

专家系统在设备检修中的应用

徐裕民 (齐齐哈尔市和平机器厂)

摘要: 专家系统是人工智能的一部分。有推理功能,可根据设备的故障现象推理出故障部位,给检修人员带来方便。应用 AutoLISP 语言编写专家系统,可将推理结论用图形显示在屏幕上,达到直观、形象、清晰的效果。本文将介绍这方面的内容。

1. 概述

专家系统(以下简称 ES)是人工智能的一部分。它有推理功能,可帮助人们寻找设备故障,诊断人身疾病...等等。近几年来正处于发展阶段。目前要给它下一个非常确切的定义,还有一定难度。但就其功能来讲,大家共同的认识是:ES 是计算机的一种程序,它能模拟人的思维方法去完成特定的任务。ES 可集中几个人类专家或几代人类专家的智慧,其诊断能力可超出任何一位人类专家的水平。只要计算机硬件不坏,它总是一丝不苟地工作。将寻找出来的故障点(或诊断出的人身疾病)用文字在屏幕上显示出来。随着生产的进一步发展,对 ES 又提出了新的要求,那就是:将寻找出来的故障点,用图形在屏幕上显示,达到直观、形象的效果,更方便于生产。用 AutoLISP 语言编写专家系统,就是出于上述目的。

2. ES 的原理与方块图

ES 一般包括五个部分,即:

- (1) 知识库——存放寻找设备故障所需的知识(规则)。
- (2) 数据库——存放已知事实(或称已知条件)。
- (3) 推理机——应用知识库中的知识与数据库中的已知事实相比较,推理出相应的结论。
- (4) 解释部分——解释为什么推理出这个结论。
- (5) 知识更新部分——对知识库中旧的或错误的知识删除,对新的知识添加。这几部分中最关键的是知识库和推理机。知识库内知识水平的高低,影响到推理结论是否准确(即准确性)。而推理机本身推理手段是否完善,又影响到计算机推进结论与人类专家诊断结论是否符合(即符合率)。一个优秀的专家系统,应该具有高水

平的专业知识和严密的推理功能,使其准确率和符合率均达到高的水平。知识库中的知识应由专业工作者提供,而推理机的程序,应由计算机软件工作者编写。

推理机就其推理方式而言,分正向推理和反向推理。而按推理精度来分,又可分为精确推理和不精确推理。根据具体情况采用不同的推理方式和不同的推理精度。我们根据设备具体情况,采用的是正向精确推理。

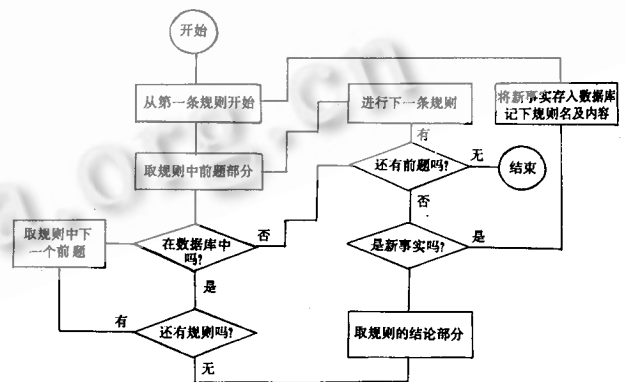


图 1 正向精确推理的方块图

推理前,先得 ES 的相应文件调入计算机内存。知识库和数据库中已分别装有知识(规则)和已知事实(故障现象)。推理开始,先从知识库中取出第一条规则(知识),再从第一条规则中取出第一个前题。将此前题与数据库中的事实一个一个地比较。若这条规则的几个前题

中有一个前题在数据库中找不到,则认为这条规则匹配失败,进行下一条规则匹配。若有一条规则的各个前题,均在数据库中,这时要进一步该规则的结论是不是已在数据库中。若已在数据库中,则认为该结论是旧事实,仍认为匹配失败。反之认为该结论是新事实,认为匹配成功。要进行以下四件事:

(1)用图形和文字显示其结论。使操作人员知道故障性质及故障位置,便于检修。

(2)将此规则存入专设的变量内,以便在解释时使用。

(3)将此规则的结论(新事实)添加到事实库中。

(4)将程序的运行指针回到程序的起点,从头开始,再匹配一次。因为数据库中新添了事实,已匹配过的规则很可能因新增添的事实而由不匹配变为匹配了。如此循环,一直到各条规则都与数据库中事实比较后,没有出现新事实时,才认为运行结束,退出程序。

整个运行过程,若一次也没有匹配成功,则表示知识库内各条知识(规则)均无法找出故障原因所在。此时屏幕应显示“本系统知识不够,找不到故障”。

3.AutoLISP 语言的应用

AutoLISP 语言有画图功能。能画各类专业的工程技术图纸。同时又能完成如图 1 所示的推理功能。因此 ES 用 AutoLISP 语言编写程序比较合适。该语言内容较丰富,本文内准备介绍有关推理用的内容。

