基于 MathML 的数学表达式等价性的研究*

刘 峰，袁春风

(南京大学 计算机软件新技术国家重点实验室，江苏 南京 210093)

摘 要：简介了数学标记语言 MathML 内容编码的规范，并提出了判断两个基于 MathML 内容编码的数学表达式等价性的方法。该方法通过建立数学规则库，对两个数学表达式进行树型模式匹配和算式变换来实现。

关键词：MathML；树型模式匹配；数学表达式；算式变换

中图法分类号：TP391 文献标识码：A 文章编号：1001-3695(2004)11-0054-03

Research on the Equipollence of MathML-based Math Expressions

LIU Feng, YUAN Chun-feng

(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing Jiangsu 210093, China)

Abstract: Describes the specification of MathML content coding and proposes a feasible approach for judging the equipollence of MathML-based math expressions. This approach is implemented by rule set building, tree pattern matching and math expression transformation.

Key words: MathML; Tree Pattern Matching; Math Expression; Expression Transformation

随着 Internet 的迅速发展，人们越来越多地通过 Web 方式进行信息的发布、获取和管理。在交流的信息中，数学表达式是一个不可或缺的内容，尤其在远程教育和数学科研领域。随着 XML 技术规范的不断发展和完善，1998 年 4 月国际互联网组织 W3C 中的数学工作小组公布了 MathML，到 2002 年 12 月发布了 MathML2.0 第二版[1]。它的出现为数学表达式提供了表达标准，因而推动了与数学表达式处理相关的应用的发展。MathML 作为 Internet 上一种重要的数学表达式标准，建立在 MathML 基础之上的语义处理也将会有广泛的应用，例如数学表达式的显示、验证和查找等。

本文实现的是试题自动批改系统中的一个子模块，主要探讨如何判断两个基于 MathML 内容编码的数学表达式是否等价。

1 MathML 标准

MathML 是在 XML 的基础上制定的。它继承了 XML 的简洁1 可扩充以及灵活性的特点，提供了统一的描述1 包装1 保存以及传递数学表达式的格式标准，不同的应用之间可以基于这个标准作数学信息的共享和交换。

MathML 有两类编码体系：展现编码(Presentation Code)和内容编码(Content Code)。展现编码用于对数学表达式的显示形式进行编码描述，不含语义信息。内容编码则对数学表达式根据语义进行编码，使其便于被计算机解析和处理。本文研究对数学表达式的语义信息的处理，待处理的数学表达式都以内容编码方式表示。

MathML 的内容编码包含约 120 个元素和 12 个属性，其中大部分元素与数学表达式的语义相关，也可以嵌套。所有元素大致分为三类：①表示各种运算符、关系式和数学函数的空元素。120 个元素中绝大多数都是这类元素，例如 < plus/ > 表示相加。这类元素使用时往往与功能元素 < apply > 相结合构成表达式。②用来表示各种数据类型的元素。例如 < matrix > 表示矩阵，< set > 表示集合等。其中最常用的是表示变量符号的 < ci > 和表示数字常量的 < cn >。③用来实现一些特定功能的元素。例如 < mrow > 表示其中包含的内容在一行子式里；< math > 表示一段 MathML 的起点和终点，所有基于 MathML 的数学表达式都以 < math > 作为根节点；< apply > 表示将函数作用到参数上，或是根据已有的数学对象构造新的数学对象。当 < apply > 用来表示将函数作用到参数上时，其第一个子元素指示出函数名，余下的子元素则按顺序对应到各个参数。例如$(a+b)^2$ 的内容编码如下：

```
< math >
< mrow >
< apply >
< power/ >
< apply >
< plus/ >
< ci > a < /ci >
< ci > b < /ci >
< /apply >
< cn > 2 < /cn >
< /apply >
< /mrow >
< /math >
```

上述 MathML 内容编码可以很自然地解析为一棵中缀表达式树(图 1)，这棵树按照后根遍历即可得到该数学表达式的逆波兰表示，其中叶子节点都是常量或变量，非叶子节点是数学运算符或函数，子节点都是父节点的参数。运算级别较高的运算符或函数排在较底层的位置，后根遍历时优先处理；最顶层的语义节点是表达式中运算级别最低的运算符或函数。

收稿日期：2003-07-06；修返日期：2003-11-21

基金项目：国家“十五”科技攻关资助项目(2001BA101A06-01)

