缺席推理的逻辑框架

David Poole

摘　要

本文提出缺席推理的一个简单逻辑框架，其语义是正规一阶模型理论；它没有改变逻辑本身，而只是改变了使用逻辑的方法。我们探讨把推理看成一种简单的理论构成情形所产生的一些结果，而不期望推理只是我们知识的(用任意逻辑)的演绎。通过把缺席处理成预定义的可能假设，我们说明了这一思想是怎样包含Reiter缺席推理背后的直觉的。文中还讨论了有关多扩充问题的解决办法。所给出的全部例子是用一个叫做THEORIST的原型系统实现的。

1. 引　言

从理论上说，在任意一个能进行个体及其间关系推理的表示系统中，至少需要一阶谓词演算。然而，使用传统逻辑进行常识推理却存在单调性这一实际问题。对此已提出了许多建议，扩充古典逻辑允许其进行非单调推理。如果人们认识到逻辑的重要性（以及含义和语义的重要性），就似乎有两种方法来解决这一非单调性问题，

(1) 第一种方法是声明古典逻辑有许多东西是错误的，因此需要定义一种新的逻辑去处理非单调推理。沿这条路，势必定义一种语法、语义和一个证明过程并给出这种逻辑的合法性和完备性定理。

(2) 第二种方法是说古典逻辑没有什么错误，我们不应期望推理只是我们的知识的演绎。就此而论，可以这样来看J. McCarthy的限制 (circumscription)：缺席是隐式假设；我们必须增加限制公式，使这些隐式假设成为显式的。然后我们能够根据完备知识库进行演绎。

本文的建议采用第二种方法，尽管其使用的某些细节非常类似于限制，但是是一个很不同的方法。我认为非单调性是一个怎样使用逻辑的问题，而不是逻辑本身的问题。这符合Israel的精神。

有理由说一个推理系统是离不开假设和测试的。这方面已做了很多研究，尤其是在科学理论的构成上。

在这里我们考虑最简单的假设推理情形，即用户提供他们准备在解释中采纳的可能的假设形式。我们有意避开科学上最困难的问题："我们怎样产生这些假设"。

我们将阐明，来自一个固定可能假设集的理论构成思想，是缺席推理的一个自然而又简单的特性。我们不是为缺席推理定义一个新的逻辑，而是说明缺席推理是把推理视为理论构成而非演绎的自然产物。逻辑告诉我们：我们的理论预示的是什么。

本文试图辩论的主题是：如果一个人允许假设推理，那么没有必要定义一个新逻辑去处理非单调推理。对于我们的建议，我们给出了一个语义，并且说明怎样把它引入一个缺席推理系统。随后提供一种程序设计方法，使人们能避免许多疑难的多扩充问题。

2. 语　义

直观的想法是用户提供一组被认为是真的事实组成的集合和一些可能的假设，这些假设准备作为解释的一部分，以便预言所期望的观察（即连同事实一起蕴含这些观察）

是否与这些事实（即不预言任何被认为是假的东西）一致。这个解释应被看成基于可能假设限制集的一个"科学理论"。把解释看成是某目标为真的脚本也是很有用的。用户提供这些脚本中所承认的是什么。

这个框架是以一阶谓词演算形式给出的，尽管这个想法不依赖于所用的特定逻辑。我们需要一组可能的假设集，（与事实一起）蕴含目标而不是谬论（或者某种被认为是假的东西）。

我们假定给定一个可数字母集上的一阶语言。所谓公式我们是指这一语言中的合式公式。所谓公式的一个实例是指语言项对公式中自由变元的一次代换。

我们假设用户给出集合$\mathcal{F}$和Δ。$\mathcal{F}$是一个被作为"事实"的封闭公式集，并假设事实是一致的。在规定的解释中，这些事实作为真语句。更确切地说，这些事实是我们在某个应用中不准备放弃的句子。Δ是一个公式集，常被称作可能假设集。如果此集合一致的话，则它的任意基例被用作一个假设。

定义2.1 $\mathcal{F}$，Δ的一个脚本是一个集合$D\cup\mathcal{F}$，其中D是使得$D\cup\mathcal{F}$一致的Δ元素的基例集。

定义2.2 如果g是一个封闭公式，则g关于$\mathcal{F}$，Δ的解释是一个蕴含g的$\mathcal{F}$，Δ脚本。

即g是关于$\mathcal{F}$，Δ可解释的，是指存在一个Δ的元素的基例集D，使得满足（1）$\mathcal{F}\cup D\vdash g$且（2）$\mathcal{F}\cup D$是一致的。

