江西财经大学博士论文文库

吴方君 / 著

Z形式规约切片的研究

Research on Z Formal Specifications Slicing

中国科学技术大学出版社

江西财经大学博士论文文

吴方君／著

Z形式规约切片的研究

Research on Z Formal Specifications Slicing

中国科学技术大学出版社

图书在版编目(CIP)数据

Z形式规约切片的研究/吴方君著. —合肥:中国科学技术大学出版社,2006.12

(江西财经大学博士论文文库)

ISBN 7-312-01354-6

Ⅰ.Z… Ⅱ.吴… Ⅲ.电子计算机—算法理论 Ⅳ.TP301.6

中国版本图书馆CIP数据核字(2006)第152242号

出版 中国科学技术大学出版社

安徽省合肥市金寨路96号,邮编:230026

网址:http://press.ustc.edu.cn

印刷 合肥义兴印务有限责任公司

发行 中国科学技术大学出版社

经销 全国新华书店

开本 880 mm×1230 mm 1/32

印张 6.25

字数 183千

版次 2006年12月第1版

印次 2006年12月第1次印刷

印数 1~3 000册

定价 18.00元

序

在漫漫历史长河中，江西英才辈出，涌现了陶渊明、欧阳修、曾巩、王安石、朱熹、文天祥、宋应星、汤显祖、詹天佑等一大批文学家、政治家、科学家，为中华文明的发展做出了重要的贡献。江西财经大学坐落于赣江之畔的历史文化名城南昌。其前身是建于1923年的江西省商业学校，经过几代江财人的努力，江西财经大学已经发展成为一所以经济类、管理类为主体的多科性教学研究型大学。现有管理科学与工程一级学科博士点1个，产业经济学、财政学、政治经济学、会计学、西方经济学等5个二级学科博士学位授权点，应用经济学博士后流动站1个，博士、硕士研究生导师三百余人。进入新世纪以来，学校各项事业的发展正迈上一个新的台阶，尤其是研究生教育快速发展。

目前，我校正处于实现建设一流多科性教学研究型大学的战略目标的重要时期，而实现这一战略目标的关键就是，学校的发展要从量的扩张转向质的提升，从做大向做强转变。

毫无疑问，培养高层次、高素质的人才是实现我校“质的提升”与“做强”目标的必由之路。博士研究生作为国家培养的高层次专门人才，是社会各行业骨干的后备军，其创新能力的高低标志着我国高层次教育水平的高低，对我国社会的人才素质有着重要的影响。基于这一认识，我校始终把提高创新能力作为研究生教育的核心工作。为此，我们广聘名师，集聚优质师资资源，培育和构建一流学科队伍。目前我校的师资团队拥有近四百位博士，为我校的可持续发展奠定了良好的人才基础。在图书信息资料建设、实验室建设等方面也投入了大量的人力物力，扎扎实实地推进我校的“养大气、育大师”工程，为提高我校研究生的培养质量提供了较好的硬件资源支持。

为检阅我校博士研究生培养质量，展示江财学子的学术成果，促进我校学科建设的进一步发展，提高我校研究生教育的质量，我们于

2004年出版了《江西财经大学博士论文文库》。这套文库的出版获得了社会的广泛好评，我们深受鼓舞。于是，我们不揣浅陋，再次推出一套《江西财经大学博士论文文库》，这套文库收录了我校博士研究生在校期间潜心研究的部分优秀成果。虽不乏前沿性、创新性的观点、方法与理论，但其中一定存在问学旅程中学步的痕迹，有着诸多不足，祈请广大专家、学者批评指正！我相信，在广大读者的关心爱护下，江西财经大学的博士论文文库将会汇集越来越多的优秀学术成果。

是为序。

廖进球

2006年12月10日

前　言

本书充分利用已有的程序切片和依赖性分析技术，结合国内外在形式规约切片及其应用方面的最新研究动态，在Z形式规约切片及其应用方面展开研究工作.

本书共八章，分成四大部分：第一部分为第1章，是全书的一个概述；第二部分包括第2～5章，主要研究了Z形式规约切片技术；第三部分包括第6～7章，主要研究了Z形式规约切片的应用；第四部分为第8章，是对全书的结论. 各章节组织如下：

1. 导论

作为本书的导论，从程序切片技术的发展等方面来分析问题提出的背景、研究目的和意义，介绍国内外现有的解决方案及其发展趋势和存在的不足；描述本书的研究方法及其创新.

