

# 先進的アルゴリズムに向けて

富田悦次

## Toward Advanced Algorithms

Etsuji TOMITA

### Abstract

We present some results toward advanced algorithms which were carried out in our laboratory and in the Advanced Algorithms Research Laboratory at the University of Electro-Communications. This paper consists of two parts. Part I deals with our historical and recent results on automata and language theory together with algorithmic learning theory. Part II deals with mainly our recent results on the maximum clique problem and its applications.

## Part I オートマトン・言語理論・学習理論

### 1. はじめに

Part I では、筆者の研究室、および「先進アルゴリズム研究ステーション」において、オートマトン・言語理論・学習理論に関する先進的アルゴリズム開発に向けて辿ってきた道をまとめる。なお、本Part Iは、文献[01]、[02]の一部に近況を加筆して修正を行ったものである。

### 2. オートマトン・言語理論・学習のアルゴリズム

筆者らがオートマトンの自動構成・適応的修正問題に取り組み[1]、[2]などの結果を発表した当時、いわゆるこのような「学習理論」に従事していたのは、国内では他に有川節夫教授（九大）などほんの一にぎりであり、国際的にも未だ極く少数であった。しかし、これらの結果はパターン認識機構の実現手段としても関心と呼んだ。

これらにおいて、オートマトンを構成するために必要な「代表記号列集合」の概念を与え、また[2]においては、後程 Angluin によって発表された正則言語の MAT 学習[3]における主要な概念を含んでいた。

この学習方式においては、オートマトンの（部分的）等価性判定（より直接的には、非等価性の判定とその証

拠（witness）の抽出）が重要となる。決定性有限オートマトンよりも上のクラスである決定性プッシュダウンオートマトン（DPDA）においては、当判定問題は未解決であったが、Valiantの博士論文[4]の結果等々、いくつかの部分クラスに対して巧妙な手法により次々と肯定的結果が与えられ、国内においても、谷口健一教授（阪大）ら[5]、大山口通夫教授（三重大）ら[6]などから精力的に成果が発表され、大きい関心を集めていた。

これに対して、筆者はそれまでの世界の大勢とは逆に、出来る限りアルゴリズムの単純さを追求した結果、新しく[7]における方式を確立した。その単純さは、同論文 Abstract 中の次の文章で表すことが出来るが、これは査読者の評価をそっくりそのまま引用させていただいたものである。

“This is the first time the branching algorithm has been used to give such a general decision procedure without ever “mixing” the two languages in question. In other words, it deals with only the equivalence equation whose left-hand side consists of a pure reachable configuration of one dpda and whose right-hand side that of the other”

---

Received on September 16, 2008.

Professor Emeritus, The University of Electro-Communications(Department of Information and Communication Engineering)

Professor, Research and Development Initiative, Chuo University

電気通信大学名誉教授（元情報通信工学科教授）

中央大学研究開発機構教授

この等価性判定法は更に効率化が可能であり[8]、具体例として図1上部の推移規則 $\delta_1$ 、 $\delta_2$ を持つ単純決定性プッシュダウンオートマトン $M_1$ 、 $M_2$ に対する等価性判定過程結果を、図1(a)(文献[7]、p.222、Fig.6より引用)及び(b)(文献[8]、p.4、Fig.2.1より引用)に示す。従来の他の手法による同じ問題に対する判定過程結果(たとえば文献[9]、p.413、Fig.11.9)においては、判定木の節点数が25で、しかも、等価式中には $M_1$ 、 $M_2$ のプッシュダウン記号(非終端記号)が混合(“mixing”)した記号列が出現している。これらを比較すると、新しく[7]、[8]において提唱した方式の単純さ、効率性が直観的に理解していただけるであろう。なお、これとは別のValiantらの流儀による手法は一層に複雑で、例題化も困難である。

決定性プッシュダウンオートマトンに対する上記判定法は非常に単純であるので、直ちにある種の**決定性文脈自由文法**同士の等価性判定法にも適用出来、[10]の結果を得ている。更に同手法は、出力機構を持った**決定性プッシュダウン変換機**に対しても統一的に拡張し、[11]として発表している。なお、そこで対象としている決定性プッシュダウンオートマトンに対する等価性判定の可解性の結果自体は既に[12]において発表していたが、そこでの手法、あるいは他で発表されている手法はいずれも非常に複雑であって、どの一つとして、極く単純な対象であっても具体的な例題でアルゴリズムの流れを示すことは不可能に近かったが、[11]においては、図1とほぼ同様の単純な具体例を掲載している。

決定性プッシュダウンオートマトンの等価性判定問題の可解範囲拡張に関して、更に大山口教授が[13]などの優れた結果を示している。筆者らもその後[14]などの結果を発表しているが、この過程において、清野和司氏(現・東芝ソリューション)の修士課程在学以来の貢献は大である。これらの努力にもかかわらず、当時この一般解は依然として非常に難問として残され、コンピュータ基礎理論ハンドブック[15]のChapter 2 *Context-Free Languages*においても、それに関する主な結果として、[13]、[11]、及びSénizerguesの[16]を挙げるに留める段階であった。

