Vol.2018-MPS-120 No.21
2018/9/26

最大クリーク抽出アルゴリズムMCTの高速化

松崎 空良[1]　富田 悦次[2]　長尾 篤樹[3,4]　伊藤 大雄[1,4]　若月 光夫[1,2]　西野 哲朗[1,2]

概要：筆者らが以前に提唱した最大クリーク抽出アルゴリズム MCT (FAW 2016, LNCS 9711, pp.215-226, 2016) の高速化を行った. このために, 最大クリーク抽出の厳密解アルゴリズムの前処理として既に有効性が確認されている最大クリーク近似解抽出アルゴリズムの改良を行い, それに加えて, 効果的な分枝限定を可能にする節点の順序と従来の節点の順序を適切に切り替える処理を導入した. 本提唱最大クリーク抽出アルゴリズムは, Chu-Min Li らの提案した最新アルゴリズム IncMC2 （INFORMS J. Computing, 30, pp.137-153, 2018）との比較実験により, 多くの問題について IncMC2 他のアルゴリズムより高速であることを示した.

An Improved MCT Algorithm for Finding a Maximum Clique

Sora MATSUZAKI[1]　Etsuji TOMITA[2]　Atsuki NAGAO[3,4]　Hiro ITO[1,4]
Mitsuo WAKATSUKI[1,2]　Tetsuro NISHINO[1,2]

1. はじめに

単純無向グラフの最大クリークを抽出する問題に対し, 実働上で効率的に解を得ることを目的としたアルゴリズムの開発が数多く行われている. 実世界の様々な問題が最大クリーク抽出問題, あるいはそれに関連した問題としてモデル化可能であることが分かっており, 最大クリーク抽出問題は工学的にも非常に重要である. 具体的な応用例として, 符号理論やバイオインフォマティクス, DNA 計算における DNA 系列設計, 画像処理, 量子論理回路の最適設計, データマイニングなどが挙げられる [14,13]. しかしながら, 最大クリーク抽出問題は NP 困難であるため, 問題となるグラフの規模に従い, その最適解を求めるのに要する計算時間が指数関数的に増加することが予想されている. したがって, 最大クリーク抽出アルゴリズムの一層の高速化が望まれている [14,13]. この問題に対し, これまでに筆者らは, 深さ優先探索を基礎とし, 逐次的な近似彩色による分枝限定法を用いた最大クリーク抽出アルゴリズム MCQ [7], MCR [8], MCS [9,10], MCT [11], k5MCT [12] を提案してきた. 本稿では, アルゴリズム MCT に対し, 主に最大クリーク抽出探索の前処理部分を改良し, より高速な最大クリーク抽出アルゴリズムを提唱する.

諸定義, 記法は文献 [11] などの標準的様式による.

[1] 電気通信大学 情報理工学研究科
[2] 電気通信大学 先進アルゴリズム研究ステーション
[3] お茶の水女子大学 基幹研究院
[4] JST CREST

2. 従来のアルゴリズム MCT

本稿では, グラフが入力されたとき, そのグラフの最大クリークを 1 つ見つけ, それを出力するアルゴリズムを対象とする. ここで, グラフは隣接行列で保持する. 以下では, 富田らがこれまでに発表してきた最大クリーク抽出アルゴリズムである基本アルゴリズム [6], MCT [11], そして k5MCT [12] について述べる.

2.1 基本アルゴリズム

深さ優先探索により, 入力グラフの最大クリークを 1 つ抽出する基本アルゴリズム [6] を説明する.

入力グラフを $G = (V, E)$ とおく. 現探索段階で保持しているクリークを Q, これまでに求めた極大クリークの中で要素数が最大のクリークを Q_{max} と表す. Q に含まれる任意の節点に隣接している節点部分集合を候補節点集合と呼び, R と表す. 始めに, $Q := \emptyset$, $Q_{max} := \emptyset$, $R := V$ と初期設定する. 基本アルゴリズムは, 次に示す探索展開手続き EX-

PAND を R に関して再帰的に実行することにより, 入力グラフ中の最大クリークを 1 つ抽出する. 一番初めに, $R = V$ に対して EXPAND を行う. 次に. $|Q| + |R| > |Q_{max}|$ ならば, ある節点 $p \in R$ に着目し, $Q := Q \cup \{p\}$, $R_p := R \cap \Gamma(p)$ とし, $R_p \neq \emptyset$ ならば, R_p を新たな候補節点集合とみなし, R_p に対して EXPNAD を行う. $R_p = \emptyset$ ならば, Q は極大クリークであるので, $|Q| > |Q_{max}|$ ならば $Q_{max} := Q$ として Q_{max} を更新する. そして, $Q := Q - \{p\}$, $R := R - \{p\}$ として, $|Q| + |R| > |Q_{max}|$ ならば, R に含まれる p とは異なる節点に着目する. $|Q| + |R| \leq |Q_{max}|$ かつ $Q \neq \emptyset$ ならば, Q から最後に Q に加えた節点 p を除き, 1 つ前の探索段階にバックトラックし, さらに $R := R - \{p\}$ とする. $|Q| + |R| \leq |Q_{max}|$ かつ $Q = \emptyset$ ならば, Q_{max} を出力し探索を終了する.

