Prolog

逻辑程序设计及应用

周立柱

清华大学出版社

Prolog
逻辑程序设计及应用

周 立 柱

清 华 大 学 出 版 社

内 容 简 介

本书内容主要由两部分组成。第一部分包括第一章至第六章，重点介绍Prolog的基本结构、语法及预定义谓词；其目的在于帮助读者学习并掌握Prolog语言的基本要素。第二部分包括第七章至第十二章，重点介绍Prolog在图论、问题求解的状态空间搜索、专家系统等领域里的应用。通过应用实例的详细讲解，力求帮助读者掌握Prolog的程序设计方法与技巧，从而为读者熟练地开发Prolog的应用系统打下基础。书中的最后一章还对Prolog的原理，即它与一阶谓词逻辑之间的关系进行了扼要介绍。

本书可供大专院校师生以及计算机工作者学习与参考。

Prolog逻辑程序设计及应用

周 立 柱

☆

清华大学出版社出版

北京 清华园

北京通县向阳印刷厂印刷

新华书店总店科技发行所发行

☆

开本：850×1168 1/32 印张：8 1/4 字数：213 千字

1991年4月第1版 1991年4月第1次印刷

印数：0001～6000

ISBN 7-302-00800-0/TP·288

定价：2.80 元

前　言

自从世界上第一台电子计算机问世以来，计算机语言一直是人们使用计算机的主要工具。它经历了由机器语言到汇编语言，再到COBOL、FORTRAN、PASCAL、ADA等高级语言这样一个由低级向高级的演变过程。与机器语言和汇编语言相比，FORTRAN和PASCAL等高级语言极大地方便了用户，从而较大幅度地提高了劳动生产率。但是从本质上讲，这些高级语言与机器语言或汇编语言都属于同一范畴——过程性语言（也称之为命令性语言）。

过程性语言的特点是，当运用这些语言进行问题求解时，人们必须详细地给出求解步骤。或者说，用户要用语言提供语句确切地告诉计算机进行问题求解时每一步都应当做些什么。对于用户而言这还是一个不小的负担。显然，如果能有这样一种语言，它只要求用户告诉计算机要解决什么样的问题，而把如何求解的细节留给计算机去完成，则对于用户而言无疑是一种解放。我们把这样一种新型的语言称之为说明性语言(或描述性语言)。

与过程性语言相比，说明性语言具有这样的特点：当运用这些语言进行问题求解时，人们所做的主要工作是运用语言来描述要解决的问题。至于如何求解这一问题，用户可以不必操心，它将由计算机的内部机制来完成。因此，从本质上讲，说明性语言与过程性语言是两种不同类型的语言。前者的着眼点在于"用户要什么"，而后者却是"计算机如何做"。

Prolog语言是一种说明性语言，它是英文PROgramming in LOGic的缩写。其理论基础是一阶谓词逻辑。它是人们把逻

辑作为程序设计的一种语言的努力结果。在运用Prolog进行程序设计时，重点在于对那些与问题有关的对象间的逻辑描述。在这种逻辑描述的基础上，Prolog运用自身具有的问题求解机制寻求答案。

本书从下述两方面对Prolog进行介绍：一、Prolog的语法及基本机制。二、Prolog在某些领域里的应用。因此，本书的重点放在如何使用Prolog求解问题的实际方面。而对于它的理论基础，我们只在第十三章里作一扼要介绍。对逻辑程序设计理论感兴趣的读者可以阅读其它有关的资料。

目前，已经出现了不少的Prolog版本。各版本在语法及语义上都略有不同。我们在介绍Prolog时将按照英国爱丁堡大学开发的Dec-10 Prolog系统的语法及语义进行，这是因为该版本具有广泛性。

与面向过程的FORTRAN、PASCAL不同，Prolog是面向问题的。它在相当大的程度上改变了人们运用计算机语言进行程序设计时的思维方法。使用Prolog进行程序设计常常成为一种有益的智力训练。

学习Prolog程序设计并不需要人们必须具备一定的程序设计经验。十分有趣的是，对于过程性语言过分熟悉而产生的某些习惯在很多情况下反而会妨碍对于Prolog的掌握。这是因为在过程性语言里很多习以为常的事，例如赋值语句，在Prolog里都是"忌讳"的。在学习Prolog时应当十分注意这一点。不要想当然地把FORTRAN、PASCAL等语言的编程习惯运用到Prolog里。

