離散最適化に対する高速な厳密アルゴリズム

岡本　吉央

1. はじめに

離散的な最適化問題では，「n 個の要素から成る集合の部分集合で，ある性質を満たすものの中から要素数が最大のものを選ぶ」といった形をしたものによく遭遇する．部分集合をすべて列挙して，その中から望みのものを探すという手法があるけれども，これでは列挙しなくてはならない対象が多すぎて（今の場合は 2^n 個存在することになり），有効な方法になりにくい．しかし，いわゆるNP困難な最適化問題に対しては計算量が指数関数的になってしまうことは避けられないと考えられている．

その一方で，近年，アルゴリズムの会議などに足を運ぶとそのような問題に対して 1.4422^n とか 1.3302^n といった計算量を持つアルゴリズムを見受けるようになった．このような計算量もいまだ指数関数であるが，2^n に比べると「遥かに」小さい．例えば，仮想的に $2^{n/2}$ という計算量（これは $\sqrt{2}^n$ と同じである）を持つアルゴリズムを考えて，これを 2^n という計算量を持つアルゴリズムと比較してみよう．計算量の比を計算すると $2^n/2^{n/2}=\sqrt{2}^n$ となり，アルゴリズムの計算量が**指数関数的に改善**されたことになる．これは，n^3 という計算量を持つアルゴリズムを n という計算量を持つアルゴリズムに改善しても，$n^3/n=n^2$ というだけの改善にしかならないので，この力は絶大である．

では，アルゴリズム設計者としての興味は，1.4422^n というような計算量を持つアルゴリズムをどのようにして組み立てればよいのか，ということである．この分野は大きいけれども，まだまだ手法が確立しておらず，また大きな未解決問題がたくさんあるといった理論情報科学の目から見ると大変興味深い分野になっている．また，以降で解説する手法の中にはこれまで発見的アルゴリズムなどと呼ばれて計算量解析があまり行なわれてこなかったものもあり，実用上役に立っていたアルゴリズムが理論的にどれほどの速さで動くのかということを教えてくれることにもなる．

この分野は「厳密計算（exact computation）」と呼ばれることもあるが，名前はまだそれほど一般的になっていない．これと関連した分野に「パラメータ化計算（parameterized computation）」あるいは「パラメータ化計算量理論（parameterized complexity）」といったものがある．
こちらもNP困難な問題に対する厳密アルゴリズムを考察する分野であるが，厳密計算の分野とは少し趣きが違う．

本稿の限られた紙面では，主にグラフの彩色問題と最大独立集合問題に焦点を絞ることにする．それを通して，この分野の手法のいくつかを紹介していきたい．

厳密計算の分野に限って解説した書物は今のところ見当たらない．解説論文ではWoeginger[26]を推薦したい．最近Woegingerは別の解説論文[27]を書いている．本稿で集中するグラフの彩色問題に関してはByskovの博士論文[4]の前半が詳しい．パラメータ化計算についてはDowney & Fellowsの本[7]があり，優れた解説論文もいくつかある．中でも，Niedermeierのハビリテーション論文[19]はよくまとまっているので，これを推薦したい．

以後，$O^*(2^n)$ のような表記をしていく．これは，$O(2^n \mathrm{poly}(n))$ を意味する．すなわち，計算量の中の多項式因子を無視して，指数関数部分にだけ注目をする，ということである．本稿で扱うアルゴリズムの目標は，その計算量に現われる指数関数の底の部分を出来る限り小さくする（1に近づける）ことにある．

2. グラフの用語

（無向）グラフ $G=(V,E)$ はその頂点集合 V と辺集合 E によって定められる．辺集合 E は V の要素の2個組のいくつかから成る集合である．グラフに関する用語の詳細についてはグラフ理論の教科書を参照

おかもと　よしお
豊橋技術科学大学 情報工学系
〒441-8580 豊橋市天伯町雲雀ヶ丘 1-1

してもらいたい．グラフが与えられたときに，その各頂点に色を塗って，隣り合う頂点同士は違う色で塗られるようにしたい．そのような彩色を真の彩色（proper coloring）と呼ぶ．グラフ G が k 色による真の彩色を持つとき，G は k 彩色可能（k-colorable）であると呼ばれる．真の彩色を行なうことのできる最小の色数のことを G の染色数（chromatic number）と呼ぶ．