その後本問題に関しては、Sénizerguesが肯定的解決を全166ページの最終論文[17]として発表し、翌2002年にはそれに対して直ちに異例の早さでGödel賞が与えられたことは記憶に新しいことであろう。

ここで、筆者の研究室では、大学院博士課程の若月光夫氏(現・電通大)が対象を単純決定性プッシュダウンオートマトンに集中して更に詳細な理論を展開し、ある意味において計算量が多項式的である効率的等価性判定法を得[18]、更に同様の手法に基づいて、ある種の文法

に対する効率的な包含性判定アルゴリズム[19]などを得ている。また樋口健氏(現・福井大)は、ある種のカウンタにおける判定問題に対して、[20]などの新しい幾つかの結果を得ている。最近では、大学院社会人博士課程にも在籍した清野氏が決定性プッシュダウン変換器に対する強力/効率的な新しいアルゴリズムを得ている[21]、[22]。

また、但馬康宏氏(現・農工大)は博士論文の中心的成果として、単純決定性言語を通常のMAT学習に近い形式で効率的に達成できる[23]との強力な結果などを示した。ここにおいて、単純決定性文法に対する効率的な等価性判定アルゴリズム[18]が非常に巧妙に活かされている。なお、本論文は、4<sup>th</sup> of TOP25 Hottest Articles within the journal: Theoretical Computer Science in Oct.-Dec., 2004, and 14<sup>th</sup> in Apr.-June, 2005となっている。

更に、若月氏は正例のみから効率良く学習を行う幾つかの新しいアルゴリズムを開発している[24]、[25]、[26]。

この間、筆者は国際会議Algorithmic Learning Theory (ALT05)のProgram Committee Chairを務め[27]、その特集号[28]のGuest Editorも務めた。また、先進アルゴリズム研究ステーションが主体となり、次ぎの国際会議も電通大において2006年9月に開催した。

International Colloquium on Grammatical Inference (ICGI06)[29]、

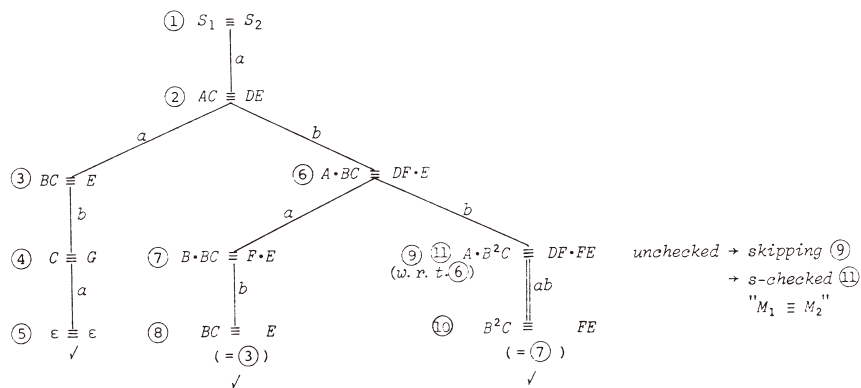
- ・ Conference Chair: E. Tomita,
- ・ Program Committee Chairs: Y. Sakakibara,  
S. Kobayashi,
- ・ Organizing Committee Chair: T. Nishino.

### 3. 教科書「オートマトン・言語理論」(森北出版)

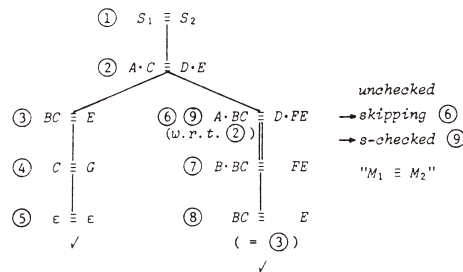
筆者は電気通信大学へ着任以来、当初は通信工学科という情報系でない電気系の学科に所属していたため、上記のオートマトン・言語理論関係の卒業論文や修士論文の研究に先立っては、情報の基礎知識が無く慣れていない学生でも、極力自ら学習し易い様にと沢山の学習資料を作成してその準備に当てていた。

そのような折に、飯島泰蔵教授(現・東工大名誉教授)より、森北出版・基礎情報工学シリーズの教科書執筆担当のお誘いをいただき、前記教材を基として、横森貴教授(現・早大、当時・電通大情報工学科助教授)にも加わっていただいてまとめ上げたのが本教科書[30]である。本書の作成から校正においては、清野和司氏、若月光夫氏、樋口健氏を始め多くの学生・卒業生諸君に、少しでもより読み易くなる様にとの協力をいただいた。このような事情から、本書は理論的内容であるにもかかわらず非常に学習し易いとの評価をいただき、多くの大学、大学院等において教科書、参考書として利用していただい

