Polynomial-Time Solvability of the Maximum Clique Problem

ETSUJI TOMITA*, HIROAKI NAKANISHI
Advanced Algorithms Research Laboratory, and
Department of Information and Communication Engineering,
The University of Electro-Communications
Chofugaoka 1-5-1, Chofu, Tokyo 182-8585, JAPAN
{tomita, hironaka}@ice.uec.ac.jp

Abstract: The maximum clique problem is known to be a typical NP-complete problem, and hence it is believed to be impossible to solve it in polynomial-time. So, it is important to know a reasonable sufficient condition under which the maximum clique problem can be proved to be polynomial-time solvable. In this paper, given a graph of n vertices and whose maximum degree is Δ, we prove that if Δ is less than or equal to $2.493d \lg n$ ($d \geq 1$: a constant), then the maximum clique problem is solvable in the polynomial time of $O(n^{2+d})$. The proof is based on a very simple algorithm which is obtained from an algorithm CLIQUES that generates all maximal cliques in a depth-first way in $O(3^{n/3})$-time (which is published in *Theoretical Computer Science* 363, 2006, as "The worst-case time complexity for generating all maximal cliques and computational experiments" by E. Tomita *et al*.). The proof itself is very simple.

Key–Words: Maximum clique, NP-complete, Time-complexity, Polynomial-time, Graph, Algorithm

1 Introduction

It is generally believed that any NP-complete problem cannot be solved in polynomial-time, and it is well known that if any one of the NP-complete problems could be solved in polynomial-time, then all NP-complete problems would become polynomial-time solvable. Considerable effort has been expended to find reasonable conditions under which some NP-complete problems can be proved to be polynomial-time solvable [13].

The maximum clique problem [13], [4], or the complementary problem, the maximum independent set problem [13], is one of the original 21 problems shown to be NP-complete by R. Karp [18]. Much work has been done on this problem, theoretically and experimentally with many applications, see, e.g., [13], [4], [16]. The maximum clique problem is known to be polynomial-time solvable for some special graphs such as planar graphs [13], chordal graphs [9], comparability graphs [7], circle graphs [10], and circular-arc graphs [11], [2]. The maximum independent set problem is also known to be polynomial-time solvable for some special graphs such as bipartite graphs [21], chordal graphs [9], circle graphs [10], and circular-arc graphs [11], [19], comparability graphs [12], and claw-free graphs [20].

For general graphs, when the maximum degree Δ is a constant, then the size of a maximum clique is bounded above by $\Delta + 1$ and hence the maximum clique problem is polynomial-time solvable [13]. The maximum clique decision problem for $k = 3$ is solvable in $O(m^{1.41})$-time [1], where m is the number of edges and the decision problem is whether there exists a maximum clique whose size is at least k (see (5) in Section 2 of this paper). If Δ is at most 2, then the maximum independent set problem is polynomial-time solvable [13].

Experience shows that a maximum clique can be found easily if the edge density of graphs is sparse, see e.g., [17] (for the complementary problem), [27], [29]. However, as yet, a nontrivial, exact condition is not known for a general graph under which the maximum clique problem can be proved to be solvable in polynomial-time. Such conditions allowing an exact solution in polynomial-time are important because satisfactory approximate solutions are difficult to achieve [14]. Theoretical time-complexity analysis of the exponential order of the number of ver-

*Corresponding author. Jointly with Research and Development Initiative, Chuo University, Kasuga 1-13-27, Bunkyo-ku, Tokyo 112-8551, JAPAN.

ISSN: 1790-5117

ISBN: 978-960-474-088-8

tices for the maximum clique problem, or the maximum independent set problem includes [26], [15], [23], [25], [3], [24], [6], [8], [22]. Among them, an algorithm MAXCLIQUE of $O(2^{n/2.863})$-time in [25] is simple and runs fast in practice, but the theoretical time-complexity analysis is very complicated.

