利用不动点求解子句逻辑推演的Petri网模型*

林 闯[1] 吴建平[2]

[1](国家信息中心经济与技术研究所 北京 100045)
[2](清华大学计算机科学与技术系 北京 100084)

摘要 文章研究了子句逻辑推演的Petri网模型表示和不动点求解方法.基于四值逻辑和冲突变迁的概念,可用Horn子句的Petri网模型方法来构造非Horn子句的Petri网模型.逻辑推演的基本方法之一就是寻找逻辑赋值的不动点.该文显示了一种基于Petri网模型的子句逻辑不动点求解算法,比现有算法更为有效.

关键词 逻辑推演,子句,Petri网,不动点,四值逻辑.

中图法分类号 TP301

逻辑推演是人工智能的基础之一,推演过程就是确定一个给定的命题是否由所收集的一组事实和子句规则所蕴含.在推理研究中,已有多种模型方法来表示知识和推理过程.Petri网之所以被选来模拟逻辑推演,不但是因为Petri网具有很好的模型描述静态和动态特性——并发、并行和冲突,而且也因为Petri网有很好的数学分析技术,可以给出问题求解的算法.将逻辑推演问题转换成Petri网模型,并用Petri网特有的分析方法去处理逻辑推演问题,可以增强月不同和有效的方法处理这类问题的机会.

Petri网已被用来描述Horn子句的逻辑推演[1,2]以及一组子句的不一致性的检查[3].Murata等人[4]已经给出了从一组Horn子句转换成一个Petri网模型的算法过程,而且建立了一组Horn子句不一致的必要条件和充分条件.他们给出了Horn子句的逻辑推演Petri网模型的不变量求解算法[1].

在文献[5]中,Petri网用于非Horn子句的逻辑推演模型,但是没有提供有效的推演算法,对负命题的表示也不确切.在本文中,我们要扩充Horn子句Petri网模型的表示方法,使之适应非Horn子句模型的表示.基于我们的非单调逻辑推演模型的工作[6],可以引入四值逻辑,给标记分配4种不同的颜色以表达命题的不同赋值.引入冲突变迁的概念,用以解决析取子句(异或表达式)的模型表示.不动点的概念在逻辑推演的研究中有重要的作用,逻辑推演的过程可以看做是不动点的形成过程.不动点可以用在逻辑编程理论的表示语义方法中.在Petri网逻辑模型中,不动点可由某个标识来表示,求解不动点就变成求解标识的最大赋值.

1 定义和术语:网、子句和真值

这一节将引入Petri网、子句和四值逻辑的相关概念和术语.

1.1 Petri网

我们仅非形式地描述Petri网的一般相关的概念,有关Petri网的定义和术语可参见文献[7].

Petri网由位置和变迁两类结点构成.这两类结点之间由有向弧连结,弧标权重在逻辑模型中为"1",标记在位置中的分布形成标识.一个变迁在其输入位置中都至少保留有一个标记的情况下,可能实施.变迁的实施,使其输入位置中都减少一个标记,且使其输出位置中都增加一个标记.

变迁的实施导致标识的变化,由一个标识达到一个后继标识.标识的这种变化也可由网的关联矩阵C来表

* 本文研究得到国家自然科学基金资助.作者林闯,1948年生,博士,研究员,主要研究领域为系统性能评价,随机Petri网,计算机网络,逻辑推理.吴建平,1953年生,博士,教授,博士生导师,主要研究领域为计算机网络与协议测试,Internet.

本文通讯联系人:林闯,北京 100045,北京三里河路58号国家信息中心经济与技术研究所

本文1997-04-14收到原稿,1998-03-16收到修改稿

达.关联矩阵C又可分别由输入矩阵I和输出矩阵O来表示.关联矩阵的列元素由网的每个位置构成,行元素由网的每个变迁构成.在矩阵I中,当位置P_j到变迁T_i有弧相连结,则I_{ij}元素为弧标,否则为零.在矩阵O中,当变迁T_i到位置P_j有弧相连结,则O_{ij}元素为弧标,否则为零.关联矩阵$C=O-I$.用C_t表达关联矩阵与变迁t相关的行,则一个变迁t在标识M下实施达到后继标识M'可表达为$M'=M+C_t$,或者$M'=M+O_t-I_t$.在Petri网图形表示中,我们用线段表示变迁,用圆表示位置,位置中的黑点表示标记(token).

