特　集　「定理証明，推論関係の新技術」

CNF充足可能性判定問題の計算複雑さ
—最近の発展—

Computational Complexity of CNF Satisfiability Testing
— Recent Developments —

岩間　一雄*1　京都大学大学院情報学研究科
Kazuo Iwama　School of Informatics, Kyoto University.
itota@jaist.ac.jp, http://www.jaist.ac.jp/~itota/

Keywords: CNF satisfiability, local search, randomized algorithms, probabilistic analysis.

1. はじめに

命題論理式の**充足可能性問題**（SAT）が理論的・実用的に重要な問題として多くの研究者の注目を集めているのは紛れもない事実である．しかし，この問題は1種類とはいえず，応用やその他の事情によって三つのタイプに分類されているようである．一つは最も基本的なもので，与えられた和積形の論理式（**CNF論理式**）に対してその項をすべて真にするような変数割当て（**充足割当て**）を求める問題である．しかし，そのような充足割当てが存在しない場合や，存在しても実用的な時間以内で求められないという観点から，「すべて」でなくてもよいから「できるだけ多く」の項を真にするような割当てを求める問題が**最大充足問題**（MAXSAT）である．さらには，充足割当てを求めることが目的ではなく，そのような充足割当てが存在しないことを効率良く証明することが目的のときは，**定理の自動証明**と呼ぶことが多く，やはり古くから多くの研究がある．

MAXSATについてはそれが最適化問題であることから近似アルゴリズムの研究が進み，例えば，よく知られた8/7近似アルゴリズム [Johnson 74] が最適である（つまり近似度をそれ以上改良できない）ことが最近証明された [Hastad 97]．半正定値計画緩和を利用することも有力で（例えば [Goemans 95]），かなり研究が進んでいるといってよいであろう．また，定理の自動証明に関しては，実用的な観点から発見的手法によって短い証明を探す研究が古くから多くの研究者を魅了してきた．しかし，本稿のテーマである計算複雑さの観点からは，そのような「短い」証明が存在しない例を発見することが60年代からの重要なテーマであった（理論研究者の性というべきかもしれない．本誌の読者とは相容れないであろう）．仮に常識的な証明システムにおいて常に短い証明が存在すれば，それはNP = coNPを意味するので，P = NPに匹敵するくらいの大事件なので妥当な研究課題といえる．最初は簡単なことのように見えてスタートした研究であったが，下限の証明の困難さが端的に現れたよい例で，20年以上の間成功しなかった．ようやく，Hakenが85年に代表的な証明システムである導出原理の指数下限（つまり，証明のステップが指数的に増大してしまうような論理式の例が存在する）を証明してから [Hakem 85]，同様な下限や異なった証明系の能力の違いを示す結果が続々と現れてきている（しかし，依然としてFregeシステムなどでは指数下限が示されていない）．これら二つの話題も非常に興味深いが今回は取りあげない．

本稿で取りあげるのは，第1の話題，つまり完全にすべての項を充足する割当てを求める問題である（以後，SATといったらこれを指すことにする）．三つの中でも最も基本的な問題で，その重要性はいうに及ばないが，1点だけ注意しておく．それはMAXSATとの違いである．実用的には全部の項を完全に充足させる必要はなく，「大部分」で問題ないだろうと考える人が多いのではなかろうか．これは以下のような理由で必ずしも正しくない．実用的な問題（例えば，大学の時間割を作成する問題）をSATの形で書くことはそれほど難しくない．その場合，各項は満たすべき条件（例えば，ある先生は1時間目の授業を嫌うのでその先生の授業は1時間目に割り当てない，つまり割り当てるとその項が偽になるような式をつくる）になる．全部の項を真にできないということは，満たされていない条件があるわけである．例えば，上のような1時間目を云々という条件は満たされなくても本質的不都合は招かないかもしれないので，満たされない条件をできるだけ少なくするという意味でMAXSATは有効である．しかし1人の先生が二つの講義を担当している場合それらの講義が同じ時間帯に割り当てられないという条件は本質的で，たった一つの条件

*1 科学技術振興事業団・創造科学技術推進事業（ERATO）今井量子計算機構プロジェクト併任．

でもそれが満たされないと全体として正当な時間割にならない．つまりすべての条件を満たすことが本質的な場合もあるのである（より詳しくは [Cha 97] を参照されたい）．

