

TM-0580

並列推論におけるヒューリスティックス知識
〔基本検討編〕

滝寛和

July, 1988

©1988, ICOT

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

並列推論におけるヒューリスティックス知識
――<並列推論システムのための知識ベース構築>――

「基本検討編」

滝 寛和

Csnet: htaki@icot.jp, Junet: htaki@icot.junet

(財)新世代コンピュータ技術開発機構

まえがき

本論文は並列推論を効率的に行うヒューリスティックス知識の基本的検討結果について述べる。並列推論システムの大きな問題として、プロセスの管理と資源の効率的割り当て問題がある。この問題は、推論による問題解決の効率に非常に大きな影響を与える。逐次推論では、問題解決の効率は、ヒューリスティックス知識に依存していた。ヒューリスティックス知識は、効率的な推論・探索を行うために、極力不必要な処理をさけるための知識である。この知識は、エキスパートシステム等の知識ベースシステムの対象に非常に依存したものである。並列推論の効率化においても、同様のヒューリスティックスが必要であり、それに応じたプロセスの優先度／資源の割り当てを考える必要がある。たとえば、1つの適切な解（最適解ではない）を得るには、不必要にOR並列プロセス（別解）を生成せず、適切な解を導くであろうプロセスに優先的に資源割り当てをおこなわせる必要がある。従来、並列推論の効率化問題は、並列マシンの各プロセスエレメントと資源（メモリなど）にプロセスを「資源の有効利用とプロセス間の通信量を減らすように」割り当てる問題として考えられてきた。しかし、前述のように、問題解決の効率化に関する情報（ヒューリスティックス知識）の方が、よりプロセスの効率的処理に関する高次の情報となっている。この情報の有効利用こそが並列推論マシンによる効率的な問題解決をもたらすことになる。

1. はじめに

知識情報処理における知識は、探索空間（探索木と概念知識）と効率的な探索方法として、とらえることができる。従来の知識処理では、逐次的な処理系であるため、選択的な状況においては、枝刈り・枝選択を中心としたヒューリスティックス知識が重要であった。しかしながら、並列推論システムにおいては、その資源内であれば、選択的な状況において、すべての枝（選択枝）を同時に（OR並列）処理できることになる。これは、従来のヒューリスティックス知識が並列推論システムでは重要でなくなる場合があることを示している。本論文では、逐次推論システムにおけるヒューリスティックスと並列推論システムにおけるヒューリスティックスの違い、及び、並列推論システムのための知識ベース構築（知識獲得）について言及する。

2. 逐次推論システムにおけるヒューリスティックス知識

エキスパートシステムなどの知識ベースシステムにおける知識は、解くべき問題の探索空間知識と探索戦略知識に大別される。探索空間知識には、問題とする分野における「概念知識」（一般には、フレームや意味ネットワークとして表現されている場合が多い）が含まれる。探索戦略知識には、推論方法（縦型探索や横型探索、backtrackingの方法）とヒューリスティックス知識（探索木の枝刈り、枝選択の評価）がある。プロダクションルールにより表現された知識には、探索空間を張る知識（探索空間を制限する知識）と探索戦略知識が混在している。ヒューリスティックス知識には、この探索空間を制限する知識やチャンクされた知識（いわゆる、浅い知識、小さい探索木）も含める。知識ベースシステムの能力（知識の質）を計る場合には、2種類の尺度がある。

- (1) その知識ベースがその使用目的を満足するかどうか
(ある状況に対して、適切な解を導出できるかどうか)
- (2) いかにも無駄な処理を行わないで、効率良く解を導けるかどうか
(注：(1)(2)ともに推論機構の能力とあわせて考える必要がある)

つまり、「何が導かれるか」・「効率良く導けるか」が知識の質を決める。前者は、「探索空間」についての評価である。後者は、探索戦略の評価である。S S：探索空間とO：観測データの組み合わせの集合が与えられたときに導出できる解をRの集合とすると次の関係が成立つ。

$$S \subseteq O \cup R$$

知識ベースの構築（知識獲得）の観点から見ると、OとRが与えられたときに、S Sを求めるのが、知識獲得である。しかしながら、S Sは探索空間を与えるのみであり、求解効率（推論効率）、求解手順（推論方式）については情報を含まない。逐次推論システムでは、その推論システムでの推論方法（ESPやPROLOGでは、縦型探索による推論方式を使用している）が与えられるがヒューリスティックス知識は、S S同様獲得しなければならない。ここで、逐次推論のためのヒューリスティックス知識を整理すると次のようになる。