1.2 子 句

在一阶逻辑推演研究中,一个子句定义(wff)有如下形式:

$$B_1,\ldots,B_q \vdash A_1,\ldots,A_p, \quad 或 \quad A_1\wedge\ldots\wedge A_p\rightarrow B_1\vee\ldots\vee B_q,$$

其中A_i和B_i都是原子公式,$\vdash$或$\rightarrow$表示蕴含,$p\geqslant 0$,$q\geqslant 0$.这个子句意味着,如果A_1和...和A_p都是真,那么,B_1或...或B_p为真.

一个Horn子句要求:$q=1$或$q=0$且$A_i(1\leqslant i\leqslant p)$和$B_1$(如果有)是非负公式.在一般子句中没有这两点限制.

逻辑推演就是确定一个给定的公式是否由一组子句所蕴含.逻辑推演的证明过程一般可以有两种方法:

(1) 推测一组子句集L是否蕴含Q就检测L和$\neg Q$的一致性,如果不一致,就证明L蕴含Q;否则,L就不蕴含Q(此种方法的应用已在另文中详细论述).

(2) 使用语义函数方法,将逻辑推演看做从初始公式赋值状态出发,通过子句规则不断赋值的过程,亦即,知识不断增加的过程,最后达到一个赋值不动点,不动点中包含着所有蕴含的信息.

1.3 四值逻辑

我们采用Sandewell对四值逻辑进行描述的记号和术语[6].假定一个4种真值的集合$J=\{u,t,f,k\}$,且具有偏序关系$\leqslant$,如图1所示.

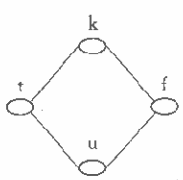


图1　四值真值组成的格

我们可有两个集合:一个语言集L,它的元素是公式;一个真值集J.赋值$V:L\rightarrow J$是一个连续单调增加的函数,它也有偏序关系$\leqslant$,定义如下:

$$V\leqslant V' \text{ iff } (\forall x)V(x)\leqslant V'(x).$$

一个赋值可以表示为知识或信念的一个状态,在这个状态中,命题可能是真(t)、假(f)、不知道(u)或矛盾(k).状态(赋值)之间的偏序关系表明状态之间包含信息的多少.如果$V\leqslant V'$就表示$(\exists x)V(x)=u$且$(V'(x)=t)\vee(V'(x)=f)$以致满足于$(\forall x)V(x)\leqslant V'(x)$,亦即,一些不知道的命题后来变为知道(t或f).

在逻辑推演过程中,几个知识状态的合并对应着4种真值格的最小上界操作.最小上界操作意味着:一个命题被某个规则赋值为u,而被其他规则赋值为t(或f).那么,这个命题就被赋值为t(或f),亦即,操作的目的是取直接上界值.由于本文不讨论一组子句存在矛盾的情况,所以在Petri网逻辑模型中仅考虑标记到$\{u,t,f\}$3种真值的映射.

2 子句Petri网模型

文献[2,4]已经给出如何将一组Horn子句转换成Petri网模型的算法.文献[3,5]也给出了基于四值逻辑的Petri网模型.现在给出一般子句转换成Petri网模型的规则.

子句的表达形式为

$$A_1\wedge\ldots\wedge A_p\rightarrow B_1\vee\ldots\vee B_q. \tag{1}$$

将公式(1)的左部称为前提,右部称为结论.当$p=0$时,右部表现为事实;当$q=0$时,左部表现为求证蕴含的公式.

我们分两种情况讨论子句向Petri网的转换.

