国防科技大学学报
JOURNAL OF NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY
第13卷第4期 1991年12月 Vol. 13 No. 4

逻辑程序设计与关系数据库集成的一种基于预编译的解释方法*

邓铁清 王志英 吴泉源

（电子计算机系）

摘 要 本文在研究逻辑程序设计与关系数据库的两种集成方法（解释和编译）的基础上，提出了一种新的基于预编译的解释方法，该方法保持了前两种方法的优点，克服了它们的不足。基于此方法，文中介绍了一个相应的集成系统的原型及主要实现技术，其中包括物理级的耦合连接，逻辑级的语言合成，源程序级的部分计算和静态优化，以及动态执行过程中的事实调度和版本管理等。目前，该原型作为知识库管理系统GKBMS的内核，已投入实际应用。

关键词 解释，编译，基于预编译的解释，逻辑程序设计，关系数据库

分类号 TP314

在大中型知识库管理系统的构造中，逻辑程序设计与关系数据库的集成是一个较活跃的研究课题[1,2,3,6]，其主要目标是：（1）高效，即系统间的通讯次数越少越好；系统间交换数据的数量愈少愈好；（2）对用户友好，即系统间的通讯完全地对用户透明；（3）广义性，即系统应提供一个以一阶谓词为基础的通用程序设计语言。

目前，逻辑程序设计与关系数据库的集成方法可分为两种，即解释方法和编译方法。解释方法采用了标准的Prolog归结策略。在Prolog程序的执行期间，如果当前文字为关系数据库谓词，则Prolog暂停执行，将该文字翻译为关系查询去访问数据库，并将提取的事实加载到Prolog工作空间中，再启动Prolog继续执行。编译方法针对给定的用户目标，将Prolog程序作为一个整体处理，分离其中的推理部分和数据库部分。编译之后，将关系查询进行分组、优化，并翻译成DML语句去访问数据库。一旦事实都被全部加载到主存空间，便开始最终的求值。

理论分析和实验结果[4][9]表明：解释方法虽然没改变Prolog的标准归结策略，为用户提供了一个完整的Prolog语言，但每一关系都得访问一次数据库，显然过于频繁，关系数据库实际上退化为文件系统；而编译方法虽然把Prolog与DBMS间的通讯降低到一次，效

* 国家高技术发展计划资助课题
** 1990年12月15日收稿

率高，但由于数据库谓词的收集引起了 Prolog 标准归结过程的改变，其实用性受到限制。

本文提出了一种新的基于预编译的解释方法，它是上述两种方法的有效结合。基于这种混合策略，我们建造了一个逻辑程序设计与关系数据库集成的系统原型。该原型作为知识库管理系统 GKBMS[9]的内核，已在 Micro VAX Ⅱ 上具体实现，并投入实际应用。本文主要介绍该原型系统的结构及其关键实现技术。

1 基于预编译的解释方法及其系统建模

基于预编译的解释方法是我们提出的一种新的演绎问题的求解方法，其基本思想早在文[11]中就有描述。给定一个用户程序，基于预编译的解释方法的主要求解步骤是：

(1) 部分计算：模拟 Prolog 的标准归结策略，并冻结或推迟目前不可求值的文字（或子目标），达到汇集数据库问题的目的。

(2) 静态优化：把部分计算后的程序进行相关性分析，识别与分离子句中的纯数据库成分，尤其是独立的联结－选择－投影运算，并进行可能的静态翻译，组织成 DBMS 的有待进一步确定化的提问。