グラフ $G=(V,E)$ において，真の彩色において，同じ色を持つ頂点の集合は独立集合（independent set，安定集合とも呼ばれる）になっている．独立集合とは，頂点部分集合で，その中のどの2頂点間にも辺が存在しないものである．要素数が最大の独立集合は最大独立集合と呼ばれ，対して，極大独立集合とは，それに他のある頂点を付け加えると独立集合にならないようなもののことである．

3．最大独立集合：列挙法と極値グラフ理論

与えられたグラフの最大独立集合を求めるための単純なアルゴリズムとして，「そのグラフの極大独立集合をすべて列挙して，その中から要素数が最大のものを見つける」というものが考えられる．

グラフのすべての極大独立集合を一つあたり線形時間で出力するアルゴリズムが知られている[25]．しかし，グラフの極大独立集合の総数は頂点数に対して指数関数的になり得るので，これでは効率的なアルゴリズムが得られないように思える．

しかし，指数関数の底を小さくするという観点から見ると，Miller & Muller[17]および Moon & Moser[18]による次の結果は非常に重要である．

事実1：頂点数が n のグラフには極大独立集合が高々 $3^{n/3}$ 個しかない．

事実1によって，上記のアルゴリズムの計算量が $O^*(3^{n/3})=O^*(1.4423^n)$ になることが分かる．これはアルゴリズム自身の工夫ではなく，計算量解析の上での工夫と考えられるが，事実1のような極値的な結果がそのまま計算量解析に用いることができる点は面白い．

4．3彩色可能性：多項式時間アルゴリズムを持つ問題への帰着

グラフの3彩色可能性問題はNP完全であるが，2彩色可能性問題は線形時間で解くことができる．その観察と先の事実1から，3彩色可能性を判定するアルゴリズムを設計しよう（これはLawler[16]による）．

まず，Christofides[6]による次の観察

事実2：色数最小の彩色で，ある同一色頂点集合が極大独立集合になっているようなものが存在する

ことを確認してもらいたい．よって，与えられたグラフ G のすべての極大独立集合 S に対して $G-S$ が2彩色可能であるかどうかを確かめれば，G が3彩色可能であるかどうかが分かる．

グラフの極大独立集合 S の数は高々 $3^{n/3}$ であり（事実1），そのそれぞれに対して $G-S$ が2彩色可能であるかの確認は線形時間でできる．ゆえに，全体の計算量は $O^*(3^{n/3})=O^*(1.4423^n)$ になる．これは自明な $O^*(3^n)$ 時間のアルゴリズム，あるいはバックトラッキングを用いた $O^*(2^n)$ 時間のアルゴリズムに比べて格段によい．

5．染色数：部分集合縦断動的計画法

部分集合縦断動的計画法（dynamic programming across the subsets）は Held & Karp[13]と Bellman[2]が巡回セールスマン問題に対して1962年の論文で独立に発見したものであり，厳密計算の分野の中でも古典的な手法である．これはほとんど全列挙法であるが，列挙する対象を上手に制御することでよりよい計算量を導いている．グラフの染色数の計算を部分集合縦断動的計画法に基づいて行なうアルゴリズムを紹介する（これは再び Lawler[16]による）．

前節の事実2にある通り，色数最小の彩色においては，その中の同一色頂点集合が極大独立集合になっているようなものが存在している．頂点部分集合 $S\subseteq V$ に対して，$F(S)$ で S が誘導する G の部分グラフの染色数を表すことにする．S が空のときは $F(S)=0$ である．S が空でないときは，$F(S)=1+\min(F(S\setminus T)|T$ は $G[S]$ の極大独立集合）が成り立つ．ただし，$G[S]$ は S の誘導する G の部分グラフである．今，動的計画法を S の要素数の小さい方から順に行うと，

最終的に $F(V)$ を計算することができて，これが G の染色数である．

各 S に対して，$G[S]$ の極大独立集合を列挙する必要があるけれども，上で見たように，これは S の要素数が s のとき $O^*(3^{s/3})$ でできる．よって，$F(S)$ は $O^*(3^{s/3})$ で計算できる．動的計画法では可能なすべての S を見ていくので，計算量は総計 $\sum_{s=0}^{n}\binom{n}{s}O^*(3^{s/3})=O^*((1+3^{1/3})^n)\approx O^*(2.4423^n)$ である．

