

TURBO PROLOG

工具库

潘金贵 等 编译
陈世福 审校

南京大学出版社

NANJING UNIVERSITY PRESS

73.87221
750.1

TURBO PROLOG 工具库

潘金贵 陈兆乾 编译
谢俊元 王锡江

陈世福 审校

JS98/15

南京太学出版社

内 容 简 介

Turbo Prolog 语言是一种可在IBM PC 及兼容机上高速运行,较为实用的编译型 Prolog 系统,其配套工具 Turbo Prolog Toolbox 进一步增强了 Turbo Prolog 的应用能力。

本书通过大量通俗易懂,并具有应用价值的完整程序实例,详细介绍了 Toolbox 中提供的设计用户接口,屏幕布局,作图,通信以及访问 Reflex,dBASEⅢ, Lotus 1-2-3, Symphony 的文件和自动生成语法分析程序等80多种工具及其使用方法。

本书不仅可供高等学校的教师、研究生、大学生用作讲授和学习Prolog语言的教学参考书,亦可供从事人工智能、知识工程、软件工程等的计算机工作者和广大科技人员使用。

Turbo Prolog Toolbox 既可配合Turbo Prolog 1.1版又可配合 Turbo Prolog 2.0版使用。因此,与本书配套的软件有Turbo Prolog Toolbox 系统(2枚盘片)、Turbo Prolog(1.1版2枚盘片,2.0版4枚盘片)语言统及其汉化版系本。

TURBO PROLOG工具库

潘金贵 陈兆乾 编译
谢俊元 王锡江

陈世福 审校

南京大学出版社出版

(南京大学校内)

江苏省新华书店发行 江苏省阜宁印刷厂印刷

1988年6月第1版 1991年5月第3次印刷

开本: 787×1092 1/16 印张: 19 字数: 464千

印数: 10001-20000

ISBN 7-305-00160-0

TP·13

定价: 6.70元

出 版 说 明

人工智能是50年代兴起的一门新兴学科。30多年来，研究的进展和取得的成就都十分惊人。因此，它被誉为本世纪的重大科学技术成就之一。

人工智能是计算机科学的一个重要分支，它的研究领域十分广泛，涉及模式识别、物景分析、自然语言理解、数据库的智能检索、博弈、自动定理证明、自动程序设计、专家系统、机器翻译、机器学习、机器人等领域。随着人工智能的基本理论和技术的深入研究和广泛应用，它不仅正在深刻地影响着计算机科学的其它领域，而且对信息科学、控制科学、系统科学、数学、心理学、电子学、生物学、医学、语言学和哲学等学科产生愈来愈深刻的影响，并带来巨大的社会效益。

应广大科技人员、计算机工作者、研究生、大学生系统地学习人工智能的需要，我们在近期内组织出版一套具有特色的人工智能丛书：

《人工智能导论》

《TURBO PROLOG 程序设计》

《TURBO PROLOG 工具库》

《知识工程语言与应用》

序 言

PROLOG 是70年代初期由法国马赛大学 Alain Colmerauer 等人作为一种方便的逻辑程序设计工具而开发的。自从日本宣布把 PROLOG 语言作为第五代计算机的核心语言之后,倍受世界注目。随着人工智能技术的普及和应用,近年来,PROLOG 语言在我国也迅速流行起来。

1986年由美国 Borland 公司研制的编译型 Turbo Prolog 语言,是目前微型机环境上颇受欢迎的 PROLOG 系统。不到一年功夫,全世界的用户数已愈10万,现仅美国就有30多所大学在教学和研究中选用 Turbo Prolog 语言。

为了便于在国内推广使用,我们对 Turbo Prolog 语言进行了汉化和二次开发,于1986年10月和1987年1月分别研制成功中西文 Turbo Prolog 1.0 版、1.1 版,并用它开发了“基于新构造控水理论找水专家系统 NCGW-1”。

我们在利用 Turbo Prolog 进行教学和研究中,体会到 Turbo Prolog 具有以下特色:

(1) 菜单驱动。Turbo Prolog 提供了一个友善的用户界面,包括多窗口、菜单系统。在主菜单窗口上列出了常用命令,而对于初学者,只须了解这些常用命令,按照系统的提示和菜单引导,就可以边上机边深入学习。所以 Turbo Prolog 给初学者的印象是:环境友善,容易入门。