第一个条件是说D预言g，而第二个条件是说D不预言任何我们认为是假的东西。

定义2.3 $\mathcal{F}$，Δ的扩充是一个关于$\mathcal{F}$，Δ的最大（相对于集包含）脚本的逻辑结论集。

这样一来，所引起的问题就是它是否是一种真实的语义。它肯定不比其它语义复杂；它将继承所有它关于一阶谓词演算方面的语义。

语义是语言中符号和句子与语义域的一种联系。我感兴趣的语义域是真实世界，它不是Herbrand全域、某种Kripke结构或某种其它数学结构的子集（尽管它也许与这些结构有某种关系），而是相当于由树与椅子和人与疾病组成的世界。我的机器人必须进入的就是这种世界，我的诊断程序为确定一个病人患何疾病而必须进行推理的也是这种世界。

古典（Tarskian）语义的乐趣是，如果在我的语言中选择了一种符号表示法（即对于我的语言中的每个符号我决定它在这个世界中表示（标记）一个个体或关系）并且以语句为真的公理形式记在这个世界中，那么我的公理的所有定理在这个世界中也是真的。

在这个世界中不存在缺席。"鸟会飞"不是一个为真的语句或者不属于一个世界；在鸟这个世界中，某些能飞，而某些却不能飞。缺席是用于建立我们的世界理论的假设。当我们采用这个理论（一种为脚本的逻辑结论集的理论）时，我们才能谈论语义和逻辑结论。正是这些脚本使语义与这个世界联系起来。

2.1 定义的性质

我们感兴趣的问题是脚本是否是有限的。注意，一个扩充既不是有限的，也没必要是有限公理化的（即，也许不存在一个有限集D，使得它的扩充是$\mathcal{F}\cup D$的逻辑结论）。

以下引理直接遵从一阶谓词演算的紧性定理。

引理2.4 如果g是$\mathcal{F}$，Δ可解释的，则它是使用一个有限脚本可解释的。

下面的引理普遍成立。

引理2.5 （单调性）如果$D_1\subseteq D_2$且$\mathcal{F}\cup D_1\vdash\alpha$则$\mathcal{F}\cup D_2\vdash\alpha$

在可解释性与存在一个扩充之间有非常密切的联系：

定理2.6 α是可解释的当且仅当α属于某扩充。

证明：如果α是可解释的，则存在一个脚本S，S蕴含α。S，Δ的任何一个扩充根据引理2.5是包含α的$\mathscr{F}$，Δ的一个扩充。α像S一样存在于一个扩充中。

相反地，假设α在某扩充中，对于某个脚本S有$\alpha \in Th(S)$ *)。因此，S是一致的且$S \vdash g$，所以S是α的一个解释。□

3. 缺席推理

这里我们将说明，如果这些可能假设是缺席，则以上语义给出了一个缺席推理的解释。

我们将把缺席看成能被用于一个解释的可能假设。可解释性相当于从缺席假设产生的结果。一个缺席是某人准备作为某物为什么被规定成真的解释的一部分。

当把可能假设考虑为缺席时，则所解释的事物意指我们预言的而不是我们实际观察到的某种事物。

例3.1. 考虑语句"鸟会飞"。它能被表示成缺席：

$\Delta = \{bird(x) \Rightarrow flies(x)\}$

这意味着，对于x的一个特殊值b，如果bird (b) 能被证明，则flies (b) 也如此，只要缺席不与特殊值b矛盾。如果 bird(b) 与¬flies (b) 是可证的，则缺席与特殊值b矛盾，因而不能使用。

如果与以上假设一起我们有事实：鸸鹋是不能飞的鸟，Polly是一个鸸鹋并且Tweety是一个鸟。这能被表示成：

$\mathscr{F} = \{\forall x\ emu(x) \Rightarrow bird(x),$
$\forall x\ emu(x) \Rightarrow \neg flies(x),$
$emu(polly),$
$bird(tweety)\}$

flies(tweety可由{bird(tweety)⇒flies (tweety) 解释，因为它与事实不矛盾。这个解释意味着：我们规定tweety会飞是因为tweety是鸟并且我们隐含地知道鸟会飞且没有理由相信tweety不会飞。flies(polly)潜在地可由{bird(pooly)⇒flies(polly)}解释，但是它却与事实不一致，因为它的否定是可证的。因此flies (polly) 是不可解释的。

*)如果A是一个公式集，则Th (A) 是A的逻辑结论的集合。

4. 与Reiter的逻辑的比较

我们与Reiter的缺席逻辑进行两种比较。这一节将说明以上逻辑是他的一般缺席理论的有限实例，而第8节则说明怎样不失简单的语义而得到更一般化的缺席逻辑的能力。

缺席$w \in \Delta$是Reiter的正规缺席 $\frac{:Mw}{w}$ 的一个语法变体。

对于闭缺席来说，Reiter的逻辑根据"扩充"来定义。这一节将说明按照最大脚本定义的一个扩充(定义2.3) 标记与按照不动点理论说明的Reiter的定义是一致的。

