6C-1 Lattice構造を使った並列横型解法による仮説推論システムの高速化

近藤朗子　牧野俊朗　石塚満
東京大学

1. はじめに

不完全な知識を取り扱うことにより、高次人工知能機能を実現する仮説推論[1]は、次世代知識ベースシステムを構築する上で重要な技術となっている。しかし、仮説推論システムは、知識ベースに不完全な知識を含むため、常に無矛盾性の管理をする必要があり、推論の速度がかなり遅くなる。そのため、現在仮説推論では、推論速度の向上が重要なテーマの1つとなっている。

今回我々は推論の高速化の一手法として、ATMS[2]の lattice 構造を利用して、無矛盾性を管理しつつ全解を並列に求める方法について検討し、システムを作成した。この方法を用いると、無矛盾・完全・健全・最小の解を高速に求めることができる。以下に、そのシステムについて報告する。

2. 仮説推論システム

まず最初に、仮説推論システムについて、簡単に記しておく。

知識ベースを、完全な知識（対象世界で常に成り立つ知識、事実）の集合 F と、不完全な知識（対象世界で常に成り立つとは限らない知識、仮説）の集合 H に分ける。仮説推論システムの基本動作は、ある観測 O が与えられた時、

$h \subseteq H$
$F \cup h \vdash O$
$F \cup h \not\vdash \square$

であるような H の部分集合 h を求めることである。（図1参照）

このとき、求められた部分集合 h が、完全知識の集合 F と無矛盾でなければならないので、generate & test を繰り返し、推論を行うことになる。

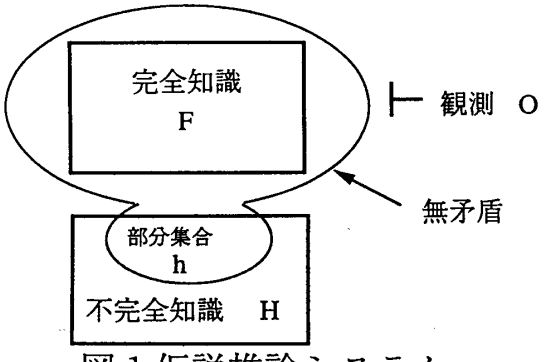

図1仮説推論システム

3. lattice 構造を使った仮説推論の並列横型解法

従来の深さ優先縦型解法の仮説推論システムでは、generate & test により推論の過程で矛盾が生じた場合、あるいは適切な仮説がない場合、その時点でバックトラックを行い、次の候補を捜している。また、全解を求める場合は、さらにバックトラックを繰り返し解を求めることになる。これらのバックトラックによって、かなり推論時間がかかっていた。

ところが、lattice 構造を使った並列横型解法の仮説推論システムでは、各サブゴールが成立するのに必要な無矛盾の仮説セットを維持しつつ推論を行うため、バックトラックは生じない。そのため、特に全解探索に対して高速化が期待できる。

この lattice 構造を使った仮説推論の並列横型解法は、次の3つの部分よりなる。

1) 観測をサブゴールに展開(unfold)する。
2) 各サブゴールに対する仮説セットを Environment lattice 上にマッピングする。
3) 最終的な解を求める。

以下に、それぞれについて説明する。

1) 観測をサブゴールに展開(unfold)する。

ATMS は前向き推論のプロダクションシステムと組み合わせて使用され、複数の環境を保持しながら推論が進む。本仮説推論システムは、論理に基づくゴール指向の後向き推論的な形式をとるので、仮説の lattice を利用するにはゴール（観測）の展開（unfold）を必要とする。

ここでは、観測を完全な知識のみで証明できるところまで選言標準形に展開し、証明できない部分はサブゴールとして残しておく。（サブゴールは、不完全な知識を含む部分ということになる。）

たとえば、g を観測、a, b, c を不完全な知識（仮説）として、

```
g :- g1 ∧ g2.
g1 :- g11.
g2 :- g21.
g2 :- g22.
g11 :- a.
g21 :- b.
g22 :- c.
```

という知識ベースがあった場合、g11, g21, g22 は不完全知識を含むので、それぞれはサブゴールとして残しておいて、

```
g :- (g11 ∧ g21)∨(g11 ∧ g22).
```

