ICOT Technical Memorandom: TM-1227

TM-1227

ノンホーン・マジックセット

太田 好彦、井上 克巳
長谷川 隆三、中島 誠 (JIPDEC)

October, 1992

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03)3456-3191～5
Telex ICOT J32964
Institute for New Generation Computer Technology

ノンホーン・マジックセット

太田 好彦 *1, 井上克巳 *1, 長谷川 隆三 *1, 中島 誠 *2

*1 (財) 新世代コンピュータ技術開発機構
*2 (財) 日本情報処理開発協会

1992 年 9 月 25 日

1 まえがき

定理証明は数学の定理証明の機械化という観点のみならず、知識情報処理システムの核として重要である。一階述語論理の定理証明器について、従来種々の研究がなされているが、最近、Prolog 上で極めて容易に実現できる SATCHMO[Manthey 88] とよばれるフル一階述語論理の定理証明器が提案された。これは、モデル生成 (Model Generation) とよばれるパラダイムに基づく定理証明器であり、与えられたプログラムの極小モデルを生成することを試みてその充足可能性をテストするシステムである。モデル生成動作の基本は、モデル候補を設定して、充足されない節を検出し、そのモデル候補を修正していくことであるととらえられる。さらに、このようなモデル生成に基づく定理証明器 MGTP [Fujita 91] が、並列言語 KL1 および疎結合型並列マシン PIM/m 上に実現され、極めて良好な性能を提供している。

しかしながら、あるモデル候補のもとで、充足されない節が複数存在する場合に、節の評価順序によって計算コストがかなり異なる。特に、複数の充足されない節の中にノンホーン節が含まれている場合には、節の評価順序によってモデル候補の爆発が生じてしまう場合がある。この節の評価順序に関して、プログラム中の負節は第一に優先しなければならないのは明らかである。また、ノンホーン節を適用する順序もまた、なるべく負節に依存した要素を追加するようなノンホーン節を優先的に選んだ方がよい。

SATCHMO の改良版および MGTP では、負節の優先機構は導入されているが、ノンホーン節と負節との依存関係 (関連性) によってノンホーン節の評価順序を決定する機構はない。

この点に関して、[Wilson 89] は、次のような関連性テスト (Relevancy Testing) を提案している。プログラムの確定節の集合と現時点のあるモデル候補 (ノンホーン節の適用によって追加された基礎アトムのみの集合でもよい) との和集合から、プログラムの任意の負節の否定を Prolog によって証明しようとして失敗した場合、その原因となったサブゴールを関連リテラル (Relevant Literal) とよぶ。充足されないノンホーン節が複数ある場合、後件の全てのアトムが関連リテラルとユニファイする場合に、そのノンホーン節を優先する。しかしながら、複数のノンホーン節が充足されないたびに Prolog を起動して、動的に関連性テストを行うオーバヘッドはかなり大きいものと思われる。さらに、いくつかのノンホーン節を複合して負節に関連するかしないかを調べることを要求すると、その関連性テストにはホーン理論のアブダクションと同等な計算が毎回必要になると考えられる。

一方、マジックセット法 (Magic Set)[Bancilhon 86] や Alexander 法 [Rohmer 86] が、従来、演繹データベースの分野で提案されている。これらの方法は、確定節の集合を対象として、ゴールに関連する基礎アトムのみを外延データベース (EDB) として生成するように内包データベー

ス (IDB) を変換するものである。また、[Bry 90] は、マジックセット法や Alexander 法を一般化した逆メタ解釈 (Upside-Down Meta-Interpretation) 法を提案している。これらの方法は、重複したサブゴールの証明は行わないというボトムアップ型の証明法の良い点を保存して、ゴールに関連したサブゴールの証明のみを行うというトップダウン型の証明法の良さを取り入れている。