(2) 自给自足。Turbo Prolog 提供了一个自给自足的软件环境,具有内部编辑程序,能自动连接,生成目标文件。

(3) 快速。编译和运行速度快。Turbo Prolog 在编译时,采用积累方式,编译中发现错误能自动回到编辑状态,并找出出错位置,等待修改,改完重编时,原来已编译好的部分基本不动,从出错点的附近重新开始编译,大大提高了效益。此外,装入数据库速度和目标码的运行速度都很快。

(4) 灵活。能与汇编、C、PASCAL 等语言编写的过程实现自然和谐的连接。

(5) 以高代低。用高级语言的标准过程和标准函数实现了一系列低层控制,如屏幕、窗口、颜色、声音、图形、绝对地址存取、数据端口控制、调用 DOS 软中断等。

由于以上特征,使得 Turbo Prolog 成为一种较为实用的 PROLOG 系统。Borland 公司1987年推出的 Turbo Prolog Toolbox (以下简称 Toolbox) 更增强了 Turbo Prolog 的应用能力。Toolbox 提供了80多个不同类型的工具(源程序达8000多行)和40多个样本程序。包括:构造友善的用户接口,快速设计屏幕布局,绘制事务图表,访问 Reflex、dBASE III、Lotus 1-2-3 和 Symphony 等系统的数据文件,自动生成语法分析程序等功能,能够帮助使用 Turbo Prolog 的程序人员开发实用专家系统,研制专家系统开发工具和环境,实现自然语言理解和机器翻译系统,开发办公自动化系统、决策支持系统,编制图形软件、文件管理、通信系统实用程序,构造编译程序和集成软件包等应用。

本书根据美国 Borland 公司1987年出版的《Turbo Prolog Toolbox》一书及配书软件编译,重点介绍 Toolbox。使用 Toolbox 需 Turbo Prolog 1.1 版的支持,故第一、二章补充

简介了 Turbo Prolog 语言 (1.1 版) 及其菜单系统, 以帮助读者快速掌握 Turbo Prolog 系统的主要使用方法, 若要进一步了解 Turbo Prolog 语言系统, 可阅读参考文献 [1—3]。

第三章介绍了 Turbo Prolog Toolbox 的安装, 第四章到第九章详细介绍了各种工具及其使用方法。希望全面了解的读者可以顺序阅读这些章节, 对于急需使用某一方面工具的读者, 完全可以先阅读最适合自己需要的那些章节。

最后, 我们给出了“参考指南”和两个附录, 为读者集中了 Turbo Prolog Toolbox 提供的工具谓词和 Turbo Prolog 系统提供的标准谓词的详细信息, 便于使用时查阅。

本书由潘金贵、陈兆乾、谢俊元、王锡江同志分工合作完成。其中序言, 第二章, 第三章, 第八章, 第九章以及附录 B 由潘金贵同志完成; 第一章, 第六章, 附录 A 由谢俊元同志完成; 第四章, 第五章由陈兆乾同志完成; 第七章由王锡江同志完成; 第十章由上述同志共同完成。

本书的编译工作是在陈世福副教授的主持下完成的, 并由他审校全文, 最后修改定稿。谢琪、陆庆文、陈彬、姚威力、潘志庚等同志为此书付绪出版作出了辛勤劳动, 值此表示谢意。

