软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2016,27(3):682−690 [doi: 10.13328/j.cnki.jos.004989]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

面向无穷数据的形式模型综述*

宋 富[1], 吴志林[2]

[1](上海市高可信计算重点实验室(华东师范大学),上海 200062)

[2](计算机科学国家重点实验室(中国科学院 软件研究所),北京 100190)

通讯作者: 吴志林, E-mail: wuzl@ios.ac.cn, http://lcs.ios.ac.cn/~wuzl

摘 要: 无穷数据广泛存在于计算机程序和数据库系统中.受到形式验证与数据库两方面应用需求的推动,面向无穷数据的形式模型已经成为理论计算机科学的研究热点之一.对面向无穷数据的形式模型(逻辑与自动机)进行了相对全面而详细的总结.主要按照不同自动机模型对无穷数据的处理方式加以组织,并关注相关判定问题,即:自动机的非空性问题、语言包含问题以及逻辑的可满足性问题的可判定性与复杂性.

关键词: 无穷数据;自动机;逻辑;非空性;语言包含;可满足性;可判定性;复杂性

中图法分类号: TP301

中文引用格式: 宋富,吴志林.面向无穷数据的形式模型综述.软件学报,2016,27(3):682−690. http://www.jos.org.cn/1000-9825/4989.htm

英文引用格式: Song F, Wu ZL. Survey on formal models to reason about infinite data values. Ruan Jian Xue Bao/Journal of Software, 2016,27(3):682−690 (in Chinese). http://www.jos.org.cn/1000-9825/4989.htm

Survey on Formal Models to Reason about Infinite Data Values

SONG Fu[1], WU Zhi-Lin[2]

[1](Shanghai Key Laboratory of Trustworthy Computing (East China Normal University), Shanghai 200062, China)

[2](State Key Laboratory of Computer Science (Institute of Software, The Chinese Academy of Sciences), Beijing 100190, China)

Abstract: Infinite data exists extensively in computer programs and database systems. Driven by the need from the applications of formal verification and database management, formal models over infinite alphabets are becoming a research focus of theoretical computer science. The main purpose of this article is to do a relatively complete and detailed survey on this topic. The article is organized according to the different mechanisms of automata models to deal with the infinite data values. The main focus is on the decidability and complexity of the related decision problems, that is, the nonemptiness and language inclusion problem of automata, and the satisfiability problem of logics.

Key words: infinite data; automata; logic; nonemptiness; language inclusion; satisfiability; decidability; complexity

1 背 景

在计算机科学中,形式模型一般指用来对特定结构(比如串和树)进行描述、接受、生成和变换的数学模型.形式模型包括逻辑、自动机、文法和重写系统等.对形式模型的研究贯穿计算机科学的发展历程,它奠定了计算机科学的很多分支的理论基础[1].

* 基金项目: 国家自然科学基金(61402179, 61100062, 61272135, 61472474, 61572478); 上海浦江人才计划(14PJ1403200); 上海晨光人才计划(13CG21)

Foundation item: National Natural Science Foundation of China (61402179, 61100062, 61272135, 61472474, 61572478); Shanghai Pujiang Program (14PJ1403200); Shanghai ChenGuang Program (13CG21)

收稿时间: 2015-07-30; 修改时间: 2015-10-20; 采用时间: 2015-11-27; jos 在线出版时间: 2016-01-05

CNKI 网络优先出版: 2016-01-05 16:39:54, http://www.cnki.net/kcms/detail/11.2560.TP.20160105.1639.005.html

图灵机——现代计算机的理论模型,是最早提出和被研究的形式模型之一,它被证明和其他几种形式模型,比如递归函数、λ演算等是等价的.而上下文无关文法是编程语言语法分析的理论基础.近 20 年来,在工业界,尤其在硬件的分析和验证上取得很大成功的自动验证(模型检测)工具,比如 SPIN 和 SMV,建立在无穷(长度为无穷)串和无穷树上的形式模型(逻辑和自动机)的基础上.另外,在数据库领域的针对 Web 上的半结构数据(XML 文档)的查询语言则建立在无秩树(unranked trees)的形式模型(逻辑和自动机)的基础上.而最近很热门的针对图数据(比如 RDF 文档)的路径查询语言则建立在有限自动机和正规表达式的基础上.

逻辑和自动机是最常见的两种形式模型,它们之间形成一种相辅相成的关系:逻辑比较简练,描述性比较强,抽象层次比较高;另一方面,自动机则注重细节,算法性比较强,抽象层级比较低.由于逻辑和自动机的这种互补关系,很多情况下都将逻辑作为一般用户使用的规范语言,而对规范的静态分析(对应于逻辑上的一些判定问题)则都通过自动机来实现.这方面的一个经典例子是时序逻辑 LTL(linear temporal logic)的模型检测问题和可满足性问题都转化为 Büchi 自动机的非空性问题来求解[2,3].

经典的形式模型的字母表一般是预先给定的有限集合Σ.直观上来说,有限字母表可以用来表示并发系统中的事件和 XML 文档中的标签(tag)集.经典的有限字母表上的各种结构(串和树上)上的形式模型(逻辑和自动机)已经被研究人员进行了广泛深入的探讨.Chomsky 层次对有限串上的形式模型按照表达能力从弱到强划分成了 4 种类型:线性文法和有限自动机、上下文无关文法和下推自动机、上下文相关文法和线性界限自动机以及短语文法和图灵机;而且这些层次的理论性质和它们之间的关系也已经被深入地研究[4].在串和树上,有限自动机都被证明和一元二阶逻辑(monadic second-order logic)具有相同的表达能力[5−7].而在无穷串上,一元二阶逻辑被证明和 Büchi 自动机有相同的表达能力[8].另外值得一提的是,串上和树上的自动机代数(半群)理论也已经建立起来.这方面最经典的一个结果是:一个串上的正规语言在一阶逻辑(first-order logic)中可表达当且仅当其语法独异点(syntactic monoid)是非周期的(aperiodic)[9,10].

