计算机科学2000Vol.27№.3

行动推理中若干问题的研究*)

On Problems in Reasoning about Action

龙也挺 朱朝晖 陈世福

(南京大学计算机科学与技术系 南京 210093)

Abstract Reasoning about action is an important research area in artificial intelligence. After a brief description of what reasoning about action is, this paper describes the problems arising in formal action reasoning and various solutions to them. Finally, some concluding remark and comment to the direction of future research are given.

Keywords Reasoning about action, Commonsense reasoning, Nonmonotonic reasoning

1 引言

我们所面临的世界是不断动态变化的,一个智能系统往往需要对动态变化的环境做出反应,其中一个重要方面是对各种行动的结果进行预测、推理,以决定下一步的目标和动作。John McCarthy 提出进行行动推理(Reasoning about action)研究[1],并认为行动推理在常识推理中占有基础性的地位。至此以后,行动推理成为人工智能的一项重要研究内容。利用形式化的方法对世界和行动进行描述和推理构成了行动推理的主要内容。行动推理有时也被称为行动逻辑。在这里我们把关于行动和变化的推理总称为行动推理。行动推理有两个主要目的:(1)预测执行了一些行动以后,世界将变成什么状态。这个问题在时序逻辑中也称为时态预测(temporal projection)问题。(2)给出系统的一个状态,解释系统从初态经过哪些行动达到目前的状态,在时序逻辑中这个问题称为时态解释问题。

行动推理和时序逻辑、非单调推理、逻辑程序设计、知识表示、规划等研究方向有着密切的联系。一些时序逻辑系统和非单调推理技术就是为了解决行动推理中的难题而提出的。

2 行动推理的形式化表示方法

行动推理中目前广泛采用的表示方法主要有情景(situation)演算[1,2]和类似 STRIPS 系统的方法两种,其它的表示方法还有基于逻辑程序设计的事件演算[3],A[4,5]语言系列,GOLOG[2]语言及基于模态逻辑的表示方法[6,7]等等。

2.1 情景演算

情景演算是一种多型(many sorted)的二阶形式语言(注:各研究者使用的情景演算在表示上略有差别)。主要的型有:

情景:表示系统在某一时刻的状态,情景变元一般用 s 表示,初始情景用 S_0 表示。

行动:表示动作,例如:用 move(x,y,z)表示把 x 从 y 上移动到 z 之上的动作。

流(fluent):与情景有关的关系或函数。

在情景演算中,关系流 Holds(p,s)表示 p 在情景 s 下成立,函数流 Result(a,s)表示在情景 s 下执行动作 a 以后的情景。关系流 Poss(a,s)表示动作 a 在情景 s 下是可执行的。

例1 下面举例说明使用情景演算描述图1所示的场景。假设 clear(x)表示 x 块上没有其它的物体,color(x,s)=c 表示块 x 的颜色在 s 情景下为 c,on(x,y)表示 x 在 y 之上,等等。

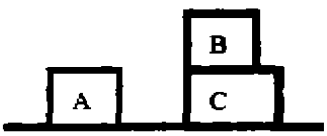

图1

Holds(onfloor(A),S_0) (1)
Holds(on(B,C),S_0) (2)
Holds(clear(A),S_0) (3)

*)本文得到自然科学基金项目(69775019)和国家教委博士点基金资助。龙也挺 硕士生,研究方向人工智能;朱朝晖 博士后,研究方向人工智能;陈世福 教授,博士生导师,研究方向人工智能。

$$Holds(on(x,z),Result(move(x,y,z),s)) \quad (4)$$
$$Holds(clear(y),Result(move(x,y,z),s)) \quad (5)$$
$$Poss(move(x,y,z),s) \equiv Holds(clear(x),s) \wedge Holds(clear(z),s) \wedge Holds(on(x,y),s) \quad (6)$$
$$color(x,s) \equiv c \supset color(x,Result(move(x,y,z),s)) \equiv c \quad (7)$$

其中，(4)(5)被称为效应公理，它们指出了某个行动的后果；(6)被称为行动预决公理，这类公理指出行动在什么情况下是可行的；(7)称为框架公理，指出了哪些行动不会改变哪些对象的属性，框架公理看上去似乎是不必要的，但在类似情景演算的推导逻辑系统中，框架公理是必需的，下面我们将讨论框架公理带来的种种问题。