本书共由13章组成。第一章到第六章介绍Prolog的基本语法、结构及预定义谓词。第七章介绍Prolog程序设计中的某些技术。第八章到第十二章介绍Prolog在有关方面的应用。第十三章简述Prolog与一阶谓词逻辑之间的关系。

本书的最初目的是为大学计算机专业的学生学习Prolog提

供一本教材。但书中各章的安排既相互联系又相对独立，因此，它们可以灵活地组合，以满足不同读者的需要。下面我们以表的形式给出各章可以形成的组合、这些组合所能满足的需要及其对读者预备知识的要求。

各章组合	可满足的读者需要	基础知识要求
1	快速了解什么是Prolog语言及其基本特点	无
1，10	快速了解Prolog语言及它与一阶谓词逻辑间的关系	对一阶谓词逻辑有所了解
1，2，3，4，5，6，7	掌握Prolog语言的语法、结构、预定义谓词及其程序设计的基本技术	无
8，9，10，11 12	了解Prolog在各方面应用开拓新的应用领域	基本掌握Prolog

目　录

第一章　Prolog简介

使用Prolog语言进行程序设计的主要任务是描述问题所涉及的物体或对象之间的形式关系。一个Prolog程序由下述三部分组成：

1．说明关于事物或对象之间的关系；

2．定义规则；

3．询问。

本章将通过一个小例子对 Prolog 程序的组成进行概括性说明。由此可以看到Prolog语言与Fortran、Pascal等过程性语言相比所具有的鲜明特点。

1.1　说明事物或对象之间的关系

假定有一个由积木a、b、c、d、e、f组成的小小的"积木世界"。在这一"积木世界"里，各物体之间具有f在c上、c在b上、b在a上和e在d上的关系。这些关系可用图1.1.1来表示。这些关系在Prolog中被称为"事实"。按照Prolog的语法，它们可表示如下：

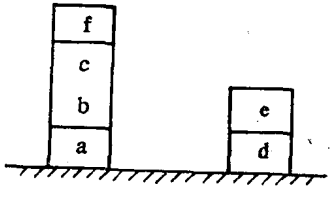


图 1.1.1　积木世界

```
on(f, c).
on(c, b).
on(b, a).
```

on(e, d).

在这些事实中，括弧前的on被称为"关系名"或简称为"关系"；括弧内的f、c、b、a、e、d等被称为"自变元"，表示具体事物。Prolog的语法规定，关系必须是以小写字母开头的一个字符串，自变元必须在括弧内用"，"隔开。括弧内的顺序反映了事实所说明的关系，在本例中"b在a上"表示为on(b, a)，而不是on(a, b)。程序设计人员可以根据自己的习惯选择顺序。然而这种顺序--旦选定，它就应该遵循固定的解释，否则将造成混乱。表1.1给出另一些符合Prolog语法规定的事实及其语义解释。

表 1.1 Prolog的事实表示

事　　实	解　　释
block(a).	a是一块积木
color(sun, golden).	太阳的颜色是金黄的
study(wang, computer, 4).	老王学了4年计算机

请注意，在事实的最后总是用一个"."表示结束。这对于Prolog是必不可少的。

表示关系的on、block、color、study也被称为"谓词"。根据谓词表示的关系中自变元的个数，block被称为一元谓词，on、color被称为二元谓词，study被称为三元谓词。Prolog对于事实中的自变元个数并无限制。程序设计人员可以根据实际需要自行决定。

对于表达事物之间关系的事实，必须把它输入Prolog系统，才能为Prolog所利用。为了输入有关事实，应首先进入Prolog系统[1]，并在终端上得到提示符"？ —"；而后再使用Prolog的预

[1] 关于如何进入Prolog系统，请参阅Prolog用户手册

定义谓词——即Prolog自身所具有的谓词"assertz"，把它们存入Prolog中。下面是存入一条事实的操作命令，用下划线标出必须由用户输入的信息：

```
? -assertz (on(b, a)).
   yes
? -
```

注意，必须用"."结束命令并换行，这时系统回答"yes"，表示已将事实存入。重复执行assertz命令，就可以输入上面所讨论的积木世界中的所有事实。

1.2 用规则定义关系

除了用事实来表示事物之间的关系外，Prolog还允许用户用"规则"来定义事物间的新的关系。这些新关系可以利用已有的事实。例如，在前面讨论的"积木世界"里，可以把图1.2.1所示的积木T、M与B之间的关系表示为"T在M之上且M在B之上"。我们给这个新关系命名为"three-store"。这种新关系被称为"规则"。按照Prolog的语法，可以把它写成