最大クリーク探索の様子は, Q の要素数と深さが対応した探索木として表現される. 探索木の分枝数とは, EXPAND の総実行回数のことである.

2.2 番号付け

現探索段階の候補節点集合 R に含まれるすべての節点に対して, 次のように R の先頭要素から逐次的に正整数の番号を割り当てる手続きのことを番号付けと呼ぶ. 節点 $p \in R$ に対して正整数の番号を割り当てることを考える. このとき, 隣接節点 $q \in \Gamma(p) \cap R$ の番号とは異なる正整数で最小の番号を p に割り当てるとする. 節点 p に割り当てられた番号を $No(p)$ と表す. このとき, 次の不等式が成り立つ.

$$\omega(G(R)) \leq \max_{p \in R}\{No(p)\} \leq |R| \quad (1)$$

式 (1) より, 次の不等式が成立するとき, R の節点に着目し, EXPAND を行う必要はない.

$$|Q| + \max_{p \in R}\{No(p)\} \leq |Q_{max}| \quad (2)$$

したがって, $No(p) > |Q_{max}| - |Q|$ を満たす節点 $p \in R$ にのみ着目すれば十分である.

以上のような番号付けの手続きを加えることにより, 基本アルゴリズムは高速化できることがわかっている [6].

2.3 MCT

Tomita らが提唱した最大クリーク抽出アルゴリズム MCT [11] は, 同じく Tomita らが提唱した最大クリーク抽出アルゴリズム MCS [9,10] を基礎としたアルゴリズムであり, 広範なグラフに対してその優位性を示している [11]. その所以の 1 つが Katayama らの近似最大クリーク抽出アルゴリズム k-opt Local Search [1] の利用である.

2.3.1 k-opt Local Search の利用

k-opt Local Search (=KLS) [1] は近似最大クリーク抽出アルゴリズムの 1 つである. MCT は, 探索の前処理として KLS を実行する. そのようにすることで, 最大クリーク探索の早い段階から式 (2) を満たすようになり, 文献 [11] においてその効果が確認されている.

2.4 k5MCT

最大クリーク抽出アルゴリズム k5MCT [12] は筆者らが提唱した MCT を基礎としたアルゴリズムである. k5MCT は KLS を（概念的には, 並列に）5 回実行し, その内の最良解を最終解とすることにより, 複数のグラフに対する近似解の精度を向上させ, それらのほとんどに対して MCT よりも高速に解を求めることに成功している [12].

3. アルゴリズムの高速化

3.1 MIS ordering

MIS ordering と呼ばれる節点の順序付けがある [3]. 文献 [3] において提案された最大クリーク抽出アルゴリズム IncMaxCLQ は, MIS ordering を行うことにより, 対象グラフによっては大きな高速化を達成している.

3.1.1 MIS ordering の手順

どの 2 節点も隣接していない節点集合を独立集合と呼び, 最も要素数の大きい独立集合を最大独立集合 (MIS) と呼ぶ. MIS ordering はグラフ G の最大独立集合に着目した節点の順序付けである. まず, G の最大独立集合 S_1 を求め, 続いて $G_1 = G - S_1$ の最大独立集合 S_2 を求める. 同様の処理を $G_k = G_{k-1} - S_k = \emptyset$ となるまで続ける. そして, 各 S_i $(1 \leq i \leq k)$ ごとに各要素を次数の非増大順に並べる. 最後に, 節点を $S_1, S_2, ..., S_k$ の順番で並べ, MIS ordering は完了である.

3.1.2 節点の順序の切り替え

MIS ordering を行い, その順序に従って番号付けを行うことにより, 多くの節点に比較的小さい番号を割り当てることができ, 番号付けによる分枝限定がより効果を発揮することが期待できる. ところが, 次数の小さい順に探索を行うと効率がよいことが実験的に示されてもいる [6,8]. IncMaxCLQ [3] においては, 節点の次数に関連した節点の並べ方である degeneracy ordering [3] と, MIS ordering をある基準に従って切り替えている. なお, IncMaxCLQ は枝密度が 0.7 より大きいグラフに限り, MIS ordering の手続きに入る. これは, 最大独立集合を求める操作そのものが NP 困難であり, その手続きが比較的簡単であると予想されるグラフに制限しているためである. 本稿においても, その方針に従う.

3.1.3 mMCT

IncMaxCLQ における degeneracy ordering と MIS ordering の切り替えの基準をそのままに, MCT において, その基準を満たした場合に, 拡張イニシャルソート [*1] [8] か

[*1] 一番初めは次数最小の節点を, その後はソート済みの節点を除い

ら MIS ordering へ切り替えることを試みた．ただし, 切り替えるのは探索木の根に限定した．このように変更した MCT において, MIS ordering が効果的なグラフに対して MIS ordering へ切り替えない場合があることが, 計算機実験により判明した．反対に, MIS ordering が効果的でないグラフ [*2] に対して MIS ordering へ切り替えてしまう場合があることも判明している．そのため, IncMaxCLQ における MIS ordering の切り替えの基準は不安定である．実際, IncMaxCLQ の改良アルゴリズムである IncMC2 [4] における MIS ordering の切り替えの基準は IncMaxCLQ におけるそれを緩和させたものであるが, MIS ordering が適用されている対象は変化していないことが文献 [4] から推測される [*3]．そこで, IncMaxCLQ, IncMC2 における切り替えの基準とは異なる新たな基準を独自に考案し, 再度 MCT に MIS ordering を組み込んだ．そのアルゴリズムを mMCT とした．