2.2 STRIPS 系统

例1也可以用 STRIPS 系统的方法来描述。该系统是一个实用的机器人规划系统，所采用的办法十分简单：用一个逻辑公式的集合来描述系统某一时刻的状态，一个行动由前提条件、加入表和删除表三部分构成。前提条件满足时行动就可以执行，行动执行以后，在系统状态中添加加入表中的事实，删去删除表中的事实。例1可以描述为：初始状态{onfloor(a),onfloor(c),on(B,C),clear(a),clear(b)}。move(x,y,z)动作的描述为：前提：{clear(x),clear(z),on(x,y)}。删除表：{on(x,y),clear(z)}。添加表：{on(x,z),clear(y)}。

3 行动推理中的难题

从上例看，行动推理似乎很简单，然而把上述方法应用于一个比较复杂的系统时，研究者们很快发现了以下三个著名的难题：

框架问题(frame problem)：在行动推理中，除要指出一个行动会改变哪些对象的属性以外，还必须指出一个行动不会改变哪些对象的属性。在情景演算以及类似的推导逻辑系统中使用框架公理(如例1的(7))来解决这个问题。但是这种框架公理的数目非常多。假设要描述 A 条公理和 F 个状态，大致需要$2\times A\times F$条框架公理，不仅描述困难而且计算非常麻烦。虽然对一个确定的系统来说，列出所有的框架公理是可能的，但对一个动态的系统来说，描述的世界和动作本身也可能会变，在这种情况下，采用穷举法列出的框架公理就必须重写。

资格问题(qualification problem)：当描述一个行动时，通常要指出其可行的前提条件(如例1中的(6))。对一个复杂的系统而言，一个动作的前提条件可能会非常多而导致描述和计算上的困难。例如，当一个机器人试图搬动一个箱子时，它必须在箱子旁边，并且手中没有其它物品，箱子必须足够结实，箱子上不能有其它的物体，搬箱子时经过的路径上没有阻挡物，箱子的目的位置上没有其它的物体，甚至于箱子不会爆炸，不会解体，等等。

结果问题(ramification problem)：指一个行动产生的所有后果可能无法完全描述，仍以搬箱子的动作为例，一个机器人搬动了一个箱子以后，其可能的后果有箱子达到新的位置，箱子内的物体达到的新位置，箱子达到新的位置以后堵住了门，门堵住以后房间空气不通，等等，这些后果在传统的行动推理中都需要在搬箱子动作的描述中加以刻画。另外，有些资格问题实际上涉及到行动的后果，结果问题的解决办法也能解决部分的资格问题。

4 框架问题的解决方案

框架问题的难点在于两个方面，一是表达上的困难，二是计算上的困难。解决框架问题的主要思想是在形式系统中对常识性概念"惯性"(inertia)进行刻画。解决的办法主要有两种，一种是对表示方式进行改造，在表示方法中体现系统的"惯性"，另一种是在推理机制中体现"惯性"。

必须指出，框架问题带来的诸多麻烦与采用的表示方法有关。采用逻辑程序语言的表示方法时，可利用"失败即否定"，当系统推不出变化时就认为不变，避免框架问题带来的种种麻烦。采用 STRIPS 系统的表示方法也可避免这些麻烦，但这种方法存在着严重的缺陷，系统设计者必须给出这种加入删除表，同时还必须保证添加删除表的一致性，当系统要增加或删除一些状态描述变量时，这种添加删除表甚至会全部重写。

4.1 "简单方法"和流演算

在情景演算的框架中，改造表示方法以体现"惯性"的解决方案是设法改造效应公理，使其包含推理所需的足够信息。其主要方法有 Pednault 方法，Hass/Schubert 方法和 Reiter 的"简单方法"[2]，Pednault 的想法是引入一个"完备性假设"，认为效应公理刻画了所有流变化的情况，然后将原来的效应公理进行转化。Pednault 方法虽然提供了系统的转化方法，但需要提供很多"空"效应公理，而这些"空"效应公理的数目几乎和原来框架公理的数目一样多，因而实际上并没有将问题简化多少。Hass/Schubert 方法的主要思想是对效应公理中的行动进行量化，从而达到减少框架公理数目的目的，但 Hass/Schubert 方法中没有提供系统的转化方法。Reiter 的"简单方法"则综合了两者的优点。现简介一下。假设某一效应公理有如下形式：

$$\Phi_R^{\alpha} \supset R(\vec{t},Result(\alpha,s)) \quad (8)$$

式(8)指出了执行动作 α 以后有 R 关系成立，所以也称为正效应公理。式(8)可改写成逻辑等价式：

$$\exists y1,y2,\cdots,yk(a=\alpha \wedge \vec{x}=\vec{t} \wedge \Phi_R^{\alpha}) \supset R(\vec{x},Reslut$$