限于水平和时间仓促, 书中不妥及错误之处, 恳请读者指正。

编译者 1987年8月
于南京大学计算机科学系

目 录

序 言

第一章 Turbo prolog 语言简介	(1)
1.1 Turbo Prolog 语言.....	(1)
1.1.1 Turbo Prolog 语言的特点	(2)
1.1.2 Turbo Prolog 的主要应用领域	(2)
1.2 Turbo Prolog 的环境.....	(2)
1.3 Turbo Prolog 的程序结构.....	(3)
1.3.1 程序结构.....	(3)
1.3.2 领域说明.....	(4)
1.3.3 谓词说明.....	(5)
1.3.4 子句.....	(5)
1.4 关于中西文 Turbo Prolog.....	(7)
第二章 Turbo Prolog (1.1版) 的菜单系统和新特征	(8)
2.1 启动 Turbo Prolog 系统.....	(8)
2.2 主菜单.....	(9)
2.2.1 运行命令(Run).....	(9)
2.2.2 编译命令(Compile).....	(9)
2.2.3 编辑命令(Edit).....	(9)
2.2.4 选择菜单(Options).....	(9)
2.2.5 文件菜单(File).....	(10)
2.2.6 设置菜单(Setup).....	(12)
2.2.7 退出命令(Quit).....	(15)
2.3 Turbo Prolog 1.1版的新特征.....	(15)
2.3.1 新扩充的功能.....	(15)
2.3.2 新增加的谓词和样本程序等.....	(17)
2.3.3 Turbo Prolog 1.1版对1.0版的改进.....	(18)
第三章 Turbo Prolog Toolbox简介	(19)
3.1 Toolbox 的用途.....	(19)
3.2 Toolbox 提供的工具.....	(19)
3.3 Toolbox 的安装和使用.....	(20)
3.3.1 工具领域和谓词.....	(20)
3.3.2 运行样本程序.....	(20)
3.3.3 在硬盘上安装 Toolbox 的方法.....	(21)
3.3.4 在软盘上安装 Toolbox 的方法.....	(21)

第四章 用户接口工具	(23)
4.1 定义状态行工具	(23)
4.1.1 定义状态行工具谓词	(23)
4.1.2 一个使用状态行的样本程序	(23)
4.2 构造各种菜单工具及其使用	(24)
4.2.1 基本菜单	24)
4.2.2 长菜单	32)
4.2.3 框型菜单	34)
4.2.4 单行菜单	36)
4.2.5 下拉菜单	37)
4.2.6 树型菜单	41)
4.3 行输入工具	(43)
4.3.1 三个行输入工具谓词	(43)
4.3.2 行输入工具的使用	(44)
4.4 文件名输入工具	(45)
4.4.1 读入文件名工具谓词	(45)
4.4.2 文件名输入工具的使用	(46)
4.5 建立上下文敏感帮助工具	(47)
4.5.1 上下文敏感帮助工具谓词	(47)
4.5.2 使用上下文敏感帮助工具的样本程序	(49)
4.5.3 工具谓词中的帮助上下文	(50)
4.6 改变窗口尺寸工具	(50)
4.6.1 改变窗口尺寸工具谓词 <code>resizewindow</code>	(50)
4.6.2 在程序中使用 <code>resizewindow</code>	(50)
4.7 BIOS调用工具	(51)
4.7.1 设置和读出校验开关	(51)
4.7.2 改变屏幕边缘颜色	(51)
4.7.3 从目录读文件名	(51)
第五章 屏幕布局工具	(53)
5.1 基本屏幕布局的定义	(53)
5.2 屏幕处理程序的基本用法	(55)
5.3 <code>SCRDEF.PRO</code> 工具的功能	(57)
5.3.1 定义屏幕布局	(58)
5.3.2 保存屏幕布局	(59)
5.3.3 装入屏幕布局	(59)
5.3.4 编辑布局定义文件	(59)
5.4 定义字段的值和动作	(59)
5.4.1 字段的值	(59)
5.4.2 字段的动作	(62)
5.4.3 定义非输入字段	(65)
5.4.4 字段描述工具的使用	(65)

6.5	三个样本程序.....	(71)
5.5.1	俱乐部分类程序.....	(71)
5.5.2	记录销售帐务.....	(76)
5.5.3	标签打印程序.....	(81)
5.6	屏幕处理程序的高级功能.....	(86)
5.6.1	定义新的专用键.....	(86)
5.6.2	字段的帮助信息.....	(87)
5.6.3	增添自定义屏幕类型.....	(87)
5.6.4	交替使用几种屏幕画面.....	(89)
5.6.5	从旧屏幕定义建立新屏幕定义.....	(89)
5.6.6	打印格式化报表.....	(90)
第六章	图形工具.....	(94)
6.1	各种不同的坐标系.....	(94)
6.1.1	虚屏坐标.....	(94)
6.1.2	五个低级工具谓词和虚拟坐标.....	(95)
6.1.3	低级图形工具谓词的使用.....	(97)
6.1.4	定义标度.....	(99)
6.2	在屏幕上画坐标轴的工具.....	(101)
6.2.1	四个画坐标轴的工具谓词.....	(101)
6.2.2	在程序中使用标度和坐标轴.....	(102)
6.3	处理坐标的其他工具.....	(107)
6.3.1	设置标度.....	(107)
6.3.2	虚屏坐标和文本坐标之间的转换.....	(107)
6.3.3	标度坐标和文本或虚屏坐标之间的转换.....	(108)
6.4	画扇形图工具.....	(108)
6.4.1	画扇形图工具谓词.....	(108)
6.4.2	画扇形图工具谓词的使用.....	(109)
6.5	画条形图工具.....	(113)
6.5.1	画条形图工具谓词.....	(113)
6.5.2	画条形图工具谓词的使用.....	(11)
6.6	EGA调色板工具及其使用.....	(116)
6.7	装入和保存图形文件.....	(120)
6.7.1	有关的工具谓词.....	(120)
6.7.2	工具谓词 loadpic 的使用.....	(120)
6.7.3	用 loadpic 和 savepic 建立联屏显示.....	(121)
第七章	与远程串行设备的通信.....	(124)
7.1	通信工具谓词.....	(124)
7.2	硬件考虑.....	(124)
7.2.1	DTE-DTE 结构的引脚.....	(125)
7.2.2	DTE-DCE 结构的引脚.....	(125)