In this paper, we prove that the maximum clique problem is polynomial-time solvable for a general graph if the maximum degree Δ of the graph in question is in a logarithmic-order of the number n of vertices of the graph. More specifically, if Δ is less than or equal to $2.493d\lg n$ ($d \geq 1$: a constant), then the maximum clique problem is solvable in $O(n^{2+d})$-time.

Prior to this paper, we proved that all maximal cliques can be generated in $O(3^{n/3})$-time, which is optimal as a function of n. The result was proved on an algorithm CLIQUES that generates all maximal cliques, in which pruning methods are employed, as in the Bron-Kerbosch algorithm [5].

The present polynomial-time-complexity result is based on an algorithm MCP_0 for finding a maximum clique, which is a slightly simplified version of CLIQUES. MCP_0 is similar to our previous algorithm MAXCLIQUE, but the time-complexity analysis is very simple.

2 Definitions and Notation

(1) We are concerned with a simple undirected graph $G = (V, E)$ with a finite set V of *vertices* and a finite set E of *unordered* pairs (v, w) of distinct vertices, called *edges*. A pair of vertices v and w are said to be *adjacent* if $(v, w) \in E$.

For a set V, $|V|$ denotes the number of elements in V.

(2) For a vertex $v \in V$, let $\Gamma(v)$ be the set of all vertices that are adjacent to v in $G = (V, E)$, i.e., $\Gamma(v) = \{w \in V \mid (v, w) \in E\}$ ($\not\ni v$).

The number of vertices in $\Gamma(v)$ is called the *degree* of v.

(3) For a subset $W \subseteq V$ of vertices, $G(W) = (W, E(W))$ with $E(W) = \{(v, w) \in E \mid v, w \in W\}$ is called a *subgraph* of $G = (V, E)$ *induced* by W.

(4) Given a subset $Q \subseteq V$ of vertices, the induced subgraph $G(Q)$ is said to be a *clique* if $(v, w) \in E$ for all $v, w \in Q$ $(v \neq w)$. In this case, we may simply state that Q is a clique. If a clique is not a proper subgraph of another clique, then it is called a *maximal* clique.

(5) *The Maximum Clique Problem* is defined here to be such a *Decision Problem* that answers, given a graph G and a positive integer k, whether the number of vertices of the maximum clique of G is at least k.

procedure MCP$_0(G)$
begin
  $global\ Q := \emptyset$;
  $global\ Qmax := \emptyset$;
  EXPAND(V)
end {of MCP_0}

  procedure EXPAND$(SUBG)$
  begin
   if $SUBG = \emptyset$ **then**
    if $|Q| > |Qmax|$
     then $Qmax := Q$ **fi**
   else u :=a vertex with the maximum degree
      in the subgraph induced by $SUBG$;
    $Q := Q \cup \{u\}$;
    $SUBG_u := \Gamma(u) \cap SUBG$;
    EXPAND$(SUBG_u)$
    $Q := Q - \{u\}$;
    $EXT_u := SUBG - \{u\} - SUBG_u$;
    for $i := 1$ **to** $|EXT_u| - 1$
     do
     v_i :=the first vertex in EXT_u;
     $SUBG_{v_i} := \Gamma(v_i) \cap (EXT_u \cup SUBG_u)$;
     $Q := Q \cup \{v_i\}$;
     EXPAND$(SUBG_{v_i})$;
     $Q := Q - \{v_i\}$;
     $EXT_u := EXT_u - \{v_i\}$
     od
  end {of EXPAND}

Fig.1 Algorithm MCP_0

3 Algorithm MCP_0

Our proof of the main result of polynomial-time solvability is based on a simple algorithm MCP_0 that finds a maximum clique. MCP_0 is a modified version of an algorithm CLIQUES [28] that generates all maximal cliques in a depth-first way in $O(3^{n/3})$-time. Hence, MCP_0 is simpler than CLIQUES because the former

ISSN: 1790-5117

ISBN: 978-960-474-088-8

has to output only the maximum among all the maximal cliques.

