ICOT Technical Memorandum: TM-0737

TM-0737

設計型問題向け並列仮説推論器

太田好彦

June, 1989

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03) 456 3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

設計型問題向け並列仮説推論器

A Parallel Hypothetical Reasoner for Design-type Problems

太田 好彦
Yoshihiko OHTA

（財）新世代コンピュータ技術開発機構　第五研究室
Fifth Research Laboratory,
Institute for New Generation Computer Technology

1．はじめに

設計型問題は、与えられた仕様を満足するシステム構造と構成要素特性とを決定する問題である[1]。このような問題を解決する場合、構成要素特性を仮定してそれらを組み合わせて設計を行い要求仕様を満たすか検査をしていく方法がある。しかしながら、それらの組み合わせが与えられた仕様を満足するかどうかは、設計の終盤でないとわからないことが多い。さらに、複数の設計解の評価は、その後でないと行えない。したがって、設計型問題においては、早期に枝刈りを行うことが難しく、診断型問題に比べて探索空間が極めて広い困難な問題とされている。このため、知識システムとして設計型問題を扱う場合、組み合わせ爆発による処理時間の著しい増大が大きな問題となっている。

ところで、仮説推論[2]は、ある目標が与えられた前提から説明できなくてもそれに仮定を導入して矛盾なく目標を説明できれば良いという推論形態である。ここで、目標を設計目標、前提を使用が規定された構成要素、仮定をシステム実現に使用できるかどうかわからない構成要素、矛盾を仕様不満足または実現不可能、説明を合成と解釈すれば、設計型問題に仮説推論が適用できる。また、逆に、仮説推論の枠組を用いて、設計型問題のうちのある部分が、比較的うまく表現できるといえる。文献（3）は、仮説推論を用いた回路設計について述べている。これは、部品候補や接続候補を仮定として取り扱っており、前記の仮定の取り扱いはこれとほぼ同様である。

上記のように、仮説推論は前提の他に仮定という概念を導入し無矛盾な解を積極的に導出しようというものである。そのため、問題解決器になんらかの無矛盾性管理機構が必要となる。この無矛盾性管理機構として、あるデータが成立する環境の集合をデータに付与して、効率的に管理するＡＴＭＳ[4]が提案されている。

ＡＴＭＳは、矛盾する環境のスーパー・セットが矛盾することを利用して効率的にその管理を行っている。また、これを利用して効率的な問題解決戦略が設定できる[5]。しかしながら、この効果は、矛盾定義が充分に行われた場合に有効であり、極めて自由度の大きい設計型問題に対しては、やはり組み合わせ爆発の問題は解決されない。

組み合わせ爆発の問題点は、問題のサイズが大きくなると急激に処理時間が長くなることが主なものである。したがって、問題解決処理の高速化を果たすことは、従来、所与の時間内に解けなかったあるサイズの問題が解けるという意味で重要である。特に、組み合わせ爆発による処理時間の著しい増大を解決するためには、劇的な高速化が期待できる高度並列処理技術を適用することが、有効であると思われる。

設計型問題解決に限らず、ＡＴＭＳ自体には組み合わせ爆発の問題が、内包されている。そこで、ＡＴＭＳの並列化の研究[6][7]が行われている。文献（６）では、１６Ｋのコネクション・マシン上でのＡＴＭＳで１３クイーン問題を解いた場合、逐次処理に比べて７０倍の高速化が実現されたと報告されている。これは汎用的な無矛盾性管理機構であるが、特に、問題解決に利用するとき、それと後続する問題解決器の性能とその並列ＡＴＭＳの性能とのバランスが大きな問題であると考えられる。

マルチ・プロセッサ上でのＡＴＭＳは、上記並列ＡＴＭＳの他に、分散協調問題解決におけるＡＴＭＳの研究[8][9]がある。しかしながら、分散協調問題解決の主な目的は、問題や知識がもともと分散環境で表現されている問題を解決することにあり、集中的に表現された問題を高速に解決することではない。

また、無矛盾性管理機構にＴＭＳ[10]があり、これの並列環境上へのインプリメントの研究[11][12]がなされている。これらの研究もまた、高速化を主な目的とはしておらず、良いモデリングやアルゴリズムの自然な記述を目的としている。

本稿の目的は、設計型問題解決に焦点を絞り、問題解決部と無矛盾性管理部とを統合した高速な並列仮説推論器の実現方法を提案することにある。特に、並列マシン上の実現では、超高速化を果たすため、競合仮定集合に着目した、ストリーム・スイッチング方式を提案している。本稿は、まず設計型問題の解決に有効であると思われる仮説推論の枠組を示し、それに基づいた仮説推論器を並列マシン上に実現する一方式について論じ、疑似並列環境上でそれをシミュレートして時間性能を評価している。