本稿では、ホーン節集合に適用できる逆メタ解釈法をノンホーン節にも適用できるように拡張した方法 (ノンホーン・マジックセットとよぶ) について述べる。この方法は、関連性テストと同様にノンホーン節を含めたプログラム節に対して、負節と関連する充足されない節にモデル候補の拡張を限定でき、充足可能性テストの計算時間を削減することができる。特に、負節と関連しないノンホーン節の適用によるモデル候補の数の爆発を防ぐことができる。また、与えられたプログラムは、静的に変換され、証明器としてはモデル生成型定理証明器のみを用いるため、複数回起動される関連性テストにおける部分的な重複計算を避けることができ、関連性テストに伴うオーバヘッドをかなり削減することができると考えられる。なお、この方法を [Wilson 89] の例題に適用し Pseudo Muti-PSI システム上の MGTP で実験を行い、極めて良好な結果を得た。

2 定義

次の式を節という。

$$A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m. \tag{1}$$

ここに、$A_i (1 \leq i \leq n; n \geq 0)$ と $B_j (1 \leq j \leq m; m \geq 0)$ は一階述語論理のアトム、“,” は連言、“;” は選言、“→” は含意記号であり、含意記号の左辺を前件、右辺を後件という。なお、$m = 0$ の節を負節、$m = 1$ の節を確定節、負節あるいは確定節である節をホーン節といい、ホーン節でない節すなわち $m > 1$ なる節をノンホーン節という。以下、節の集合をプログラムといい、もとの節をプログラム節という。プログラム節は、その後件に出現する変数が全てその前件に出現していなければならない。この条件を満たす節を値域限定された (Range-restricted) 節と呼ぶ。

プログラム P のエルブラン基底の任意の部分集合 I_P をプログラム P のエルブラン解釈という。プログラム P の任意のエルブラン解釈 I_P は、以下の条件が成り立つときかつその時にかぎり節 (1) を充足するという。

1. I_P の要素でない $A_i\theta$ $(1 \leq i \leq n)$ が存在する。
2. I_P の要素である $B_j\theta$ $(1 \leq j \leq m)$ が存在する。

ここに、θ は節 (1) に出現する全ての変数に P のエルブラン領域の任意の要素を代入するものである。

プログラム P の任意のエルブラン解釈 I_P がプログラム P の全ての節を充足する場合、I_P は P を充足するという。P を充足する任意のエルブラン解釈 I_P は P のモデルとよばれる。なお、プログラム P のモデルの集合のうち、極小な要素の集合を P の極小モデルの集合という。

以下、P のモデルが存在するとき、P は充足可能であるという。また、P のモデルが存在しないとき、P は充足不能であるという。

3 モデル生成型定理証明器

モデル生成型定理証明器は、与えられたプログラム P の極小モデルを生成することを試み、P の充足可能性を判定するものである。モデル生成は、あるエルブラン解釈をモデルと仮定した

場合 (このようなエルブラン解釈をモデル候補とよぶ) に充足されない節 (違反する節ともいう) を検出し、その節を充足するようにエルブラン解釈を修正することによる。

MGTP の動作は次のようである。まず、モデル候補を最小とおく ($I_P := \emptyset$)。このモデル候補に違反する節を検出する。すなわち、P の任意の節 (1) で、全ての i ($1 \leq i \leq n$) について $A_i\theta \in I_P$ かつ全ての j ($1 \leq j \leq m$) について $B_j\theta \notin I_P$ なる基礎代入 θ が存在する場合、節 (1) は違反する。このような節 (1) を充足するようにモデル候補 I_P を修正するには、ある j ($1 \leq j \leq m$) について、$I_P := I_P \cup \{B_j\theta\}$ とすればよい (このようにモデル候補を修正することをモデル候補拡張とよぶ)。ここに、修正案は m 個存在する。$m \geq 2$ の場合は、そのうちの一つを選択すると共に $m-1$ 個のプロセスを新たに生成し、他の修正案も並列に実行する (これをモデル候補分岐とよぶ)。このように、修正された I_P について、同様に違反する節を検出し修正を試みる。もし、I_P で違反する節が P に存在しなければ、I_P は P のモデルであり、P は充足可能である。また、$m = 0$ の節が違反する場合、その節を充足するようにモデル候補 I_P を修正することはできないのでそのモデル候補を棄却する。もし、全てのモデル候補が棄却されれば P は充足不能である。