3.1.4 mMCT における節点の順序の切り替え

グラフ $G = (V, E)$ に対し, MIS ordering の手続きにより得た最大独立集合の列を $S_1, ..., S_k$ とする．ここで, 拡張イニシャルソートされた節点集合 V の末尾の節点 $p\ (= V[|V|])$ と節点 $q = S_k[|S_k|]$ に着目する．現段階で保持している最大のクリークを Q_{max}, 拡張イニシャルソートに従った番号付けにより割り当てられた番号を No, MIS ordering に従った番号付けにより割り当てられる番号を No' としたとき, 2 つの実数 $t_1 = |\{v \in V \cap \Gamma(p) | No(v) > |Q_{max}| - 1\}|$, $t_2 = |\{v \in V \cap \Gamma(q) | No'(v) > |Q_{max}| - 1\}|$ に対して, mMCT は, 次の式 (3) または (4) が真のとき MIS ordering を適用する．

$$\frac{t_1}{t_2 + 1} \times \frac{t_1 - t_2}{|V|} > 0.3 \tag{3}$$

$$\max_{v \in R}\{No'(v)\} = |Q_{max}| \tag{4}$$

節点 p, q は, 前者は MIS ordering を適用しない場合, 後者は適用した場合, その後の手続き EXPAND おいて, 一番初めに着目される節点である．式 (3) の左辺の値が大きい程, q から派生する部分問題 $V \cap \Gamma(q)$ において, 探索すべき節点は $V \cap \Gamma(p)$ のそれよりも少ないと予測される．そこで, 式 (3) の左辺の値が計算機実験を基に設定した閾値 0.3 を超える場合は, 実際に V に対して MIS ordering を適用する．また, 式 (4) が真の場合, その後の手続き EXPAND が即時終了するため, MIS ordering を適用する．

3.1.5 mMCT における MIS ordering の求め方

グラフ $G = (V, E)$ に対し, $\bar{G} = (V, \bar{E})$ $(\bar{E} = \{(u, v) \in V \times V | u \neq v \wedge (u, v) \notin E\})$ を G の補グラフと呼ぶ. G_i の最大独立集合 S_i $(1 \leq i \leq k)$ は, G_i の補グラフ $\bar{G}_i$ 中の最大クリークを抽出することにより求める．$\bar{G}_i$ のサイズは徐々に縮小するため, 最大クリーク抽出アルゴリズムとしては比較的単純なアルゴリズムである MCS を用いる．ただし, その際, 次のようにアルゴリズム MCS の変更を行う．MCS の最大クリーク探索の前処理である, 拡張イニシャルソート, 簡易的な番号付け, および (条件を満たした場合 [*4]) 近似最大クリークを求める手続き EXTENDED-INITIAL-SORT-NUMBER において求まった近似最大クリーク Q'_{max} は, $\bar{G}_i$ において比較的次数の大きな節点で構成されている [*5]．仮に $|Q'_{max}| = \omega(\bar{G}_i)$ の場合, $\bar{G}_{i+1} = \bar{G}_i - Q'_{max}$ が比較的疎なグラフになり, $\omega(\bar{G}_{i+1})$ が小さくなってしまい, 結果として V が細かい独立集合に分割されて, MIS ordering に従った番号付けにおける最大番号の増加に繋がることが懸念される．それを防ぐため, 最大クリーク探索の前に Q'_{max} から Q'_{max} に含まれる任意の 1 節点を除去し, 改めて Q'_{max} と同等以上の要素数の最大クリークを MCS の EXPAND により抽出する．MCS における EXPAND は, 探索木の根においては, 拡張イニシャルソートに従い, 次数最小の節点から順に着目し, 根以外では, 拡張イニシャルソートに従い比較的次数の大きな節点から順に番号付けにより割り当てられた番号が大きい節点から順に着目する．このため, 比較的次数の大きな節点で構成された最大クリークを抽出する可能性は低い．また, $\omega(\bar{G}_i) \leq \omega(\bar{G}_{i-1})$ $(i > 1)$ より, EXPAND において $|Q_{max}| = \omega(\bar{G}_{i-1})$ なるクリーク Q_{max} が見つかった場合, 即時探索を打ち切り, $S_i := Q_{max}$ とする．なお, 隣接行列の再構築 [9, 10] は行わない．

3.2 近似最大クリーク抽出アルゴリズム IKLS

近似最大クリーク抽出アルゴリズム KLS [1] よりも良い精度が期待される近似最大クリーク抽出アルゴリズムとして, Katayama らの Iterated k-opt Local Search (=IKLS) がある [2].

IKLS における Local-Search の反復回数は, 近似解の精度と IKLS の計算時間に大きく影響する．そのため, ある解が得られた場合, その解が適切であるかどうかを判断する基準を設定し, 適切である場合には, その後 IKLS を速やかに終了するような反復回数の制御を行う必要がある．本稿では, そのような IKLS における Local-Search の反復回数の制御を実現し, MCT と mMCT に対して, KLS に代えて

た節点で誘導される部分グラフにおいて次数最小の節点を順に末尾から並べる手続き (次数がすべて等しい場合は, 元の並びに従って残りの節点を先頭から並べて終了する).

