

並行プロセスによる高速仮説推論法

松 尾 豊 石 塚 満

東京大学工学部

筆者らが以前考案した SL 法は、仮説推論問題を、線形計画問題および非線形計画問題に置き換え、2 段階の探索により準最適解を多項式時間で得る。その際に用いた線形計画法、非線形計画法は、各変数、制約をそれぞれプロセッサと考えると、プロセッサのインタラクションとして実現することができる。本論文では、このような視点から、並行なプロセッサによる探索の枠組みを示し、仮説推論問題に関連するいくつかの手法の関係を導く。さらに、仮説推論問題に対し有効な 2 つの手法を示す。ひとつは、SAT 問題等他の問題に対しても効果的な breakout 法と同じタイプのアルゴリズムであり、低次多項式オーダーの探索時間で準最適解を得ることができる。もうひとつは、異なる種類のプロセッサを併用することで、より質の高い解を見つけ出す手法である。

Fast Hypothetical Reasoning System by Parallel Processing

YUTAKA MATSUO and MITSURU ISHIZUKA

School of Engineering, University of Tokyo

Hypothetical reasoning is an important framework for knowledge-based systems because it is theoretically founded and useful for many practical problems. Since the inference time of hypothetical reasoning grows exponentially with respect to problem size, its inefficiency becomes the most crucial problem when applied to practical problems.

In this paper, we develop a new framework for hypothetical reasoning that uses parallel software processors. Our earlier SL method, which can find a near-optimal solution for cost-based hypothetical reasoning in polynomial time (with respect to problem size), uses both linear programming and nonlinear programming techniques. In the new method, these techniques are realized as the interaction of parallel processors. Taking this approach, we may generalize related methods such as the breakout method or Gu's nonlinear optimization method for SAT problems, and introduce two superior algorithms. One algorithm is similar to the breakout method, and the other achieves good-quality solutions by adding new processors during search iterations.

1. はじめに

常に成立する知識（背景知識）に加え、他の知識と矛盾の可能性を持つ仮説という知識を扱い、与えられたゴール（あるいは観測事実）を説明するのに必要な無矛盾な仮説の集合（解仮説）を求める仮説推論は、基盤性と実用性を兼ね備えた知識処理の有用な枠組みである。しかし、仮説推論は非単調推論の一種となるため、推論速度が十分でないことが実用上最も大きな問題となる⁵⁾。

仮説推論と関係の深い SAT 問題では、以前から GSAT を中心とした解法が大きな成果を上げている。GSAT は、山登り法とランダムな再スタートを組み合わせた手法であるが、一回の山登りでより探索可

能性を高めるには、違反した制約の重みを上げていく GSAT extension¹²⁾ や、これと同様な breakout 法⁸⁾、heuristic repair method⁷⁾ などの手法が有効である。さらに、制約充足だけでなく、目的関数も考慮した同様の手法として、DLM(discrete Lagrangian based search method)¹⁴⁾ や GLS(Guided Local Search)¹³⁾ が挙げられ、近年 MAX-SAT 問題などで良好な成果を挙げている。これらに共通するのは、山登り法を基本としながら、探索が局所解に陥った場合に、違反している制約の重みを上げることで探索空間の形を変え、局所解から脱出することである。

制約だけを扱う SAT 問題などでは、制約が緩いときには、制約を満たす解を得るのは簡単であり、また過制約のときには、解がないことを示すのは簡単であ

るが、その中間の領域では解を見つける／解がないことを示すのが難しい。このような phase transition と呼ばれる現象に関する研究も進んでいる²⁾。一方、制約だけでなく目的関数も考慮する必要がある、コストに基づく仮説推論などの問題においては、例えば制約が緩くても最適解を見つけることは難しい。実際、生成可能な仮説が与えられたとき、ゴールを満たすコスト最小の解仮説を求める仮説推論問題は NP 困難である。もちろん、制約が過制約に近いときには、ホーン節を満たす解(可能解)を見つけることも難しくなるので、アルゴリズムの構成にあたっては、なるべく最適解に近い解(準最適解)を提示できること、および制約が厳しい場合にも可能解を提示できることという両面について考慮する必要がある。

