IEICE TRANS. INF. & SYST., VOL.E100–D, NO.3 MARCH 2017

PAPER Special Section on Foundations of Computer Science —New Trends in Theoretical Computer Science—

An Exact Algorithm for Lowest Edge Dominating Set

Ken IWAIDE[†a)], ***Nonmember and*** **Hiroshi NAGAMOCHI**[†b)], ***Member***

SUMMARY Given an undirected graph G, an edge dominating set is a subset F of edges such that each edge not in F is adjacent to some edge in F, and computing the minimum size of an edge dominating set is known to be NP-hard. Since the size of any edge dominating set is at least half of the maximum size $\mu(G)$ of a matching in G, we study the problem of testing whether a given graph G has an edge dominating set of size $\lceil \mu(G)/2 \rceil$ or not. In this paper, we prove that the problem is NP-complete, whereas we design an $O^*(2.0801^{\mu(G)/2})$-time and polynomial-space algorithm to the problem.

key words: *graph theory, edge dominating set, algorithm, NP-completeness, fixed parameter tractable*

1. Introduction

In an undirected graph $G = (V, E)$ with a set V of n vertices and a set E of m edges, an *independent set* (resp., a *matching*) is a subset of V (resp., E) that contains no two adjacent vertices (resp., edges). A *vertex cover* is defined to be the complement of an independent set over V, and an *edge dominating set* is a subset F of E whose end-points form a vertex cover, or every edge in $E \setminus F$ is adjacent to an edge in F. These four notions are among the most fundamental features of graph structures, and the optimization problems of finding a minimum vertex cover and a minimum edge dominating set are highlighted by Garey and Johnson [5] in their work on NP-completeness. It is important to investigate not only the standard min-max formulas among them but also the computational complexity to know when formulas tightly hold. It is known that the maximum size $\mu(G)$ of a matching of G can be found in $O(\sqrt{n}m)$ time [11], whereas finding the minimum size $\tau(G)$ of a vertex cover of G is NP-hard. Note that $\mu(G)$ is a lower bound on $\tau(G)$. Gavril [6] showed whether G has a vertex cover of size $\mu(G)$ or not can be decided in $O(n + m)$ time. In this paper, we study the complexity to know whether the size of an edge dominating set in G is the lowest with respect to the matching size $\mu(G)$. We first review the results on algorithms for edge dominating sets.

The Minimum Edge Dominating Set problem requests us to find a minimum edge dominating set of a given graph. Yanakakis and Gavril [16] indicated that the problem is NP-hard even for planar or bipartite graphs of maximum degree 3 and they also showed that the size of a minimum edge dominating set can be efficiently approximated within a factor of 2. Fujito and Nagamochi [4] showed that the size of a *minimum weight* edge dominating set can be also approximated to within a factor of 2. We use O^* notation to suppress all polynomially bounded factors. For Minimum Edge Dominating Set, Randerath and Schiermeyer [9] designed an $O^*(1.4423^m)$-time and polynomial-space algorithm, and Raman et al. [8] improved this to $O^*(1.4423^n)$. Using the treewidth of graphs, Fomin et al. [3] obtained an $O^*(1.4082^n)$-time and exponential-space algorithm. Analyzing with the measure and conquer method, van Rooij and Bodlaender [10] designed an $O^*(1.3226^n)$-time and polynomial-space algorithm and later Xiao and Nagamochi [14] presented an $O^*(1.3160^n)$-time and polynomial-space algorithm, which currently attains the best time bound to Minimum Edge Dominating Set. For graphs of maximum degree 3, an $O^*(1.2721^n)$-time and polynomial-space algorithm is designed by Xiao and Nagamochi [15].

The Parameterized Edge Dominating Set problem is given a graph $G = (V, E)$ with an integer k to decide whether or not G has an edge dominating set of size at most k, which is known to be FPT. For the problem, Fernau [2] presented an $O^*(2.6181^k)$-time and polynomial-space algorithm. Using the bounded treewidth of the graph, Fomin et al. [3] gave an $O^*(2.4181^k)$-time and exponential-space algorithm. Analyzing with the measure and conquer method, Binkele-Raible and Fernau [1] designed an $O^*(2.3819^k)$-time and polynomial-space algorithm, and Xiao et al. [12] gave an $O^*(2.3147^k)$-time and polynomial-space algorithm. Recently, Iwaide and Nagamochi [7] presented an $O^*(2.2351^k)$-time and polynomial-space algorithm, which currently attains the best time bound to Parameterized Edge Dominating Set. For graphs of maximum degree 3, an $O^*(2.1479^k)$-time and polynomial-space algorithm is designed by Xiao and Nagamochi [13].

We observe the size of edge dominating sets of a graph G is bounded from below by $\lceil \tau(G)/2 \rceil \geq \lceil \mu(G)/2 \rceil$, since the set of endpoints of all edges in any edge dominating set is a vertex cover. As in the relationship between the minimum vertex cover and the maximum matching, we are interested in the issue of whether an edge dominating set with the lowest size in terms of $\mu(G)$ if one exists can be found in polynomial time or faster than the current best algorithms

Manuscript received March 22, 2016.
Manuscript revised July 19, 2016.
Manuscript publicized December 21, 2016.
[†]The authors are with Department of Applied Mathematics and Physics, Graduate School of Informatics, Kyoto University, Kyoto-shi, 606–8501 Japan.
a) E-mail: iwaide@amp.i.kyoto-u.ac.jp
b) E-mail: nag@amp.i.kyoto-u.ac.jp
DOI: 10.1587/transinf.2016FCP0005

for Minimum Edge Dominating Set and Parameterized Edge Dominating Set. The problem we study in this paper is described as follows.

Lowest Edge Dominating Set
Instance: An undirected graph G.
Question: Does G have an edge dominating set of size $\lceil \mu(G)/2 \rceil$?

In this paper, we first prove that Lowest Edge Dominating Set is NP-complete and then design an $O^*(2.0801^{\mu(G)/2})$-time and polynomial-space algorithm to Lowest Edge Dominating Set. The algorithm runs faster than the one with run time $O^*(2.2351^k)$ to Parameterized Edge Dominating Set where we can assume that $k \geq \mu(G)/2$, and the one with run time $O^*(1.3160^n) = O^*(2.9993^{n/4})$ time to Minimum Edge Dominating Set, where $n/2 \geq \mu(G)$ always holds.