SATのアルゴリズムに関しては膨大な論文がある．AIの分野では実験的に高速なアルゴリズムが好まれ，特に局所探索法は人気のあるアプローチである（例えば [Cha 95]）．ただ，このアプローチの弱点は，解析が困難で計算複雑さの理論的結果がほとんど得られていない点であった．最近Schöningが見事な解析を行ったので [Schöning 99] その紹介を中心にして解説する．なお，我々によるある種の改良 [Iwama 01] についても簡単に触れる．さらには，理論的な計算複雑さという観点からは若干外れるが，実用的観点からの局所探索法の研究動向を簡単に解説する．

n 変数の式の充足解を求めるには n 変数に対するすべて偽（以下では0を使用する）からすべて真（1を使用）までの 2^n 通りの割当てをチェックすれば明らかに求まる（あるいは解が存在しないことがわかる）．この手法の計算量はおよそ 2^n である（実際はこれに多項式の係数が付くが慣例に従って無視する）．この 2^n を c^n（c は2より小さい定数）に改良するのがとりあえずの目標である．これは与えられた式がいわゆる kSAT 式の場合はそれほど困難ではない．例えば，$k=3$ の場合で，バックトラック法で解くことを考えてみよう．バックトラック法（厳密には違うのかもしれないが，Davis Putnam法と呼ばれることも多い）は，要するに，変数を選んでそれに0と1を代入して分岐し，ある項が0になるか全体が充足されるまで木状に展開していく．ここで，もし根から葉へのすべてのパスで n 変数すべてに対して2分岐が行われれば，木のサイズは 2^n になる．しかし，3SATであることを考えるとこのことは起こりずらい．理由は簡単である．一つの項は3個のリテラルしか含まないので，それらのうち2個の値が0に決まれば残りの1個は1にしなければならないので分岐は生じない．値を決める変数は木の各頂点で自由に選べるので，上手に選べば多くのパスで定数分の1程度の頂点で分岐が生じないようにできる．つまり，木のサイズを1より小さな c に対して 2^{cn} にできるのである．Schöningの成果は局所探索に対する最初の解析であるばかりでなく，このような意味での計算量 1.334^n が現時点での世界最速になっているのである．

なお，SAT関連のwebサイトとしては，SATLIB（http://www.intellektik.informatik.tu-darmstadt.de/SATLIB/）やSatEx（http://www.lri.fr/~simon/satex/satex.php3）がある．

2. Schöningのアルゴリズム

SATに対する局所探索法は一般に以下のような図式となる．与えられた n 変数の kCNF論理式 F に対して，(i) 初期割当て a を $\{0,1\}^n$ から適当な方法で（例えばランダムに）選ぶ．(ii) a を F に代入してすべての項が1になるかどうかを調べる．すべて1になっていれば終了．(iii) そうでなければ a の1個の変数の値をフリップ（0なら1，1なら0に変える）した近傍を調べて，現在の a よりも「より良い」近傍 a' を探す．現在の a を a' に変えて (ii) へ．ただし，あらかじめ設定した条件が満たされる場合は，(i) に戻って新たな探索を開始する．

Schöningのアルゴリズムも完全にこの図式に従っている．まず，(i) の初期割当ては完全に一様ランダムに選ぶ．また (iii) の近傍探索については，現在の割当てで充足されていない項を任意に選び，その項の k 個のリテラルからランダムに一つ選んでその値をフリップする．$3n$ 回この操作を繰り返した後は (i) に戻って新たな探索を開始する．(iii) の近傍探索を例で説明しよう．例えば現在の割当てが $x_1=1, x_2=0, x_3=1, x_4=0, \cdots$ であったとしよう．すると例えば，F の $(\overline{x_1}+x_2+x_4)$ という項が0になる．この項に含まれる3変数の中から1変数をランダムに選んで（例えば x_2）その値をフリップする（0を1に変える）のである．明らかに現在の割当てで充足されていないこの項は新しい割当てでは充足されるので「より良い」といってもよいであろう（ただし，この操作によって現在充足されている項が非充足になってしまう可能性はある）．近傍探索の方法としてはけっして新しいものではなく，古くから使用されている比較的人気の高いものである．

