論　文

Technical Papers

ATMSを用いた前向き仮説推論システムにおける効率的な推論方式

An Efficient Inference Method for Forward-chaining Hypothetical Reasoning with the ATMS

太田　好彦*　井上　克巳*
Yoshihiko Ohta　Katsumi Inoue

* （財）新世代コンピュータ技術開発機構
Institute for New Generation Computer Technology, Tokyo 108, Japan.

1990年3月14日　受理

Keywords: hypothetical reasoning, default logic, assumption-based truth maintenance system, forward-chaining system, Rete algorithm.

Summary

Hypothetical reasoning is a basic technique for building AI system based on incomplete knowledge. Knowledge bases include hypotheses that are not always valid. Systems for hypothetical reasoning usually take a lot of time because they need to maintain the consistency of knowledge bases.

This paper presents an efficient method for hypothetical reasoning on a forward-chaining system with the assumption-based truth maintenance system (ATMS). The forward-chaining system consists of a compiler of Horn clauses and normal defaults into a Rete-like network and a Rete-based inference engine. The Rete-like network as a flow graph consists of a root node, one-input nodes, two-input nodes and terminal nodes. The Rete algorithm is an efficient method for matching a large collection of objects with many conjunctive patterns. The inference engine is extended to reason in multiple contexts without conflict resolution. It gives justifications that are propositional Horn clauses to the ATMS. In the ATMS, each datum is labeled with a collection of sets of assumptions where the datum holds.

The feature of a proposed reasoning method is that the inference engine gives intermediate justifications to the ATMS and stores intermediate dependent assumptions of two-input nodes, allowing faster hypothetical reasoning.

By means of this method, the APRICOT/0 system for hypothetical reasoning has been implemented on the PSI-II machine (Personal Sequential Inference machine) in ESP (Extended Self-contained Prolog). An experimental result shows that APRICOT/0 is about seven times faster than a system that neither gives intermediate justifications to the ATMS nor stores the intermediate dependent assumptions under a tested knowledge base. The compared system, called SCS, is a simple combination system of a conventional inference engine and the ATMS.

The cost of total ATMS label computations on APRICOT/0 is generally less than the cost of the compared system in the following cases:

(1) When the Rete-like network includes a two-input node shared by some clauses or defaults.
(2) When a one-input node passes tokens to its successor that is a successor of another two-input node.

1. まえがき

仮説推論は，常に正しいかどうかわからない知識の集合を仮説集合とし，この中から常に正しい知識（事実）の集合と矛盾しない部分集合を加えて結論を導く推論形態であり，不完全な知識ベースに基づく知識システム構築のために重要な要素技術である[1][2]．しか

しながら，導入した仮説部分集合を含めた知識ベースの無矛盾性をチェックしなければならず，推論速度の点で問題がある．

従来提案されているAssumption-based Truth Maintenance System (ATMS)[3]は，命題論理の事実集合と仮説集合とを入力としデータの真偽値を管理するシステムである．知識システムに使用する場合は直接このインタフェースを用いる場合は少なく，問題解決器を介するのが一般的である．文献(4)は，ATMSに制約言語の問題解決インタフェースを設けたシステムである．また，一階述語論理ベースの問題解決器とATMSとにより構成されたシステムは，問題解決器が推論を行いATMSが与えられた事実集合と仮説部分集合との無矛盾性の管理を行って，全体として仮説推論を行う．このような問題解決器は，対象問題の解決に必要とする知識ベースと推論エンジンとから構成される．推論エンジンは，前向き推論エンジンあるいは後向き推論エンジンに分けられる．このうち，前向き推論は，多くの許容される解が存在する計画や設計の問題において重複した計算を避けることができるので好ましい．

本論文では前向き推論エンジンとATMSとを結合した仮説推論システムについて論じる．従来提案されている例として文献(5)や文献(6)があげられる．これらは，ATMSとプロダクションシステムとを結合した仮説推論システムである．従来，プロダクションシステムにおける高速パターンマッチングアルゴリズムとしてRete[7]が提案されている．Reteアルゴリズムは，ルールセットをネットワークに変換しておき，そのなかでパターンマッチングを段階的に行い，その結果をネットワークに保存しておく方法である．文献(5)や文献(6)ではReteネットワーク内でパターンマッチングとATMSで行われるデータが成り立つ仮説部分集合の計算（ラベル計算）を融合し仮説推論の高速化を図っている．文献(5)は，独自の仮説ネットワークと調整アルゴリズムとを提案し，推論エンジンと無矛盾性管理機構とを完全に融合しているが，課題として，多重コンテキストを扱えるようにすることをあげている．文献(6)では，その処理方式についての詳細は述べられていない．また，文献(8)や文献(9)のシステムでも，前向き推論方式にReteアルゴリズムを採用し，無矛盾性管理をATMSによって行っているが，Reteネットワークに対応付けてラベル計算は行われていない．

本論文で述べる前向き仮説推論方式は，Reteに類似したネットワークを用い，そこで段階的に作成されるパターンマッチングの結果に対応して事実や仮説をATMSに送り，ATMSによって計算されるそのパターンマッチングの結果に対応するデータが成立する仮説部分集合を保存することにより，効率的な仮説推論を実現するものである．従来のシステム例と比較した本方式の特徴は，入力知識ベースをReiterの正規デフォルト理論[10]のサブセットに基づいた論理プログラム（一階述語論理ベース）としていること，ATMSと前向き推論エンジンとをモジュール化してそれぞれ別々にも使用できるような方式であること，さらに，ルールの適用に関する競合解消を行う必要がなく多重コンテキストで推論を行う方式であること等があげられる．また，本推論方式を採用した仮説推論システムAPRICOT/0[11]をPSI-II[12]マシン上に言語ESP[13]でインプリメントし，その有効性を確認した．

2章で知識ベースを定義し，**3章**で前向き推論エンジンとATMSを単純に結合した仮説推論システムについて示し，**4章**でAPRICOT/0の推論方式について説明し**5章**で評価する．

2. 入力知識ベース

2・1 定　　義

本論文で取り扱う仮説推論システムの入力となる知識ベースは，Reiterの正規デフォルト理論[10]のサブセットに基づいた論理プログラムであり，次に示すFとDとにより定義される．

Fは，ホーン節の集合である．ホーン節（単に節と呼ぶこともある）は，以下の二つの式の何れかである．

$$\alpha_1 \wedge \cdots \wedge \alpha_n \rightarrow \beta. \quad (1)$$

$$\alpha_1 \wedge \cdots \wedge \alpha_n \rightarrow \perp. \quad (2)$$

ここに，$\alpha_i (i=1, \cdots, n\ ; n \geqq 0)$および$\beta$は一階述語論理のアトム（Atomic formula）であり，$\wedge$は連言，$\rightarrow$は含意，$\perp$は偽である．また，$\alpha_1 \wedge \cdots \wedge \alpha_n$を前件，$\beta$を後件と呼び(節(2)の後件は偽)，後件に含まれるすべての変数は前件に出現していなければならないとする．

Dは，$\alpha_1 \wedge \cdots \wedge \alpha_n : \beta / \beta$のような推論規則で定義される正規デフォルト（単にデフォルトと呼ぶこともある）の集合である．ここに，$\alpha_1 \wedge \cdots \wedge \alpha_n$をデフォルトの前提，$\beta$をデフォルトの結論と呼び，結論に含まれるすべての変数は前提に出現していなければならないとする．