The paper is organized as follows. Section 2 introduces basic notations on graphs and a property of tight edge dominating sets. Section 3 proves that Lowest Edge Dominating Set is NP-complete. Section 4 presents our exact algorithm for Lowest Edge Dominating Set by designing reduction and branching operations and analyzes the time bound. Section 5 makes some concluding remarks.

2. Preliminaries

Let G stand for a simple undirected graph in this paper. The sets of vertices and edges in G are denoted by $V(G)$ and $E(G)$, respectively. For a vertex $v \in V(G)$, denote by $E_G(v)$ the set of edges incident to vertex v, and by $d_G(v)$ the degree of vertex v, where $d_G(v) = |E_G(v)|$. For a vertex subset X, let $N_G(X)$ denote the set of neighbors of X, vertices in $V(G) \setminus X$ adjacent to some vertex in X. When $X = \{v\}$ for a vertex v, we may denote $N_G(X)$ by $N_G(v)$. For a set $S \subseteq V(G)$ of vertices, we let $G[S]$ denote the subgraph of G induced by S and let $G - S$ denote the graph $G[V(G) \setminus S]$. For a set $F \subseteq E(G)$, let $V(F)$ denote the set of vertices incident to at least one edge in F, and let $G[F]$ denote the subgraph $(V(F), F)$ of G.

We let M_G denote the union of all maximum matchings of G; i.e., $M_G = \{e \in E(G) \mid \mu(G) - \mu(G - V(\{e\})) = 1\}$. We let R_G denote the union of $V(G) \setminus V(M)$ over all maximum matchings M of G; i.e., $R_G = \{v \in V(G) \mid \mu(G) = \mu(G - \{v\})\}$. Note that M_G and R_G can be obtained in polynomial time.

We say that a subset $F \subseteq E(G)$ *dominates* (resp., 1-*dominates* and 2-*dominates*) an edge uv if $|\{u, v\} \cap V(F)| \geq 1$ (resp., $|\{u, v\} \cap V(F)| = 1$ and 2), where possibly $uv \in F$ when $|\{u, v\} \cap V(F)| = 2$. Then an edge subset F is an edge dominating set of G if and only if F dominates all edges in G. We call an edge dominating set F *tight* if $|F| = \lceil \mu(G)/2 \rceil$. Given two disjoint subsets $C, D \subseteq V(G)$, an edge dominating set F of G is called a (C, D)-*eds* of G if $C \subseteq V(F) \subseteq V(G) \setminus D$.

The next lemma states some structural property of tight edge dominating sets when the maximum matching size is even.

Lemma 1: Let G be a graph such that the maximum matching size $\mu(G)$ is even, and assume that G admits a tight edge dominating set F. Then every edge in F 1-dominates exactly two edges in any maximum matching of G, and F is a matching of G with $F \cap M_G = V(F) \cap R_G = \emptyset$ and 1-dominates each edge in M_G.

Proof. Let M be an arbitrary maximum matching of G. Note that every edge in F dominates at most two edges in M. Since $\mu(G)$ is even, it holds $|F| = \mu(G)/2 = |M|/2$. Then F can dominate all edges in M only when every edge in F 1-dominates exactly two edges in M. Hence F is a matching of G, $F \cap M = \emptyset$, and $V(F) \subseteq V(M)$. For any other maximum matching M' in G, F 1-dominates each edge in M' and satisfies $V(F) \cap (V(G) \setminus V(M')) = \emptyset$. This implies that F 1-dominates every edge in M_G and $F \cap M_G = V(F) \cap R_G = \emptyset$. □

3. NP-Completeness

This section proves the NP-completess of Lowest Edge Dominating Set in the following statement.

Theorem 2: Lowest Edge Dominating Set is NP-complete even if a given graph is bipartite and admits a perfect matching with even size.

Clearly, Lowest Edge Dominating Set is in the class NP. Thereby we establish the NP-hardness by a polynomial-time reduction from the NP-hard problem One-In-Three 3SAT [5].

One-In-Three 3SAT
Instance: A pair $(\mathcal{X}, C)$ of a set $\mathcal{X}$ of n variables $x_1, x_2, \ldots, x_n$ and a set C of m clauses $c_1, c_2, \ldots, c_m$ on $\mathcal{X}$ such that each clause c_j consists of exactly three literals ℓ_j^1, ℓ_j^2 and ℓ_j^3.
Question: Is there a truth assignment $\mathcal{X} \to \{\texttt{true}, \texttt{false}\}$ such that each clause c_j has exactly one true literal?

Given an instance $I = (\mathcal{X}, C)$ of One-In-Three 3SAT, we will construct a bipartite graph G_I that consists of
- n graphs, called *variable gadgets* $G_1^{\mathrm{V}}, G_2^{\mathrm{V}}, \ldots, G_n^{\mathrm{V}}$;
- m graphs, called *clause gadgets* $G_1^{\mathrm{C}}, G_2^{\mathrm{C}}, \ldots, G_m^{\mathrm{C}}$; and
- sets $E_{i,j}$ of edges between G_i^{V} and G_j^{C}, $1 \leq i \leq n$ and $1 \leq j \leq m$.

• For each variable $x_i \in \mathcal{X}$, define G_i^{V} to be a bipartite graph with a set $\{x_i^1, x_i^2, \ldots, x_i^8\}$ of eight labeled vertices and a set $M_i^{\mathrm{V}} \cup T_i^{\mathrm{V}} \cup F_i^{\mathrm{V}}$ of eight edges such that

$$M_i^{\mathrm{V}} = \left\{x_i^p x_i^{p+1} \mid p = 1, 3, 5, 7\right\},$$
$$T_i^{\mathrm{V}} = \left\{x_i^2 x_i^3, x_i^6 x_i^7\right\} \quad \text{and} \quad F_i^{\mathrm{V}} = \left\{x_i^2 x_i^5, x_i^4 x_i^7\right\},$$

as illustrated in Fig. 1.