2．本仮説推論の枠組

本仮説推論の枠組による設計型問題は、構成要素の無矛盾な組み合わせを求めることとしている。

いま、設計目標としているシステムをｓとし、構成要素の集合をＥとする。構成要素集合Ｅの要素は、単独では全て実現可

能とする。構成要素は、いくつかの属性を持っているが、レベルはアトム（あるいはグランド・インスタンス）である。この構成要素集合Eを、前提集合Pと仮定集合Aに分けて考える。

$$E=P \vee A . \qquad ①$$

前提集合Pの要素piは、使用が規定された構成要素ととらえる。使用が規定された構成要素とは、仕様から考えてこれらの構成要素は必ず使用すべきであると考えられるものである。

$$P=\{p_0, p_1, \cdots, p_n\} . \qquad ②$$

これら前提の構成要素だけからシステムsが実現できることがわかっていれば、設計目標に達しており、問題はない。しかしながら、設計には極めて次元の大きい仮定集合Aがある。この仮定集合Aの要素は、システムs実現に使用されるかどうかわからない構成要素ととらえる。しかしながら、システムs実現に必要な構成要素の上位概念はわかっているものとする。この上位概念をAiで表すとすると、仮定集合Aは上位概念Aiのユニオンで表現されることとする。

$$A=\bigvee_{i=0}^{n} A_i . \qquad ③$$

上位概念Aiに属する単独で実現可能な構成要素aijは、今までの設計事例やカタログ等を参照することによって得ることができる。

$$A_i=\{a_{i1}, a_{i2}, \cdots, a_{ik}\} . \qquad ④$$

この上位概念Aiに属する任意の2つの要素aijとailとはシステムsを実現するのに同時に使用することはないものとする。つまり、要素aijとailとは競合する仮定であるとする。これにより、上位概念Aiは、競合仮定集合ととらえられる。

$$a_{ij} \wedge a_{il} \rightarrow \perp . \qquad ⑤$$

ここに、j，l＝1，2，・・・，k，j≠l、記号 ⊥ は、矛盾（contradiction）であるまたは実現不可能であることを意味する。ここでは、組み合わせても無意味であることを示している。

また、上位概念Ai間にも、組み合わせを禁止する物理的あるいは経験的等の知識がある。この知識は、抽象概念間の関係で、いま、変数αiの領域をAi、上位概念Aiとほかの上位概念Aj（j＝0，1，・・・，（i－1））間の関係をriとすると、その意味は⑥式で表現されるものとする。ここで、前提Pに関する必要な情報は既にその中に埋め込まれていると考えてもよい。また、関係riの集合をRとし、⑦式で表す。ここに、集合Rの要素をm個としたが、任意の上位概念間の関係を表現できる。例えば、α1とα3との関係は、関係r3に選言関係で付加すればよい。

$$(\forall \alpha_i) \cdots (\forall \alpha_i) (\forall \alpha_0) [r_i(\alpha_i, \cdots, \alpha_i, \alpha_0) \Rightarrow \{(\bigwedge_{j=0}^{i} \alpha_j) \rightarrow \perp\}]. \qquad ⑥$$

$$R=\{r_0(\alpha_0), r_1(\alpha_i, \alpha_0), \cdots, r_m(\alpha_m, \cdots, \alpha_i, \alpha_0)\} . \qquad ⑦$$

⑥式で、i＝0，1，・・・，mであり、記号 ⟹ は、”関係rが内容的に真ならば”を意味している。また、関係rは仮定αの順序付けは表現できない。

以上より、本設計型問題向け仮説推論におけるシステムsの設計を⑧式のように定義する。

$$\left.\begin{array}{l} \forall \alpha_m \cdots \forall \alpha_1 \forall \alpha_0 \{(\bigwedge_{i=0}^{n} p_i) \wedge (\bigwedge_{j=0}^{n} \alpha_j) \rightarrow s\} , \\ \text{かつ、} s \text{ は無矛盾である。} \end{array}\right\} \qquad ⑧$$

3．並列仮説推論器の構成と動作

まず、入力としては、前提集合P、仮定集合Aおよび関係集合Rであり、出力としては、無矛盾な構成要素組み合わせにより支持された設計目標sである。また、本仮説推論器のタスクは、設計型問題に重要であると思われる次のタスクである。

・仮定を選択するタスク
・選択された仮定の結合を生成するタスク
・生成された仮定の結合に関する無矛盾性を検証するタスク

以下に、本並列仮説推論器のプロセスの構成法、知識の分散法、プロセスの動作について示す。このプロセスの構成は、木構造であるが、競合仮定集合のおのおのの要素同志は組み合わせが禁止されているので、以下に述べるストリーム・スイッチング方式により、トークンを異なるプロセスに分散させても独立に（異なるプロセス間の通信なしに）処理が進められる点が高速化に関する基本的なアイデアである。これによっても解の完全性が保証される。

