Vol.2016-AL-160 No.11
2016/11/24

分枝限定法による最大辺重みクリーク抽出法

清水 悟司[1,a)]　山口 一章[1,b)]　増田 澄男[1,c)]

概要：辺に重みが付与された無向グラフが与えられたとき，辺の重みの和が最大のクリークを求める問題を最大辺重みクリーク問題という．最大辺重みクリーク問題は NP 困難である．従来，最大辺重みクリーク問題に対し，0-1 整数計画問題や二次計画問題へと定式化を行い，数理計画ソルバを用いて厳密解を求めるアプローチが研究されてきた．本稿では，分枝限定法に基づく最大辺重みクリーク問題の厳密解法を提案する．提案法は従来の方法に比べ，非常に短い計算時間で厳密解が得られることを計算機実験により確認した．

キーワード：最大辺重みクリーク問題, 分枝限定法,NP 困難

A branch-and-bound algorithm for the maximum edge-weight clique problem

Satoshi Shimizu[1,a)]　Kazuaki Yamaguchi[1,b)]　Sumio Masuda[1,c)]

Abstract: Given an edge-weighted undirected graph, to find the clique of maximum edge-weight sum is called the maximum edge weight clique problem(MEWCP). The MEWCP is NP-hard. Previous studies formulate the MEWCP as the 0-1 integer programming or the quadratic programming. And they use mathematical programming solvers to obtain exact solutions. In this paper, we propose an exact algorithm based on branch-and-bound for MEWCP. By some numerical experiments, we confirm our proposal algorithm is greatly faster than previous methods.

Keywords: maximum edge weight clique problem, branch-and-bound, NP-hard

1. まえがき

無向グラフ $G = (V, E)$ において，$V' \subseteq V$ による G の頂点誘導部分グラフを $G(V')$ と記す．$G(V')$ に存在する辺の集合を $E(V')$ と記す．頂点 $v \in V$ に隣接する頂点の集合を $N(v)$ と記す．各頂点に与えられた非負の重みを $w_v(\cdot)$，各辺に与えられた非負の重みを $w_e(\cdot,\cdot)$ と記す．$W_v(V') = \sum_{v \in V'} w_v(v)$ と定め，$W_e(V') = \sum_{(v,u) \in E(V')} w_e(v,u)$ と定める．

V の部分集合 C について，$G(C)$ が完全グラフであるとき，すなわち全頂点間に辺が存在するとき，C をクリークと呼ぶ．要素数最大のクリークを求める問題は，最大クリーク問題 (MCP) と呼ばれる．$W_v(C)$ が最大となるクリークを求める問題を最大重みクリーク問題 (MWCP) と呼ぶ．$W_e(C)$ が最大となるクリークを求める問題を最大辺重みクリーク問題 (MEWCP) と呼ぶ．MCP は NP 困難であることが知られている [1]．MWCP および MEWCP は MCP を一般化した問題であり，同様に NP 困難である．

また，MEWCP に類似する問題として Maximum diversity problem(MDP) がある．MDP は，入力として完全グラフ $G = (V, E)$，辺重み $w_e(\cdot,\cdot)$ およびパラメータ b が与えられたとき，$|C| \leq b$ を満たし，$W_e(C)$ が最大となる $C \subseteq V$ を出力する問題である．MDP は b-clique 問題とも呼ばれる．

MCP や MWCP に対しては，分枝限定法を元にした厳密

[1] 神戸大学大学院工学研究科
Kobe University
[a)] ss81054@gmail.com
[b)] ky@kobe-u.ac.jp
[c)] masuda@kobe-u.ac.jp

解法が多数提案されている [2], [3], [4], [5], [6]．また，MDP に対しては，分枝カット法によるアルゴリズム [7] や，分枝限定法によるアルゴリズム [8] が提案されている．

一方，MEWCP に対する厳密解法は，MEWCP を数理計画問題へと定式化し，数理計画ソルバによって厳密解を得る方法 [9] しか見当たらなかった．本稿では，MEWCP に対する，分枝限定法による厳密解法を提案する．計算機実験により，提案法は非常に短い時間で厳密解を得られることを確認した．

