人　工　知　能　104－4
（1996．5．14）

推論パスネットワークによる仮説合成時の無矛盾性チェックの改善案

野畠英明　木村春彦　広瀬貞樹　広林茂樹
金沢大学工学部電気・情報工学科人工知能研究室
〒920 石川県金沢市小立野 2-40-20

仮説推論は，不完全な知識を扱うことのできる有用な枠組であるが，低い推論速度が大きな問題として挙げられる．この推論速度の問題を克服するアプローチは，大きく分けて2通り考えられる．1つは準最適解を求める問題と捉え高速化を図るアプローチ，もう1つはあくまで最適解を求める問題として高速化を図るアプローチである．本稿では，後者の最適解を求める方法において有名な方法である推論パスネットワークによる高速仮説推論システムを基に無矛盾チェックを軽減するアルゴリズムを提案し，実験によりその効果を示す．

Improvement of the Consistency Check on Hypothetical Synthesis using Inference-Path Network

Hideaki Nobata, Haruhiko Kimura, Sadaki Hirose and Shigeki Hirobayashi

Faculty of Technology, Kanazawa University.
2-40-20, Kodatsuno, Kanazawa, Ishikawa, 920 Japan

A logic-based hypothetical reasoning can handle incomplete knowledge as a hypothesis. But the problem is that the hypothetical reasoning has a fault of low reasoning speed. We think that there are two approaches to resolve this problem. The one is an approach for reducing the synthesis process that find the sub-optimal solutions. The other is an approach that exactly find the optimal solutions. In this paper, we propose the algorithm for reducing the consistency check based on the fast hypothetical reasoning system that is famous for the latter appoach. And we prove that proposed algorithm is effective for hypothetical reasoning systems by experiment using real knowledge bases.

1 はじめに

仮説推論は, 不完全な知識を扱うことのできる有用な枠組であるが, 低い推論速度が大きな問題として挙げられる. この推論速度の問題を克服するアプローチは, 大きく分けて 2 通り考えられる. 1 つは準最適解を求める問題と捉え高速化を図るアプローチ, もう 1 つはあくまで最適解を求める問題として高速化を図るアプローチである.

これまでに, 後者のアプローチで推論速度の問題を克服しようとした研究がいくつか発表されているが, その中で有名な方法として推論パスネットワークによる高速仮説推論システム [伊藤 91] がある. このシステムはパス生成フェーズと仮説合成フェーズからなり, パス生成フェーズは線形時間で解ける. しかし, 仮説合成フェーズにおいて, 無矛盾性チェックと他の仮説の組合せに包摂される組合せを排除する最小性チェックの計算コストが高いことが問題として挙げられる.

本稿では, この高速仮説推論システム [伊藤 91] を基に無矛盾チェックを軽減するアルゴリズムを提案し, 実験によりその効果を示す.

2 論理に基づく仮説推論

論理に基づく仮説推論 [Poole87] とは, 他と矛盾の可能性のない常に成立する背景知識 (Σ) と, 他と矛盾の可能性のある仮説知識 (H) が与えられているとき, Σ と合わせてゴール (g) を証明できる H の無矛盾で最小の部分集合である解仮説 (h) を見出すことをいう. つまり, 以下の条件を満たす解仮説 h を見出すことである.

- $h \subseteq H$
- $\Sigma \cup h \vdash g$
- $\Sigma \cup h \not\vdash \square$
- 上記の 3 条件を満たす h'(h'$\subset$h) は存在しない

具体的な用途としては, システムの各部品の故障状態など可能な状態を各要素仮説とし, 故障の観測事象をゴールとすれば, 仮説推論により故障診断が可能となる. また, 設計に使用可能な部品や部品間の可能な接続法を要素仮説とし, 満たすべき設計の仕様をゴールとすることによって, 設計システムが可能となる.

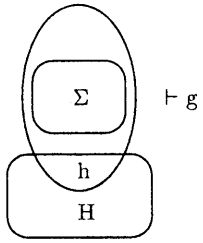

図 1: 仮説推論

しかし, 非単調推論の一種である仮説推論の最大の課題は低い推論速度であり, 推論速度を向上させることが重要な問題となっている. 実際, 命題論理の仮説推論の計算でさえ, NP- 完全であることが証明されている [Selman90, Bylander91].