2　数学表达式的等价判断

精确性是数学语言的一个重要特点, 每个数学概念、符号、规则都有其精确的含义, 没有含糊不清或易产生歧义的内容, 数学中的结论意义分明, 不存在似是而非、模棱两可的断言。因此, 数学表达式的语义完全可以形式化地表现出来。并且, 对于任意两个数学表达式, 依据数学规则作形式上的充分化简和变换后, 若最终得到的结果表达式完全一致, 则这两个数学表达式必定等价, 否则不等价。MathML 为数学表达式提供了形式化表示方法, 可以在 MathML 的基础上实现两个数学表达式等价与否的判断。

2.1　树型模式匹配

模式匹配是计算机科学和人工智能等许多领域中最基本的运算之一。其内容不仅包含两个常量串之间的 "完全匹配", 如 D. Knuth 的字符串的快速匹配算法中所处理的在串 T 中找与串 P 完全相同的子串[2], 而且包含 "一致匹配", 如 R. Kowalski在对逻辑程序作过程解释时, 对一个子目标与一个规则首部进行匹配, 只要求能够通过 "合一替换"使它们相同, 即匹配成功[3]。

在许多复杂问题中, 匹配目标和模式自身的结构都不是串所能简单刻画的, 在这种情况下要借助于更复杂的结构——树。为了区分于 "一致匹配", 我们引入 "结构相容"的概念。我们称一棵目标子树 T_j 与一个模式(树) P_i "结构相容", 是指 T_j 具有与 P_i 中非叶子节点部分完全相同的结构, 且 P_i 中的叶子节点与 T_j 中相应成分(可能是叶子节点, 也可能是子树) 能够建立一一对应的映射关系(简称对应) 。例如, 对于图 2 所示的目标子树 $T_1: x_1 + x_2$, $T_2: x_3 + x_3$, $T_3: a + b \times c$ 和模式(树) $P_1: x + y$, $P_2: x + x$, 只有 T_2 和 P_2 结构相容, 而 T_1, T_2, T_3 都与 P_1 结构相容。

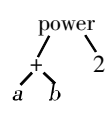

图 1　一棵中缀表达式树

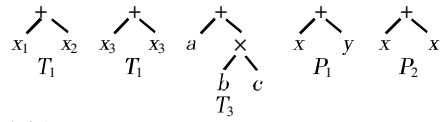

图 2　目标子树和模式(树)

在这个例子中, P_1 的 x 与 x_1, x_3, a 对应, y 与 x_2, x_3, $b \times c$ 对应, T_1, T_2 和 T_3 与 P_1 是类型相同的表达式; 但 P_2 中的 x 只能与 T_2 中的 x_3 对应, 在一个模式中, 相同的 x 只能与目标中相同的对象对应(从这个意义上讲 "对应"是 "一致"的扩展) 。如果能断定某目标(树) T_j 与模式(树) P_i 结构相容, 则称它们 "相容匹配"成功, 简称匹配成功。给定一个目标(树) T 和一个模式(树) P, 判断 T 中是否存在一个子树与 P 相容的过程, 称为树型模式匹配, 简称匹配。

2.2　数学表达式的变换

按数学语言的文法, 一个数学表达式 T 可以自然地被表示为一棵树, 而一个规则 P 则可以表示为两棵树所组成的对偶(P_l, P_r) 。其中 P_l 为规则左部对应的树, 它给出被变换的数学表达式的模式; P_r 为规则右部对应的树, 它给出变换后的结果模式。将规则 P 作用于数学表达式 T 时, 首先要将 T 与 P_l 进行树型模式匹配, 一旦匹配成功, 则存在 T 的一个子树 T_j 与 P_l 结构相容, 按匹配时所得的(相容) 信息, 将 P_r 进行实例化处理, 产生相容的目标子树, 记为 $P_{r'}$, 用 $P_{r'}$ 替换 T 中的 T_j, 得到 T′(T′≠ T) , 这就是数学表达式的一次变换。

一个源数学表达式 T 应用数学规则库作变换, 就是将 T 与规则库中的所有规则左部逐个作匹配, 若有子树匹配成功则变换。一直重复这个过程, 直至得到变换结果 T^*, 使 T^* 的所有子树与所有规则的左部都不存在匹配的情况, 此时的 T^* 就是 T 应用规则库作变换得到的最终结果。

2.3　规则库的描述

由于本模块主要是对数学表达式作化简和变换, 所以包含的规则不仅有数学公式和定律, 而且有一些化简规则。就本模块处理的初等数学而言, 规则的具体内容包括以下几个方面:

(1) 常数运算化简的规则, 即若判明运算符的参数都是常数而不是变量, 则将其结果计算出来, 以代替原来的表达式。如 3 + 4 化简为 12 等。

(2) 与特殊数字相关的化简规则。如 0 × x 化简为 0 等。

(3) 相同变量间的化简运算规则。如 x - x 可化简为 0, x / x 化简为 1 等。

(4) 部分初等数学公式。如 $a^2 - b^2 = (a + b)(a - b)$ 等。

(5) 对表达式结构产生影响的数学定律。如分配律: $a \times (b + c) = a \times b + a \times c$ 等。

为保证处理过程的合理性, 我们构造的规则集满足以下两个特性:

①不可逆性。若有规则 A 使得表达式由结构形态 α变换到 β, 则不存在规则 B 使得该表达式由结构形态 β变换到 α。例如, 若规则库中包含了分配律 $a \times (b + c) = a \times b + a \times c$, 则不能再包含结合律 $a \times b + a \times c = a \times (b + c)$。不可逆性保证在应用规则库时不会产生死循环。

②不传递性。若有规则 A 使得表达式由形态 α变换到 β, 且有规则 B 使得该表达式由形态 α变换到 γ, 则不存在规则 C 使得该表达式由形态 β变换到 γ。不传递性保证一个规则的左部不会与另一个规则的右部相匹配, 这样子树表达式本身匹配一条规则作变换后, 不会再有规则和子树的变换结果再匹配, 所以子树表达式本身最多依照规则库变换一次。

顺序遍历规则库时, 虽然规则库中的规则任意排列最终都能正确地完成变换, 但由于会有多条规则同时适用于一个表达式的情况, 所以规则的排列会影响到实现过程和效率。为了提高变换效率和简化变换结果, 我们根据上述五类规则的不同变换效果赋予其优先级, 优先级从高到低依次为: 与特殊数字相关的化简规则、相同变量间的运算化简规则、常数运算化简的规则、部分初等数学公式和对表达式结构产生影响的数学定律。在组织规则库时, 将优先级较高的规则组排在前面, 组内则按规则左部 P_l 树的高度排序, 高度较高的排序较前。由于规则库的不传递性, 一个表达式本身依照规则库至多变换一次, 因此将变换后结果比较简单的规则排在前列, 顺序地使用规则库时, 可以保证每次都会得到最简的变换结果。

2.4　子树匹配顺序

给定一个待变换的数学表达式树 T 和一组规则 P_i(i = 1, 2, …, n) , 必须按照一定的顺序对 T 中的子树与规则进行匹配, 使得每次仅得到一种变换结果。一般采用顺序处理的策略, 对 T 中的子树规定一个选取顺序, 可以先根次序遍历 T, 自顶向下地选取子树, 也可以后根次序遍历 T, 自底向上地选取子树。

2.4.1 自顶向下的遍历算法

这是最自然的遍历算法,即从源表达式树的根节点开始,按先根顺序取一个子树与一个规则的左部匹配,结果有两种可能:

(1) 若匹配失败,则取下一个规则与这个子树匹配;若所有规则左部均不能匹配成功,则按先根次序取下一个子树与各规则匹配。

(2) 若匹配成功,则用匹配的相容信息,根据相应规则右部 P_r 的模式生成实例化的结果子树,并用它替换源目标子树。由于规则库的不传递性,直接从变换结果中的相应位置开始继续按先根次序取下一个子树与规则库匹配。

由于子树变换后会影响到其祖先的树型结构,比如使得原来不相同的子树变得相同,所以对源表达式树的一次先根遍历并不能保证得到的是最终结果,必须以一次遍历下来没有发生变换为终止条件。

这种自顶向下的算法是一种逐步顺序变换的策略,思想朴素,操作自然,当表达式能够迅速简化时处理效果比较好。但有时要经过数次遍历才能达到最终结果,其中包含许多重复的工作,代价显然较高。

2.4.2 自底向上的遍历算法

这是一种企图一次遍历求得最终变换结果的策略。它从源表达式树最底层的非叶子节点开始,按后根顺序取子树与规则库作匹配变换,直至到达数学表达式树的根为止。由于子树的变换只对其祖先的树型结构产生影响,不会使其子孙的树型结构发生变化,所以一次遍历就可以得到最终的变换结果。

这种算法的特点是:在处理每个子树时,这个子树的所有真子树均已处理完毕,不会出现自顶向下时的重复遍历的情况,没有重复的工作。但由于是自底向上遍历,有时在处理可以一步化简的表达式时,可能会变简为繁,经过若干步变换才获得原来只需一步变换就可以得到的结果,降低了效率。