本稿の構成は以下の通りである．2 で各問題の数理計画問題での定式化について述べる．3 で MWCP の分枝限定法について述べる．4 で提案法 EWCLIQUE について述べる．5 で計算機実験の結果を示す．最後に 6 で本稿の結果をまとめる．

2. 数理計画問題での定式化

本節では，MWCP，MEWCP および MDP を数理計画問題で定式化した際の違いについて述べる．MWCP は整数計画問題 (IP) での定式化が可能である．MDP および MEWCP は，整数計画問題 (IP) および二次計画問題 (QP) での定式化が可能である．IP と QP では定式化した際の変数の数が異なる．

2.1 MWCP の定式化

MWCP は以下のように IP で定式化できる．

$$\text{maximize}: \sum_{v_i \in V} w_v(x_i) \tag{1}$$
$$\text{s.t.}: x_i + x_j \leq 1, \forall (v_i, v_j) \notin E \tag{2}$$
$$x_i \in \{0,1\}, \forall v_i \in V \tag{3}$$

頂点 v_i がクリークに含まれるとき，バイナリ変数 x_i は 1 となる．制約式 (2) は，非隣接な 2 頂点は同時にクリークに含まれることはできないという制約を意味する．$\forall v \in V, w_v(v) = 1$ とすると，MCP と等価になる．

2.2 MEWCP の定式化

MEWCP は IP と QP で定式化できる．ここで紹介する定式化の他に，グラフ G の最大クリークのサイズの上界が与えられた場合に，それを用いて定式化する方法が提案されているが [9]，本稿では扱わない．

2.2.1 二次計画問題 (QP) での定式化

MEWCP は，以下のように QP で定式化できる．

$$\text{maximize}: \sum_{(v_i,v_j) \in E} w_e(v_i, v_j) x_i x_j \tag{4}$$
$$\text{s.t.}: x_i + x_j \leq 1, \forall (v_i, v_j) \notin E \tag{5}$$
$$x_i \in \{0,1\}, \forall v_i \in V \tag{6}$$

MEWCP の QP での定式化と，MWCP の IP での定式化は，目的関数のみ異なり，制約式は同じである．

2.2.2 整数計画問題 (IP) での定式化

MEWCP は，以下のように IP で定式化できる．QP での定式化と異なり，目的関数が線形であるが，変数の数が QP に比べて $|E|$ 個多くなる．

$$\text{maximize}: \sum_{(v_i,v_j) \in E} w_e(v_i, v_j) y_{ij} \tag{7}$$
$$\text{s.t.}: y_{ij} \leq x_i, y_{ij} \leq x_j, \forall (v_i, v_j) \in E \tag{8}$$
$$x_i + x_j \leq y_{ij} + 1, \forall (v_i, v_j) \in E \tag{9}$$
$$x_i + x_j \leq 1, \forall (v_i, v_j) \notin E \tag{10}$$
$$x_i \in \{0,1\}, \forall v_i \in V \tag{11}$$
$$y_{ij} \in \{0,1\}, \forall (v_i, v_j) \in E \tag{12}$$

頂点 v_i がクリークに含まれるとき，バイナリ変数 x_i は 1 となる．制約式 (8) および (9) により，頂点 v_i および v_j の両方がクリークに含まれるとき，バイナリ変数 y_i は 1 となる．制約式 (10) は MWCP の制約式 (2) と同じく，非隣接な 2 頂点は同時にクリークに含まれることはできないという制約を意味する．

2.3 MDP の定式化

MDP は IP と QP で定式化できる．

2.3.1 二次計画問題 (QP) での定式化

MDP は，以下のように QP で定式化できる．

$$\text{maximize}: \sum_{(v_i,v_j) \in E} w_e(v_i, v_j) x_i x_j \tag{13}$$
$$\text{s.t.}: \sum_{v_i \in V} x_i \leq b \tag{14}$$
$$x_i \in \{0,1\}, \forall v_i \in V \tag{15}$$

MEWCP の QP での定式化と，MDP の QP での定式化は，制約条件のみ異なり，目的関数は同じである．制約式 (14) は，出力される頂点集合のサイズは b 以下であるという制約を意味している．