(3) 由用户提问启动 Prolog 进程。

(4) 以标准的 Prolog 策略求解目标表中的子目标，直到当前子目标为(2)中的 DBMS 提问为止。

(5)将该确定化的 DBMS 提问动态翻译成 Oracle 的查询语句，即 SQL * PLUS 命令。

(6) 启动 Oracle 执行该 SQL * PLUS 命令。

(7) 翻译其结果为 Prolog 事实，并装入 Prolog 工作空间。

(8) 激活 Prolog 进程继续执行。

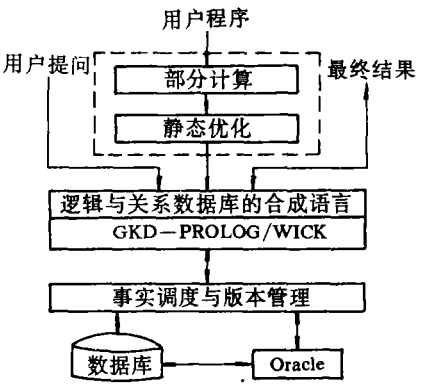

图 1 基于预编译的解释方法的系统原型

图 1 是基于该方法的一个逻辑程序设计 GKD-Prolog/WICK[10]与关系数据库 Oracle 的集成系统原型。

2 物理级：Prolog 与一个关系型 DBMS 的集成

在我们的原型系统中，GKD-Prolog/WICK 与 Oracle 系统的物理级集成，采用了松与紧两种计算策略。其中，前者利用 Oracle 系统的集合检索，回答某些询问时具有很高的执行速度。但这是以消耗内存空间为代价的。后者模拟了 Prolog 系统的一次一个元组的求解策略，一次仅取出一个元组。所以紧耦合能支持大中型知识管理系统的开发。基本接口谓词主要是下列两个：

sql (SQLPlus)

retrieve (Mode, FactName, SELECT)

其中，前者的作用是进入 Oracle 系统执行由参数 SQLPlus 特例化的 SQL * Plus 命令，该命令执行完毕，返回原处继续执行。后者由于参数 SELECT 特例化的结果为一个 SQL * Plus

检索语句，在 Oracle 系统执行后，检索结果或按松耦合方式（Mode＝loose）转化为 Prolog 的语法形式，作为原子 FactName 的事实装入 Prolog 内部数据库；或按紧耦合方式（Mode＝tight）存入中间关系 FactName 中。

同 Educe 系统一样，我们的原型系统运用程序中问题的递归性、关系中数据的存储量、以及内存中空间的剩余数自动控制和调度松耦合与紧耦合的求解方式，这样，Prolog 与关系数据库之间是深入物理级集成的。

3 逻辑级：逻辑语言与关系数据库语言的合成

在基本接口谓词的基础上，我们进一步在 GKD-Prolog/WICK 中以谓词形式合成了关系型的数据定义语言、域演算语言和元组演算语言。我们设计合成语言的目标是：

（1）从语言一级体现谓词演算与关系演算的紧密融合。即在 Prolog 语言中嵌入关系数据库语言以反映关系演算的特征，并方便两种演算信息的传递。

（2）合成语言应当考虑到集成系统的设计，利于集成系统的高效执行。

（3）合成语言应当尽可能体现各种关系演算的风格，满足不同用户的要求。

3.1 数据定义语言

数据定义语言由一组用于创建、删除、修改关系和索引表，以及用于提取公共数据字典信息的谓词组成。

例如：

```
? —create(proj, [projno＝number, pname＝char(10),
                 budget＝number(8,2)]).
```

系统首先根据谓词 create 中参数的特例化情况，将它翻译成下面等价的 SQL * PLUS 命令：

```
CREATE TABLE proj
        (projno number,
         pname char (10),
         budget number (8, 2));
```

然后调用接口谓词 sql/1，利用系统完成创建关系 proj 的功能，并成功返回。

3.2 域演算语言

域演算语言由一组基于域关系演算的数据处置谓词组成。这类谓词的特征是以域演算公式作为谓词的分量。

定义 3.1 （域演算公式）

（1）域演算原子是域演算公式。域演算原子呈下列两种形式：

$r(X_1,\cdots X_k)$　　及　　$X\theta Y$

其中 r 是一个 k 目关系，$X,Y,X_i(1\leqslant i\leqslant k)$或是常数，或是域变量，$\theta$ 是 Prolog 的算术比较运算符。