[*2] 例として, MCT で 1 秒未満で求まるベンチマークグラフ p_hat300-3 に対し, 前処理として MIS ordering を適用すると, 1 分以上かかるようになってしまう．また, 文献 [4] の切り替えの基準を満たしている．

[*3] 文献 [4] において, 切り替え基準の緩和による変化は明記されていない．

[*4] グラフ $G = (V, E)$ とソート済みでない節点 V' に対し, V' に含まれる各節点の次数が $G(V')$ において等しいとき拡張イニシャルソートは終了するが, その次数が $|V'| - 1$ と等しいとき, V' はクリークであるため, $Q'_{max} := V'$ とする．

[*5] 拡張イニシャルソートの定義より, Q'_{max} は比較的高次数節点で構成されている．

```
1: procedure MIS_ORDERING(R, No)
2:     k := 0
3:     R_1 := R
4:     repeat
5:         k := k + 1
6:         S_k := MCS(Ḡ(R_k))
7:         S_k 中の節点を次数の非増大順となるようにソート
8:         R_{k+1} := R_k − S_k
9:     until R_{k+1} = ∅
10:    for i = 1 to k do
11:        for j = 1 to |S_i| do
12:            No'(S_i[j]) := i
13:        od
14:    od
15:    No_th := |Q_max| − |Q|
16:    p := R[|R|], q := S_k[|S_k|]
17:    t_1 := |{v ∈ R ∩ Γ(p)|No(v) > No_th − 1}|
18:    t_2 := |{v ∈ R ∩ Γ(q)|No'(v) > No_th − 1}|
19:    if ((t_1/(t_2+1)) × ((t_1−t_2)/|V|) > 0.3) or (max_{v∈R}{No'(v)} = No_th)
       then
20:        R := S_1 ∪ ... ∪ S_k
21:        No := No'
22:    fi
23: end procedure
```

図 1 MIS ordering

IKLS を組み込んだ最大クリーク抽出アルゴリズム iMCT, miMCT を開発した．この制御方法の詳細は [5] において発表予定である．

なお, 今回方式とは別に, 以前に筆者らが提唱したアルゴリズム k5MCT [12] と同様に IKLS を 5 回（並列）実行しその内の最良解を最終解とする方式も試しているが, 今回では近似解法として KLS ではなく IKLS を採用し, かつその反復回数を適切に制御する方式を採用している下においては, 今回の（並列ではない）方式の方が良好な結果を得られることを確認している．

3.3 MCT′

入力グラフの枝密度が $0.1+\epsilon$ 以下ならば, MCS に KLS を組み合わせたアルゴリズムである MCS_1 [11] を, 枝密度が $0.1+\epsilon$ より大きく $0.7+\epsilon$ 以下ならば MCT [*6] を, 枝密度が $0.7+\epsilon$ より大きいならば miMCT を適用するアルゴリズムを MCT′ とした．ここで, ϵ (=0.01) は, ランダムグラフの枝存在確率と実際の枝密度の差を許容するための定数である．

4. 計算機実験

テストグラフとして広く用いられている DIMACS ベンチマークグラフ [*7], BHOSLIB ベンチマークグラフ [*8] と, 自作したランダムグラフ rn.p (n は節点数, p は枝存在確率) を対象とした計算機実験を行い, 表 1 にまとめた．ここで, 表 1 の書き方は文献 [11] の Table 2 に従う．表 1 に記載されているアルゴリズム IncMC2 [4] は, Chu-Min Li らが 2018 年に発表した最新アルゴリズムであり, 文献 [4] において, そこで比較した幾つかの他のアルゴリズムに対し多くの問題についてより高速であることを実験的に示している．IncMC2 の実行コードは `https://www.mis.u-picardie.fr/~cli/programs-for-joc.zip` に在るものによる．ランダムグラフに関する実験結果は乱数のシード値が異なる 10 個のグラフに対する平均値を記載している．ただし, r200.9 と r200.95 に限り, 100 個のグラフに対する平均値を記載．なお, MCT は $dens \leq 0.1$ のグラフに対しては MCS に切り替えるアルゴリズムであるが, 本稿では $dens \leq 0.1+\epsilon$ のグラフに対して MCS に切り替えるものとする．計算機環境は次の通り．OS:Linux CentOS7, CPU:Intel CPU Core i7-4790 3.6GHz, コンパイラ:gcc -O3, 主メモリ:8GB, キャッシュメモリ:8MB.

```
1: procedure MCT'(G = (V, E))
2:     global Q := ∅, Q_max := ∅
3:     dens := 2|E| / (|V|(|V|−1))
4:     global Th_1 := 0.4, Th_2 := 0.1, ε := 0.01
5:     if dens ≤ 0.1 + ε then
6:         MCS_1(G)
7:     else
8:         Q'_max := ∅
9:         EXTENDED-INITIAL-SORT-NUMBER(V, Q'_max)
10:        隣接行列の再構築
11:        if dens ≤ 0.7 + ε then
12:            KLS(V, Q'_max)
13:        else
14:            IKLS(V, Q'_max)
15:        fi
16:        if |Q'_max| > |Q_max| then
17:            Q_max := Q'_max
18:        fi
19:        NUMBER-R+(V, No)
20:        if dens > 0.7+ε and max_{v∈V}{No(v)} > |Q_max| then
21:            MIS_ordering(V, No)
22:        fi
23:        stage := 1
24:        EXPAND(V, No, stage)
25:    fi
26: return Q_max
27: end procedure
```