• For each clause $c_j \in C$, define G_j^{C} to be a bipartite graph with a set $\{c_j^1, c_j^2, \ldots, c_j^{16}\}$ of 16 labeled vertices and a set $M_j^{\mathrm{C}} \cup \bigcup_{k=1,2,3} \left(T_{j,k}^{\mathrm{C}} \cup F_{j,k}^{\mathrm{C}}\right)$ of 17 edges such that

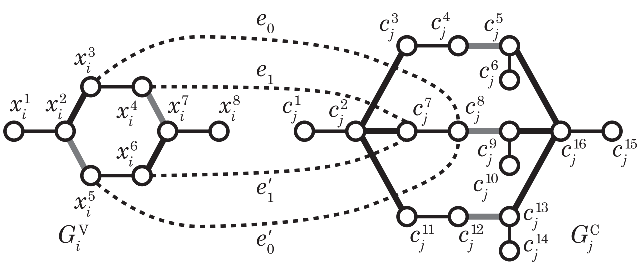

Fig. 1 A variable gadget G_i^{V}, a clause gadget G_j^{C}, the two edges $e_0, e_1 \in E_{i,j}^2$ with $\ell_j^2 = x_i$ and the two edges $e'_0, e'_1 \in E_{i,j}^2$ with $\ell_j^2 = \neg x_i$, where edges in T_i^{V} or $T_{j,k}^{\mathrm{C}}$ in are depicted by thick black lines and edges in F_i^{V} or $F_{j,k}^{\mathrm{C}}$ are depicted by thick gray lines.

$$\begin{aligned} M_j^{\mathrm{C}} &= \left\{ c_j^p c_j^{p+1} \mid p = 1,3,5,7,9,11,13,15 \right\}, \\ T_{j,k}^{\mathrm{C}} &= \left\{ c_j^2 c_j^{4k-1}, c_j^{4k+1} c_j^{16} \right\} \quad \text{and} \\ F_{j,k}^{\mathrm{C}} &= \left\{ c_j^{4k} c_j^{4k+1} \right\} \text{for } k = 1,2,3, \end{aligned}$$

as illustrated in Fig. 1.

• For each tuple (i, j, k) with $1 \le i \le n$, $1 \le j \le m$ and $1 \le k \le 3$, define a set $E_{i,j}^k$ of edges between G_i^{V} and G_j^{C} to be

$$E_{i,j}^k := \begin{cases} \{x_i^3 c_j^{4k}, x_i^4 c_j^{4k-1}\} & \text{if } \ell_j^k = x_i; \\ \{x_i^5 c_j^{4k}, x_i^6 c_j^{4k-1}\} & \text{if } \ell_j^k = \neg x_i; \\ \emptyset & \text{otherwise,} \end{cases}$$

as illustrated in Fig. 1. We set $E_{i,j} := \bigcup_{k=1,2,3} E_{i,j}^k$.

Let G_I be an instance in Lowest Edge Dominating Set constructed from a disjoint union of bipartite graphs G_i^{V}, $i = 1, \ldots, n$ and G_j^{C}, $j = 1, \ldots, m$ by adding the edge sets $E_{i,j}$ with $1 \le i \le n$ and $1 \le j \le m$. Obviously G_I can be constructed in polynomial time.

We see that G_I is a bipartite graph, since the indices k and ℓ of any type of edges $x_i^k x_i^\ell$, $c_j^k c_j^\ell$ and $x_i^k c_j^\ell$ in G_I have different parities. Each variable gadget G_i^{V} has a perfect matching M_i^{V} of size 4, and each clause gadget G_j^{C} has a perfect matching M_j^{C} of size 8. Therefore these matchings form a perfect matching of G_I with even size $\mu(G_I) = 4n + 8m$.

The remaining task is to prove the correctness of the reduction.

Lemma 3: Instance $I = (\mathcal{X}, \mathcal{C})$ is satisfiable if and only if G_I has an edge dominating set L of size $\mu(G_I)/2$.

Proof. (I) Only if part: Given a satisfiable truth assignment $\alpha : \mathcal{X} \to \{\texttt{true}, \texttt{false}\}$ to $I = (\mathcal{X}, \mathcal{C})$, we construct an edge dominating set

$$L = \bigcup_{1 \le i \le n} L_i^{\mathrm{V}} \cup \bigcup_{1 \le j \le m} L_j^{\mathrm{C}}$$

by choosing an edge set L_i^{V} from each G_i^{V} and an edge set L_j^{C} from each G_j^{C}.

From each variable gadget G_i^{V}, choose a set of two edges:

$$L_i^{\mathrm{V}} := \begin{cases} T_i^{\mathrm{V}} & \text{if } \alpha(x_i) = \texttt{true}; \\ F_i^{\mathrm{V}} & \text{if } \alpha(x_i) = \texttt{false}. \end{cases}$$

For each clause gadget G_j^{C}, let $h \in \{1,2,3\}$ be the unique index such that literal ℓ_j^h in clause c_j satisfies $\alpha(\ell_j^h) = \texttt{true}$, let $\{t, t'\} = \{1,2,3\} \setminus \{h\}$ be the remaining indices, and choose a set of four edges:

$$L_j^{\mathrm{C}} := T_{j,h}^{\mathrm{C}} \cup F_{j,t}^{\mathrm{C}} \cup F_{j,t'}^{\mathrm{C}}.$$

Clearly $|L| = 2n + 4m = \mu(G_I)/2$. We prove that L is an edge dominating set in G_I. For each variable x_i, graph $G_i^{\mathrm{V}} - V(L_i^{\mathrm{V}})$ has no edge, and for each clause c_j, graph $G_j^{\mathrm{C}} - V(L_j^{\mathrm{C}})$ has no edge. Therefore, to prove that L is an edge dominating set of G, it suffices to show that each edge in $E_{i,j}^k \neq \emptyset$ with $k = 1,2,3$, $i = 1, \ldots, n$ and $j = 1, \ldots, m$ is incident to a vertex in $V(L)$.

Without loss of generality consider the case of $k = 2$, as shown in Fig. 1, where $E_{i,j}^2 = \{e_0 = x_i^3 c_j^8, e_1 = x_i^4 c_j^7\}$ or $\{e'_0 = x_i^5 c_j^8, e'_1 = x_i^6 c_j^7\}$. Let $A = \{x_i^3, x_i^4, x_i^5, x_i^6, c_j^7, c_j^8\}$ be the set of the endpoints of these edges, and examine $V(L) \cap A$ below.