（2）用逗号“,”（表示“与”关系）连接起来的多个域演算公式构成的表达式是域演算公式。

（3）仅由(1)和(2)构成的表达式是域演算公式。

域演算公式实际上是一个联结－选择－投影运算，并且，一般来说，若演算中有多个原子均为关系，则它们之间构成联结运算；若关系中含有常量，或演算中含有比较类型的原子，或某个域变量在一个或多个关系中重复出现，则它们均构成选择条件；若演算公式中存在域变量，则它们将构成投影操作。这种对应关系本身即是域演算映射到接口谓词 sql/1 或 retrieve/3 的基础，当然，诸如数据表示的变换以及变量的特例化亦是不可忽视的问题。

3.3 元组演算语言

元组演算语言由一组基于元组关系演算的数据处置谓词组成。这类谓词的特征是以元组演算公式作为谓词的一个分量。

定义 3.2 （元组演算公式）

(1) 元组演算原子是元组演算公式。元组演算原子呈下列形式：

T1 θ T2

其中，θ是 Prolog 的算术比较或项比较运算符，T1 和 T2 为常数或元组变量。元组变量的一般形式是：

RelationName::AttributeName

(2) 用逗号“,”（表示“与”关系）连接起来的多个元组演算公式构成的表达式是元组演算公式。

(3) 仅由(1)、(2)构成的表达式是元组演算公式。

域演算与元组演算可以互相转换，因此，合成语言中的元组演算类谓词实际上均化为域演算类谓词实现。

4 预编译

4.1 部分计算

部分计算是一种重要的程序变换和编译优化技术，其一般定义是：设有函数 $f(x_1,\cdots,x_n)$，若当 $x_1=a_1,\cdots,x_k=a_k(1\leqslant k\leqslant n)$时，$f(x_1,\cdots,x_n)=f'(x_{k+1},\cdots,x_n)$，则称 f' 为 f 在 $x_i=a_i(1\leqslant i\leqslant k)$下的部分计算。

我们运用 Prolog 元、目标级混合程序设计技术，成功地开发了一个面向 Prolog 全集及嵌入的数据库语言的部分计算器[8]。该部分计算器不仅具有可靠的循环检测，而且能够正确地处理内部谓词。其主要实现技术参见文[7][8]。

部分计算在预编译中的作用概括起来是：

(1) 汇集数据库谓词。

(2) 尽可能特例化数据库问题中的变量，减少 Prolog 与 DBMS 间的信息交换量。

(3) 优化源程序。

4.2 静态优化

静态优化的功能主要在于识别、分离子句体中关于数据库的联结－选择－投影运算，构造极大关系演算表达式。

定义 4.1 rdb($(B'_j,\cdots,B'_k)$)是一个极大关系演算表达式，当且仅当 $B_j,\cdots,B_k$ 是一个极大关系演算序列。其中 $B_{j+p}(0\leqslant p\leqslant k-j)$隐去符号 rdb 后是 B'_{j+p}.

定义 4.2　$B_j,\cdots,B_k$ 是一个极大关系演算序列，当且仅当 B_{j-1}，B_j，…，B_k 和 B_j，…，B_k，B_{k+1}都不是关系演算序列。

定义 4.3　(1) B_j 是一个关系演算序列，如果 B_j 是一个关系型演算谓词；(2) $B_j,\cdots,B_k$ 是一个关系演算序列，如果每个 $B_{j+p}(0\leqslant p\leqslant k-j)$都是关系演算谓词，且每个 $B_{j+i}(0\leqslant i\leqslant k-j)$至少与一个 $B_{j+p}(p\neq i)$有一个共享或公用变量。

考虑到极大关系演算表达式的精确识别算法非常费时，从实用角度出发，我们采用一种弱化的处理策略——毗邻相关判别法。

算法 4.1　(关系演算表达式的识别)

设有子句 H：$-B_1,\cdots,B_n$. B_j 是 $B_1,\cdots,B_n$. 中的第一个关系型演算谓词，由 B_j 引出的演算表达式为 RCE.