$\delta_1$	$\delta_2$
$S_1 \xrightarrow{a} AC$	$S_2 \xrightarrow{a} DE$
$A \xrightarrow{b} AB$	$D \xrightarrow{b} DF$
$A \xrightarrow{a} B$	$D \xrightarrow{a} \epsilon$
$B \xrightarrow{b} \epsilon$	$E \xrightarrow{b} G$
$C \xrightarrow{a} \epsilon$	$F \xrightarrow{b} \epsilon$
	$G \xrightarrow{a} \epsilon$



(a) [7] Fig.6



(b) [8] Fig.2.1

図1 等価性判定例

ている。このため、出版以来ほぼ毎年改訂を伴った増刷を重ね、2009年には第20刷を迎える予定となっている。この間知り得る限りでは、2000年11月に大学生協書籍ベスト20・工学部門の第7位にランクされ、Amazonにおける、“オートマトン”に関する和書についてのベストセラーでも、しばしばトップにランクされている。

**謝辞** これまで多大の協力・貢献をしてきていただいている当研究室若月光夫氏、清野和司氏、樋口健氏、但馬康宏氏、西野哲朗教授、小林聡教授、高橋治久教授はじめ研究室、先進アルゴリズム研究ステーションの方々へ深謝いたします。なお、これらの研究は科研、電通大

研究・教育活性化支援システム、船井情報科学財団、井上科学振興財団などによる支援を受けている。

**参考文献**

[01] 富田悦次: “電子情報通信学会フェロー受賞記念講演: オートマトン・言語理論・学習理論と組合せ最適化の研究及び教育,” 電子情報通信学会コンピュータシオン研究会, COMP2003-60, pp.45-52(2003).  
 [02] Etsuji Tomita: “Advanced algorithms and their applications,” Proc. Advanced ICT(AICT 2007), Beijing, China, pp.1 -6 (2007).  
 [1] H. Enomoto, E. Tomita, and S. Doshita: “Synthesis of automata that recognize given strings and characterization of automata by representative sets of strings,”