図 2 MCT′

4.1 考察

brock400_3, brock400_4 に対する MCT の実行時間は, IncMC2 の実行時間より大きいが, IKLS による近似解精度向上により, MCT′ では IncMC2 よりも高速に解を求めている．MCT′ において MIS ordering が適用された, keller5, frb30-15-i ($i = 1, ..., 5$), frb35-17-1 に対して, MCT の実行時間は IncMC2 の実行時間よりもかなり大きいが, MCT′

[*6] 当該の枝密度においては, k5MCT [12] よりも MCT が概ね高速であることが分かっている．
[*7] `https://turing.cs.hbg.psu.edu/txn131/clique.html`
[*8] `http://sites.nlsde.buaa.edu.cn/~kexu/benchmarks/graph-benchmarks.htm`

```
1: procedure EXPAND(R, No, stage)
2:     for i = |R| downto 1 do
3:         p := R[i]
4:         if (stage = 1 and |Q| + max_{v∈R}{No(v)} > |Q_max|)
5: or (stage ≠ 1 and |Q| + No(p) > |Q_max|) then
6:             Q := Q ∪ {p}
7:             R_p := R ∩ Γ(p)
8:             No_p := No
9:             No_th := |Q_max| − |Q|
10:            if R_p ≠ ∅ then
11:                T := |{v∈R | No_p(v) > No_th}| / |R_p| × dens
12:                if stage = 1 and Th_1 ≤ T then
13:                    拡張イニシャルソート (R_p)
14:                    NUMBER-R+(R_p, No_p)
15:                    newstage := 1
16:                else if dens > 0.95 + ε or Th_2 ≤ T then
17:                    NUMBER-R(R_p, No_p)
18:                    newstage := 2
19:                else
20:                    NUMBER-RL(R_p, No_p)
21:                    newstage := 3
22:                fi
23:                EXPAND(R_p, No_p, newstage)
24:            else if |Q| > |Q_max| then
25:                Q_max := Q
26:            fi
27:            Q := Q − {p}
28:            R := R − {p}
29:        fi
30:    od
31: return
32: end procedure
```

図 3 EXPAND

はいずれもその差を縮めることに成功している．また, MCT が IncMC2 よりも高速なベンチマークグラフに対して, brock200_1 を除き, MCT′ もまた IncMC2 よりも高速である．r200.9, r200.95, r10000.2 を除くランダムグラフに対して, MCT′ は IncMC2 よりも高速である．（なお, r10000.2 における次の順番の 10 個のランダムグラフに対する平均実行時間は, MCT′: 1,139sec, IncMC2: 1,154sec であり, MCT′ は IncMC2 よりも高速. 即ち, 高速性は対象グラフ依存.） r300.8, r400.8 に対しては, 近似解精度向上により, MCT′ は MCT, IncMC2 のいずれに対しても高速である. MCT は $dens \leq 0.1 + \epsilon$ のグラフに対しては MCS に切り替えるアルゴリズムであるが, MCT′ は MCS_1 に切り替えることにより, $dens \leq 0.1 + \epsilon$ かつ MCS の実行時間が 10 秒以上のグラフに対して高速になっている．以上より, MCT′ は IncMC2 と比較して多くの対象グラフに関して高速, または MCT との間に存在した差を縮めており, 着実な高速化を達成している.

5. おわりに

本稿では, 最大クリーク抽出アルゴリズム MCT の主に前処理部分を改良したアルゴリズム miMCT と, アルゴリズム MCS_1, MCT, miMCT を切り替えるアルゴリズム MCT′ を提唱し, 着実な高速化と, Chu-Min Li らの最新アルゴリズム IncMC2 [4] との比較により, 多くの問題について他のアルゴリズムより高速であることを示した.

謝辞 近似最大クリーク抽出アルゴリズム KLS, IKLS についてご教授をいただいた岡山理科大学・片山謙吾 教授, 有益な議論をいただいた電通大・中西裕陽 協力研究員に感謝します．本研究は科研, JST CREST, 柏森情報科学振興財団などの支援を受けている.