2・2 知識ベースの例

以下の例題を考え，前記知識ベースの一例を示す．

部門$s1$の1人と部門$s2$の1人と同時にミーティングを行う．部門$s1$のaとb，部門$s2$のcとdはこのミーティングに出席できると考えられる．また，そのミーティングの場所の候補は，会議室1から3，ラウンジeまたはwである．会議室は空いていること，ラウンジは静かであるという条件がある．普通は，会議室は空いていて，ラウンジは静かである．しかし，会議室1が予約済みであること，および，ラウンジwは騒々しいことがこのときすでにわかっている．以上の知識のもとで，誰とどこでこのミーティングを行えるか．Fig. 1はこの例題を解くための知識ベースの一例である．

```
%% F %%

r (1) .
r (2) .
r (3) .
l (e) .
l (w) .
nv (1) .
nq (w) .

p (X, Y) ∧np (X, Y)  → ⊥ .

v (X) ∧nv (X)  → ⊥ .

q (X) ∧nq (X)  → ⊥ .

p (X, s1) ∧p (Y, s2) ∧v (Z)  → m (X, Y, Z) .

p (X, s1) ∧p (Y, s2) ∧q (Z)  → m (X, Y, Z) .

%% D %%

:p (a, s1) /p (a, s1) .
:p (b, s1) /p (b, s1) .
:p (c, s2) /p (c, s2) .
:p (d, s2) /p (d, s2) .

r (X) :v (X) /v (X) .

l (X) :q (X) /q (X) .

%% r      :room,
%% l      :lounge,
%% nv     :not vacant,
%% nq     :not quiet,
%% p      :present,
%% np     :not present,
%% v      :vacant,
%% q      :quiet,
%% m      :meeting,
%% X, Y, Z:variables.
```

Fig.1 Knowledge base of the meeting plan.

3. 前向き仮説推論システムの一例

3・1 概　　要

Fig. 2は，前向き推論エンジンとATMSとを単純に結合した仮説推論システム（以下，Simple Combination System, SCSと呼ぶ）の構成を示すものである．

ATMSは，命題論理レベルのホーン節(理由付けと呼ぶ）の集合と仮説集合とを入力し，命題論理レベルのアトム（データと呼ぶ）の真偽値を管理して出力するシステムである．入力は，推論エンジンからインクリメンタルにATMSに与えられる．いま，あるデータが真であるとすると，それはそのデータがある仮説部分集合のもとで現在信じられている（IN状態と呼ぶ）という意味である．すなわち，後から追加される知識によって，そのデータが信じられていない状態(OUT状態と呼ぶ）になる可能性を持っている．

推論エンジンは，ATMSから通知されるIN状態である基礎アトムと節の前件とのマッチングにより融合を行い，基礎化代入された後件を導く．このとき，その後件への理由付けをATMSに通知する．同様に，ATMSから通知されるIN状態である基礎アトムとデフォルトの前提とのマッチングにより融合を行い，前提が空になったときに基礎化代入された結論が真であるという仮説をたて，その結論を導く．このとき，その結論への理由付けをATMSに通知する．

3・2 ATMS

理由付け（Justification）は，ホーン節に対応するが，先のFの要素と区別するために，以下の表記を用いる．

式(1)とその基礎化代入θおよび$a_i\theta$（$i=1,\cdots,n$）が

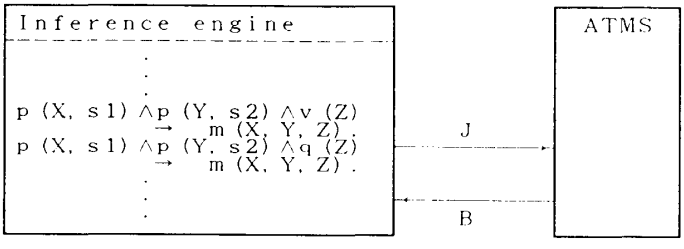


Fig.2 Configuration of the Simple Combination System (SCS).

IN状態であることから$\beta\theta$が導かれる．このとき，推論エンジンがATMSに通知する理由付けを下式で表す．

$$\alpha_1\theta,\cdots,\alpha_n\theta\Rightarrow\beta\theta.$$

また，式(2)とその基礎化代入θおよび$\alpha_i\theta$ ($i=1,\cdots,n$)がIN状態であることから，偽が導かれる．このとき，推論エンジンがATMSに通知する理由付けを下式で表す．

$$\alpha_1\theta,\cdots,\alpha_n\theta\Rightarrow\perp.$$

さらに，$a_1\wedge\cdots\wedge a_n:\beta/\beta$とその前提の基礎化代入$\theta$および$\alpha_i\theta$ ($i=1,\cdots,n$)がIN状態であることだけからは$\beta\theta$をIN状態とすることはできないので，$\beta\theta$が真である仮説（これを$\Gamma\beta\theta$と記述する）をたて$\beta\theta$を導く．このとき，推論エンジンがATMSに通知する理由付けを下式で表す．

$$\alpha_1\theta,\cdots,\alpha_n\theta,\Gamma\beta\theta\Rightarrow\beta\theta.$$

ここに，記号“⇒”の左を理由付けの前件，右を理由付けの後件と呼ぶ．

ATMSは，データ（基礎アトム）に対応して以下のような構造体（ATMSノードと呼ぶ）で真偽値を管理する．

＜データ，ラベル，理由付け＞．

理由付けの前件（“(”と“)”とで一つ前件を表す）はATMSノードの理由付けの項に記録される．また，現時点までに与えられた理由付けの集合をJで表す．

仮説の集合を環境(Environment)と呼びEで表す．ラベルは，環境の集合である．あるデータは，対応するATMSノードのラベルの任意の要素の環境と理由付けの集合Jとで証明される．

環境Eと理由付けの集合Jとにより，ある基礎アトムが導かれるようなすべての環境Eの集合をその基礎アトムの完全ラベルという．また，ある環境Eと理由付けの集合Jより偽が導かれるとき，EをJに対して矛盾環境という．理由付けの集合Jと矛盾しないある環境のもとで導くことができるすべてのデータの集合を一つのコンテキストと呼ぶ．完全ラベルの任意の要素が他の要素を包含する環境（上位集合）ならばその環境（上位集合のほう）を完全ラベルから取り除いた集合を完全ラベルの極小集合と呼ぶ．ラベルは，完全ラベルの極小集合から矛盾環境を取り除いた環境の集合である．

$\alpha_i\theta$ ($i=1,\cdots,n$；$n\geqq 0$)を前件とする理由付けが与えられたとき，更新が必要なラベルは以下の手順により計算される．例は，次節で示す．

① 後件に対応するATMSノードのラベルをL，$\alpha_i\theta$に対応するATMSノードのラベルをL_iとして，L'を計算する．

$$L'=\{x|x=\bigcup_i E_i,\ E_i\in L_i\}.$$

L'は，環境をビットベクタで表現しておくとビット演算orにより計算できる．

② $L\cup L'$から矛盾環境および他を包含する環境を取り除いた集合L''を後件に対応するATMSノードの新しいラベルとする．

③ もし$L=L''$ならば終了し，さもなければ④を行う．

④ もし後件が偽ならば$E\in L''$なるEを包含する環境を偽に対応するATMSノードを除くすべてのATMSノードのラベルから削除する．さもなければ，この後件が理由付けの項に記録されているATMSノードのラベル計算を①より行う．

