KBASE－P：一个知识库程序设计语言*

施伯乐　朱扬勇　郭德培

（复旦大学计算机科学系　上海　200433）

摘要　KBASE－P是一个知识库程序设计语言. 它以KBASE作为查询语言，以FD－PROLOG为过程性的宿主语言. 二者具有相同的编程风范，都是Horn子句风范，并且系统对任何磁盘数据访问都是以“每次一个集合”的方式进行，因此在KBASE－P中，查询语言与宿主语言之间的“阻抗不匹配”问题尽可能地减小了. KBASE－P以RDBMS来管理事实（包括中间求值结果）和规则，因此能够有效地管理大容量的数据，并且数据是可共享的，适合于处理大规模的知识密集型应用. 本文介绍了KBASE－P语言及其特点，并与相关的系统进行了比较.

关键词　数据库，知识库，逻辑程序.

本文研究的知识库系统是以J. D. Ullman对知识库系统的定义[1]为指导的. 他认为：一个知识库系统是具有2个特征的逻辑程序设计系统. ① 有一个既作为查询语言又作为宿主语言的描述性语言. ② 支持数据库系统的全部功能，如：大批量数据的高效存取、数据共享、并发控制及错误恢复等.

该定义较好地体现了数据库与人工智能的结合，得到了承认，并据此展开了大量的理论研究与系统实验，提出了DATALOG(DATAbase in LOGic)语言.[1,2]在DATALOG中，谓词符号表示关系，有2种定义关系的方法：存储在数据库中的关系称为外延数据库(EDB)关系，对应的谓词称为EDB谓词；由逻辑规则定义的关系称为内涵数据库(IDB)关系，对应的谓词称为IDB谓词. DATALOG语言的提出为“每次1个集合”地处理递归奠定了基础(而PROLOG是以“每次1个元组”方式处理递归的).

KBASE是我们以前在863计划支持下开发的一个说明性的知识库查询语言. 它是DATALOG的扩充，支持模块分层的否定、函数等. KBASE查询语言的规则形为：

$$h:-p_1,p_2,\cdots,p_k.$$

其含义为：“如果$p_1,p_2,\cdots,p_k$都为真，则h为真”. 谓词的否定用$\sim p$表示；函数用$f_$〈函数名〉表示. KBASE系统以RDBMS管理大容量的事实数据，以Semi－Naive算法进行递归查询求值，实现了魔集(Magic－Set)重写、计数(Counting)方法及左右线性变换等优化算

---

* 本文研究得到国家863高科技项目基金资助. 作者**施伯乐**，1935年生，教授，博士导师，主要研究领域为数据库，知识库和面向对象技术. **朱扬勇**，1963年生，讲师，主要研究领域为数据库，知识库和多媒体. **郭德培**，1969年生，博士生，主要研究领域为数据库，逻辑程序和多媒体.

本文通讯联系人：施伯乐，上海 200433，复旦大学计算机科学系

本文收到1995－08－30收到修改稿

法.系统实现了一个良好的优化策略自动选择算法;提出了查询模式的概念,使得在一定条件下,规则的重写优化可以在给出查询前进行(详细内容见文献[3,4]).

80年代的研究表明:扩充DATALOG使其成为通用的程序设计语言是不成功的.当前的作法是将描述性语言(用于查询、实现较多的查询优化技术)与过程语言(用于控制、I/O和DB操作)的有机结合.如:CORAL[5], Glue－Nail[6]等.KBASE－P是一个知识库程序设计语言,它以KBASE作为查询语言,以FD－PROLOG(我们开发的一个PROLOG)为过程性的宿主语言.KBASE－P系统的设计以实用为目标,尽量减小"阻抗不匹配"[7]和提高系统处理问题的能力.在KBASE－P中,查询语言与宿主语言具有相同的编程风范,都是Horn子句风范,并且系统对任何磁盘数据访问都是以"每次1个集合"的方式进行,因此KBASE－P的"阻抗不匹配"问题尽可能地减小了.KBASE－P语言与PROLOG是兼容的,通常的PROLOG程序可以不作修改地在系统中运行,并且PROLOG形式的事实和数据库关系事实可以相互转换.这有助于将许多基于PROLOG的应用系统移植到KBASE－P系统上,使原有的系统在规模和实用水平上有所提高.