下面的定理对应于[26]中给出的闭扩充的定义。这说明我们按照最大理论所作扩充的定义与他的依照不动点理论的定义是一致的。

定理4.1 假设E是$\mathscr{F}$，Δ的一个扩充，则

(1) $\mathscr{F} \subseteq E$。

(2) $Th(E) = E$。

(3) 如果α是Δ的元素的一个基例并且$\neg\alpha \notin E$，则$\alpha \in E$。

进而E是服从以上三个性质的最小集。

证明：如果E是$\mathscr{F}$，Δ的一个扩充，则$E = Th(\mathscr{F} \cup D)$，其中D是Δ元素基例的某最大集(可能是无限的)。

因为对于任意的D，$\mathscr{F} \subseteq Th(\mathscr{F} \cup D)$，所以性质 (1) 成立。

又由于对于任意S，$Th(Th(s)) = Th(S)$，所以根据Th的这一性质，定理的性质 (2) 也成立。

为了证明性质(3)，设α是一个可能假设集的基例，并且$\neg\alpha \notin E$。如果$\alpha \notin E$，则$\mathscr{F} \cup D \cup \{\alpha\}$是一个有限的$\mathscr{F} \cup D$的严格超集且为一致的 (因为$\neg\alpha \notin Th(E)$)，这与E的极大性矛盾。

为了证明对于以上三个性质$E = th(\mathscr{F} \cup D)$是极小的，设$E' \subseteq E$，$E \not\subseteq E'$，并且$E'$是一个具有以上三个性质的集合，则必然有Δ的一个元素的某个基例α，使得$\alpha \in D$

且$\alpha\notin E'$（∵若D中的每个元素属于E'，我们有$\mathcal{F}\cup D\subseteq E'$且$E=Th(\mathcal{F}\supset D)\subseteq Th(E')=E'$）。$\alpha$是可能假设的一个基例且$\neg\alpha\notin E'$（由于$E'\subseteq E$且$\neg\alpha\notin E$（由于$\alpha\in E$且E是一致的））。但是$\alpha\notin E'$，与（3）矛盾，因此不存在这样的$E'$。□

5. 命名缺席

要做的一件好事是命名缺席的能力，即在一个缺席中用自由变量参数化名字。这样做有以下三点理由：

（1）它们使得打印解释更容易。

（2）当实现我们的系统时，我们只需关心名字并基本上能译出缺席的结构。我将证明把系统限制为非常简单的缺席形式不会失去任何功能。

（3）我们希望能清晰地讨论一个缺席以便我们能，例如，清晰地说什么时候它是不可用的。这对解决多扩充问题是有用的。

如果$w[x]$是一个含有自由变量$x=x_1,\cdots,x_n$的缺席，则我们能用$P_w(x_1,\cdots,x_n)$命名w，其中P_w是不在系统（即在$\mathcal{F}$和Δ中）其它地方出现的n元谓词符号。

在这一节，我们想说明怎样使用一个名字。必不可少地我们使Δ只包含缺席的名字，且使这个名字蕴含缺席作为一个事实。首先，我们想说明限制Δ到名字不减少功能，否则会改变系统的语义。我们然后讨论怎样用名字解决多扩充问题。命名的使用与McCarthy的非正态性谓词密切相关，而且翻译也确实相似于Grosof的命名的使用和Brewka的APPL谓词的使用。

$$\Delta'=\{p_w(x)s.th.w[x]\in\Delta\},$$

$$\mathcal{F}'=\mathcal{F}\cup\{\forall x p_w(x)\Rightarrow w[x]s.th.w[x]\in\Delta\}$$

$\mathcal{F}'$，Δ'是其中所有缺席被命名的系统，并且它们代替了Δ中的缺席。下面的定理说明，如果没有任何意外的副作用的话，这总是成立的。

定理5.1（换名原理）　如果谓词符号p_w不在$\mathcal{F}$，Δ或g中出现，则g是由$\mathcal{F}$和Δ可解释的当且仅当g是由$\mathcal{F}'$和Δ'可解释的。

证明：假设g是由$\mathcal{F}$和Δ可解释的，则存在某个解释$D=\{w[c]\}$，其中$w[x]\in\Delta$，使得$\mathcal{F}\cup D\vdash g$且$\mathcal{F}\cup D$是一致的。

设$D'=\{p_w(c)s.th.w[c]\in D\}$。为证明$\mathcal{F}'\cup D'\vdash g$，相反地，假设$\mathcal{F}'\cup D'\cup\{\neg g\}$在解释I中为真。于是$\mathcal{F}$和$\neg g$在I中为真。对于D中的每个$w[c]$，我们知道$p_w(c)$和$\forall x\,p_w(x)\Rightarrow w[x]$在I中为真，因此，$w[c]$在I中为真，这使得$\mathcal{F}\cup D\cup\{\neg g\}$的模型I与$\mathcal{F}\cup D\vdash g$矛盾。所以这样的模型不存在，因此，$\mathcal{F}'\cup D'\vdash g$。