4　幅優先のノンホーン・マジックセット

以下、与えられたプログラムを P、P 中の任意の正節の集合を T とする。また、新しい述語記号 $goal/1$ を導入し、それをその対象が P のアトムであるメタ述語とする。$goal(A)$ はアトム A がゴールであると読む。

いま、$n \geq 1$ および $m \geq 0$ とし、$P \setminus T$ の任意の要素

$$A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m. \tag{2}$$

に対して、

$$goal(B_1), \cdots, goal(B_m) \rightarrow goal(A_1), \cdots, goal(A_n). \tag{3}$$

なる変換を施した節の集合を $(P \setminus T)^C$ とする。この節で $goal/1$ を論理否定 "$\neg$" に置換え、後件の "," を ";" で置換えると、この節はもとの節の対偶 (Contraposition) に対応する。後件の ";" を "," で置換えているのは、モデル候補分岐を行うと異なるモデル候補間で重複した関連リテラルを解くことが生ずるので、これを避けるためである。この変換で、もし節 (2) で $m = 0$ の場合は、後件に恒偽を意味する記号 $false$ が記述されているものとし、変換された節集合中 $goal(false)$ を恒真を意味する $true$ に置換えるものとする。すなわち、P 中の全ての負節の前件の各アトムはゴールである。また、負節以外の P の節にこの変換を施した節は、B_1 ないし B_m が全てゴールならば、A_1 ないし A_n は全てゴールである。

なお、この変換において、正節は変換対象にしていないから、$(P \setminus T)^C$ には負節は存在しない。

また、P の任意の要素

$$A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m. \tag{4}$$

に対して、

$$goal(B_1), \cdots, goal(B_m), A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m. \tag{5}$$

なる変換を施した節の集合を P^R とする。この変換を P の任意の負節に施した場合、もとの節が得られる。ここで、変換された節は、B_1 ないし B_m が全てゴールである場合に限り、もとの節をモデル生成の対象にせよと節の適用を制限していると読むことができる。

この様に変換された節集合に関して、以下の定理が得られる。

定理 1 与えられたプログラム P が充足不能であるための必要十分条件は、$(P \setminus T)^C \cup P^R$ が充足不能であることである。

[証明]

- 必要条件の証明
  $(P \setminus T)^C \cup P^R$ が充足不能ならば P が充足不能であることを証明するために、P が充足可能ならば $(P \setminus T)^C \cup P^R$ が充足可能であるを証明する。いま、P のある極小モデルを M とする。ここで、P のエルブラン基底の任意の要素 A について $goal(A) \in G$ とし、$M \cup G$ を考える。$(P \setminus T)^C$ の任意の節

$$goal(B_1), \cdots, goal(B_m) \rightarrow goal(A_1), \cdots, goal(A_n)$$

  は $n \geq 1$ である。よって、$M \cup G$ は、M の取り方に関係なく $(P \setminus T)^C$ の任意の節を充足する。次に、P^R の任意の節

$$goal(B_1), \cdots, goal(B_m), A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m$$

  について考える。全ての $j \ (1 \leq j \leq m)$ について、$goal(B_j)$ の任意の基礎例 $goal(B_j)\theta$ は $M \cup G$ に属する。また、M が P の極小モデルであることからある $i \ (1 \leq i \leq n)$ について $A_i\theta \notin M \cup G$ であるかまたはある $j \ (1 \leq j \leq m)$ について $B_j\theta \in M \cup G$ である。よって、$M \cup G$ は $(P \setminus T)^C \cup P^R$ のモデルであり、定理の必要条件が示された。

- 十分条件の証明
  P 充足不能ならば $(P \setminus T)^C \cup P^R$ が充足不能であることを証明するために、$(P \setminus T)^C \cup P^R$ のモデル M が存在すると仮定して矛盾を導く。

  M がモデルであることから P^R の任意の要素

$$goal(B_1), \cdots, goal(B_m), A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m$$

  の任意の基礎化代入 θ について、以下のいずれかが成り立つ。

  1. ある $i \ (1 \leq i \leq n)$ について $A_i\theta \notin M$ である。
  2. ある $j \ (1 \leq j \leq m)$ について $goal(B_j)\theta \notin M$ である。
  3. ある $j \ (1 \leq j \leq m)$ について $B_j\theta \in M$ である。