Case 1. $\ell_j^2 = x_i$: If $\alpha(\ell_j^2) = \alpha(x_i) = \texttt{true}$ (resp., $\texttt{false}$), then $L_i^{\mathrm{V}} = T_i^{\mathrm{V}}$, $L_j^{\mathrm{C}} \supseteq T_{j,2}^{\mathrm{C}}$ and $V(L) \cap A = \{x_i^3, x_i^6, c_j^7\}$ (resp., $L_i^{\mathrm{V}} = F_i^{\mathrm{V}}$, $L_j^{\mathrm{C}} \supseteq F_{j,2}^{\mathrm{C}}$ and $V(L) \cap A = \{x_i^4, x_i^5, c_j^8\}$).

Case 2. $\ell_j^2 = \neg x_i$: If $\alpha(\ell_j^2) = \alpha(\neg x_i) = \texttt{true}$ (resp., $\texttt{false}$), then $L_i^{\mathrm{V}} = F_i^{\mathrm{V}}$, $L_j^{\mathrm{C}} \supseteq T_{j,2}^{\mathrm{C}}$ and $V(L) \cap A = \{x_i^4, x_i^5, c_j^7\}$ (resp., $L_i^{\mathrm{V}} = T_i^{\mathrm{V}}$, $L_j^{\mathrm{C}} \supseteq F_{j,2}^{\mathrm{C}}$ and $V(L) \cap A = \{x_i^3, x_i^6, c_j^8\}$).

In any case, each edge in $E_{i,j}^2$ is incident to a vertex in $V(L) \cap A$. Consequently L is an edge dominating set of G_I of size $\mu(G_I)/2$.

(II) If part: Let L be a tight edge dominating set in G_I. Note that G_I has a perfect matching $M = \bigcup_{1 \le i \le n} M_i^{\mathrm{V}} \cup \bigcup_{1 \le j \le m} M_j^{\mathrm{C}}$. By Lemma 1, L is a matching of G_I with $L \cap M_{G_I} = \emptyset$ and each edge in L 1-dominates exactly two edges in $M \subseteq M_{G_I}$. By the structure of gadgets, we see that such a matching L of G_I must satisfy the following conditions:

(a-1) For every variable gadget G_i^{V}, $L \cap E(G_i^{\mathrm{V}})$ equals either T_i^{V} or F_i^{V}; and

(a-2) For every clause gadget G_j^{C}, there is an index h with $\{h, t, t'\} = \{1,2,3\}$ such that

$$L \cap E(G_j^{\mathrm{C}}) = T_{j,h}^{\mathrm{C}} \cup F_{j,t}^{\mathrm{C}} \cup F_{j,t'}^{\mathrm{C}}.$$

Let $\alpha : \mathcal{X} \to \{\texttt{true}, \texttt{false}\}$ be a truth assignment obtained from L as follows: for each variable $x_i \in \mathcal{X}$,

$$\alpha(x_i) := \begin{cases} \texttt{true} & \text{if } L \cap E(G_i^{\mathrm{V}}) = T_i^{\mathrm{V}}; \\ \texttt{false} & \text{if } L \cap E(G_i^{\mathrm{V}}) = F_i^{\mathrm{V}}. \end{cases}$$

We then ensure that this truth assignment is satisfiable to the original instance I; that is, each clause $c_j \in \mathcal{C}$ has exactly one true literal. For this, it suffices to prove that, for the indices h, t, t' in (a-2), it holds

$$\alpha(\ell_j^k) = \begin{cases} \texttt{true} & \text{if } k = h; \\ \texttt{false} & \text{if } k = t, t'. \end{cases}$$

Without loss of generality consider the case of $k = 2$, as shown in Fig. 1, where $E_{i,j}^2 = \{e_0 = x_i^3 c_j^8, e_1 = x_i^4 c_j^7\}$ or

$\{e'_0 = x_i^5 c_j^8, e'_1 = x_i^6 c_j^7\}$.

Case 1. $h = 2$: Then $c_j^8 \notin V(L)$, but F dominates all edges incident to this vertex. Hence if $\ell_j^2 = x_i$ (resp., $\ell_j^2 = \neg x_i$), then $x_i^3 \in T_i^{\mathrm{V}} \subseteq V(L)$ and $\mathtt{true} = \alpha(x_i) = \alpha(\ell_j^2)$ (resp., $x_i^5 \in F_i^{\mathrm{V}} \subseteq V(L)$ and $\mathtt{false} = \alpha(x_i) = \alpha(\neg\ell_j^2)$), as required.

Case 2. t or $t' = 2$: Then $c_j^7 \notin V(L)$, but F dominates all edges incident to this vertex. Hence if $\ell_j^2 = x_i$ (resp., $\ell_j^2 = \neg x_i$), then $x_i^4 \in F_i^{\mathrm{V}} \subseteq V(L)$ and $\mathtt{false} = \alpha(x_i) = \alpha(\ell_j^2)$ (resp., $x_i^6 \in T_i^{\mathrm{V}} \subseteq V(L)$ and $\mathtt{true} = \alpha(x_i) = \alpha(\neg\ell_j^2)$), as required.

Consequently, the truth assignment α is satisfiable to $I = (\mathcal{X}, \mathcal{C})$. □

4. Exact Algorithm

This section designs an exact branching algorithm to Lowest Edge Dominating Set after making some preparation.

4.1 Odd Size of Maximum Matchings

The next lemma tells that an instance with an odd size of maximum matchings can be converted into several instances with an even size of maximum matchings

Lemma 4: Let $G = (V, E)$ be a graph with odd $\mu(G)$. Then G has a tight edge dominating set if and only if for some edge $uv \in M_G$, the graph $G' = (V \cup \{x, y\}, E \cup \{ux, vy\})$ augmented with new vertices x, y and edges ux and vy has a tight edge dominating set, where always $\mu(G') = \mu(G) + 1$ holds.

Proof. The if part and $\mu(G') \leq \mu(G) + 1$: Let F be a tight edge dominating set of G', where we can assume that $F \cap \{ux, vy\} = \emptyset$ since if $F \cap \{ux, vy\} \neq \emptyset$ then $F' = (F \setminus \{ux, vy\}) \cup \{uv\}$ is also an edge dominating set of G' with $|F'| \leq |F|$. Since $F \cap \{ux, vy\} = \emptyset$, F is also an edge dominating set of G, where $|F| = \lceil \mu(G')/2 \rceil$ since F is tight in G'. Let M be a maximum matching of G', where we can assume that $\{ux, vy\} \subseteq M$ since $M' := (M \setminus (E_{G'}(u) \cup E_{G'}(v))) \cup \{ux, vy\}$ is also a matching of G' with $|M'| \geq |M|$. Since $(M \setminus \{ux, vy\}) \cup \{uv\}$ is a matching of G, we have $\mu(G) \geq |M| - 1 = \mu(G') - 1$. Hence $|F| = \lceil \mu(G')/2 \rceil \leq \lceil (\mu(G) + 1)/2 \rceil = \lceil \mu(G)/2 \rceil$ since $\mu(G)$ is odd, implying that F is also tight in G.