Eppstein[8]は，まず頂点数 n のグラフにおいて要素数が k 以下の極大安定集合の数が高々 $3^{4k-n}4^{n-3k}$ になることを証明した．これを利用することで，彼は $O^*(2.4150^n)$ で染色数を計算するアルゴリズムを与えている．Byskov[3]はグラフの4彩色可能性，5彩色可能性，6彩色可能性を判定する問題をそれぞれ $O^*(1.7504^n)$，$O^*(2.1592^n)$，$O^*(2.3289^n)$ という計算量で解くアルゴリズムを与えている．また，これらのアルゴリズムを用いることで，染色数を $O^*(2.4023^n)$ 時間で計算することを可能にした．これはこれら四つの問題に対する現在最速のアルゴリズムであり，アイディアはEppstein[8]の流れに従っている．

6. 最大独立集合：探索木の縮小

問題解決に対する一般的な方法に「場合分け」がある．それを組織立って行うと「探索木」になる．ここでは，分枝規則（場合分けの方法）の工夫によって，探索木が小さくなるようにすることを考え，そこから，高速なアルゴリズムが得られる．ここでは探索木の縮小によって，$O^*(1.3803^n)$ で最大独立集合を見つけてみたい．

グラフ G の頂点の次数がすべて2以下である場合を考えてみる．すると，G の各連結成分はパスかサイクルである．よって，この場合は簡単に（高々線形時間で）最大独立集合を発見できる．したがって，G に次数が3以上の頂点が存在すると仮定してよい．この頂点を v とする．グラフ G の最大独立集合 I を見つけたいとすると，I は v を含むか含まないかのいずれかである．I が v を含むとき，I は v の近傍を含むことができない．よって，I のその他の頂点は G から v と v の近傍を取り除いたグラフ G_1 の中にあることになり，このグラフの頂点数は高々 $n-4$ である（なぜなら，v の次数が3以上と仮定したからである）．I が v を含まないとき，I の頂点は G から v を取り除いたグラフ G_2 の中にあり，このグラフの頂点数は $n-1$ である．この両方の場合を考えて，ここで分枝（場合分け）を行なう．すなわち，$\alpha(G)$ でグラフ G の最大独立集合の大きさを表すとすると，$\alpha(G)=\max(1+\alpha(G_1),\alpha(G_2))$ が成り立つ．右辺の G_1, G_2 に対して，今の手続きを再帰的に繰り返すことで G の最大独立集合を見つけるのである．

では，$T(n)$ でこのアルゴリズムが頂点数 n のグラフの最大安定集合を求めるために必要な時間の最大値を表すことにする．グラフに次数3以上の頂点が存在しないとき，$T(n)$ は n の多項式である．そうでないときは，上記のアルゴリズムに従って二つの場合に分かれる．よって，$T(n)\leq T(n-4)+T(n-1)+\mathrm{poly}(n)$ となる．今，多項式因子は無視していて，計算量の上界に興味があるので，上の式を $T(n)=T(n-4)+T(n-1)$ と書き換えよう．ここで，$T(n)=\lambda^n$ と置くと，$\lambda^n=\lambda^{n-4}+\lambda^{n-1}$ が得られて，この方程式の実数解で正のものは $\lambda\approx1.3803$ である．ゆえに，アルゴリズムの計算量は $O^*(1.3803^n)$ である．

最大独立集合問題に対する厳密アルゴリズムを扱った最初の論文はTarjan & Trojanowski[24]であり，計算量は $O^*(1.2599^n)$ であった．アイディアは上で述べたものと同じであるが，次数の高い頂点の近傍に関してより詳細な場合分けを施している．さらに詳細な場合分けがいくつかの論文で議論されていて，最大独立集合問題に対する現在最速のアルゴリズムはRobson[22]のものである．その計算量は $O^*(1.1888^n)$ であると報告されていて，アルゴリズムに現われる場合分けの数は膨大であり，かなり複雑である．また，探索木に基づく手法では，効果的な分枝規則を見つけることが重要であるが，それは容易ではない．そこで，分枝規則，単純化規則の発見を自動化し，また，計算量の算出も自動化する試みが行なわれている[9，12，10，15]．