- (ヒューリスティックス1)：推論フローの選択・制御を行う。
一般的には、探索木のO R枝の選択知識を言う。
この知識には、固定的なO R枝選択優先度も含む。
確信度などの評価値と評価関数もこの知識に含まれる。
- (ヒューリスティックス2)：推論フローの制限を行う。
いわゆる、探索木のO R枝の内、不要な枝を刈る知識。
論理条件に合わない枝や評価の低い枝などを刈る。
確信度などの評価値と評価関数もこの知識に含まれる。
- (ヒューリスティックス3)：推論の短絡（バイパス）を行う。
必ず成立する枝と必ず失敗する枝の情報を知識として持つ。
探索木の構造自体を変更して、推論を行う。
(推論の完全性を維持する場合、構造を縮退しないで、
優先探索木を追加するのが一般的である。例：部分計算)

これらのヒューリスティックス知識は、エキスパートシステムなどの逐次推論向きの知識処理の効率向上知識である。一般に探索空間が大きい場合には、ヒューリスティックスなしに有限時間で適切な解を得ることが難しい。

3. 並列推論システムにおける擬似ヒューリスティックス知識

ここでは、まず、並列推論のメカニズムを理想的に見た並列推論システムを考える。

「理想並列推論システム」

- (1) 資源無限大：無限の並列度を得られる。
プロセスの増減に係わらずプロセス実行速度は一定。
- (2) 通信コストは一定：情報アクセスはすべてのプロセスとも一定。
(共通知識ベースへのアクセスの競合はない、
どのプロセス間の通信コスト／時間ともに一定)

並列推論における並列性には2種類ある。つまり、AND並列とOR並列である。逐次推論では、探索木のOR枝に注目して、推論効率を向上させている。ヒューリスティックス1とヒューリスティックス2がそれに当たる。ところが、理想並列推論システムでは、OR枝選択のヒューリスティックスは重要ではない。なぜならば、すべての枝をプロセスとして実行しても、求解の効率（最初の解を得るまでの時間）は落ちない（同じファクト知識、概念知識をアクセスしてもアクセス競合がおきないため）。逐次推論のヒューリスティックス3は、探索木の構造を変更することによる推論の効率化であるので、並列推論システムでの知識処理にも有効である。さて、AND並列についての推論処理の効率は、並列に分割されたプロセスの処理時間の中最大のものとは逐次処理時間の比較で表わされる。つまり、いくつかの部分問題・部分プロセスに分解できるかだけが問題ではなく、その中で処理時間の最大のものとの効率的処理が重要である。

AND並列の推論効率の向上としては、ヒューリスティックスに対応するものとして、
(擬似ヒューリスティックス4)：AND並列での部分問題中の最大処理時間
プロセスが最小となる部分問題に分割する。(分割数よりもこの評価が重要)

これは、問題対象に依存するのではなく、探索空間の構造に依存している。ヒューリスティックス1、2、3は問題対象の構造に依存しているが、この擬似ヒューリスティックス4は探索空間の構造のみに依存しているので、知識源からの知識獲得するのではなく、問題解決知識の構造を解析することで得られる。

(擬似ヒューリスティックス5)：他の並行(OR)プロセスの情報を利用する。

逐次処理では、一つの推論プロセス、または、過去の推論プロセスの中間結果を利用して、同じ求解処理は行わないで効率的に推論することができる。逐次推論での、この応用には、インテリジェントバックトラッキングがある。ATMSでは、並行プロセスでの仮説の禁止組み合わせ情報(nogood)を利用している。基本的に、各推論プロセスは、並行プロセスの情報がなくとも結果を得ることが可能である。この情報には、2種類ある、ATMSのnogoodの様にプロセスの禁止／無効に関するものと、そのプロセスが求解するものを先取りするものがある。どちらもそのプロセス内で求解できる情報である。