3・3 推論方式

Fig. 3は，SCSの推論アルゴリズムを示すものである．図中fは，融合により新しい基礎アトムが導かれたら1となるフラグである．**2章**に示した例題を用いてこれを説明する．なお，処理の流れにそって生成されるATMSノードあるいはそのラベルが変更されるATMSノードをFig. 4に示す．

Fig. 3の①は，Fの要素のうち前件が空の節βに対して理由付け⇒βをATMSに送る処理を示す．このβに対応するATMSノードは，

$$<\beta,\{\{\}\},\{()\}>$$

である．ラベルは，βが仮説の数が0の環境で成り立っていることを示している．このような理由付けの例は，$r(1)$について，$\Rightarrow r(1)$である．

Fig. 3の②は，Dの要素のうち前提が空のデフォルト：β/βに対して，理由付け$\Gamma\beta\Rightarrow\beta$をATMSに送る処理を示す．この仮説$\Gamma\beta$に対応するATMSノードは，

$$<\Gamma\beta,\{\{\Gamma\beta\}\},\{(\Gamma\beta)\}>$$

である．このような理由付けの例は，：$p(a,s1)/p(a,s1)$について，$\Gamma_{p(a,s1)}\Rightarrow p(a,s1)$である．

以上の処理で$f=1$となる．

Fig. 3の③は，前件が空でない節や前提が空でないデフォルトにより新しい基礎アトムを導く処理を示している．まず，$f\leftarrow 0$とする．Dの要素$r(X):v(X)/v(X)$と$r(1)$がIN状態であることから，仮説$\Gamma_{v(1)}$をたてて，以下の理由付けをATMSに送る．

$$r(1),\ \Gamma_{v(1)}\Rightarrow v(1).$$

ATMSは$v(1)$に対応するATMSノードのラベル計算を行う．まず，$\{\{\}\cup\{\Gamma_{v(1)}\}\}$を計算する．以前，このATMSノードは存在しなかったので，それから矛盾環

```
 begin
 for all C in F do
  if C=β then
①   begin
      give the justification ⇒β to the ATMS;
      f←1
    end;
 for all R in D do
 if R=(:β/β) then
②  begin
     give the justification Γβ⇒β to the ATMS;
     f←1
   end;
 while f=1 do
   begin
     f←0;
     for all C in F do
      if C=(α1∧…∧αn→β) then
       if αiθ (i=1, …, n) are believed then
         begin
           give the justification (α1θ, …, αnθ⇒βθ) to the ATMS;
           if βθ is a new datum then
             f←1
③          end
      else if C=(α1∧…∧αn→⊥) then
       if αiθ (i=1, …, n) are believed then
         give the justification (α1θ, …, αnθ⇒⊥) to the ATMS
     for all R in D do
      if R=(α1∧…∧αn:β/β) then
       if αiθ (i=1, …, n) are believed then
         begin
           give the justification (α1θ, …, αnθ, Γβθ⇒βθ) to the ATMS;
           if βθ is a new datum then
             f←1
         end
   end
 end
```

Fig. 3 Reasoning algorithm on SCS.

境および他を包含する環境を取り除く．現時点では矛盾環境の極小集合（⊥に対応するATMSノードのラベル）は{}であり，その集合の要素は一つなので，ラベルは$\{\{\Gamma_{v(1)}\}\}$である．後件は偽でないので，この後件が理由付けの項に記録されているATMSノードのラベル計算を行う．この後件$v(1)$が理由付けの項に記録されているATMSノードは，現在存在しないのでラベル計算は終了する．$r(2)$と$r(3)$についても同様である．また，Dの要素$l(X):q(X)/q(X)$と$l(e)$がIN状態であることから，同様に$q(e)$もIN状態になる．$l(w)$についても同様である．上記のように，新しいデータが生成されて$f\leftarrow 1$となる．

$f=1$なので，Fig. 3の③の処理を行う．$f\leftarrow 0$とする．現時点において，$nv(1)$，$v(1)$が信じられているので，以下の理由付けをATMSに送る．

$$nv(1),\ v(1)\Rightarrow\perp.$$

これらのラベル計算も前記と同様であるが，後件が偽であるので，$\{\Gamma_{v(1)}\}$を包含する環境を他のすべてのATMSノードのラベルから削除する．$v(1)$に対応するATMSノードのラベルから$\{\Gamma_{v(1)}\}$が削除されラベルは{}（成り立つ環境がない，すなわちOUT状態）となる．$nq(w)$，$q(w)$についても同様である．

続いて，$p(a,s1)$，$p(c,s2)$，$v(2)$が信じられていることより，以下の理由付けをATMSに送る．

$$p(a,s1),\ p(c,s2),\ v(2)\Rightarrow m(a,c,2).$$

この理由付けを受けたATMSは，以下のラベル計算を行う．すなわち以下の式を計算する．

$$\{\Gamma_{p(a,s1)}\}\cup\{\Gamma_{p(c,s2)}\}\cup\{\Gamma_{v(2)}\}. \tag{3}$$

次に，$\{\{\Gamma_{p(a,s1)},\ \Gamma_{p(b,s2)},\ \Gamma_{v(2)}\}\}$から矛盾環境および他を包含する環境を取り除く．現時点で，矛盾環境の極小集合は$\{\{\Gamma_{v(1)}\},\{\Gamma_{q(w)}\}\}$であり，また，この集合の要素は一つなので，それがラベルとなる．後件は偽でないので，この後件が理由付けに記録されているATMSノードのラベル計算を行う．しかしながら，この後件$m(a,c,2)$が理由付けの項に記録されているATMSノードは，現在存在しないのでラベル計算は終了する．同様に，$p(b,s1)$，$p(d,s2)$，$v(3)$および$q(e)$が信じられていることより，その他の解も導かれる．