2.3.2 整数計画問題 (IP) での定式化

MDP は，以下のように IP で定式化できる．QP での定式化と異なり，目的関数が線形であるが，変数の数が QP に比べて $|E|$ 個多くなる．

$$\text{maximize}: \sum_{(v_i,v_j) \in E} w_e(v_i, v_j) y_{ij} \tag{16}$$
$$\text{s.t.}: y_{ij} \leq x_i, y_{ij} \leq x_j, \forall (v_i, v_j) \in E \tag{17}$$
$$x_i + x_j \leq y_{ij} + 1, \forall (v_i, v_j) \in E \tag{18}$$
$$\sum_{v_i \in V} x_i \leq b \tag{19}$$
$$x_i \in \{0,1\}, \forall v_i \in V \tag{20}$$
$$y_{ij} \in \{0,1\}, \forall (v_i, v_j) \in E \tag{21}$$

目的関数は，MEWCP の IP での定式化と同じである．制約条件は MEWCP に比べ，制約式 (10) と (19) のみ異な

る．制約式 (19) は MDP の QP での定式化の制約式 (14) と同じく，出力される頂点集合のサイズは b 以下であるという制約を意味している．

3. MWCP に対する分枝限定法

MWCP に対する従来法を紹介する．提案法はこれらのアルゴリズムを改変し，MEWCP を解くために利用する．

MWCP に対する分枝限定法で扱う部分問題を $P_v(C,S)$ と記す．C はクリークであり，S はクリークに加える頂点の候補集合を意味する．S は $\forall v \in S, C \subseteq N(v)$ を満たす．入力グラフ $G=(V,E)$ に対応する部分問題は $P_v(\emptyset,V)$ である．

分枝限定法は各部分問題に対し，分枝操作と限定操作を行う．分枝操作では，部分問題 P_v を $|S|$ 個の問題に分割し，深さ優先で再帰的に探索を行う．限定操作では，各部分問題に対して実行可能解の重みの上界を計算し，不要な探索の枝刈りを行うことによって，計算時間を削減する．分枝限定法では，分枝操作における部分問題の分割方法や，限定操作における上界計算の精度および計算時間が性能を決める重要な要素である．

3.1 Östergård のアルゴリズム

Östergård によって，MWCP に対する分枝限定アルゴリズムが提案されている [4]．Östergård のアルゴリズムでは，はじめに何らかの方法でソートを行い，頂点系列 $[v_n, v_{n-1}, \cdots, v_1]$ を得る．その頂点系列に対し，頂点集合 $\{v_i, v_{i-1}, \cdots, v_1\}$ を V_i と記す．以降，$P_v(\emptyset,V_1), P_v(\emptyset,V_2), \cdots, P_v(\emptyset,V_n)$ の順に，部分問題の最適解を分枝限定法で求める．このとき，求められた各 $P_v(\emptyset,V_i)$ の最適解の重みは，配列 $c[i]$ に保存される．$V_n = V$ であるため，Östergård のアルゴリズムは最終的に $P_v(\emptyset,V)$ の最適解，すなわち入力グラフの最適解を得ることができる．

配列 $c[\cdot]$ は限定操作で上界として利用される．部分問題 $P_v(C,S)$ から得られる任意の実行可能界を F とする．また $i = \max\{j \mid v_j \in S\}$ とする．このとき，$S \subseteq V_i$ であるため，

$$\begin{aligned} W_v(F) &= W_v(C \cap F) + W_v(S \cap F) \\ &= W_v(C) + W_v(S \cap F) \\ &\leq W_v(C) + W_v(V_i \cap F) \\ &\leq W_v(C) + c[i] \end{aligned}$$

が成立する．各 $P_v(C,S)$ に対する分枝操作では，上界である $W_v(C) + c[i]$ をできるだけ小さくするために，インデックスが大きい頂点を C に含む部分問題から順に探索を行う．