他にも，探索木自体は変えず，解析を変更することでよい計算量の上界を得るという研究がある．例えば，Chen，Kanj & Xia[5]による探索木の慣らし解析（amortized analysis）や，Fomin，Grandoni & Kratsch[11]による測度統治法（measure and conquer）がある．

7. 最大独立集合：記憶化による高速化

探索木による方法では，同じグラフが異なる再帰呼出しで何度も出現することがある．その場合，同じグラフに対して全く同じ計算を行うことになり，計算時間の無駄が生じる．それを避けようとすることが記憶

化（memorization）のアイディアである（基本的な考え方はJian[14]およびRobson[21]による）．

まず，確認しておきたいことは，上記の探索木に基づくアルゴリズムの中で出現するグラフ G_1 と G_2 は元のグラフ G の誘導部分グラフであるということである．したがって，アルゴリズムの中で登場するグラフはすべて G の誘導部分グラフである．

新しいアルゴリズムは次のようになる．あるパラメータ α を考える（これは後で定める定数である）．まず，与えられたグラフ G の頂点部分集合で要素数が αn 以下のものをすべて列挙して，それぞれが誘導する部分グラフの最大独立集合を求めてデータベースとして記憶する．その後，探索木に基づくアルゴリズムを G に対して実行するが，分枝によって得られるグラフ G_1 あるいは G_2 の頂点数が αn 以下になったとき，その場合の計算結果はデータベースから取って来ることにするのである．これによって，G の最大独立集合が計算できる．

では，計算量を見積もってみよう．要素数 n の集合における要素数 αn 以下の部分集合 S の総数は $O^*(\binom{n}{\alpha n})$ である．そのそれぞれに対して，$G[S]$ の最大独立集合を計算する．これは，動的計画法に基づけば，部分集合を要素数の小さい順に列挙することで，各 S に対して $O^*(1)$ 時間で行える．部分集合を要素数の小さい順に列挙することは1出力に対して定数時間でできるので，データベース構築にかかる時間は $O^*(\binom{n}{\alpha n})$ である．

探索木に基づく後半部は，前節と同様に $T(n) \leq T(n-4)+T(n-1)+\mathrm{poly}(n)$ という再帰式が得られるが，$k \leq \alpha n$ に対しては $T(k)=O^*(1)$ になる点が先とは違う．これを考慮すると $T(n)=O^*(1.3803^{(1-\alpha)n})$ が得られる．したがって，全体の計算量は $O^*(\binom{n}{\alpha n}+1.3803^{(1-\alpha)n})$ になる．パラメータ α が大きくなると，この式の第1項は大きくなり，第2項は小さくなる．よって，この二つの項が釣り合うような α を定めることで総計算量を小さくできるわけである．すなわち，$\binom{n}{\alpha n}=1.3803^{(1-\alpha)n}$ となるような α を見つけるのである．この方程式を厳密に解くことは難しいが，Stirlingの公式などを用いて近似的に解くと，$\alpha=0.086521$ がよいパラメータ値であることが分かり，このとき，全体の計算量は $O^*(1.3803^{(1-0.086521)n})=O^*(1.3424^n)$ となる．

Jian[14]は探索木自身も更に縮小して，記憶化も用いたアルゴリズム自体は上で見たものよりも速く，その計算量は $O^*(1.2218^n)$ である．同様に，Robson[21]は $O^*(1.2108^n)$ 時間でこの問題を解いている．

8. 彩色可能性：CSPへの一般化

彩色可能性に対して次節と次々節で見るアルゴリズムはより一般的な制約充足問題（constraint satisfaction problem, CSP）に対して動く．本節ではCSPを定義しよう．