$f=1$なので，Fig. 3の③の処理を行う．$f\leftarrow 0$とする．今回は，新しいデータが一つも導かれないので

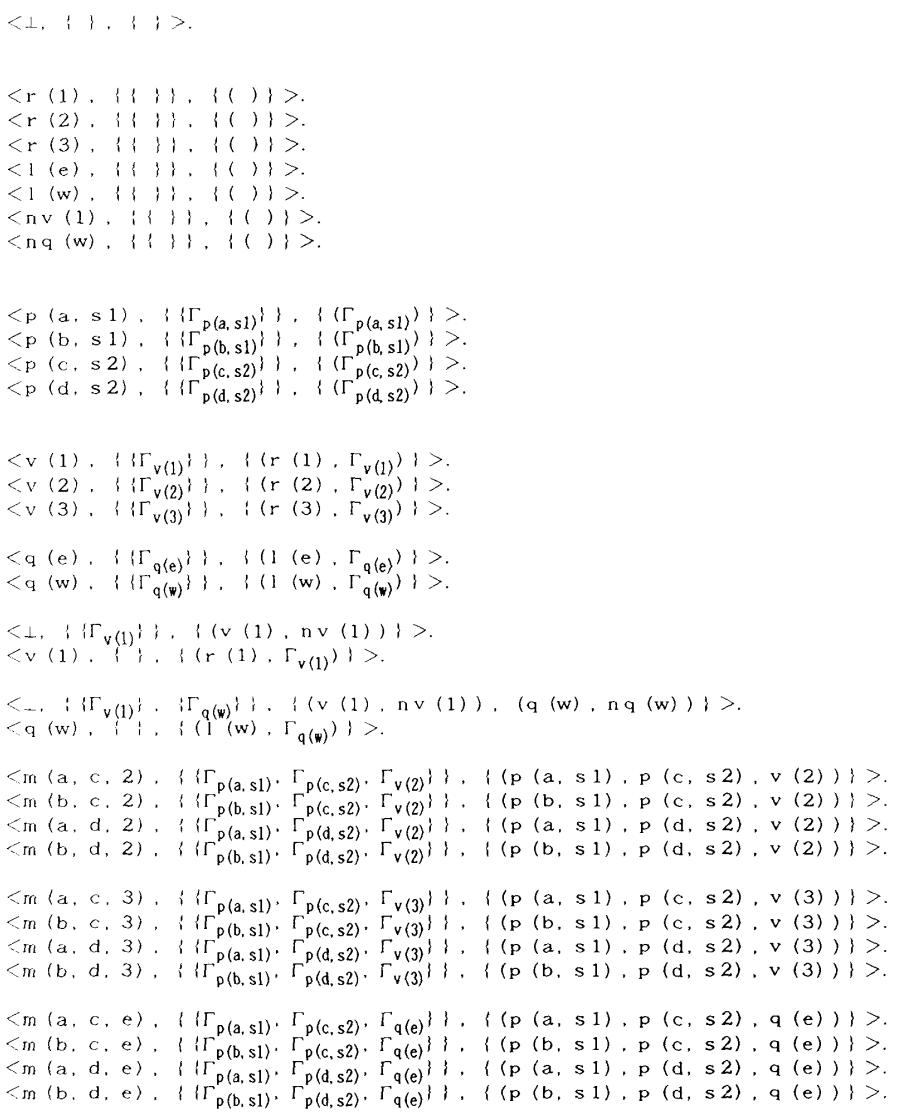


Fig. 4 ATMS nodes of the meeting plan.

$f=0$ のまま推論エンジンは停止する.

このように，最終的に少なくとも IN 状態のすべての基礎アトムはそのラベルとともに必ず出力される (Fig. 4 参照).

4. APRICOT/0

4・1 概　　要

APRICOT/0 は，入力知識ベースを Rete-like ネットワークと呼ぶフローグラフに変換するコンパイラ，Rete-like ネットワークに基づく前向き推論エンジンおよび ATMS とから構成される. ATMS は前章で示したものと同一である. 前向き推論エンジンおよびそれと ATMS との結合方式が異なる. Rete-like ネットワークには，基礎アトムまたは基礎アトムと連言によって定義される基礎式に対応するデータ（これをトークンと呼ぶ）が流れることにより推論が実行される.

4・2 Rete-like ネットワーク

Rete-like ネットワークは，Rete ネットワークと同様に 1 個の Root ノード，複数の 1 入力ノード，2 入力ノード，終端ノードおよびそれらを結ぶリンクから構成される.

① Root ノード

Root ノードは，スロット SUCCESSOR を持ち，すべての 1 入力ノード（この後続ノード）へのポインタの集合を保持している.

② 1 入力ノード

F の任意の要素 $\alpha_1 \wedge \cdots \wedge \alpha_n \rightarrow \beta$, $(n \neq 0)$ の前件あるいは D の任意の要素 $\alpha_1 \wedge \cdots \wedge \alpha_n : \beta / \beta$ $(n \neq 0)$ の前提に記述されている各アトム α_i $(i=1, \cdots, n\ ;\ n \geqq 1)$ について，α_i を PATTERN とした 1 入力ノードが生成される. このとき，Root ノードの SUCCESSOR スロットにこの 1 入力ノードへのポインタを追加しておく.

この1入力ノードは，PATTERNについて一元管理されており，以下のスロットを持つ．

PATTERN：α_i.

SUCCESSOR：2入力ノード（この後続ノード）へのポインタの集合．

TERMINAL：もし，PATTERNと一致する前件の節または前提のデフォルトが存在すればその節またはデフォルトが記録された終端ノードへのポインタの集合．

WM：推論実行時に使用するワーキングメモリで，基礎アトム$\alpha_i\theta$の集合（初期状態は空）．

③ 2入力ノード

Fの任意の要素$\alpha_1 \wedge \cdots \wedge \alpha_n \rightarrow \beta$，$(n \neq 0)$の前件あるいは$D$の任意の要素$\alpha_1 \wedge \cdots \wedge \alpha_n : \beta / \beta$ $(n \neq 0)$の前提の各アトムα_j $(j=2, \cdots, n)$と式$\alpha = \wedge_i \alpha_i$ $(i=1, \cdots, j-1)$との連言$\alpha \wedge \alpha_j$をスロットPATTERNの内容とする2入力ノードが生成される．この2入力ノードは，PATTERNについて一元管理されている．これは，1入力ノードが有しているスロットに加え，先行ノードへのポインタの集合（αに対応する2入力ノードおよびα_jに対応する1入力ノードへのポインタ）を保持するPREDECESSORなるスロットを持つ．なお，WMは基礎式$(\alpha \wedge \alpha_j)\theta$の集合となる．

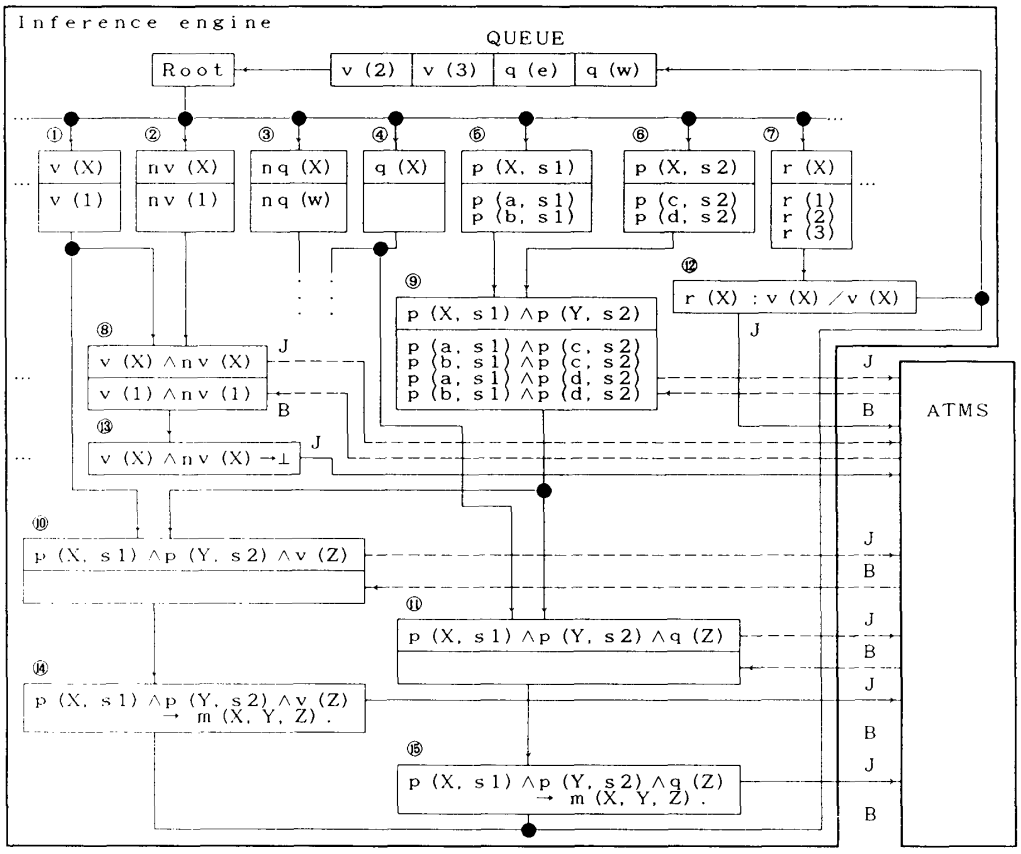
④ 終端ノード

Fの一要素$\alpha_1 \wedge \cdots \wedge \alpha_n \rightarrow \beta$ $(n \neq 0)$あるいはDの一要素$\alpha_1 \wedge \cdots \wedge \alpha_n : \beta / \beta$ $(n \neq 0)$が記録されている