3 推論パスネットワークによる仮説推論システム

推論パスネットワークによる仮説推論システムは, ホーン節命題論理に対する充足可能性判定の線形時間アルゴリズムである Dowling&Gallier の手法 [Dowling84] を基盤にした推論パスネットワーク生成フェーズと, そのネットワークに沿う前向きの仮説合成フェーズからなる.

3.1 パス生成フェーズ

まず, 基盤となる手法 [Dowling84] の説明をする.

この手法で用いるネットワークのノードは各アトムに対応するが, その他に true と false の 2 つのノードがある. リンクは有向で, 節の型に応じて次のように張られる.

1. **ルール型の節:** ボディ部の各アトムに対応するノードそれぞれからヘッド部のアトムに

対応するノードに向けてリンクを張る．リンクにはすべて同じ番号 (節番号) を付ける．

2. **矛盾型の節:** inc をヘッドに持つ節および空節をヘッドに持つ節 (通常，ゴール節と呼ばれる) のボディ部の各アトムに対応するノードそれぞれから false ノードに向けてリンクを張る．これらのリンクにすべて同じ番号を付ける．

3. **ファクト型の節:** true ノードからこの節のアトムに対応するノードに向けてリンクを張る．

各ノードは，真と偽の 2 つの真理値の内，どちらかをとる．初めに true ノードは真に設定されており，他のすべてのノードは偽に設定されている．その上で，同一番号の有向リンクの元のノードがすべて真であるとき，それらのリンクの先のノードを真に変える．新たに真になるノードがなくなるまで，この操作を繰り返す．操作が終わったとき，false ノードが真であれば元の論理式は充足不能である．偽のときは充足可能である．同一のアトムを同じノードに割り当ててネットワーク化する一種のコンパイルによって，同じサブゴールを複数回推論するような無駄を回避できる．

あるアトムのノードが推論できるかどうかは，その調べたいアトムのノードからゴール駆動で後向きに真理値伝搬を行うことによって調べることができる．真理値伝搬によりそのアトムの真理値が真となれば，推論できることを意味する．

高速仮説推論システム [伊藤 91] では，仮説はすべて単一アトム (単節) で，探索木上でリーフ位置にくるようになっている．ルール型の仮説知識は，新たにアトムを導入することによりルール型の背景知識と単一アトムの仮説知識に変換できる．例えば，a :- b. という仮説知識はアトム c を付け加えることによって，以下のように変換できる．

仮説　a :- b

⇓

背景知識　a :- b, c.

仮説　c.

また，仮説は ATMS と同様にビットベクトル表現 (n 仮説のときは n ビットの数値を用い，各仮説をビットの位置で表す) で管理し，仮説合成フェーズでの包含関係のチェックが効率的に行えるようにしている．

パス生成フェーズでは，ネットワーク化 [Dowling84] と，この上での真理値伝搬による推論器を次のように変更して，推論パスネットワークを形成している．まず，真理値は真と偽の 2 値を，“恒真”，“仮説の支持により真”，“恒偽” の 3 値に変更し，各ノードの真理値の初期値を次のように決める．

1. 背景知識集合中のファクト型知識に現れるアトムに対応するノードを “恒真” とする．

2. 仮説に現れるアトムに対応するノードを “仮説の支持により真” とする．

3. その他のノードを “恒偽” とする．

観測をゴールとするゴール駆動の後向き探索により，ネットワークに沿って真理値伝搬を行う．あるノードに入る同一番号のリンクの元ノードがすべて “恒真” のときには，そのノードを “恒真” とする．また，同一番号のリンクの元のノードがすべて “仮説の支持により真” か “恒真” のとき，そのノードを “仮説の支持により真” とする．