  したがって、もし全ての $j \ (1 \leq j \leq m)$ について $goal(B_j)\theta \in M$ であるならば、M がもとの節を充足する。いま、全ての $j \ (1 \leq j \leq m)$ について $goal(B_j)\theta \in M$ である P^R の基礎例

$$(goal(B_1), \cdots, goal(B_m), A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m)\theta$$

  に対応するもとの節の基礎例

$$(A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m)\theta$$

  の集合を P_r とすれば、M は P_r のモデルである。

  さらに、P の任意の基礎例の集合を P' とし、$P' \setminus P_r$ の任意の基礎節

$$A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m$$

に対して、ある j $(1 \leq j \leq m)$ について $goal(B_j) \notin M$ である B_j の集合を M' とすると、$M \cup M'$ は、$P' \setminus P_r$ の任意の節を充足する。

また、P_r の任意の基礎節

$$A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m$$

については、全ての j $(1 \leq j \leq m)$ について $goal(B_j) \in M$ であって、その $B_j \notin M'$ である。また、それに対応する $(P \setminus T)^C$ の基礎例

$$goal(B_1), \cdots, goal(B_m) \rightarrow goal(A_1), \cdots, goal(A_n)$$

が存在し、全ての i $(1 \leq i \leq n)$ について $A_i \notin M'$ であるから、$M \cup M'$ は P_r の全ての節を充足する。

よって、$M \cup M'$ は P のモデルである。これは、P が充足不能であることに矛盾する結果となるので、定理の十分条件が示された。

ところで、MGTP のプログラム節は、値域限定されて (Range-restricted) いなければならない。ここで示した変換方法は、もとの節が値域限定されていても、値域限定条件に違反する節を変換によって生成してしまうことがある。もし後件に出現する変数が前件に出現していないとき、[Manthey 88] は、dom 述語を前件に付加して値域限定された節に変換する方法を示している。しかしながら、dom 述語の項は、与えられたプログラムのエルブラン領域の任意の要素になるため、推論速度の低下は著しい。

ここで示した変換方法について、もとの節が値域限定されているとすると、値域限定条件違反を起こす可能性がある変数はメタ述語 $goal/1$ の内部にしか出現しない。よって、トップダウン型の証明法での入出力モードがオブジェクトレベルの各述語に関して一様であれば、次の方法によって上記問題点が解決できる。すなわち、

$$goal(B_1), \cdots, goal(B_m) \rightarrow goal(A_1), \cdots, goal(A_n).$$

において、前件に出現しない後件の変数すなわち出力モードの変数は、与えられたプログラムに出現しない定数 (以下の例では $\$(N)$;$N$ は自然数) で置換える。さらに、

$$goal(B_1), \cdots, goal(B_m), A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m.$$

なる節においては、$goal(B_j)$ $(1 \leq j \leq m)$ の各 B_j の出力モードの変数を新しい変数 (以下の例では匿名変数を用いている) に置換える。このようにすると、原理で示した変換方法を MGTP の機構を変更することなく用いることができる。

なお、もとのプログラム P を $(P \setminus T)^C \cup P^R$ に変換した場合、P のノンホーン節は、後件の全てのアトムが P 中の負節と関連している (本稿の場合は後件の全てのアトムがゴールである) 場合にもとのノンホーン節をモデル拡張の対象にするようにしており、$(P \setminus T)^C \cup P^R$ は関連性テストを模擬しているといえる。

また、ここで示した変換方法は、対象をホーン節に限り、$goal(A)$ を A に対する $magic$ 述語に置換えれば、幅優先のトップダウン型の証明法を模擬している。そこで、もとのプログラム P を $(P \setminus T)^C \cup P^R$ に変換してモデル生成型定理証明器で充足可能性テストを行うこの方法を幅優先のノンホーン・マジックセット法とよぶ。