The only if part and $\mu(G') \geq \mu(G) + 1$: We choose a maximum matching M and a tight edge dominating set F in G so that $|V(M) \cap V(F)|$ is maximized. Observe that every edge in F 1-dominates at least two edges in M, since if some edge $e \in F$ 1-dominates no edge in M or only one edge $e' \in M$, then $M \cup \{e\}$ or $M \cup \{e\} \setminus \{e'\}$ would be a maximum matching having more common endpoints with F. Since $\mu(G) = |M|$ is odd and $|F| > |M|/2$, some edge $uv \in M$ is dominated by two edges $aa', bb' \in F$, where $a, b \in \{u, v\}$ and aa' dominates another edge $a'w \in M$. We see that $a \neq b$, because if $a = b$ then $F \cup \{a'w\} \setminus \{aa'\}$ would be a tight edge dominating set having more common endpoints with M. Hence $a \neq b$, and F 2-dominates edge $uv \in M$. Clearly $M' = (M \setminus \{uv\}) \cup \{ux, vy\}$ is a matching of G' and $\mu(G') \geq \mu(G) + 1$. Since $u, v \in V(F)$ and F is tight in G, we see that F is an edge dominating set of G' such that $|F| = (\mu(G) + 1)/2 \leq \mu(G')/2$, implying that F is also tight in G'. □

If a graph G such that $\mu(G)$ is odd is given for Lowest Edge Dominating Set, then based on Lemma 4, we can construct $|M_G| = O(n^2)$ graphs G' for the problem such that $\mu(G')$ is even, in order to solve the original instance. In the rest of this paper, we assume that $\mu(G)$ in a graph G is even.

4.2 Restricted Lowest Edge Dominating Set

To Lowest Edge Dominating Set, we design a branching algorithm which branches into two cases: a vertex v is in the set $V(F)$ of a tight edge dominating set F or not. During this process, a set C of some vertices *decided* to be *covered* by $V(F)$ and a set D of some vertices *decided* to be *discarded* from $V(F)$ for a tight edge dominating set F will be specified. In fact, we handle Lowest Edge Dominating Set with the following restriction in our algorithm.

Restricted Lowest Edge Dominating Set
Instance: A tuple (G, C, D) of a graph G such that $\mu(G)$ is even and two disjoint subsets $C, D \subseteq V(G)$.
Question: Does G have a tight (C, D)-eds?

Notice that a tight $(\emptyset, \emptyset)$-eds of G is a tight edge dominating set of G. Given an instance (G, C, D), we always denote by U the set $V(G) \setminus (C \cup D)$ of *undecided* vertices. A connected component of a graph is called a *clique component* if it is a complete graph. Van Rooij and Bodlaender [10] found the following solvable case.

Lemma 5: [10] A minimum (C, D)-eds of an instance (G, C, D) such that $G[U]$ contains only clique components can be found in polynomial time.

Let U_1 denote the set of vertices of all clique components in $G[U]$, and let $U_2 = U \setminus U_1$. We call an instance (G, C, D) such that $U_2 = \emptyset$ a *leaf instance*, to which we can test whether or not G has a tight (C, D)-eds in polynomial time by checking if a minimum (C, D)-eds F obtained by Lemma 5 meets $|F| = \mu(G)/2$ or not.

Our algorithm, called RLEDS, consists of two procedures, called Reduce and Branch.

Procedure Reduce applies some reduction rule to a given instance (G, C, D), by which a new instance with a smaller set U of undecided vertices is constructed or it turns out that the given instance is *infeasible*, i.e., it admits no tight (C, D)-eds. Procedure Reduce then returns a *reduced instance*, an instance to which no reduction rule is applicable or a message that the given instance is infeasible. Given a reduced instance (G, C, D), procedure Branch applies a branching rule by which two new instances (G, C'_i, D'_i), $i = 1, 2$ are constructed, and tests whether G has a tight (C, D)-eds or not by examining whether one of the two instances admits a tight (C'_i, D'_i)-eds. Procedure Reduce is recursively executed until the resulting instance becomes a leaf instance (G, C', D'), to which we test whether there is a tight (C', D')-eds or not based on Lemma 5. Sections 4.3 and 4.4 describe the reduction rules and branching rule in procedures Reduce and Branch, respectively.

4.3 Reduction Rules

This section first introduces three reduction rules. We let (G, C, D) be a given instance.

Reduction rule 1: Assume that $U \cap R_G \neq \emptyset$. Let $\Delta D = U \cap R_G$. Then a subset $F \subseteq E(G)$ is a tight (C, D)-eds if and only if it is a tight (C', D')-eds with $D' = D \cup \Delta D$ and $C' = C \cup N_{G[U]}(\Delta D)$.

Correctness. By Lemma 1, every vertex in $V(F)$ for any tight (C, D)-eds F must be incident to an end-vertex of some maximum matching M of G. Hence moving all the vertices in $\Delta D = U \cap R_G$ to D will not lose any tight (C, D)-eds. In this case, we also include $N_{G[U]}(\Delta D)$ into C, since the other endpoint of any edge incident to a vertex in D needs to be in C. □

Reduction rule 2: Assume that $U \cap N_{G[M_G]}(C) \neq \emptyset$. Let $\Delta D = U \cap N_{G[M_G]}(C)$. Then a subset $F \subseteq E(G)$ is a tight (C, D)-eds if and only if it is a tight (C', D')-eds with $D' = D \cup \Delta D$ and $C' = C \cup N_{G[U]}(\Delta D)$.

Correctness. For all edges of M_G between C and U, their end-vertices in C are required to be included in $V(F)$ of any tight (C, D)-eds F. By Lemma 1, moving all the vertices in $\Delta D = U \cap N_{G[M_G]}(C)$ to D will not lose any tight (C, D)-eds. In this case, we also include $N_{G[U]}(\Delta D)$ into C, as in Reduction rule 1. □

Reduction rule 3: If there is an edge $uv \in M_G$ with $\{u, v\} \subseteq C$, then the instance (G, C, D) has no tight (C, D)-eds.