真理値伝搬が完了した時点で “恒真” のノードは，常に成立するノードで仮説合成に無関係である．また “恒偽” のノードは，ゴールノードの子孫ではなく，ゴールの証明に無関係なノードである．したがって，ゴールの証明に関与する仮説は，真理値伝搬の際に “仮説の支持により真” のノードだけから生成される．そこで，ノード間のリンクだけを残して仮説合成フェーズで用いるネットワークを切り出す．以上のようにしてネットワークを切り出した結果，あるノードから 1 本のリン

クだけが出て，そのリンクが入るノードにリンクが入ってこないときは，この2つのノードを共通化する．

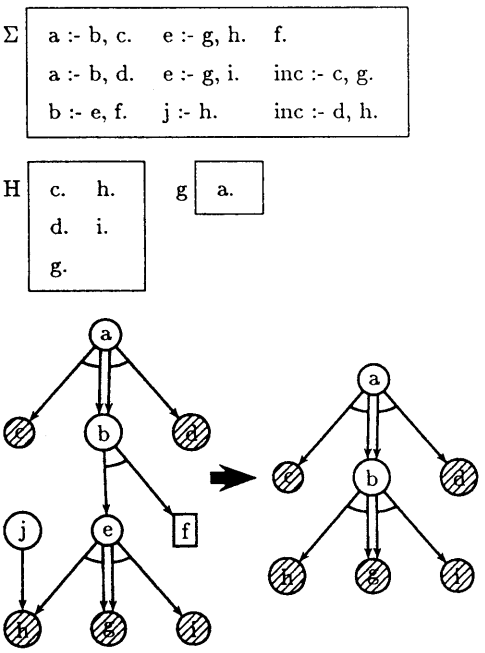

図 2: 推論パスの生成例

パス生成フェーズでの推論パス生成の具体例を図2に示す．この例では，fノードは仮説生成に無関係なノードであるため削除され，jノードはゴールの証明に無関係なノードであるため削除される．その結果，eノードがbノードに共通化される．

以上が推論パス生成フェーズである．これによって次に続く仮説合成フェーズの計算量を小さくすることができる．

3.2 仮説合成フェーズ

パス生成フェーズで生成されたネットワークは仮説合成フェーズに渡される．ネットワークの各ノードは，そのノードを真とするために必要な仮説の集合 (支持仮説) を保持する．初期状態ではすべてのノードは支持仮説を持たない．次に，仮説に対応するノードはその仮説を支持仮説とする．ネットワーク内の各ノードは，そのノードに入ってくる同一番号のリンクの元のノードが，すべて支持仮説を持つとき，その和集合を支持仮説とする．このように仮説を合成しながら新たな仮説を生成する．新しく生成された仮説は，そのノードを推論するために必要な仮説であり，推論パスネットワーク上のノードは，ゴールの証明に必要なもののみであるから，必ずゴールの推論に関係がある．

生成された仮説は，合成時に無矛盾性制約を満足しているかどうかの検査を受ける．この検査はATMSと同様に，ビットベクトル表現された仮説の組合せと矛盾仮説集合の包含関係でチェックする．また，ここで矛盾であるとして取り除かれた仮説は，親ノードへの伝搬が行われない．

無矛盾性制約の検査と同時に，支持仮説集合の中に最小性を満たさない冗長なものがあるかどうかを調べて，冗長な仮説は削除する必要がある．この検査も集合の包含関係でチェックすることができる．

しかし，これまで述べてきた無矛盾性チェックと最小性チェックを単純に行うと，支持仮説内のすべての仮説の組合せを検査しなければならず，非常に大きな計算時間を必要とする．そこで，高速仮説推論システム [伊藤 91] ではステージと冗長マーカという2つの概念を導入し，これら2つのチェックの高速化を図っている．

3.2.1 ステージ

ステージという概念を導入する目的は，仮説合成をいくつかの段階に分割することにより，生成される支持仮説をある程度把握できるようにし，あるノードでの無矛盾性および最小性チェックを簡略化しようというものである．仮説合成フェーズで，ある仮説のアトムに対応するノードの支持仮説にその仮説を加えて仮説合成を始めることを，そのノードを発火させるという．1つのステージSは，そのステージを特徴付ける仮説 h_S に対応するノードを発火させることにより始まる．そして，仮説合成ができなくなるまで合成