为了证明$\mathcal{F}'\cup D'$是一致的，设I是$\mathcal{F}\cup D$的一个模型。设I'是一个除了当$w(c)$在I中为真时$p_w(c)$在I'中为真外与I一样的解释。那么I'是一个关于$\mathcal{F}'\cup D'$的模型，这是因为$\mathcal{F}$在I'中为真（因为$p_w(c)$不在$\mathcal{F}$中出现且$\mathcal{F}$在I中为真），由定义$p_w(x)\Rightarrow w[x]$在I'中为真，且因为D在I中为真，所以D'在I'中为真。

关于定理的充分性部分，设g是$\mathcal{F}'$和Δ'可解释的。于是存在某个D'使得$\mathcal{F}'\cup D'\vdash g$和$\mathcal{F}'\cup D'$是一致的。设$D=\{w[c]|s.th.p_w(c)\in D'\}$，根据以上两点，得$\mathcal{F}\cup D\vdash g$和$\mathcal{F}\cup D$是一致的。□

这说明我们能在命名的系统中和一般的系统中工作得一样好，且仅把原子当作缺席不会降低系统的功能。

6. 关于缺席的程序设计语言

到现在为止，我们已经给出了一个关于缺席推理的语义。在这一节，我们为缺席推理提出了一个程序设计语言（称为THEORIST），和一个程序设计方法，这样可解决不理想的多扩充问题。

下面定义并说明一个非常简单的语言*）：

—**fact** w，其中w是一个公式，意思是"$\forall w$"$\in\mathcal{F}$。

—***default*** n，其中n是一个名字（只把自由变量作为参数的谓词），意思为$n\in\Delta$。

—default n:w，其中w是一个公式，n是具有与w一样的自由变量的名字，意思是w是一个缺席，它的名字是n。其形式叙述是$n\in\Delta$且"$\forall n\Rightarrow w"\in\mathscr{F}$。

—explain g，其中g是一个公式，并且问我们是否可能从$\mathscr{F}$和Δ解释$\exists g$。一致的解释被返回。

在程序设计语言中，我们按照PROLOG的习惯，变量是大写字母，常量是小写字母，而无界变量蕴含为全称量化的。我们还用符号"←"表示实质蕴含，读作"如果"。

例6.1 第3节中的鸟会飞的例子可被表示成：

```
default birdsfly(X), flies(X)←
                              bird (X).
fact ¬flies(X)←emu(X).
fact bird(X)←emu(X).
fact bird(tweety).
fact emu (polly).
```

这里提问

```
explain flies(tweety).
```

将返回一个回答"yes"，设{birdsfly (tweety)}。注意这个解释给出了通常用来支持我们谓词的假设。它告诉我们进行谓词演算后的推理结果。我们预言tweety能飞是因为Tweety是一个鸟且能飞。

关于提问

```
explain flies (pobly).
```

将返回一个回答"no"（意思是它无法解释）。

6.1 程序设计方法和多扩充问题

在本系统中编写程序的思想是增加作为缺席的一般化知识，并且增加我们知道为真的知识为事实。我们还增加一些说明什么时候一个缺席不能被使用（用一个缺席的名字）的知识，它们如下：

如果我们有一个缺席公式w，并且我们知道，它在条件C下不能应用，我们能给系统

```
default  n:w
fact  ¬n←c
```

我们说我们能假设n并且得到w，但如果我们能证明¬w或c时，那么这是不一致的（因而不能被使用）。第二条规则与McCarthy的相约继承公理相对应。

例6.2 考虑下面语句：

```
default mammals-don't-fly(X): ¬flies(X)
        ←mammal (X).
default bats-fly(X): flies(X)←bat(X).
default dead-things-don't-fly(X): ¬fli-
        es (X)←dead(X).
fact mammal(X)←bat(X).
fact bat (dracula).
fact dead (dracula).
```

我们能由{mammals-don't-fly (dracula)解释¬flies (dracula)。即我们预言Dracula不能飞是因为Dracula是一个哺乳动物(mammal)，而哺乳动物，根据缺席是不能飞的。

我们能用{bats-fly (dracula)}解释flies (dracula)，即，因为dracula是一个蝙蝠(bat)，蝙蝠典型地能飞，所以我们能期望它会飞。

我们还能用{deadthings-don't-fly (dracula) 解释¬flies (dracula)，我们能预言dracula不能飞是由于它是一个死蝙蝠并且死蝙蝠根据缺席是不能飞的。