3.1 A Basic Algorithm

Our algorithm finds maximal cliques of increasing size, in a stepwise manner, until it arrives at a maximum clique. More precisely, we maintain global variables Q and Q_{max}, where Q consists of vertices of a current clique, and Q_{max} consists of vertices of the largest clique found so far, respectively. Let $SUBG \subseteq V$ consist of vertices (candidates) that may be added to Q. We begin the algorithm by letting $Q := \emptyset$, $Q_{max} := \emptyset$, and $SUBG := V$ (the set of all vertices). We select a certain vertex v from $SUBG$ and add v to Q ($Q := Q \cup \{v\}$). Then we compute $SUBG_v = SUBG \cap \Gamma(v)$ as a new set of candidate vertices. This procedure (EXPAND) is applied recursively, while $SUBG_v \neq \emptyset$.

When $SUBG_v = \emptyset$ is reached, Q constitutes a maximal clique. If Q is maximal and $|Q| > |Q_{max}|$ holds, Q_{max} is replaced by Q. We then backtrack by removing v from Q and $SUBG$. We select a new vertex w from the resulting $SUBG$ and continue the same procedure until $SUBG = \emptyset$. This is a well known basic algorithm for finding a maximum clique (see, e.g., [29]). In general, when a current clique is $Q = \{p_1, p_2, \ldots, p_d\}$ then

$$SUBG = V \cap \Gamma(p_1) \cap \Gamma(p_2) \cap \ldots \cap \Gamma(p_d).$$

3.2 Exclusion of Adjacent Vertices

In this basic algorithm, first we choose a vertex u with the maximum degree in the subgraph induced by $SUBG$. Then, we get a set $SUBG_u$ of vertices that are adjacent to u, and a set EXT_u of vertices that are not adjacent to u, i.e.,

$$SUBG_u = \Gamma(u) \cap SUBG, \quad \text{and}$$
$$EXT_u = (SUBG - \{u\}) - SUBG_u.$$

Then, we consider a set $\{u\} \cup EXT_u \cup SUBG_u$ arranged in this order from left to right to be a newly ordered set $SUBG$ of vertices. Note that for any maximal clique Q in $SUBG_u$, we always have a $larger$ clique $Q \cup \{u\}$ because every vertex in $Q \subseteq SUBG_u = \Gamma(u) \cap SUBG$ is adjacent to u. Therefore, when all the expansions from vertex u are made to search for a maximum clique, we can exclude searching from vertices in $SUBG_u$. Such a pruning technique is also used in [5], [28], and we call it an Exclusion of Adjacent Vertices.

3.3 Exclusion of the Last Vertex in EXT_u

As described in 3.2, for the set $SUBG = \{u\} \cup EXT_u \cup SUBG_u$ of vertices, we have to expand searching only from $\{u\} \cup EXT_u$. We let

$$EXT_u = \{v_1, v_2, ..., v_{|EXT_u|}\}.$$

and we apply searching from left to right step by step. In this case, we need not expand searching from the last vertex $v_{|EXT_u|}$. The reason is as follows. If the last vertex $v_{|EXT_u|}$ were to be expanded, it should be after all of $u, v_1, v_2, ..., v_{|EXT_u|-1}$ have been deleted. Then, we have

$$\begin{aligned} SUBG_{v_{|EXT_u|}} &= \Gamma(v_{|EXT_u|}) \cap (\{v_{|EXT_u|}\} \cup SUBG_u) \\ &= \Gamma(v_{|EXT_u|}) \cap SUBG_u \\ &\subseteq SUBG_u. \end{aligned}$$

Thus, the expansion from $v_{|EXT_u|}$ cannot find a larger clique than that in $SUBG_u$.

The process of searching for a maximum clique by MCP_0 is represented by a search forest, i.e., a collection of search trees [28]. (See, e.g., Fig. 3 in [28].) Here, for a vertex v, every vertex in $\Gamma(v)$ is a child of v in the search forest.