(1) 当$q\leqslant 1$时,使用下列规则进行子句向Petri网模型的转换:

• 一个变迁表示一个子句;事实被表示为源变迁(即没有输入位置的变迁);

· 一个不同的命题使用一个不同的位置来表示;

· 前提命题表示为变迁的输入位置,结论命题表示为变迁的输出位置;

· 与正命题位置相关联的弧,标注为"t";与负命题位置相关联的弧,标注为"f".

例 1:给定一组子句 L:

① $\neg A$　　② $\neg B$　　③ $\neg A \wedge \neg B \rightarrow C$

④ $C \wedge \neg B \rightarrow D$　　⑤ $D \rightarrow \neg A$　　⑥ $D \rightarrow C$

我们可得 L 的 Petri 网模型和输出矩阵,如图 2 所示.

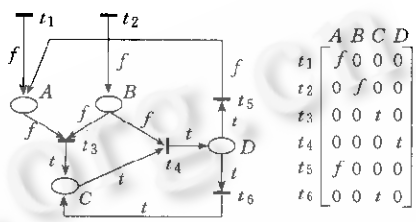


图2　例1子句所对应的Petri网模型和输出矩阵

在子句所对应 Petri 网模型中,由于其弧标引入了 t 和 f 以表示相应颜色标记的流动,隐含着标记有了真值颜色.在位置没有设置标记时,就假定位置包含着颜色为 u 的标记.变迁实施的条件要求输入位置包含与弧标相应值的标记,变迁实施后产生与输出弧标相应值的标记.在假定不存在矛盾的情况下,实施结果同输出位置中的原有标记无关.

(2) 在 $q>1$ 时,即子句有多个析取结论命题时,每个析取操作可能有 3 种语义表达方式.这里用一个例子加以说明,有一个子句:

$A \wedge B \rightarrow C \vee D$.

它可能包括 3 种表达:

(a) $A \wedge B \rightarrow D/C$(同或);

(b) $A \wedge B \rightarrow D \wedge C$(与);

(c) $A \wedge B \rightarrow D$ or C(异或).

在前两种表达式中,我们的系统中的每一个子句可以看做由两个子句组成,分别是 $A \wedge B \rightarrow C$ 和 $A \wedge B \rightarrow D$,它们可由前面所述的方法转换成 Petri 网模型.对于后一种情况,两个结论 C 和 D 不能同时成立,它们处于冲突中,可以引入一个判定位置和两个冲突变迁表示这个子句的两种异或情况,如图 3 表示.

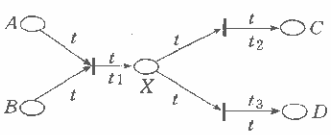


图3　一个多结论子句的Petri网模型

在图 3 中,位置 X 为逻辑判定位置,它的输出变迁为逻辑冲突变迁,t_2 和 t_3 永远处于逻辑冲突中,亦即,一旦一个变迁被选择实施,处于逻辑冲突中的其他变迁永不再被实施.这种定义的逻辑冲突变迁与一般 Petri 网冲突变迁的定义不同.一般 Petri 网模型中,冲突与标记存在情况相关,冲突变迁不永远冲突.为了表示这种冲突特性,当模型中包含相互逻辑冲突变迁时,我们可将其分解成多个不含相互逻辑冲突变迁的模型,也就是,在每个模型中的每一组冲突变迁中仅选择一个变迁,删除其余变迁以及连接弧.每个模型的逻辑推演结果为异或关系.

因此,在 $q>1$ 时,只要子句的每个析取结论之间的关系是确定的,我们就可以通过子句的分析和引入逻辑判定位置及逻辑冲突变迁来完成子句向 Petri 网模型的转换.

例 2:给定一组子句 L:

① A　　② B　　③ $A \wedge B \rightarrow \neg C$ or D

④ $D \rightarrow E \wedge F$　　⑤ $\neg C \rightarrow G$

按照上述转换方法,子句集 L 可以写成子句集 L':

① A　　② B　　③ $A \wedge B \rightarrow X$　　④ $X \rightarrow \neg C$

⑤ $X \rightarrow D$　　⑥ $D \rightarrow E$　　⑦ $D \rightarrow F$　　⑧ $\neg C \rightarrow G$

其中 X 是引入的逻辑判定命题.