筆者らは、以前、コストに基づく仮説推論問題において、線形計画法と非線形計画法を併用する手法(SL法)を提案した⁶⁾。SL法は、低次多項式オーダーの計算時間で準最適解を得ることができ、NBP法⁹⁾など手法に比べ、シンプルで分かりやすい手法となっている。この手法の要点は、0-1 解の条件を緩和した線形計画法を効率的な単体法で解き、この実数最適解を初期探索点とした周囲の近傍探索により、準最適となる 0-1 解を求めることである。近傍探索段階においては、SAT 問題を非線形最適化問題に帰着させる手法⁴⁾に準じて、仮説推論問題を変換した非線形関数上で探索を行う。GSAT などの方法との最も大きな違いは、探索を 0-1 空間ではなく、実数空間で行っていることである。

本論文では、制約や変数をそれぞれプロセッサと考へ、線形計画法や非線形計画法を、プロセッサのインタラクションと捉える。2 章で仮説推論問題を数理計画問題に変換する方法を示した後、3 章では、並行計算の手法である Augmented Lagrangean Method に基づいて、各プロセッサが自分の担当する変数値の更新、メッセージの伝達を行う図式を示す。さらに、このプロセッサのインタラクションの変形として、breakout 法のような違反した制約の重みを上げていくアルゴリズムや、Gu による非線形探索の手法が得られることを 4 章で述べる。5 章では、2 種類の違ったタイプのプロセッサをうまく用いることで、より最適解に近い解を得る手法を示す。この手法では、できるだけ単体法の解の近くを、制約違反に対処しながら探索する。6 章で本論文で述べた手法の比較、考察を述べる。

2. 仮説推論問題の数理計画問題への変換

2.1 線形計画問題への変換

仮説推論問題は、次のように 0-1 整数計画問題に置き換えることができる。

まず、論理変数の真/偽を 1/0 に対応させる。目的関数を、

$$\text{Min. } f = \sum_{i \in N} w_i x_i \quad (1)$$

(但し、 w_i は論理変数 i の重みであり要素仮説でないものは 0、 N は論理変数の集合)とする。また、以下のようにホーン節知識を不等式制約に変換する。 $a \leftarrow b \vee c$ を $x_a \geq x_b + x_c$ に変換する。また、 $a \leftarrow b \wedge c$ を $x_a \geq x_b, x_a \geq x_c$ に変換する*。

ホーン節 j に対するこのような不等式を

$$g_j^{LI}(x) \leq 0 \quad (2)$$

と書くことにする。この制約は、ホーン節を満たすための必要条件となっている。さらに、変数 x_i は、

$$x_i = 0 \text{ or } 1 \quad (3)$$

を満たす必要がある。

制約 (3) を $0 \leq x_i \leq 1$ に緩和することにより、線形計画問題となり、単体法を用い高速に解を得ることができる。得られた解が 0-1 解であれば、仮説推論問題の厳密な最適解となる。例えば 0-1 解とならなくても、その解コストは、0-1 解の解コストの下界値を示し、また、最適な 0-1 解への良い指針となるため、線形計画法を用いるメリットは大きい。

2.2 非線形計画問題への変換

SL 法では、SAT 問題に対する Gu の手法⁴⁾をもとに、仮説推論問題を次のように置き換える。

- 変数 x_i, x_j をそれぞれ $(1 - x_i)^2, x_j^2$ と書き換える**。

- and, or をそれぞれ $+, \times$ に置き換える。

この置き換えで得られた非線形関数 f^{NLP} が最小値(= 0)をとる仮説集合が、可能な解仮説となる。例えば、 $1 \leftarrow g, g \leftarrow a \vee b$ なら、 $f^{NLP} = (1 - x_g)^2 + x_g^2(1 - x_a)^2(1 - x_b)^2$ となる。 f^{NLP} において、ホーン節 j に対応する項を f_j^{NLP} と表すことにする。

2.3 等式制約への変換

以下の章の議論の準備として次のような変換を考え

* 各節に対して完備化を施した後、トップダウン制約は省略して解を得、その後でトップダウン制約も満たす解に修正することができる¹¹⁾。そのため、ここではボトムアップ制約しか考慮しない。以下の議論でも同様である。

** SL 法では、 $[-0.5, 0.5]$ の範囲で探索を行っていたが、ここでは、 $[0, 1]$ の範囲としている。

る。ホーン節知識は、より直接的に等式制約としても表すことができる。

- 各ホーン節ごとに、変数 $x_i, \bar{x}_j$ をそれぞれ $x_i, 1-x_j$ と書き換え、or を $\times$ に置き換える。これを左辺とし、右辺を 0 とする。