まず、ストリームにより受信されたデータにより駆動され、結果をストリームにより送信するプロセスが木構造に配置されたマルチ・プロセス・システムにおいて、木の深さiにおけるプロセスに必要な前提部分集合Pi、競合仮定集合Aiおよび関係riを分散（記憶）させておく。任意のプロセスiの子の数は、⑨式で示される自然数biであるものとする。この式は、右辺を越える数の子プロセスを配置させても、明らかに高速化という観点からは意味がないことを示している。また、右辺と等しくした場合は、本方法による最高速度（高速化の限界）を要し、最良の分散と呼ぶ。しかし、このとき、かなりの数のプロセスを要する。この自然数biは、木の深さiのプロセスにおける枝分かれ率と呼ぶ。

$$\left.\begin{array}{l} b_0 \leq k_0 , \\ b_i \leq \left\lceil \prod_{j=0}^{i-1} k_j / b_j \right\rceil \cdot k_i . \end{array}\right\} \qquad ⑨$$

プロセスiにその子プロセスへ順序付けられたbi本のストリームをbi個の子プロセスとの間に設ける。プロセスiは、結果をその順序に基づいてそのうち1本のストリームに送信する。つまり、最初のトークンを先頭のストリームに流し、次のトークンについては次のストリームに流す。終端のストリームにトークンを流した後には、再び先頭のストリームにトークンを流す。すなわち、サイクリックに切り変わるスイッチがそのプロセスiの出力端とbi本のストリーム入力端の間に設けられたイメージである。そこで、このような通信方式をストリーム・スイッチング方式と呼ぶ。また、このようなスイッチが入力端に設けられたストリーム・セットをスイッチング・ストリ

ームと呼ぶ。

木の深さ0におけるプロセスは、木の根に相当し、ルート・プロセスと呼ぶ。ルート・プロセスの動作は、A^0の要素が自身が保持する関係r0を満たさなければ単にスイッチング・ストリームにそのA^0の要素を送信する。関係r0を満たせば、その仮定は送信せず、次のA^0の要素について処理を行う。全ての要素の処理が終わった後に終了コードを全てのストリームに送信し死滅する。

木の深さ1から（m－1）におけるプロセスiは、ストリームよりトークンを入力し、自身が保持するAiの要素と組み合わせ、さらに自身が保持する関係riを満たさなければその仮定組み合わせをスイッチング・ストリームに送信する。関係riを満たせば、その仮定組み合わせは送信せず、次のAiの要素と組み合わせて処理を行う。Aiの全ての要素と組み合わせて処理をおこなった後に再びトークンの入力を行う。入力トークンが終了コードであれば、終了コードを全てのストリームに送信し自身は死滅する。

木の深さmにおけるプロセスmは、木の葉に相当し、リーフ・プロセスと呼ぶ。リーフ・プロセスの動作も木の深さ1から（m－1）におけるプロセスと同様である。しかし、これ以上、仮定との組み合わせは行われないため、結果の送信はスイッチング・ストリームではなく、単なるストリームであってよい。このリーフ・プロセスのストリームは、この仮説推論器の結果を受信（さらに、解の評価を行い表示等を行う）するプロセス（グランドと呼ぶ）で、マージされる。ここで可能ならば、グランドをいくつかのプロセスで構成してもよい。

以上、プロセスは論理的な概念として示したが、実際の物理的プロセッサとのマッピングもまた、これと1対1におこなわれるものとしている。

図1は、本仮説推論器の並列モデルを示す構成図である。以下、同一深さiのプロセスに自然数jを付加し（i，j）でプロセスを指定する。ここに、全てのjについて、内包されている知識は全て等しいことは前記のとおりである。

なお、文献（8）の分散環境におけるＡＴＭＳで、Ｎｏｇｏｏｄの通信を必要としていた。しかしながら、本研究では、一括高速問題解決を目標とし、後から仮定や関係が追加されることを考慮していない。これに基づき、枠組設定および知識の分散を適切に行い、Ｎｏｇｏｏｄの通信を不必要としているが、健全性は保証されている。仮定の追加に関しては、他プロセスより送られてきたデータを保持しておくメモリーを設ければ可能である。一方、関係の追加に関しては、一般に、ＡＴＭＳを用いた問題解決においては、関係の削除（矛盾の撤回）が困難であるのに、関係の追加だけサポートしても、問題をもっと限定して考えないとその意味を見出すことはできない。文献（13）は、局所矛盾の撤回を扱った研究である。

さらに、本方法は、動的プロセス生成機構を採用しておらず、それに比較し不必要となるかもしれないプロセスを予め生成しておく無駄があるかもしれない。しかしながら、一般に、動的プロセス生成には少なくともいくらかの処理時間を要する。本研究では、その時間も縮めようと静的プロセス配置を採用している。