を続けてステージSが終了する．1つのステージでは，ただ1つの仮説ノードしか発火しないものとする．このh_Sによって特徴付けられるステージSで生成された支持仮説は，必ず仮説h_Sを含む．

図3は，ステージを用いた仮説合成の例である．図中左の番号は各ステージにつけられたステージ番号である．ステージ番号に続けて各ステージで合成される支持仮説を示した．

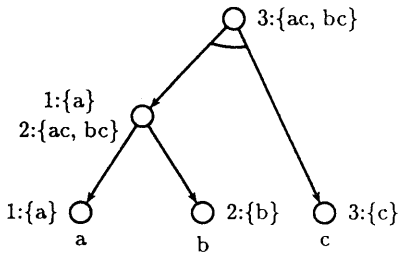

図 3: ステージを用いた仮説合成の例

ステージによる仮説合成では，ステージS以前のステージで生成された支持仮説は仮説h_Sを含むことがない．したがって，それらはステージSで生成された仮説に対して冗長になることはないため，最小性チェックの対象は，現ステージSで生成された仮説だけとなる．

新しく矛盾である仮説が発見されたとき，今までに得られた支持仮説集合から，新しく矛盾であることがわかったものを除かなければならない．あるステージSで矛盾であることがわかった仮説は，そのステージを特徴付ける仮説h_Sを必ず含んでいる．取り除かなければならない支持仮説は，この仮説の上位集合であるので，やはり仮説h_Sを含んでいる．したがって，現在のステージで生成された支持仮説だけを削除対象とすれば良い．

3.2.2 冗長マーカ

冗長マーカについては，最小性チェックのみに関わる部分であり，本稿では説明を省略する．

4 推論パスネットワークによる高速仮説推論システムの問題点

高速仮説推論システム [伊藤 91] の問題点として，

- 無矛盾性チェックの計算コストが大きい．
- 最小性チェックの計算コストが大きい．

の2点が挙げられる．いずれも大きな推論時間を必要とするもので，全体の推論時間に大きな影響を及ぼすものである．

4.1 無矛盾性チェック

仮説推論では，常に成り立つとは限らず，他と矛盾を引き起こす可能性のある知識である仮説を扱うことができる点が最大の特徴であり，演繹推論に対する利点となっている．しかし，生成された仮説が矛盾を引き起こすかどうかをチェックする必要性が生じる．これに対し，高速仮説推論システム [伊藤 91] では，矛盾を引き起こす仮説の生成を防ぎ，チェックの手間を削減するために以下の方法を用いている．

- ビットベクトル表現された仮説を用いて，矛盾仮説集合との包含関係で調べる．
- ステージの導入．

しかし，まだ無矛盾性チェックを行う仮説を絞り込む余地があり，高速化が可能であると考えられる．

本稿では，高速仮説推論システム [伊藤 91] の改善を目指し，仮説合成時の矛盾が起こる必要十分条件を求めて，無矛盾性チェックを高速にするアルゴリズムを提案する．また，実験によりその効果を示す．

4.2 最小性チェック

仮説推論で生成される解仮説には，冗長なものを含まないという最小性を満たしているものでなければならない．これを単純にチェックすると，

すべての仮説の組に対して冗長チェックを行う必要があり，計算コストが大きくなる．高速仮説推論システム [伊藤 91] では，冗長な仮説の生成を防ぎ，チェックの手間を削減するために以下の方法を用いている．

- ビットベクトル表現された仮説を用いて最小性を包含関係で調べる
- 冗長マーカ
- ステージの導入

このような手法により，冗長チェックの計算コストは小さくなっているが，まだ冗長な仮説の生成されないノードでも冗長チェックを行っており，改善の余地がある．

本稿では，この問題に関しての改善は行わない．

5 仮説合成時に矛盾が起こる必要十分条件

本稿で対象とする矛盾は，二項関係の矛盾に限定する．その理由は，二項関係の矛盾が最も一般的であること，また仮説合成時の無矛盾性チェックに矛盾が起こるための必要十分条件の知識を取り込むためには，そのオーバヘッドタイムが小さくなければならないからである．

