RETEアルゴリズム利用による高速無矛盾性検査機構を備えた仮説推論システム

6C－2

門脇修司　牧野俊朗　石塚　満
東京大学生産技術研究所

1．はじめに

仮説推論システムは多くの特徴を持つが，推論スピードの遅さがネックとなっており，実用化するためには，推論速度の向上が必要とされている．

ここでは，無矛盾性の検査にRETEアルゴリズムを利用して推論速度の向上を図った仮説推論システムについて，その概要を紹介する．

2．論理に基づく仮説推論 (1) (2)

論理に基づく仮説推論は，知識として対象世界で常に成り立つ知識の集まりである事実の知識の集合Fと，対象世界で常に成り立つとは限らない知識の集まりである仮説の知識の集合Hを用意し，証明すべき対象である観測事実Oが与えられたときに，

$H \supseteq h$　　hはHの部分集合
$F \cup h \vdash O$　FとhよりOが証明できる
$F \cup h$ は無矛盾

を満たすような仮説集合hを求めることを目的とする．

仮説集合hは，仮説の知識の集合の中から知識を組み合わせて仮のh'を作成し，$F \cup h'$ から観測Oが証明できるかどうかの検査と，$F \cup h'$ が無矛盾であるかどうかの検査を行ない，両方を満足するものをhとすることにより求められる．Prologをベースにしたシステムでは，h'が変更される毎に無矛盾性の検査を行ない，inconsistent述語が $F \cup h'$ から証明できれば矛盾，できなければ無矛盾としている．

3．RETEアルゴリズム (3) (4) (5)

RETEアルゴリズムは，プロダクションシステム記述言語であるOPSの中で最初に使用された高速のマッチングアルゴリズムである．その有効性を認められて，その後の多くのプロダクションシステムに採用されている．RETEアルゴリズムの特徴として，次のことが挙げられる．

・ネットワークを利用して，この中に照合ステップの中間結果をすべて保存する
・ワーキングメモリの変更内容だけをトークンとしてネットワークに流し，関係するノードだけを通ってテストされる
・ネットワーク内のノードは出来る限り共用され，ネットワークのサイズを小さくしている

4．RETEアルゴリズムを利用した無矛盾性検査機構

仮説推論では推論スピードの遅さが実用化する上での大きな問題となっていた．推論の方法を考えると，inconsistent述語の証明（Consistencyのチェック）は実行回数が多く，推論時間の多くを占めている．この検査にかかる処理時間を速めることは，推論時間全体の短縮につながる．

このConsistencyのチェックは，毎回毎回少しずつ変化するサブセット(h')と事実の知識の集合から，inconsistent述語を証明できるかどうかを調べている．

本システムは，サブセットの変更内容をトークンとし，inconsistent述語の右辺からネットワークを作ることで，RETEアルゴリズムを利用する．

こうすることで，前回までの証明の経過をネットワーク内に保存し，トークンとしてネットワークに入ってきた仮説に関係するテストを実行するだけですむ．毎回inconsistent述語の証明を導出木の根から始めることに比べると，処理速度が大幅に向上する．

本システムは仮説推論機構をSICStus Prolog上のメタインタプリタの形で記述し，無矛盾性の検査機構はCのプログラムとして作成し，Prologの述語として呼び出せるようにしている．図1に判定機構の構成を示し，その構成要素について次に説明する．

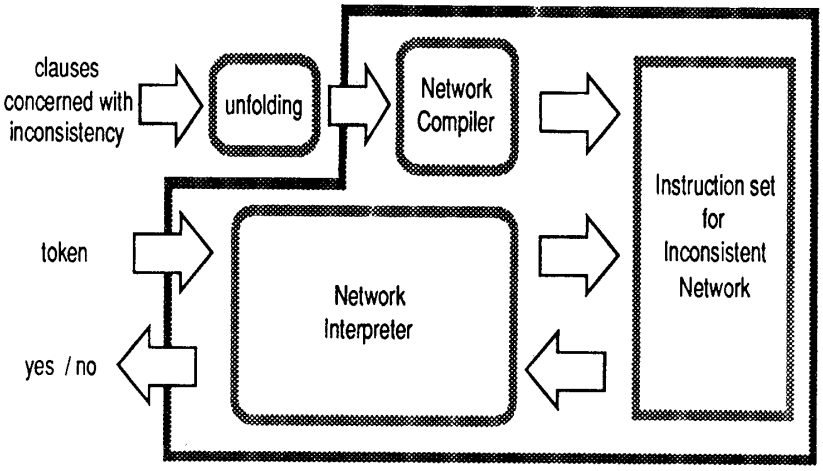

図1　無矛盾性判定機構の構成

Hypothetical reasoning system with efficient consistency check mechanism using Rete algorithm
Shuji Kadowaki, Toshiro Makino, Mitsuru Ishizuka
Institute of Industiral Science, University of Tokyo

(1) トークン

トークンは推論機構から判定機構に渡される入力データであり，またノードからノードへ渡されるデータでもある．トークンは追加／削除を表す符号　(tag: 追加なら＋，削除なら－）と，h'に追加/削除された仮説の知識（要素: element）から成る．