Schöningの論文の真価はその解析にある．まず，以下の基本的事実を確認してほしい．上で述べた $3n$ 回の繰返しをアルゴリズムの1回の実行と考えると，1回の実行で充足解を見つける確率 p は，ある n の関数 $T(n)$ を用いて $p \geq 1/T(n)$ と評価できる（パラメータとしては n しか現れていないことに注意）．したがって，定数 N に対して，$N \cdot T(n)$ 回実行しても充足解を見つけることができない確率はたかだか $(1-1/T(n))^{N \cdot T(n)}$ である．この確率は充足可能であるにもかかわらず，充足不能であると誤って判断してしまう確率であり，この式からわかるように e^{-N} でおさえられる．N を十分大きくすればこの誤り確率を任意に小さくすることができる．ゆえに，このアルゴリズムの必要な実行回数は $1/p$ と評価してよい．

問題はこの確率 p を求めることである．入力として充足可能な論理式と，それに対する充足割当て a^* を一つ固定する．確率変数 X はランダムな初期値 a と充足解 a^* のハミング距離を表すものとする．つまり a と a^* の異なるビットの数を表す確率変数である．X は2項分布に従う．すなわち $Pr(X=j)=\binom{n}{j}2^{-n}$，$(j=0,\cdots,n)$ である．アルゴリズムの過程はマルコフ連鎖であると考えることができる．すなわち次のような状態をある確率で遷移しながら，アルゴリズムは探索をすすめていく．

状態としてはアルゴリズムの開始時点である初期状態と，a と a^* とのハミング距離に対応している $j \in \{0, 1, \cdots, n\}$ がある．初期状態から $0, 1, \cdots, n$ のどれかの状態に遷移することは初期値をランダムに選ぶことに対応している．初期状態から状態 j に遷移する確率はさきほど求めたように，$\binom{n}{j}2^{-n}$ である．状態 0 に遷移するということは充足解を見つけたことを意味する．

ある節 C が割当て a によって充足されていないとしよう．すると C に含まれるリテラルの現在の値の中で少なくとも 1 個は a^* と異なるものが存在する（全部同じなら充足されているはずである）．したがって，充足されていない k-CNF の節から一つのリテラルをランダムに選び，その値をフリップするということは，現在の状態 j から状態 $j-1$ に遷移する確率（つまり a^* と異なっているリテラルを「当てて」それをフリップして正解とのハミング距離が 1 減少させる確率）が少なくとも $1/k$ であり，状態 j から状態 $j+1$ に遷移する確率（つまり「外れ」てしまって a^* と合致しているビットを反転させてしまう確率）はたかだか $(k-1)/k$ であることを意味する．

いま初期状態から状態 j に遷移したとする．このときアルゴリズムが状態 0 に到達する確率 q_j を評価したい．状態 0 に到達するためには j ステップ必要である．もし i 回の誤った方向への遷移を行うとすると，アルゴリズムは $j+2i$ ステップ後に状態 0 に到達することになる．このような遷移のしかたの総数は $\binom{j+2i}{j}\cdot j/(j+2i)$ で与えられる（これは自明ではない．単純に $j+2i$ から i を選ぶと最後が外れのものや，最後が当たりでもその直前の二つが外れのものなどを含んでしまう．これらは当然除外されるべきである）．したがって求める確率は次のように評価できる．

$$q_j \geq \sum_{i=1}^{j}\binom{j+2i}{i}\cdot\frac{j}{j+2i}\cdot\left(\frac{k-1}{k}\right)^i\cdot\left(\frac{1}{k}\right)^{i+j}$$

$$\geq \frac{1}{3}\cdot\sum_{i=0}^{j}\binom{j+2i}{i}\cdot\left(\frac{k-1}{k}\right)^i\cdot\left(\frac{1}{k}\right)^{i+j}$$

この値は j の指数関数になっており，和全体を和の中の最大値で置き換えても,多項式倍しか異ならない（いずれにしても q_j は下からおさえているので右辺を小さめに見積もることは問題ない）．最大値を求めるために 2 項係数に関する以下の関係を用いる．

$$\binom{n}{\alpha n}\sim 2^{h(\alpha)n}=\left(\frac{1}{\alpha}\right)^{\alpha n}\left(\frac{1}{1-\alpha}\right)^{(1-\alpha)n}$$