3 结束语

该模块处理时先读入规则文件将其在内存中解析成链表;然后读入学生答案和标准答案,并将其解析成两棵树 S 和 T,应用规则链表分别对 S 和 T 作变换得到 S^* 和 T^*,再由 S^* 和 T^* 是否一致判断 S 和 T 是否等价,最后输出结果。

就目前实验的初等数学规则库而言,本模块可以判断任意复杂度的两个初等数学表达式的等价与否,对规则库作适当扩充后,本模块可以对更广泛内容的数学表达式作等价性判断。

参考文献:

[1] athematical Markup Language (MathML), version 2.0 (Second Edition) [EB/OL]. http://www.w3.org/TR/2003/WD-MathML2-20030411, 2003.

[2] E Knuth, J Morris, V R Pratt. Fast Pattern Matching in Strings[J]. SIAM Journal on Computing, 1977, 6(1): 323-360.

[3] mden, M H van, Kowalski R A. The Semantics of Predicate Logic as a Programming Language[J]. Journal of the ACM, 1976, 23(4): 733-742.

作者简介:

刘峰(1976-),硕士研究生,主要研究方向为自然语言处理、信息抽取技术等;袁春风(1962-),教授,主要研究方向为多媒体信息处理、自然语言理解等。

(上接第 29 页)息,将查询分解至各有关信息源。

(5) 信息源接收上层传来的查询请求,利用自身的查询功能进行查询,返回查询结果。

(6) 将由各信息源返回的查询结果进行整合处理,以 XML 文档的形式返回给用户。

3 结论

在分析现有信息检索技术的基础上,根据分布式异构信息空间的特点和用户需求,提出了基于 LDAP 目录服务的异构信息检索模型。在该模型中,各资源中心的元信息及信息资源仓储实现了局部信息资源的管理,而 LDAP 目录服务则将所有资源中心有机地结合起来,实现了全局元信息的有效管理及信息检索服务。

与传统的信息检索相比,该模型具有如下的优点:①大数据量容纳性。采用了信息资源中心和 LDAP 目录两级元信息管理模式,可管理的数据量大大提高。②负载均衡。将目录信息分布存储在多个服务器上,与集中式目录服务相比,减轻了单个目录服务器的压力,具有负载均衡的特点。③访问速度快。一般情况下由本地服务器提供信息查询服务,在本地服务器不能满足要求时才向其他服务器转发请求,因此降低了查询响应时间,提高了信息检索的效率。④扩展性强。多个服务节点可容纳更大的数据量。⑤可靠性高。采用 LDAP 目录服务的复制功能,对整个目录进行多点复制,在一个服务器发生故障时仍可通过其他服务器访问到所需的信息。与集中式信息检索相比,系统可靠性大大提高。但是与高可用性相伴随的是系统实现和管理的复杂性,各服务器间复制信息的一致性、服务器间的请求转发、查询请求分解和查询结果整合等问题都是在实现过程中需要重点解决的问题,因此从模型设计到系统实现还有很多艰苦的工作要做。

参考文献:

[1] atnasamy S, Francis P, et al. A Scalable Content Addressable Network[C]. ACM SIGCOMM, 2001. 161-172.

[2] toica I, Morris R, et al. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications[C]. Proceedings of the '01 ACM SIGCOMM Conference, San Diego, California, USA, 2001. 149-160.

[3] Czajkowski, S Fitzgerald, et al. Grid Information Services for Distributed Resource Sharing[C]. Proceedings 10th IEEE International Symposium on High Performance Distributed Computing (HPDC-10 '01), IEEE Computer Society, 2001. 181-194.

[4] einz Johner, et al. Understanding LDAP[M]. IBM Corp., 1998.

[5] eong W, Howes T, Kille S. The Lightweight Directory Access Protocol [S]. RFC 1777, 1995.

[6] Fitzgerald, I Foster, et al. A Directory Service for Configuring High-Performance Distributed Computations[C]. Proc. 6th IEEE Symp. on High-Performance Distributed Computing, 1997. 365-375.

[7] Callan. Distributed Information Retrieval[M]. Kluwer Academic Publishers, Advances in Information Retrieval: Recent Research from the CIIR, Chapter 5, 2000. 127-150.

作者简介:

孙青(1970-),女,硕士研究生,主要研究方向为信息管理与智能决策技术;邓苏,教授,博士,主要研究方向为数据仓库技术、智能决策技术;黄宏斌,讲师,博士研究生,主要研究方向为信息管理与智能决策技术;黄金才,讲师,博士,主要研究方向为信息管理与智能决策技术。