CSPでは変数の集合 V と制約の集合 $\mathcal{C}$ で与えられる．$n=|V|$，$m=|\mathcal{C}|$ とする．各変数 $x_i \in V$ は集合 D_i からその値を取るものとし，$d=\max_i|D_i|$ とする．$\prod_i D_i$ の要素 $(a_1, \ldots, a_n)$ は割当と呼ばれて，a_i が変数 x_i の取る値を表す．各制約 $C \in \mathcal{C}$ は変数とその変数の取り得る値から成る2個組の集合である．すなわち，ある $x_i \in V$ と $a_i \in D_i$ を用いて $C=\{(x_1, a_1), (x_2, a_2), \ldots, (x_l, a_l)\}$ のような形に書けるものである．割当 $b \in \prod D_i$ が制約 C を充足するとは，ある $(x_i, a_i) \in C$ に対して $a_i \neq b_i$ となることである．割当 a がすべての制約を満たすとき，この割当を充足割当と呼ぶ．充足割当が存在するCSPのインスタンスは充足可能であると呼ばれる．CSPとは，V と $\mathcal{C}$ が与えられたとき，それが充足可能であるか判定する問題である．CSPにおける重要なパラメータは n，m，d，l であり（ただし，$l=\max_{C\in\mathcal{C}}|C|$），$d$ と l を固定した場合の問題は (d, l)-CSPと呼ぶ．

グラフの k 彩色可能性問題は $(k, 2)$-CSPとして定式化できる．この場合，n 個の変数はグラフの n 個の頂点に対応して，m 個の制約はグラフの m 個の辺に対応する．2個組 $\{u, v\}$ がグラフの辺であるとき，「$\{(x_u, a), (x_v, a)\}$」という制約を k 個の色 a について与えるのである．CSPの他の例として k 充足可能性問題（k-SAT）があり，これは $(2, k)$-CSPとして定式化できる．

9. CSP：ランダムウォーク

局所探索法は離散最適化で用いられる一般的な解法であり，その応用は留まるところを知らない．ランダムウォークによる局所探索を厳密アルゴリズムの視点から再考してみる（本節の内容は基本的にSchöning[23]によるものであり，それはPapadimitriou[20]のアイディアに基づいている）．

(d, l)-CSPにおいて，n 個の変数に対する割当の種類は d^n 通りある．今，この d^n 個の割当を頂点とする次のようなグラフを考える．すなわち，二つの割

当 a, a' が辺で結ばれるのは，a, a' の中のちょうど1成分が異なるとき，そして，そのときに限る，とするのである．このグラフをとりあえず G と書くことにする．では，G の上でランダムウォークを行うことで充足割当を見つけてみよう．すなわち，アルゴリズムは次のようになる．まず，初期割当 $a \in \prod D_i$ を一様ランダムに選ぶ．割当 a が充足割当ならば，その時点で終了である．そうでない場合，$\mathcal{C}$ の中には a が充足しない制約が存在する．それを C としよう．C には高々 l 個の変数が含まれているので，それを一つ一様ランダムに選び，その変数に対する割当を a のものから一様ランダムに変更する．このようにして変更した割当を新たな初期割当と置き直して，上記の手続きを $3n$ 回繰り返すのである．

では，与えられたインスタンスが充足可能である場合，このアルゴリズムが充足割当を見つける確率はどれほどだろうか？　非自明な計算によって，この確率は $\frac{1}{d^n}\left(1+\frac{d-1}{l(d-1)-1}\right)^n$ 以上になることが分かる．よって，その逆数程の計算量 $O^*\left(\left(d\frac{l(d-1)-1}{l(d-1)+d-2}\right)^n\right)$ で充足割当が高確率で見つかる．これは自明なアルゴリズムの計算量 $O^*(d^n)$ よりも遥かによい．

このアルゴリズムを k-SATに適用すると，$O^*((2(1-1/k))^n)$ という計算量が得られる．これは k-SATに対して知られている中で最速の（確率的）アルゴリズムであり，k を固定した場合の改善や，アルゴリズムの脱確率化など，その後の進展も多い．しかし，このアルゴリズムを3彩色可能性問題に適用すると，$O^*((9/5)^n)=O^*(1.8^n)$ という計算量が得られて，これは以前に見た計算量 $O^*(1.4423^n)$ よりも悪い．次節の目標は，このCSPによるアプローチで k 彩色可能性問題に対する計算量の改善することである．

10．CSP：制約の単純化

上のランダムウォークとは異なる原理に基づく確率的アルゴリズムを考える（これはBeigel & Eppstein [1]のアイディアである）．簡単のため，(3,2)-CSPに限定して議論を進める．