7.3 一般的串行通信.....	(125)
7.3.1 打开一个串行端口.....	(126)
7.3.2 关闭一个串行端口.....	(127)
7.3.3 传输状态信息的获取.....	(127)
7.3.4 从串行端口发送一个字符.....	(128)
7.3.5 从串行端口接收一个字符.....	(129)
7.3.6 输入和输出队列大小的获取.....	(129)
7.3.7 删除输出缓存.....	(130)
7.3.8 删除输入缓存.....	(130)
7.4 四个完整的样本程序.....	(130)
7.4.1 一个打印机驱动程序.....	(130)
7.4.2 终端仿真程序.....	(132)
7.4.3 具有超时功能的查询通信程序.....	(134)
7.4.4 使用 XMODEM 通信规程真子集的传输.....	(135)
7.5 调制解调器通信.....	(143)
7.5.1 发送一个中断信号到调制解调器.....	(143)
7.5.2 建立调制解调器方式.....	(144)
7.5.3 发送一个命令或数据到调制解调器.....	(144)
7.5.4 从调制解调器接收回答信息.....	(145)
7.6 一个菜单驱动的串行通信程序.....	(146)
第八章 访问数据库等其他系统的数据文件.....	(155)
8.1 访问 Reflex 文件.....	(155)
8.1.1 一次读出 Reflex 文件的所有记录.....	(155)
8.1.2 一次读一个 Reflex 记录.....	(157)
8.2 访问 dBASE III 文件.....	(160)
8.2.1 一次读出 dBASE III 文件的所有记录.....	(160)
8.2.2 一次读一个 dBASE III 记录.....	(161)
8.3 访问 Lotus1-2-3 或 Symphony 文件.....	(164)
8.3.1 读出所有的表元.....	(166)
8.3.2 读一个指定表元.....	(166)
第九章 语法分析程序生成器.....	(169)
9.1 语法分析程序的作用.....	(169)
9.1.1 语法分析程序的不同类型.....	(169)
9.1.2 词法分析: 扫描程序.....	(170)
9.1.3 文法描述.....	(170)
9.1.4 BNF 文法.....	(170)
9.2 Toolbox 语法分析程序生成器.....	(172)
9.2.1 语法分析程序生成器输入的指定.....	(172)
9.2.2 书写自己的文法.....	(174)
9.3 演示语法分析程序生成器的例子.....	(177)
9.3.1 文法的 Toolbox 描述.....	(177)

9.3.2	语法分析程序生成器的输入文法例.....	(178)
9.3.3	由语法分析程序生成器生成的语法分析程序领域文件的例子.....	(179)
9.3.4	生成的语法分析程序的扫描程序的例.....	(179)
9.3.5	生成的语法分析程序的例.....	(181)
9.3.6	扫描程序和生成的语法分析程序结合使用的例.....	(182)
9.3.7	出错处理.....	(183)
9.3.8	在一个实例中使用整个系统.....	(184)
9.4	语法分析程序生成器的编译和使用.....	(195)
9.4.1	语法分析程序生成器系统.....	(195)
9.4.2	有关生成的语法分析程序的系统.....	(196)
9.5	自举语法分析程序.....	(196)
第十章	参考指南.....	(199)
10.1	引言.....	(199)
10.2	Turbo Prolog Toolbox 文件及其环境.....	(199)
10.3	按用途分类的工具谓词.....	(202)
10.4	以字母顺序组织的工具谓词.....	(207)
附录A	工程的编译.....	(267)
A.1	编译Toolbox 中的工程.....	(267)
A.2	工程文件的建立.....	(267)
附录B	Turbo Prolog1.1版标准谓词.....	(268)
B.1	算术函数.....	(268)
B.2	标准谓词分类表.....	(269)
B.3	以字母顺序组织的标准谓词.....	(272)
附录C	在Turbo Prolog2.0下使用Turbo Prolog Toolibox.....	(285)
参考文献		(287)