近 10 年来,由于受形式验证、数据库理论中的 XML 数据和图数据处理这两方面研究的推动,面向无穷数据(或者无穷字母表上)的形式模型的研究成为计算机科学的一个研究热点[11].在面向无穷数据的形式模型中,字母表不再是有限集合Σ,而变成$\Sigma\times D$.这里,D是无穷集合(比如自然数集).这种无穷字母表可以这样直观地来理解:在形式化方法中,如果Σ表示一个事件,则 D 可以表示事件所发生的时间或者事件所属进程(process)或线程(thread)的标识(identifier);而在 XML 文档和图数据的处理中,如果Σ表示 XML 文档的元素(element)(或图数据的结点)的标号(tag),则 D 可以表示元素(或结点)的属性(attribute).

- 形式验证

由于传统的模型检测工具只能验证有穷状态系统,而程序一般含有无穷数据,所以在传统的程序验证中,一般将具有无穷域的数据(比如整数)的程序抽象成有限状态空间系统,然后再使用模型检测工具(只能验证有限状态系统)来进行验证[12−14],而这个抽象的过程一般需要人工干预.

最近,研究人员探讨是否可以直接使用无穷字母表上的可判定的形式模型来对包含无穷数据的程序进行完全自动化地推理和验证:Alur,Cerny 和 Weinstein 考虑了如何将无穷字母表上的可判定结果应用到数组处理程序的验证中[15];Alur 和 Cerny 提出了无穷字母表上的流转换器(streaming transducer)模型,并将其应用到链表操作程序的验证中[16];另外,Abdulla,Holik 和 Jonsson 等人考虑了使用带有序数据约束的森林自动机(forest automata)来对操作动态数据结构的程序进行自动分析与验证.

- 半结构化数据处理

半结构化数据(XML 数据、图数据)的处理是数据库领域近 10 年来很热门的一个方向.XML 文档的最简单的抽象是标号来自有限集合的有序无秩树.这种抽象对于探讨 XML 文档结构的性质很有好处,因为这样可以利用已有的(有穷字母表上的)树上的逻辑和自动机的已有结果[17,18].然而,实际的 XML 文档都含有属性(取值范围为无穷数据域),这些无穷数据不能被一个有限字母表所涵盖,而且在对 XML 文档进行查询、静态分析和变换的过程中都需要考虑这些无穷数据,所以研究人员开始对如何在 XML 文档处理的过程中将无穷数据包括进来进行探讨:Alon,Milo 和 Neven 等人考虑了带数据的 XML 文档的类型检测问题[19];Fan, Libkin 考虑了带数据的

XML 的完整性约束,比如键约束、包含约束等[20];另外 Schwentick 考虑了带数据的 XML 的查询语言 XPath 的查询包含问题[21].图数据的路径查询语言是数据库理论的经典研究领域,最近,研究人员开始探讨在路径查询语言中加入数据约束,但同时保持原有路径查询语言的良好性质.比如:Libkin 和 Vrgoc 提出了正规数据路径查询语言[22];Libkin,Martens 和 Vrgoc 考虑了如何将带数据的 XPath 作为图数据库的查询语言[23].

虽然在很多情况下,在程序验证或者半结构化数据处理中加入数据约束会很快地导致不可判定性,但是研究人员已经发现:在一些情况下,即使将无穷数据约束包括进去,仍然可以得到可判定甚至高效的形式模型.受到这些可判定模型的鼓舞,面向无穷数据的形式模型的研究近年来成为理论计算机科学的热点方向之一.

面向无穷数据的形式模型,从理论上来说是有限字母表上形式模型的自然扩展;而从应用上来说,是与形式验证和半结构化数据处理的应用密切相关的一个领域.所以,开展这方面的研究对于形式化方法和数据库理论这两方面无论从理论还是实践来说都具有重要意义.Segoufin 在 2006 年曾经对无穷字母表上的形式模型做过一次综述[11].与 Segoufin 的综述相比,本文的总结更加全面,而且包含了 2006 年之后的最新进展.

2 研究现状

一般将字母表为$\Sigma\times D$ 的串或树称为数据串或数据树.为了获得面向无穷数据的可判定的形式模型,需要对数据域或者不同位置之间的数据比较的能力进行限制.研究人员最初考虑只能比较数据值是否相等或者只能比较数据值大小的数据域上(不能进行算术运算).在这个限制下,研究人员已经找到了一些数据串上的可判定的形式模型.而且,也探讨了对不同位置之间的数据比较能力进行限制已获得可判定的形式模型.下面我们将对面向无穷数据的形式模型方面的工作进行更具体的介绍.由于自动机模型在面向无穷数据的形式模型的研究中处于较核心的地位,我们将按照自动机模型对无穷数据的不同处理方式来进行介绍,具体区分为存储自动机、数据自动机、石子自动机、变量自动机、符号自动机等.而且,我们主要关注相关判定问题的可判定性与复杂性,比如自动机的非空性问题和语言包含问题以及逻辑公式的可满足性问题.