今，(3,2)-CSPのインスタンスが与えられて，ある変数 x_i の定義域 D_i の要素数が3に満たないとする．これが1の場合，x_i の値をそれに固定できる．では，これが2の場合はどうだろうか？　仮に $D_i=\{a,b\}$ としよう．このとき，$A=\{(x_j,a')\mid\{(x_i,a),(x_j,a')\}\in\mathcal{C}\}$，$B=\{(x_j,b')\mid\{(x_i,b),(x_j,b')\}\in\mathcal{C}\}$ として，新しい(3,2)-CSPのインスタンスを次のようにして作る．変数の集合は V から x_i を除いたものとする．制約の集合は $\mathcal{C}$ から x_i を含むものを除き，代わりに，$A\times B$ の要素をすべて加える．このとき，元のインスタンスが充足可能であることと新たなインスタンスが充足可能であることの同値性を示すことができる（証明は省略する）．新たなインスタンスでは変数の数が一つ小さくなっていることに注意する（制約の数は高々多項式個しか増加していないことにも注意する）．

この単純化規則を用いてアルゴリズムを設計する．与えられた(3,2)-CSPのインスタンスの制約を一つ選ぶ．これを $\{(x_i,a),(x_j,b)\}$ としよう．上の単純化規則から，$|D_i|=|D_j|=3$ と仮定してよいので，$D_i=\{a,a',a''\}$，$D_j=\{b,b',b''\}$ とする．ここで，D_i と D_j をそれぞれ D_i' と D_j' に置き換えようと思うが，それは次の四つの可能性を一様ランダムに選んで行なうものとする．(1) $D_i'=\{a',a''\}$，$D_j'=\{b,b'\}$，(2) $D_i'=\{a',a''\}$，$D_j'=\{b,b''\}$，(3) $D_i'=\{a,a'\}$，$D_j'=\{b',b''\}$，(4) $D_i'=\{a,a''\}$，$D_j'=\{b',b''\}$．これによって新しい(3,2)-CSPのインスタンスが得られる．この新しいインスタンスが充足割当を持つとき，元のインスタンスも充足割当を持つ．しかし，逆が常に正しいとは限らない．なぜなら，例えば x_i に a を割り当てて，x_j に b' を割り当てるものが充足割当である場合，上の四つの可能性の中の(1)か(2)を選んでしまうと，これが見つけられないからである．しかし，このパターンが見つけられなくなる確率はちょうど1/2であり，これはどのパターンに対しても正しい．したがって，元のインスタンスが充足可能であるとき，新しいインスタンスも充足可能である確率は1/2以上である．

今，新しいインスタンスにおいて $|D_i'|=|D_j'|=2$ なので，先の単純化規則を用いて変数 x_i と x_j を除去する．これを $n/2$ 回繰り返すことで，自明なインスタンスを得ることができる．元のインスタンスが充足可能であるとき，この自明なインスタンスも充足可能である確率は $(1/2)^{n/2}$ 以上であり，よって，この手続きを $O^*(2^{n/2})$ 回繰り返すことで元のインスタンスの充足割当を高確率で見つけることができる．すなわち，このアルゴリズムの計算量は $O^*(2^{n/2})=O^*(1.4143^n)$ になる．

Beigel & Eppstein [1]はより詳細な単純化規則と探索木に基づいて計算量が $O^*(1.3645^n)$ の決定性アルゴリズムを(3,2)-CSPに対して構築している．3彩色可

能性問題という特殊ケースに対しては，それに加えて，グラフ特有の構造を生かした観察を用いることで計算量を $O^*(1.3289^n)$ にまで改善している．これは3彩色可能性問題に対して現在最速のアルゴリズムである．一般の $(k,2)$-CSP に対しては，$O^*((0.4518k)^n)$ 時間の確率的アルゴリズムを設計している．

11. 最後に

厳密計算の分野における異なる手法を見てきた．ここで紹介したものはこの分野の手法のごく一部であり，また計算量理論の視点から計算量の下界を導く研究もなされていることは補足しておきたい．

始めの節で名前だけ紹介したパラメータ化計算の分野とともに，厳密計算の分野は進展し始めている．2004年の9月にパラメータ化計算と厳密計算に関する最初の国際会議 International Workshop on Parameterized and Exact Computation がノルウェーのベルゲンで開かれた[1]．しかし，この会議での日本人発表者は私だけであった．本稿で紹介した結果のいくつかは極めて古いものであり，また他のいくつかは極めて新しいものである．本稿によって，より多くの日本人研究者がこの古くて新しい分野に魅力を感じていただければ幸いである．

謝辞　本稿を書く機会を与えて下さった文教大学の根本俊男氏に感謝したい．

参考文献

[1] R. Beigel and D. Eppstein: 3-Coloring in time $O(1.3289^n)$. J. Algor. **54** (2005) 168-204.