例えば、 $1 \leftarrow g, g \leftarrow a \vee b$. なら、 $1-x_g = 0$, $x_g(1-x_a)(1-x_b) = 0$ となる。ホーン節 j に対するこの制約を

$$h_j^{BQ}(x) = 0 \quad (4)$$

と表すことにする。この等式制約は、変数値の丸めを行うことでもとのホーン節を満たす必要十分条件となっている。

3. 並行プロセス

仮説推論問題は、各変数がホーン節制約によってローカルに結び付いている構造をしており、図1のようにネットワークとして表現することができる。さらに、仮説推論問題を変換した線形計画法や非線形計画法は、このネットワーク上での値のやりとりとして表すことができる。ここでは各変数、制約をそれぞれプロセッサと考え、それらがメッセージをやりとりする枠組みを考える。

このような並行なプロセッサによる計算法は、従来から研究されているが、ここでは、制約の違反に対するペナルティを徐々に厳しくしていく乗数法¹⁾の一種であり、制約つき最適化問題の優れた一般解法である Augmented Lagrangian Method¹⁾を取り上げる。なお、本論文では、アルゴリズムの構成を考える上で、並行なプロセッサを想定する方がより直観的であるため、複数のプロセッサを想定しているのであって、実際の計算はすべて単一の計算機で行う。そのため、一般的に並行プロセスを考える際に問題となる通信に関するコストも無視し、ここでは、各プロセッサが必要ときに近傍のプロセッサからメッセージを受け取ることができるものとする。

3.1 Augmented Lagrangian Methods

本節では、Augmented Lagrangian Method について簡単に概説する。まず、以下の制約つき最適化問題を考える。

$$\begin{aligned} &\text{Minimize } F(x) \\ &\text{subject to } h_j(x) = 0 \quad (j \in C) \\ &\quad \quad \quad x \in P, \end{aligned} \quad (5)$$

ただし、 C は制約集合、 P は x の実行可能多面体。これに対する Lagrangian 関数は以下ようになる。

$$L(x) = F(x) + \sum_{j \in C} p_j h_j(x)$$

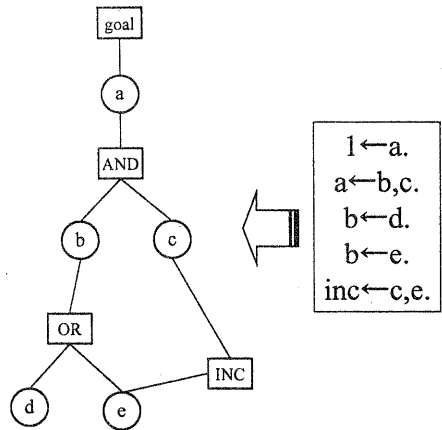


図1 仮説推論問題

ここで、主変数 x_i を担当する変数プロセッサと、双対変数 p_j を担当する制約プロセッサを考える。変数プロセッサは L を減らすように x の値を更新し、制約プロセッサは L を増やすように p の値を更新する。これを反復することで解を得るというのが、並行なプロセッサを用いた主双対アプローチによる最適化の要点である。

しかし、この反復の際、主コスト関数 $F(x)$ が厳密に凸でない場合に双対変数が微分不可能となり扱いづらいので、目的関数を以下のように変更する。

$$F_c(x) = F(x) + \frac{c(t)}{2} \sum_{j \in C} h_j(x)^2$$

$c(t)$ は、反復 t に対して非減少な正の関数である^{**}。第2項は、制約を充足している場合には必ず0となるので、最適解はもとの問題と変わらない。Lagrangian 関数 (Augmented Lagrangean 関数) は以下のようになる。