例題に対応するRete-likeネットワークの部分をFig. 5の推論エンジン内に示す．図において，①から⑦までは1入力ノード，⑧から⑪までは2入力ノード，⑫から⑮までは終端ノードである．1入力ノードおよび2入力ノードは，上半分にPATTERNの内容，下半分にWMの内容が示されている．図は，さらにAPRICOT/0における構成を示しており，前向き推論エンジンとATMSとの結合が示されている．また，推論エンジンは，トークンを蓄えておく一つのキュー(QUEUE)を有している．

4・3 推論方式

Fig. 6およびFig. 7に推論アルゴリズムを示す．Fig. 6は，前件が空の節または前提が空のデフォルトに対応する理由付けをATMSに送り，対応する基礎アトムをキューに連結する(①)．次に，キューから要



J: Justifications
B: Beliefs

Fig. 5 Configuration of APRICOT/0.

```
 begin
 make QUEUE empty;
 for all C in F do
  if C=β then
   begin
    give the justification ⇒β to the ATMS;
    concatenate β to the end of QUEUE
①  end
 for all R in D do
  if R=(:β/β) then
   begin
    give the justification Γβ⇒β to the ATMS;
    concatenate β to the end of QUEUE
   end;
 while QUEUE not empty do
  begin
  let τ be the first element on QUEUE;
② move τ from QUEUE to the root node;
③ for all j in SUCCESSOR of the root node do
   begin
    let α_j be PATTERN of j;
④   if τ=α_jθ then
     begin
⑤     store the copy of τ in WM of j;
⑥     PASS(τ, j, QUEUE, ATMS)
     end
   end
  end
 end
```

Fig.6 Reasoning algorithm on APRICOT/0.

素を取り出し(②)，Rootノードよりすべての1入力ノードに流す(③)．1入力ノードとマッチするトークン(④)はそのWMに保存され(⑤)後続ノードに流される(⑥)が，そのアルゴリズムはFig. 7に示されている．Fig. 7は，あるノードが指示する終端ノードが記録されている節やデフォルトに対応する理由付けをATMSに送り，基礎化代入された後件や結論をキューに連結する(⑦)．また，トークンを後続ノードにパスし(⑧)，その一方の先行ノードのWMに保存されている基礎アトムとの連言をとったデータ（中間的なデータと呼ぶ）とPATTERNとのマッチングを行い(⑨)，成功すればその中間的なデータへの理由付けをATMSに送る(⑩)とともに中間的なデータをWMに保存する(⑪)．その中間的なデータをトークンとしてさらに後続ノードへパスする(⑫)．以下，2章に示した例題を用いてこれらを説明する．

まず，$r(1)$，$r(2)$，$r(3)$，$l(e)$，$l(w)$，$nv(1)$，$nq(w)$ および$p(a,s1)$，$p(b,s1)$，$p(c,s2)$，$p(s2,d)$がキュー上に並んでいる．この先頭要素からトークンとしてRootノードよりRete-likeネットワークに流す．$r(1)$ は，Fig. 5に示す1入力ノード⑦でマッチングに成功し，そのWMに保存される．1入力ノード⑦は終端ノード⑫を持っているので，その処理を行う．その結果，$v(1)$ がキューの末尾に連結される．$r(2)$，$r(3)$，$l(e)$，$l(w)$についても同様に処理される．次に$nv(1)$，$nq(w)$は，それぞれ1入力ノード②③のWMに保存される．

次に，$p(a,s1)$ および$p(b,s1)$ がトークンとして流れ，1入力ノード⑤でマッチングに成功し，そのWMに保存される．さらに，$p(c,s2)$ がトークンとして流れ，1入力ノード⑥のWMに保存される．そのトークンは，後続ノード⑨の先行ノード⑤のWMの要素$p(a,s1)$および$p(b,s1)$と連言によって結合される．これらの要素は現時点でIN状態であり，連言によって結合されたデータは2入力ノード⑨のPATTERNとマッチする．よって，以下の中間的な理由付けがATMSに送られる．

$$p(a,s1),\ p(c,s2)\Rightarrow p(a,s1)\wedge p(c,s2). \quad (4)$$

$$p(b,s1),\ p(c,s2)\Rightarrow p(b,s1)\wedge p(c,s2). \quad (5)$$

理由付け(4)を受けたATMSは，後件に対応するATMSノードのラベルを計算するために，以下の式を計算する．

$$\{\Gamma_{p(a,s1)}\}\cup\{\Gamma_{p(c,s2)}\}. \quad (6)$$

同様に理由付け(5)によるラベル計算が行われる．これらの理由付けの後件は2入力ノード⑨のWMに保存される．トークン$p(d,s2)$ についても同様である．

また，$v(1)$は，1入力ノード①のWMに保存される．後続ノード⑧において，先行ノード②のWMには$nv(1)$が保存されている．$nv(1)$はIN状態であり，連言によって結合されたデータ$v(1)\wedge nv(1)$は2入力ノ

```
procedure PASS (τ, i, QUEUE, ATMS) :
begin
 for all T in TERMINALS of i do
  if T=(α1∧…∧αn→β) then
   if τ=(α1∧…∧αn)θ then
    begin
     give the justification τ⇒βθ to the ATMS;
     if βθ is a new datum then
      concatenate βθ to the end of QUEUE
    end
  else if T=(α1∧…∧αn→⊥) then
   if τ=(α1∧…∧αn)θ then
    give the justification τ⇒⊥ to the ATMS
  if T=(α1∧…∧αn:β/β) then
   if τ=(α1∧…∧αn)θ then
    begin
     give the justification (τ, Γβθ⇒βθ) to the ATMS;
     if βθ is a new datum then
      concatenate βθ to the end of QUEUE
    end;
 if τ is believed then
  for all j in SUCCESSOR of i do
   begin
   let k be another predecessor node (k≠i) of j;
   for all ε in WM of k do
    if ε is believed then
     begin
     let (α∧αj) be PATTERN of j;
     if τ∧ε=(α∧αj)θ then
      begin
       give the justification (τ, ε⇒τ∧ε) to the ATMS;
       if τ∧ε is believed then
        begin
         store the copy of τ∧ε in WM of j;
         PASS (τ∧ε, j, QUEUE, ATMS)
        end
      end
     end
   end
end
```

Fig. 7 Token pass algorithm on APRICOT/0.