2.1 存储自动机及其变种、LTL的冻结量词扩展、XPath、正规表达式的带存储器的扩展

Kaminski 和 Francez 在 20 世纪 90 年代中期最早考虑了数据串上的存储自动机(register automata)[24].存储自动机使用有限多个存储器来对数据进行处理,在对一个数据串从左到右扫描的过程中,自动机可以将当前数据与某个存储器中的数据进行(是否相等的)比较,或者将当前数据放入某个存储器.Kaminski 和 Francez 证明了存储自动机的非空性问题是 PSPACE 完全的,而存储自动机的语言包含和等价问题则是不可判定的.后来,Kaminski 和 Tan 进一步考虑了数据树上的存储自动机[25].

Demri 和 Lazic 考虑了在 LTL 中加入冻结量词(freezing quantifier)的扩展[26,27].冻结量词的基本思想是引入原子公式$\downarrow_r$和$\uparrow_r$.$\downarrow_r$的意思是将当前位置的数据值存入存储器 r,而$\uparrow_r$的意思是当前的数据值和存储器 r 中的数据值相等.他们证明了如果含有两个存储器,则 LTL 的带冻结量词的扩展的可满足性问题是不可判定的;而 LTL 的只含有一个存储器的带冻结量词的扩展是可判定的.为了获得可判定性的结果,他们对 Kaminski 和 Francez 提出的存储自动机进行了扩展,考虑了交替存储自动机(alternating register automata).他们证明了:如果只含有一个存储器,则交替存储自动机的非空性问题是可判定的.Jurdzinski 和 Lazic 考虑了数据树上的带一个存储器的交替存储自动机(alternating 1-register tree automata),证明了其非空性问题是可判定的.从这个结论,他们推出 XPath 的不含逆向算子的、带有(受限)数据比较的子集的可满足性是可判定的[28].而 Figueira 则进一步对数据树上的带一个存储器的交替存储自动机进行了扩展,证明了所谓的前向 XPath(forward XPath),即,XPath 不含有逆向算子的、带有不受限的数据比较的子集的可满足性问题是可判定的[29].另外,Figueira 和 Segoufin 提出了自下而上的数据树上的带一个存储器的交替自动机,证明了其非空性问题的可判定性,并用这个结论证明了垂直 XPath(vertical XPath),即:XPath 的只包含垂直算子(父母-子女、祖先-子孙关系),而不含有水平算子(兄弟姐妹关系)的子集的可判定性[30].

以上提到的存储自动机模型及其变种都只能比较数据值之间的相等关系.而在实际应用中,比如在 XML 文档处理中,数据一般都有大小,比如日期、年龄等.研究人员也对能够比较数据值大小的存储自动机及其变种进

行了研究.Figueira,Hofman 和 Lasota 建立了时间自动机(timed automata)和能够比较数据值大小的存储自动机之间的联系[31],使得时间串(timed words)上的形式模型与能够比较数据值大小的数据串上的形式模型之间的可判定性和复杂性结果可以互推,比如:从时间自动机的非空性问题是 PSPACE 完全的结果,可以推出能够比较数据值大小的存储自动机的非空性问题是 PSPACE 完全的;而数据串上的能够比较数据值大小的带一个存储器的交替存储自动机的可判定性可以从带一个时钟的交替时间自动机的可判定性[32]推出.

图数据库上的查询语言是数据库理论的经典领域,由于受在线社交网络、生物信息和语义万维网应用的推动,最近,图数据库上的查询语言又受到研究人员的广泛关注.图数据库是边上带标号的有向图.正规路径查询语言是使用正规表达式来描述图数据库中路径上的边的标号序列所需要满足的约束查询语言.带数据的图数据库是每个结点上包含取值范围为无穷域的属性的图数据库.Libkin 和 Vrgoc 使用正规表达式的带存储器的扩展来描述带数据的图数据库上的路径约束[22].Kaminski 与 Tan 最早考虑了正规表达式的带存储器的扩展[33],而 Libkin 和 Vrgoc 则提出了适合对带数据的图数据库进行查询的正规表达式的带存储器的两种扩展[34],并且详细讨论了这些查询在图数据库上进行求解的复杂度[22].Libkin,Tan 和 Vrgoc 等人进一步考虑了正规表达式的带存储器的另外一种扩展[35].

2.2 数据自动机、一阶逻辑的子集

Bojanczyk,Muscholl 和 Schwentick 等人在 2006 年提出了只能比较数据值是否相等的数据串上的数据自动机(data automata)的概念[36].数据自动机包含两个部件:一个字符到字符的有限状态转换器(letter-to-letter transducer)$A:\Sigma^*\to\Gamma^*$和一个字母表为Γ的有限自动机 B.数据串(w,d)能够被数据自动机(A,B)所接受,当且仅当 A 能够从 w 生成 w',使得对于每一个(w,d)中出现的数据值 v,w'的对应于 v 出现的位置的子串都能够被 B 所接受.他们证明了数据自动机的非空性问题的可判定性,并用其证明了数据串上的一阶逻辑的含有两个变量的子集(FO2)的可满足性问题是可判定的.然而,他们给出的算法具有非初等(nonelementary)的复杂度.同时,他们也证明了数据串上的一阶逻辑的含有 3 个变量的子集的可满足性问题是不可判定的.而且,他们考虑了只能比较数据值是否相等的无秩数据树上的一阶逻辑的含有两个变量的子集,证明了如果一阶逻辑的词汇表(vocabulary)被限制为局部的,即,只含有无秩有序树上的水平和垂直后继而不包含它们的传递闭包, FO2 的可满足性问题是可判定的[37].