$$L_c(x) = F_c(x) + \sum_{j \in C} p_j h_j(x)$$

各プロセッサの更新式は以下のとおりである。

変数プロセッサの更新式

$$x_i := \arg \min_{0 \leq x_i \leq 1} L_c(x) \quad (6)$$

制約プロセッサの更新式

$$p_j := p_j + c(t)h_j(x) \quad (7)$$

* 正確には Lagrange 乗数であるが、ここでは簡単に双対変数と呼ぶことにする。

** この $c(t)$ は、収束の速さを左右するので、予備実験により適切な関数に設定する必要がある。一般的には数十回の反復ごとに2~10倍するのがよい¹⁾。

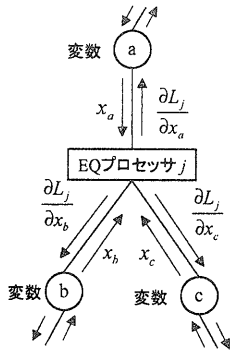


図2 EQ プロセッサのメッセージの伝達

これは、現在の p の値からあまり離れないように、かつ L_c を増加させるように p を更新する式である。

3.2 仮説推論問題への適用

次に、Augmented Lagrangian Method を仮説推論問題に適用する。等式制約 (4) のもとで、目的関数 (1) を最小化する問題を考える。Lagrangian 関数は次のようになる。

$$L_c(x) = \sum_{i \in N} w_i x_i + \sum_{j \in C} p_j h_j^{EQ}(x) + \frac{c(t)}{2} \sum_{j \in C} h_j^{EQ}(x)^2$$

各プロセッサが更新式に基づいて変数を更新する際、必要となる情報はメッセージとしてまわりのプロセッサから伝達される。このメッセージを図2に示す。制約プロセッサ j から変数プロセッサ i へのメッセージは、

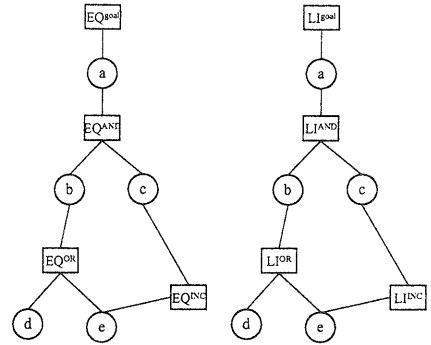
$$\frac{\partial L_j}{\partial x_i} = p_j \frac{\partial h_j^{EQ}(x)}{\partial x_i} + c(t) h_j^{EQ}(x) \frac{\partial h_j^{EQ}(x)}{\partial x_i} \quad (8)$$

である。なお、 L_j は、 L_c における制約 j に関連する項である。各変数は、近傍の制約プロセッサから、この値を受け取り、それらの和に自分の重み w_i を足しあわせ、

$$\frac{\partial L}{\partial x_i} = w_i + \sum_{j \in \text{Neighbor}(i)} \frac{\partial L_j}{\partial x_i}$$

を計算する。 $\partial L / \partial x_i$ が正なら x_i を減らし、負なら x_i を増やす。そして $\partial L / \partial x_i$ が 0 となる (L が最小となる) 値を探し、その値に x_i を更新する。これがひとつの変数プロセッサの一反復の処理である。この処理により、変数 x_i について L_c を最小化していることになる。

一方、制約プロセッサは、近傍の変数プロセッサから受け取った変数値をもとに、式 (7) に基づいて値を更新する。これにより L_c の値は増えることになる。



(a) EQ プロセッサ

(b) LI プロセッサ

図3 制約プロセッサと変数プロセッサ

このように等式制約 (4) に基づいてメッセージを返すプロセッサを EQ (Equality) プロセッサと呼ぶことにする。

また、不等式制約 (2) に基づいて同様のメッセージを返すプロセッサを LI (Linear Inequality) プロセッサと呼ぶことにする。LI プロセッサのメッセージ、更新式は、EQ プロセッサよりもやや複雑であるが、Augmented Lagrangian Method に基づいて、同様に行うことができる。

さて、図3(a)のように、EQ プロセッサと変数プロセッサにより反復を行えば、等式制約 (4) のもとで目的関数 (1) を最小化する探索になる。同様に、図3(b)のように、LI プロセッサと変数プロセッサにより反復を行えば、2.1 節で述べた線形計画問題の解を見つける探索になる。

4. 他手法との関係

EQ プロセッサから変数プロセッサへのメッセージである式 (8) は、変数プロセッサの変数値を更新する際に用いられるが、厳密に式 (8) の値でなくとも、結果的に変数プロセッサの更新した変数値が Lagrangian 関数を小さくするのに貢献すれば良い。

本章では、式 (8) の第 2 項を無視すれば、0-1 空間で探索を行う breakout 法と同じタイプのアルゴリズムになり、また、第 1 項を無視すれば、Gu の非線形最適化と同じタイプのアルゴリズムになることを示す。