(1)car 函数。其功能是取表中的第一个元素。如(car'(A B C))计算机执行结果取 A。在程序中取知识库中第一条知识,就是用这个函数。

(2)cdr 函数。其功能是将表中去掉第一个元素后,将留下的各元素组成一个新表。如(cdr'(A B C))计算机执行结果,新组成的表为'(B C)。在程序中,第一条知识匹配不成后,需取第二条知识,就将原来的表先留尾,再取头。此留尾即用 cdr 函数。

(3)cadadr 函数。这是 cad 函数和 cdr 函数的复合函数。计算机执行时,先执行字母“r”前一个字母的含意,然后逐个向左执行。如(cadadr (idl cif (R T)) (then (mA T))))计算机运行时先去头(即留尾)剩下((if (R T)) (Then (mA T))))再取其头,即(if (R T)),最后再取其尾,即(R T)。

(4)cdaddr 函数。这也是 cad 函数和 cdr 函数的复

合函数。仅多取一次表中的剩余部分。各字符的含意和先后执行顺序,同 cdadr 函数。

(5)cons 函数。其功能是将一个新元素加入表中,并作为首元素。如(cons 'A '(B C))执行后的结果是'(A B C)。在程序中,由键盘输入的各已知条件,组成一个数据库,就用该函数。

(6)cond 函数。其结构如下:

(cond (<测试 1>...<结果 1>)

(<测试 n>...<结果 n>))

执行这句时,先执行第一个子句,求<测试 1>的结果。若为 T,则这子句的其余各元素均求值。而其各子句不再执行。反之,若<测试 1>的结果为 N,则这个子句的其余各元素不再求值,执行下一子句。一直到某一子句的<测试 n>为 T 时,求该句各元素值。计算机运行后,将返回最后一个元素求得之值。在程序中,当有 n 个前题,要求每个前题与数据库中的已知事实相比较,就用这个函数。

(7)setq 函数。其功能是将某一值或某字符赋给某一变量。如:

(setq x1 "a1" x2 "a2" x3 "a3")

计算机运行后,变量 X1、X2、X3 内分别存入字符 a1、a2、a3。计算机返回 a3,在程序中,由键盘输入客观条件(故障现象)后,输入变量内,就用该函数。

(8)Defun 函数。其结构如下:

(Defun <函数名> (变量名) (过程描述))

其功能是,软件人员可根据程序的需要,自己定义该函数,执行软件人员编制的语句,达到独特的功能。

<函数名>可由软件人员自定,输入一串字符。

(变量名)为本函数运行中自用之局部变量。若没有局部变量,也可用空括号代替。

(过程描述)就是这个函数需要执行的各项语句。

(9) member 函数。其结构如下:

(member <expr> <Lisp>)

计算机运行时,搜索表内各元素,若表中有一个元素与表达式的结果相同,则计算机就返回表中相同元素及其后面各元素组成之表。反之若表中无一个元素与之相同,则返回 N。在程序中,当某一条规则(知识)的结论已在事实库中,便认为是旧事实。若不在事实库中,便认为是新事实。这时就用该函数。

4.推理结论的图形显示

当计算机推理出结论后,就应在屏幕上显示出来。显示的方式可以是文字的,也可以是图形的。一般来讲,文字描述总没有图形来得直观。尤其在工程技术上,除将有故障的部位显示外,还应注明故障原因、检修工艺、另件图纸……等等。配合画图仅可一并输出。

图形显示也有二种方法:第一种方法,用语句命令让计算机执行画圆,画直线……等等。这种办法太麻烦,编写程序时也容易出错,最好不用。第二种方法是在编程时,先用“手工”将所需图形画好。每张图插入总图一次,使该图成为总图的一个“块”。需要调入时,将该“块”插入图内。这种方法,效率高,所写程序也简明,一看就懂。我们在工作中就用插入“块”的方法。其语句格式如下:

```
(if (= car rule) 'id5)
(command "insert" "id5" '(0 0)
"","","")
```

其中第一个子括号(car rule)是取规则 rule 内之第一个字符串(规则内第一个字符串就是规则名)。此规则名若与后面的字符串(此例内为 id5)相同,则执行插入“块”的命令。在本例内即:

```
(command "insert" "id5" '(0 0) "","","")
```

语句内:Command 为执行 AutoLISP 语言的画图命令。insert 为插入“块”的命令。id5 为插入“块”的块名。'(0 0)为插入“块”的基准点。“”表示回车。

5.程序编制

编制程序前,先要考虑三点:

(1)将一台设备分成几大部分,如上、中、下和底部。其各部故障也分类归纳。好比看病,先想一下看内科还是外科。这样在运行时,哪部分的故障,就调入该部分的内容。既可减少内存容量,又加快了推理速度。