参考文献

[1] Katayama, K., Hamamoto, A., Narihisa, H.: An effective local search for the maximum clique problem, Information Processing Letters, Vol. 95, No. 5, pp. 503-511, 2005.
[2] Katayama, K., Sadamatsu, M., Narihisa, H.: Iterated k-opt local search for the maximum clique problem, EvoCOP 2007, LNCS 4446, pp. 84-95, 2007.
[3] Li, C. M., Fang, Z., Xu, K.: Combining maxsat reasoning and incremental upper bound for the maximum clique problem, IEEE ICTAI 2013, pp. 939-946, 2013.
[4] Li, C. M., Fang, Z., Jiang, H., Xu, K.: Incremental upper bound for the maximum clique problem, INFORMS J. Computing, Vol. 30, No. 1, pp. 137-153, 2018.
[5] 長尾，松崎，富田，伊藤，若月，西野 : 近似最大クリーク抽出アルゴリズム IKLS の反復回数に対する適切な制御方法，電子情報通信学会コンピュテーション研，2018，10 月予定.
[6] 富田，藤井 : 最大クリーク抽出の効率化手法とその実験的評価，電子通信学会論文誌 D，Vol. 68，No. 3，pp. 221-228，1985.
[7] Tomita, E., Seki, T.: An efficient branch-and-bound algorithm for finding a maximum clique, DMTCS 2003, LNCS 2731, pp. 278-289, 2003.
[8] Tomita, E., Kameda, T.: An efficient branch-and-bound algorithm for finding a maximum clique with computational experiments, J. Global Optimization, Vol. 37, No. 1, pp. 95-111, 2007.
[9] Tomita, E., Sutani, Y., Higashi, T., Takahashi, S. Wakatsuki, M.: A simple and faster branch-and-bound algorithm for finding a maximum clique, WALCOM 2010, LNCS 5942, pp. 191-203, 2010.
[10] Tomita, E., Sutani, Y., Higashi, T. Wakatsuki, M.: A simple and faster branch-and-bound algorithm for finding a maximum clique with computational experiments, IEICE Trans. Information and Systems, Vol. E96.D, No. 6, pp. 1286-1298, 2013.
[11] Tomita, E., Yoshida, K., Hatta, T., Nagao, A., Ito, H., Wakatsuki, M.: A much faster branch-and-bound algorithm for finding a maximum clique, FAW 2016, LNCS 9711, pp. 215-226, 2016.
[12] Tomita, E., Matsuzaki, S., Nagao, A., Ito, H., Wakatsuki, M.: A much faster algorithm for finding a maximum clique with computational experiments, J. Information Processing, Vol. 25, pp. 667-677, 2017.
[13] Tomita, E.: Efficient algorithms for finding maximum and maximal cliques and their applications, WALCOM 2017, LNCS 10167, pp.3-15, 2017.
[14] Wu, Q., Hao, J. K.: A review on algorithms for maximum clique problems, European J. Operational Research, Vol. 242, No. 3, pp. 693-709, 2015.