5 深さ優先のノンホーン・マジックセット

先に示した幅優先のノンホーン・マジックセット法と同様な原理によって、深さ優先のトップダウン型の証明法を模擬することができる。

いま、与えられたプログラムを P とし、P の正節の集合を T とする。T の任意の正節については、原理で示した変換 T^R を用いる。$P \setminus T$ の節については、次の変換を行う。

$P \setminus T$ の任意の節

$$A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m. \tag{6}$$

に対して

$goal(B_1), \cdots, goal(B_m) \rightarrow goal(A_1), cont_{k,1}(V).$

$cont_{k,1}(V), A_1 \rightarrow goal(A_2), cont_{k,2}(V).$

$\cdots$

$cont_{k,(n-1)}(V), A_{n-1} \rightarrow goal(A_n), cont_{k,n}(V).$

$cont_{k,n}(V), A_n \rightarrow B_1; \cdots; B_m.$

なる $n+1$ 個の節に変換する。

ここに、V はもとの節に含まれた全ての変数の組、k はもとの節を他の節と区別するための識別子である。この変換された節は次のように読むことができる。

- B_1 ないし B_m が全てゴールであれば、A_1 がゴールであり、かつゴール B_1 ないしゴール B_m の変数の束縛情報と共に $(k,1)$ の次に継続 (Continuation) する。
- $(k,1)$ の次に継続しかつ A_1 であれば、A_2 はゴールでありかつゴール B_1 ないしゴール B_m の変数の束縛情報および A_1 の束縛情報と共に $(k,2)$ の次に継続する。
- $\cdots$
- $(k,n-1)$ の次に継続しかつ A_{n-1} であれば、A_n はゴールであり、かつゴール B_1 ないしゴール B_m の変数の束縛情報および A_1 ないし A_{n-1} の束縛情報と共に $(k,n-1)$ の次に継続する。
- $(k,n-1)$ の次に継続しかつ A_n であれば、B_1 または B_m である。

また、節(6)で $n \geq 1$ かつ $m \geq 0$ である。この変換で、もし $m = 0$ の場合は、恒偽を意味する記号 $false$ が記述されているものとし、変換された節集合中 $goal(false)$ を恒真を意味する $true$ に置換えるものとする。また、この変換においても変換後の節集合中には値域限定の条件に違反する節が存在するようになるが、先に述べた方法と同様な方法で解消する。なお、この変換によって得られたプログラムを $(P \setminus T)^{CR}$ と書く。基本原理で示したように、この変換においても以下の定理が同様にして示される。

定理 2 与えられたプログラム P が充足不能であるための必要十分条件は、$T^R \cup (P \setminus T)^{CR}$ が充足不能であることである。

以下、先に述べた幅優先のノンホーン・マジックセット法と比較して、この定理の証明の概略について示す。

必要条件の証明においては、P のモデルが存在すれば、それに全ての $goal(A)$ と $cont_{k,l}(V)$ の基礎例を追加すればそれが変換後のプログラムのモデルとなることが示せる。

十分条件の証明においては次のようである。本変換では、前件のあるアトム A_i より左に記述されているサブゴール A_{i-1} が解けなければ、その $goal(A_i)$ はモデル候補に追加されないようになっている。すなわち、A_{i-1} を解くことができなければ、その節は負節に関連していない節とみなす。負節に関連する節は、変換後のプログラムの極小モデルを M とすれば、M はその節を充足する。一方、負節に関連しない節

$$A_1, \cdots, A_n \rightarrow B_1; \cdots; B_m$$

は、ある A_{i-1} は M の要素ではないかまたはある $goal(B_j)$ が M の要素でない。A_{i-1} が M の要素ではない場合、当然、その節は充足される。また、ある $goal(B_j)$ が M の要素でない場合は、B_j を追加することによりその節を充足するようにできる。また、加えた B_j は、負節に関連する節中に出現しないかまたは M によって充足されない A_i が存在する。よって、M から先のものと同様に P のモデルが作れる。