(1) 令当前文字 L 为 B_j，关系演算表达式 RCEO 为 rdb((B'_j)). (B'_j 由 B_j 隐去符号 rdb 得到)。

(2) 子句体中 L 的下一文字送 L.

(3) 若 L 是关系型演算谓词，且与 RCEO 至少共享一个变量，则令 RCEO＝rdb((RCE'O,L'))，返回(2).

(4) 若 L 是 Prolog 的比较型谓词，且其中变量均属于 RCEO 的变量集，则令 RCEO＝rdb((RCE'O,L))，返回(2).

(5) 令 $L1$ 为 L，子句体中 L 的下一文字送 L，若 $L1$ 是关系型演算谓词，L 是 Prolog 的比较型谓词，且 L 中变量均属于 RCEO 与 $L1$ 的并集，则令 RCEO＝rdb((RCE'O,L1'，L))，返回(2).

(6) 否则置 RCE 为 RCEO.

5　事实调度与版本管理

由于数据库提问之间可能存在依赖关系，所以，无论在哪一种接口方式下，都会出现事实冗余的问题。为此，我们引入了版本的概念，并实现了版本管理的机制。

定义 5.1　一个版本定义为某个极大通项域演算表达式的一个实例。

定义 5.2　rdb(($B'_j,\cdots.B'_k$))被称为是与极大域演算表达式 rdb(($B_j,\cdots,B_k$))相联系的极大通项域演算表达式，如果每个域演算类型的原子 $B_{j+p}(0\leqslant p\leqslant k-j)$，当其中的各常量都被变量代换后为 B'_{j+p}.

对于与同一极大通项域演算表达式相联系的任何极大域演算表达式，在事实调入时采用同一谓词名，或在数据生成时采用相同文件（或关系）名，并且谓词参量或关系属性的个数与该表达式中出现的域变量的个数完全相同。这种表示在实现中显然更有利于判断各极大域演算表达式（即版本）之间的依赖关系。

版本管理实际上是一个历史记载和更新的过程，具体算法如下：

算法 5.1　(版本管理)

(1) 若历史版本序列 $V_i(1\leqslant i\leqslant n)$中没有可与当前版本 V_c 一致化的版本，则将 $|V_c|$ 中的事实全部装入 Prolog 数据库，或将 $|V_c|$ 中的数据全部存入生成的中间文件中，并将 V_c 记入历史版本序列。

(2) 若历史版本序列 $V_i(1\leqslant i\leqslant n)$ 中存在某个版本 V_i，使 $|V_c|\subseteq|V_i|$，则 $|V_c|$ 中的事实或数据成为冗余，不必加载。

(3) 否则，若 V_c 与 $V_i(1\leqslant i\leqslant n)$ 的最一般一致化取代为 θ_i，则调入的事实或生成的数据为集合 $V_c-(|V_1\theta_1|\cup\cdots\cup|V_n\theta_n|)$，并且，消除历史版本序列 $V_i(1\leqslant i\leqslant n)$ 中所有被 V_c 覆盖的旧版本，将 V_c 记入新的历史版本序列中。

6 结 论

本文提出了一种基于预编译的解释方法。该方法继承了编译和解释的优点，克服了两者的不足，达到了逻辑程序设计与关系数据库集成的最终目标。

基于预编译的解释方法，我们实现了一个逻辑程序设计 GKD-PROLOG/WICK 和关系数据库系统 Oracle 的集成原型。该原型具有下列特点：

(1) 采用松耦合与紧耦合两种方式集成了逻辑程序设计与关系数据库，其中松耦合策略充分利用现存关系数据库系统的集合检索，紧耦合策略模拟了逻辑系统一次一个元组的问题求解技术。两种策略的并存和自动选择使得集成原型能够支持大中型知识处理。