5.1 定義

以下の条件をすべて満足するノードxをノードa,bの交差点と呼ぶ．

- xはAND節点である．
- ノードa,bはそれぞれ仮説a,bに対応し，inc:-a,bである．
- xの2つの子をy,zとすると，yの子孫にリーフaがあり，zの子孫に葉bがある．ただし，y=a,z=bであってもよい．

5.2 定理

ノードxの仮説合成で矛盾が生じる必要十分条件は，xがinc:-a,bとなるa,bの交差点となることである．

6 提案アルゴリズム

高速仮説推論システム [伊藤 91] では，AND節点で毎回各矛盾とのチェックをしなければならなかった．それに対し，本稿で提案する無矛盾性チェックは，交差点だけでよい．しかも，全ての矛盾をチェックする必要はなく，該当する矛盾のみでよい．以下に，交差点を探し出すアルゴリズムを示し，仮説合成の仕方についてふれる．

定義 inc:-a,b. に対して，ノードa,bの交差点Oからa,bへの有向道$[O, L_1, L_2, \ldots, L_x, a]$，$[O, R_1, R_2, \ldots, R_y, b]$をそれぞれ左有向道，右有向道と呼ぶ

入力 AND/ORグラフで表された推論パスネットワークと2項関係矛盾

出力 各矛盾に対応する交差点

方法

1. 各矛盾inc:-a,b. に対して，2～4の操作を繰り返す．
2. リーフaから先祖に向かって根まで，到達した各ノードにaの印を付ける．ただし，右有向道の任意のR_tがAND節点であるとき，この操作は交差点Oまでよい．
3. リーフbから先祖に向かって根まで，到達した各ノードにbの印を付ける．ただし，左有向道の任意のL_tがAND節点であるとき，この操作は交差点Oまでよい．
4. aとbの印が付加されたAND節点をinc:-a,b. のチェックを必要とするa,bの交差点として登録する．

各矛盾に対応する交差点が求まれば, 高速仮説推論システム [伊藤 91] と同様にステージを用いて仮説を合成し, 交差点に来たら, 該当する矛盾のチェックを行なって矛盾する仮説の組を取り除いていけばよい.

図 4にアルゴリズムの実行例を示す. ただし, 図中の●は交差点である.

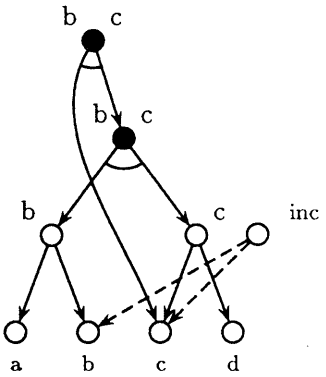

図 4: アルゴリズムの実行例

7 実験

提案する無矛盾性チェックの効果を示すために, 以下の実験を行なった. 比較対象は高速仮説推論システム [伊藤 91] である. パス生成フェーズは同じであり, 仮説合成フェーズのみ異なるので, 仮説合成フェーズの実行時間を比較した.

実験対象には, 故障の可能性のあるゲートの状態を仮説とするデコーダ回路の故障診断を用いた.

測定は SPARCstation-4 上で行い, 推論システムは C++ 言語でインプリメントした.

7.1 実験 1

まず, デコーダ回路として, 入力数 3 で出力数を 5 ～ 8 に変化させた場合 (仮説数 39 ～ 57) の結果を図 5に示す.

従来方式では仮説数の増加に応じて処理時間が増加しているが, 提案方式ではほとんど変わっていない. 十分な効果が表れていると言える. また, 交差点を求めるための手間は, 図 5の範囲では 10^{-1} 秒のオーダであり, 全体の処理時間に対して無視できる時間であった.

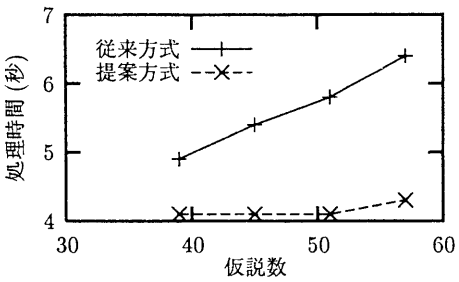

図 5: 実験結果

7.2 実験 2

次に, 表 1に入力数 3 で出力数 8 の場合 (仮説数 57) と入力数 4 で出力数 10 の場合 (仮説数 94) との仮説合成時間の比較を示す.