- Proc. First USA-Japan Computer Conf., Tokyo, Japan, pp.21-27(1972).
- [2] 榎本 肇, 富田悦次: “代表記号例集合による決定性有限オートマトンの適応的修正法,” 電子通信学会論文誌(D), vol.J60-D, no.10, pp.777-784(1977).
- [3] D. Angluin: “Learning regular sets from queries and counterexamples,” Information and Computation, vol.75, pp.87-106(1987).
- [4] L. G. Valiant: “Decision procedures for families of deterministic pushdown automata,” Ph. D. Dissertation, University of Warwick, Coventry(1973).
- [5] K. Taniguchi and T. Kasami: “A result on the equivalence problem for deterministic pushdown automata,” J. Computer and System Sciences, vol.13, pp.38-50(1976).
- [6] M. Oyamaguchi, N. Honda and Y. Inagaki: “The equivalence problem for real-time strict deterministic languages,” Information and Control, vol.45, pp.90-115(1980).
- [7] E. Tomita: “A direct branching algorithm for checking equivalence of some classes of deterministic pushdown automata,” Information and Control, vol.52, pp.187-238(1982).
- [8] E. Tomita and H. Tsuchiya: “Improvements on a direct branching algorithm for checking equivalence of DPDA's,” Technical Report of IECE, COMP86-9, pp. 1-10(1986).
- [9] M.A. Harrison: “Introduction to Formal Language Theory,” Addison-Wesley(1978).
- [10] E. Tomita: “A direct branching algorithm for checking equivalence of strict deterministic vs.  $LL(k)$  grammars,” Theoretical Computer Science, vol.23, pp.129-154(1983).
- [11] E. Tomita and K. Seino: “A direct branching algorithm for checking the equivalence of two deterministic pushdown transducers, one of which is real-time strict,” Theoretical Computer Science, vol.64, pp.39-53(1989).
- [12] 富田悦次: “一方が  $\varepsilon$ -動作なし空スタック受理式である決定性プッシュダウン変換器の等価性判定,” 電子通信学会論文誌(D), vol.J62-D, no. 7, pp.467-474(1979).
- [13] M. Oyamaguchi: “The equivalence problem for real-time DPDAs,” J.ACM, vol.34, pp.731-760(1987).
- [14] E. Tomita and K. Seino: “The extended equivalence problem for a class of non-real-time deterministic pushdown automata,” Acta Informatica, vol.32, pp.395-413(1995).
- [15] J. V. Leeuwen: Handbook of Theoretical Computer Science, vol.B, Formal Models and Semantics, MIT Press, Cambridge/Elsevier, Amsterdam, Mass(1990) (広瀬健・野崎昭弘・小林孝次郎 監訳: コンピュータ基礎理論ハンドブックII, 丸善(1994))
- [16] G. Sénizergues: “Some decision problems about controlled rewriting systems,” Theoretical Computer Science, vol.71, pp.281-346(1990).
- [17] G. Sénizergues: “ $L(A)=L(B)$ ? decidability results from complete formal system,” Theoretical Computer Science, vol.251, pp. 1-166(2001).
- [18] 若月光夫, 富田悦次: “単純決定性プッシュダウンオートマトンの等価性判定の改良分岐アルゴリズムとその最大時間計算量,” 電子情報通信学会論文誌(D-I), vol. J74-D-I, no.9, pp.595-603(1991).
- [19] M. Wakatsuki and E. Tomita: “A fast algorithm for checking the inclusion for very simple deterministic pushdown automata,” IEICE Trans. Information and Systems, vol.E76-D, no. 10, pp. 1224-1233(1993).
- [20] K. Higuchi, E. Tomita and M. Wakatsuki: “A polynomial-time algorithm for checking the inclusion for strict deterministic restricted one-counter automata,” IEICE Trans. Information and Systems, vol.E78-D, no.4, pp.305-313(1995).
- [21] 清野和司, 富田悦次, 若月光夫: “ $\varepsilon$ -推移を許したある決定性プッシュダウン変換器対の等価性判定,” 電子情報通信学会論文誌D, vol.J90-D, no.10, pp.2675-2690(2007).
- [22] 清野和司, 富田悦次, 若月光夫: “実時間空スタック受理式決定性限定ワンカウンター変換器対の多項式時間等価性判定,” 電子情報通信学会論文誌D, vol.J91-D, no.5, pp.1188-1201(2008).
- [23] Y. Tajima, E. Tomita, M. Wakatsuki and M. Terada: “Polynomial time learning of simple deterministic languages via queries and a representative sample,” Theoretical Computer Science, vol.329, p.203-221(2004).
- [24] M. Wakatsuki, E. Tomita and G. Yamada: “A unified algorithm for extending classes of languages identifiable in the limit from positive data,” International Colloquium on Grammatical Inference (ICGI 2006), Tokyo, Japan, Lecture Notes in Artificial Intelligence 4201, pp.161-174(2006).
- [25] M. Wakatsuki and E. Tomita, “Polynomial time identification of finite state transducers in some class,” Proc. Advanced ICT(AICT 2007), Beijing, China, pp. 7-12(2007).
- [26] M. Wakatsuki and E. Tomita: “Polynomial time identification of strict deterministic restricted one-counter automata in some class from positive data,” IEICE Trans. Information and Systems, vol.E91, no.6, pp.1704-1718(2008).
- [27] S. Jain, H. U. Simon and E. Tomita(Eds.): “Algorithmic Learning Theory,” 16th International Conference on Algorithmic Learning Theory, ALT 2005, Lecture Notes in Artificial Intelligence 3734(2005).
- [28] H. U. Simon and E. Tomita(Eds.): “Algorithmic Learning Theory,” Theoretical Computer Science, vol.387(2007).
- [29] Y. Sakakibara, S. Kobayashi, K. Sato, T. Nishino and E. Tomita(Eds.): “Grammatical Inference: Algorithms and Applications,” 8th International Colloquium on Grammatical Inference, ICGI 2006, Lecture Notes in Artificial Intelligence 4201(2006).
- [30] 富田悦次, 横森 貴: 「オートマトン・言語理論」, 森北出版(1992).

## Part II The Maximum Clique Problem and Its Applications [32]

### 1 The Maximum Clique Problem

Many problems can be formulated as graphs where a graph consists of a set of vertices and a set of edges, in which the vertices stand for objects in question and the edges stand for some relations among the objects. A *clique* is a subgraph in which all vertices are pairwise adjacent[6]. Hence, a clique represents a set of objects in which every pair is related. In addition, a maximum clique of the direct product of two graphs represents a maximum matching in the two graphs[6]. Therefore, a maximum clique and maximal cliques play an important role and have received considerable attention[11, 6].

However, the so called *maximum clique problem* is considered to be very hard to solve, that is, it is proved to be an *NP-complete* problem. Nevertheless, many researchers including the author are engaged in devising as fast algorithms as possible for finding a maximum clique and generating all maximal cliques, because of their importance in practice.

#### 1.1 Fast Algorithms for Finding a Maximum Clique

Recently, we presented a simple and fast branch-and-bound algorithm **MCQ**[28] for finding a maximum clique, and we improved it to get a new algorithm **MCR**[30], primarily by introducing more appropriate sorting of vertices at the beginning. MCR in turn was improved to a more efficient algorithm MCR-Re[31], by employing sophisticated approximate coloring and sorting of vertices. In addition, we have improved MCR-Re to have an algorithm **MCS**(previously called  $MCS_0$ ) by localizing the memory usage in order to

make more effective use of cache memory[22].