Alur,Cerny 和 Weinstein 提出了数据自动机的一种扩展,证明了该扩展与数据自动机在表达能力上是等价的,并将其应用到对数组处理程序的验证中[15].而 Bojanczyk 和 Lasota 则提出了分类自动机(class automata),对已有的数据串和数据树上的自动机模型进行了统一,比如:数据自动机和前面提到的带一个存储器的交替存储自动机都可以看成类自动机的特殊情况;而且,数据树上的分类自动机的表达能力强于 XML 文档上的(包含数据比较的)查询语言 XPath[38].David,Libkin 和 Tan 针对文献[36]中考虑的数据串上 FO2 的一个子集,通过归约到 Presburger 算术提出了具有更低复杂度的可满足性判定算法[39].Wu 提出了数据自动机的一种可判定扩展,并且建立了与优先计数自动机(priority counter automata)的联系[40];而且,Wu 提出了可交换数据自动机的概念,证明其非空性问题具有初等时间的复杂度[41].

Schwentick 和 Zeume 考虑了能够比较数据值大小(但每个位置只有一个数据)的数据串上 FO2 的可判定性,他们证明了:在比较数据值大小的数据串上,如果 FO2 的词汇中只含有串位置的序关系(<),而没有后继关系(successor relation,+1),则该逻辑的可满足性可以在指数空间(EXPSPACE)内判定[42].而 Tan 基于数据自动机的思想,提出了数据树上的能够比较数据值大小的一个自动机模型,并且证明了其非空性问题是可判定的[43].

以上的结果都假设数据串或数据树的每个位置只含有一个数据值.Kara,Schwentick 和 Zeume 对每个位置包含多个数据属性(数据只能比较是否相等)的数据串上的时序逻辑也进行了探讨:他们定义了 BD-LTL,即,LTL 的针对单个数据属性的导航(navigation)扩展;通过将多个数据属性用单个数据属性进行编码,他们将 BD-LTL 的可满足性问题归约为数据自动机的非空性问题,从而证明了其可满足性问题是可判定的[44].而且,Habermehl 等人考虑了 ND-LTL,即,LTL 的针对多个数据属性的导航扩展,他们证明了 ND-LTL 的两个子集的可满足性问题是可判定的[45].在证明过程中,他们引入了嵌套数据自动机的概念,并且证明了其非空性问题是可判定的.

2.3 石子自动机

Neven,Schwentick 和 Vianu 等人提出了的石子自动机(pebble automata)的概念[46].石子自动机使用石子(pebble)而不是存储器来处理数据:在对一个数据串扫描的过程中,自动机可以将当前位置的数据与某个已用石子所在位置的数据进行比较来决定是将某个空闲石子放到当前位置,还是将放在当前位置的石子移走.然而,由于石子比存储器有更大的灵活性,Neven,Schwentick 和 Vianu 证明了具有两个石子的石子自动机的非空性是不可判定的.

为了获得可判定性的结果,Tan 提出了弱石子自动机的概念,他证明了只含有两个石子的弱石子自动机的非空性问题是可判定的.而且,他引入了上视图(top-view)石子自动机的概念,证明了其非空性问题是可判定的(对使用的石子数目没有限制)[47].

2.4 变量自动机、时序逻辑的带变量的扩展、数据树模式

Grumberg,Kupferman 等人提出了在 Büchi 自动机的字母表中加入变量的方式来定义无穷数据串上的变量自动机(variable automata),而且证明了这个模型的非空性问题是 NL 完全的[48].Grumberg,Kupferman 等人提出了 LTL 的带变量的扩展,对其可满足性问题与模型检测问题的可判定性与复杂性进行了初步的探讨[49,50].Grumberg,Kupferman 等人考虑的公式限制在前束范式(prenex normal form)上,即,量词只能出现在公式的最前面.

Song 和 Wu 对时序逻辑的带变量的扩展进行了比较全面的探讨,他们同时考虑了 LTL 和 CTL 的带变量的扩展,而且量词可以出现在公式的任何位置,不一定是前束范式.他们比较准确地界定了可满足性问题与模型检测问题的可判定性边界[51].与此相关的是针对参数化系统进行规约与验证的带标号的时序逻辑(indexed temporal logic),其中的变量解释在进程标号集合上.最近,Chen,Song 等人对带标号的线性时序逻辑(indexed LTL)的可满足性问题的可判定性与复杂性进行了详尽的探讨[52].

类似于关系数据库,研究人员也对于数据树上的合取查询(conjunctive query)进行了探讨.合取查询是数据库中的一种最基本的查询,即,多个原子公式的合取.首先,David 对数据树模式(data tree patterns,可以看作,是一种特殊的合取查询)的模型检测与可满足性问题的复杂性进行了详细的分析[53].几乎同时,Bjorklund,Martens 和 Schwentick 对合取查询在数据树上的可满足性问题的复杂性进行了探讨[54].另外,研究人员也提出了基于数据树模式的数据树上的重写系统(rewriting system),而且探讨了如何对该重写系统进行静态分析和验证[55,56].

2.5 符号自动机、符号转换器

符号自动机(symbolic automata)的概念最早是 Watson 提出的针对大数据或无穷数据字母表的有限自动机的扩展[57],其基本思想是:将有限自动机中状态迁移规则的确定原子字符替换成某(无穷)字母表有效布尔代数(effective Boolean algebras)公式(formula 或 predicate),当输入的字符满足状态迁移规则标记的公式时,状态迁移则可以发生.伴随着符号自动机的提出和研究的深入,研究人员对于符号转换器(symbolic transducer)也进行了深入的研究.