4．シミュレーションと評価

本方式の高速化の程度を疑似並列環境上でシミュレーションを行い示す。そのためのマシンをＰＳＩ－ＩＩ[14]としているので、各プロセスiの動作アルゴリズムを言語ＥＳＰ[15]のｌｏｃａｌ述語を用いて図2に示す。なお、深さiのプロセスを

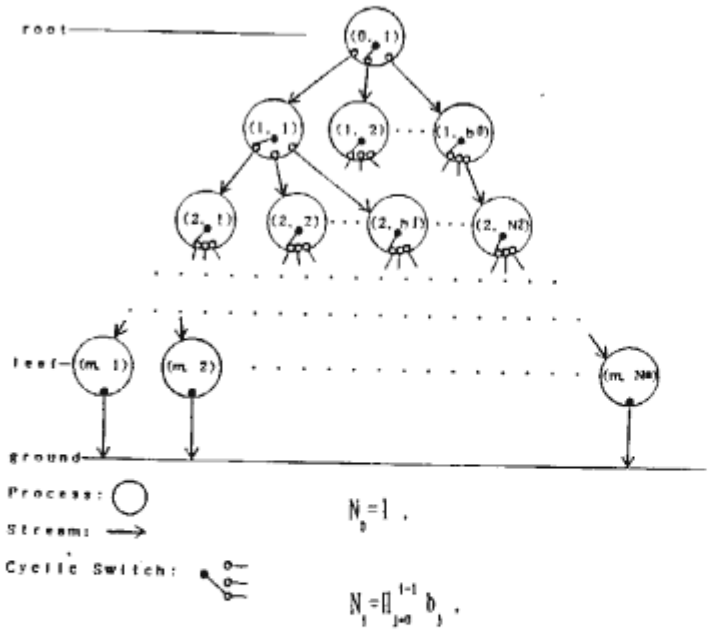


図1　本並列仮説推論器の構成

```
goal(I):-
      repeat,
      :get(I!in, Data),
    %データ入力
      process(Data, I)
      ;
process(end, I):-
      !, :put(I!out, end)
    %データ出力
      ;
process(Data, I):-
      :get(I!a, Aij),
    %仮定の取り出し、オールタナティブあり。
      Token=[Aij|Data],
      :not(I!r, Token),
    %関係を満たしていない。
      :put(I!out, Token),
      fail
      ;
```

図2　プロセスiの動作アルゴリズム

インスタンス・オブジェクトIで表し、Iのスロットにはｉｎ，ｏｕｔ，ａ，ｒがある。ｉｎは入力ストリーム、ｏｕｔはスイッチング・ストリーム、ａは競合仮定集合記憶オブジェクト、ｒは関係記憶オブジェクトをそれぞれ保持している。また、ｅｎｄを終了コードとする。

任意のプロセス（ｉ，ｊ）のｐｒｏｃｅｓｓ第２節の処理時間ｔij（：ｇｅｔ（Ｉ！ａ，Ａｉｊ）からｆａｉｌして再び：ｇｅｔ（Ｉ！ａ，Ａｉｊ）にバックトラックするまでの時間）をそのプロセス（ｉ，ｊ）に流れてくる全てのデータおよびそのプロセス（ｉ，ｊ）が保持する競合仮定集合の全ての要素につき一定と仮定する。また、このｔijは、全てのｊにつき一定と仮定し、単にｔiと記述する。この仮定は、対象とする問題で関係集合Ｒが空集合（φ）の場合はうまく成立するものと考えられる。関係検査の負荷が軽い場合において、関係集合Ｒが空集合の時は枝刈りが行われず、問題解決器の負荷が最大となる。このとき、問題解決器の処理時間は最悪の場合に相当する。また、任意のプロセス（ｉ，ｊ）からデータを流しその子プロセス（ｉ－１，１）に達するまでの伝達遅れ時間は、深さが同じ全てのプロセスおよび全てのデータにつき同一であると仮定し、これをτiとする。以上の仮定は、シミュレーション結果の解析においても仮定している。

さらに、一般的な解析のために、次の仮定を設ける。深いプロセスは浅いプロセスよりもｐｒｏｃｅｓｓ第２節の処理時間ｔiが大きいものと仮定する。この仮定は、深くなるにつれＴｏｋｅｎの長さが長くなるし、関係検査の負荷も大きくなるものと考えられ、大体において満足する。また、次段のプロセスが保持する競合仮定集合の要素数が前段のプロセスが保持する競合仮定集合の要素数とあまり変わらないものとすると、一度トークンを受信し活性化したプロセスは死滅するまで全く待つことはない（前段の処理が遅いことに起因するトークンの入力待ちはない）といえる。したがって、全ての枝分かれ率ｂiを競合仮定集合の要素数ｋiと等しくとり、かつ、前記最悪の場合の処理時間をＴｐとすると⑩式のようになる。また、これと比較するために、関係集合Ｒが空集合であるとした逐次マシン上の処理時間をＴｓとすると⑪式のようになる。

$$T_p=\sum_{i=0}^{n}\left(k_i\cdot t_i+\tau_i\right) \qquad ⑩$$

$$T_s=\sum_{i=0}^{n}\left\{\left(\prod_{j=0}^{i}k_j\right)\cdot t_i\right\} . \qquad ⑪$$