A comparison of MCQ, MCR, and MCS is shown in Table 1, where the branches correspond to the extent of search spaces[28, 30] and  $\omega$  is the size of a maximum clique in a given graph. Some computational time comparisons with other algorithms are shown in Tables 2 and 3. In Table 2,  $n$  is the number of vertices and  $p$  the edge probability. The computer used in these experiments has a Pentium4 3.6GHz CPU operating on Linux [31]. It is confirmed that MCS is by far the fastest among all the presently existing algorithms for almost all cases.

We have also developed algorithms for weighted graphs[27, 24].

#### 1.2 Algorithms for Generating Maximal Cliques

In addition to finding a maximum clique, generating all maximal cliques is required in many diverse applications such as clustering, data mining and others[6, 13]. We proposed an algorithm **CLIQUES**[29] for generating all maximal cliques.

A part of its computational time comparisons with other algorithms is shown in Table 4, where #cliques is the number of maximal cliques. The computer used in this experiment has a Pentium4 2.2GHz CPU operating on Linux[29]. CLIQUES is very fast and space efficient.

Some variations of CLIQUES have also been developed[14, 15].

#### 1.3 Theoretical Analyses

We proved that the worst-case time complexity of CLIQUES is  $O(3^{n/3}) = O(2^{0.528n})$  for an  $n$ -vertex graph,

Table 1 : Comparison of MCQ, MCR, and MCS

Graph		CPU time[sec]			branches $\times 10^{-3}$		
Name	$\omega$	MCQ	MCR	MCS	MCQ	MCR	MCS
brock400_2	29	748	742	297	116,224	116,328	33,513
brock400_4	33	680	639	248	118,855	114,925	30,855
MANN_a27	126	2.61	2.54	0.78	38	38	9
MANN_a45	345	2,775	3,090	281	2,852	2,952	225
p_hat300-3	36	17	11	3	2,473	1,546	235
p_hat500-3	50	2,895	1,788	150	237,077	138,300	7,923
p_hat700-3	62	122,264	68,187	2,392	7,046,183	3,733,665	88,168
p_hat1000-2	46	2,764	2,434	221	221,797	197,147	12,618
san200_0.9_3	44	10.59	0.16	0.06	1,182	22	6
san400_0.9_1	100	32.8	3.4	0.1	708	74	2
sanr200_0.9	42	322	289	41	42,865	40,470	3,471

and that is *optimal* with respect to  $n$ .

Steady improvements have been made to the time complexity for finding a maximum clique in an  $n$ -vertex graph in polynomial-space from  $O(2^{0.333n})$ [26] to  $O(2^{0.288n})$  [9] in the last almost 30 years. We have remarkably improved this complexity to  $O(2^{0.19669n})$ [16] by an algorithm that is based on CLIQUES[29, 20]. Our algorithm is also fast in practice[20, 25]. Further theoretical analysis is in progress.

## 2 Applications

The above algorithms and their extensions are being successfully applied to many problems. These include the followings:

- Clustering[35],
- Bioinformatics[2], [3], [4], [7], [1],
- Image processing[10],

- Design of quantum circuits[17],
- Design of DNA and RNA sequences for biomolecular computation[12], [23].

Parallel processing is also under study[33], [34] in order to solve large practical problems.

## Acknowledgment

The author would like to express his gratitude to his students and colleagues, Yoichi Sutani, Takanori Higasi, Shinya Takahashi, Hiroaki Nakanishi, Mitsuo Wakatsuki, Haruhisa Takahashi, Tetsuro Nishino, Satoshi Kobayashi, and others, for their collaborative studies.

This research was partially supported by Grants-in-Aid for Scientific Research Nos. 13680435, 16300001, 19500010, and others from the MEXT, Japan.