如果我们因不想第一个缺席用于蝙蝠而不喜欢第一种解释，则我们可以增加

```
fact¬mammals-don't-fly(X)←bat(X).
```

因此如果我们证明X是一个蝙蝠，则我们不能用缺席"因为X是一个哺乳动物并且哺乳动物不能飞，所以X不能飞"。

如果我们对死蝙蝠不想要第二种解释，则我们可以增加事实

```
fact¬bats-fly(X)←dead(X).
```

在最终的这个系统中，我们不能解释flies(dracula)而只能解释¬flies(dracula)，正确的理由是：由于它是死蝙蝠并且死蝙蝠典型地不能飞。

例6.3 这个例子来源于Reiter[37]，其中声明正常的缺席不适应于例外的规则。

```
default unemp-if-st(x): ¬employed(X)←
        uni-student(X).
```

*) $\forall w$是w的全称子句，即，如果w是具有自由变量v的公式，则$\forall w$的意思是$\forall vw$。同样地，$\exists w$是w的存在子句，即为$\exists vw$。

default emp-if-ad(X): employed(X)←adult(X).

default ad-if-st(X): adult(X)←uni-student(X).

fact uni_student(paul).

我们能用{emp-if-ad (paul), ad-if-st (paul)}解释employed(paul)。即我们能预言Paul能被雇用，因为他是个学生，又这根据了缺席"成年人"和缺席"被雇用"。

我们能用{unemp-if-st (paul)}解释¬employed(paul)。如果我们因为不想缺席emp-if-ad被用于大学生而不喜欢第一个解释，则我们可以增加事实

fact ¬emp-if-ad(X)←uni-student(X)

这使第一个解释不成立。

注意，我们这里的讨论是基于缺席而不是解释观察的。若是这样的话，在某些情况下，我们能解释出某些命题及其否定都是合理的。这意味着有两种推测的证据，因此，从语义角度不偏向任何一种。(然而，建立在启发式基础之上，例如，可能性理论，可能喜欢一种解释胜过其它解释)

例6.4 考虑Reiter[27]教友派信徒-共和党员的例子。

default quakers-are-pacifists(X): pacifist(X)←qnaker(X).

default republican-so-not-pacifist(X): ¬pacifist(X)←republican(X).

fact ¬quakers-are-pacifists(X)←republican(X).

fact ¬republican-so-not-pacifist(X)←quaker(X).

fact republican(ron).

fact quaker (george).

fact quaker (dick).

fact republican (dick).

这里，我们能用解释{quakers-are-pacifists (george)}来解释pacifist(george)。我们能用解释{republican-so-not-pacifist (ron)}解释¬pacifist (ron)。关于Disk's pacifism，我们不能给出任何解释

7. 脚本集合的裁剪

上一节给出了其中认为具有半正规缺席是必要的许多情况，并说明了在除正规缺席间单形式之外不增加任何内容下如何加以解决的方法。然而，也产生了一个问题。

例7.1 (Etherington[8])考虑能假设某人是嫌疑犯的问题。如果我们观察到他们没犯罪，那他们就不应该是嫌疑犯，然而我们不应仅仅因为他们是嫌疑犯而得到他们有罪的结论。在第一种情况下，对可能的脚本要做某种限制。即¬guilty(X)⇒¬suspect(X)。我们要排除某人没犯罪但仍是嫌疑犯的任何脚本。然而我们不能让这个限制作为事实，否则我们会因为某人是嫌疑犯而得到他是罪犯的结论。

看来，我们似乎需要一种裁剪脚本集合的办法，而不只是增加更多的事实。

当取消了缺席时，产生了一个类似的问题：

例7.2 设我们有下列THEORIST片断：

default birdsfly(X):flies(X)—bird(X).

fact ¬birdsfly(X)←emu(X).

这是说，我们能用birdsfly(X)推测某种鸟会飞，但如果我们能证明X是emu，它就不能应用。然而，由此可以得到其它预示。

由此我们可以对任何b假设birdsfly(b)。这预示着¬emu(b).因此对任何b我们能解释¬emu(b)。人们坚持认为这是合理的，因为如果我们假设了任何鸟会飞而它对emus又不起作用，我们就隐含地假设了任何鸟都不是emu。

用这种缺席我们能从¬flies(b)得到¬bird(b)。这也被坚持认为是合理的，正如同演绎时逆反式是有意义一样。

例7.3 假设我们想有缺席：如果X是人，那我们假设他们不是糖尿病人，这可以表示为

default not-diabetic-if-person(X): ¬diabetic(X)←person(X)。

这不仅可以用来从person(robin)解释¬diabetic(robin)而且可以从diabetic(john)解释¬person(john)。

对这里出现的情况似乎有两种观点。第一种是说这里没有任何错误。当你在假设如果某物是鸟它就会飞时，你隐含地假设它不是emu。对于第一个例子，"guilty"这个词有不正确的解释。在意向性解释中"if someone is not guilty, they are not a sus-

pect"是错误的。如果我们换一个意思"possibly guilty"，那就没问题了。例7.3的问题是右边的限制*person*(X)与缺席无关（如果它不是无关的，那么结论non diabetics are not human是合理的）。事实上，对于所有Reiter和Etherington工作中的例子只需使用前一节描述的命名转换实现。