⑩式と⑪式に示すように、逐次の場合それぞれ競合仮定集合の要素数が大きくなると処理時間は爆発するのに対して、本方法によればそれぞれの線形結合で表現されているので処理時間の爆発が防げる。本方法は、このような時間性能の向上という効果はあるが、それに要するプロセスの数をＮとすると⑫式で示すようにこれが極めて大きくなる可能性が高い。

$$N=\sum_{i=0}^{n-1}\left(\prod_{j=0}^{i}b_j\right)+1 . \qquad ⑫$$

時間性能評価用の例題として、４×４のチェス盤上に４個のクイーンが衝突しないように配置する４クイーン問題をとりあげる。

本並列仮説推論器の入力として、以下の情報が必要である。

Ｐ＝｛４×４のチェス盤｝．

問題を４×４のチェス盤と４個のクイーンとの無矛盾（矛盾は、クイーンが衝突する）な組み合わせを求めることとすると、４×４のチェス盤は使用が規定されたシステム構成要素ととらえられる。この情報は、関係集合に直接埋め込まれる。

また、システム合成に使用されるかどうかわからない構成要素は、次のとおりである。

A^0＝｛ｑＩ（１），ｑ１（２），ｑ１（３），ｑ１（４）｝．
Ａ１＝｛ｑ２（１），ｑ２（２），ｑ２（３），ｑ２（４）｝．
A^2＝｛ｑ３（１），ｑ３（２），ｑ３（３），ｑ３（４）｝．
A^3＝｛ｑ４（１），ｑ４（２），ｑ４（３），ｑ４（４）｝．

ここで、ｑＩ（Ｊ），　Ｉ，Ｊ－１，２，３，４であり、チェス盤上Ｉ行Ｊ列にクイーンを置くという仮定を意味するものとする。さらに、競合仮定集合おのおのについて、⑤式を満たす。

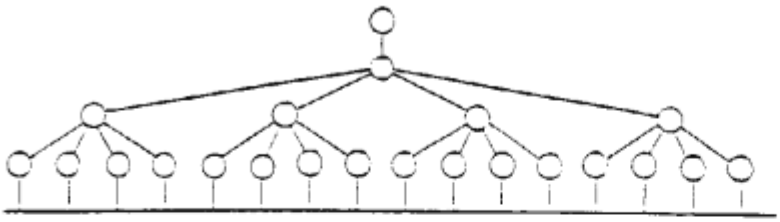
関係の内容は以下のとおりである。

```
r0（［q1（I）］）：
        fail；
r1（［q2（I），q1（J）］）：－
        （（I＝J，！；I－J＝1），！；J－I＝1）；
r2（［q3（I），q2（J），q1（K）］）：－
        （（（（（I＝J，！；I＝K），！；I－J＝1），！；
        J－I＝1），！；I－K＝2），！；K－I＝2）；
r3（［q4（I），q3（J），q2（K），q1（L）］）：－
        （（（（（（（I＝J，！；I＝K），！；I＝L），！；
        I－J＝1），！；J－I＝1），！；I－K＝2），！；
        K－I＝2），！；I－L＝3），！；L－I＝3）；
```

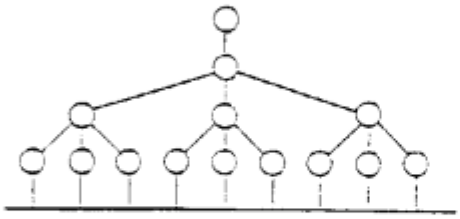
図３は、本シミュレーションのプロセスの配置とストリームの接続関係を示すものである。本シミュレーションは、ＰＳＩ－ＩＩ、ＳＩＭＰＯＳ（ＰＳＩ－ＩＩのプログラミング・アンド・オペレーティング・システム）上のマルチ・プロセスとストリームを利用してなされている。ＳＩＭＰＯＳでは、最大６４個までのプロセスが生成できるようになっている。そのため、プロセス数に大きく依存するｂ０は１とした。ここで、ｂ１＝ｂ２＝ｂとして、ＡＴＭＳを利用した逐次マシン上の仮説推論システムＡＰＲＩＣＯＴ／０[15]の処理時間を基準とした高速化について実験を行った結果を図４に示す。また、ｂ１＝ｂ２＝ｂとして、ｂ＝０，すなわち逐次処理の場合の処理時間を基準とした高速化について実験を行った結果を図５に示す。なお、伝達遅れ時間τiは実現する並列マシンのＨ／Ｗに大きく依存するため、本シミュレーション結果の解析においては、０としている。