ここで，$h(\alpha)=-\alpha\log_2\alpha-(1-\alpha)\log_2(1-\alpha)$ である．次の二つの関数は多項式倍の違いを除いて等しい（$i=\alpha j$ と置いていることに注意）．

$$\binom{(1+2\alpha)j}{\alpha j}\sim\left\{\left(\frac{1+2\alpha}{\alpha}\right)^{\alpha}\cdot\left(\frac{1+2\alpha}{1+\alpha}\right)^{1+\alpha}\right\}^j$$

以下の式の 2 行目は $\alpha=1/(k-2)$ のときに最大になることがわかるので，

$$q_j \geq \frac{1}{3}\cdot\sum_{i=0}^{j}\binom{j+2i}{i}\cdot\left(\frac{k-1}{k}\right)^i\cdot\left(\frac{1}{k}\right)^{i+j}$$

$$\geq\left\{\left(\frac{1+2\alpha}{\alpha}\right)^{\alpha}\cdot\left(\frac{1+2\alpha}{1+\alpha}\right)^{1+\alpha}\cdot\left(\frac{k-1}{k}\right)^{\alpha}\cdot\left(\frac{1}{k}\right)^{1+\alpha}\right\}^j$$

$$=\left(\frac{1}{k-1}\right)^j$$

と評価できる．以上まとめて以下の補題を得る．

[補題 1] 充足可能な k-CNF の論理式に対しその充足解 a^* とのハミング距離が j である初期値 a から局所探索を開始したとき，アルゴリズムが a^* を見つける確率 q_j について以下が成り立つ．

$$q_j \geq\left(\frac{1}{k-1}\right)^j$$

したがってアルゴリズムが a^* を見つける確率はこの章の初めに述べたように，

$$p\geq\left(\frac{1}{2}\right)^n\sum_{j=0}^{n}\binom{n}{j}\left(\frac{1}{k-1}\right)^j$$

$$=\left(\frac{1}{2}\left(1+\frac{1}{k-1}\right)\right)^n$$

と評価できる．

[定理 1] 充足可能な n 変数の k-CNF の論理式に対しアルゴリズムが 1 回の実行で成功する確率 p は以下の式を満たす．

$$p\geq\left(\frac{1}{2}\left(1+\frac{1}{k-1}\right)\right)^n$$

したがって，アルゴリズムは以下の計算時間（実行回数）で，高い確率で充足解を（もしあるなら）発見することができる．

$$\left\{2\left(1-\frac{1}{k}\right)\right\}^n$$

例えば $k=3$ のときはこの計算時間（何度もいっているように多項式の係数は無視）は $(4/3)^n=1.334^n$ となる．2^n より真に速いアルゴリズムは数多く報告されている（例えば，[DGH+ 00, PPSZ 98, PPZ 97]）がこの 1.334^n は最速である．ごく最近，その程度は大きくないが一般的改良に成功したという報告がある [SSW 01]．なお，Schöning は，SAT よりも一般的な制約充足問題（CSP）に対しても同様な解析を行っているが，省略する．

3. アルゴリズムの拡張

Schöning の成果は簡潔でわかりやすかったため，多くの研究者がその改良の可能性を追求したようすである．しかし，そのアルゴリズムおよび解析は大変良くできていて，一般的改良は容易ではないことがわかった．

著者らは一般的改良ではないが，ある種の例題に対しては格段に高速化できる手法を開発したので簡単に紹介する．

前章の解析において，初期値と充足解のハミング距離が j であるとき局所探索が成功する確率は Lemma 1 より

$$q_j \geq \left(\frac{1}{k-1}\right)^j$$

であった．我々はまずこの式を次の補題のように一般化した．

［補題 2］ 変数 x_i に正しい値を割り当てる確率が t_i，正しくない値を割り当てる確率が f_i であるとする．このときアルゴリズムが 1 回の実行で成功する確率 p について以下の式が成立する．