(a,s)) (9)

其中 y1、y2等为式(9)中除 s 以外的所有自由变元，$\vec{x}$ 为引入的新变量，把(9)记为

$$\gamma_R^+(\vec{x},a,s)\supset R(\vec{x},Result(a,s)) \quad (10)$$

相应地所有负效应公理可改写为：

$$\gamma_R^-(\vec{x},a,s)\supset\neg R(\vec{x},Result(a,s)) \quad (11)$$

引入"完备性假设"，即认为式(10)、(11)刻画了所有 R 变化的情况，式(10)、(11)可改写为：

$$\neg R(\vec{x},s)\wedge R(\vec{x},Result(a,s))\supset\gamma_R^+(\vec{x},a,s) \quad (12)$$

$$R(\vec{x},s)\wedge\neg R(\vec{x},Result(a,s))\supset\gamma_R^-(\vec{x},a,s) \quad (13)$$

把式(12)、(13)合并即可得到后继状态公理。在形式上看这简单了许多，但框架问题在计算上的困难没有解决。另外，该方法还有不能表示不确定的动作，添加动作和状态变量会改写整个后继状态公理集等问题。

为了解决"简单方法"中的计算上的困难，Thielscher[8]等利用流演算来改造效应公理，主要的想法是在改造效应公理时对流而不是动作进行量化。这样做的好处是，利用改造后的一条效应公理就可以推断执行某一个动作以后的整个状态。

4.2 非单调推理

非单调推理是人工智能的重要研究领域，著名的非单调推理系统有 Reiter 的缺省逻辑[9]，McMarthy 的限定论(circumscription)[10]，Moore 的自认知逻辑等等。事实上，行动推理是促使非单调逻辑诞生与发展的直接原因之一。缺省逻辑和限定论在行动推理中有广泛的应用。

4.2.1 缺省逻辑

在情景演算中使用缺省逻辑的做法是引入下列缺省规则：

$$\frac{Holds(x,s):M\ Holds(x,Result(y,s))}{Holds(x,Result(y,s))} \quad (14)$$

例如：在例1中加入式(14)作为推理规则，去掉框架公理(7)，仍可得出 color(x,move(x,y,z))=c。然而，缺省逻辑目前没有很好的计算方法，而且在进行行动推理时存在多扩充的问题，即给定一个状态和一个行动，可能有多个结果状态，另外缺省逻辑目前不能把各个扩充区分开来。关于多扩充的问题，下面还将继续讨论。在缺省逻辑中举一个极端的例子，如果我们同时有如下两条缺省规则：

$$\frac{Q(x):M\ P(x)}{P(x)} \quad (15) \qquad \frac{R(x):M\ \neg P(x)}{\neg P(x)} \quad (16)$$

并且同时有 Q(n)和 P(n)成立，那么，多扩充将是不可避免的；若取结果为所有扩充的交，那么我们就什么结论也得不到。虽然有些学者认为多扩充问题在某些情况下(如缺乏必要的信息)是符合客观实际的、合理的[11,12]，然而在某些情况下，多扩充产生了在直觉上不想要的结果。

4.2.2 限定论

在行动推理中，限定论应用得更广泛。MaCarthy[10]第一个试图在行动推理中使用限定论解决框架问题。做法是在情景演算中加入下列公理：

$$\forall f,a,s(\neg ab(f,a,s))\supset(holds(f,result(a,s))\equiv holds(f,s)) \quad (17)$$

以及一些反常公理，如：

$$\forall p,s\ ab(AWAKE(p),GOTOSLEEP(p),s)$$

(表示 p 睡觉以后不可能醒着) (18)

然后再利用限定论的方法进行推理。传统限定论的做法除了有计算上的问题以外，和缺省逻辑一样存在多扩充的问题。Hanks 和 McDermott[13]举了一个著名的时序推理中的难题 YSP(Yale Shooting Problem)问题，说明一些非单调逻辑会产生直觉上不想要的结果。YSP 问题简述如下：假设射杀一只鸭子，ALIVE，DEAD 表示鸭子的状态，LOADED 表示子弹上膛的状态，SHOOT 和 LOAD 表示射击和推子弹上膛的动作，形式化如下[13]

$$Holds(ALIVE,S_0) \quad (19)$$

$$\forall s\ Holds(LOADED,Result(LOAD,s)) \quad (20)$$

$$\forall s\ Holds(LOADED,s)\supset ab(ALIVE,SHOOT,s)\wedge Holds(DEAD,Result(SHOOT,s)) \quad (21)$$

$$\forall f,a,s(\neg ab(f,a,s))\supset(holds(f,result(a,s))\equiv holds(f,s)) \quad (22)$$