4.1 breakout タイプの方法との関係

式 (8) の第 2 項を無視すると、アルゴリズムは次のようになる。

- (1) 変数プロセッサ i は、 L_c を最小化するように変数値 x_i を更新するが、式 (8) の第 2 項を無視しているため、 L_c は x_i に関して線形となり、

結果的に端点である0か1のいずれかの値（もしくは現在の値のまま）を取ることになる。

- (2) EQ プロセッサは、式 (7) に基づいて双対変数値を更新するが、主変数の値が0か1であるので、 $h_j^{BQ}(x)$ は0もしくは1であり、結果的に、まわりの変数がホーン節に違反している場合には p_j を $c(t)$ だけ大きくする操作になる。

より簡単に書けば、変数プロセッサは、まわりの違反制約の重み (p_j) の和が最も小さくなるように変数値をフリップし、EQ プロセッサは、自分の担当する制約が違反している場合だけ制約の重みを大きくする。

これは、GSAT extension や breakout 法、DLM などと同じタイプのアルゴリズムである。実数領域での手法が、結果として0-1空間での探索と等価になるのは興味深い。

この方法では、0-1空間のみを探索するため計算効率が良く、 c を適切に設定することで良い解が得られる。6章で述べるように、実験的にも、探索時間、解の質ともに非常に優れており、仮説推論問題に対しても有効な手法である。

4.2 Gu の非線形最適化問題との関係

一方、式 (8) の第1項を無視すると、双対変数 p が意味を失うことになり、EQ プロセッサは単純に $h_j^{BQ}(x)$ の値にだけ基づいてメッセージを返すようになる。つまり、

$$f = \sum_{i \in N} w_i x_i + c(t) \sum_{j \in C} (h_j^{BQ}(x))^2$$

を目的関数とする無制約最適化問題を解いていることになる。これは、SAT 問題における Gu の手法（もしくは2.2節で述べた最適化問題）と同じタイプになる*。しかし、この方法では、双対変数 p を扱わないため、多くの場合、局所解に陥り、脱出する術がない。SL 法のように、違反しているホーン節に着目し強制的に変数値を決めるなど、局所解から脱出するなんらかの工夫が必要となる。

5. 必要最小 EQ 法

前章で述べた EQ プロセッサにより解を得る方法は、探索可能性が高い優れた方法であるが、双対変数 p の値が大きくなるにしたがって、Lagrange 関数 L ともとの目的関数 f の違いが大きくなり、得られる解のコストが悪くなってしまう。

一方、仮説推論問題を線形計画問題に置き換えたと

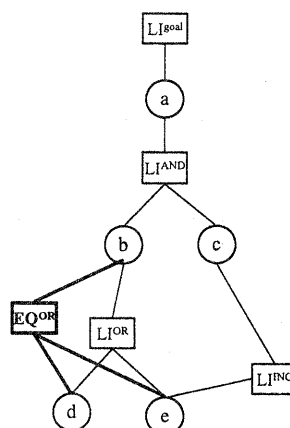


図4 EQ プロセッサを取りつける

き、つまり LI プロセッサを用いた場合には、コスト最小の実数最適解が得られるのであるから、EQ プロセッサと LI プロセッサを併用する次のような方法を考える。

- (1) LI プロセッサにより、線形計画問題の解を得る。0-1 解であれば終了。
- (2) 違反しているホーン節に、EQ プロセッサを取りつける。
- (3) 各プロセッサごとに値の更新を行う。収束すれば、(2) へ。反復中に0-1 解への丸めを行い、実行可能解となれば (4) へ。
- (4) 真となっている要素仮説を一時的に偽としてもゴールの証明が可能か試みる。成功したらこの仮説と証明試行中に偽となるノードを偽とする。

この方法は、線形制約 (LI プロセッサ) による実行可能空間の内部を、ホーン節に違反している変数を EQ プロセッサでガイドしながら探索する方法であり、できるだけ少ない数の EQ プロセッサだけを用いるため、必要最小 EQ 法と呼ぶことにする。

上記アルゴリズム中 (2) の EQ プロセッサを取りつける段階では、最も $h_j^{BQ}(x)$ の値の大きい（すなわち違反度の高い）制約1つに対し、EQ プロセッサを取りつける（図4参照）。6章で示すように、すべての制約に EQ プロセッサを取りつける方法よりも良い解が得られる。