在自然语言处理领域,Noord 和 Gerdemann 对符号自动机和符号转换器进行了探讨,并将有限自动机/转换器中的大部分基本布尔运算和判定过程扩展到了符号自动机和符号转换器[58].在验证领域,Pill 和 Cimatti 等人分别引入了类似的符号自动机,用于表达顺序扩展正则表达式(sequential extended form of regular expressions),并分别设计实现了基于 DNF(disjunctive normal form)和 BDD(binary decision diagraph)的符号自动机软件包用于系统验证[59,60].但 Pill 和 Cimatti 等人提出符号自动机的目的是为了简洁地表述大字母表的有限自动机,尚未讨论无穷字母表的场景,因此其表达能力等同于传统的有限自动机.

Veanes 等人在文献[61]中首次将有限自动机和 Satisfiability Modulo Theory(SMT)结合,提出了真正意义上的符号自动机.该符号自动机的状态迁移规则由原来的原子字符替换成了 SMT 中的公式,而公式表示满足该公式的所有字符,当输入的字符满足公式时,状态迁移才能发生,因此可以刻画无穷字母表上的正则语言.他们将大部分布尔操作从有限自动机扩展到了基于 SMT 的符号自动机,并设计实现了基于 SMT 约束求解器 Z3 的符

号自动机软件包 Rex[61].当使用的 SMT 理论的可满足性可判定时,符号自动机的判断过程包括非空性问题和包含问题都是可判定的.之后,Veanes 等人将符号自动机的思想扩展到了符号下推机和符号下推转换器[62−64],这些模型的非空性问题也取决于 SMT 理论的可满足性问题的判定性.为了分析 Javascript 恶意软件,Veanes 等人将存储器(registers)引入到符号自动机和符号转换器,提出了扩展符号自动机/转换器(extended symbolic automata/transducers)[65].扩展符号自动机/转换器具有更强的表达能力,可以分析字符串编码器(encoder)和解码器(decoders),而符号自动机/转换器仅能分析符号编码器.扩展符号自动机/转换器一般是不可判定的.为了获得可判定性,他们也对扩展符号自动机/转换器的子模型进行了探讨.另外,D'Antoni 和 Veanes 提出了一种带有界预见能力(lookahead)的符号自动机/转换器[66].而且,D'Antoni 和 Alur 提出了符号可视化下推自动机(symbolic visibly pushdown automata)——即可视化下推自动机的基于 SMT 的符号化扩展.与符号自动机相比,即可视化下推自动机具有更强的表达能力,但弱于符号下推自动机.即可视化下推自动机与可视化下推自动机一样,拥有符号下推自动机所不具备的良好的闭包性和可判定性[67].符号自动机和符号转换器以及他们的扩展在 XML/HTML 分析、字符串净化函数分析、Javascript 恶意软件、自然语言处理和入侵检测等领域有了广泛的应用[65].

符号树自动机和转换器(symbolic tree automata/transducers)由 Veanes 和Bjørner于 2011 年提出[68].类似带有界预见能力(lookahead)的符号自动机/转换器,D'Antoni 和 Veanes 等人提出了具有正则预见能力(regular lookahead)的符号树自动机/转换器[69].而且,Veanes 进一步深入研究了其布尔闭包性质和判定程序[70].

2.6 针对动态内存的形式模型的带数据扩展

对操作动态内存的程序行为进行分析与推理,是程序分析与验证领域的一个难点问题.对此,研究人员已经提出了不同的形式模型,比如分离逻辑、树正规模型检测(tree regular model checking)、森林自动机(forest automata)等.目前,已有的对操作动态内存的程序分析与验证的绝大部分工作集中在对动态内存的形状性质的描述与推理上.但一般的动态数据结构都同时需要考虑形状性质和数据约束,比如用链表来实现排序算法的程序.一方面我们需要在排序算法执行完之后还是保持链表的结构;另一方面,我们需要保证在排序算法执行完之后数据的有序性.为了同时对形状性质与数据约束进行推理,需要对已有的形式模型进行扩展.研究人员已经在这方面进行了初步的探讨.

分离逻辑是 2000 年左右由 Reynolds,O'Hearn 和 Yang 等人提出的对动态内存进行规约与推理的程序逻辑[71−73].分离逻辑本质上是在一阶逻辑的基础上加入两个分离算子,即,分离合取(separating conjunction)与分离蕴涵(separating implication)算子,从而实现对操作动态内存的程序行为的局部推理.由于很多动态数据结构(比如链表、二叉树等)都需要归纳定义,因此研究人员也考虑了向一阶分离逻辑中加入归纳定义之后的判定程序.Bansal,Brochenin 和 Lozes 考虑了一阶分离逻辑的带有序数据约束的扩展,确定了其可满足性问题的可判定性的边界[74].另外,Chin,David 等人、Chu,Jaffar 等人以及 Enea,Sighireanu 等人分别提出了带归纳定义的分离逻辑的带数据约束的扩展的蕴涵问题的可靠但非完备的启发式判定算法[75−77].

森林自动机是由 Habermehl,Holik 等人提出的对操作动态数据结构的程序进行自动化分析与验证的一种自动机模型[78].类似于分离逻辑,森林自动机只能对形状性质进行推理.最近,Abdulla 和 Holik 等人也考虑了森林自动机的带有序数据约束的扩展,用于对排序算法、嵌套链表、二叉排序树等数据结构进行完全功能正确性的验证[79].

Alur 和 Cerny 提出了所谓的数据串上的流转化器(streaming transducer),用来对链表操作程序进行验证[16]:他们用流转化器证明了循环不嵌套的操作链表的某一类命令式程序的功能等价性是可判定的.