现在的问题是在从 S_0 出发经过动作序列 LOAD、WAIT、SHOOT 以后鸭子的状态是什么。期望的结果当然是 DEAD(鸭子是死的)成立。传统的限定论的做法为 Cirum(A,ab,*Holds*)，表示对 ab 进行极小化，同时 Holds 的解释可以变化。然而这种做法的结果却有两个，一个结果是 DEAD 成立，另一个结果是 ALIVE 成立。实际上第一个结果是从初始状态向后推理得出的，第二个结果是倒过来，从后向前推理得出的结果。

针对上述做法的缺陷，许多学者提出了各自的解决方案。一种想法是在系统中引入某种机制，使系统只能够从前向后推理。Lifschitz，Kautz，Shoham 从这个思想出发，分别提出了各自的形式化系统解决了 YSP 问题。他们在推理中采用了一种"时序极小"[13](chronologically minimal)方法，直观的想法是在推理中尽早使用反常规则，这种方法对解决 YSP 问题是有效的，但不能保证应用于其它问题同样有效，例如对反向推理和时态解释问题就无能为力。

随后研究者们发现，如果极小化的成分与情景无关，YSP 问题就不会出现。基于这种想法，Lifschitz[15]引入了因果关系，其做法是在情景演算中引入诸如式(23)—(25)形式的表达式和规则。

$$causes(load,loaded,true) \quad (23)$$

$$precond(loaded,shoot) \quad (24)$$

$$success(a,s)\wedge causes(a,f,v)\supset(Holds(f,result(a,s))\equiv v=true) \quad (25)$$

式(23)的意思是动作 load 导致 loaded 成立，式(24)的意思是动作 shoot 的前提条件是 loaded，式(25)的意思是动作 a 的前提满足并有 causes(a,f,v)，那么执行动作 a 以后有 f 成立。然后对谓词 causes 进行极小化，但采用因果极小化的方法，其表达能力受到了限制，许多与情景有关的约束关系无法用 causes 谓词来表达。Lifschitz[14]提出了一些改进的方法，主要的想法是把演算中的流分为基本的和非基本的流，非基本流的值由基本流的值所决定。这种方法的问题是缺乏通用性。

与 Lifschitz 的做法不同，Baker[15]采用了状态极小化的方法，做法是采用 Cirum(A,ab,Result)，而不是 Cirum(A,ab,Holds)，在极小化 ab 时让 Result 的解释变化，由于这种方法极小化的成分与情景无关，因而 YSP 问题也不会出现，而且 Baker 的方法更为自然，还为解决结果问题提供了方便。但这种办法不能用于不确定的动作效果，Kartha[17]中举了两个例子说明这一问题，Kartha 还指出，不确定效果的问题可以用 Sandewall[16]的方法来改进，主要想法是只使用系统中的部分公理进行极小化计算。Kartha 把描述初始情景下一些流成立的公理称为"观察公理"，并认为在极小化时由于观察公理的存在使结果不是直觉上想要的结果，因而在极小化时不考虑观察公理就可以得到正确的结果。

5 结果问题的解决方案

最初，在一个动作的形式化描述中需要描述该动作的所有后果，后来发现这是不合理的。目前，结果问题的主要解决方法是区别对待动作的直接效果和间接效果。在这方面，一些工作基于类似 STRIPS 系统的描述方法。

5.1 可能世界和可能模型方法

Ginsberg 和 Smith 使用了一种可能世界方法[19,20]，其基本表示方法和 STRIPS 系统相同，但在算法上作了改进，首次引入了"域限定"的概念，域限定是系统在任何时候都必须满足的条件。例如：在机器人搬箱子的例子中，可能有这么一条域限定公理：$on(x,y) \wedge y \neq x \supset \neg on(x,z)$意思是在任何时候，x 上面已有物体时，不能有另一个物体。