ード⑧の PATTERN とマッチする．これにより以下の中間的な理由付けが ATMS に送られる．

$v(1),\ nv(1) \Rightarrow v(1) \wedge nv(1)$.

さらに，2 入力ノード⑧から指示される終端ノード⑬により，以下の理由付けが ATMS に送られる．

$v(1) \wedge nv(1) \Rightarrow \perp$.

この結果，$\{\Gamma_{v(1)}\}$ が矛盾環境となりデータ $v(1)$ およびトークンである $v(1) \wedge nv(1)$ の信念の状態が OUT となる．したがって，トークン $v(1) \wedge nv(1)$ は 1 入力ノード①の後続ノード⑩以下には流れない．Fig. 5 のキューや各 WM の状態は，この時点である．

続いて，$v(2)$ がトークンとして流れ，1 入力ノード⑧の WM に保存される．後続ノード⑧の先行ノード②の WM には $nv(1)$ が保持されている．$nv(1)$ は IN 状態であるが，連言によって結合されたデータ $v(2) \wedge nv(1)$ は 2 入力ノード⑧の PATTERN とマッチしない．そこで，もう一方の後続ノード⑩にトークン $v(2)$ が流れていく．その先行ノード⑨の WM には Fig. 5 に示すデータが保存されている．まず，$p(a, s1) \wedge p(c, s2)$ について，以下の中間的な理由付けが ATMS に送られる．

$p(a, s1) \wedge p(c, s2),\ v(2)$
$\Rightarrow p(a, s1) \wedge p(c, s2) \wedge v(2)$

この理由付けを受けた ATMS は，後件に対応する ATMS ノードのラベルを求めるために，以下の式を計算する．

$$\{\Gamma_{p(a,s1)}, \Gamma_{p(c,s2)}\} \cup \{\Gamma_{v(2)}\}. \tag{7}$$

トークン $p(a, s1) \wedge p(c, s2) \wedge v(2)$ は，2 入力ノード⑩の後続ノードにパスされる．2 入力ノード⑩は終端ノード⑭をもつので，以下の理由付けを ATMS に送り，新しいデータ $m(a, c, 2)$ を導く．

$p(a, s1) \wedge p(c, s2) \wedge v(2) \Rightarrow m(a, c, 2)$.

この理由付けの結果，ATMS はデータ $m(a, c, 2)$ に対応する ATMS ノードのラベルを計算するが，これは前件に対応する ATMS ノードのラベルと等しい．データ $m(a, c, 2)$ はキューの末尾に連結される．同様に，Fig. 5 に示す 2 入力ノード⑨の WM に保存されている他の中間的なデータとトークン $v(2)$ との処理

および$v(3)$がトークンとしてRootノードから流れた場合の処理が行われる．

SCSでは，データ$m(a,c,2)$のラベルを計算するのに式(3)で示したように集合演算∪が2回とられた．APRICOT/0でも一つのデータ$m(a,c,2)$のラベル計算に着目すれば式(6)，(7)に示すように集合演算∪が計2回である．このことは，Fig. 5に示す2入力ノード⑨のWMに保存されている他の中間的なデータとトークン$v(2)$との処理においても同様である．

一方，トークン$v(3)$がRootノードから流れた場合の処理については，例えば$m(a,c,3)$のラベルを計算するときには集合演算∪が下式で示す1回でよい．

$$\{\Gamma_{p(a,s1)},\ \Gamma_{p(c,n2)}\}\cup\{\Gamma_{v(3)}\}.$$

これは，すでに中間的なデータに対する理由付け(4)により，そのラベル$\{\{\Gamma_{p(a,s1)},\ \Gamma_{p(c,s2)}\}\}$が計算され保存されているからである．同様に，Fig. 5に示す2入力ノード⑨のWMに保存されている他の中間的なデータとトークン$v(3)$との処理において，APRICOT/0のほうが集合演算∪の回数が1回ずつ少なくなる（計4回）．

また，$q(e)$および$q(w)$がトークンとして流れ，1入力ノード④でマッチングに成功し，そのWMに保存される．トークン$q(w)$のほうは前記と同様にOUTの状態になる．トークン$q(e)$のほうは，その1入力ノードの後続ノード⑪にパスされる．2入力ノード⑪の一方の先行ノード⑨のWMにはFig. 5に示すデータが保存されている．まず，2入力ノード⑨のWMの要素$p(a,s1)\wedge p(c,s2)$について，以下の中間的な理由付けがATMSに送られる．

$$p(a,s1)\wedge p(c,s2),\ q(e)\Rightarrow$$
$$p(a,s1)\wedge p(c,s2)\wedge q(e).$$

この理由付けを受けたATMSは，後件に対応するATMSノードのラベルを求めるために，以下の式を計算する．

$$\{\Gamma_{p(a,s1)},\ \Gamma_{p(c,s2)}\}\cup\{\Gamma_{p(e)}\}. \tag{8}$$

トークン$p(a,s1)\wedge p(c,s2)\wedge q(e)$は，2入力ノード⑪の後続ノードにパスされる．2入力ノード⑪は，終端ノード⑮を持つので，以下の理由付けがATMSに送られ，新しいデータ$m(a,c,e)$が導かれキューの末尾に連結される．

$$p(a,s1)\wedge p(c,s2)\wedge q(e)\Rightarrow m(a,c,e).$$

同様に，$m(b,c,e)$，$m(a,d,e)$，$m(b,d,e)$も導かれキューの末尾に連結される．

SCSでは，データ$m(a,c,e)$のラベルを計算するのに式(3)で示したのと同様に集合演算∪が2回とられる．APRICOT/0は，集合演算∪が式(8)の1回でよい．同様に，$m(b,c,e)$等のラベルを求めるためには集合演算∪が1回で済む．この場合，SCSよりAPRICOT/0のほうが集合演算∪の回数が計4回少なくなる．

現在，キューには述語記号mの基礎アトムが存在する．これらをトークンとして流すが，すべての1入力ノードでマッチしない．よって，キューが空となり推論エンジンは停止する．

5. 評　　価

5・1 ラベル計算量の比較

本方法の効果について述べる．ある論理式αとマッチするデータに対応するすべてのATMSノードのラベルの環境数の総和を論理式αの環境数と呼び，$N(\alpha)$で表す．

以下の節がFの要素であるとする．

$$\alpha_1,\cdots,\alpha_{n-1},\alpha_n\rightarrow\beta. \tag{9}$$

βとマッチするデータに対応するすべてのATMSノードのラベルを求めるためのSCS上で必要な集合演算∪の回数をNs，APRICOT/0上で必要なその回数をNaとすると，それぞれ以下のようになる．

$$Ns=(n-1)\prod_{i=1}^{n}N(\alpha_i).$$

$$Na=\sum_{i=2}^{n}\{N(\bigwedge_{j=1}^{i-1}\alpha_j)\cdot N(\alpha_i)\}.$$

ここに，$1\leqq j\leqq n$として，

$$N(\bigwedge_{i=1}^{j}\alpha_i)\leqq\prod_{i=1}^{j}N(\alpha_i). \tag{10}$$

である．不等号であるのは，左辺は右辺から矛盾環境を取り除いた数になるからである．いま，等号の場合についてNaとNsとの比をとると以下のようになる．

$$Na/Ns=$$
$$[1/\{N(\alpha_3)\cdot N(\alpha_4)\cdot\cdots\cdot N(\alpha_n)\}+$$
$$1/\{N(\alpha_4)\cdot N(\alpha_5)\cdot\cdots\cdot N(\alpha_n)\}+$$
$$\cdots+1/N(\alpha_n)+1]/(n-1). \tag{11}$$