3.2 最長路法

MWCP に対する上界計算法として，最長路法が提案されている [5]。ここでは，文献 [5] で提案されている方法のうち，本稿の提案法で用いる部分のみ説明する．部分問題 $P_v(C,S)$ に対し，頂点誘導部分グラフ $G(S)$ の各辺を，任意の向きの有向辺に置き換えたグラフを $\vec{D}(G(S))$ とする。“路の長さ” を，路上の頂点の重みの和と定める。$\vec{D}(G(S))$ の最長路の長さを $LP(\vec{D}(G(S)))$ と記す。$P_v(C,S)$ の任意の実行可能解を F とする。$\vec{D}(G(S))$ には，$F \cap S$ の全頂点を通る路が必ず存在するため，以下の関係が成り立つ。

$$\begin{aligned} W_v(F) &= W_v(C \cap F) + W_v(S \cap F) \\ &= W_v(C) + W_v(S \cap F) \\ &\leq W_v(C) + LP(\vec{D}(G(S))) \end{aligned}$$

よって，最長路を計算することにより，部分問題の実行可能解の重みの上界を得ることができる。[5] では，最長路の長さを短くするための，効率のよい $\vec{D}(G(S))$ の作成方法などが提案されている。

4. 提案法 EWCLIQUE

提案法は，分枝限定法に基づくアルゴリズムである．提案法を Algorithm 1,2,3 に示す．MEWCP に対する分枝限定法で扱う部分問題を $P_e(C,S)$ と記す．MWCP の部分問題と同様，C はクリークであり，S はクリークに加える頂点の候補集合を意味する．S は $\forall v \in S, C \subseteq N(v)$ を満たす．入力グラフ $G=(V,E)$ に対応する部分問題は $P_e(\emptyset,V)$ である．

提案法は，はじめに小さい部分問題を解き，次にそれよりも少し大きい部分問題を解き，という操作を繰り返し，最終的に元の問題の解を得る．限定操作では，一部の辺の重みを仮の頂点重みに変換し，複数の上界計算法を組み合わせることで，実行可能解の重みの上界を得る．

4.1 では提案法の分枝操作について述べ，4.2 で提案法の限定操作について述べる．

Algorithm 1 EWCLIQUE

INPUT: $G=(V,E)$, $w_e(\cdot,\cdot)$
OUTPUT: the maximum edge-weight clique C_{max}
GLOBAL VARIABLES: C_{max}, $c[\cdot]$

1: Calculate $\sum_{u \in N(v)} w_e(v,u)$ for all $v \in V$.
2: Create a vertex sequence of nonincreasing $\sum_{u \in N(v)} w_e(v,u)$.
3: $C_{max} \leftarrow \emptyset$
4: **for** i from 1 to n **do**
5: EXPAND$(\emptyset, V_i)$
6: $c[i] \leftarrow W_e(C_{max})$ ▷ After EXPAND$(\emptyset, V_i)$, C_{max} is the maximum edge-weight clique of $G(V_i)$.
7: **end for**
8: **return** C_{max}

Algorithm 2 Solving a subproblem

INPUT: a subproblem $P_e(C, S)$
OUTPUT: Update C_{max} if better cliques are found.
GLOBAL VARIABLES: C_{max}, $c[\cdot]$

```
 1: procedure EXPAND(C, S)
 2:     lp[·] = LONGESTPATH(C, S)
 3:     while S ≠ ∅ do
 4:         i ← max{j | v_j ∈ S}
 5:         if W_e(C) + c[i] + lp[i] ≥ W_e(C_max) then
 6:             EXPAND(C ∪ {v_i}, S ∩ N(v_i))
 7:         end if
 8:         S ← S \ {v_i}
 9:     end while
10:     if W_e(C) > W_e(C_max) then
11:         C_max ← C
12:     end if
13: end procedure
```

4.1 分枝操作

提案法の分枝操作は，MWCP に対するÖstergård のアルゴリズム [4] と同じ方法を用いる．はじめに，頂点集合 V の各頂点 v に対し，v に接続する辺の重みの和 $\sum_{u\in N(v)} w_e(v,u)$ を計算する (Algorithm 1, 1 行目)．次に，$\sum_{u\in N(v)} w_e(v,u)$ の降順でソートされた頂点系列 $[v_n, v_{n-1}, \cdots, v_1]$ を計算する．

以降，Östergård のアルゴリズムと同様に，$P_e(\emptyset, V_1), P_e(\emptyset, V_2), \cdots, P_e(\emptyset, V_n)$ の順に，部分問題の最適解を分枝限定法で求める．得られた各 $P_e(\emptyset, V_i)$ の最適解の重みは配列 $c[i]$ に保存される．