本文介绍了KBASE－P语言及其特点.目前,单用户的KBASE－P版本已经实现,Client/Server结构的KBASE－P版本正在设计中.

1　设计KBASE－P语言的几点考虑

1.1　关于知识库查询语言的扩充

作为一个知识库程序设计语言,I/O操作和DB更新是必需的.一些知识库系统在查询语言中加入一些过程性机制,试图将查询语言扩充为通用的知识库程序设计语言.如LDL[8],RDL1[9]等.但这样做所提供的控制机制很少,作为通用的语言来说I/O操作、用户控制机制还不够,即用户在限制严格而又及为有限的控制机制下难以对应用编程.因此,将DATALOGe(DATALOG的扩充)嵌入过程语言就成为必然.如:Glue－Nail将NAIL! 嵌入过程语言Glue;CORAL将DATALOGe嵌入C＋＋和一个命令子语言.

因此,我们不打算将I/O操作和DB更新能力加入到KBASE语言中,更不打算破坏KBASE的纯描述性这一优点,而采用将KBASE与一过程语言相结合的方法来实现一个通用的知识库语言.

1.2　过程语言的选取

我们选择了PROLOG(主要是考虑到它的控制策略)作为KBASE－P的过程性部分.PROLOG作为一种良好的人工智能语言已经得到广泛承认和使用.KBASE和PROLOG都是基于Horn子句的语言,具有相同的编程风范.事实上,DATALOG就是用于数据库查询的1个PROLOG版本.因此,选择PROLOG作过程语言,至少可以减小查询语言和宿主语言编程风范的不匹配.其次,对于一个通用的程序设计语言来说,每次1个元组的计算是必要的.而PROLOG正是以"每次1个元组"方式求值的.现有的人工智能应用系统大都是用PROLOG实现的,因此选择PROLOG就更具有实用价值.

1.3　减小阻抗不匹配问题

数据库操纵语言与宿主语言的"阻抗不匹配"问题是指二者支持不同的编程风范和数据

类型.[7]解决数据库操纵语言与宿主语言之间的阻抗不匹配,是知识库系统研究的初始原因之一.为了减小"阻抗不匹配问题",KBASE－P 语言采用 KBASE 与 PROLOG 集成的形式,并采用了"每次 1 个集合"的访盘方式.由于二者都是基于 Horn 子句的语言,因此没有查询语言和宿主语言编程风范的不匹配.其次,对用户来说,二者的数据类型是一致的.在系统实现上,设计了内存关系管理器,使得所有对外存数据的访问都是以"每次 1 个集合"的方式进行.从而减小了由于数据类型的不匹配而引起的"每次 1 个集合"的求值方式与"每次 1 个元组"的求值方式的不匹配.

2 KBASE－P 语言

2.1 谓 词

从用户角度来看,有 4 类谓词.**EDB 谓词**:是指其事实是存放在二级存储器中的谓词.即以关系形式存于关系数据库中,由 RDBMS 管理.EDB 谓词不出现在规则头中.**IDB 谓词**:是指其事实是由规则定义的谓词,它的事实为一个关系.每个 IDB 谓词至少出现在 1 条规则的头中.**内部谓词**:是指比较谓词、I/O 谓词以及其它一些系统提供的谓词.**P 谓词**:至少出现在一条规则的头中,没有变元,只取"true"或"false"值的谓词.这类谓词不表示数据库关系,这是与 IDB 谓词的不同点.

从系统求值角度来看,有 3 类谓词.**KBASE 谓词**:KBASE 谓词能够被 KBASE 系统处理.EDB 谓词、算术比较谓词是 KBASE 谓词;如果一个头谓词的所有规则的子目标谓词都是 KBASE 谓词,则该头谓词是 KBASE 谓词;一个 KBASE 谓词的否定也是 KBASE 谓词.**RGT 谓词**:由":＝语句"定义的谓词.它们用 RGT 算法(规则目标树扩展)求值.**PROLOG 谓词**:除了 KBASE 谓词和 RGT 谓词以外的谓词.它们用 PROLOG 解释器求值.