第二种观点认为上面的结论是错的。THEORIST系统的用户们已经发现他们需要一种方法来描述"在这种情况下这个缺席不能用"而且这种描述不产生任何副作用。这节我们定义了人们能够这样做的限制表示法。应用中发现它们是一种很有用的机制。别的一些系统在避免这样的结论时采用了把缺席当作规则，而该规则只能单方向使用并且有明显的例外，或者采用了固定的/变化的谓词（这只部分解决了问题）。

这种思想是定义一个用来裁剪脚本集合的限制集，只用以抛弃一些情况说明而不用以解释任何事情。限制是使脚本保持一致的公式。

我们引入闭公式集C称为限制集*)。脚本的定义修正如下：

定义7.4 $\mathscr{F}$，C，Δ的一个脚本是一个集合D∪$\mathscr{F}$，其中D是使D∪$\mathscr{F}$∪C不一致的Δ的元素的基例集。

可解释的和扩充的定义也相应变化了。

下面是定理2.6的一个推论，证明是对2.6证明的释义。

推论7.5 具有限制的系统中α是可解释的当且仅当α在某个扩充中。

我们用constraint w描述扩充程序设计语言，其中w是某个公式（自由变量隐含地全称量化了），意义为"∀w"∈C。

通过给出constraint ¬d←c，使得如果不允许用¬c作解释的话，在条件c下，可以用限制避免缺席d被应用。这个限制所做的是抛弃任何蕴含d和c两者的脚本，而没有[illegible]它作用。

通过增加限制constraint ¬d←¬c，[illegible]们能阻止default d：c←b的逆反用法。（就是说，我们避免使用缺席d从¬c得到¬b）。如果我们知道c不真，我们不能用缺席d。这个限制的唯一效果是不允许蕴含¬c和d两者的脚本。

注意这里$\mathscr{F}$∪D仍是一阶语义，因此a←b和¬b←¬a是逻辑等价的。我们已经增加了限制来限制缺席可以被应用的环境。

例7.6 从

```
default  birdsfly(X) : flies(X) ←bird(X).
constraint  ¬birdsfly(X)←¬flies (X).
fact  bird (polly).
fact  ¬flies (bruce).
```

我们能解释flies (polly)，但不能解释¬bird (bruce) 若是限制不存在的话，则我们可以，或者不能解释flies (david) （若限制是一个事实的话，则我们可以）。

例7.7 考虑例6.2，通过如下规定，我们能把它变为不允许逆反或某物不是蝙蝠或没死的这类结论，

```
default  mammals-don't-fly (X) : ¬flies
         (X) ←mammal(X).
constraint  ¬mammals-don't-fly (X) ←
        flies (X) .
default  bats-fly(X) : flies(X)←bat(X).
constraint  ¬bats-fly(X)←flies(X).
constraint  ¬mammals-don't-fly(X)←bat
         (X).
default  dead-things-don't-fly(X):¬flies
         (X)←dead (X) .
constraint  ¬dead-things-don't-fly (X)←
         flies(X).
constraint  ¬bats-fly(X)←dead (X) .
fact  mammal(X)←bat (X) .
fact  mammal(bruce).
fact  bat (paul).
fact  bat (dracula).
fact  dead(dracula).
```

*)这个概念连同Gagné(9)一起被发展了。

这里我们能得到结论Bruce不会飞，Paul会飞，Dracula不会飞。对于其中每一种，我们不会解释出它的否定。我们也声明：它没有任何意外的副作用。

8. Reiter的一般缺席

第4节我们说明了我们的缺席是如何视为一个受限的Reiter的正规缺席的。这一节我们要说明Reiter一般缺席的额外表达能力是不需要的。

Reiter的一般缺席具有形式

$$\frac{\alpha(x):\ M\beta_1(x),\ \cdots,\ M\beta_n(x)}{\gamma(x)}$$

（其中x是自由变量的集合）。这个公式意味着如果α(c)被证明了，而且$\beta_i(c)$是一致的，那么γ(c)能被推出。

在THEORIST中通过对每个β_i建立名字M_{β_i}并且对我们不能证明$\neg\beta_i(x)$的相应实例我们能假设其任何实例，建立关系$M_{\beta_i}(x)$，这个公式可以被模拟。这种模拟是对于每个i，有

default $M_{\beta_i}(x)$.

constraint $\neg M_{\beta_i}(x) \leftarrow \neg\beta_i(x)$.