第一章 Turbo Prolog语言简介

1.7 Turbo Prolog语言

PROLOG 是 Programming in Logic 的缩写, 意思为逻辑程序设计语言。简单地说, 逻辑程序设计是把逻辑系统当作计算机语言使用。

英国的 Kowalski 提出了一阶谓词逻辑可以作为计算机程序设计语言的思想, 1972到1973年间法国马赛大学提出的第一个 PROLOG 系统, 当时是为了一个法语对话系统的需要而研制的。尽管人们很快地在许多别的方面找到了 PROLOG 的应用(关系数据库、抽象问题求解、专家系统等), 但对该语言的实质及其重要性并无深刻的认识, PROLOG 只在大学和研究机构中流传, 对整个计算机界的影响是有限的。

到了70年代末, 情况发生了变化。我们在这里仅举出两点重大的变化。

第一, 以 FORTRAN 语言设计者 Backus 为代表的一批学者提出, “软件危机”之所以得不到很好的解决, 是因为通用的程序设计语言(如 FORTRAN, PASCAL 等)存在着某些根本性的缺陷。要进一步提高软件生产率, 必须研制表达能力更强的新语言; 要进一步提高软件可靠性, 新语言必须建立在严格的数学系统上, 以便对程序进行数学论证。Backus 等提出了 FP 等函数型语言, 它们满足上述两个条件, 并认为它们代表了新的方向。人们发现, 从1960年以来在人工智能领域广泛应用的 LISP 语言就是函数型语言的雏形。人们也认识到, PROLOG 语言同样满足上述两个条件。

第二, 日本的一些学者认为, 新一代计算机系统必须把当前计算机领域的几大分支——软件工程、关系数据库、人工智能和并行处理机——统一起来, 而 PROLOG 语言正是连结这几大分支的纽带。基于这样的思想, 日本决定将 PROLOG 作为第五代计算机系统的核心语言。

这样一来, PROLOG 语言的身份就起了变化, 它不再是数百种程序设计语言中的普通一员, 而成了众目所矚的一种了。

PROLOG 语言接近于自然语言, 易于学习使用。PROLOG 程序简洁明了, 易写易读, 易于正确性证明。PROLOG 的推理性质和特点很适合人工智能程序设计。所以它在关系数据库、数理逻辑、抽象问题求解、专家系统、自然语言理解、自动程序设计等人工智能的许多领域中都得到广泛的应用。

由于 PROLOG 语言对计算机的执行速度和内存容量提出了较高的要求, 基于微型计算机的解释型 PROLOG 语言(如 micro-PROLOG)由于受到功能和效率的明显限制, 所以一般只能用作教学示范或实验, 难以完成实用的人工智能课题, 影响了 PROLOG 语言的普及和实用化。

1986年, 美国 Borland International 公司为 IBM PC 及其兼容微机研制的 Turbo

PROLOG 语言是典型的 PROLOG 编译程序。它不仅保持了 Clocksin—Mellish 所描述的 PROLOG 语言的主要特点,而且比解释型 PROLOG 运行速度快得多。使得人们对微机上的 PROLOG 语言刮目相看。

1.1.1 Turbo Prolog 语言的特点

Turbo Prolog 语言成为人工智能领域的较为实用的开发工具,应归功于该语言的以下特点:

(1) Turbo Prolog 提供了既安全又方便的程序开发环境。同时能对源代码进行跟踪调试。

(2) 运行速度快,内存需求小。

(3) Turbo Prolog 具有丰富的独树一帜的内部谓词集。可以利用窗口、图形、声响等内部谓词编写图文并茂的程序。Turbo Prolog 系统内部建有完整的整数、实数算术操作以及科学计算函数。