L' 相应的 Petri 网模型可分解成不含相互冲突变迁的两个 Petri 网模型,如图 4 所示.

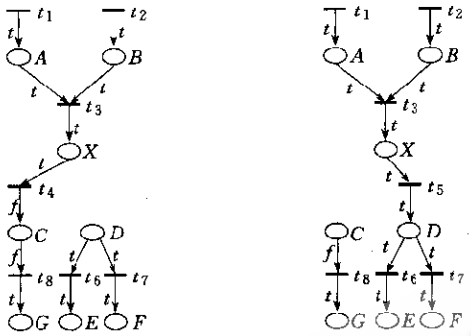


图4　例2子句集的Petri网模型

3　子句模型逻辑推演的不动点计算

在子句的 Petri 网模型中，信息状态对应着网的一个标识，一次赋值可以和变迁的一次实施对应，一个推演关系对应着网标识之间的一种可达关系. 由于在逻辑推演中，命题的赋值只单调增加；而在 Petri 网中，位置的标记既可增加又可减少，因此，在我们的子句 Petri 网模型中，定义标识之间的可达关系为

$$M[t_i>M' \rightarrow M'=\mathrm{Max}(M,O^{t_i})]. \tag{2}$$

实际上，公式(2)所描述的直接可达关系属于一般 Petri 网定义中的直接可达关系. 因为子句 Petri 网模型初始标识为 0，标记是由事实变迁生成的. 一位置中的标记得而复失，可由事实变迁以及其他变迁的实施而失去再来. 将间接可达压缩为直接可达，可以简化标识不动点的计算.

下面分两种情况来讨论子句 Petri 网模型的可达树构造算法，亦即，逻辑推演不动点的计算方法.

(1) 当模型中不包含相互逻辑冲突变迁时，可以有如下算法.

算法 1.

输入：一个没有相互逻辑冲突变迁的子句 Petri 网模型.

输出：模型可达标识的不动点.

```
Begin
  初始 Newnode 为 0 向量{每个分量都为 u};
  Repeat
    Currentnode:=Newnode;
    For 每一个能在 Currentnode 下实施的 t_i;
    Begin
        实施 t_i 且产生直接后继标识 M_i;
        Newnode:=Max(Newnode,M_i);
    End;
  Until(Currentnode=Newnode);
  return(Currentnode);
End.
```

让我们举一个例子来显示上述算法.

例 3：对于图 2 中的 Petri 网模型，其可达标识的不动点计算过程如下：

$$\begin{array}{l} \quad A\ \ B\ \ C\ \ D \\ M_0=[0,\ 0,\ 0,\ 0]-[u,\ u,\ u,\ u] \\ \qquad\qquad\qquad\qquad \downarrow t_1, t_2 \\ \qquad\qquad\qquad\quad [f,\ f,\ u,\ u] \end{array}$$

$\downarrow t_1, t_2, t_3$

$[f, f, t, u]$

$\downarrow t_1, t_2, t_3, t_4$

$[f, f, t, t]$

$\downarrow t_1, t_2, t_3, t_4, t_5, t_6$

$[f, f, t, t]$

最后可达标识即为模型标识的不动点.

定理1. 算法1求得的模型最后可达标识,即为模型对应子句集 L 的逻辑推演的不动点.

证明:因为 $M[t_i>M' \rightarrow M'=\mathrm{Max}(M,O^i)$,又因为 $M[t_i>M'$ 且 $M[t_j>M'' \rightarrow Newnode=\mathrm{Max}(M',M'')$,所以,$M$ 与 $Newnode$ 的可达关系符合逻辑推演关系.

由于 $M_0=0$ 且事实变迁都是源变迁,因此事实状态可达.