和事实

fact $\gamma(x) \leftarrow M_{\beta_1}(x) \wedge \cdots \wedge M_{\beta_n}(x) \wedge \alpha(x)$.

如果缺席是半正规的话，这很接近[26]。例如，缺席

$$\frac{\alpha(x):\ M\beta(x) \wedge \gamma(x)}{\gamma(x)}$$

可以使用以上建立名字$M_{\beta\gamma}$的技术把它近似地定义成：

default $M_{\beta\gamma}(x):\ \gamma(x) \leftarrow \alpha(x)$.

constraint $\neg M_{\beta\gamma}(x) \leftarrow \neg\beta(x)$.

constraint $\neg M_{\beta\gamma}(x) \leftarrow \neg\gamma(x)$.

第一个限制是说，对于我们能证明$\neg\beta$的缺席，我们不能用它的任何实例。第二个阻止对缺席的逆反用法。

这个翻译还不精确。下面是一个得出不同答案的例子。我将说明，它们的一个差异在于Reiter缺席给出的结果不直观。

例8.1 考虑我们有缺席emus和ostriches都会跑，并且我们知道polly要么是一个emu，要么是一个ostrich（它们看上去很像）的例子。用Reiter的标记方式，这可以表示为，

emu(polly)∨ostrich(polly),

$$\frac{\text{emu}(X):\ M\text{runs}(X)}{\text{runs}(X)}$$

$$\frac{\text{ostrich}(X):\ M\text{runs}(X)}{\text{runs}(X)}$$

在Reiter的系统中我们不能得到runs(polly)，因为我们不能检验任何一个缺席的前件。如果我们考虑THEORIST的翻译，我们有

fact emu(polly) ∨ostrich(polly).

default emu-run(X), runs(X) ←emu(X).

constraint ¬emu-run(X)←¬runs(X).

default ostriches-run(X), runs(X) ← ostrich(X).

constraint ¬ostriches-run(X)←¬runs(X).

由此我们能用一致的{emus-run(polly), ostriches-run(polly)}（尽管我们有emus和ostriches是相交类）解释runs(polly)。

注意，在这个例子中重要的不是限制的运用，而是Reiter缺席的前提条件的特殊状态。

同样要注意，域表达方式的微小变化会使Reiter系统给出不同答案。例如，如果有big birds with short feathers这样一类，它覆盖了emus和ostriches，通过缺席，它们会跑，那么Reiter系统能解释出polly会跑。

第二点差异是我们没有半正规缺席的无扩充问题。Etherington给出下列例子：

$$\frac{:A \wedge \neg B}{A},\quad \frac{:B \wedge \neg C}{B},\quad \frac{:C \wedge \neg A}{C}.$$

根据一般缺席的上述翻译方法，它被翻译为*)

default manb:a.

dafault mbnc:b.

default mcna:c.

*) 我们不需要¬manb←¬a限制，因为我们已有事实a←manb。

constraint —manb←b.
constraint —mbnc←c,
constraint —mcna←a.

这里我们有三种扩充，每一种对应假设一个缺席的情况。例如，我们可以假设manb,使我们能预示a。第一个限制是说，我们已隐含假设了¬b。半正规缺席的无扩充问题的产生是因为它们不允许记录已经隐含地假设了什么。

定理8.2 （缺席的单调性）增加缺席只能增加可解释事件的数目，增加限制只能减少可解释事件的数目。

证明：假设从$\mathcal{F}$，Δ，C可解释g，那么存在某个D，一个Δ元素的实例集，使$\mathcal{F}\cup D\vdash g$并且$\mathcal{F}\cup D\cup C$是一致的。如果$\Delta\subseteq\Delta'$，那么用同样的D，从$\mathcal{F}$，Δ′，C可解释g。如果$C'\subseteq C$，那么从$\mathcal{F}$，Δ，C′可解释g，因为$\mathcal{F}\cup D\vdash g$并且$\mathcal{F}\cup D\cup C'$是一致的。□

推论8.3 如果$\mathcal{F}\cup C$一致，总存在一个扩充。

我们能保证这点是一个有用情形，是指

(1) 显式的事实（那些用事实说明加进的事实）是一致的，

(2) 所有的限制具有¬d←c形式，这里d是缺席名字，并且

(3) 在显式事实中没有缺席名字出现。这意味着在$\mathcal{F}$中仅有的缺席名字只存在于缺席说明中。

这是因为显式事实和所有缺席名字为假的模型的解释是一个$\mathcal{F}\cup C$的模型。

Reiter缺席的翻译遵循这个规定，因此如果事实一致，总有一个扩充。

定理8.2具有除仅保证我们有扩充之外的结论，考虑以下例子：

例8.4 假设我们有一个系统具有缺席

$$\frac{A:B}{B},\quad \frac{A:C}{C},\quad \frac{C:\neg B\wedge D}{D}.$$

根据Reiter[26]语义，有一个包含B和C而没有D的扩充。第一个缺席迫使第三个缺席不能使用。

在我们的翻译中，我们有两种扩充，一是包含B和C，二是包含C和D。第二种扩充是运用第三个缺席隐含假设B为假，而使第一个缺席不能用。

8.1 继承网络

我们对于半正规缺席的翻译的另一个有趣特点在于有例外的继承网络的翻译。我们可以遵循Etherington的翻译方法，而把上面定义的翻译从半正规缺席翻译为THEORIST*)。这通过缺席的命名完成。于是从A到B的缺席IS-A变为说明

default b-if-a(X): b(X)←a(X).
constraint ¬b-if-a(X)←¬b(X).