(4) Turbo Prolog 是一个集成式、模块化的程序开发环境。可以将多个程序模块联结成一个可执行的程序。还能够同C、PASCAL、汇编等语言书写的模块联结成一个可执行的程序,提供了和这些语言之间交互的接口。

(5) 可与 DOS 方便自如地交互。

1.1.2 Turbo Prolog 的主要应用领域

由于上述特征使得 Turbo Prolog 成为微型计算机上一个较为实用的 Prolog 系统。适用于以下广泛的应用领域:

(1) Turbo Prolog 可以快速地实现任一实际应用系统的原型。

(2) Turbo Prolog 可以完全取接PC机的输入/输出端口,可以进行监督生产或实行控制。

(3) 可以实现动态关系数据库处理。

(4) Turbo Prolog 能进行符号处理,构造符号微积分等软件包,并可以进行语言转换,实施机器翻译,实现自然语言理解。

(5) 利用 Turbo Prolog 的演绎推理功能,很适合构造定理证明等各种人工智能软件包。

(6) 利用和其他语言的接口,实现和现有软件的交互。可充分利用原有的软件资源。

(7) 构造专家系统和专家系统工具。

1.2 Turbo Prolog 的环境

Turbo Prolog 为用户提供了下拉式菜单及多窗口的交互环境。主屏幕由五个窗口组成,如图1.1所示。

各窗口的功能和用途简述如下:

(1) 顶窗口是一个功能菜单,可以发布编辑、编译和运行一个程序、设置编译开关、执行有关文件操作、设置环境配置以及退出 Turbo Prolog 系统等命令。

(2) 信息窗口随时显示文件操作、编译步骤等当前活动信息。

(3) 对话框在程序运行时接收用户的输入并显示有关结果。

(4) 跟踪窗口在用户打开了跟踪开关时,可逐步或选择跟踪用户程序的运行,并显示有关跟踪信息。

(5) 编辑窗口具有和Word Star文本编辑相似的全屏幕编辑功能,并且提供一个辅助编辑窗口,为用户提供了两个文件同时编辑的功能。在编译时发现的源程序错误位置及错误信息同时在编辑窗口显示。用户可以不间断地编译和改正源程序中的错误。