図４について考察する。ｂ＝０のときに、Ｒ≠φにおいて１８倍、Ｒ＝φにおいて２倍の高速化が得られているのは、問題解決器と無矛盾性管理部との完全融合により、さらに機能を特化させたための効果である。また、ｂ＝１においての高速化はＡＮＤ並列処理による効果である。さらに、ｂが増加するに従った高速化の増加は、ストリーム・スイッチング方式の効果である。この範囲で示せる結論は、Ｒ≠φのときで、ＡＴＭＳを利用した逐次マシン上の仮説推論システムＡＰＲＩＣＯＴ／０に比べ、５６倍の高速化が実現されるということである。

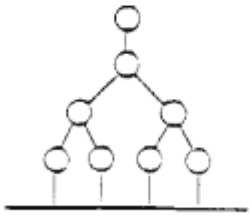
次に、図５について考察する。これは、並列化の効果を示すものである。したがつて、ｂ＝０のときに、Ｒ≠φおよびＲ＝φにおいて高速化は１である。本例題のＲ≠φでは、解は２つ



（a）枝分かれ率b＝4



（b）枝分かれ率b＝3



（c）枝分かれ率b＝2

（d）枝分かれ率b＝1

図3　シミュレーションに用いたプロセス構成

しかなく、Rによる枝刈りの効果は大きい。そのため、b＝3以上としても高速化はあまり向上しない。この場合、枝刈りが充分に行われてしまうb2の分散には効果はあまりない。むしろ、枝刈りがあまり進んでいないb0を分散させたほうがよい。しかし、このようなことは、実際に定義された関係を評価して初めてわかることである。結局、R＝φのとき、b＝4で15倍の高速化となっている。このときのプロセス数は図3に示すように、22である。しかしながら、このときはまだ、高速化について、飽和状態に達しておらず、b＝16まで速度向上が得られるものと考えられる。

また、図4および図5が示すように、R≠φの場合、b＝3以上にすると高速化の効果が減少してくるが、R＝φの場合はb＝4にしても高速化の効果が伸びている。よって、本方式は関係を充分に定義できない場合に有効であることがわかる。はじめにで述べたように、早期に枝刈りを行うことが難しい設計型問題に用いた場合、本並列仮説推論器の浅いプロセスにおいてbを大きくしておき、枝刈りが充分に行うことができると考えられる深いプロセスにおいてはbを小さくしておいたほうがよいといえる。

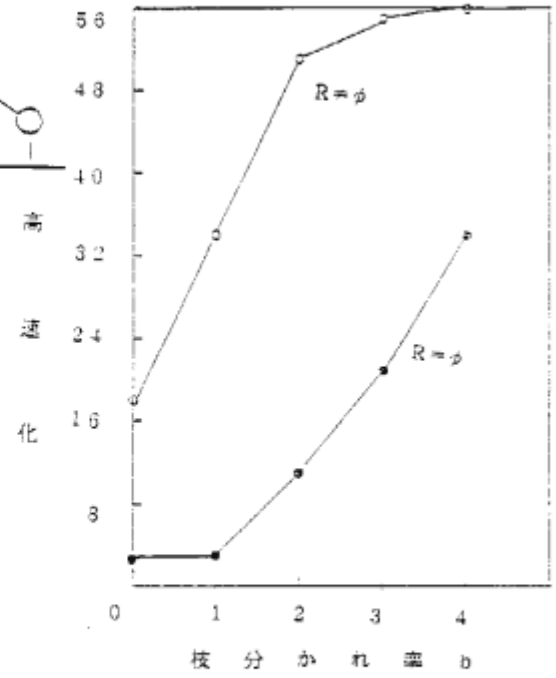


図4　枝分かれ率b－APICOT/0に対する高速化

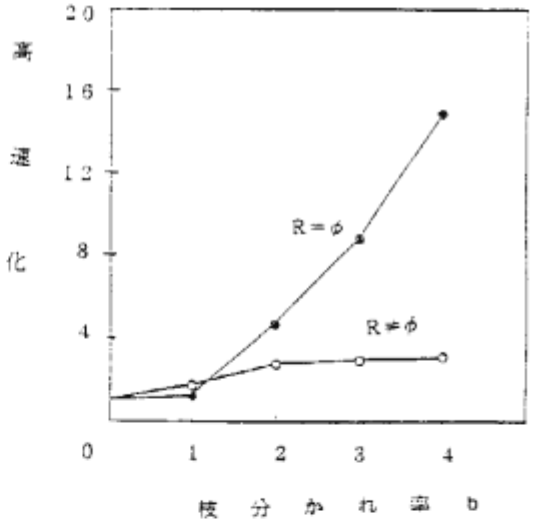


図5　枝分かれ率b－逐次処理に対する高速化

5．まとめ

本稿では、仮説推論を用いた設計型問題解決を構成要素の無矛盾な組み合わせを求めることとし、その問題解決器を並列マシン上に実現する一方式を示した。この方式を疑似並列環境上に実現し、シミュレーションにより本方式の時間性能を評価した。