この方法の特徴として次のような点が挙げられる。

- 収束ごとに EQ プロセッサを加えていくが、変数値を決定しているわけではないので、バックトラックの必要はなく、また、解空間を狭めているわけでもない。

* もちろん、 $\sum_{i \in N} w_i x_i$ の項がある点で異なるが、Gu の方法で目的関数を考慮すれば同じになる。

- 各プロセッサにおいて、前回の収束時の変数値を利用することができるため、2回目以降のプロセッサ間のインタラクションは速やかに収束する場合が多い。また、最初にLIプロセッサによる実数最適解を求める段階(1)は、単体法を用いることで高速に解を得ることができる。

この方法では、得られる解が厳密な最適解である保証はないが、取りつけるEQプロセッサを必要最小限とすることで、より実数最適解に近い0-1解が得られる。LIプロセッサを反復的に使用し、実数最適解に近い0-1解を探索するという点では、単体法の反復適用¹⁰⁾と同じ考えに基づいている。

6. 実験結果と評価

本論文で述べた各手法に対し、実験を行った。システムはC言語で記述し、SGI Onyx上で実行した。扱う問題は、ランダムに生成した問題であり、各ホーン節本体のATOM数は2~7個、どのATOMも知識ベース上での出現数の上限は10としている。

矛盾制約を含まない問題^{*1}と、ホーン節のうち18~30%が矛盾制約である問題^{*2}の2種類について、それぞれノード数60~1000の範囲の271問について、実験を行った。

用いた手法は以下の通りである。

LP / EQ 単体法の解から探索を開始。すべての制約にEQプロセッサを取りつける。

0/EQ すべての変数値を0とし探索を開始。すべての制約にEQプロセッサを取りつける。

SL法 単体法の解から探索を開始。GuタイプのEQプロセッサと局所解からの脱出機構を用いて探索を行う。

LP/ breakout 単体法の解から探索を開始。すべての制約にbreakoutタイプのEQプロセッサを取りつける^{*3}。

0/ breakout すべての変数値を0とし探索を開始。すべての制約にbreakoutタイプのEQプロセッサを取りつける。

LP/ LI+EQ 単体法の解から探索を開始。すべての制約にLIプロセッサとEQプロセッサの両方を取りつける。

LP/ LI+leastEQ 単体法の解から探索を開始。すべての制約にLIプロセッサを取りつけ、さらに収束ごとに1つEQプロセッサを追加する。

探索時間を図5,6に示す。また、解コストに関する結果を表1,2に示す。

探索時間に関しては、いずれの方法も多項式オーダー^{*4}であるが、最もオーダーが大きいのが単体法の計算時間であり、図5,6からはノード数に対して約2.2乗である。単体法以後の探索時間はSL法では約1.8乗、breakoutタイプが約1.25乗、LI+EQやLI+leastEQが約1乗であるが(別図より算出)、初期点として単体法を使う場合には、問題規模が大きくなると単体法の計算時間が支配的になるため、実質的に2.2乗のオーダーとなると考えられる。

従って、2.2乗以下の多項式オーダーの解法を実現するには、少なくとも単体法の初期値を用いては不可能である。単体法の初期値を用いない方法のなかで、最もよい結果であったのが、0/breakoutタイプであり、1.25乗という低次多項式オーダーでありながら、SL法と同程度かそれ以上の解が得られている。

また、単体法の初期値を用いるとすると、どの方法も単体法以後は2.2乗以下のオーダーなので、解コストの良否と探索可能性が焦点となる。解コストの面から最もよい解法は、必要最小EQ法(LP/LI+leastEQ)である。理論的にも、実数最適解から最も離れにくい解法となっているため、これは当然の結果であると考えられる。

以上をまとめると、手法として優れているのは、breakoutタイプ(単体法なし)もしくは、必要最小EQ法であり、この両者にはトレードオフがある。

7. おわりに

文献⁵⁾では、探索問題を0-1空間ではなく実数空間で探索する方が、勾配を利用することができるため有望であると予見されている。最近、制約違反に対して重みを増やすアプローチが成功を納めているが、これは、いわば双対空間での探索であり、実数空間での探索のひとつの形と言えるのではないだろうか。本論文では、仮説推論問題に対し、並列なプロセッサとい