(2) 提出并系统地在逻辑语言中合成了关系型的数据定义语言、域演算语言和元组演算语言，既从语言级体现了谓词演算与关系演算的紧密融合，又从物理级支持了集成系统的高效执行。

(3) 将先进的部分计算用于 Prolog-DBMS 耦合程序的源级优化，以及基于静态优化的有效的预编译技术，大大改善了程序的执行速度。

(4) 采用基于版本管理的动态优化技术避免推理中冗余事实的载入，既节省了存储空间，又减少了系统间的通讯次数和交换数据的数量。

(5) 文中的各种方法和技术具有通用性，许多模块和工具甚至可以直接用于其它类型的集成系统。

参 考 文 献

[1] Kowloumdjan J. Shell and Interfaces with DBMS. Information Processing 89, North-Holland, 1989

[2] Floyd S. W. Logic as a Foundation for Deductive Database Systems. Information Processing 89, North-Holland, 1989

[3] Warren D. H. D. Logic Programming and Knowledge Bases, Proc. of the Isalmorada Workshop on Large Scale Knowledge Base and Reasoning Systems, Islamorada, Florida, USA, February, 1985

[4] Li Lei (李磊), Moll G. H. and Kowloumdjan J. Prolog-DBMS Coupling: a Hybrid Approach, Half Interpreted, Half Compiled. Proc. of 3rd Int'l Conf. on Data and Knowledge Bases, Jerusalem, Israel, 1988

[5] Bocca J. Educe, a Marriage of Convenience: PROLOG and a Relational DBMS. Third Symposium on Logic Programming, Salt Lake City, USA, September, 1986

[6] Deng Tieqing (邓铁清), Su Jinshu (苏金树), Wu Quanyuan (吴泉源) and Wang Zhiying (王志英). Researches on a Prolog-Based Knowledge Programming Environment, 1990 Int' l Conf. on Computer Processing of Chinese and Oriental Languages, CHang sha, P. R. China, April, 1990

[7] Deng Tieqing (邓铁清), Jin Zhi (金芝) and Wu Quanyuan (吴泉源). Further Researches on the Partial Evaluation of Prolog Programs. The Third Int' l Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems, Paris, France, July, 1990

[8] Deng Tieqing (邓铁清), Jin Zhi (金芝) and Hu Yunfa (胡运发). Investigation and Implementation of a Partial

Evaluation for Prolog Programs. The Fourth Japanese-Sino Sapporo Int'l Conf. on Computer Applications, Sapporo, Japan, October, 1990

[9] 邓铁清. 知识库管理系统 GKBMS 技术报告. 国防科技大学计算机系鉴定材料, 1990

[10] 胡运发, 高洪奎, 胡子昂, 卢肇川, 邓铁清. 逻辑程序设计的集成化环境系统 GKD-PROLOG/WICK 技术报告及用户手册. 国防科技大学计算机系鉴定材料, 1989

[11] 邓铁清. 演绎数据库系统 G-DDBS/VMS 的设计与实现。国防科技大学硕士学位论文, 1987

A Precompilation—Based Interpretive Approach for the Integration of Logic Programming and Relational Database

Deng Tieqing Wang Zhiying Wu Quanyuan

(Department of Computer Science)

Abstract

Based on two integrated methods (interpretation and compiling)of logic programming and relational database, this paper presents a new precompilation-based interpretive approach. This approach retains the advantages of the previous two methods and overcomes their drawbacks. A prototype of a corresponding integrated system based on this approach is introduced in the paper, and its chief impremention techniques are also discussed which include the coupling at the physical level, the compound of languages at the logical level, the source-level partial evaluation and static optimization , and the fact scheduling and version management during the dynamic execution. At present, this prototype has been put into practical applications as the kernel of the Knowledge Base Management System GKBMS.

Key words compiling, interpretation, precompilation-based interpretation, logic programming, relational database