$$p \geq \prod_{i=1}^{n}\left(t_i + \frac{f_i}{k-1}\right)$$

Schöning のアルゴリズムでは $t_i = f_i = 1/2$ であった．この $(t_i + f_i/(k-1))$ を大きくする方法を考えたい．そのために充足解について次のことがわかっているとしよう．つまり充足解 a^* に現れる 0 と 1 の個数が l と $n-l$ であるとわかっているとする（0 と 1 の個数がバランスしていないという事実が重要なのであって，そのアンバランスの度合いが事前にわかっているかどうかは余り重要ではない．なぜなら，適当なアンバランスの度合いを仮定してアルゴリズムを実行しても多項式程度の時間を損するだけだからである）．このとき初期値を一様ランダムに発生させるのではなく，ある確率分布に従って発生させるとどうなるかを考える．$p_0 = l/n$, $p_1 = (n-l)/n$ とし，初期値に 0 と 1 をそれぞれ確率 q_0, q_1 で発生させたものを割り当てる．

x_i は 0 を割り当てるべき変数であるとすると，初期値として正しい値が割り当てられる確率が q_1，間違った値が割り当てられる確率が q_0 である．すなわち $t_i = q_0$, $f_i = q_1$ である．x_i が 1 を割り当てるべき変数であるときも同様でこのときは $t_i = q_1$, $f_i = q_0$ である．したがって，Lemma 2 より次のことを示すことができる．

［定理 2］ 充足可能な n 変数の k-CNF の論理式でその充足解の 0 の個数が $p_0 n$, 1 の個数が $p_1 n$ であるものに対し，初期値として各変数に 0 と 1 をそれぞれ確率 q_0, $q_1 = 1 - q_0$ で割り当てたとき，アルゴリズムが 1 回の実行で成功する確率 p について以下の式が成立する．

$$p \geq \begin{cases} p_0^{p_0 n} p_1^{p_1 n}\left(\dfrac{k}{k-1}\right)^n & \text{for } \dfrac{1}{k} \leq p_0 \leq \dfrac{k-1}{k}, \\ \left(\dfrac{k}{k-1}\right)^{\min\{p_0 n, p_1 n\}} & \text{for } p_0 < \dfrac{1}{k} \text{ or } p_0 > \dfrac{k-1}{k}. \end{cases}$$

ただし

$$q_0 \geq \begin{cases} 1 & \text{for } p_0 < \dfrac{1}{k}, \\ \dfrac{kp_0 - 1}{k-2} & \text{for } \dfrac{1}{k} \leq p_0 \leq \dfrac{k-1}{k}, \\ 0 & \text{for } p_0 > \dfrac{k-1}{k}. \end{cases}$$

例えば 3SAT（$k=3$）の場合を見てみよう．$p_1 = 1/3$ のときは計算時間は 1.260^n，また $p_1 = 0.1$ のときは 1.072^n となって効果は大きい．注意すべきは上の定理における p_0 の値（充足解の 0 と 1 のアンバランス）と p_0 の値（初期値として割り当てるときの 0 と 1 のアンバランス）の違いである．例えば $p_0 = 0.6$ のときは $q_0 = 0.8$ となり，$p_0 \geq 2/3$ のときは $q_0 = 1.0$ である．すなわち，割当てのアンバランスの度合いは充足解のアンバランスの度合いよりかなり大きくするのがよい．

それではどんな場合にこのような 0 と 1 のアンバランスが生じるのであろうか．これは，他の問題を SAT に自然な方法で変換した場合には，むしろ普通に生じるというべきである．例えば，[Cha 97] で議論されている大学の時間割作成問題でも充足解の 0 と 1 の個数の間には大きな差があるし，SAT が NP 完全であると最初に証明した Cook の論文 [Cook 71] における変換でも明らかに偏りがある．[Iwama 01] ではより抽象的な組合せ問題の例を示している．それは 3 次元マッチングと呼ばれる問題で，kSAT に変換した例題に上で述べた充足解の偏りを利用した解法を適用することによって，直接解く場合よりも計算量を非自明に減少できることが示されている．