表 1 実験結果

Graph				Sol		Time [sec]		Time [sec]				
Name	n	$dens$	ω	KLS [1]	IKLS [2]	KLS	IKLS	MCS [9]	MCT [11]	miMCT	MCT′	IncMC2 [4]
brock200_1	200	0.745	**21**	**21**	**21**	0.02	0.36	0.35	**0.22**	0.55	0.55	0.50
brock200_2	200	0.496	**12**	10	**12**	0.00	0.05	**0.00**	**0.00**	0.05	**0.00**	0.02
brock200_3	200	0.605	**15**	14	**15**	0.01	0.17	**0.02**	**0.02**	0.18	**0.02**	0.05
brock200_4	200	0.658	17	**16**	**16**	0.01	0.24	0.06	**0.05**	0.25	**0.05**	0.12
brock400_1	400	0.748	27	**25**	**25**	0.09	1.83	282	116	118	118	**110**
brock400_2	400	0.749	29	24	**25**	0.09	1.80	121	51.6	**48.0**	**48.0**	80.9
brock400_3	400	0.748	**31**	25	**31**	0.09	1.79	191	58.4	**19.2**	**19.2**	60.9
brock400_4	400	0.749	**33**	25	**33**	0.09	1.77	101	46.0	**11.2**	**11.2**	16.7
brock800_1	800	0.649	23	20	**21**	0.23	4.41	4,122	1,965	1,957	1,965	**1,635**
brock800_2	800	0.651	24	20	**21**	0.23	4.49	3,683	1,721	**1,640**	1,721	1,659
brock800_3	800	0.649	25	21	**22**	0.23	4.50	2,540	1,106	1,008	1,106	**674**
brock800_4	800	0.650	26	**20**	**20**	0.23	4.43	1,768	**819**	826	**819**	1,147
C250.9	250	0.899	**44**	**44**	**44**	0.09	0.53	1,151	405	405	405	**151**
C2000.5	2,000	0.500	**16**	15	**16**	0.57	5.46	33,899	**21,012**	21,086	**21,012**	34,203
gen200_p0.9_44	200	0.900	**44**	**44**	**44**	0.06	0.35	0.17	0.07	0.36	0.36	**0.02**
gen200_p0.9_55	200	0.900	**55**	**55**	**55**	0.07	0.41	0.44	0.08	0.42	0.42	**0.03**
gen400_p0.9_55	400	0.900	**55**	53	**55**	0.27	1.84	22,536	135	1.85	1.85	**0.91**
gen400_p0.9_65	400	0.900	**65**	**65**	**65**	0.28	0.16	57,385	0.50	**0.22**	**0.22**	0.24
gen400_p0.9_75	400	0.900	**75**	**75**	**75**	0.30	2.60	108,298	0.31	2.60	2.60	**0.15**
MANN_a45	1,035	0.996	345	**344**	**344**	21.8	16.4	53.23	76.09	76.06	76.06	**6.88**
p_hat500-1	500	0.253	**9**	**9**	**9**	0.00	0.06	**0.01**	**0.01**	0.08	**0.01**	0.04
p_hat500-2	500	0.505	**36**	**36**	**36**	0.06	1.11	0.28	**0.11**	1.11	**0.11**	0.19
p_hat500-3	500	0.752	**50**	**50**	**50**	0.25	3.76	55.9	17.6	21.4	21.4	**11.3**
p_hat700-1	700	0.249	**11**	9	**11**	0.00	0.14	**0.04**	**0.04**	0.16	**0.04**	0.09
p_hat700-2	700	0.498	**44**	**44**	**44**	0.12	2.32	2.11	**0.68**	2.98	**0.68**	0.77
p_hat700-3	700	0.748	**62**	**62**	**62**	0.49	8.30	882	215	223	223	**122**
p_hat1000-1	1,000	0.245	**10**	**10**	**10**	0.01	0.32	0.22	**0.20**	0.51	**0.20**	0.33
p_hat1000-2	1,000	0.490	**46**	**46**	**46**	0.25	4.91	83.4	28.7	33.4	28.7	**20.5**
p_hat1000-3	1,000	0.744	**68**	**68**	**68**	1.07	17.18	305,146	39,149	39,207	39,207	**18,212**
p_hat1500-1	1,500	0.253	12	**11**	**11**	0.02	0.92	1.78	**1.41**	2.30	**1.41**	2.95
p_hat1500-2	1,500	0.506	**65**	**65**	**65**	0.78	14.76	6,299	1,552	1,571	1,552	**685**
san1000	1,000	0.502	**15**	10	**15**	0.05	1.91	1.06	**0.46**	1.92	**0.46**	1.13
san400_0.7_1	400	0.700	**40**	**40**	21	0.06	0.01	0.25	**0.06**	0.17	**0.06**	0.22
sanr200_0.7	200	0.697	**18**	**18**	**18**	0.01	0.32	0.14	**0.10**	0.40	**0.10**	0.13
sanr200_0.9	200	0.898	**42**	**42**	**42**	0.06	0.28	15.07	4.73	4.94	4.94	**1.47**
sanr400_0.5	400	0.501	**13**	**13**	**13**	0.01	0.39	0.34	**0.25**	0.58	**0.25**	0.40
sanr400_0.7	400	0.700	**21**	**21**	**21**	0.07	1.36	76.0	**40.7**	42.1	**40.7**	51.9
DSJC500.5	500	0.502	**13**	**13**	**13**	0.03	0.65	1.50	**1.20**	1.76	**1.20**	1.77
DSJC1000.5	1,000	0.500	**15**	14	**15**	0.11	3.05	138	**96.2**	96.6	**96.2**	117
keller4	171	0.649	**11**	**11**	**11**	0.00	0.13	**0.01**	**0.01**	0.13	**0.01**	0.03
keller5	776	0.752	**27**	**27**	**27**	0.34	6.22	82,422	9,770	280	280	**143**
hamming10-2	1,024	0.990	**512**	**512**	**512**	8.66	24.54	**0.04**	9.61	26.11	26.11	26.32
frb30-15-1	450	0.824	**30**	28	**30**	0.16	2.92	740	149	2.95	2.95	**0.09**
frb30-15-2	450	0.823	**30**	**30**	**30**	0.17	2.94	1,048	122	2.94	2.94	**0.10**
frb30-15-3	450	0.824	30	28	**29**	0.16	2.95	670	123	3.06	3.06	**0.15**
frb30-15-4	450	0.823	**30**	29	**30**	0.16	2.97	2,248	527	3.03	3.03	**0.15**
frb30-15-5	450	0.824	30	**29**	**29**	0.16	2.99	972	148	3.03	3.03	**0.11**