2. 基于依赖性分析的Z形式规约切片

针对Z形式规约的特点，提出了一种Z形式规约依赖性分析的模型，该模型详细地描述了模式内和模式间的依赖；在此基础上给出了一种有效的基于图的可达性分析的Z形式规约切片方法，该方法具有较高的精确度.

(1) Z形式规约的依赖性分析. 为了更好地描述Z形式规约和强调操作模式发生时应满足的条件，除了分析传统的数据依赖和控制依赖外，我们引入一种新的依赖关系——逻辑依赖，它是由操作模式的前置条件引起的. 逻辑依赖是控制依赖的一种特殊形式.

(2) Z形式规约的图形化表示. 鉴于Z形式规约没有类似程序代码的“主程序”，我们采用面向对象系统依赖图的形式对其进行图形化表示，对于模式和形式规约分别引入模式依赖图和形式规约依赖图的概念.

(3) 基于图的可达性分析的Z形式规约切片方法. 图形可达性算法是计算程序切片的一个有效方法，研究者已经提出了几种高效

的基于依赖图的计算程序切片的方法，我们也在形式规约的图形化表示，即模式依赖图和形式规约依赖图的基础上利用图形可达性算法求解模式切片和形式规约切片.

3. 基于依赖性分析的Z形式规约切片的形式化描述

尝试着用Z语言来形式化描述程序切片，考虑了Weiser M.的程序切片和Ottenstein K. J.程序切片、程序依赖图、系统依赖图、子程序切片算法和过程间切片算法等常用的方法.因为形式规约切片是程序切片的一种特殊形式，所以对于程序切片的形式化描述同样可以应用于形式规约切片.

(1) 切片的形式化描述.考虑Weiser M.的程序切片和Ottenstein K. J.程序切片.

(2) 依赖图的形式化描述.根据依赖图的基本概念和定义，把依赖图分成节点和边两个基本组成部分进行形式化描述；根据依赖图的类型把依赖图分成程序依赖图和系统依赖图进行形式化描述.

(3) 程序切片算法的形式化描述.对子程序切片算法和过程间切片算法进行形式化描述.

4. 基于关系演算的Z形式规约切片

提出了一种运用关系演算计算切片的方法.对于Z形式规约中每个模式引入三个关系来辅助切片的求解过程，该过程主要利用关系代数的选择、投影和连接等运算和Z语言自带的关系演算，如定义域限定、值域限定、求关系的定义域和值域等运算来计算切片.

(1) 基于关系代数演算的Z形式规约切片.对于Z形式规约中每个模式引入三个关系来辅助切片的求解过程，该过程主要利用关系代数的选择、投影和连接等运算来计算切片.

(2) 基于Z关系演算的Z形式规约切片.对于Z形式规约中每个模式引入三个关系来辅助切片的求解过程，该过程主要利用Z语言自带的关系演算，如定义域限定、值域限定、求关系的定义域和值域等运算来计算切片.

5. 变量定义和使用情况的探讨

第2章和第4章都涉及谓词对变量的定义和使用情况，而Z语言使用相对简单且为人们所熟悉的集合符号、关系、映射、序列、包和

一阶谓词逻辑等表示法来描述系统，所以我们分别分析和讨论这些数学抽象对变量的定义和使用情况，作为对第 2 章和第 4 章的补充和完善.

6. Z 形式规约切片在提升和定理证明中的应用

(1) 把 Z 形式规约切片应用到提升中去，给出一种求解提升模式 Promote 的公式.

(2) 把 Z 形式规约切片应用到定理证明中去，运用形式规约后向切片技术，从结论出发找出影响结论的前提.

7. 基于依赖性分析的 Z 形式规约度量

(1) 在模式依赖性分析的基础上提出一组针对 Z 形式规约的耦合性度量准则. 该组度量准则考虑了模式修饰、模式包含、模式演算和模式作为类型等多个方面.

(2) 运用实例检验度量准则与人们的经验是否一致.

(3) 为了更好地说明提出的 Z 模式耦合性度量与其他度量的联系，我们同时也考察著名的 CK 度量，并比较这两种度量在评估系统时的优劣.

8. 总结与展望

这是对本书研究和实践工作的总结，综述我们在程序切片研究领域获得的成果以及在其他相关领域所做的研究工作，并且指出现有研究工作的局限性以及有待提高和改进的方面，阐述我们正在进行或将要进行的研究工作的要点.