在算法1中,$Newnode$ 的值在所有可能发生变迁实施下不能再增加,即 $Currentnode=Newnode$,亦即表明,$Currentnode$ 为逻辑推演的不动点. □

现对算法1的复杂性进行分析. 假定 Petri 网模型有 n 个变迁和 m 个位置,那么,每一个标识最小上界的计算量为 m,在每个标识下最多可能有 n 个变迁实施. 可达集最多包含 n 个标识,因此,算法1的计算复杂性为 $O(n^2*m)$.

为了进一步提高算法的效率,我们可以改进算法1. 当子句集合本身存在矛盾时,命题一经赋值就不再改变. 换句话说,在我们的子句 Petri 网模型的推演中,一个变迁的一次实施就固定了它输出位置命题的值,在以后的推演中不再改变. 因此,在固定点的求解中,每一变迁实施后,就可从变迁集 T 中删除.

算法2.

输入:一个没有相互逻辑冲突和矛盾的子句 Petri 网模型.

输出:模型可达标识的不动点.

```
Begin
  初始 Newnode 为0向量{每个分量为 u};
  Repeat
    Currentnode:=Newnode;
    For 每一个能在 Currentnode 下实施的 t_i∈T;
    Begin
        实施 t_i 且产生直接后继标识 M_i;
        T:=T-t_i;
        Netwnode:=Max(Newmode,M_i);
    End;
  Until(Currentnode=Newnode);
  Return(Currentnode);
End.
```

算法2的复杂性可降低为 $O(n*m)$,与 Onaga 等人的子标识可达算法(其算法复杂性为 $O(m*2^n)$ 或 $O(m^2*n)$)[9]相比,获得的信息更多,算法更为简单.

例3的计算过程可简化为:

$A\ B\ C\ D$

$M_0=[0, 0, 0, 0]=[u, u, u, u]$

$\downarrow t_1, t_2$

$[f, f, u, u]$

$\downarrow t_3$

$$[f, f, t, u]$$
$$\downarrow t_4$$
$$[f, f, t, t]$$
$$\downarrow t_5, t_6$$
$$[f, f, t, t]$$

(2) 当模型中包含相互逻辑冲突变迁时，我们可将模型分解成多个不含相互逻辑冲突变迁的模型，然后再使用算法1或算法2计算模型标识的不动点.

例4：图4包括了两个分解后的 Petri 网模型，它们的不动点计算结果如下：

$$\begin{array}{c} A\ B\ X\ C\ D\ E\ F\ G \\ M_0=[u, u, u, u, u, u, u, u] \\ \downarrow t_1, t_2 \\ [t, t, u, u, u, u, u, u] \\ \downarrow t_3 \\ [t, t, t, u, u, u, u, u] \\ \downarrow t_4 \\ [t, t, t, f, u, u, u, u] \\ \downarrow t_8 \\ [t, t, t, f, u, u, u, t] \end{array}$$

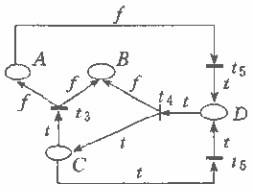
(a)

$$\begin{array}{c} A\ B\ X\ C\ D\ E\ F\ G \\ M_0=[u, u, u, u, u, u, u, u] \\ \downarrow t_1, t_2 \\ [t, t, u, u, u, u, u, u] \\ \downarrow t_3 \\ [t, t, t, u, u, u, u, u] \\ \downarrow t_6 \\ [t, t, t, f, u, u, u, u] \\ \downarrow t_6, t_7 \\ [t, t, t, u, t, t, t, u] \end{array}$$

(b)

在以上求解不动点的过程中，我们采用的是向前推演算法，从初始标识 $M_0=0$ 出发，推出标识的不动点. 我们也可以颠倒 Petri 网模型所有弧的方向，给求证的目标命题位置设置标记值，然后采用相同的算法进行向后推演模型的不动点计算. 当求得的不动点有关事实命题的赋值与已知事实的真值有偏序关系且小于等于已知值时，目标得证，否则，目标不成立.