Correctness. Immediate from Lemma 1. □

We apply the above three rules as much as possible in this order. Note that Reduction rule 1 is no longer applicable once $U \cap R_G = \emptyset$ holds. Only Reduction rule 2 may be applied more than once. If none of the above three rules is applicable, then the algorithm switches to procedure Branch. Formally, we describe procedure Reduce as follows.

Procedure Reduce(G, C, D)

Input: A graph $G = (V, E)$ and two disjoint subsets $C, D \subseteq V$.
Output: "infeasible" if Reduction Rule 3 is applied during a process of reducing the input instance; otherwise a reduced instance from (G, C, D)

/* Reduction Rule 1 */
$\Delta D := U \cap R_G$; $C := C \cup N_{G[U]}(\Delta D)$; $D := D \cup \Delta D$;
/* Reduction Rule 2 */
while $U \cap N_{G[M_G]}(C) \neq \emptyset$ **do**
  $\Delta D := U \cap N_{G[M_G]}(C)$;
  $C := C \cup N_{G[U]}(\Delta D)$; $D := D \cup \Delta D$
end while;
/* Reduction Rule 3*/
if $\exists uv \in M_G$ with $\{u, v\} \subseteq C$ **then**
  return "infeasible"
else /* Now (G, C, D) is a reduced instance */
  return (G, C, D)
end if

We observe the structure of reduced instances.

Lemma 6: Any reduced instance (G, C, D) satisfies all the following three conditions:

(a) $M_G \cap \{uv \in E(G) \mid u \in C, v \in U\} = \emptyset$;
(b) $U \subseteq V(M_G)$; and
(c) For every connected component H in $G[U_2]$, the set $M_G \cap E(H)$ contains a perfect matching of H and is the union of all perfect matchings of H.

Proof. (a) There is no edge in M_G between C and U, because otherwise Reduction rule 2 would be applicable.
(b) Any vertex $v \in U$ is incident to an edge in M_G; otherwise $v \notin V(M_G)$ implies $v \in R_G$ and Reduction rule 1 would be applicable.
(c) No edge in M_G exists between C and U because of condition (a). Hence (i) any maximum matching of H is contained in some maximum matching of G; and (ii) for any maximum matching M of G, the set $M \cap E(H)$ is a maximum matching of H, where $M \cap E(H)$ is a perfect matching of H since otherwise $V(H) \setminus V(M) \neq \emptyset$ would imply $U \cap R_G \neq \emptyset$, contradicting that Reduction rule 1 is not applicable. Therefore $M_G \cap E(H)$ is the union of all perfect matchings of H. □

4.4 Branching Rule

This section presents a branching rule in procedure Branch. We let (G, C, D) denote an instance given to the procedure. First we give a priority among the vertices in $G[U_2]$: A vertex $v \in U_2$ is called *optimal* if it satisfies condition (c-i) below with the minimum i over all vertices in $G[U_2]$:

(c-1) $d_{G[U_2]}(v) \geq 4$;
(c-2) $d_{G[U_2]}(v) \geq 2$ and there is a neighbor $u \in U_2$ of v such that $d_{G[U_2]}(u) \geq 2$ and $uv \in M_G$;
(c-3) $d_{G[U_2]}(v) \geq 2$ and there is a neighbor $u \in U_2$ of v such that $d_{G[U_2]}(u) = 2$ and $uv \in E(G) \setminus M_G$; and

(c-4) v is of maximum degree in $G[U_2]$.

The algorithm applies the following branching rule on an optimal vertex.

Branching rule 1: Let v be a vertex in $G[U_2]$. Then G has a tight (C, D)-eds if and only if G has a tight $(C \cup \{v\}, D)$-eds or a tight $(C \cup N_{G[U]}(v), D \cup \{v\})$-eds.

Correctness. For a (C, D)-eds F of G, the vertex $v \in U_2$ is contained in $V(F)$ or in $V(G) \setminus V(F)$. In the first case, F is a $(C \cup \{v\}, D)$-eds of G. In the second case, all the neighbors of v in G must be contained in $V(F)$ so that F dominates all edges in $E_G(v)$. Therefore F is a $(C \cup N_G(v), D \cup \{v\})$-eds of G. □

Formally, we describe procedure Branch as follows.

Procedure Branch(G, C, D)

Input: A graph $G = (V, E)$ and two disjoint subsets $C, D \subseteq V$.
Output: `true` if G has a tight (C, D)-eds; otherwise `false`.

```
if REDUCE(G, C, D) returns "infeasible" then
    return false
else
    (G, C, D) := REDUCE(G, C, D);
    if (G, C, D) is a leaf instance then
        Compute a minimum (C, D)-eds F of G
        based on Lemma 5;
        if |F| = μ(G)/2 then /* F is tight */
            return true
        else /* F is not tight */
            return false
        end if
    else
        Let v be an optimal vertex in G[U_2];
        return BRANCH(G, C ∪ {v}, D)∨
                BRANCH(G, C ∪ N_G[U](v), D ∪ {v})
    end if
end if
```

Then our algorithm is described as follows.

Procedure RLEDS

Input: A graph $G = (V, E)$.
Output: `true` if G has a tight edge dominating set; otherwise `false`.

```
return BRANCH(G, C := ∅, D := ∅)
```

4.5 Analysis

This section analyzes the time complexity of the algorithm by establishing the following theorem.

Theorem 7: Algorithm RLEDS can test whether or not a given graph G has a tight edge dominating set in $O^*(1.44225^{\mu(G)}) = O^*(2.0801^{\mu(G)/2})$ time and polynomial space.

We easily see that the space complexity is polynomial in n. We evaluate the time complexity as an upper bound on the size of the search tree of RLEDS, or the number of leaf instances generated by RLEDS. For a tight edge dominating set F of G, it holds $|V(F)| \le 2|F| = \mu(G)$. Then we define the measure $\tau(I)$ of an instance $I = (G, C, D)$ to be

$$\mu(G) - |C| - \sum_{\text{clique components } Q \text{ in } G[U]} (|V(Q)| - 1),$$

where $\tau(I) \le \mu(G)$. Let $T(\tau)$ be the maximum number of leaf instances that can be generated from an instance of measure τ by algorithm RLEDS. By solving some recurrences on $T(\tau)$ in the following, we derive an upper bound on $T(\tau)$ for an instance $I = (G, C, D)$ with $\tau(I) = \tau$ as an exponential function $O^*(\beta^\tau)$ of τ $(\le \mu(G))$.