Table 2 : CPU time [sec] for random graphs

Graph			dfmax [11]	MCS (Ours)	New [18]	COCR [19]
$n$	$p$	$\omega$				
100	0.6	11-13	0.0041	<b>0.0016</b>	0.0022	0.092
	0.7	14-16	0.018	<b>0.0036</b>	0.0067	0.12
	0.8	19-21	0.14	● <b>0.0078</b>	0.065	0.15
	0.9	29-32	3.67	★ <b>0.013</b>	0.66	0.20
	0.95	39-48	23.74	★ <b>0.0028</b>	0.20	
	0.98	56-68	26.54	★★★ <b>0.00087</b>		
150	0.7	16-18	0.36	● <b>0.047</b>		0.33
	0.8	23	6.88	○ <b>0.23</b>		0.75
	0.9	36-39	1,058.96	<b>1.01</b>		1.16
	0.95	50-59	37,436.79	★★★ <b>0.35</b>		
	0.98	73-85	$>10^5$	★★★ <b>0.0061</b>		
200	0.5	11-12	0.038	<b>0.015</b>	0.020	0.25
	0.6	14	0.29	○ <b>0.072</b>	0.17	0.52
	0.7	18-19	3.85	○ <b>0.41</b>	3.02	1.65
	0.8	24-27	192.68	<b>4.48</b>	147.29	8.69
	0.9	40-44	$>10^5$	73.62		○ <b>36.79</b>
	0.95	58-66	$>10^5$	★★★ <b>58.83</b>		
	0.98	90-103	$>10^5$	★★★ <b>0.21</b>		
300	0.5	12-13	0.36	<b>0.13</b>	0.20	1.13
	0.6	15-16	4.88	○ <b>0.99</b>	3.50	4.98
	0.7	19-21	144.11	★ <b>12.00</b>	121.02	
	0.8	28-29	26,235.96	★ <b>393.57</b>		
	0.9	49	$>10^5$	<b>79,628.80</b>		
500	0.5	13-14	8.99	○ <b>2.79</b>	7.25	17.43
	0.6	17	242.29	○ <b>40.70</b>	183.28	
	0.7	22-23	24,998.42	★ <b>1,538.74</b>		
	0.75	26	$>10^5$	○ <b>20,403.68</b>		
1,000	0.3	9-10	1.98	<b>1.15</b>	1.64	
	0.4	12	33.28	<b>13.25</b>	23.19	
	0.5	15	1,107.70	○ <b>290.03</b>		
	0.6	20	$>10^5$	● <b>13,554.05</b>		
5,000	0.1	7	6.29	<b>3.32</b>		
10,000	0.1	7-8	137.05	○ <b>59.55</b>		
15,000	0.1	8	792.57	○ <b>326.78</b>		

Entries indicated by ★★★, ★, ●, and ○ represent those that are more than or equal to 1000, 10, 5, and 2 times faster than all the others confirmed within the time limits in the same row, respectively.

Table 3 : CPU time [sec] for DIMACS benchmark graphs

Graph		dfmax [11]	MCS (Ours)	New [18]	$\chi$ +DF [8]	COCR [19]	MIPO [5]	Target/5 [21]
Name	$\omega$							
brock200_1	21	14.53	★ <b>0.86</b>	12.12	68.70			69.80
brock200_4	17	0.90	★ <b>0.14</b>	0.22	6.04	0.91		3.60
brock400_1	27	22,051	★ <b>693</b>		>10,640			>4,320
brock400_2	29	13,519	★ <b>297</b>		>10,640	>415		>4,320
brock400_3	31	14,795	★ <b>468</b>		>10,640			>4,320
brock400_4	33	10,633	★ <b>248</b>		>10,640	>415		>4,320
brock800_1	23	>10 <sup>5</sup>	★ <b>9,347</b>		>10,640			
brock800_2	24	>10 <sup>5</sup>	★ <b>8,368</b>		>10,640	>415		
brock800_3	25	91,031	★ <b>5,755</b>		>10,640			
brock800_4	26	78,737	★ <b>3,997</b>		>10,640	>415		
c-fat500-10	126	>10 <sup>5</sup>	○ <b>0.026</b>	0.016	<b>0.015</b>			
hamming8-4	16	1.85	○ <b>0.20</b>	<b>0.19</b>	4.51	1.00	29.13	
hamming10-2	512	>10 <sup>5</sup>	○ <b>0.19</b>	0.56	3.81			
johnson16-2-4	8	0.75	○ <b>0.14</b>	0.060	5.88		<b>★0.0017</b>	
MANN_a27	126	>10 <sup>5</sup>	○ <b>0.78</b>	>2,232	7,647	2.75		
MANN_a45	345	>10 <sup>5</sup>	★★ <b>281</b>		>10,640			
p_hat300-2	25	0.63	★ <b>0.018</b>	0.22	2.23	0.61		0.20
p_hat300-3	36	780	○ <b>2.55</b>		633	5.39		
p_hat500-1	9	0.051	<b>0.030</b>	0.065	0.44			
p_hat500-2	36	133	★★ <b>0.74</b>	95.71	151			>4,320
p_hat500-3	50	>10 <sup>5</sup>	★★ <b>150</b>		>10,640			>4,320
p_hat700-1	11	0.20	<b>0.10</b>	0.15	1.98	2.74		1.40
p_hat700-2	44	5,300	○ <b>5.60</b>		1,542	25.44		>4,320
p_hat700-3	62	>10 <sup>5</sup>	★ <b>2,392</b>		>10,640	>415		>4,320
p_hat1000-1	10	1.05	○ <b>0.49</b>	1.30	12.14			
p_hat1000-2	46	>10 <sup>5</sup>	★★ <b>221</b>		>10,640			
san200_0.9_1	70	>10 <sup>5</sup>	○ <b>0.22</b>	0.060	46.27		<b>0.050</b>	4.60
san200_0.9_2	60	>10 <sup>5</sup>	○ <b>0.41</b>	0.96	1,427		○ <b>0.15</b>	65.60
san200_0.9_3	44	42,643	★★ <b>0.063</b>		144		15.15	>4,320
san400_0.5_1	13	433	○ <b>0.020</b>	○ <b>0.0067</b>	4.98		85.44	0.80
san400_0.7_1	40	>10 <sup>5</sup>	★ <b>0.54</b>	>2,232	315			24.40
san400_0.7_2	30	>10 <sup>5</sup>	★★ <b>0.13</b>	113	118		505	113.20
san400_0.7_3	22	>10 <sup>5</sup>	★★ <b>1.44</b>		456			>4,320
san400_0.9_1	100	>10 <sup>5</sup>	★★★ <b>0.12</b>		5,335			>4,320
san1000	15	>10 <sup>5</sup>	<b>2.17</b>	<b>★0.11</b>	2,249			
sanr200_0.9	42	86,954	★★★ <b>41</b>		>10,640			
sanr400_0.5	13	2.12	○ <b>0.72</b>	1.48	17.06			
gen200_p0.9_44	44	48,262	○ <b>0.47</b>			1.88	13.01	
gen200_p0.9_55	55	9,281	○ <b>1.23</b>			0.96	<b>● 0.19</b>	
gen400_p0.9_55	55	>10 <sup>6</sup>	★ <b>58,502</b>					
C125.9	34	50.05	● <b>0.060</b>			0.56	46.6	
C250.9	44	>10 <sup>6</sup>	★★ <b>3,257</b>					