该书是在本人博士学位论文的基础上加工整理而成的，由于本人水平有限，错误和遗漏在所难免，恳请读者批评指正。衷心感谢江西省自然科学基金(编号:0501773)对部分研究的资助.

目　录

第1章 导 论

1.1 Z形式规约切片的研究依据

1.1.1 研究背景

软件工程中的形式化方法是一种基于数学的软件开发方法，它可应用于软件工程的各个阶段，不仅可以提高软件系统的正确性和可靠性，而且还可提高软件开发的效率. 该方法日益受到计算机界和工业界的高度重视，在航空、航天、网络协议、安全协议、嵌入式系统等方面获得了广泛的应用. 例如，美交通碰撞避免系统 TCAS Ⅱ (traffic collision avoidance system)是保障飞行安全的重要系统，就是使用形式规约语言 RSML 来刻画的.

在安全性要求极高的领域，如军事等尖端领域，软件可靠性是一个非常重要的问题. 目前人们主要依赖软件测试来提高软件可靠性，但无论是提高测试的覆盖率还是修正测试中所发现的错误，都需要付出高昂的代价，而且无法从根本上解决程序的正确性问题. 形式化方法从软件的本质和规律入手，在开发过程中通过严格的数学推导和证明来保证程序的正确性. 形式化方法的出发点是数学逻辑方法，它是公理方法发展的高级形态，也是实现软件自动化的重要基础. 前期的规范化虽然增加了工作量，但在测试阶段节约的时间非常明显，系统整体质量的显著提高则是更为丰厚的回报.

使用形式规约语言描述软件的需求规约说明，有下列优点：

(1) 基于严格的数学概念和理论，避免了用自然语言描述时可能带来的模糊性和歧义性；

(2) 具有较强的抽象性，避免了在需求分析阶段对数据结构和算法等细节的详细描述；

(3) 形式规约可以交由计算机自动处理,借助于相应的软件工具对形式规约进行分析、查错、验证以及求精变换等;

(4) 便于对形式规约的各种性质进行推理和证明.

到目前为止,已有数量众多的基于数学的形式规约语言,这些语言的方法和理论基础涉及的领域十分广泛,从集合论、谓词逻辑以及Z语言的模式演算,到OBJ语言的代数基础和范畴论基础.根据对目标软件系统说明方式的不同,形式规约语言大体上可分为两类[1]:基于代数方法的形式规约语言和基于状态方法的形式规约语言.VDM、Z、B都是基于状态方法的,在工业界均得到广泛应用,其中VDM和Z已成为正式的国际标准.

VDM是维也纳开发方法的简称,包括一系列形式规约和开发计算机系统的技术,它源自于IBM维也纳实验室20世纪60年代和70年代的研究工作.VDM语言有一套数学符号系统和基于谓词逻辑和集合理论的证明规则.VDM方法是一种基于形式语义的软件开发方法,规定了大型软件开发的三个阶段:提出要求、形式定义、开发步骤.它也是自顶向下逐步求精的开发方法.VDM的主要特点是中间的形式定义阶段,体现了用形式语义来刻画语言的思想.VDM方法不仅能用于规范,而且能用于设计和对应的实现中,它广泛应用于程序设计语言、数据库、操作系统等应用领域中.

Z语言就是一种流行的基于状态方法的形式规约语言.Z语言是20世纪70年代末至80年代初由英国Oxford大学程序研究组PRG的Jean Raymond Abrial和Bernard Sufrin等人设计的.Z语言是一种目前流行的形式规约语言,它使用相对简单且为人们所熟悉的集合符号、关系、映射、序列、包和一阶谓词逻辑等表示法来描述系统[1-3].Z语言借助于模式来表达系统结构,提供了一种能独立实现的、可推理的系统数学模型,具有精确、简洁、无二义性等优点,有利于保证程序的正确性,尤其适用于无法进行现场调试的高安全性系统的开发.Z利用模式和模式演算对软件系统的结构和行为特征进行抽象描述,其中状态模式可对软件系统的结构特征进行描述,操作模式可对软件系统的行为特征进行描述.Z语言一个主要的特点是可以对Z形式规约进行推理和证明,这种特点使得软件开发人员或用户能够很快找出形式规约的不

一致、不完整之处，提高他们对软件的信心.