ここで示した変換方法は、対象をホーン節に限れば、$goal(A)$ を A に対するマジック述語、$cont/1$ 述語を共有変数の束縛情報を伝播させる効果をもちかつ修飾子 (Adornments) 付きのマジック述語と解釈でき、$(P \setminus T)^{CR}$ の各変換節の最後の節は変形ルールに相当する。それ以外の節が、マジックルール (Magic Rules) に相当する。マジックセット法は本来、深さ優先のトップダウン型の定理証明法を模擬しており、もとのプログラム P を $T^R \cup (P \setminus T)^{CR}$ に変換してモデル生成型定理証明器で充足可能性テストを行う方法を深さ優先のノンホーン・マジックセット法とよぶ。

深さ優先のノンホーン・マジックセット法は、節の前件に共有変数を含んでいる場合に、その束縛情報によって生成するアトムの数を減少させることができる点が幅優先のノンホーン・マジックセット法と比べての利点である。欠点としては、新しい述語 $cont/1$ が導入されていることでモデル候補の要素数が大きくなる点が挙げられる。また、幅優先のノンホーン・マジックセット法はひとつの節が変換によって 2 つの節になるが、深さ優先のノンホーン・マジックセット法ではもとの節の前件の数を n として $n+1$ に増えてしまう。

6 実験結果

ノンホーン・マジックセット法を [Wilson 89] の例題に適用し Pseudo Muti-PSI システム上の MGTP で実験を行った結果について示す。なお、以下では、含意記号を "-->" で表している。

まず、以下の命題論理のプログラムを考える。

```
p-->false.
q-->false.
true-->r.
r-->a;b.
r-->p;c;d.
r-->p;q.
```

このプログラムを MGTP で実行させると、以下の 10 個のモデル候補を生成して、充足不能とわかる。

```
{p,a,r},{p,c,a,r},{q,c,a,r},{p,d,a,r},
{q,d,a,r},{p,b,r},{p,c,b,r},{q,c,b,r},
{p,d,b,r},{q,d,b,r}
```

なお、 Pseudo Multi-PSI システムでの MGTP の実行時間は 13[msec] であった。

このプログラムは変数を含んでいないから、幅優先のノンホーン・マジックセット法を適用し、以下のプログラムを生成する。

```
p-->false.
true-->goal(p).
q-->false.
true-->goal(q).
goal(r)-->r.
goal(a),goal(b)-->goal(r).
goal(a),goal(b),r-->a;b.
goal(p),goal(c),goal(d)-->goal(r).
goal(p),goal(c),goal(d),r-->p;c;d.
goal(p),goal(q)-->goal(r).
goal(p),goal(q),r-->p;q.
```

このプログラムを MGTP で実行すると、以下のような 2 つのモデル候補を生成して、充足不能とわかる。もとのプログラムでは、10 個のモデル候補を生成していたのが、負節に関連したモデル候補のみを生成して、以下の 2 個に減少させることができた。

```
{p,r,goal(r),goal(q),goal(p)},
{q,r,goal(r),goal(q),goal(p)}.
```

また、Pseudo Multi-PSI システムでの MGTP の実行時間は 9[msec] となり、もとのプログラムよりもやや短くなっている。

つぎに、以下のような変数を含んだプログラムを考える。このプログラムは、s(a),s(b) の方が s(c) よりも優先的に処理されると、モデル候補の数が爆発してしまう問題がある。実際、このプログラムを MGTP によって実行すると 20 分程度以内には解が得られなかった。

```
p(c,X,Y)-->false.
q(X,c,Y)-->false.
r(X,Y,c)-->false.
true-->s(c).
true-->s(b).
true-->s(a).
s(X),s(Y),s(Z)-->p(X,Y,Z);q(X,Y,Z);r(X,Y,Z).
```

なお、SATCHMO ではモデル候補の要素がキューで保存されているのに対して、MGTP ではスタックで保存されているため、

```
true-->s(c).
true-->s(b).
true-->s(a).
```