可能世界方法在描述行动时，用一组逻辑公式描述系统的状态和行动的直接后果，行动的直接后果公式集记为 C，行动执行前系统状态的公式集记为 S，在推理中不希望改变的逻辑公式集合记为 P，显然 S∪C 可能矛盾。可以找到一个公式集合 T，使 T 有如下性质：①T⊂S∪C；②C⊂T；③P⊂T；④T 是一致的。满足①～④的 T 称为一个可能世界，如果某个 T 是在集合论意义下所有 T 中最大的，那么这个 T 就是我们所求的结果。求 T 的一种算法是针对 C 中的每一个公式 q 构造一 q 的证明，假设证明为公式集合 S'，取结果状态集为 S－S'即可。

以图1为例，执行动作 move(B,C,A)，把 B 从 C 上移到 A 上。S＝{onfloor(A),onfloor(B),on(b,c)}，P＝{$on(x,y) \wedge y \neq x \supset \neg on(x,z)$}，C＝{on(B,A)}；动作执行以后 S∪C 不一致，对 C 中的每一个公式 q 我们构造一 q 的证明。这里 q 只有一条：on(B,A)。一on(B,A)的证明有(结合域限定公理)S_1＝{on(B,C)}取结果状态为 S－S_1，容易验证，S－S_1是协调的。

可能世界方法没有 STRIPS 系统中对应的删除列表。通过"域限定"来保证结果的一致性。因为只需要描述行动的直接后果，解决了部分的结果问题；若结果状态无法构造，就认为动作不可行，部分解决了资格问题。然而，这种方法也有多扩充的问题。虽然，作者认为这是由于信息不够引起的，但实际上这种域限定公理只是描述了系统状态变量之间静态的约束关系，而没有描述在推理时应该满足的动态约束关系，换言之，因限定公理的描述能力不足，多扩充是不可避免的，考虑下面的例子[1]：

例2 一个灯泡由两个串联的开关控制，初始状态下，开关一闭合，开关二打开，灯不亮，分别用 Sw1、$\overline{Sw2}$、$\overline{light}$表示。有域限定公理：

$$Sw1 \wedge Sw2 \supset light \tag{26}$$

容易验证，如果一个动作把 Sw1闭合，结合式(26)进行计算可得到两个结果状态{Sw1，Sw2，light}{Sw1，$\overline{Sw2}$，$\overline{light}$}。一种解决办法可以设法使 light 比 Sw2更容易变化(categorization-based approaches)，但这种方法只能针对具体的应用领域，不能一般化。实际上(26)式逻辑蕴含式，等价于$\overline{light} \supset \overline{Sw1} \vee \overline{Sw2}$，所以会有多扩充的情况出现。解决的另一个办法是改造域限定公理的形式，譬如改造成因果关系。

可能世界方法还有另外一个问题，它的计算结果和问题的表示形式有关，两个逻辑上等价的公式集合，可能会有不同的计算结果。例如：S1＝{p,p⊃q}，S2＝{p,q}，某个动作的结果集 C＝{¬q}，S1和 S2在逻辑上等价，但在 S1上执行该动作的结果有两个，{p}和{p⊃q}，而在 S2上执行的结果为{p}。Winslett 和 Marrian 提出了可能模型方法，解决了这个问题。其主要思想是，对一个状态描述 S，用一个模型来刻画，在这里，一个模型是一个原子命题公式的集合。假设动作执行前的状态 S1，模型 M1，执行动作以后，模型为 M2，如果某一个 M2在所有的 M2中，与 M1相比变化最小，就选取这个 M2为结果。因为模型中只有原子命题，所以逻辑上等价的公式集合的计算结果是一样的。但是，可能模型方法在解决例2时和可能世界方法有同样的多

扩充问题。

可能世界和可能模型方法都不能描述后果不确定的动作，为了解决这个问题，Woo[21]采用了一种 PAS 方法，Doherty 和 Lukaszewicz 采用 MPMA 方法[22]来解决这个问题。

5.2 采用因果关系描述间接后果

为了解决域限定公理描述能力不足的问题，Thielscher[11]引入了一种影响关系(influence information)，就例2而言，有关系(Sw1，light)，(Sw2，light)，表示 Sw1和 Sw2状态的改变可以影响 light 的状态，影响关系结合域限定公理可以生成因果关系，例如：

$$\overline{\text{Sw1}}\ \text{Cause}\ \overline{\text{light}}\ \text{if}\ \text{T} \quad (27)$$

$$\text{Sw2}\ \text{Cause light if Sw1} \quad (28)$$

式(27)的意思是$\overline{\text{Sw1}}$一定导致$\overline{\text{light}}$，式(28)的意思是在 Sw1打开时，Sw2导致 light 成立。因果关系与限定公理相比，更好地刻画了推理时系统应该满足的约束关系，就例2而言，上述的多扩充问题不再出现。