Entries indicated by ★★★, ★★, ★, ●, and ○ represent those that are more than or equal to 1000, 100, 10, 5, and 2 times faster than all the others confirmed within the time limits in the same row, respectively.

Table 4 : CPU time [sec] for random graphs

Graph		CLIQUES [29]	AMC [13]	AMC* [13]
Name	#cliques			
r1000.1	118,325	0.2	143.1	19.4
r1000.2	1,183,584	2	4,486	830
r3000.1	2,945,211	11	>86,400	5,905
r5000.1	18,483,855	87	>86,400	>86,400

## References

- [1] T. Akutsu, M. Hayashida, D. K. C. Bahadur, E. Tomita, J. Suzuki, K. Horimoto. Dynamic programming and clique based approaches for protein threading with profiles and constraints. *IEICE Trans.*, E89-A: 1215-1222, 2006.
- [2] D. K. C. Bahadur, T. Akutsu, E. Tomita, T. Seki, A. Fujiyama. Point matching under non-uniform distortions and protein side chain packing based on an efficient maximum clique algorithm. *Genome Inform.*, 13: 143-152, 2002.
- [3] D. K. C. Bahadur, E. Tomita, J. Suzuki, K. Horimoto, T. Akutsu. Protein side-chain packing problem: A maximum edge-weight clique algorithmic approach. *J. Bioinform. Comput. Biology*, 3: 103-126, 2005.