frb35-17-1	595	0.842	**35**	32	**35**	0.33	6.39	62,022	5,065	6.47	6.47	**0.65**
r200.6	200	0.600	13-14	13.5	**13.8**	0.006	0.166	**0.029**	**0.029**	0.179	**0.029**	0.049
r200.7	200	0.700	17-19	**18.2**	**18.2**	0.010	0.322	0.173	**0.116**	0.411	**0.116**	0.226
r200.8	200	0.800	24-27	25.2	**25.3**	0.030	0.387	1.64	**0.82**	1.13	1.13	1.14
r200.9	200	0.900	39-44	41.3	**41.3**	0.064	0.306	26.14	10.29	10.68	10.68	**3.92**
r200.95	200	0.950	58-66	62.36	**62.39**	0.108	0.299	20.35	11.00	10.31	10.31	**0.59**
r300.6	300	0.600	15-16	15.0	**15.1**	0.019	0.388	0.473	**0.352**	0.670	**0.352**	0.415
r300.7	300	0.700	20-20	19.7	**20.0**	0.030	0.732	5.28	**2.90**	3.41	**2.90**	4.91
r300.8	300	0.800	28-29	28.3	**28.4**	0.063	1.136	160	60.3	**59.3**	**59.3**	93.9
r400.5	400	0.500	12-13	12.6	**12.9**	0.010	0.351	0.32	**0.26**	0.56	**0.26**	0.41
r400.6	400	0.600	16-17	16.0	**16.1**	0.031	0.713	3.59	**2.31**	2.90	**2.31**	2.80
r400.7	400	0.700	21-22	21.3	**21.4**	0.067	1.336	75.1	**36.0**	36.1	**36.0**	63.4
r400.8	400	0.800	30-31	30.2	**30.5**	0.120	2.320	6,467	2,170	**1,982**	**1,982**	4,509
r500.5	500	0.500	13-14	12.7	**13.1**	0.021	0.446	1.31	**1.08**	1.45	**1.08**	1.48
r500.6	500	0.600	17-18	16.4	**16.8**	0.059	1.160	18.0	**11.3**	12.0	**11.3**	12.4
r500.7	500	0.700	22-23	21.8	**22.1**	0.109	2.158	733	338	**328**	338	707
r600.4	600	0.400	11-11	**10.3**	10.2	0.010	0.183	0.359	**0.309**	0.477	**0.309**	0.448
r600.5	600	0.500	14-14	13.2	**13.3**	0.040	0.763	4.21	**3.15**	3.82	**3.15**	3.58
r600.6	600	0.600	17-18	16.8	**17.3**	0.088	1.712	85.1	51.8	**49.8**	51.8	61.2
r1000.3	1,000	0.300	9-10	8.7	**9.1**	0.005	0.240	0.461	**0.429**	0.642	**0.429**	0.701
r1000.4	1,000	0.400	12-12	11.0	**11.3**	0.030	0.942	5.94	**5.12**	5.93	**5.12**	5.60
r1000.5	1,000	0.500	15-16	14.0	**14.6**	0.118	2.586	134	**93.4**	94.5	**93.4**	118
r1000.6	1,000	0.600	19-20	18.2	**18.5**	0.264	5.172	6,591	3,483	**3,480**	3,483	5,228
r2000.2	2,000	0.200	8-8	7.1	**7.5**	0.001	0.652	0.673	**0.671**	1.302	**0.671**	1.005
r2000.3	2,000	0.300	10-11	9.4	**10.0**	0.039	2.471	15.0	**13.4**	15.7	**13.4**	17.1
r2000.4	2,000	0.400	13-14	12.0	**12.3**	0.179	4.506	452	**361**	365	**361**	473
r3000.1	3,000	0.100	6-7	5.0	**5.5**	0.000	0.156	0.210	0.210	0.210	**0.207**	0.912
r3000.2	3,000	0.200	9-9	7.6	**7.7**	0.009	0.774	3.62	**3.44**	4.19	**3.44**	5.25
r3000.3	3,000	0.300	11-11	**10.0**	9.9	0.096	1.801	121	**108**	109	**108**	118
r3000.4	3,000	0.400	14-14	**12.6**	12.5	0.459	5.971	6,411	**5,175**	5,187	**5,175**	5,590
r4000.1	4,000	0.100	7-7	5.3	**5.5**	0.006	0.326	0.529	0.529	0.529	**0.525**	1.715
r4000.2	4,000	0.200	9-9	7.5	**7.8**	0.025	0.712	14.8	**13.4**	14.1	**13.4**	20.6
r4000.3	4,000	0.300	11-12	**10.1**	9.7	0.194	1.576	633	**548**	548	**548**	748
r5000.1	5,000	0.100	7-7	5.0	**6.0**	0.010	0.800	1.13	1.13	1.13	**1.12**	2.91
r5000.2	5,000	0.200	9-10	7.9	**8.0**	0.043	3.119	44.1	**39.0**	42.0	**39.0**	61.0
r5000.3	5,000	0.300	12-12	**10.2**	**10.2**	0.315	7.297	2,269	**1,879**	1,881	**1,879**	2,328
r10000.05	10,000	0.050	6-6	4.0	**4.9**	0.011	0.711	**1.88**	**1.88**	**1.88**	1.89	9.83
r10000.1	10,000	0.100	7-7	6.0	**6.1**	0.028	4.315	13.7	13.7	13.7	**13.1**	20.7
r10000.15	10,000	0.150	9-9	7.0	**7.4**	0.064	7.673	129	**118**	130	**118**	148
r10000.2	10,000	0.200	10-10	8.2	**8.8**	0.198	18.895	1,331	1,141	1,161	1,141	**1,126**
r10000.25	10,000	0.250	11-12	9.7	**9.9**	0.611	20.031	15,304	**12,966**	12,996	**12,966**	18,972
r15000.05	15,000	0.050	5-5	3.2	**5.0**	0.022	1.988	**6.96**	**6.96**	**6.96**	7.24	25.07
r15000.1	15,000	0.100	8-8	6.0	**6.5**	0.051	6.133	65.3	65.3	65.3	**62.4**	73.3
r15000.15	15,000	0.150	9-9	7.1	**7.7**	0.174	9.725	1,015	**903**	936	**903**	1,135
r15000.2	15,000	0.200	10-11	**9.0**	8.7	0.488	14.175	10,565	**9,497**	9,506	**9,497**	9,777
r20000.05	20,000	0.050	6-7	3.7	**5.1**	0.037	3.481	**18.3**	**18.3**	**18.3**	18.7	49.2
r20000.1	20,000	0.100	8-8	6.2	**6.3**	0.094	10.022	251	251	251	**239**	308
r20000.15	20,000	0.150	9-10	**7.7**	7.4	0.264	7.656	4,169	**3,687**	3,828	**3,687**	4,153