各部分問題 $P_e(C, S)$ に対しては，インデックスの大きい頂点をクリーク C に含む部分問題から順番に探索を行う (Algorithm 2, 6 行目)．

4.2 限定操作

提案法は，Algorithm 2 の 5 行目で，上界による枝刈りを行う．部分問題 $P_e(C, S)$ の任意の実行可能解を F とする．このとき，

$$\begin{aligned} W_e(F) &= W_e(C\cap F) + W_e(S\cap F) \\ &\quad + \sum_{u\in C\cap F}\sum_{v\in S\cap F} w_e(u,v) \\ &= W_e(C) + W_e(S\cap F) + \sum_{u\in C}\sum_{v\in S\cap F} w_e(u,v) \end{aligned}$$

が成り立つ．すなわち，F の評価値 $W_e(F)$ を計算するためには，図 1 に示す 3 つの構成要素を考慮する必要がある．

$W_e(C)$ の正確な値は分枝操作の際に計算済みである．よって，$W_e(F)$ の上界を計算するためには，以下の 2 つを考慮する必要がある．

上界 1 $W_e(S\cap F)$ の上界
上界 2 $\sum_{u\in C}\sum_{v\in S\cap F} w_e(u,v)$ の上界

提案法はこれらを計算し，$W_e(F)$ の上界として利用する．上界 1 の計算法について 4.2.1 で，上界 2 の計算法について 4.2.2 でそれぞれ述べる．

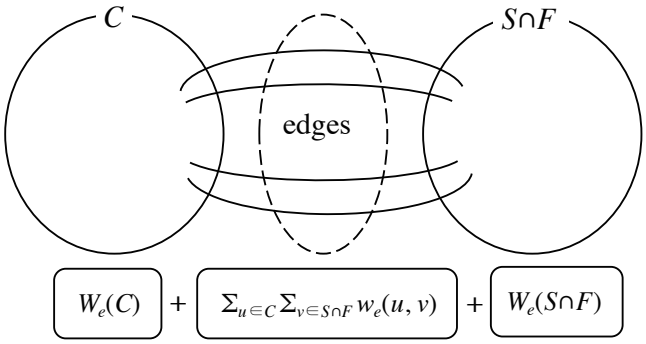

図 1 Components of $W(F)$ calculation

4.2.1 上界 1: $W(S\cap F)$ の上界

$P_e(C, S)$ に対し，$i = \max\{j \mid v_j \in S\}$ とする．$S \subseteq V_i$ であるため，以下の関係が成り立つ．

$$W_e(S\cap F) \le W_e(V_i\cap F) \le c[i]$$

よって，$c[i]$ は，上界 1 として利用することができる．

4.2.2 上界 2: $\sum_{u\in C}\sum_{v\in S\cap F} w_e(u,v)$ の上界

提案法は部分問題 $P_e(C, S)$ の各 $v \in S$ に対し，仮の頂点重み $\rho(C, v) = \sum_{u\in C} w_e(v, u)$ を導入する．$\rho(C, v)$ は，v と各 $u \in C$ との間の辺の重みを，仮の頂点重み (pseudo vertex weight) として v に割り当てたものである．$W_\rho(V') = \sum_{v\in V'} \rho(C, v)$ と定める．このとき，

$$W_\rho(S\cap F) = \sum_{u\in C}\sum_{v\in S\cap F} w_e(u,v)$$

が成り立つ．仮の頂点重みにより，MWCP の上界計算法を用いて上界 2 を計算することが可能になる．提案法は MWCP に対する上界計算法である最長路法 [5] を仮の頂点重みに対して適用し， 上界 2 を計算する (Algorithm 3)．このとき，辺の重みは無視する．

提案法は，最長路法を用いるための有向グラフ $\vec{D}(G(S))$ を以下のように生成する。各辺 (v_i, v_j), $i < j$ を，インデックスの小さい頂点 v_i からインデックスの大きい頂点 v_j へと向きを付けた有向辺に置き換える。