- [4] D. K. C. Bahadur, E. Tomita, J. Suzuki, K. Horimoto, T. Akutsu. Protein threading with profiles and distance constraints using clique based algorithms. *J. Bioinform. Comput. Biology*, 4: 19-42, 2006.
- [5] E. Balas, S. Ceria, G. Cornuéjols, G. Pataki. Polyhedral methods for the maximum clique problem. In [11] : 11-28, 1996.
- [6] I. M. Bomze, M. Budinich, P. M. Pardalos, M. Pelillo. The maximum clique problem.. In: D. -Z Du, P. M. Pardalos(Eds.). *Handbook of Comb. Optim.*, Suppl. vol. A, Kluwer Acad. Publ.: 1-74, 1999.
- [7] J. B. Brown, D. K. C. Bahadur, E. Tomita, T. Akutsu. Multiple methods for protein side chain packing using maximum weight cliques. *Genome Inform.*, 17: 3-12, 2006.
- [8] T. Fahle. Simple and fast: Improving a branch-and-bound algorithm for maximum clique. *ESA 2002, LNCS 2461*: 485-498, 2002.
- [9] F. V. Fomin, F. Grandoni, D. Kratsch. Measure and conquer: A simple  $O(2^{0.288n})$  independent set algorithm. *Proc. SODA 2006*: 18-25, 2006.
- [10] K. Hotta, E. Tomita, H. Takahashi. A view-invariant human face detection method based on maximum cliques. *Trans. IPSJ*, 44, SIG14(TOM9): 57-70, 2003.
- [11] D. S. Johnson, M. A. Trick (Eds.). Cliques, Coloring, and Satisfiability, *DIMACS Series in DMTCS*, vol.26, Amer. Math. Soc.: 1996.
- [12] S. Kobayashi, T. Kondo, K. Okuda, E. Tomita. Extracting globally structure free sequences by local structure freeness. *Proc. DNA 9*: 206, 2003.
- [13] K. Makino, T. Uno. New algorithms for enumerating all maximal cliques. *SWAT 2004, LNCS 3111*: 260-272, 2004.
- [14] T. Nakagawa, E. Tomita. An efficient algorithm for generating large maximal cliques. *Tech. Rep. IPSJ SIG*, 2006MPS-57: 49-52, 2005.
- [15] T. Nakagawa, E. Tomita. An algorithm for generating all maximal bipartite cliques based on CLIQUES that generates all maximal cliques. *Tech. Rep. IPSJ SIG*, 2006-MPS-62: 73-76, 2006.
- [16] H. Nakanishi, E. Tomita. An  $O(2^{0.19669n})$ -time and polynomial-space algorithm for finding a maximum clique *Tech. Rep. IPSJ SIG*, 2007-AL-115: 17-24, 2007.
- [17] Y. Nakui, T. Nishino, E. Tomita, T. Nakamura. On the minimization of the quantum circuit depth based on a maximum clique with maximum vertex weight. *Tech. Rep. RIMS*, 1325, Kyoto Univ.: 45-50, 2003.
- [18] P. R. J. Östergård. A fast algorithm for the maximum clique problem. *Disc. Appl. Math.*, 120: 197-207, 2002.
- [19] E. C. Sewell. A branch and bound algorithm for the stability number of a sparse graph. *INFORMS J. Comput.*, 10: 438-447, 1998.
- [20] M. Shindo, E. Tomita. A simple algorithm for finding a maximum clique and its worst-case time complexity. *Syst. Comput. in Japan*, 21: 1-13, 1990.
- [21] V. Stix. Target-oriented branch and bound method for global optimization. *J. Global Optim.*, 26: 261-277, 2003.
- [22] Y. Sutani, T. Higashi, E. Tomita, S. Takahashi. H. Nakatani. A faster branch-and-bound algorithm for finding a maximum clique. *Tech. Rep. IPSJ SIG*, 2006-AL-108: 79-86, 2006.
- [23] Y. Sutani, E. Tomita, S. Kobayashi. A branch-and-bound algorithm for finding a maximum clique in a uniform hypergraph. *Tech. Rep. IPSJ SIG*, 2007-MPS-66: 111-114, 2007.
- [24] J. Suzuki, E. Tomita, T. Seki. An algorithm for finding a maximum clique with maximum edge-weight and computational experiments. *Tech. Rep. IPSJ SIG*, 2002-MPS-42: 45-48, 2002.
- [25] T. Tamada, E. Tomita, H. Nakanishi. Experimental evaluations of algorithms with known theoretical time-complexity for finding a maximum clique. *Tech. Rep. IPSJ SIG*, 2007-MPS-67/2007-BIO-11: 25-28, 2007.
- [26] R. E. Tarjan, A.E. Trojanowski. Finding a maximum independent set. *SIAM J. Comput.*, 6: 537-546, 1977.
- [27] E. Tomita, Y. Wakai, K. Imamatsu. An efficient algorithm for finding a maximum weight clique and its experimental evaluations. *Tech. Rep. IPSJ SIG*, 1999-MPS-26: 33-36, 1999.
- [28] E. Tomita, T. Seki. An efficient branch-and-bound algorithm for finding a maximum clique. *DMTCS 2003, LNCS 2731*: 278-289, 2003.
- [29] 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: 28-42, 2006.
- [30] E. Tomita, T. Kameda. An efficient branch-and-bound algorithm for finding a maximum clique with computational experiments. *J. Global Optim.*, 37: 95-111, 2007.
- [31] E. Tomita, Y. Sutani, T. Higashi. A more efficient algorithm for finding a maximum clique with an improved approximate coloring. *Proc. PDPTA 2007*: 719-725, 2007.
- [32] E. Tomita . The maximum clique problem and its applications -Invited Lecture-, IPSJ SIG Technical Report, 2007-MPS-67/2007-BIO-11: 21-24, 2007.
- [33] S. Urabe, E. Tomita. NetMCQ: A distributed exact maximum clique solver. *Tech. Rep. IPSJ SIG*, 2007-MPS-67/2007-BIO-11: 29-32, 2007.
- [34] M. Wakatsuki, S. Takahashi, E. Tomita. Parallelization of an algorithm for finding a maximum clique *Tech. Rep. IPSJ SIG*, 2008-MPS-71: 17-21, 2008.
- [35] C. Yonemori, T. Matsunaga, E. Tomita. An analysis of enterprise communities by cliques. *Tech. Rep. IPSJ SIG*, 2007-MPS-67/2007-BIO-11: 33-36, 2007.