4 The Worst Case Time-Complexity

Given $G = (V, E)$ with $V \neq \emptyset$, we evaluate the worst-case running time of the algorithm MCP_0. This is equivalent to evaluating the worst-case running time of EXPAND(V).

Let $T(n) = T(|SUBG|)$ be the worst-case running time of EXPAND($SUBG$) when $|SUBG| = n$.

Let us consider a non-recursive procedure $EXPAND_0(SUBG)$ that is obtained from EXPAND($SUBG$) by replacing recursive calls EXPAND($SUBG_u$) and EXPAND($SUBG_{v_i}$) with EXPAND($\emptyset$) and EXPAND($\emptyset$), respectively. The running time of $EXPAND_0(SUBG)$ when $|SUBG| = n$ can be made to be $O(n^2)$ as in [28], and so we assume that the running time of $EXPAND_0(SUBG)$ is bounded above by $Cn^2 = C|SUBG|^2$ for some constant C.

Then, we have the following lemma.

Lemma 1. For a subgraph induced by a set $SUBG$ of vertices, the worst-case running time $T(n) = T(|SUBG|)$ of EXPAND($SUBG$) is as follows:

$$T(|SUBG|) \leq T(|SUBG_u|) + \sum_{i=1}^{|EXT_u|-1} T(|SUBG_{v_i}|) + C|SUBG|^2,$$

where u is a vertex with the maximum degree in the subgraph induced by $SUBG$, $SUBG_u = \Gamma(u) \cap SUBG$, $EXT_u = SUBG - \{u\} - SUBG_u = \{v_1, v_2, ..., v_{|EXT_u|}\}$, and $SUBG_{v_i} = \Gamma(v_i) \cap ((EXT_u - \{v_1, v_2, ..., v_{i-1}\}) \cup SUBG_u)$.

Proof. This is obvious from the procedure EXPAND($SUBG$) and the definition of the constant C. □

To prove the main theorem, we prove the following important lemmas with regard to the maximum degree Δ of the graph in question.

Lemma 2. Consider a subgraph induced by a set $SUBG$ of vertices. Let the maximum degree of the subgraph be $\Delta \geq 0$, and let $C' = 250C$. Then the worst case time complexities of EXPAND($SUBG_u$) and EXPAND($SUBG_{v_i}$) are as follows (where $|SUBG_u| \leq \Delta, |SUBG_{v_i}| \leq \Delta$) :

$$T(|SUBG_u|) \leq C'2^{0.4009\Delta}(\Delta+1)^2,$$
$$T(|SUBG_{v_i}|) \leq C'2^{0.4009\Delta}(\Delta+1)^2 \quad (1 \leq i \leq |EXT_u| - 1).$$

Proof. The proof is by induction on the maximum degree Δ.

To begin with, we consider the case where $\Delta = 0$. Then, these inequalities simply hold, because $SUBG_u = \emptyset, SUBG_{v_i} = \emptyset$.

Next, we assume that the following inequalities hold for all nonnegative integers Δ that are less than or equal to some fixed value:

$$T(|SUBG_u|) \leq C'2^{0.4009\Delta}(\Delta+1)^2,$$
$$T(|SUBG_{v_i}|) \leq C'2^{0.4009\Delta}(\Delta+1)^2 \quad (1 \leq i \leq |EXT_u| - 1),$$

and consider the case where the maximum degree of the subgraph induced by $SUBG$ is $(\Delta + 1)$. Let u be the vertex in $SUBG$ with the maximum degree $(\Delta + 1)$, and let $SUBG_u = SUBG \cap \Gamma(u)$, then $|SUBG_u| = \Delta + 1$ and the maximum degree of children of vertex u in the search forest is less than or equal to Δ. Then, the induction hypothesis applies for $SUBG_u$. We let the maximum degree of children of vertex u be $\Delta - k$ $(0 \leq k \leq \Delta)$.