4. 実用的な局所探索アルゴリズム

局所探索法で現在の割当ての近傍を探すとき，現在の割当てよりも「より優っている」割当ての条件は，上記 Schöning では，現在の割当てで充足されていない項を充足させることであった．しかし，このように決めた新しい割当ては以前の割当てで充足されていた項を非充足にしてしまうことも当然あって，充足する項の数を増やすとは必ずしもいえない．そこで，より強い条件として，充足する項の数を増加させるという条件も考えられる（新たな項を充足させられないなら，明らかに充足する項は増えないのでこれは Schöning の条件より真に強い）．素直に考えるとこちらのほうがより自然である．

SAT に局所探索を利用することは自然な考えに思えるのであるが，最初の発表（実は若干の論争があるのであるが）は意外に新しく，[Selman 92] であるとされている．Selman らはこの論文で，近傍探索としては上記の充足項を増やすような割当てを探し，局所最適解（近傍がいずれも現在の割当てより改善されない）に陥ったときは，その近傍の中で最も「まし」な割当て（つまり充足項の多い割当て）をランダムに選んでそこに移動す

表1 各種SATアルゴリズムの比較

Instance Name	# variables	# clauses	Runtime (Sec.)								
			Our GSAT	Selman's GSAT	Other SAT programs						
					1	2	3	4	5	6	7
aim-100-2 0-yes1-1	100	200	1	227	0	17	135	2	398	N.A.	1
aim-100-2 0-yes1-2	100	200	2	96	0	29	2	5	239	13,929	1
aim-100-2 0-yes1-3	100	200	1	49	0	13	17	1	63	22,500	1
aim-100-2 0-yes1-4	100	200	2	226	0	15	0	0	1,456	N.A.	0
f400.cnf	400	1,700	3	48	2,844	34		5,727	210,741	60	10,870
f800.cnf	800	3,400	182	N.A.		1,326				27,000	
f1600.cnf	1,600	6,800	*509	N.A.						N.A.	
f3200.cnf	3,200	13,600	*19,840	N.A.						N.A.	
g125.17.cnf	2,125	66,272	261	N.A.		103,310			N.A.	453,780	N.A.
g125.18.cnf	2,250	70,163	17	4		126			N.A.	480	N.A.
ii32b3.cnf	348	5,734	15	55	2	4	4	1	1	5,400	17
ii32c3.cnf	279	3,272	8	3	1	3	0	5	1	12,180	
ii32d3.cnf	824	19,478	38	25	973	19	10	3	20	1,200	N.A.
ii32e3.cnf	330	5,020	11	0	1	3	4	1	3	3,900	3
par16-2-c.cnf	349	1,392	183	N.A.	23		329	48	1,464	N.A.	145
par16-4-c.cnf	324	1,292	554	N.A.	6		210	116	1,950	N.A.	145
ssa7552-159.cnf	1,363	3,032	*30	N.A.	1	82	6	1	1	N.A.	1
ssa7552-160.cnf	1,391	3,126	*40	N.A.	1	86	6	1	23	175,500	1

るというアルゴリズムを提案した．このアルゴリズムはGSATと呼ばれ，SATに対する局所探索法の代名詞になるほど人気を集めた．

その後続々と改良・新提案が続いた．例えば，本稿の主題であったSchöningの近傍探索条件はランダムウォークという名前で呼ばれ，[Selman 93b] に登場している．また，局所解に陥ったときにいかにして脱出するかが主要な研究課題になった．例えば，GSATのように現在より質の悪い割当てに移動すると，次にまたもとに戻ってしまうことが考えられる．これを防ぐためにタブー表を利用することは自然な考え方であろう．一つおもしろいのは，局所解に陥ったなら，与えられた式に変形を加えてその割当てを局所解でなくしてしまうという考えである．これは以下のアイディアで実現できる．局所解ということは，直観的には，その割当てaの周辺の部分のほうがより多くの項で覆われている（その割当てで0になる項をその割当てを覆う項と呼んでいる）ということである．したがって，aを覆う項を増やしてやれば局所解は解消されることが予想される．式の充足解を変化させないという条件のもとでこのことを行う最も簡単な方法は，式にすでに存在しているaを覆う項を（同じものを）追加することである（もちろん実際には項に重みを定義し，その重みを増大させてやればよい）．この方法は重み付け法と呼ばれ [Morris 93, Selman 93a]，非常に有効であることが知られている．