有关 Turbo Prolog 窗口系统的进一步描述及有关操作将在第二章中详细介绍。

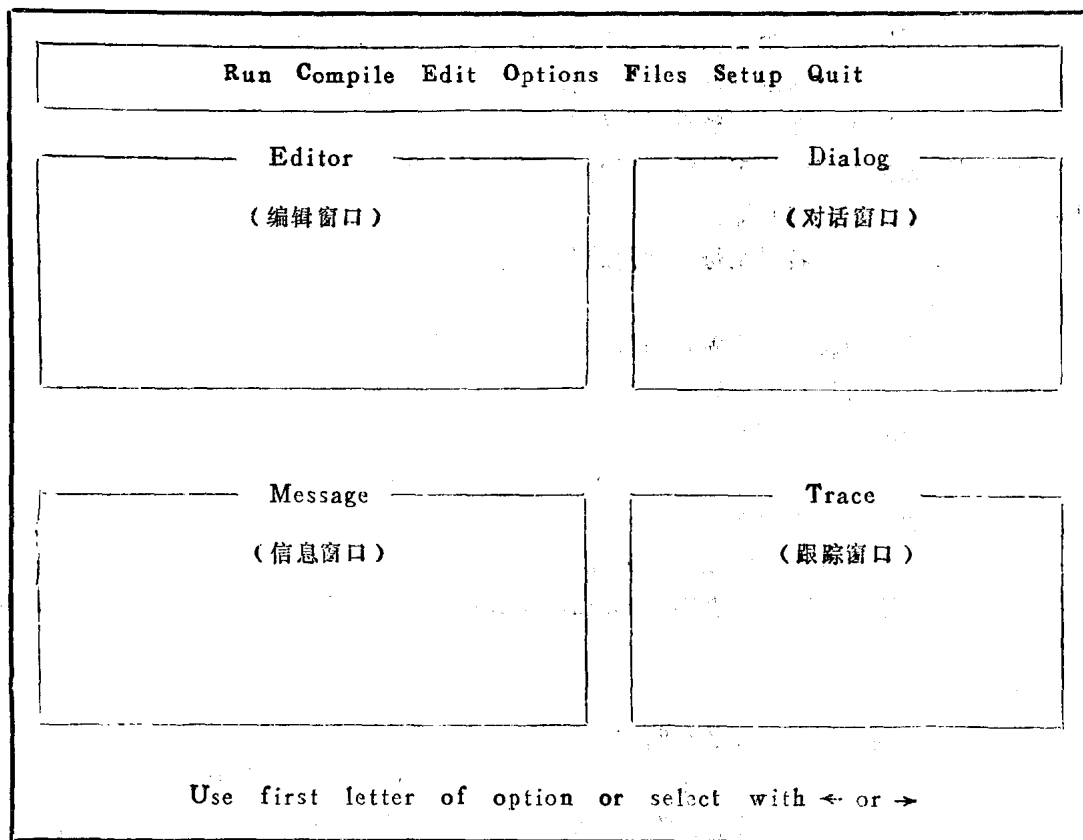


图1.1 主菜单及四个系统窗口

1.3 Turbo Prolog 的程序结构

Turbo Prolog 语言的语法和 Clocksin—Mellish 定义的 PROLOG 有明显的不同。Turbo Prolog 语言更加结构化并采用强制域类型检查。语法的改变主要是迎合了编译技术的需要。

1.3.1 程序结构

Turbo Prolog 程序是由若干说明段组成,有些段是可缺省的。

下面是 Turbo Prolog 程序的完整结构:

领域段	(domains)
全局领域段	(global domains)
数据库段	(database)
谓词段	(predicates)
全局谓词段	(global predicates)
目标段	(goal)
子句段	(clauses)

每一段用上面括号中的一个关键字标识。这些关键字按上述的顺序出现在程序中。其中：

- 领域段是程序的第一个说明段，该段由用户定义所有的局部领域类型。全局领域段说明能在别的程序模块中取接的领域类型。所有使用的非标准领域都应在这两段中说明。

- 数据库段描述了用在动态数据库中的谓词序列。Turbo Prolog 的动态数据库仅是事实的集合。对动态数据库中的事实的处理与普通谓词是不同的。对于前者系统可以采用提高速度的手段以及使用二级存储器。

- 谓词定义局部谓词名以及每个谓词的参数数目和领域。全局谓词段说明了可以被其他程序模块取接的谓词。

- 目标段允许用户标识一个固定的目标，即对 Prolog 程序的询问。如果用户想在内存试各种不同的目标，可以省去目标段。运行时系统将在对话窗口提示输入目标。这种方式类似于解释型的 Prolog 系统。但是编译成执行文件的程序必须包含一个非空的目标段。

- 子句段是程序的最后一段，它包含所有的事实和规则。在子句段中，除内部谓词外所用的谓词必须和谓词段的说明保持一致。

一个程序可以含有几个领域段、谓词段或子句段，但应遵循下述限制：

- 程序段必须以相应的关键字 domains, database, predicates, clauses 或 goal 开头。

- 编译期间，只能有一个目标段。

- 描述同一谓词的所有子句必须紧挨在一起。

- 编译期间，最可多碰到一个全局谓词段，且仅当前面没有普通谓词说明。

- 含有 database 谓词的程序段必须出现在所有全局谓词说明和一般谓词说明之前。

以下，我们对领域段、谓词段和子句段作进一步的描述。

1.3.2 领域说明(Domain Declarations)

领域段含有领域说明，有四种格式可以采用：

(1) name = d

这种格式把 name 说明为一个领域，该领域中的元素取自于标准领域类型 d，而 d 是整数、字符、实数、串或符号。这种说明用于语法结构相同，而语义不同的对象。

(2) mylist = elementDom*

mylist 是一个由 elementDom 中元素组成的表领域。elementDom 可以是用户定义的领域类型或者是标准类型领域之一。星号*读作“表”。现假定有如下表领域说明：

numberlist = integer*

显然，它定义了一个关于整数表的领域，例如：

```
[ 1, - 5, 2, - 6 ]  
(3) mycompDom = f1(d11, ..., d1N),  
              f2(d21, ..., d2M),  
              .....,  
              fM(dN1, ..., dNK)
```

这是一个由复合对象组成的领域，它指出了函子及其所有子元素的领域。

例如，设有

```
owns ( john, book(Wuthering_heights, bronte))
```

可以定义 owners 为

```
owners = owns(name, book)
```

其中 owns 是复合对象的函子，name 和 book 是其子元素的领域。等号的右部可由用分号隔开的多个选择项组成。每个复合对象必须含有一个唯一的函子和实际子元素的领域描述。

```
(4) file = name1; name2; ..., nameN
```