3 结 语

无穷字母表上的形式模型是由形式验证和 XML 文档处理的研究推动的理论计算机科学的最新热点领域,这方面的研究与有限字母表上的形式模型的研究相比需要新的思路和方法.虽然在这方面已经有不少工作,但是整体来说这方面的研究仍然处于起步阶段,还存在很多未解决的问题.另外,该领域目前的大部分工作还处于理论探讨阶段,离实际应用距离还比较远.该领域的研究工作需要更注重考虑如何更好地和实际应用相结合,提

高判定算法的效率,开发相应的原型工具,并进行实例研究.

References:

[1] van Leeuwen J, ed. Handbook of Theoretical Computer Science, Vol.B: Formal Models and Semantics. Elsevier and MIT Press, 1990.

[2] Wolper P, Vardi M, Sistla A. Reasoning about infinite computation paths. In: Proc. of the FOCS 1983. 1983. 185−194. [doi: 10.1109/SFCS.1983.51]

[3] Varidi M, Wolper P. An automata-theoretic approach to automatic program verification. In: Proc. of the LICS 1986. 1986. 332−344.

[4] Hopcroft J, Ullman J. Introduction to Automata Theory, Languages and Computation. Addison-Wesley, 1979.

[5] Büchi J. Weak second-order arithmetic and finite automata. Mathematical Logik Quarterly Mlq, 1960,6:66−92. [doi: 10.1002/malq.19600060105]

[6] Elgot C. Decision problems of finite automata design and related arithmetic. IEEE Trans. on American Mathematical Society, 1961, 98:21−52. [doi: 10.2307/1993511]

[7] Thatcher JW, Wright JB. Generalized finite automata theory with an application to a decision problem of second-order logic. Theory of Computing Systems, 1968,2(1):57−81. [doi: 10.1007/BF01691346]

[8] Buchi J. On a decision method in restricted second-order arithmetic. In: Proc. of the Int'l Congress for Logic, Methodology and Philosophy of Science. Standford University Press, 1962. 1−11. [doi: 10.1007/978-1-4613-8928-6_23]

[9] Schutzenberger MP. On finite monoids having only trivial subgroups. Information and Computation, 1965,8:190−194. [doi: 10.1016/S0019-9958(65)90108-7]

[10] McNaughton R, Papert S. Counter-Free Automata. MIT Press, 1971.

[11] Segoufin L. Automata and logics for words and trees over an infinite alphabet. In: Proc. of the CSL 2006. 2006. 41−57. [doi: 10.1007/11874683_3]

[12] Balaban I, Pnueli A, Zuck L. Shape analysis by predicate abstraction. In: Proc. of the VMCAI 2005. 2005. 164−180. [doi: 10. 1007/978-3-540-30579-8_12]

[13] Gopan D, Reps T, Sagiv M. A framework for numeric analysis of array operations. In: Proc. of the POPL 2008. 2008. 338−350. [doi: 10.1145/1040305.1040333]

[14] Gulwani S, McCloskey B, Tiwari A. Lifting abstract interpreters to quantified logical domains. In: Proc. of the POPL 2008. 2008. 235−246. [doi: 10.1145/1328438.1328468]

[15] Alur R, Cerny P, Weinstein S. Algorithmic analysis of array-accessing programs. In: Proc. of the CSL 2009. LNCS 5771, 2009. 86−101. [doi: 10.1007/978-3-642-04027-6_9]

[16] Alur R, Cerny P. Streaming transducers for algorithmic verification of single-pass list-processing programs. In: Proc. of the POPL 2011. 2011. 599−610. [doi: 10.1145/1926385.1926454]

[17] Neven F. Automata, logic, and XML. In: Proc. of the CSL 2002. 2002. 2−26.

[18] Vianu V. A Web odyssey: From codd to XML. In: Proc. of the PODS 2001. 2001. 1−15. [doi: 10.1145/776985.776999]

[19] Alon N, Milo T, Neven F, Suciu D, Vianu V. XML with data values: Typechecking revisited. Journal of Computer and System Sciences, 2003,66(4):688−727. [doi: 10.1016/S0022-0000(03)00032-1]

[20] Fan W, Libkin L. On XML integrity constraints in the presence of DTDs. Journal of ACM, 2002,49(3):368−406. [doi: 10.1145/567112.567117]

[21] Schwentick Th. XPath query containment. SIGMOD Record, 2004,33(1):101−109. [doi: 10.1145/974121.974140]

[22] Libkin L, Vrgoc D. Regular path queries on graphs with data. In: Proc. of the ICDT 2012. 2012. 74−85. [doi: 10.1145/2274576.2274585]

[23] Libkin L, Martens W, Vrgoc D. Querying graph databases with XPath. In: Proc. of the ICDT 2013. 2013. 129−140. [doi: 10.1145/2448496.2448513]

[24] Kaminski M, Francez N. Finite memory automata. Theoretical Computer Science, 1994,134(2):329−363. [doi: 10.1016/0304-3975(94)90242-9]

[25] Kaminski M, Tan T. Tree Automata Over Infinite Alphabets. Pillars of Computer Science, 2008. [doi: 10.1007/978-3-540-78127-1_21]

[26] Demri S, Lazic R. LTL with the freeze quantifer and register automata. In: Proc. of the LICS 2006. 2006. 17−26. [doi: 10.1109/LICS.2006.31]

[27] Demri S, Lazic R, Nowak D. On the freeze quantifier in constraint LTL. In: Proc. of the TIMES 2005. 2005. 113−121. [doi: 10.1109/TIME.2005.28]

[28] Jurdzinski M, Lazic R. Alternating automata on data trees and XPath satisfiability. ACM Trans. on Computational Logic, 2011, 12(3):19. [doi: 10.1145/1929954.1929956]