From Lemma 1 and the induction hypothesis to $SUBG_u$, we can prove the following:

$$\begin{aligned}
&T(|SUBG_u|) \\
&\leq ((\Delta+1) - (\Delta-k) - 1) \cdot C'2^{0.4009(\Delta-k)}((\Delta-k)+1)^2 + C(\Delta+1)^2 \\
&\leq kC'2^{0.4009(\Delta-k)}(\Delta+1)^2 + C(\Delta+1)^2 \\
&< C'2^{0.4009\Delta}\left(\frac{k}{2^{0.4009k}} + \frac{1}{250 \cdot 2^{0.4009\Delta}}\right)(\Delta+2)^2 \\
&\leq C'2^{0.4009\Delta}\left(\frac{k}{2^{0.4009k}} + \frac{1}{250}\right)(\Delta+2)^2.
\end{aligned}$$

We have that $\frac{k}{2^{0.4009k}} < 1.3163$ for all $k \geq 0$, then

$$\begin{aligned}
&T(|SUBG_u|) \\
&< C'2^{0.4009\Delta}(1.3163 + 0.004)(\Delta+2)^2 \\
&= C'2^{0.4009\Delta} \cdot 1.3203 \cdot (\Delta+2)^2 \\
&< C'2^{0.4009\Delta} \cdot 2^{0.4009}(\Delta+2)^2 \\
&= C'2^{0.4009(\Delta+1)}((\Delta+1)+1)^2.
\end{aligned}$$

In the same way as above, we can prove that

$$T(|SUBG_{v_i}|) \leq C'2^{0.4009(\Delta+1)}((\Delta+1)+1)^2 \quad (1 \leq i \leq |EXT_u| - 1).$$

Thus, the objective inequalities also hold for $\Delta + 1$.

Therefore, the objective inequalities hold for all $\Delta \geq 0$.

Hence, the result. □

Lemma 3. Consider a graph with n vertices whose maximum degree is $\Delta \geq 0$. Let us define some constants as $C' = 250C$, $C'' = 3.0 \cdot 10^{11}$, and $C''' = C' \cdot C'' + C$.

The worst-case time-complexity $T(n) = T(|SUBG|)$ of EXPAND($SUBG$) is as follows:

$$T(n) \leq C'''2^{0.401\Delta}n^2.$$

Proof. From Lemma 1, we have

$$T(n) \leq T(|SUBG_u|) + T(|SUBG_{v_1}|) + T(|SUBG_{v_2}|) + ... + T(|SUBG_{v_{|EXT_u|-1}}|) + Cn^2.$$

Then, by Lemma 2, we have

$$\begin{aligned}
&T(n) \\
&\leq (n - \Delta - 1) \cdot C'2^{0.4009\Delta}(\Delta+1)^2 + Cn^2 \\
&\leq (n-1) \cdot C'2^{0.4009\Delta}(\Delta+1)^2 + Cn^2.
\end{aligned}$$

Here, from the definition of the constant $C'' = 3.0 \cdot 10^{11}$, we can easily prove that

$$(\Delta + 1)^2 < C''2^{0.0001\Delta}$$

holds for all Δ where $0 \leq \Delta \leq n-1$.

Therefore,

$$\begin{aligned}
&T(n) \\
&\leq (n-1) \cdot C'2^{0.4009\Delta} \cdot C''2^{0.0001\Delta} + Cn^2 \\
&< n^2 C'C''2^{0.4009\Delta + 0.0001\Delta} + Cn^2 \\
&= n^2 C''C'2^{0.401\Delta} + Cn^2 \\
&= 2^{0.401\Delta} n^2 (C'C'' + \tfrac{C}{2^{0.401\Delta}}) \\
&\leq (C'C'' + C)2^{0.401\Delta} n^2 \\
&= C'''2^{0.401\Delta} n^2. \quad \square
\end{aligned}$$

Now, we have the main result of this paper.