Lemma 8: When algorithm RLEDS branches on a vertex v satisfying condition (c-1) in $G[U_2]$, the measure change meets the following recurrence:

$$T(\tau) \le T(\tau - 1) + T(\tau - 4), \tag{1}$$

which solves to $T(\tau) = O(1.3803^k)$.

Proof. The first (resp., second) branch includes v (resp., $N_{G[U]}(v)$) into C, which decreases the measure by 1 (resp., $|N_{G[U]}(v)| \ge 4$). Hence we have the recurrence (1). □

Lemma 9: When algorithm RLEDS branches on a vertex v satisfying condition (c-2) in $G[U_2]$ with a neighbor u of v, the measure change meets the following recurrence:

$$T(\tau) \le 2T(\tau - 2), \tag{2}$$

which solves to $T(\tau) = O(1.4143^k)$.

Proof. The first branch includes vertex v into C, and then Reduction rule 2 is applied to vertex u, implying that $N_{G[U_2]}(u)$ is included into C before the next branching. Since $d_{G[U_2]}(u) \ge 2$, the measure decreases by at least $|N_{G[U]}(u)| \ge 2$. The second branch includes $N_{G[U_2]}(v)$ into C, implying that the measure decreases by $|N_{G[U]}(v)| \ge 2$. Hence we have the recurrence (2). □

The next lemma is used to prove Lemma 11 and Lemma 13.

Lemma 10: Let (G, C, D) be a reduced instance where $U_2 \ne \emptyset$ and no vertex in $G[U_2]$ satisfies condition (c-2). Then $G[U_2]$ has a unique perfect matching M, it holds $M = M_G \cap E(G[U_2])$, and each edge in M joins a vertex of degree 1 and a vertex of degree 2 or 3 in $G[U_2]$, while each edge in $E(G[U_2]) \setminus M$ joins vertices of degree 2 or 3.

Proof. Each vertex in U_2 of degree at most 3 in $G[U_2]$ because no vertex satisfies in U_2 condition (c-1). Since v does not satisfy condition (c-2), it holds $d_{G[U_2]}(u) = 1$ or

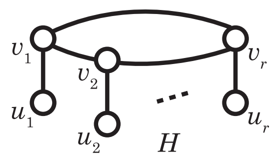

Fig. 2 A connected component H with $r \geq 3$ in $G[U_2]$ when none of conditions (c-1), (c-2) and (c-3) is applicable to a reduced instance (G, C, D).

$d_{G[U_2]}(v) = 1$. At least one of u and v is of degree 2 or 3, since otherwise $G[U_2]$ would contain a clique component of size 2. Then $G[U_2]$ has a unique perfect matching M, which must be equal to $M_G \cap E(G[U_2])$ by Lemma 6(c). Clearly any edge in $G[U_2]$ incident to a vertex of degree 1 must be in the maximum matching M. □

Lemma 11: When algorithm RLEDS branches on a vertex v satisfying condition (c-3) in $G[U_2]$ with a neighbor u of v, the measure change meets the following recurrence:

$$T(\tau) \leq 2T(\tau - 2), \tag{3}$$

which solves to $T(\tau) = O(1.4143^k)$.

Proof. Since v satisfies condition (c-3), it has a neighbor $u \in U_2$ with $d_{G[U_2]}(u) = 2$, where $uv \notin M_G$ since v does not satisfy condition (c-2). By Lemma 10, the other neighbor $w \in U_2$ of u in $G[U_2]$ is of degree 1. Therefore after the first branch includes v into C, vertices u and w induce a clique component of size 2 and will be included into U_1. This implies that the measure decreases in total by 2 after the first branch. In the second branch, $N_{G[U]}(v)$ is included into C and the measure decreases by $|N_{G[U]}(v)| \geq 2$. Hence we have the recurrence (3). □

Lemma 12: Let (G, C, D) be a reduced instance where no vertex in $G[U_2]$ satisfies any of conditions (c-1), (c-2) and (c-3). Then any connected component H in $G[U_2]$ consists of a cycle of length $r \geq 3$ and r vertices of degree 1 adjacent to each vertex in C, as illustrated in Fig. 2.

Proof. From Lemma 10, we see that any edge $uv \in E(H) \setminus M_G$ such that the degree of u or v is 2 satisfies that condition (c-3). Hence now no such edge exists and each vertex in H is of degree either 1 or 3. This determines the structure of H to be a union of a perfect matching M on $V(H)$ and a cycle C of length $|V(H)|/2$ that visits exactly one of the endpoints of each edge in M. Since $|C| \geq 3$ in a simple graph, it holds $|V(H)| \geq 6$. □

Lemma 13: When algorithm RLEDS branches on a vertex v satisfying condition (c-4) in $G[U_2]$, the measure change meets the following recurrence:

$$T(\tau) \leq 3T(\tau - 3), \tag{4}$$

which solves to $T(\tau) = O(1.44225^k)$.

Proof. By Lemma 12, the connected component H in $G[U_2]$ containing the vertex v consists of a set $\{u_i, v_i \mid i = 1, \ldots, r\}$ of $2r \geq 6$ vertices, where $v = v_1$, and a set $\{u_i v_i, v_i v_{i+1} \mid i = 1, \ldots, r\}$ of $2r$ edges, where $v_{r+1} = v_1$.

After the first branch includes $v = v_1$ into C decreasing the measure by 1, vertex u_1 will be moved to D by Reduction rule 2. In the resulting graph $G[U \setminus \{v_1, u_1\}]$, both vertices v_2 and v_r become of degree 2 in $G[U \setminus \{v_1, u_1\}]$, which are adjacent to vertices u_2 and u_r of degree 1, respectively; therefore each of v_3 and v_{r-1} satisfies condition (c-3) in $G[U \setminus \{v_1, u_1\}]$. Note that no vertex in $V(H)$ satisfies condition (c-1) or (c-2) in $G[U \setminus \{v_1, u_1\}]$, since no vertex in $V(H)$ is of degree at least 4 in $G[U \setminus \{v_1, u_1\}]$ and every edge $e \in E(H)$ is adjacent to an endpoint of degree 1 in $G[U \setminus \{v_1, u_1\}]$ or $e \notin M_G$ by Lemma 12. Then the algorithm branches on a vertex in $G[U \setminus \{v_1, u_1\}]$ satisfying condition (c-3) with the recurrence (3).