类似地缺席ISN'T-A有翻译

default not-b-if-a(X): ¬b(X)←a(X).
constraint ¬not-b-if-a(X)←b(X).

例外链翻译为限制。从结点c到表示命名为缺席d（在上面的缺席中给的名字）的弧的弧变为constraint ¬d←c.

这里，我们使继承层次中每个弧的模块化语句变为缺席系统中数量不多的规则。不存在与其它弧有关的翻译规则。这种翻译因而避免了Touretzky对Etherington的翻译的驳斥。

9.实 现

THEORIST的实现很简单，它与Reiter[26]的自顶向下的缺席证明紧密相关，并且在其它地方[22]做了描述。

为求g的解释，我们试图用事实$\mathcal{F}$和作为公理的可能假设Δ来证明g。我们使D_0为证明中所用的Δ的元素的实例集，使D_1为D_0的基（用唯一的常量替换所有自由变元）。我们然后知道$\mathcal{F}\cup D_1\vdash g$。我们可以用一个完全定理证明器通过不能证明$\mathcal{F}\cup C\cup D_1$不一致来证明它是一致的。一般说，这是不可判定的，但这不是我们所讨论的领域中的问题。

目前THEORIST有一个基于用PROLOG

（转第56页）

*) 我们假设读者熟悉[7,P56]中的那种翻译，我们只用大写字母表示变量的不同规定。

```
CALCULATE()
  /* transforms numericaldata, i.e., scenario, as described in the calculation models */

ext  NUMERICALDATA: wr Scenario
     OPERATIONCONTROL: wr Script
pre  operationcontrol(CALCULATE) = TRUE
post numericaldata' = modelrun(numericaldata) &
     operationcontrol(CALCULATE) = FALSE
```

图9 后模型计算数值数据的值

```
/* This simply requires consistency of data in IDSS and rules of expert system. In other words,
the data in IDSS would not change if expert system were applied. That is, the data is the fixed
set against operation of expert system. */

inv-state = (∀n)
    (≤topinfer(convertsheet(interfacesheet,r),n)=convertsheet(interfacesheet,r)
```

图10 状态不变性

符号化知识推理的结果。电子表格用作一种工作台，而专家系统不仅可用于符号推理，也能缩小可选择的搜索空间。因此，用户在电子表格上的操作由专家系统监控并且按自调节反馈机理加以简单实现。

今后的问题包括：(具体问题的)财政计划系统的开发（作为IDSS生成器问题）。

·实现DSS数据库、电子表格、专家系统和图形库的更一致的数据结构。

·建造具有高功能的专家系统（现已有了Prolog可用）。

·开发新的接口特性，一般实现各种层次的自调节反馈机理。

参考文献（略）

〔**李海泉**译自《ICSSE'88》**杨 俊，仲 源**校〕

（接第65页）

写的完全定理证明器的解释器，还有一个把THEORIST的事实、限制和缺席变换为PROLOG的编译器。现在我们正在构作一个基于真值维护和依赖制导回溯的系统。我们也在构作一个能做并发一致性检查的系统。

在理论中通过要求我们只使用缺席的基例，避免了缺席中由存在和全称量化变量引起的问题。

10．结论及今后的工作

本文的主题是，缺席推理问题不是一个逻辑问题，而是如何运用逻辑的问题。这种方法给了我们一种探讨其它问题的巧妙办法。这就是说，出于某种目的，决定什么时候一个脚本"好"于另一个脚本。Poole认为，欲最佳地解决网络的继承问题得最好选择最具体的理论，即当具体知识和更一般知识可用时，我们最好使用最具体的知识。Goebel提出了公理化框架公理中的多扩充问题是如何在THEORIST框架中通过选择时序最大持续理论加以解决的。对于诊断系统，我们最好选择最合理的解释。

这一理论奠定了THEORIST系统的基础，已被用于不同领域，包括诊断、建立用户模型、类比推理的问题求解、代词辨析和学生的学习障碍诊断。

参考文献（略）

〔**付志明 马 蔚**译自《Artificial Intelligence》36 (1988) P.27—47，**刘毅之仲源**校〕