[29] Figueira D. Forward-XPath and extended register automata on data-trees. In: Proc. of the ICDT 2010. 2010. 231−241. [doi: 0.1145/1804669.1804699]

[30] Figueira D, Segoufin L. Bottom-Up automata on data trees and vertical XPath. In: Proc. of the STACS 2011. 2011. 93−104. [doi: 10.4230/LIPIcs.STACS.2011.93]

[31] Figueira D, Hofman P, Lasota S. Relating timed and register automata. In: Proc. of the EXPRESS 2010, Vol.41. EPTCS, 2010. 61−75. [doi: 10.1017/S0960129514000322]

[32] Lasota S, Walukiewicz I. Alternating timed automata. In: Proc. of the FSTTCS 2005, Vol.3441. 2005. 250−265. [doi: 10.1007/ 978-3-540-31982-5_16]

[33] Kaminski M, Tan T. Regular expressions for languages over infinite alphabets. Fundamenta Informaticae, 2006,69(3):301−318. [doi: 10.1007/978-3-540-27798-9_20]

[34] Libkin L, Vrgoc D. Regular expressions for data words. In: Proc. of the LPAR 2012. 2012. 274−288. [doi: 10.1007/978-3-642-28717-6_22]

[35] Libkin L, Tan T, Vrgoc D. Regular expressions with binding over data words for querying graph databases. In: Proc. of the DLT 2013. 2013. 325−337. [doi: 10.1007/978-3-642-38771-5_29]

[36] Bojanczyk M, Muscholl A, Schwentick T, Segoufin L, David C. Two-Variable logic on words with data. In: Proc. of the LICS 2006. 2006. 7−16. [doi: 10.1109/LICS.2006.51]

[37] Bojańczyk M, David C, Muscholl A, Schwentick T, Segoufin L. Two-Variable logic on data trees and XML reasoning. In: Proc. of the PODS 2006. 2006. 10−19. [doi: 10.1145/1142351.1142354]

[38] Bojanczyk M, Lasota S. An extension of data automata that captures XPath. In: Proc. of the LICS 2010. 2010. 243−252. [doi: 10.1109/LICS.2010.33]

[39] David C, Libkin L, Tan T. On the satisfiability of two-variable logic over data words. In: Proc. of the LPAR 2010. 2010. 248−262. [doi: 10.1007/978-3-642-16242-8_18]

[40] Wu ZL. A decidable extension of data automata. In: Proc. of the GandALF 2011. 2011. 116−130. [doi: 10.4204/EPTCS.54.9]

[41] Wu ZL. Commutative data automata. In: Proc. of the CSL 2012. 2012. 528−542. [doi: 10.4230/LIPIcs.CSL.2012.528]

[42] Schwentick T, Zeume T. Two-Variable logic with two order relations. In: Proc. of the CSL 2010. 2010. 499−513. [doi: 10.1007/978-3-642-15205-4_38]

[43] Tan T. An automata model for trees with ordered data values. In: Proc. of the LICS 2012. 2012. 586−595. [doi: 10.1109/ LICS. 2012.69]

[44] Kara A, Schwentick T, Zeume T. Temporal logics on words with multiple data values. In: Proc. of the FSTTCS 2010. 2010. 481−492. [doi: 10.4230/LIPIcs.FSTTCS.2010.481]

[45] Decker N, Habermehl P, Leucker M, Thoma D. Ordered navigation on multi-attributed data words. In: Proc. of the CONCUR 2014. 2014. 497−511. [doi: 10.1007/978-3-662-44584-6_34]

[46] Neven F, Schwentick T, Vianu V. Finite state machines for strings over infinite alphabets. ACM Trans. on Computational Logic, 2004, 15(3):403−435. [doi: 10.1145/1013560.1013562]

[47] Tan T. On pebble automata for data languages with decidable emptiness problem. Journal of Computer and System Sciences, 2010, 76(8):778−791. [doi: 10.1016/j.jcss.2010.03.004]

[48] Grumberg O, Kupferman O, Sheinvald S. Variable automata over infinite alphabets. In: Proc. of the LATA 2010. 2010. 561−572. [doi: 10.1007/978-3-642-13089-2_47]

[49] Grumberg O, Kupferman O, Sheinvald S. Model checking systems and specifications with parameterized atomic propositions. In: Proc. of the ATVA 2012. 2012. 122−136. [doi: 10.1007/978-3-642-33386-6_11]

[50] Grumberg O, Kupferman O, Sheinvald S. An automata-theoretic approach to reasoning about parameterized systems and specifications. In: Proc. of the ATVA 2013. 2013. 397−411. [doi: 10.1007/978-3-319-02444-8_28]

[51] Fu Song, Zhilin Wu. Extending temporal logics with data variable quantifications. In: Proc. of the FSTTCS 2013. 2013. 253−265. [doi: 10.4230/LIPIcs.FSTTCS.2014.253]

[52] Chen T, Fu S, Wu ZL. On the satisfiability of indexed linear temporal logics. In: Proc. of the CONCUR 2015. 2015. 254−267. [doi: 10.4230/LIPIcs.CONCUR.2015.254]

[53] David C. Complexity of data tree patterns over XML documents. In: Proc. of the MFCS 2008. 2008. 278−289. [doi: 10.1007/978-3-540-85238-4_22]

[54] Bjorklund H, Martens W, Schwentick T. Optimizing conjunctive queries over trees using schema information. In: Proc. of the MFCS 2008. 2008. 132−143. [doi: 10.1007/978-3-540-85238-4_10]