Lin[23]在情景演算的框架中引入因果关系来表示动作的间接效果，其表示方法和 Lifschitz[15]的做法有类似的地方，但出发点和目的均不同，Lin[24]引入了一个"自然动作"来表示间接效果和进行推理。Nakashima 等[25]也引入了因果关系进行推理。

必须指出，上述许多方法都要利用极小化原则，使结果状态和原状态在某种意义下尽量接近，关于这个原则也有争论，有些学者认为极小化不合理，在某些场合得不到直觉上想要的结果，原因之一在于极小化原则只是一种技术上的做法，实际情况中一个动作执行以后，结果状态和原状态相比，不一定要变化最小。

小结 经过多年的研究，行动推理已经有了很大的发展，但仍存在许多问题。虽然已经发展了许多逻辑系统来进行行动推理，但实用的还是 STRIPS 系统的方法(GOLOG 可能是一个例外[2])，原因在于其它各种方法的计算复杂性太高。很多行动推理的研究只是面向有限的几个问题，还没有形成一个完整的理论体系，许多工作只是技术上的"补丁"，做法缺乏直观的背景，就使用得最多的极小化原则而言，它是否符合客观实际情况也是值得商榷的，至于文中提到的各种解决办法，许多找不到直观背景，只有在理论和实际两个方面都作了深入的研究以后，才有可能构造一个通用的行动推理的框架，而目前，把行动推理应用于实用系统的工作还很少。

自90年代以来，随着分布式人工智能的发展，并发、并行和不确定的行动推理受到日益广泛的注视。与经典的行动推理技术相比，并发和并行的行动推理难度更大，试举一例[26]：

一张桌子上有一个花瓶，假设有两个动作：lift_left 和 lift_right，分别表示抬桌子的左边和桌子的右边。如果两个动作只做一个，桌子倾斜，花瓶会倾倒；但如果两个动作一起做，虽然桌子提高了，花瓶却不会倾倒。用经典的行动推理技术是无法推出这个结论的。

并发和并行行动推理存在着形式化表示的问题。由于事件演算中显式地引入了时间点，可以表示行动之间开始的先后关系，但各个动作之间相互作用的推理却存在着问题。Chen 等[27]将过程代数引入行动推理，很好地解决了花瓶问题，但它采用的表示方法没有时间点的概念，动作要么同一时刻开始和结束，要么一个接一个地执行，这与现实的实际情况有很大的差异。另外还有采用 STRIPS 系统方法来表示并发行动的[28]。

总之，行动推理目前还有很多尚待解决的问题，并发、并行与不确定的行动推理以及与实际相结合的行动推理系统最有可能成为行动推理未来的发展方向。

(参考文献共28篇，略)

(上接第84页)

- 在数据选择阶段，粗集方法和统计方法都可以实现对属性的选择；
- 在数据清理阶段，回归方法或神经网络方法可以用于对缺值的预测，统计方法或利用元规则的约束推理可以用于清除噪音和数据修正；
- 在数据预处理阶段，统计聚类或神经网络聚类方法都可以用于对连续数值的离散化，OLAP 或 AOI 方法可以改变原始数据的抽象层次；
- 在数据挖掘阶段，可以根据输入数据源的特点和任务的要求来选择合适的方法。

参考文献

1 Fayyad U M，et al. From Data Mining to Knowledge Discovery：An Overview. Advances in Knowledge Discovery and Data Mining，AAAI/MIT Press，1996

2 沈清，汤霖. 模式识别导论. 国防科技大学出版社，1991

3 Heckerman D. Bayesian Networks for Data Mining. Data Mining and Knowledge Discovery，1997(1)：79～119

4 Michalski R. Machine Learning：An Artificial Intelligence Approach，volume 1. Morgan Kaufmann，Sam Mateo，California，1984

5 洪家荣. 归纳学习-算法 理论 应用. 科学出版社，1997

6 Quinlan J R. Induction of decision trees. Machine Learning，1986

7 Pawlak Z. Rough Sets Theoretical aspects of reasoning about data. Kluwer Academic Publishers，1991

8 曾黄麟. 粗集理论及应用. 重庆大学出版社，1998

9 Chaudhri S，Dayal U. An overview of data warehousing and OLAP technology. SIGMOD Record，1997，26：65～74

10 Han J，et al. Knowledge Discovery in Databases：An Attribute-Oriented Approach. In：Proc. of the 18thVLDB conference. 1992. 335～350