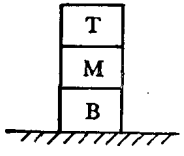


图 1.2.1 三层积木

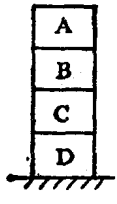
```
three_store(T, M, B) :-on(T, M), on(M, B).
```

这里，由":-"和连字符"-"组成的":-"读做"若"、"如果"。在on(T, M)与on(M, B)之间的逗号","表示"并且"、"与"的含义。上述规则读做"T、M，与B构成三层结构，若T在M之上，并且M在B之上"。

在定义规则three_store时，我们使用了大写字母T、M、B。在Prolog中这样的大写字母或以大写字母开头的字符串被称为"变量"。顾名思义，它们在程序执行中其具体内容是可以变化

的° 与变量相对应的是常量，常量在Prolog中以小写字母开头，例如"积木世界"里的a、b、c、d等都是常量。常量用来表示具体的对象。

已经定义了的规则在 Prolog 中又可以被用来定义其它规则。图1.2.2示意了A、B、C、D四个积木构成的四层结构以及用规则three_store定义的A、B、C、D之间的关系four_store (A, B, C, D)。Prolog的这种利用规则来定义规则或者说利用关系来定义关系的功能赋予它自身很强的描述能力。




```
four_store (A, B, C, D) :-
        three_store (A, B, C) ,
        three_store (B, C, D) .
```

图 1.2.2 用three_store定义的four_store

与输入事实的操作一样，可以用下述命令把规则存入Prolog：

```
? -assertz (three_store (T, M, B) :-
          on(T, M), on (M, B) ) .

  yes
? -assertz (four_store (A, B, C, D) :-
          three_store (A, B, C) , three_store (B, C, D) ) .

  yes
```

1.3 对象及其关系的询问

通过说明的事实以及定义的规则，我们完成了对于客观世界的逻辑描述。现在，在Prolog里已经包含了"积木世界"的下述事

实与规则：

```
on (f, c) .
on (c, b) .
on (b, a) .
on (e, d) .
three_store (T,M, B) :-on (T,M) , on (M,B) .
four_store (A, B, C, D) :-three_store (A, B,C) ,
                          three_store (B, C, D) .
```

以上事实与规则的集合被称为"数据库"。

为了求解我们感兴趣的问题，必须向Prolog提出询问．只有通过询问，Prolog才能调用有关事实或规则，给出答案。下面我们给出一些问题求解的例子，在这些例子中"？-"是进入 Prolog 后 Prolog 的提示符，带下划线的信息仍然表示用户从终端的输入，其它为Prolog的求解答案。

```
? -on (b, a) .
  yes
```

这一询问的目的是要证实b在a上是否成立。由于数据库中已含有此事实，因此Prolog回答yes。

```
? -on (X, d) .
  X=e;
  no
```

这一询问的目的是要查找在d上的积木。Prolog 查找数据库，给出答案 e，用户敲入"；"要求得到另一答案。由于数据库中不含有其它积木与 e 的on关系，因此Prolog回答no。

在上面的询问中，用小写字母开头的常量表示已知事实，用大写字母开头的变量表示要求解的答案。对于 Prolog 给出的答案，用户有两种选择：

(1) 敲入"．"表示结束询问；

(2) 敲入";"表示求解另一答案。

下面再给出一些利用规则进行求解的例子。

```
? -three_store (c, b, a) .
   yes.
```

这一询问的实质是要证实c、b、a是否构成一个三层结构。根据three_store的定义，以及数据库中含有的on(c, b)与on(b, a)事实，three_store(c, b, a)成立，因而Prolog回答yes。

在对规则的询问中，同样可以使用变量。例如，在上面的询问中，把常数a、b、c换为变量Fst、Snd、Trd，于是问题就成为寻找在积木世界中哪三个积木具有我们所定义的三层结构关系：

```
? -three_store (Fst, Snd, Trd) .
   Fst=c
   Snd=b
   Trd=a;
   Fst=f
   Snd=c
   Trd=b;
   no
```

根据已有的数据库中on的关系以及规则three_store的定义，读者不难验证上面的答案。

值得注意的是，针对three_store提出的询问的答案并不是从Prolog数据库中直接得到的，而是利用已有事实和规则进行推导的结果。这显示了Prolog所具有的推理能力。

在进行询问时，还可以使用","进行"合取"即"与"的询问，例如：

```
? -three_store (X, Y, b) , on (b, Z) .   (b是否为三层积木中的
   X=f                                    底层，并且b在某一物体
   Y=c                                    上，得到解答Z=a)
```