このシミュレーションの結果、4クイーン問題で、ATMSを利用した逐次マシン上の仮説推論システムAPRICOT/0に比べ、56倍程度の高速化が実現されることが示された。また、問題解決器と無矛盾性管理部との完全融合により、逐次マシン上においても性能向上が見られることが明らかになった。さらに、ストリーム・スイッチング方式は、早期に枝刈りを行うことが難しい設計型問題に有効であることが明らかになった。

本仮説推論の枠組において、逐次処理の場合、それぞれ競合仮定集合の要素数が大きくなると処理時間が爆発する。これに対して、本方法による最良の分散を行った場合、競合仮定集合の要素数の線形結合オーダなので、処理時間の爆発が防げる。本方法は、このような時間性能の向上という効果はあるが、それに要するプロセスの数が極めて大きくなる可能性が高い。この解決策として、枝刈りを充分に行うことができると考えられる深いプロセスにおいては枝分かれ率を小さくしておく方法があげられる。また、浅いプロセスの処理が早く終わることに着目し、これを再利用する等の方法を確立することは、今後の課題である。

さらに今後の課題として、実際の並列マシン（Multi-PSI[17]、PIM[18]）上での評価があげられる。また、実際的な設計型問題、特に従来、機械による支援が困難であったその上流に適用し、有効性を示すとともに知識表現（本仮説推論の枠組）等の問題を洗い出し解決することがあげられる。実際的な設計型問題に適用したとき、さらに大きいシステムとしての性能バランスの問題が明らかになってくると思われる。

謝辞

本研究の機会を与えて下さり常に御指導頂いているＩＣＯＴ渕所長、第五研究室生駒室長に深く感謝致します。また、日頃、御世話になっている第五研究室の皆様に深く感謝致します。

文献

（１）小林重信、関根史磨、菊地一成、森啓、千吉良英敏、中島俊哉　他：”知識情報処理システムに関する調査研究報告書：第１分冊　計画・設計型知識システムの構築方法論”、ＩＣＯＴ－ＪＩＰＤＥＣ　ＡＩセンター（１９８９）

（２）國藤進：”仮説推論”，人工知能学会誌　Ｖｏｌ．２，Ｎｏ．１，ｐｐ．２２－２９（１９８７）

（３）牧野俊朗、石塚満：”仮説推論による回路設計システムにおける制約式に基づく効率的推論法”，情報処理学会第３８回全国大会講演論文集（Ｉ），ｐｐ．４２４－４２５（１９８９）

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

（５）ｄｅ　Ｋｌｅｅｒ，　Ｊ．：”Ｐｒｏｂｌｅｍ　Ｓｏｌｖｉｎｇ　ｗｉｔｈ　ｔｈｅ　ＡＴＭＳ”，Ａｒｔｉｆｉｃｉａｌ　Ｉｎｔｅｌｌｉｇｅｎｃｅ　２８，ｐｐ．１９７－２２４（１９８６）

（６）Ｍ．Ｄｉｘｏｎ，Ｊ．ｄｅ　Ｋｌｅｅｒ：”Ｍａｓｓｉｖｅｌｙ　Ｐａｒａｌｌｅｌ　Ａｓｓｕｍｐｔｉｏｎ－ｂａｓｅｄ　Ｔｒｕｔｈ　Ｍａｉｎｔｅｎａｎｃｅ”，Ｐｒｏｃ．ｏｆ　ＡＡＡＩ－８８，ｐｐ．１９９－２０４（１９８８）

（７）原田拓、溝口文雄：”データ依存関係に基づく無矛盾性管理の並列処理方式”，情報処理学会第３８回全国大会講演論文集（Ｉ），ｐｐ．４１８－４１９（１９８９）

（８）橋尾真、石田亨　：”分散協調問題解決におけるＡＴＭＳの利用”人工知能学会第２回全国大会論文集、ｐｐ．１４１－１４４（１９８８）

（９）Ｃｉｎｄｙ　Ｌ．Ｍａｓｏｎ，Ｒｏｗｌａｎｄ　Ｒ．Ｊｏｈｎｓｏｎ：”ＤＡＴＭＳ：Ａ　Ｆｒａｍｅｗｏｒｋ　ｆｏｒ　Ｄｉｓｔｒｉｂｕｔｅｄ　Ａｓｓｕｍｐｔｉｏｎ　Ｂａｓｅｄ　Ｒｅａｓｏｎｉｎｇ”，Ｃｏｌｌｅｃｔｅｄ　Ｄｒａｆｔ　Ｐａｐｅｒｓ　ｏｆ　ｔｈｅ　１９８８　ＷＯＲＫＳＨＯＰ　ＯＮ　ＤＩＳＴＲＩＢＵＴＥＤ　ＡＲＴＩＦＩＣＩＡＬ　ＩＮＴＥＬＬＩＧＥＮＣＥ（１９８８）