2.2 事 实

一般来讲,KBASE－P 语言程序的事实是以元组关系的形式存于外存数据库中的,因而是持久数据,并且是可共享的.但系统也允许事实以 PROLOG 项的形式出现在程序中,并且 2 种形式的事实可以方便地相互转换.即 EDB 关系形式的事实和 PROLOG 项形式的事实可以同时出现在一个程序中.

2.3 语 句(规 则)

KBASE－P 语言有 2 种形式的规则:

$$h:\!-p_1,p_2,\cdots,p_k.$$

$$h:=p_1,p_2,\cdots,p_k.$$

其中 h 称为规则头,$p_1,p_2,\cdots,p_k$ 为规则体,并且体中的谓词称为子目标.语句的含义为:"如果 $p_1,p_2,\cdots,p_k$ 都为真,则 h 为真".

:－语句:这是通常的 PROLOG 语句,可以用 PROLOG 解释器以"每次 1 个元组"的方式求值.但是,如果头谓词是 KBASE 谓词,则系统将用 KBASE 以"每次 1 个集合"的方式求值.

:＝语句:这类语句将用 RGT 算法求值.这种语句是为高级用户(他们熟悉各种求值方式的优缺点)设置的,目的是:通过合理使用这种语句,尽可能地提高应用程序的执行效率.

对于具体的程序来说,这种语句不是必须的.该语句的要求是体中只允许有EDB谓词、IDB谓词和某些内部谓词.

例1:我们来讨论下面的KBASE－P语言程序:

r_1　answer1(X):=read(Y),ancestor(X,Y).

r_2　answer2(X):－ancestor('*john*',X),print('*john*',X),
　　nl,print('－－－'),fail.

r_3　ancestor(X,Y):－parent(X,Y).

r_4　ancestor(X,Y):－ancestor(X,Z),parent(Z,Y).

其中parent(X,Y)是EDB谓词.

规则r_3和r_4是DATALOG规则,用KBASE系统求值.answer1(X)是*RGT*谓词,而answer$_2$(X)是PROLOG谓词,不能用KBASE处理.r_1可以"每次1个集合"地进行处理,而r_2则不行.如果我们用下面的r_{11}代替r_1

r_{11}　answer1(X):－read(Y),ancestor(X,Y),fail.

则结果完全一样.但是,因为r_1是"每次1个集合"地处理,而r_{11}是"每次1个元组"地处理,所以r_{11}的执行时间要比r_1多得多.□

2.4　程序和模块

程序　KBASE－P语言程序是KBASE－P语句的集合,语句从上到下有序,子目标从左到右有序.这就是说KBASE－P的控制流与PROLOG是相同的.虽然实际上所有的KBASE语句是纯描述性的(即顺序无关),但是KBASE谓词和PROLOG谓词对用户是透明的.

模块　为了便于编制大规模的应用程序,KBASE－P语言提供了模块构造机制.一个模块是关于一个称为"顶谓词"的谓词的完整的定义.当提问顶谓词目标时,模块被执行并返回结果.模块以下面形式说明.

module(〈顶谓词〉|〈谓词参量表〉).
　〈定义顶谓词的规则集〉
end(〈顶谓词〉).

有一个限制:当模块被调用时,其〈谓词参量〉必须是实例化的.

例2:下面规则给出了一个模块的定义及其被调用的形式.

r_0　answer(X):=read(Y),module(tc(X,Y)|parent(X,Y)).
　module(tc(X,Y)|e(X,Y)).
　　tc(X,Y):－e(X,Y).
　　tc(X,Y):－tc(X,Z),e(Z,Y).
　end(tc).

其中模块tc(X,Y)是传递闭包规则.规则r_0读入一个Y值,然后调用该模块,谓词参量实例化为parent(X,Y).规则r_0的结果和例1中的规则r_1一样.