B是B-Technologies的简称，包括B-Method，B-Tool和B-Toolkit，由Jean Raymond Abrial等人在20世纪80年代中期开始开发，是一套完整的用于软件工程的方法和工具集[4]. B方法以集合论为理论基础，使用抽象机作为描述系统规范的基本机制，并通过一系列精化过程来得到可执行的代码. 通过不断地删除伪代码中不可执行的语句，引入程序设计语言的典型控制结构，以及将抽象数据结构变换到可编程的数据结构，可以对抽象机进行逐步精化. B方法并不严格地区分规范、设计和实现之间的层次差异，规范和其后续精化都通过抽象机来进行统一的描述. B方法很接近于VDM和Z，它们都是基于状态方法的，并且在工业界均得到广泛应用，其中VDM和Z已成为正式的国际标准. 在本书研究中，我们选取Z语言作为分析语言.

在Z语言的早期发展阶段，它就被应用于工业界和学术界中. 最著名的两个例子是：英国Hursley的IBM公司将Z应用到客户信息和控制系统(custom information and control system，简称CICS)获得成功，将软件开发费用降低了9%；Oxford大学的程序研究组PRG将Z成功地应用于为Transputer Inmos T800版本提供的浮点运算支持. 在2001年Praxis公司发布了Multos信用卡认证系统，该系统是为国际信用卡联盟(万事达)开发，并在全球范围内广泛使用. 该认证系统负责处理信用卡用户存取交易时的身份验证，因此需要非常苛刻的可靠性和商业安全性，必须达到ITSEC E6标准. 另外，出于性能考虑，该系统必须是一个分布式系统，并且使用一些现有的主流技术. Praxis的做法是把系统中与安全相关的关键部分和其他部分隔离开，在系统一级使用Z和CSP进行形式规范，使用SparkTM Ada编写关键部分的代码并自动生成测试代码，对其他部分，如GUI的代码则使用C++编写. 该认证系统大约有100 000行代码，花费了10个人年的时间，每千行代码的出错率为0.04. 这些指标均大大优于使用普通软件工程技术开发类似项目时的平均标准.

目前，人们已经开发了一些针对Z语言的商品化和工业化工具. 如，英国Oxford大学开发的Btool和Fuzz、York大学开发的CADiZ，它能够对Z形式规约进行语法检查和类型检查，并支持排

版，允许用户通过人机交互的方式查看Z形式规约中某些特性[5]；美国芝加哥De Paul大学计算机科学和信息系研制的类型检查工具ZTC，它可以接受LATEX-zed格式和ZSL格式的形式规约[6]；IT-RU研制的支持Z语言模式演算（模式修饰、模式复合、隐藏、求前置条件等）的工具ZedB tool[7]；南京大学研制的COOZ-tools[8]和上海大学研制的Z User Studio[9]等.

1.1.2 研究意义

目前，基于数学理论的形式规约要在软件工业界得到广泛应用还存在一些问题[1-3].

(1) 形式化方法能够减少由误解引起的错误，但却不能保证在编写形式规约时不出现语法错误、不一致、不完整等问题.

(2) 形式规约的正确性证明费时费力，正确性证明一般由手工进行或由人与计算机交互进行.

(3) 形式化方法是模型化、抽象化的静态工具，尽管也可以用静态方法对客观世界的动态方面进行特征描述，但却无法演示客观世界的动态行为.

(4) 形式规约难以阅读. 一方面，大多数软件开发人员习惯于非形式化方法，缺少形式化方法的训练；另一方面，在Z语言中，软件系统的结构和特征都用模式来描述，随着系统的增大，模式也会越来越多，而Z语言中没有有效的机制来管理这些模式，最终导致Z形式规约难以阅读.

(5) 从许多用户的反馈信息来看，Z语言在对大型软件系统进行规约说明时层次不清晰，模块能力不强，从而影响了Z形式规约的可理解性.

(6) 难以识别影响某一状态模式的所有操作模式. 在Z语言中，一个操作模式可能涉及多个状态模式，为了确定影响状态模式的所有操作模式，就需要逐个全面检查操作模式的声明部分，这对于大型软件系统的形式规约来说是不实际的. 对于无法进行现场调试的高安全性系统，Z形式规约的易读性和可靠性显得尤为重要.