[55] Abiteboul S, Segoufin L, Vianu V. Static analysis of active XML systems. In: Proc. of the PODS 2008. 2008. 221−230. [doi: 10.1145/1620585.1620590]

[56] Genest B, Muscholl A, Wu ZL. Verifying recursive active documents with positive data tree rewriting. In: Proc. of the FSTTCS 2010. 2010. 469−480. [doi: 10.4230/LIPIcs.FSTTCS.2010.469]

[57] Bruce W. Watson: Implementing and using finite automata toolkits. Natural Language Engineering, 1996,2(4):295−302. [doi: 10.1017/S135132499700154X]
[58] van Noord G, Gerdemann D. Finite state transducers with predicates and identities. Grammars, 2001,4(3):263−286. [doi: 10.1023/A:1012291501330]
[59] Pill I. Requirements engineering and efficient verification of PSL properties [Ph.D. Thesis]. Graz Univeristy of Technology, 2008.
[60] Cimatti A, Mover S, Roveri M, Tonetta S. From sequential extended regular expressions to NFA with symbolic labels. In: Proc. of the CIAA 2010. 2010. 87−94. [doi: 10.1007/978-3-642-18098-9_10]
[61] Veanes M, Grigorenko P, de Halleux P, Tillmann N. Rex: Symbolic regular expression explorer. In: Proc. of the ICST 2010. 2010. 498−507. [doi: 10.1109/ICST.2010.15]
[62] Veanes M, Bjørner N, de Moura LM. Symbolic automata constraint solving. In: Proc. of the LPAR 2010. 2010. 640−654. [doi: 10.1007/978-3-642-16242-8_45]
[63] Hooimeijer P, Livshits B, Molnar D, Saxena P, Veanes M. Fast and precise sanitizer analysis with BEK. In: Proc. of the USENIX Security Symp. 2011.
[64] Veanes M, Bjørner N. Symbolic automata: The toolkit. In: Proc. of the TACAS 2012. 2012. 472−477. [doi: 10.1007/978-3-642-28756-5_33]
[65] Veanes M, Hooimeijer P, Livshits B, Molnar D, Bjørner N. Symbolic finite state transducers: Algorithms and applications. In: Proc. of the POPL 2012. 2012. 137−150. [doi: 10.1145/2103656.2103674]
[66] D'Antoni L, Veanes M. Static analysis of string encoders and decoders. In: Proc. of the VMCAI 2013. 2013. 209−228.
[67] D'Antoni L, Alur R. Symbolic visibly pushdown automata. In: Proc. of the CAV 2014. 2014. 209−225. [doi: 10. 1007/978-3-642-35873-9_14]
[68] Veanes M, Bjørner N. Symbolic tree transducers. In: Proc. of the Ershov Memorial Conf. 2011. 377−393. [doi: 10. 1007/978-3-642-29709-0_32]
[69] D'Antoni L, Veanes M, Livshits B, Molnar D. Fast: A transducer-based language for tree manipulation. In: Proc. of the PLDI 2014. 2014. 40. [doi: 10.1145/2594291.2594309]
[70] Veanes M, Bjørner N. Symbolic tree automata. Information Processing Letters, 2015,115(3):418−424. [doi: 10.1016/j.ipl.2014. 11.005]
[71] Ishtiaq S, O'Hearn P. BI as an assertion language for mutable data structures. In: Proc. of the Principles of Programming Languages (POPL 2001). 2014. 14−26. [doi: 10.1145/360204.375719]
[72] Yang H. Local reasoning for stateful programs. Technical Report, UIUCDCS-R-2001-2227, University of Illinois at Urbana-Champaign, 2001.
[73] Reynolds JC. Separation logic: A logic for shared mutable data structures. In: Proc. of the LICS 2002. 2002. 55−74. [doi: 10.1109/LICS.2002.1029817]
[74] Bansal K, Brochenin R, Lozes É, Shapes B. Lists with ordered data. In: Proc. of the FOSSACS 2009. 2009. 425−439. [doi: 10.1007/978-3-642-00596-1_30]
[75] Chin WN, David C, Nguyen HH, Qin SC. Automated verification of shape, size and bag properties via user-defined predicates in separation logic. Science of Computer Programming, 2012,77(9):1006−1036. [doi: 10.1016/j.scico.2010.07.004]
[76] Chu DH, Jaffar J, Trinh MT. Automatic induction proofs of data-structures in imperative programs. In: Proc. of the PLDI 2015. 2015. 457−466. [doi: 10.1145/2737924.2737984]
[77] Enea C, Sighireanu M, Wu ZL. On automated lemma generation for separation logic with inductive definitions. In: Proc. of the ATVA 2015 2015. 80−96. [doi: 10.1007/978-3-319-24953-7_7]
[78] Habermehl P, Holík L, Rogalewicz A, Simácek J, Vojnar T. Forest automata for verification of heap manipulation. Formal Methods in System Design, 2012,41(1):83−106. [doi: 10.1007/s10703-012-0150-8]
[79] Abdulla PA, Holík L, Jonsson B, Lengál O, Trinh CQ, Vojnar T. Verification of heap manipulating programs with ordered data by extended forest automata. In: Proc. of the ATVA 2013. 2013. 224−239. [doi: 10.1007/s00236-015-0235-0]

宋富(1983－),男,浙江宁波人,博士,讲师,主要研究领域为模型检测,软件分析与验证,自动机与逻辑,计算机安全.

吴志林(1980－),男,博士,副研究员,主要研究领域为自动机与逻辑,软件的形式分析与验证,数据库理论.