2.5　KBASE－P中的I/O与DB更新操作

KBASE－P用内部谓词来实现I/O与DB更新操作.除了PROLOG语言通常拥有的I/O谓词以外,考虑到现代应用对表格、图形的需要,我们设计了大约有20个关于图形的内

部谓词. KBASE－P 语言提供了 2 种形式的 DB 更新操作内部谓词. ①是内部谓词 sql ('SQL 命令'),该谓词直接将 SQL 命令传给 DBMS 执行返回结果;②是用 insert,delete,update 等内部谓词实现 DB 更新操作. DB 更新涉及到事务. 我们采用了 LDL 的作法:每个 DB 更新谓词作为一个事务.

3 实现简介

3.1 编 译

KBASE－P 语言编译包括词法、语法分析;知识分类(分离出 KBASE 谓词、RGT 谓词和 PROLOG 谓词);最后分别将 KBASE 规则和 PROLOG 规则编译到 ERA 树和 HASH 树.

除了 PROLOG 所采用的技术外,KBASE－P 编译器还在以下 2 个方面进行了优化:如果某个 PROLOG 规则体中,有相邻的 KBASE 谓词,则用一个辅助谓词代替它们,辅助谓词的参数取所有这些相邻 KBASE 谓词的参数;对于 KBASE 谓词,KBASE－P 编译器将它的规则集进行 Magic－Set 变换.

3.2 KBASE－P 解释器

KBASE－P 解释器是 1 个改进的 PROLOG 解释器,能够调用 KBASE 的不动点求值器和 RGT 求值器(深度优先每次 1 个集合). KBASE－P 解释器(根据目标和谓词依赖图)自顶而下解释执行程序;如果当前目标是 KBASE 谓词,则调用 KBASE 系统求出匹配目标的关系,并交给"内存关系管理器"管理,"内存关系管理器"将该关系的一个子集(最多 M 个元组)装入内存;如果当前谓词为":－"规则的头谓词,则调用 RGT 算法,计算出匹配目标的关系,也交给"内存关系管理器"管理,做同样的工作. KBASE－P 解释器在内存事实用完后调用"内存关系管理器"将该关系的下一个子集装入内存,然后继续工作下去.

我们采用先 KBASE 谓词、再 RGT 谓词、最后才是 PROLOG 谓词的优先次序,其好处是尽可能地使用了"每次 1 个集合"的求值方式,从而提高了系统的执行效率.

3.3 内存关系管理器

一般地,每个 EDB/IDB 谓词的关系都存放于外存. 这样使系统能够处理大规模的应用问题. 为了提高系统的执行效率,每个 EDB/IDB 谓词都有一个不多于 M 个(可设置)元组且匹配相应谓词目标的内存关系. 内存关系只是一个副本,用于求值. 当没有可用元组时,再从外存装入 M 个元组将以前的元组(用完了的)覆盖.

内存关系管理器的使用保证了所有的外存访问都是以集合方式进行的,从而提高了 PROLOG 解释器的执行效率.

4 与相关系统的比较

几个类似的语言(DATALOGe 嵌入过程语言形式的语言)都是在 1991～1994 年期间研制的,并且仍在研制当中. 如:LDL,Glue－Nail,CORAL,XSB[10],KBASE－P 等. 表 1 展示了这些系统和 KBASE－P 的对照.

表 1　各种知识库系统的对照

系统名称	优化技术	语　言	否　定	数据管理	用户数据空间
LDL	Magic Sets	LDL	Modular Stratified	Data Storage Manager	Main Memory
Glue－Nail	Magic Sets	Nail+Glue	Well－founded	Data Storage Manager	Main Memory
CORAL	Magic Templates	CORAL+C++	Modular Stratified	Data Storage Manager	Main Memory
XSB	SLG	HiLog	Well－founded	Data Storage Manager	Main Memory
KBASE－P	Magic Sets	KBASE+ FD－PROLOG	Modular Stratified	RDBMS	Main Memory and Disk