如何理解、管理和维护Z形式规约成为了一个十分重要而又迫

切需要解决的问题.我们可以借助程序切片技术把相对大的形式规约分解成一个个相对小的形式规约来帮助人们分析和理解.程序切片是由 Weiser M. 提出的一种重要的程序分析和理解的方法[10],用于从源程序 P 中抽取对程序中特定点 p 上的特定变量 V 有影响的语句和控制条件,组成新的程序(称作切片),然后通过分析切片来分析源程序的行为,其中$\langle p,V\rangle$称为切片标准.如果把程序切片技术引入到形式规约中,将会在一定程度上有助于人们对形式规约进行分析与理解.

虽然程序切片的思想可以引入到形式规约中,但程序和形式规约有着较大的区别.形式规约有两种重要的抽象,即过程抽象和数据抽象.过程抽象描述的是软件系统要实现的功能,而不是如何实现其功能的具体步骤.数据抽象就是在规约中使用集合、关系、映射、序列、包等抽象的数学结构,而不必担心这些结构最终是如何实现的.程序则要考虑这些抽象数学结构的具体实现.因此形式规约切片有着自己的特殊之处,也是一项值得研究的技术.本书在形式规约切片原理及其在提升、定理证明和度量上的应用等方面展开工作,辅助人们阅读 Z 形式规约、识别模式间的相互关系,在一定程度上帮助人们对形式规约的分析与理解.我们认为本书的研究在理论研究和应用实践方面都有着重要价值.

1.1.3 研究目的

本书研究的目的是通过对形式规约切片原理的研究,一定程度上解决形式规约切片及其在提升和度量应用等方面存在的问题,提出基于依赖性分析的 Z 形式规约切片和基于关系演算的 Z 形式规约切片,并在此基础上把 Z 形式规约切片应用到提升、定理证明和度量上.本书还将对基于依赖性分析的 Z 形式规约切片的形式化描述等问题进行深入的探讨.

1.2 国内外研究现状

程序切片是由 Weiser M. 提出的一种重要的程序分析、理解方

法[10]，用于从源程序P中抽取对程序中特定点p上的特定变量V有影响的语句和控制条件，组成新的程序(称作切片)，然后通过分析切片来分析源程序的行为，其中$\langle p,V\rangle$称为切片标准. 二十多年来，人们对它进行了广泛而深入的研究和应用开发[11-37]，提出了子程序切片[38]和过程间切片[39,40]、静态切片[10,41]和动态切片[38,42-44]、前向切片[15,45]和后向切片[15]、传统切片和广义切片[46-59]、面向对象程序切片[60-65]和非面向对象程序切片[66]、顺序程序切片[41]和并发程序切片、条件切片[67]、砍片[68,69]、分解切片[70]、单子语义切片[71]、无定型切片[72,73]等切片方法，并且取得了一些具有理论和应用价值的成果，而且已经开发了一些实用的程序切片工具[11-18,25]，因而也受到了广大程序研究与开发人员的高度重视.

目前常用的计算程序切片的方法主要可分为两种. 一种是Weiser M.提出的基于数据流分析的方法[10]，人们已经开发出了一些工具予以支持. 如，Harrold M. J.等开发的程序分析工具Aristotle就是利用数据流求解程序切片[39]；Atkinson D. C.则开发了一个静态程序切片工具Sprite，利用数据流求解系统依赖图SDG中的summary边，然后在此基础上仍用图的可达性方法计算程序切片[74]. 另一种是Ferrante J.等[75]和Horwitz S.等[40]引入的基于依赖性分析的方法. Ferrante J.等人引入基于程序依赖图PDG的图形可达性算法来计算子程序后向切片[75]，Horwitz S.等引入了前向切片概念及其算法和过程间切片概念以及基于系统依赖图SDG的两阶段图形可达性算法[40]. 基于依赖性分析的程序切片方法已被大家广泛接受[40,41,56,57,59,62,75-78]，而且人们也已经开发出了各种各样的工具予以支持[11,12,14,15,17,18].

虽然人们对程序切片技术进行了长达二十多年的研究，但对形式规约切片(formal specification slicing)的研究起步却比较晚，而且研究还有待于进一步深入[47-51,79].

1.2.1 Oda和Araki的切片方法

Oda T.和Araki K.最先于1993年把程序切片的思想引入到Z形式规约中[51]，定义形式规约切片是形式规约的一部分，该部分定