The second branch includes $N_{G[U]}(v_1)$ into C decreasing the measure by $|N_{G[U]}(v_1)| = 3$.

Therefore we have the following recurrence:

$$T(\tau) \leq 2T(\tau - 1 - 2) + T(\tau - 3) = 3T(\tau - 3),$$

which is the recurrence (4). □

Proof of **Theorem 7**. Among all the recurrences (1), (2), (3) and (4), the maximum branch factor 1.44225 is attained by recurrence (4). Note that the measure τ is at most $\mu(G)$. Therefore the algorithm solves the problem in $O^*(1.44225^{\tau}) = O^*(1.44225^{\mu(G)}) = O^*(2.0801^{\mu(G)/2})$ time. □

5. Conclusions

In this paper, we have studied Lowest Edge Dominating Set, which asks us to test whether a given graph G has an edge dominating set whose size is equal to $\lceil \mu(G)/2 \rceil$, a lower bound on the size of an edge dominating set of G. We proved that the problem remains NP-complete and showed that it admits an $O^*(2.0801^{\mu(G)/2})$-time and polynomial-space algorithm, whose time bound is better than the currently best bound $O^*(2.2351^{\mu(G)/2})$ to Parameterized Edge Dominating Set [7]. We see that the bottleneck of the time bound is attained by the branching on a vertex in a component in $G[U_2]$ mentioned in Lemma 12 with $r = 3$.

There arises a further question: for another parameter $\Delta \geq 0$, the problem of testing whether a given graph G has an edge dominating set of size at most $\lceil \mu(G)/2 \rceil + \Delta$ or not can be solved in $O^*(2.2351^{\mu(G)/2} \cdot 2.2351^{\Delta})$ time by setting $k = \lceil \mu(G)/2 \rceil + \Delta$ in the $O^*(2.2351^k)$-time algorithm to Parameterized Edge Dominating Set [7]. Does the problem admit an algorithm with a better time bound, say $O^*(2.0801^{\mu(G)/2} \cdot 2.2351^{\Delta})$? Notice that for $\Delta = 0$, we have shown that it can be solved in $O^*(2.0801^{\mu(G)/2})$ time.

References

[1] D. Binkele-Raible and H. Fernau, "Parameterized Measure & Conquer for Problems with No Small Kernels," Algorithmica, vol.64, no.1, pp.189–212, 2012.

[2] H. Fernau, "Edge Dominating Set: Efficient Enumeration-Based Exact Algorithms," In H. Bodlaender, M. Langston (eds.), IWPEC 2006. LNCS 4169, Springer-Verlag, pp.142–153, 2006.
[3] F.V. Fomin, S. Gaspers, S. Saurabh, and A.A. Stepanov, "On Two Techniques of Combining Branching and Treewidth," Algorithmica, vol.54, no.2, pp.181–207, 2009.
[4] T. Fujito and H. Nagamochi, "A 2-Approximation Algorithm for the Minimum Weight Edge Dominating Set Problem," Discrete Applied Mathematics, vol.118, no.3, pp.199–207, 2002.
[5] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to The Theory of NP-Completeness, Freeman, San Francisco, 1979.
[6] F. Gavril, "Testing for Equality Between Maximum Matching and Minimum Node Covering," Information Processing Letters, vol.6, no.6, pp.199–202, 1977.
[7] K. Iwaide and H. Nagamochi, "An Improved Algorithm for Parameterized Edge Dominating Set Problem," Journal of Graph Algorithms and Applications, vol.20, no.1, pp.23–58, 2016.
[8] V. Raman, S. Saurabh, and S. Sikdar, "Efficient Exact Algorithms through Enumerating Maximal Independent Sets and Other Techniques," Theory of Computing Systems, vol.42, no.3, pp.563–587, 2007.
[9] B. Randerath and I. Sciermeyer, "Exact Algorithms for Minimum Dominating Set," Technical Report zaik 2005-501, Universität zu Köln, Cologne, Germany, 2005.
[10] J.M.M. van Rooij and H.L. Bodlaender, "Exact Algorithms for Edge Domination," In M. Grohe, R. Niedermeier (eds.), IWPEC 2008. LNCS 5018, Springer-Verlag, pp.214–225, 2008.
[11] V.V. Vazirani, "A Theory of Alternating Paths and Blossoms for Proving Correctness of the $O(\sqrt{V}E)$ General Graph Maximum Matching Algorithm," Combinatorica, vol.14, no.1, pp.71–109, 1994.
[12] M. Xiao, T. Kloks, and S.-H. Poon, "New Parameterized Algorithms for the Edge Dominating Set Problem," Theoretical Computer Science, vol.511, pp.147–158, 2013.
[13] M. Xiao and H. Nagamochi, "Parameterized Edge Dominating Set in Cubic Graphs," In M. Atallah, X.-Y. Li, B. Zhu (eds.), FAW-AAIM 2011. LNCS 6681, Springer-Verlag, pp.100–112, 2011.
[14] M. Xiao and H. Nagamochi, "A Refined Exact Algorithm for Edge Dominating Set," In A. Manindra, S. Barry, L. Angsheng (eds.), TAMC 2012. LNCS 7287, Springer-Verlag, pp.360–372, 2012
[15] M. Xiao and H. Nagamochi, "Exact Algorithms for Annotated Edge Dominating Set in Graphs with Degree Bounded by 3," IEICE Trans. Inf. & Syst., vol.E96-D, no.3, pp.408–418, 2013.
[16] M. Yanakakis and F. Gavril, "Edge Dominating Set in Graphs," SIAM J. Appl. Math., vol.38, no.3, pp.364–372, 1980.

Ken Iwaide was born on December 24, 1991. He received the B.A. degree from Kyoto University in 2014. His research interests include optimization problems and graph theory.

Hiroshi Nagamochi was born in Tokyo, on January 1, 1960. He received the B.A., M.E. and D.E. degrees from Kyoto University, in 1983, in 1985 and in 1988, respectively. He is a Professor in the Department of Applied Mathematics and Physics, Graduate School of Informatics, Kyoto University. His research interests include network flow problems and graph connectivity problems. Dr. Nagamochi is a member of the Operations Research Society of Japan, the Information Processing Society, and the Japan Society for Industrial and Applied Mathematics.