最長路法により，以下の関係が成り立つ．

$$\begin{aligned} \sum_{u\in C}\sum_{v\in S\cap F} w_e(u,v) &= W_\rho(S\cap F) \\ &\le LP(\vec{D}(G(S))) \end{aligned}$$

以上より，提案法は $LP(\vec{D}(G(S)))$ を上界 2 として利用する． そのために，提案法は分枝操作の度に，各 $v_i \in S$ に対して，$LP(\vec{D}(G(S\cap N(v_i)\cap V_i)))$ を計算し，配列 $lp[i]$ に格納する (Algorithm 2, 2 行目)．限定操作の際には，配列 $lp[\cdot]$ に格納された値を上界として用いる (Algorithm 2, 5 行目)．

Algorithm 3 Calculate longest path

INPUT: a subproblem $P_e(C, S)$
OUTPUT: an array $lp[\cdot]$ that contains length of longest path

procedure LONGESTPATH(C, S)
  $S' \leftarrow S$
  while $S' \neq \emptyset$ **do**
    $i \leftarrow \min\{j \mid v_j \in S'\}$
    if $S \cap N(v_i) \cap V_i = \emptyset$ **then**
      $lp[i] \leftarrow \rho(C, v_i)$
    else
      $lp[i] \leftarrow \rho(C, v_i) + \max\{lp[u] \mid u \in S \cap N(v_i) \cap V_i\}$
    end if
    $S' \leftarrow S' \setminus \{v_i\}$
  end while
  return $lp[\cdot]$
end procedure

5. 計算機実験

提案法 EWCLIQUE を C++で実装し，計算機実験を行った．比較対象は，MEWCP を二次計画問題と整数計画問題で定式化し，それぞれを IBM の数理計画ソルバ CPLEX 12.5 で解いたものである．

5.1 実験環境

使用したコンパイラは g++ 5.4.0，最適化オプションは -O2 である．実験に使用した計算機の OS は Linux 4.4.0，CPU は Intel®Core^{TM}i7-6700 CPU 3.40 GHz，メモリは 16GB である．

実験に用いた入力は，一様ランダムグラフである．辺の重みは 1 から 10 の整数値とした．全ての条件で，乱数の種の異なるグラフを 10 個ずつ生成し，それらを解く計算時間の平均値を計測した

5.2 実験結果

表 1 にランダムグラフに対する実験結果を示す．表中の d は辺密度 $\frac{|E|}{|V|C_2} = \frac{2|E|}{|V|(|V|-1)}$ を表す．提案法は MEWCP を解く分枝限定アルゴリズムであり，CPLEX は数理計画問題を解く分枝カットアルゴリズムであるため，直接の比較はできないが，参考のため，探索木のサイズも表 1 に示す．

全ての条件で，提案法は CPLEX(IP) および CPLEX(QP) に比べ，非常に短い時間で厳密解を得ることができた．CPLEX が数百秒かかるインスタンスでも，提案法は，数十ミリ秒で解いている．また，CPLEX は $|V|$ が数百程度のインスタンスしか解くことができないが，辺密度 d が小さい場合，提案法は $|V| = 15000$ のインスタンスでも解を得ることができた．以上より，提案法は MEWCP を数理計画問題に定式化して，解く方法に比べ，非常に性能が良いと言える．

6. まとめ

本稿では，MEWCP に対する分枝限定アルゴリズム EWCLIQUE を提案した．EWCLIQUE は部分問題の一部の辺の重みを仮の頂点重みに変換する．残った辺の重みと，仮の頂点重みのそれぞれに対して上界を計算することで，最大辺重みクリークの重みの上界を得る．

MEWCP を数理計画問題に変換して数理計画ソルバに解かせる方法と，提案法 EWCLIQUE を計算機実験によって比較し，提案法の性能が優れていることを確認した．