当用户需用符号名读写文件时，必须按上述方式定义文件领域。每个程序只能有一个文件类型领域，符号文件名作为待访问文件的选择。

1.3.3 谓词说明(Predicate Declarations)

谓词说明段以关键字 predicates 开头跟以一组谓词说明。谓词说明应指出谓词名及其参量的领域：

```
predname(domain1, domain2, ..., domainN)
```

其中 predname 是谓词名，而 domain1, domain2, ..., domainN 表示用户定义的领域或标准类型领域。

谓词亦可仅由名字构成。例如，可以用如下的规则来定义谓词 choose_teams:

```
choose_teams; -  
same_league(X, Y), never_played(X, Y),  
write(X, Y).
```

其中，谓词 choose_teams 就不带参量。

允许对同一谓词作多次说明。例如下面谓词 member 说明对数和名字均有效：

```
member(name, namelist)  
member(number, numberlist)
```

其中，name, namelist, number 和 numberlist 都是用户定义的领域。在对同一谓词作多次说明时，参量个数必须相同，但不要求具有一样的参数。

1.3.4 子句(clauses)

子句是对应于已说明的谓词有关的事实或规则。通常子句是一个原子，或者由一个原子后接：- 和一串原子表 构成。这些原子表之间用逗号或分号隔开。其中有些符号可以用下面符号代替：

符号 “: -” 可用关键字 if 代替。

符号“,”可用关键字 and 代替。

符号“;”可用关键字 or 代替。

例如, Turbo Prolog 事实

same_league(ucla, use).

是由单一原子(本身为一名字) same_league 和一个用括号括起来的项表(ucla, use)构成。

项可以是(简单的)常量、变量或复合项。现分别详细地说明如下。

(1) 简单常量(Simple constants)

简单常量是属于字符领域、整数领域、实数领域、串领域、符号领域和文件领域六种标准类型领域之一。

①字符(Character)属于字符类型领域(两个单引号括起来的8位ASCII字符)。ASCII字符可以用一个换码符(ESCAPE Symbol)即反斜线(\)后跟一个ASCII代码来标识。例如, \n 和 \t 分别产生一个换行字符和一个制表字符。反斜线后紧跟的任一其他字符产生那个字符本身。

②整数(Integer)属于整数领域类型,其取值范围为-32768到32767间所有整数。

③实数(Real)属于实数领域类型,其取值范围为 $\pm 1E-307$ 到 $\pm 1E+308$ 。实数的书写格式为一个正负号后接带(或不带)小数点的尾数部分,再接由一个正负号、E和指数组成的指数部分。格式中均不能有空格。格式中的正负号、小数点、指数部分均是可缺省的,必要时,整数将自动地转换成实数。

④串(String)属于串领域类型(是由双引号括起来的任意的字符序列)。字符串中可以包含由上面所述的ESCAPE序列即反斜线所产生的字符。

⑤符号常量(Symbolic constant)属于符号领域类型(是以小写字母开头的名字)。串也可作为符号,但符号要保存在一张查找表格中以便快速匹配。符号表格的缺点是造表需要耗费时间和占用存储空间。

⑥符号文件名(Symbolic filename)属于文件领域,是以小写字母开头的名字,出现在文件领域说明的右边,或是预先定义的符号名: printer, screen, keyboard 和 com1 等。

(2) 变量(Variables)

变量是以大写字母开头的名字,至于无名变量,可用一个下横线表示。若对变量的值不感兴趣,则可用无名变量表示之。未同其他项结合的变量称之为自由变量。例化后就称为约束变量。例如,已同一个项合一后的变量即为约束变量。谓词 free(X)判定变量X是否为自由变量,仅当变量X的值未知即X是自由变量时,该谓词成功。与此相反,谓词 bound(X)成功的条件是:变量X已约束为一个值。

(3) 复合项或结构(Compound terms or structures)

复合项(或结构)是一个由称为描述名的函子和一组称为成分(components)的其他对象组成的单对象。这些成分被括在圆括号中,且两两之间用逗号隔开。函子写在左圆括号之前。

例如,下述复合项由函子 author 和三个成分构成:

author(family, bronte, 1818)

复合项属于用户定义的领域。对应复合项 author 的领域定义为:

domains