^{*1} すべての要素仮説を真とすればゴールは証明できるため、可能解が少なくとも1つは存在する

^{*2} ランダムに生成した問題のうち約27%でいずれの手法においても可能解が発見できなかった。

^{*3} なお、変数値が実数値をとるのは最初の反復だけである。以後は0-1空間での探索となる。

^{*4} 厳密には、ある一定以上の時間で解が得られない問題を探索失敗として扱うために、多項式オーダーとなる。実際には³⁾に示されているように、非常に長く計算時間のかかるケースまで考慮すると、多項式オーダーとは言えないかもしれない。このようなオーダーの議論は、問題の構造や比率、探索失敗をどう扱うかによって変わってくるが、ここでは、今までの議論と同様に、実用的にどのようなオーダーになるか着目し、グラフから一般的にオーダーが多項式となることを示した。

表 1 矛盾制約を含まない問題

	0/EQ	LP/EQ	SL 法	0/breakout	LP/breakout	LP/LI+EQ	LP/LI+leastEQ
探索失敗	0	0	0	0	0	0	0
解コスト平均	115.43%	105.74%	100%	98.88%	95.92%	93.85%	92.37%
SL 法との勝-敗-引分	50-167-44	57-114-100	0-0-271	87-91-93	95-56-120	93-31-147	110-22-139

※ 271 問中

表 2 矛盾制約を含む問題

	0/EQ	LP/EQ	SL 法	0/breakout	LP/breakout	LP/LI+EQ	LP/LI+leastEQ
探索失敗	73	73	119	73	73	73	73
解コスト平均	119.33%	105.01%	100%	99.71%	98.49%	96.84%	95.88%
SL 法との勝-敗-引分	21-89-42	29-55-68	0-0-271	41-41-70	42-30-80	45-23-84	46-22-84

※ 271 問中

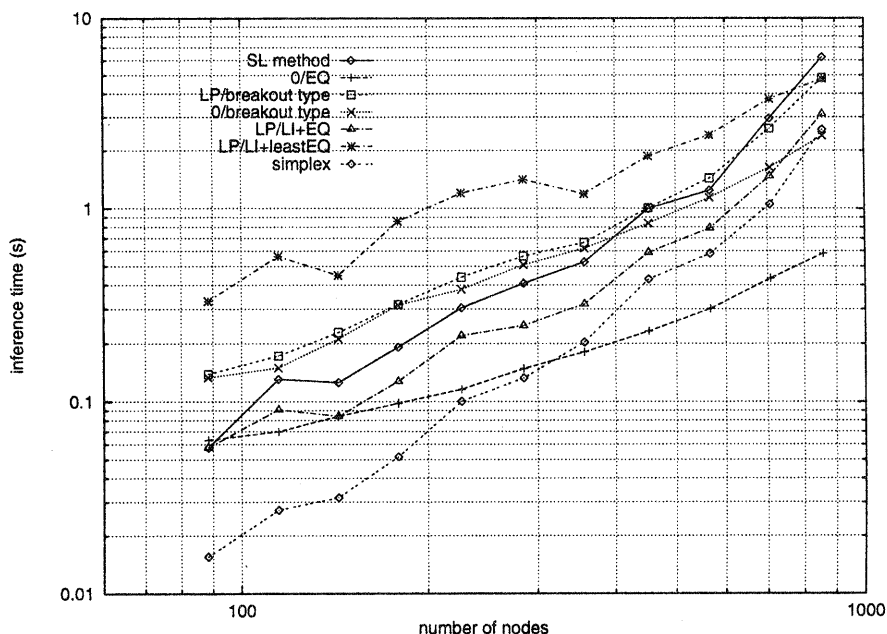


図 5 探索時間 (矛盾制約を含まない問題)

う視点からそれらのアルゴリズムの関係をまとめ、仮説推論問題に対して有効な 2 つの方法を示した。1 つは、breakout タイプであり、これは SAT や CSP など他の問題にも同様の解法が多く提案されており、仮説推論問題にも有効であることを確認することができた。もう 1 つは、線形計画問題にひとつずつ等式制約を足していく方法であり、その解コストが非常に良いのが特徴である。