局所探索を高速化する（成功する確率を上げるといったほうがよいかもしれない）鍵は上で述べた局所解に陥ったときの処理であることは間違いない．しかし，これは理論的解析はほとんど不可能で確実性に乏しく，「実験をやってみなければわからない」というのが実情である．それに比べて，明らかに高速化に寄与するのが，一回の近傍探索を高速化する方法である．例えば，現在の割当てを式に代入して非充足の項を数え，各変数をフリップした新しい割当てを素直にまた代入して非充足の項を数える，といった工夫のないアルゴリズムでは大きな損をすることになる．データ構造をちょっと工夫することによって，一回の近傍探索をほとんど定数時間にまで速めることができるのである（一回のフリップで値が変化するのは1個の変数だけなので，多くの項の値は変化しない．変化する項だけを上手に探索することによってこのことが実現できる）．

さらに並列化によって高速化できる．局所探索は，上で述べたように局所解の処理を工夫したとしても，やはり何回も最初からやり直す必要がある．この最初からやり直すという部分が自然に並列化可能である．それぞれの探索は独立に実行できるので，探索間の通信も必要なく理想的な並列化が容易に実現できるのである．表1に最近の著者らの論文 [Iwama 00] で発表したデータを引用する．これは，各種SATアルゴリズム（プログラム）のDIMACSベンチマーク [Johnson 96] に対する実行時間を表している．この表でOur GSATと書かれた列が我々の並列化したGSATプログラムの実行時間を表す．なお，並列化は富士通社のVPP800という機種（40プロセッサ）向けに行ったので，一部ベクトル化も併用しているし，またデータ構造の工夫も行っている．さらに*をつけたデータは上で述べたランダムウォークを併用して高速化した結果である．Selman's GSATはオリジナルのGSATプログラムである．さらに1から7のプログラムは，それぞれ，Duboisらのバックトラック，Hampsonらのヒルクライム，Jaumardらの Davis-Putnum-Loveland法，Pretolaniの Davis-Putnum-Loveland法，Resendeらの貪欲法，Spearsらの simulated annealingによる局所探索，Gelderらの resolutionに基づく方法である．これらのアルゴリズムに関し

ての詳細は [Johnson 96] を参照されたい．

データにおける数値の単位は秒である．N.A.はプログラムが（試みたが有意な時間内には）解けなかったことを示す．0は1秒未満を意味する．各研究者が実験に使用した計算機の性能は最大で10倍近い差がある（詳細は [Iwama 00] 参照）のでこの表だけでは正確な比較はできないが傾向はよく出ていると思われる．我々の並列化局所探索はやはり高速で，他のどのプログラムも解けなかった例題を2個解いている．表を見るとすぐに気付くこととして，例えば1のアルゴリズムは表の下のほうのssaの例題に対して強力であるが，中央上部のf例題に対しては弱い．この傾向は4のアルゴリズムも同様である．実は，1も4もバックトラックを基本にしたアルゴリズムになっている．2やGSATのアルゴリズムは明らかに逆の傾向を示している．これらは局所探索のアルゴリズムになっている．このように，局所探索法とバックトラック法では得意とする例題に違いが見られるようである．

5. おわりに

最初に述べたように，SATは非常に人気の高い研究テーマであって，AI分野と計算量理論分野の多くの研究者に支持されている．多くの「手頃な」未解決問題が存在するという意味でも魅力のある分野である．一つ例を上げると3SATのランダム例題のしきい値問題である．3SATの例題をランダムに生成するとその項数が $4.2n$ の付近で充足例題から非充足例題に突然変化する．この事実は比較的古くから知られていて，その数学的証明も試みられてきたが，いまだに完成されていない．例えば [Kirousis 96] を参照されたい．

◇ 参考文献 ◇

[Cha 95] B. Cha and K. Iwama: Performance test of local search algorithms using new types of random CNF formulas, *Proc. International Joint Conference on Artificial Intelligence (IJCAI-95)*, Montreal, pp. 304-310, 1995.

[Cha 97] B. Cha, K. Iwama, Y. Kambayashi and S. Miyazaki: Local search algorithms for partial MAXSAT, *Proc. AAAI'97*, pp. 263-268, 1997.