後述するタイプBノードから出力されるトークンは，一つの符号と複数の要素からなる．

(2) ノード

ネットワークは無矛盾性の検査をするために様々なテストを行なうが，ネットワークはそれ自身一つのテストしかしないノードを組み合わせて，作られている．

ノードはそのテスト内容によってタイプAノードとタイプBノードに分けられる．他に，特別なノードとして，ORノード，ANDノード，ルートノード，終端ノードがある．

(a) タイプAノード

このノードは，一つの要素についてその特徴をテストするもので，要素内テストノードと呼ばれ，

・Spec (name / arity)
・N番目の引数とM番目の引数との関係
・N番目の引数と定数との関係

をテストする．このノードは一つの入力と一つ以上の出力を持ち，テストに成功したトークンだけ出力する．

(b) タイプBノード

二つの要素又はトークンの間のテストをするものであり，要素間テストノードと呼ばれ，

・両者の引数の関係
・両者が同じものかどうか

をテストする．このノードには，入力が一つのもの（タイプB1ノード）と，二つのもの（タイプB2ノード）とがあり，出力はどちらも一つ以上ある．

タイプB1ノードは，そのノードに内部メモリを持ち，入力されたトークンを保存する．入力されたトークンと内部メモリ内のトークンとの間のテストを行ない，成功した組を要素として新しくトークンを作り，出力する．

タイプB2ノードは，左から来たトークンを保存する左メモリと，右から来たトークンを保存する右メモリを持つ．左から来たトークンは右メモリと，右から来たトークンは左メモリとの間でテストを行ない，成功した組を要素としたトークンを新しく作り，出力する．

(c) 特殊ノード

ORノードは入力されたトークンを，無条件で出力する．

ANDノードはその全ての入力にトークンが到着したときに，それらのトークンを要素として新しくトークンを作り出力する．

ルートノードはネットへの入り口であり，トークンを受け取り該当するその下の一つのノードに渡す．

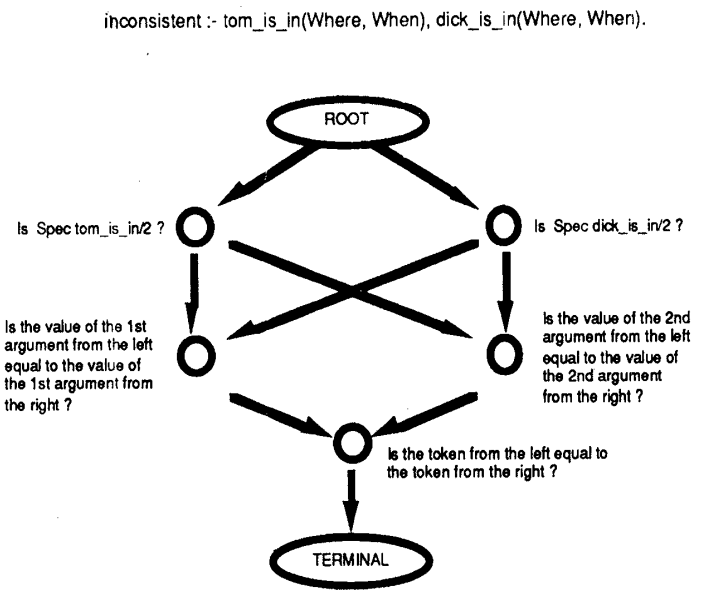

図2　inconsistent 述語とそのネットワークの例

終端ノードは，１入力ノードであり，ネットワークの終わりを示し，トークンが入力されたときにその内容に関係なく，「矛盾あり」と通知させるためのノードである．

(3) ネットワークコンパイラ

コンパイラは，unfolding された inconsistent 述語から，上で述べたようなノードを結合してネットワークを作る．図２に，inconsistent 述語と，それから作られたネットワークの例を示す．

(4) ネットワークの表現

ネットワークは RETE アルゴリズムと同じように，アセンブラに似た言語表現を使って記述され，メモリ上にはそれぞれの命令が数バイトのコードで格納される．メモリ上に格納されたネットワークは，ネットワークインタプリタによって解釈，実行される．

４．おわりに

RETE アルゴリズムを仮説推論システムの無矛盾性の判定に応用する方法についてその概要を述べた．

現在，システムのインプリメントを進めている．

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, Vol.3, No.5, pp.552-562 (1988)
2. 牧野, 石塚:仮説推論による機能ブロック図設計システム, 情報処理学会第37回全国大会講演論文集, 2G-5, pp.1158-1159 (1988)
3. Charles L. Forgy : Rete : A Fast Algorithm for the Many Pattern / Many Object Pattern Match Problem, Artificial Intelligence 19 (1982)
4. 石田亨:プロダクションシステムと並列処理, 情報処理, Vol. 26, No.12 (1985)
5. 伊藤秀昭:OPS5の高速化, Computer Today, No.13, サイエンス社 (1986)