（１０）Ｊ．Ｄｙｌｅ，”Ａ　Ｔｒｕｔｈ　Ｍａｉｎｔｅｎａｎｃｅ　Ｓｙｓｔｅｍ”，Ａｒｔｉｆｉｃｉａｌ　Ｉｎｔｅｌｌｉｇｅｎｃｅ，Ｖｏｌ．１２，Ｎｏ．３，ｐｐ．２３１－２７２（１９７９）

（１１）Ｃｈａｒｌｅｓ　Ｊ．Ｐｅｔｒｉｅ，Ｊｒ．：”ＡＤＩＦＦＵＳＩＮＧ　ＣＯＭＰＵＴＡＴＩＯＮ　ＦＯＲ　ＴＲＵＴＨ　ＭＡＩＮＴＥＮＡＮＣＥ”，Ｐｒｏｃ．ｏｆ　Ｉｎｔｅｒｎａｔｉｏｎａｌ　Ｃｏｎｆｅｒｅｎｃｅ　ｏｎ　Ｐａｒａｌｌｅｌ　Ｐｒｏｃｅｓｓｉｎｇ，ｐ．ｐ．６９１－６９５（１９８６）

（１２）久野禎子、久野靖：”並列オブジェクト指向言語を用いたＴＭＳの再構成”、人工知能学会誌　Ｖｏｌ．４　Ｎｏ．１、ｐｐ．６２－６９（１９８９）

（１３）西岡真吾、堀雅洋、池田満、溝口理一郎、角所収：”ＡＴＭＳの運用について”，第８回知識工学シンポジウム論文集ｐｐ．７５－８０（１９８８）

（１４）Ｈ．Ｎａｋａｓｈｉｍａ，Ｋ．Ｎａｋａｊｉｍａ：”ＨＡＲＤＷＡＲＥ　ＡＲＣＨＩＴＥＣＴＵＲＥ　ＯＦ　ＴＨＥ　ＳＥＱＵＥＮＴＩＡＬ　ＩＮＦＥＲＥＮＣＥ　ＭＡＣＨＩＮＥ：ＰＳＩ－ＩＩ”，Ｐｒｏｃ．ｏｆ　４ｔｈ　Ｓｙｍｐｏｓｉｕｍ　ｏｎ　Ｌｏｇｉｃ　Ｐｒｏｇｒａｍｍｉｎｇ，ｐｐ．１０４－１１３（１９８７）

（１５）Ｃｈｉｋａｙａｍａ，Ｔ．：”Ｕｎｉｑｕｅ　Ｆｅａｔｕｒｅｓ　ｏｆ　ＥＳＰ”，Ｐｒｏｃ．ｏｆ　ＦＧＣＳ’８４，ｐｐ．２９２－２９８（１９８４）

（１６）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）

（１７）Ｋ．Ｔａｋｉ：”Ｔｈｅ　ｐａｒａｌｌｅｌ　ｓｏｆｔｗａｒｅ　ｒｅｓｅａｒｃｈ　ａｎｄ　ｄｅｖｅｌｏｐｍｅｎｔ　ｔｏｏｌ：Ｍｕｌｔｉ－ＰＳＩ　ｓｙｓｔｅｍ”，Ｐｒｏｃ．ｏｆ　ｔｈｅ　Ｆｒａｎｃｅ－Ｊａｐａｎ　ＡＩ　ａｎｄ　Ｃｏｍｐｕｔｅｒ　Ｓｃｉｅｎｃｅ　Ｓｙｍｐ．８６，Ｉｎｓｔｉｔｕｔｅ　ｆｏｒ　Ｎｅｗ　Ｇｅｎｅｒａｔｉｏｎ　Ｃｏｍｐｕｔｅｒ　Ｔｅｃｈｎｏｌｏｇｙ（１９８６）

（１８）Ａ．Ｇｏｔｏ　ａｎｄ　Ｓ．Ｕｃｈｉｄａ：”Ｔｏｗａｒｄ　ａ　ｈｉｇｈ　ｐｅｒｆｏｒｍａｎｃｅ　ｐａｒａｌｌｅｌ　ｉｎｆｅｒｅｎｃｅ　ｍａｃｈｉｎｅ－Ｔｈｅ　ｉｎｔｅｒｍｅｄｉａｔｅ　ｓｔａｇｅ　ｐｌａｎ　ｏｆ　ＰＩＭ”，Ｔｅｃｈｎｉｃａｌ　Ｒｅｐｏｒｔ　ＴＲ－２０１，ＩＣＯＴ（１９８６）