の順を [Wilson 89] と逆順にしてある。

また、このプログラムを深さ優先のノンホーン・マジックセット法で変換すると、次のようになる。

```
true-->goal(p(c,$(0),$(1))),cont_11([$(0),$(1)]).
cont_11([_,_]),p(c,X,Y)-->false.
true-->goal(q($(0),c,$(1))),cont_21([$(0),$(1)]).
cont_21([_,_]),q(X,c,Y)-->false.
true-->goal(r($(0),$(1),c)),cont_31([$(0),$(1)]).
cont_31([_,_]),r(X,Y,c)-->false.
goal(s(_))-->s(c).
goal(s(_))-->s(b).
goal(s(_))-->s(a).
goal(p(X,_,_)),goal(q(_,Y,_)),goal(r(_,_,Z))-->
    goal(s(X)),cont_41([X,Y,Z]).
cont_41([X,Y,Z]),s(X)-->goal(s(Y)),cont_42([X,Y,Z]).
cont_42([X,Y,Z]),s(Y)-->goal(s(Z)),cont_43([X,Y,Z]).
cont_43([X,Y,Z]),s(Z)-->p(X,Y,Z);q(X,Y,Z);r(X,Y,Z).
```

このプログラムを MGTP で実行すると、その実行時間は、31[msec] であり、以下のような 3 つのモデル候補のみを生成して充足不能とわかる。

```
{p(c,c,c),cont_43([c,c,c]),cont_42([c,c,c]),
 s(a),s(b),s(c),cont_41([c,c,c]),goal(s(c)),
 cont_31([$(0),$(1)]),goal(r($(0),$(1),c)),
 cont_21([$(0),$(1)]),goal(q($(0),c,$(1))),
 cont_11([$(0),$(1)]),goal(p(c,$(0),$(1)))}.

{q(c,c,c),cont_43([c,c,c]),cont_42([c,c,c]),
 s(a),s(b),s(c),cont_41([c,c,c]),goal(s(c)),
 cont_31([$(0),$(1)]),goal(r($(0),$(1),c)),
 cont_21([$(0),$(1)]),goal(q($(0),c,$(1))),
 cont_11([$(0),$(1)]),goal(p(c,$(0),$(1)))}.

{r(c,c,c),cont_43([c,c,c]),cont_42([c,c,c]),
 s(a),s(b),s(c),cont_41([c,c,c]),goal(s(c)),
 cont_31([$(0),$(1)]),goal(r($(0),$(1),c)),
 cont_21([$(0),$(1)]),goal(q($(0),c,$(1))),
 cont_11([$(0),$(1)]),goal(p(c,$(0),$(1)))}.
```

もとのプログラムでは、ノンホーン節に対して,次のようなモデル候補が生成されなければ充足不能であることがわからない。まず、項 a および b について $2^3 = 8$ の基礎例が存在し、各々 3 個のモデル候補を生成し、つぎに、変数 X, Y, Z のうち 1 つだけ c を代入する $2 \times 2 \times 3 = 12$ の基礎例について、各々 2 個のモデル候補が残り、つぎに、それらに対して、変数 X, Y, Z のうちいずれか 2 つの変数に c を代入する $3 \times 2 = 6$ の基礎例について、各々 1 個のモデル候補が残る。最終的に、変数 X, Y, Z の全てに c が代入される 1 個の基礎例について、そのモデル候補が

残らないのでここで充足不能であることがわかる。すなわち、生成されるモデル候補の総数は最大で、$3^8 \times 2^{12} \times 1^6 = 26,873,856$ となる。

また、

```
goal(s(_))-->s(c).
goal(s(_))-->s(b).
goal(s(_))-->s(a).
```

の順序を逆にして念のため、実験を行ってみたが、同様な結果が得られた。

なお、もとのプログラムに共有変数は含まれていないから、幅優先のノンホーン・マジックセットを適用したほうが、この場合効率的である。実際に実験してみると、3つのモデル候補を生成し充足不能であることがわかり、実行時間は 12[msec] であった。

7 関連研究

[Stickel 91] は、モデル生成型定理証明器上で逆メタ解釈 (Upside-Down Meta-Interpretation) を用いてモデル消去型定理証明手続き (Model Elimination Theorem-Prover Procedure) を模擬する方法を提案している。しかしながら、モデル消去型定理証明手続きでは、プログラムの任意の節の任意のリテラルをヘッドにもつ対偶ルール (Contrapositive Rules) が必要で、その節に出現するリテラルの数を n とすると、n 個のルールが必要になる。その各々は、ホーン節と同様な形式であり、これに深さ優先のホーン理論の逆メタ解釈法を適用すると n 個のルールが生成される。よって、ひとつの節に対して n^2 のルールが生成されることになる。本稿の方法によれば、1つの節に対して高々 n オーダのルールを生成すればよい。