これは，$\prod_i N(\alpha_i)$個の環境のうち，矛盾環境がないと仮定したものであり，矛盾環境が存在する場合には本方法による効果はより大きい．式(11)によって，APRICOT/0上での集合演算∪の回数がSCS上での集合演算∪の回数よりも小さくなるのは，$i=3,4,\cdots,n$の任意のiについて，$N(\alpha_i)\geqq 2$であるときである．つまり，2入力ノードがありかつその2入力ノードの後続ノードに1入力ノードから複数のトークンまたは対応するATMSノードのラベルの要素数が複数であるトークンが流れ込む場合にラベル計算コストが減少する．

また，γおよびβ'をアトムとして，以下の節もFの要

素であるものとする．

$$\alpha_1,\cdots,\alpha_{n-1},\gamma\rightarrow\beta'. \quad (12)$$

いま，節(9)によるβとマッチするデータに対応するすべてのATMSノードのラベル計算が終了しているものとする．β'とマッチするデータに対応するすべてのATMSノードのラベルを求めるためのSCS上で必要な集合演算$\cup$の回数Ns'およびAPRICOT/0上で必要なその回数Na'はそれぞれ以下のようになる．

$$Ns'=(n-1)\cdot\{\prod_{i=1}^{n-1}N(\alpha_i)\}\cdot N(\gamma).$$

$$Na'=N(\bigwedge_{j=1}^{n-1}\alpha_j)\cdot N(\gamma).$$

ここでも式(10)で両辺が等しいとして，Na'とNs'との比をとると以下のようになる．

$$Na'/Ns'=1/(n-1).$$

これにより，APRICOT/0上での集合演算$\cup$の回数がSCS上での集合演算$\cup$の回数よりも小さくなるのは，nが2を越える場合である．つまり，2入力ノードがあり，かつその2入力ノードが前件のアトムの数が3以上の複数の節に共通である場合にラベル計算コストが減少する．

これらの考察は，デフォルトについても同様である．

5・2 推論時間に関する比較実験

本節では，APRICOT/0とSCSの総推論時間(パターンマッチングやラベル計算等の仮説推論に必要なすべての時間を含む)の比較を実験により示す．

Fig. 8は知識ベースの規模に応じた本方法の効果を評価するためのテスト知識ベースである．会議の参加者を10名，空き会議室の数をkとして，kをパラメータとして各システム上で総推論時間を測定する．

実験結果をFig. 9に示す．この比較実験により，APRICOT/0はSCSに比べ，知識ベースの規模によらず，約7倍単位知識当りの推論時間が短いことが示された．

また，テスト知識ベースにおいては，Dの要素はすべて前提が空のデフォルトであるので，Dの一つ要素に対応する理由付けに伴うラベル計算コストは$O(1)$である．Dの要素数が$O(k)$であるので，この理由付けの数も$O(k)$である．また，Fのすべての節は，前件のアトムの個数がkに依存しない．さらに，その前件のアトムとマッチする基礎アトムに対応するATMSノードのラベルの要素数はすべて1である．このことより，Fの一つの要素に対応する理由付けに伴うラベル計算コストも$O(1)$になる．Fの要素数が$O(k)$であるので，この理由付けの数は$O(k)$である．APRICOT/0(この問題では，Rete-likeネットワークの1入力ノードの数が$O(k)$となる)もSCSも$O(k)$のパターンマッチングを経て一つの理由付けを生成する．

以上のことおよびFig. 9が示すように，テスト知識ベースによって表現された問題の解決は両者とも$O(k^2)$であることがわかる(図の縦軸は総時間をkで割った時間である)．通常，仮説集合の要素数をkとす

```
                        %% F %%
p (1) ∧p (2) ∧…∧p (10) ∧v (1)  → m (1, 2, ・・・, 10, 1) .
                                .
                                .
                                .
p (1) ∧p (2) ∧…∧p (10) ∧v (i)  → m (1, 2, ・・・, 10, i) .
                                .
                                .
                                .
p (1) ∧p (2) ∧…∧p (10) ∧v (k)  → m (1, 2, ・・・, 10, k) .

                        %% D %%
                      : p (1) ／p (1) .
                      : p (2) ／p (2) .
                                .
                                .
                                .
                     : p (10) ／p (10) .

                      : v (1) ／v (1) .
                                .
                                .
                                .
                      : v (i) ／v (i) .
                                .
                                .
                                .
                      : v (k) ／v (k) .

%% i = 1, 2, ・・・, k.
```

Fig. 8 Tested knowledge base.

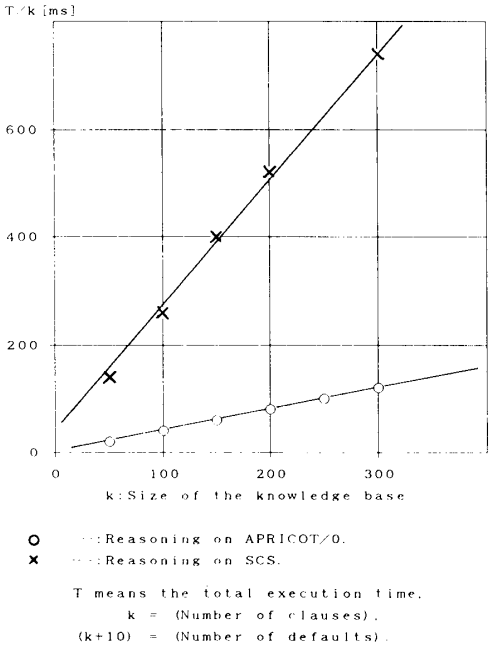
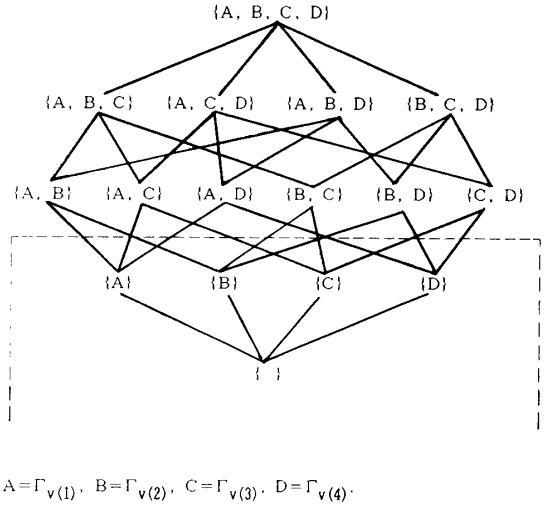


Fig. 9 Evaluation of APRICOT/0.

ると最悪の場合，$O(2^k)$個の環境が存在する．すなわち，指数オーダの探索空間が存在する．ATMSは，矛盾環境によって，その探索空間の枝刈りを効率的に行う．本前向き仮説推論システムにおいては，さらに無矛盾な環境であってもATMSが全く扱わない環境が存在する．例えば，この問題においては，空き会議室であるという仮説どうしを組み合わせるような理由付けは前向き推論エンジンは与えない．Fig. 10に示す仮説$\Gamma_{v(1)}$から$\Gamma_{v(4)}$までを考えた環境束（図では，仮説$\Gamma_{p(1)}$から$\Gamma_{p(10)}$に関しては示していない）においては，矛盾環境の数が0であるにもかかわらず（16個の無矛盾な環境が存在する），ATMSが扱う環境は5個でよい．



$A=\Gamma_{v(1)}$, $B=\Gamma_{v(2)}$, $C=\Gamma_{v(3)}$, $D=\Gamma_{v(4)}$.