从表 1 我们看到,这些系统在支持否定和查询优化方面是类似的.在语言方面,LDL 试图扩充查询语言为通用程序设计语言,这一工作至今仍没有完成,并且现行的 LDL 语言被认为是难以使用的.其它系统都采用了"查询语言＋通用过程语言"的形式.相比之下,KBASE－P 中二者的结合是紧集成,对用户来说可以认为是一种语言,没有编程风范的不匹配.在数据管理方面,KBASE－P 以 RDBMS 来管理事实和规则,因此能够有效地管理大容量的数据,并且数据是可共享的.同时,这样做符合了 J. D. Ullman 关于"知识库系统是具有数据库系统能力的逻辑程序设计系统"的主张.[1]而其它系统只有简单的存储管理器,不具有数据库能力.正因为如此,其它系统都是主存系统,即用户程序和数据以及求值都在主存中,不能处理大规模应用.而 KBASE－P 的数据(EDB 关系和 IDB 关系)都是放在外存的,它可以处理大规模应用.并且,当内存关系管理器的内存元组数 M 设置的足够大时,KBASE－P 也可以变成一个主存系统.这说明 KBASE－P 更具有灵活性.

5　结　论

基于 DATALOG 查询优化的研究已达到相当程度,很难再有实质性进展.所以,当前的支持否定、函数、集合的扩充的 DATALOG 语言将逐步成为知识库查询语言的标准.事实上,一个通用的程序设计语言必须要有用户控制.因此,知识库程序设计语言应该是过程语言(用于控制)与描述性语言(用于查询、实现较多的查询优化技术)的有机结合.已经有一些系统是这样做的.如:Glue－Nail, CORAL 等.KBASE－P 系统也是这样做的,并且更适合于处理大规模知识密集的应用(这是知识库应用系统的基本特征).我们认为:采用"DATALOGe 嵌入过程语言"的形式开发知识库程序设计系统是正确的也是有效的.

参考文献

1　Ullman J D. Principles of database and knowledge－base systems. Computer Science Press, 1989, I, II.

2　Ullman J D. Implementation of logic query language for databases. ACM Transactions on Database Systems, 1985, **10(3)**: 289～321.

3　Shi Baile, Ma Xueqiang. Logic query language KBASE for knowledge－base systems. Database Technology, 1992, **4(3)**: 163～167.

4　马学强,施伯乐.知识库系统 KBASE 中的规范化理论.软件学报,1990, **1(4)**: 40～47.

5　Ramakrishnan R *et al.* CORAL－control, relation and logic. Proc. of the 18th VLDB Conf., Canada, 1992.

6　Phipps G *et al.* Glue－nail: a deductive database system. Proc. of the ACM SIGMOD Conf. on Management of Data, 1991.

7 Copeland G, Maier D. Making smalltalk a database system. Proc. of the SIGMOD Conf. on Management of Data.
8 Naqvi S, Tsur S. A logic language for data and knowledge base. Computer Science Press, 1989.
9 Kiernan G *et al*. Making deductive database a practical technology: a step forward. Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data, 1990. 237～246.
10 Sagonas K, Swift T, Warren D S. XSB as an efficient deductive database engine. Proc. of the ACM SIGMOD Conf. on Management of Data, 1994.

KBASE—P: A KNOWLEDGE—BASE PROGRAMMING LANGUAGE

Shi Baile Zhu Yangyong Guo Depei

(*Department of Computer Science Fudan University Shanghai* 200433)

Abstract KBASE—P is a knowledge—base programming language. It uses KBASE as query language and FD—PROLOG as its procedural host language. The two languages have the same programming paradigm — Horn clause paradigm, and the KBASE—P system accesses disk—resident data in set—at—a—time mode, thus the impedance mismatch problem has been minimized in KBASE—P. KBASE—P manages facts and rules including intermediate results with the help of RDBMS, hence it can manage large amount of data which can be shared by users, and is suitable for large—scale applications. KBASE—P language and its features are described in the paper, and some comparison with related systems are given also.

Key words Database, knowledge—base, logic programming.