4. 現実の並列推論システム向きヒューリスティクス知識

4. 1 並列推論の効率化問題

並列推論の高速化を考える場合に、どのレベルでの効率の改善を考えるかが重要である。現在の並列処理の効率化のほとんどは、並列処理向きアーキテクチャのレベルで考えられてきた。そして、そのアーキテクチャとは独立に研究されてきた並列言語（含む並列推論言語）をそのアーキテクチャ上にいかにうまくマップするかが問題となっている。これは、言い換えると、言語レベルに表れる情報のみに基づいて、並列化を行い、マシンの資源配分を行う問題である。しかしながら、知識情報処理の立場では、もう一段上のレベルである知識レベルの効率化改善が考えられる。これは、ヒューリスティクス知識に基づく推論の効率化に根ざしたものであり、ある種のアプローチの改善知識による方法である。各レベルの問題を次にまとめる。

(1) 知識レベルの効率化問題

知識の完全性・健全性を保ったまま効率的に求解する問題解決方法をさがす問題である。つまり、問題解決の戦略知識の並列化（並列向きアルゴリズムを探す）とヒューリスティクス知識の獲得がこのレベルの効率改善につながる。問題解決対象の問題分析が重要である。その分析結果から、問題の部分問題化や必要なヒューリスティクスをきめることが必要である。

(2) 推論処理をマシンに適合化させる問題

知識レベルで分析され、並列処理向きに再構成された知識（プログラム）とマシンアーキテクチャを適合させる必要がある。アーキテクチャに最適な資源の配分を行う。マシンアーキテクチャ・ハードウェアの能力制約と上記並列知識表現間のマッピング問題を解く必要がある。

(3) 効率的なマシンアーキテクチャ／インストラクション問題

- ・問題向き並列アーキテクチャ／ハード構成（バス構成・メモリ構成など）
- ・効率的インストラクション

(4) 実装問題

- ・インストラクションの高速化（ファーム／ハードレベル）
- ・素子の選択

4. 2 実際の並列マシンの制約とヒューリスティクス知識

並列推論システムの目的は、知識情報の効率的処理と自然な知識処理記述を目指すものである。さて、現在、並列推論の枠組みとして、並列論理プログラム言語（GHC）が供給されているが、知識情報処理を行う為の知識表現と問題解決記述の概念はまだない。記号レベルでの道具立てはできているが、知識レベルでのサポートはない。

さて現実の並列推論システムでは、種々のハードウェア制約／並列管理の制約などにより理想並列システムのような効率化は難しい。1つは、ハードウェアや並列管理システムに依存した効率化方法が考えられる。もう1つは、逐次推論のためのヒューリスティクスを並列推論向きに作り換えて使用することである。最初の問題（並列制約問題と呼

ぶことにする)はどのような制約が、並列推論のどの分割/資源割り当てに影響するかを知ることが重要である。後者の問題は、逐次推論のヒューリスティックスと並列制約問題の関係をj知ることが重要である。ここでは、いくつかの並列制約とヒューリスティックスの関係をみることにする。(厳密な関係はより深い検討が必要)

(1) 通信コスト

- (a) プロセッサ-共有メモリ間: 共有メモリの転送能力・アクセス競合
- (b) プロセッサ-プロセッサ間: 転送能力
- (c) プロセッサの結合構成(トポロジー)

通信コストと並列度はトレードオフの関係である。

並列度を上げると擬似ヒューリスティックス5を使い他の処理の結果を利用できるので効率が向上するはずであるが、ヒューリスティックス5は、通信量を増すことになる。通信能力を見ることで、ヒューリスティックス5の適用範囲を制限する。

(2) 並列処理管理コスト

- (a) プロセッサの数
- (b) プロセススイッチングオーバーヘッド

プロセッサ数を越える並列処理は、プロセスを擬似並列化(マルチタスク化)して、プロセススイッチングをおこなわなければならない。この場合は、プロセッサ数を越える並列を行うよりも部分的には逐次処理を行う方が効率は向上する。この制約は並列度に大きく影響する。

(3) その他のヒューリスティックスに影響するとおもわれる制約

- (a) 共有メモリのサイズ
- (b) ローカルメモリのサイズ
- (c) データ更新のタイミング管理方法
- (d) その他のオーバーヘッド(動的に割り当てを行うときの計算量など)