Fig. 10 Part of the environment lattice.

6. むすび

前向き推論エンジンとATMSとを結合した仮説推論システムにおいて，効率的な推論方式について述べた．本仮説推論方式は，Rete-likeネットワークの2入力ノードにおいて生成される中間的なデータに対する理由付けを推論エンジンがATMSに送り，ATMSによって計算されるその中間的なデータに対応するATMSノードのラベルを保存することにより，効率的な仮説推論を実現している．本方法による高速化の効果は，与えられた知識ベースをRete-likeネットワークにコンパイルしたとき，2入力ノードがありかつその2入力ノードの後続ノードに1入力ノードから複数のトークンまたは対応するATMSノードのラベルの要素数が複数であるトークンが流れ込む場合，またはその2入力が前件のアトムの数が3以上の節または前提のアトムの数が3以上のデフォルトに共通である場合に有効である．

この方式を用いて，仮説推論システムAPRICOT/0をPSI-II上にインプリメントした．また，同じマシン上にこの推論方式を採用しない仮説推論システムSCSをインプリメントし，推論実行速度について評価を行った．その結果によれば，APRICOT/0はSCSに比べ，知識ベースの規模によらず，単位知識当りの処理時間について推論速度が定数倍速いことがわかった．

また，仮説推論システムAPRICOT/0上に論理回路合成問題[14]に対する知識ベースをインプリメントし，本方式の推論速度に関する有効性を確認している[15]．なお，大規模な知識ベースシステムの開発で，開発完了までに何度も知識ベースの修正を繰り返すような場合，知識ベースからRete-likeネットワークへのインクリメンタルコンパイル法[11][16]もまたすでに提案されている．

本前向き仮説推論の方法は，複数の解が存在し全解を要求する問題の解決には極めて効率的である．しかしこの方法は，ある観測を成り立たせる仮説部分集合のみを求めたい問題においては，その解決に無関係な空間を探索する可能性がある．この点については，マジックセット法[17]等を用いることにより解決されると考えられる．

なお，各知識を共有化できないような知識ベースに基づく仮説推論の高速化が課題としてあげられる．このとき，特に疎結合並列マシン上の仮説推論システムを考えていきたい．

謝　辞

本研究の機会を与えて下さり，常に御指導いただいているICOT渕一博所長，第五研究室 長谷川隆三室長に深く感謝致します．また，御指導いただいたNTT藤井裕一氏，ならびにNTTデータ生駒憲治氏，本稿をまとめるにあたり多くの貴重なコメントをいただいたICOT古川康一次長，論理回路合成問題への適用実験に御協力いただいたJIPDEC大崎宏氏ならびに中島誠氏に深く感謝致します．さらに，討論していただいたICOT研究員の方々に感謝します．

◇参考文献◇

(1) 石塚　満：不完全な知識の操作による次世代知識ベース・システムへのアプローチ，人工知能学会誌，Vol. 3, No. 5, pp. 22-32 (1988).

(2) Inoue, K.：Problem Solving with Hypothetical Reasoning, *Proc. of the Inte'l Conf.* on *Fifth Generation Computer Systems*, Vol. 3, pp. 1275-1281 (1988).

(3) de Kleer, J.：An Assumption-based TMS, *Artif. Intell.*, Vol. 28, pp. 127-162 (1986).

(4) de Kleer, J.：Problem Solving with the ATMS, *Artif. Intell.*, Vol. 28, pp. 197-224 (1986).

(5) 飯島泰裕，成田良一，吉田裕之，泉　寛幸：仮説ネットワークを用いた仮説推論器，人工知能学会研究会資料 SIG-FAI-8701-2, pp. 7-14 (1987).

(6) 飛鳥井正道，森沢秀一，村田真人，浅野俊昭：エキスパートシステム構築ツール CHORUS (2) 仮説推論機能，情報処理学会第 37 回全国大会講演論文集，Vol. 2, pp. 1218-1219 (1988).

(7) Forgy, C. L.：Rete：A Fast Algorithm for the Many Pattern／Many Object Pattern Match Problem, *Artif. Intell.*, Vol. 19, pp. 17-37 (1982).

(8) Flann, N. S., Dietterich, T. G. and Corpron, D. R.：Forward Chaining Logic Programming with the ATMS, *Proc. of AAAI-87*, pp. 24-29 (1987).

(9) Junker, U.：Reasoning in Multiple Contexts, Working Paper, No. 334, GMD (1988).

(10) Reiter, R.：A Logic for Default Reasoning, *Artif. Intell.*, Vol. 13, pp. 81-132 (1980).

(11) 井上克巳，太田好彦：仮説推論システム APRICOT/0による知識コンパイル，人工知能学会研究会資料 SIG-KBS-8805-6, pp. 51-60 (1989).

(12) Nakashima, H. and Nakajima, K.：Hardware Architecture of the Sequential Inference Machine：PSI-II, *Proc. of the Symposium on Logic Programming*, pp. 104-113 (1987).

(13) Chikayama, T.：Unique Features of ESP, *Proc. of the Inte'l Conf. on Fifth Generation Computer Systems*, pp. 292-298 (1984).

(14) Maruyama, F., Kakuda, T., Masunaga, Y., Minoda, Y., Sawada, S. and Kawato, N.：co-LODEX：A Cooperative Expert System for Logic Design, *Proc.* of the *Inte'l Conf. on Fifth Generation Computer Systems*, Vol. 3, pp. 1299-1306 (1988).

(15) Ohta, Y. and Inoue, K.：A Forward-Chaining Multiple-Context Reasoner and Its Application to Logic Design, *ICOT Technical Report*, TR-571, ICOT (1990).

(16) 太田好彦，井上克巳：ATMS による Rete Like ネットワークの逐次構築，情報処理学会第 38 回全国大会講演論文集，Vol. 1, pp. 420-421 (1989).

(17) Bancilhon, F., Maier, D., Sagiv, Y. and Ullman, J. D.：Magic Sets and Other Strange Ways to Implement Logic Programs, *Proc. ACM PODS*, pp. 1-15 (1986).

〔担当編集委員・査読者：石塚　満〕

著者紹介

太田　好彦（正会員）

1983 年千葉大学工学部電子工学科卒業．1985 年同大大学院工学研究科修士課程修了．同年三菱電機（株）入社．1988 年 7 月より（財）新世代コンピュータ技術開発機構に出向．現在，同研究所第五研究室所属．知識システムの研究開発に従事．特に，仮説推論に興味を持つ．情報処理学会，電子情報通信学会各会員．

井上　克巳（正会員）

1982 年京都大学工学部数理工学科卒業．1984 年京都大学大学院工学研究科数理工学専攻修士課程修了．同年，松下電器産業（株）入社．同社情報通信東京研究所に所属．1986 年 10 月より（財）新世代コンピュータ技術開発機構に出向．現在，同研究所第五研究室所属．推論や探索などの人工知能の研究に従事．情報処理学会会員．