表 1: 仮説合成の処理時間

仮説数	57	94
従来方式 (秒)	6.4	28533
提案方式 (秒)	4.2	28122

表 1を見ると, 仮説数 57 の場合は提案方式の効果が大きいが, 仮説数 94 の場合は提案方式の効果が小さいことがわかる.

表 2に従来方式による無矛盾性チェックの処理時間と最小性チェックの処理時間の比較を示す.

表 2: 無矛盾性チェックと最小性チェックの比較

仮説数	57	94
無矛盾性チェック (秒)	3.5	118
最小性チェック (秒)	2.8	26056

表 2を見てわかるように, 仮説数 57 の場合には無矛盾性チェックの処理時間が大きな割合を占め

ているのに対し，仮説数94では最小性チェックの処理時間が非常に大きな割合を占めている．

表1の結果は，このことから明らかなように，問題の変化により無矛盾性チェックが仮説合成の処理時間に関して主要な要素ではなくなったことが原因となっている．

8 まとめ

推論パスネットワークによる仮説推論システムでは，各AND節点ですべての矛盾に対して無矛盾性チェックを行うものであった．それに対し，提案した方法はあらかじめ交差点を求め，交差点でのみ無矛盾性チェックをする．しかも該当する矛盾に対してだけである．そのため，問題となるのが削減された無矛盾性チェックの手間と，交差点を求めるオーバヘッドタイムの大小関係である．これはトレードオフの関係にある．

推論パスネットワークでの，ネットワークが大きくなればなるほど，またネットワークの上層部にいけばいくほど1ノード当たりの支持仮説の組の数が多くなる．それも指数関数的に増えていく傾向がある．そのため，仮説数が少ない時には，オーバヘッドタイムの影響力が大きいと考えられるが，仮説数が多くなると上層部にいけばいくほど，1ノード当たりの無矛盾性チェックの手間が大きくなるので，全体として削減される手間の方がオーバヘッドタイムを上回ることが予想される．また，矛盾仮説のペアが多い場合は，交差点を求める手間が増え，オーバヘッドタイムも大きくなると思われる．

実験1の結果で，従来方式での処理時間の増加に対して，提案方式の処理時間の増加が小さいのは，オーバヘッドタイムが十分小さく，削減される無矛盾性チェックの手間が大きいということを示している．

実験2の結果が実験1の結果のようによくないのは，最小性チェックかかる手間の方が無矛盾性チェックの手間よりも非常に大きいためであり，最小性チェックを改善する必要性を示している．

本稿では，推論パスネットワークによる高速仮説推論システムの仮説合成フェーズで，問題となる支持仮説集合の無矛盾性チェックの改善案を示した．今後の課題は包摂処理の改善案を求めることである．

参考文献

[Poole87] Poole,D.,Aleliunas,R. and Goeble,R.: "Defaults and Diagnosis, in The Knowledge Frontier: Essays in the Knowledge Frontier: Essays in the Knowledge Representation", Springer-Verlag,N.Y.(1987).

[Selman90] Selman,B. and Levesque,H.J.: "Abductive and Default Reasoning: A computational Core", Proc. AAAI-90, pp.343-348(1990).

[Bylander91] Bylander,T.,Allemang,D.,Tanner,M.C. and Josephson,J.R.: "The Computational Complexity of Abduction", Artif.Intell., Vol.49, pp.25-60(1991).

[伊藤91] 伊藤 史朗, 石塚 満: "推論パスネットワークによる高速仮説推論システム", 人工知能学会誌, Vol.6, No.4, pp.501-509(1991).

[Dowling84] Dowling,W.F. and Gallier,J.H.: "Linear-time Algorithm for Testing the Satisfiability of Propositional Horn Formulae", J. of Logic Program, Vol. 3, pp.267-284(1984).