(4) 問題解決に対する要求とヒューリスティックスの関係

- (a) 複数解が必要な場合: 全解/部分解
- (b) 単一解の場合: 最適解/適切解

上記の条件は求解のアルゴリズムに大きく影響する。

5. 結論

知識レベルの並列推論を効率的に行う知識ベースの構築には、

- (1) 並列推論の為の知識分析方法を確立する必要がある。
分析に考慮すべき概念を十分に整理しなければならない。
- (2) マシンアーキテクチャを考慮して分析内容にヒューリスティックスの追加/修正を行い知識を再構成する技術が必要である。
- (3) さらに、静的な解析で得られないヒューリスティックスを得るには、
動的な推論過程を通じて、動的データから並列推論ヒューリスティックスを
獲得/修正しなければならない。

上記事項に関して、ヒューリスティックス知識の観点から基本的検討を行った。また、

知識の構造のデータ依存性の分析やデータフロー解析は、部分問題化（AND並列化）に役立つと考えられる。さらに、上記の（3）は、ある種の効率完全に関する並列ヒューリスティックスの学習であり、部分計算との関連も深い。

[Appendix. 1 ESP向きパラダイムとGHC向きパラダイム]

知識情報処理における問題解決の方法として、「生成検査法（Generate & Test）がある。このパラダイムの基本は、解の候補を生成し、条件に合う解を選ぶことである。一般に、生成空間が大きいときは、少量（また、1つ）を生成し、検査する。この時に、適切な解が得られないときは、バックトラックを行い、その次の解候補を生成する。バックトラックの改善であるインテリジェントバックトラックもあるが、基本的にバックトラックのパラダイムは、ESP向きである。此のような、問題解決をGHC向きにするには、バックトラックの処理を除く必要がある。それには、検査（Tester）部を生成（Generator）部に移動して、知的な生成のみの問題解決にすれば良い。検査部を生成部に移動するには、ヒューリスティックス知識が役立つ。逐次推論のある種のヒューリスティックスは、ESP向き問題解決アルゴリズムをGHC向き問題解決アルゴリズムにするのに効果がある。

[Appendix. 2 推論過程の部分問題]

並列推論を分析するのに、その分析の為の概念が必要である。ここでは、その幾つかを定義する。実際の分析に必要な概念は、これだけでは不十分であろうが、分析の為の第1歩として、紹介しておく。

[推論過程]

推論過程 - (MIUの集合, MIUの推論フロー, 各MIUの参照データ, 各MIUの生成データ（修正を含む）)

最小推論処理単位: MIU (Minimal Inference Unit)とは、そのなかに選択的探索木を含まない、かつ、AND並列性を含まない推論単位を言う。

[推論順序依存性]

MIU (i) がMIU (j) に依存しているとは、MIU (j) が生成するデータ、または、そのデータから導出されるデータ（他のMIUでの推論を経て得られるもの）を参照していることである。

[直接推論依存性]

MIU (i) がMIU (j) に直接依存しているとは、MIU (j) が生成するデータを直接参照していることである。

[完全部分問題]

ある推論過程に含まれる部分プロセスを $MS(i)$ として、それが MIU の集合として定義されるときに、 $MS(i)$ と $MS(j)$ ($i \neq j$)が互いに他方に含まれる MIU 間に共通 MIU がなく、互いに参照することなしにそのプロセスを実行し、求解（成功）するときには、 $MS(i)$ と $MS(j)$ は完全部分問題プロセスであると言う。

〔準完全部分問題〕

ある推論過程に含まれる部分プロセスを $MS(i)$ として、それが MIU の集合として定義されるときに、 $MS(i)$ と $MS(j)$ ($i \neq j$)が互いに他方に含まれる MIU 間に共通 MIU があり、互いに参照することなしにそのプロセスを実行し、求解（成功）することが可能であるが、参照によりどちらかの共通 MIU 処理を省略できるときには、 $MS(i)$ と $MS(j)$ は準完全部分問題プロセスであると言う。

〔不完全部分問題〕

ある推論過程に含まれる部分プロセスを $MS(i)$ として、それが MIU の集合として定義されるときに、 $MS(i)$ と $MS(j)$ ($i \neq j$)が互いに参照することなしにそのプロセスを実行し、求解（成功）することができないときには、 $MS(i)$ と $MS(j)$ は不完全部分問題プロセスであると言う。このときに、 $MS(i)$ は $MS(j)$ を参照しなくても良いが、 $MS(j)$ は $MS(i)$ を参照するときに、 $MS(j)$ は $MS(i)$ に依存した部分プロセスであると言う。相互依存部分プロセス関係もある。