[Cook 71] S. Cook: The complexity of theorem-proving procedures, *Proc. 3rd Ann. ACM Symp. on Theory of Comp.*, pp. 151-158, 1971.

[Dantsin 00] E. Dantsin, A. Goerdt, E. A. Hirsch, R. Kannan, J. Kleinberg, C. Papadimitriou, P. Raghavan and U. Schöning: A deterministic $2-\frac{2}{k+1}$ algorithm for k-SAT based on local search, to appear in *Theoretical Computer Science*.

[Goemans 95] M. Goemans and D. Williamson: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, *Journal of the ACM*, 42, pp. 1115-1145, 1995.

[Haken 85] A. Haken: The intractability of resolution, *Theoretical Computer Science*, 39, pp. 297-308, 1985.

[Hastad 97] J. Hastad: Some optimal inapproximability results, *Proc. 29th Ann. ACM Symp. on Theory of Comp.*, pp. 1-10, 1997.

[Iwama 00] K. Iwama, D. Kawai, S. Miyazaki, Y. Okabe and J. Umemoto: Parallelizing Local Search for CNF Satisfiability Using Vectorization and PVM, *Proc. Workshop on Algorithm Engineering (WAE 2000)*, 2000.

[Iwama 01] K. Iwama and S. Tamaki: Exploiting Partial Knowledge of Satisfying Assignments, *Workshop on Theory and Applications of Satisfiability Testing*, Boston, Massachusetts, 2001 (to be presented).

[Johnson 74] D. Johnson: Approximation Algorithms for Combinatorial Problems, *J. Comput. System Sci.*, 9, pp. 256-278, 1974.

[Johnson 96] D. Johnson and M. Trick, Eds.: Cliques, Coloring, and Satisfiability, *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 26, American Mathematical Society, 1996.

[Kirousis 96] L. Kirousis, E. Kranakis and D. Krizanc: Approximating the unsatisfiability threshold of random formulas, *Proc. 4th Ann. European Symp. on Algorithms (ESA'96)*, pp. 27-38, 1996.

[Morris 93] P. Morris: The breakout method for escaping from local minima, *Proc. AAAI-93*, pp. 40-45, 1993.

[Paturi 97] R. Paturi, P. Pudlák and F. Zane: Satisfiability coding lemma, *Proc. 38th Ann. Symp. on Foundations of Computer Science*, pp. 566-574, 1997.

[Paturi 98] R. Paturi, P. Pudlák, M. E. Saks and F. Zane: An improved exponential-time algorithm for k-SAT, *Proc. 39th Ann. Symp. on Foundations of Computer Science*, pp. 628-637, 1998.

[Schuler 01] R. Schuler, U. Schöning and O. Watanabe: An improved randomized algorithm for 3-SAT. Preprint, 2001.

[Schöning 99] U. Schöning: A probabilistic algorithm for k-SAT and constraint satisfaction problems, *Proc. 40th Ann. Symp. on Foundations of Computer Science*, pp. 410-414, 1999.

[Selman 92] B. Selman, H. Levesque and D. Mitchell: A New Method for Solving Hard Satisfiability Problems, *Proc. 10th National Conference on Artificial Intelligence (AAAI-92)*, pp. 440-446, San Jose, 1992.

[Selman 93a] B. Selman and H. Kautz: An empirical study of greedy local search for satisfiability testing, *Proc. AAAI-93*, pp. 46-51, 1993.

[Selman 93b] B. Selman and H. Kautz: Local search strategies for satisfiability testing, *2nd DIMACS Challenge Workshop*, 1993.

2001年6月19日　受理

著者紹介

岩間　一雄（正会員）

1980年生まれ．1973年京都大学工学部電気工学科卒業．1980年同大学院博士課程修了．工学博士．1973年京都産業大学理学部計算機科学科講師．1982年同助教授．1983～84年までカリフォルニア大学バークレー客員準教授．1990年九州大学工学部情報工学科助教授，1992年同教授を経て，1997年京都大学大学院工学研究科情報工学専攻教授．現在同大学院情報学研究科教授．アルゴリズムと計算の複雑さの理論の研究に従事．情報処理学会，電子情報通信学会，ACM，SIAM各会員．