今後は、MAX-SAT や PCSP 等、他の問題へこのような方法が適用できるか検討していく予定である。また、ランダムに生成した仮説推論問題自体の性質を調べることも、探索手法をより正確に評価する上で必要である。

参 考 文 献

- 1) D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation*. Prentice-Hall, 1989.
- 2) C. Gomes and B. Selman. Problem structure in the presence of perturbations. *Proc. AAAI-97*, pp. 221-227, 1997.
- 3) C. Gomes, B. Selman, and N. Crato. Heavy-tailed phenomena in satisfiability and constraint satisfaction problems. *J. of Automated Reasoning*, Vol. 24, pp. 67-100, 2000.
- 4) J. Gu. Global optimization for satisfiability problem. *IEEE Trans. on Knowledge and Data Engineering*, Vol. 6, No. 3, pp. 361-381, 1994.
- 5) 石塚満. 知識の表現と高速推論. 丸善, 1996.

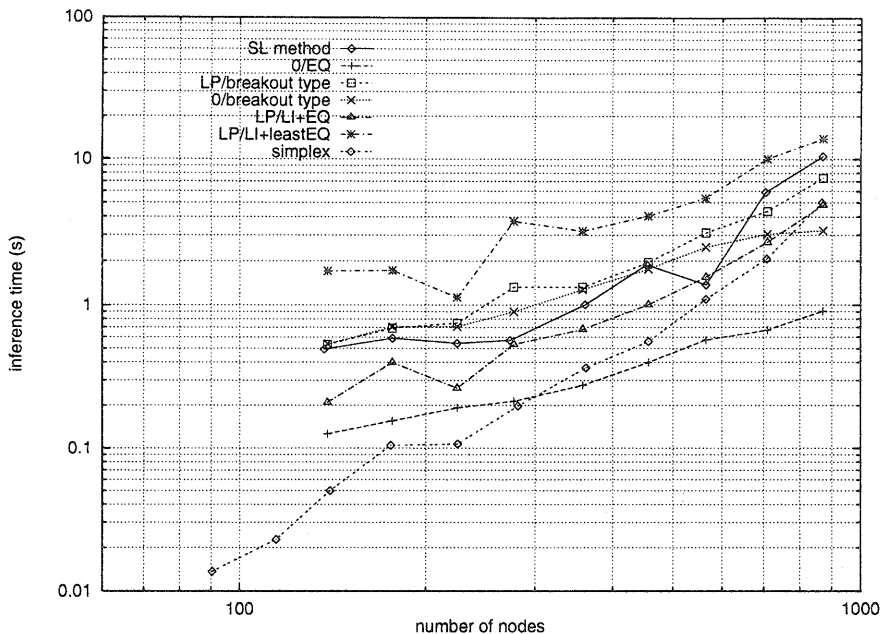


図 6 探索時間 (矛盾制約を含む問題)

- 6) 松尾, 二田, 石塚. SI 法 : 線形・非線形計画法の併用によるコストに基づく仮説推論の準最適解計算. 人工知能学会誌, Vol. 13, No. 6, pp. 953-961, 1998.
- 7) S. Minton, M. D. Johnston, A. B. Philips, and P. Laird. Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, Vol. 58, pp. 161-205, 1992.
- 8) P. Morris. The breakout method for escaping from local minima. *Proc. AAAI-93*, pp. 40-45, 1993.
- 9) Y. Ohsawa and M. Ishizuka. Networked bubble propagation: A polynomial-time hypothetical reasoning method for computing near-optimal solutions. *Artificial Intelligence*, Vol. 91, pp. 131-154, 1997.
- 10) 斎藤, 石塚. 単体法の反復適用による不完全制約充足問題の解法の研究. 博士学位請求論文, 東京大学, 2000.
- 11) E. Santos Jr. A linear constraint satisfaction approach to cost-based abduction. *Artificial Intelligence*, Vol. 65, pp. 1-27, 1994.
- 12) B. Selman and H. Kautz. Domain-independent extensions to gsat: solving large structured satisfiability problems. *Proc. IJCAI-93*, pp. 290-295, 1993.
- 13) C. Voudouris and E. P. K. Tang. Guided local search. Technical report csm-247, University of Essex, UK, 1995.
- 14) B. W. Wah and Y. Shang. Discrete lagrangian-based search for solving max-sat problems. *Proc. IJCAI-97*, pp. 378-383, 1997.