また、[Demolombe 91] は、Alexander 法をノンホーン節に適用できるように拡張している。しかしながら、[Demolombe 91] の目的は、与えられたゴールが充足されるプログラムの全ての極小モデルを求めることである。プログラムの全ての極小モデル上でそのゴールを充足するかをテストするという本稿の目的とは異なる。当然、変換方法も異なる。

[Ramsay 91] は、モデル生成型定理証明において、プログラムの任意の節の前件に出現しない述語のアトムがノンホーン節の後件に少なくともひとつ出現するノンホーン節はそのノンホーン節を除いたプログラムで充足可能性テストを行ってもよいことを示している。この方法も冗長なモデル候補の生成を押さえる方法である。本稿の方法では、これを包含した冗長なモデル候補の生成を押さえることが可能である。

8 むすび

以上、ノンホーン理論に適用できるモデル生成型定理証明器のマジックセット法について述べた。この方法によって変換されたプログラムによって、もとのプログラムの充足可能性の判定ができることを証明した。また、Pseudo Multi-PSI システム上のモデル生成型定理証明器 MGTP 上で実験を行い、もとのプログラムで充足可能性検査を行うよりも変換後のプログラムを用いた方が少ないモデル候補を生成して問題を解決できかつ短時間に結果が得られることを示した。

なお、“失敗による否定 (Negation as Failure)” を含んだプログラムをモデル生成型定理証明器上で扱うことができる [Inoue 92] が、失敗による否定を含んだプログラムについてのマジックセット法は今後の課題である。

参考文献

[Bancilhon 86] Bancilhon,F., Maier,D., Sagiv,Y. and Ullman,J.D.: Magic Sets and Other

Strange Ways to Implement Logic Programs, *Proc. of the 5th ACM SIGMOD-SIGACT Symp. Principles of Database Systems*, pp.1–15 (1986).

[Bry 90] Bry,F.: Query evaluation in recursive databases: bottom-up and top-down reconciled, *Data & Knowledge Engineering*, **5**, pp.289–312 (1990).

[Demolombe 91] Demolombe,R.: An efficient strategy for non-Horn deductive databases, *Theoretical Computer Science*, **78**, pp.245–259 (1991).

[Fujita 91] Fujita,H. and Hasegawa,R.: A model generation theorem prover in KL1 using a ramified-stack algorithm, *Proc. of the 8th International Conference on Logic Programming*, pp.535–548 (1991).

[Inoue 92] Inoue,K., Koshimura,M. and Hasegawa,R.: Embedding negation as failure into a model generation theorem prover. *Proc. of the 11th International Conference on Automated Deduction, Lecture Notes in Computer Science*, **607**, pp.400–415 (1992).

[Manthey 88] Manthey,R. and Bry,F.: SATCHMO: a theorem prover implemented in Prolog, *Proc. of the 9th International Conference on Automated Deduction, Lecture Notes in Computer Science*, **310**, pp.415–434 (1988).

[Ramsay 91] Ramsay,A.: Generating Relevant Models, *Automated Reasoning*, **7**, pp.359–368 (1991).

[Rohmer 86] Rohmer,J., Lescoeur,R. and Kerisit,J.M.: The Alexander Method — A Technique for The Processing of Recursive Axioms in Deductive Databases, *New Generation Computing*, **4**, pp.273–285 (1986).

[Stickel 91] Stickel,M.E.: *Upside-Down Meta-Interpretation of the Model Elimination Theorem-Prover Procedure for Deduction and Abduction*, Technical Report TR–664, ICOT, Tokyo (1991).

[Wilson 89] Wilson,D.S. and Loveland,D.W.: *Incorporating Relevancy Testing in SATCHMO*, Technical Report CS-1989–24, Department of Computer Science, Duke University, NC (1989).