例5：图2的子句 Petri 网模型的向后推演模型及不动点计算过程可如图5所示.



$$\begin{array}{c} A\ B\ C\ D \\ M_0=[u, u, u, t] \\ \downarrow t_4 \\ [u, f, t, t] \\ \downarrow t_3, t_5 \\ [f, f, t, t] \\ \downarrow t_5 \\ [f, f, t, t] \end{array}$$

图5

获得的不动点中的 A 和 B 命题赋值与已知事实值相同，因此结论 D 得证.

在一般情况下，向后推演比向前推演更有效. 在推理系统比较巨大和复杂时，亦即事实和推理规则比较多时，这种差异就更明显. 究其原因，在向前推演中，事前我们并不知道哪些事实和规则与问题有关，只能从大范围的事实出发进行推演，势必造成冗余计算；而在向后推演中，可以从相关的目标出发进行推演，无关的推演规则和事实不必索引，这样可以减少不必要的计算. 换句话说，向前推演盲目性大，向后推演目的性强.

4 结 论

本文展示了子句 Petri 网模型表示逻辑推演不动点的两种推演执行方法：向前推演和向后推演以及它们的算法和例子. 本文是文献[1]Horn 子句 Petri 网模型逻辑推演工作的继续，其主要贡献是：(1) 提供了子句 Petri 网模型逻辑推演的不动点方法；(2) 提供了不动点求解的有效算法；(3) 给出了向前推演和向后推演的模型表示和推演方法. 本文的研究成果对一阶谓词逻辑的推演也是适用的，限于篇幅，这里不再加以描述.

参考文献

1 Lin C, Chandhury A, Whinston A B *et al*. Logical inference of Horn clauses in Petri net models. IEEE Transactions on Knowledge and Data Engineering, 1993,5(3):416～425

2 林闯. Petri 网用于 Horn 子句的逻辑推演. 软件学报,1993,4(4):32～37
(Lin Chuang. Application of Petri nets to logical inference of Horn clauses. Journal of Software, 1993,4(4):32～37)

3 Murata T, Subrahmanian V S, Wakayama T. A Petri net model for reasoning in the presence of inconsistency. IEEE

Transactions on Knowledge and Data Engineering, 1991,3(3):281～292
4 Peterka G, Murata T. Proof procedure and answer extraction in Petri net model of logic programs. IEEE Transactions on Software Engineering, 1989,15(2):209～217
5 Chardhury A, Marinescu D C, Whinston A B. Net-based computational models of knowledge processing systems. IEEE Expert, April 1993. 79～86
6 Lin C, Murata T. A Petri net model for nonmonotonic reasoning based on annofafed logic programs. EIECE Transactions on Fundamentals, 1994,E77-A(10):1579～1587
7 Murata T. Petri nets: proPetries, analysis and applications. Proceedings of the IEFE, 1989,77(4):541～580
8 Sandewell E. A functional approach to non-monotonic logic. Computer Intelligence, 1985,(1):80～87
9 Watanabe T, Mizobata Y, Onata K. A Petri net-based algorithm for proofs in Horn clause prepositional logic. In: Proceedings of the ISCAS'90. May 1990

Logical Inference of Clauses in Petri Net Models Using Fixpoint

LIN Chuang[1]　WU Jian-ping[2]

[1](*Information Science Institute　State Information Center　Beijing* 100045)
[2](*Department of Computer Science and Technology　Tsinghua University　Beijing* 100084)

Abstract　The Petri net models of the clauses for logical inference using fixpoint are studied and the four-valued logic and the conflict transition concept are introduced. The authors can construct the Petri net models of non-Horn clauses based on the models of Horn clauses. Finding the fixpoint of logical values is one of the fundamental methods for logical inference. In this paper, an algorithm for the fixpoint based on the Petri net models of values is presented, and it is more efficient than the previous ones.

Key words　Logical inference, clauses, Petri net, fixpoint, four-valued logic.