と展開する。

An Efficient Inference for Hypothetical Reasoning System by Using Parallel Solving Technique with Environment Lattice
Akiko KONDO, Toshiro MAKINO, Mitsuru ISHIZUKA
Univ. of Tokyo

この展開は Plolog の推論メカニズムを利用することにより、比較的容易に実現できる。

2) 各サブゴールに対する仮説セットを Environment lattice 上にマッピングする。

Environment lattice 上で、各サブゴールが成立するすべてのノードについてフラグを立てる。（これをマッピングと呼ぶ。）

たとえば、仮説を a,b,c,d として、サブゴール g1 に関し、

```
g1 :- a.
g1 :- b.
g1 :- c.
```

という知識があり、さらに、b ∧ c で矛盾が発生するという知識、

```
nogood(b,c).
```

がある場合、サブゴール g1 が成立する環境ノードは（nogood を考慮して）、

{ [a], [b], [c], [a,b], [a,c], [a,d], [b,d], [c,d], [a,b,d], [a,c,d] }

である。

そこで、サブゴール g1 の Environment lattice へのマッピング結果は、図2のようになる。

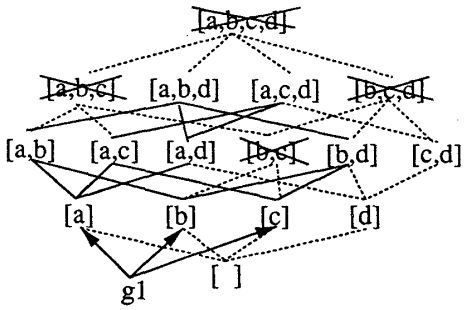

図2 サブゴール g1 の Environment lattice へのマッピング

3) 最終的な解を求める。

2) で求めた各サブゴールのマッピング結果を 1) の展開した結果に適用して、最終ゴールの 解を求める。

具体的には、連言関係があれば、それらのサブゴールのマッピング結果が重なっているところを解とする。その後、すべての解から冗長なものを省いたものが最終的な解となる。

4. インプリメント時における問題点とその解決策

Environment lattice は、n 個の仮説がある場合、2^n 個のノードが生じる。それをすべて記述することは、仮説の数が増えると現実問題としては不可能になってくる。

そこで、我々は Prolog 上でのインプリメントの際に次のような方法をとった。

① 各仮説をビットで表す。

例）仮説 a,b,c,d がある場合、それぞれの仮説の対応ビットを下記のように

```
□□□□  ･･･ ビット
↕↕↕↕      ↕
d c b a  ･･･ 仮説
```

と定義した場合、各ノード（環境）は、

```
[a,b] -> 0011
[b,d] -> 1010
etc.
```

となる。

② 推論の際、lattice 上に表された必要なノードのうち、最小のものだけ（ minimul set ）を保持しているようにする。

実際、lattice の構造（図2参照）から明らかなように、必要なノードの minimul set はその上位の必要なノードすべてを暗に含んでいるといえる。

さらに、①のように各仮説をビットで表したので、どのノードがどのノードの super set になっているかということは、ビット演算により容易に計算できる。

例）あるノード A,B に関して、

A ∧ B = A

ならば A は B の super set である。

以上の点に従ってインプリメントを行うと、実行時、各サブゴールが常に無矛盾・完全・健全・最小の解（ノード）を維持しながら推論を進めることになる。

5. まとめ

仮説推論のネックとなっている推論速度を向上させる一手法として、本稿では lattice 構造を使った並列横型解法について報告した。また、そのシステムは Prolog によりインプリメントした。

実際にいくつかの例で推論速度の比較を行ったところ、特にバックトラックが多く、従来の深さ優先縦型解法ではかなりの推論時間がかかったものについては、推論速度が飛躍的に向上した。

今後は、仮説の数がさらに多くなった場合の対応と、ハッシュテーブルの利用による高速化について検討していくつもりである。

＜参考文献＞

[1] 石塚：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.552-562 (1988)

[2] de Kleer.J. : An Assumption-based TMS, Artificial Intelligence, 28, pp.127-162 (1986)