[2] R. Bellman: Dynamic programming treatment of the traveling salesman problem. J. ACM **9** (1962) 61-63.

[3] J. M. Byskov: Enumerating maximal independent sets with applications to graph colouring. Oper. Res. Lett. **32** (2004) 547-556.

[4] J. M. Byskov: Exact algorithms for graph colouring and exact satisfiability. Ph. D. Dissertation, University of Aarhus, 2004.

[5] J. Chen, I. A. Kanj and G. Xia: Labeled search trees and amortized analysis: improved upper bounds for NP-hard problems. Algorithmica, to appear.

[6] N. Christofides: An algorithm for the chromatic number of a graph. Computer J. **14** (1971) 38-39.

[7] R. G. Downey and M. R. Fellows: Parameterized Complexity. Springer, Berlin, 1999.

[8] D. Eppstein: Small maximal independent sets and faster exact graph coloring. J. Graph Algor. **7** (2003) 131-140.

[9] D. Eppstein: Quasiconvex analysis of backtracking algorithms. Proc. 15th SODA (2004) 788-797.

[10] S. S. Fedin and A. S. Kulikov: Automated proofs of upper bounds on the running time of splitting algorithms. Proc. 1st IWPEC (2004) 248-259.

[11] F. V. Fomin, F. Grandoni and D. Kratsch: Measure and conquer: Domination-A case study. Proc. 32nd ICALP, (2005) 191-203.

[12] J. Gramm, J. Guo, F. Hüffner and R. Niedermeier: Automated generation of search tree algorithms for hard graph modification problems. Algorithmica **39** (2004) 321-347.

[13] M. Held and R. M. Karp: A dynamic programming approach to sequencing problems. Journal of SIAM **10** (1962) 196-210.

[14] T. Jian: An $O(2^{0.304n})$ algorithm for solving maximum independent set problem. IEEE Trans. Comput. **35** (1986) 847-851.

[15] A. S. Kulikov: Automated generation of simplification rules for SAT and MAXSAT. Proc. 8th SAT (2005) 430-436.

[16] E. L. Lawler: A note on the complexity of the chromatic number problem. Inf. Proc. Lett. **5** (1976) 66-67.

[17] R. E. Miller and D. E. Muller: A problem of maximum consistent subsets. Research Report RC-240, IBM Research Center, 1960.

[18] J. W. Moon and L. Moser: On cliques in graphs. Israel J. Math. **3** (1965) 23-28.

[19] R. Niedermeier: Invitation to Fixed-Parameter Algorithms. Habilitation Thesis, University of Tübingen, 2002.

[20] C. H. Papadimitriou: On selecting a satisfying truth assignment. Proc. 32nd FOCS (1991) 163-169.

[21] J. M. Robson: Algorithms for maximum independent sets. J. Algor. **7** (1986) 425-440.

[22] J. M. Robson: Finding a maximum independent set in time $O(2^{n/4})$. Manuscript, 2001.

[23] U. Schöning: A probabilistic algorithm for k-SAT and constraint satisfaction problems. Algorithmica **32** (2002) 615-623.

[1] この国際会議の第2回が2006年にチューリッヒ（スイス）で開かれる予定のようである．

[24] R. E. Tarjan and A. E. Trojanowski: Finding a maximum independent set. SIAM J. Comput. 6 (1977) 537-546.
[25] S. Tsukiyama, M. Ide, H. Ariyoshi and I. Shirakawa: A new algorithm for generating all maximal independent sets. SIAM J. Comput. 6 (1977) 505-517.
[26] G. J. Woeginger: Exact algorithms for NP-hard problems: a survey. Combinatorial Optimization-Eureka! You shrink! (2003) 185-207.
[27] G. J. Woeginger: Space and time complexity of exact algorithms: some open problems. Proc. 1st IWPEC (2004) 281-290.