参考文献

[1] Garey, M. R. and Johnson, D. S.: *Computers and Intractability: A Guide to the Theory of NP-completeness*, WH Freeman and Company, New York (1979).
[2] Tomita, E. and Kameda, T.: An efficient branch-and-bound algorithm for finding a maximum clique with computational experiments, *Journal of Global optimization*, Vol. 37, No. 1, pp. 95–111 (2007).
[3] Tomita, E., Sutani, Y., Higashi, T., Takahashi, S. and Wakatsuki, M.: A simple and faster branch-and-bound algorithm for finding a maximum clique, *WALCOM: Algorithms and computation*, Springer, pp. 191–203 (2010).
[4] Östergård, P. R.: A new algorithm for the maximum-weight clique problem, *Nordic Journal of Computing*, Vol. 8, No. 4, pp. 424–436 (2001).
[5] Yamaguchi, K. and Masuda, S.: A new exact algorithm for the maximum weight clique problem, 23rd International Conference on Circuits/Systems, Computers and Communictions (ITC-CSCC' 08), pp. 317–320 (2008).
[6] Shimizu, S., Yamaguchi, K., Saitoh, T. and Masuda, S.: Optimal Table Method for Finding the Maximum Weight Clique, *WSEAS International Conference. Proceedings. Recent Advances in Computer Engineering Series*, No. 12, WSEAS (2013).
[7] Sørensen, M. M.: New facets and a branch-and-cut algorithm for the weighted clique problem, *European Journal of Operational Research*, Vol. 154, No. 1, pp. 57–70 (2004).
[8] Martí, R., Gallego, M. and Duarte, A.: A branch and bound algorithm for the maximum diversity problem, *European Journal of Operational Research*, Vol. 200, No. 1, pp. 36–44 (2010).
[9] Gouveia, L. and Martins, P.: Solving the maximum edge-weight clique problem in sparse graphs with compact formulations, *EURO Journal on Computational Optimization*, Vol. 3, No. 1, pp. 1–30 (2015).

表 1 CPU time for randomgraphs[sec]

		Computation time [sec]			Number of Search tree nodes		
$\|V\|$	d	EWCLIQUE	CPLEX(IP)	CPLEX(QP)	EWCLIQUE	CPLEX(IP)	CPEX(QP)
300	0.1	less than 0.01	124.94	313.46	2043.2	77165.7	193796.7
350	0.1	less than 0.01	258.84	706.68	2991.8	102579.4	308754.9
15000	0.1	455.00	over 1000	over 1000	720818041.5	-	-
250	0.2	less than 0.01	315.26	250.55	7405.5	704421.5	438526.7
280	0.2	less than 0.01	664.09	417.06	10504.1	1243089.2	636879.1
5500	0.2	443.50	over 1000	out of memory	1556794986.6	-	-
200	0.3	less than 0.01	470.07	154.74	17218.1	1375256.2	923563.2
250	0.3	0.02	over 1000	531.50	39760.2	-	2301720.2
2500	0.3	482.60	over 1000	out of memory	1951311370.4	-	-
160	0.4	0.01	528.41	97.61	35977.1	2058272.9	1563023.3
200	0.4	0.02	over 1000	428.47	89147.9	-	4983953.9
1400	0.4	777.40	over 1000	over 1000	3063389386.8	-	-
140	0.5	0.02	556.29	111.73	111264.0	2933500.6	3028762.4
170	0.5	0.07	over 1000	498.58	274576.5	-	9767768.2
750	0.5	650.30	over 1000	over 1000	2586024061.7	-	-
120	0.6	0.07	405.38	129.01	341565.5	3274026.2	4976548.2
130	0.6	0.10	634.81	256.59	452163.7	4496672.8	8570931.1
450	0.6	758.70	over 1000	over 1000	2843290969.5	-	-
100	0.7	0.17	262.75	125.42	724350.4	3213925.0	6703943.1
110	0.7	0.37	557.70	356.28	1654286.2	5078319.2	15772011.0
270	0.7	713.75	over 1000	over 1000	2605915210.5	-	-
80	0.8	0.43	135.77	71.41	1970465.6	2107836.8	5237500.2
90	0.8	1.14	386.01	266.13	4790318.1	5293378.0	16189357.9
160	0.8	471.24	over 1000	over 1000	1662516877.7	-	-
70	0.9	3.87	50.80	16.69	16770613.2	993629.0	1335799.1
80	0.9	22.97	181.36	118.65	93445789.5	2941909.6	7694204.9
100	0.9	518.21	over 1000	over 1000	1900897874.9	-	-