(2)编制推理机部分时,在各运转环节,要增加一些暂停和显示状态的有关命令。以便在调试时,可知道程序已运转到什么地方。若需修改时,便于寻找。待正式运行后,可逐句删除,以增加运行速度。

(3)有时一个故障点,需显示两张图形,这时可用屏幕上坐标的不同,将两张图形分开显示。也可先将第一张图形显示完后,再显示第二张图形。

下面介绍程序各部分的编写方法。

(1)事实库的建立(数据库)。建立一个自定义函数

fq

(a)先将事实库置空(setq facts n)

(b)用 getstring 函数,从键盘给变量赋值(设变量名为 aa),以便选择故障部位。

(c)用条件语句选择应向操作者询问设备的故障现象。由键盘回答后,其相应的变量(如 B102)就已赋值故障现象之真(T)或假(N)。

(d)用条件语句及函数 cons 将故障之真或假(即“T”或“N”)存入库内。

运行 fq 函数后,变量 facts 内就建立起事实库(数据库)了。

(2)知识库(规则库)的建立。建立一个自定义函数 rq

(a)先将知识变量及解释变量置空

(setq rules n rules-used n)

(b)用条件语句选择必需引入的知识。其格式如下:

```
(if c=aa "1") (setq rules 'C cidol
(if (b101 t)) (then ((1,.....)(2,.....))))))
```

rules 为知识变量。存储运行的各条知识。

if (b101 t)条件句的条件

(then ((1,.....)(2,.....)))为本条知识的推理结论。

运行 rq 函数后,在 rules 变量内就存有各条知识(规则)了。

(3)推理机和解释库

(a)自定义函数 recall。其功能是将知识库内各条知识的前题与事实库中各事实相比较。相同时返回非 N,相异时返回 N。

(b)自定义函数 test-if。其功能是当函数 recall 的结果为非 N 时,将本条知识的其余前题与事实库中的事实比较。若各前题均找到,返回非 N。反之返回 N。

(c)自定义函数 remember。其功能是若某条知识的推理结论,已在事实库中,则认为旧事。反之,则认为新事实。

(d)自定义函数 use-then。其功能是当 Test-if 返回 T,remember 函数返回 T 时(说明已匹配成功),将推理结论用图形显示在屏幕上。

(e)自定义函数 Try-rule。其功能是当 Test-if 函

数返回 T, use-then 函数返回非 N 时(即匹配成功), 将此条知识存入变量 rules-used 内, 待解释时使用。

(f) 自定义函数 step-forward。其功能是知识库内有一条知识匹配成功, 则返回 T, 否则返回 N。

(g) 自定义函数 deduce。其功能是当函数 step-forward 返回 T 时, 将程序从头至尾再运行一遍。若整个运行过程一次也没有匹配成功, 说明知识库内现存知识无法找到故障原因, 在屏幕上应显示“本系统知识不够, 找不到故障”。

(h) 自定义函数 D04。其功能是将其中所包函的各函数, 从头至尾全运行一次。

附录(1) 设备故障诊断专家系统推理机程序清单

```
(defun recall (fact)
  (cond ((member fact facts) fact) (t n)))
(defun test-if ()
  (setq ifs (cdadr rule))
  (while (and (/ = ifs n) (/ = a 4))
    (cond ((recall (car ifs)) (setq ifs (cdr ifs)))
          (t (setq a 4))))
  (if (and (= ifs n) (/ = a 4))
      (= t t)
      (setq a n))
  )
  )
(defun remember (new)
  (cond ((member new facts) n)
        (t (setq facts (cons new facts)) new)))
(defun use-then ()
  (setq thens (cdaddr rule))
  (while (/ = thens n)
    (cond ((remember (car thens))
          (print (car rule))
          (getstring "\n an use-then"))
          (setq rules-usedn (cons rule-used)) t)))
```

```
(defun step-forward()
  (setq rule-list rules) (setq c n)
  (while (and (/ = rule-list n) (/ = c 5))
    (setq rule (car rule-list)) (getstring "\n an 8a c")
    (t (setq rule-list (cdr rule-list))
      (getstring "\n an 8 c"))
    )
  )
  (if (and (= rule-list n) (/ = c 5))
      (= n t)
      (= t t)
    )
  )
  (defun deduce ()
    (setq e t)
    (while (/ = e n)
      (cond ((step-forward) (setq progress t)
            (getstring "\n an 9a c")
            (t (setq e n) (getstring "\n an 9b c"))
            )
            )
      )
    (if (= progress t)
        (getstring "\n 运行结束请敲一键")
        (getstring "\n 本系统知识不够,找不到故障")
      )
    )
  )
```

参考文献:

- [1] 知识处理与专家系统 何新贵编著 国防工业出版社 90年9月第1版
- [2] autocad-2 绘图软件包用户手册 华远技术公司编译 84年11月第1版
- [3] 专家系统原理与实践 林尧瑞等编著 清华大学出版社 88年4月第1版