Theorem. Given a graph with n vertices, if the maximum degree $\Delta \leq 2.493 d \lg n$ ($d \geq 1$: a constant), then the maximum clique problem is solvable in $O(n^{2+d})$-time.

Proof. When the inequality $\Delta \leq 2.493 d \lg n$ holds in Lemma 3, we have the following:

$$\begin{aligned}
T(n) &\leq C'''2^{0.401 \cdot 2.493 d \lg n} n^2 \\
&< C''' n^d \cdot n^2 = C''' n^{2+d}.
\end{aligned}$$

Therefore, $T(n) = O(n^{2+d})$.

This result specifies an upper bound on the complexity of the NP-hard optimization problem of finding a maximum clique. The corresponding result for the NP-complete decision problem of the Maximum Clique Problem follows directly. $\square$

In particular, we have the following property.

Corollary. The maximum clique problem is solvable in $O(n^2)$-time when Δ is bounded above by a constant.

Proof. This is a direct consequence of Lemma 3. $\square$

5 Concluding remarks

When the prerequisite condition in the theorem is satisfied, the edge density of the graph is at most $(2.493 d \lg n)/(n-1)$. Thus, the theorem matches the experience as in [17], [27], [29].

As for polynomial-time solvability, exhaustive search could reach a similar conclusion as long as the maximum degree is a logarithmic order of the number of vertices. However, to the best of the authors' knowledge, no such explicit quantitative analysis result is ever reported. The constant 2.493 in this paper can be made larger by using other results for finding a maximum clique or a maximum independent set, but it is to be noticed that not only our present algorithm but also the proof of its time-complexity are straightforward and very simple.

The algorithm MCP_0 is considered to be reinforced by using the techniques in [29], [30] to improve the time-complexity of MCP_0. Our present technique is expected to be a new basis for better time-complexity analysis of the maximum clique problem of general graphs.

Acknowledgements:

The authors are grateful to Hiro Itoh and Eric Harley for their quite helpful suggestions and comments. They also wish to express their thanks to Takahiro Tamada who engaged in extensive computational experiments of some related algorithms. This work was supported in part by Grants-in-Aid for Scientific Research Nos. 16300001 and 19500010 from the Ministry of Education, Culture, Sports, Science and Technology, Japan, and Special Grant for the Strategic Information and Communications R&D Promotion Programme (SCOPE) Project by the Ministry of Internal Affairs and Communications, Japan. The authors were also given a grant by the Funai Foundation for Information Technology.

References:

[1] N. Alon, R. Yuster, U. Zwick, Finding and counting given length cycles, *Algorithmica*, 17, 1977, pp.209-223.

[2] B. K. Bhattacharya, An $O(m + n \log n)$ algorithm for the maximum-clique problem in circular-arc graphs, *J. of Algorithms*, 25, 1997, pp.336-358.

[3] R. Biegel, Finding maximum independent sets in sparse and general graphs, *Proc. Symp. on Discrete Algorithms*, 1999, pp.856-857.

[4] I. M. Bomze, M. Budinich, P. M. Pardalos, and M. Pelillo, The maximum clique problem, in: D.-Z. Du and P. M. Pardalos (Eds.). *Handbook of Combinatorial Optimization*, Supplement vol. A, Kluwer Academic Publishers, 1999, pp.1-74.

[5] C. Bron, J. Kerbosch, Algorithm 457, finding all cliques of an undirected graph, *Comm. ACM*, 16, 1973, pp.575-577.

[6] J. Chen, I. A. Kanji, G. Xia, Labeled search trees and amortized analysis: improved upper bounds for NP-hard problems, *Algorithmica*, 43, 2005, pp.245-273.

[7] S. Even, A. Pnueli, A. Lempel: Permutation graphs and transitive graphs, *J. Assoc. for Comput. Mach.* , 19, 1972, pp.400-410.

