	3	4	5	6	7	8
a	*				*			
b			*	*			*	*
c			*	*			*	*
d			*	*			*	*
e	*	*	*	*				
f	*	*	*	*				
g	*	*	*	*	*	*	*	*
h	*	*	*	*	*	*	*	*
i	*	*	*	*	*	*	*	*
j	*	*	*	*				

表五：约简后的形式背景

	1	2	3	4	5	6	7	8
a	1	0	0	0	1	0	0	0
b/c/d	0	0	1	1	0	0	1	1
e/f/j	1	1	1	1	0	0	0	0
ghi	1	1	1	1	1	1	1	1

为了方便构造形式背景的概念格，将单值形式背景转换为带有父子关系（继承关系）的单值形式背景，也就是对属性个数进行排序。按照属性个数从上到下进行排序，将 j 放在 f 下面，如表六。

表六：带有父子关系的单值形式背景

	1	2	3	4	5	6	7	8
a	*				*			
b			*	*			*	*
c			*	*			*	*
d			*	*			*	*
e	*	*	*	*				
f	*	*	*	*				
j	*	*	*	*				
g	*	*	*	*	*	*	*	*
h	*	*	*	*	*	*	*	*
i	*	*	*	*	*	*	*	*

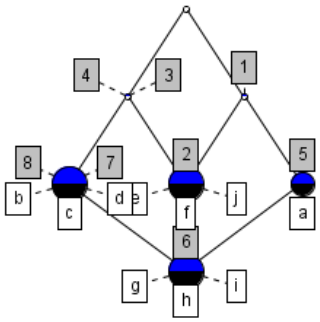

图 1 所表示背景对应的概念格

4 4．分析与总结

根据形式背景生成的形式背景概念格，概念格如图 1 所示。

通过概念格，我们可以发现一些规律。

1. 通用数据结构操作的平均时间复杂度基本小于或等于数据操作的最坏时间复杂度。从表中可以看出 a, b, c, d, g, h, i 的平均时间复杂度与最坏的时间复杂度相同，e, f, j 的平均时间复杂度小于最坏的时间复杂度。这和实际情况是相同的，正常来说，平均时间复杂度一定小于或等于最坏的时间复杂度。
2. 通用数据结构操做的最坏时间复杂度低的时候，它的平均时间复杂度也相应的较低。这是因为一个操作的平均时间复杂度小于或等于最坏的时间复杂度。所以当最坏时间复杂度低的时候，平均时间复杂度也必然是比较低。

5 结束语

本文通过对通用数据结构在查找、插入、与删除的平均时间复杂度和最坏时间复杂度构建形式背景，给出了从概念转化为形式概念、背景转化为形式背景、约简形式背景转化为单值形式背景再构造概念格的整体过程，全面分析其特征和关系。通用数据结构是计算机的基础部分。在程序设计中，选用合适的数据结构对程序运行效率提高有很大

帮助。将形式概念引入通用数据结构中，可以更有利帮助我们运用这些数据结构。

6 参考文献

[1]周超, 任志宇, 毋文超. 基于形式概念分析的语义角色挖掘算法[J]. 计算机科学, 2018, 45(12):117-122+129.

[2]石光莲, 杨敏. 基于FCA的Folksonomy用户兴趣研究述评[J]. 现代情报, 2017, 37(05):172-177.

[3]Ganter B,Wille R.Formal concept analysis mathematical oundations[M].Berlin:Springer-Verlag,1999.

[4] Ganter B,Wille R,Franzke C.Formal Concept Analysis: Mathematical Foundations[M].Berlin,Germany:Springer,1997.

[5] 徐伟华等. 形式概念分析的理论与应用[M]. 北京:科学出版社, 2016.

[6] 罗佳琪, 吴霞, 张家录, 杜佳甜. 基于形式概念的对象粒的属性逻辑公式描述[J]. 模糊系统与数学, 2020, 34(01):41-48.

[7] 白冬辉, 张涛, 魏昕宇. 基于属性度的属性排序算法[J]. 计算机工程与应用, 2017, 53(05):64-6