[8] F. V. Fomin, F. Grandoni, D. Kratsch, Measure and conquer: A simple $O(2^{0.288n})$ independent set algorithm, *Proc. Symp. on Discrete Algorithms*, 2006, pp.18-25.

[9] F. Gavril, Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph, *SIAM J. on Computing,* 1, 1972, pp.180-187.

[10] F. Gavril, Algorithms for a maximum clique and a maximum independent set of a circle graph, *Networks*, 3, 1973, pp.261-273.

[11] F. Gavril, Algorithms on circular-arc graphs, *Networks*, 4, 1974, pp.357-369.

[12] F. Gavril, Some NP-complete problems on graphs, *Proc. Conf. on Information Sciences and Systems*, 1977, pp.91-95.

[13] M. R. Garey, D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman and Company, New York, NY, USA, 1979.

[14] J. Håstad, Clique is hard to approximate within $n^{1-\varepsilon}$, *Acta Mathematica,* 182, 1999, pp.105-142.

[15] T. Jian, An $O(2^{0.304n})$ algorithm for solving maximum independent set problem, *IEEE Trans. on Computers*, 35, 1986, pp.847-851.

[16] D. S. Johnson, M. A. Trick (Eds), *Cliques, Coloring, and Satisfiability, DIMACS Series in Disc. Math. and Theoret. Comput. Sci*., vol. 26, American Math. Soc., 1996.

[17] D. S. Johnson, M. Szegedy, What are the least tractable instances of max independent set, *Proc. Symp. on Discrete Algorithms*, 1999, pp.927-928.

[18] R. Karp, Reducibility among combinatorial problems, in R. E. Miller, J. W. Thatcher (Eds.), *Proc. Complexity of Computer Computations*, Plenum Press, New York, 1972, pp.85-103.

[19] S. Masuda, K. Nakajima, An optimal algorithm for finding a maximum independent set of a circular-arc graph, *SIAM J. on Computing*, 17, 1988, pp.41-52.

[20] G. J. Minty, On maximal independent sets of vertices in claw-free graphs, *J. Combin. Theory*, Ser. B, 28, 1980, pp.284-304.

[21] R. Mosca, Polynomial algorithms for the maximum stable set problem on particular classes of P_5-free graphs, *Information Processing Letters*, 61, 1997, pp.137-143.

[22] I. Razgon, A faster solving of the maximum independent set problem for graphs with maximal degree 3, *ACiD 2006*, Durham, UK, 2006.

[23] J. M. Robson, Algorithms for maximum independent sets, *J. of Algorithms*, 7, 1986, pp.425-440.

[24] J. M. Robson, Finding a maximum independent set in time $O(2^{n/4})$, *Tech. Rep. 1251-01, LaBRI, Universite Bordeaux*, 2001.

[25] M. Shindo, E. Tomita, A simple algorithm for finding a maximum clique and its worst-case time complexity, *Systems and Computing in Japan*, 21, 1990, pp.1-13.

[26] R. E. Tarjan, A. E. Trojanowski, Finding a maximum independent set, *SIAM J. on Computing*, 6, 1977, pp.537-546.

[27] E. Tomita, T. Seki, An efficient branch-and-bound algorithm for finding a maximum clique, *Discrete Math. and Theoret. Comput. Sci. 2003, LNCS 2731*, 2003, pp.278-289.

[28] E. Tomita, A. Tanaka, H. Takahashi, The worst-case time complexity for generating all maximal cliques and computational experiments, (Invited paper for the special issue on COCOON 2004), *Theoret. Comput. Sci.*, 363, 2006, pp.28-42.

[29] E. Tomita, T. Kameda, An efficient branch-and-bound algorithm for finding a maximum clique with computational experiments, *J. Global Optimization*, 37, 2007, pp.95-111 .

[30] E. Tomita, The maximum clique problem and its applications (Invited lecture), *IPSJ SIG Tech. Rep.*, 2007-MPS-67, 2007, pp.21-24.