Detecting Whether a Graph Has a Fixed-point-free Automorphisms Is in Polynomial Time

Yasunori Ohto

Abstract

The problem that to determining whether a graph has a fixed-point-free automorphism is NP-complete. We show that it is solvable in polynomial time. First, we obtain the automorphisms of an input graph G by using a spectral method. Next, we prove the Theorem used to detect whether there is a fixed-point free automorphism in G. Next, we construct an algorithm to detect whether G has a fixed-point-free automorphism using this result. This algorithm cannot obtain a fixed-point-free automorphism even if it exists. The computational complexity of this problem is $\mathcal{O}(n^5)$. Then, the complexity classes P and NP are the same.

Index Terms

fixed-point-free automorphism, graph spectrum, polynomial time computation, NP-complete problem.

I. Introduction

The P versus NP problem [1], [2] is one of the major problems in theoretical computer science. An answer to this problem would determine whether problems that can be verified in polynomial time can also be solved in polynomial time. Attempts have been made to prove that P is not equal to NP. However, it has been shown that this cannot be proven or is difficult to prove using the methods of relativizing proofs [3], natural proofs [4], and algebrizing proofs [5]. On the other hand, many attempts have been made to show the lower bound of NP problems [6], [7], mainly by pruning conditional searches [8], but it is still unclear whether P and NP are equal. In contrast, we will show a lower bound on the computational complexity of an NP-complete problem by introducing a spectral method to handle multiple states at once.

If a problem is NP and all other NP problems are polynomial-time reducible to it, the problem is NP-complete [9]. If one of the NP-complete problems can be solved in polynomial time, the complexity classes P and NP are the same. The problem of determining whether a given graph has a fixed-point-free automorphism is NP-complete [10]. In this paper, we show that it is solvable in polynomial time. Note that this algorithm has a limitation in that it can only detect whether an input graph has a fixed-point-free automorphism, not obtain a fixed-point-free automorphism even if it exists.

First, we define the following functions. Let S be a vertex-weighted graph. Let $V_{w0}(S)$ be the set of vertices with weight 0 of S. Let $Sg(S, v, w)$ be a vertex-weighted graph in which a weight $w \in \mathbb{N}$ is given to a vertex v of S. Let $Ev(S)$ be the eigenvalue set of the adjacency matrix of S. Next, we use Theorem II.2 [11] to obtain the automorphisms of an input graph G using eigenvalue sets.

Theorem II.2. Let $S_{v_i} = Sg(S, v_i, w)$ and $S_{v_j} = Sg(S, v_j, w)$ with $v_i, v_j \in V_{w0}(S)$, $v_i \neq v_j$ and $w > 0$. When $Ev(S_{v_i}) = Ev(S_{v_j})$, S_{v_i} and S_{v_j} are isomorphic.

Next, we prove Theorem II.5 to detect whether there is a fixed-point-free automorphism in G.

Theorem II.5. We obtain the vertex sets $V_\lambda \subset V$ with the same $\lambda_v = Ev(Sg(S, v, w))$, $v \in V$, $w > 0$. $V_{\lambda_{v_i}} > 1$ for all vertex sets of λ_{v_i} if, and only if, G has a fixed-point-free automorphism.

Next, we construct an algorithm to detect whether a graph has a fixed-point-free automorphism using this result. Since the elements of an adjacency matrix of a vertex-weighted graph are all integers, the coefficients of the eigenequation of this matrix are all integers. Then, we calculate the Frobenius normal form [12], [13] to obtain the coefficients of the eigenequation of this matrix without real number calculations. Then, we compare the coefficients to determine whether the sets of eigenvalue are the same. The computational complexity of detecting whether a graph has a fixed-point-free automorphism is $\mathcal{O}(n^5)$. Since one of the NP-complete problems is solvable in polynomial time, the complexity classes P and NP are the same.

This paper is organized as follows. Section II provides the proofs used to determine whether a given graph has a fixed-point-free automorphism. Section III presents an algorithm to solve this problem. Finally, Section IV presents a conclusion regarding the result of this paper.

II. Proof

In this section, we provide the proofs for the results of detecting whether a given graph has a fixed-point-free automorphism.

A. Preparation

We define the following functions, which will be used in the proofs and the methods. Suppose S is a vertex-weighted graph. Let $V_{w0}(S)$ be the set of vertices of S with weight 0. Let $Sg(S, v, w)$ be the vertex-weighted graph in which the weight $w \in \mathbb{N}$ is given to vertex v of S. Denote the adjacency matrix of S by $A(S)$. Let $Ev(S)$ be the set (with multiplicities) of eigenvalues of $A(S)$.

B. Obtain the automorphisms

In this subsection, we provide the proof and use the Theorem and the Corollary for obtaining the automorphisms in G.

1) Automorphism composition: Let $Aut(G)$ be the automorphism group of G. We can obtain $\psi \in Aut(G)$ by composition of automorphisms of order two.

Corollary II.1. *There are certain automorphisms $\psi_1, \psi_2, \ldots, \psi_m \in Aut(G)$ of order two, and we can explain $\psi = \psi_m \psi_{m-1} \cdots \psi_1$.*

Proof. Permuting vertices of ψ consists of transposition (automorphism of order two) and cycling (automorphism of order above two). Suppose that a composition of automorphisms has a cycle σ_r of order r. Now, there exists two transpositions. One is the automorphism containing the transposition $\sigma_{1,2}$. And the other is the automorphism containing the transposition $\sigma_{2,r}$. Then, we obtain σ_r by successively applying the automorphism containing $\sigma_{1,2}$ and the automorphism containing $\sigma_{2,r}$. Therefore, we can reduce all cycling to composition of transpositions. □

For example, suppose a cycle σ_{c5} of order 5. The automorphism containing the transposition $\sigma_{1,2}$ is $(\sigma_1, \sigma_2)(\sigma_3, \sigma_5)$. The automorphism containing the transposition $\sigma_{2,5}$ is $(\sigma_2, \sigma_5)(\sigma_3, \sigma_4)$. Thus, we obtain $\sigma_{c5} = (\sigma_1, \sigma_2)(\sigma_3, \sigma_5)(\sigma_2, \sigma_5)(\sigma_3, \sigma_4)$. And another example, suppose a cycle σ_{c6} of order 6. The automorphism containing the transposition $\sigma_{1,2}$ is $(\sigma_1, \sigma_2)(\sigma_3, \sigma_6)(\sigma_4, \sigma_5)$. The automorphism containing the transposition $\sigma_{2,6}$ is $(\sigma_2, \sigma_6)(\sigma_3, \sigma_5)$. Thus, we obtain $\sigma_{c6} = (\sigma_1, \sigma_2)(\sigma_3, \sigma_6)(\sigma_4, \sigma_5)(\sigma_2, \sigma_6)(\sigma_3, \sigma_5)$.

2) Obtain the automorphisms that contain transposing two vertices: An automorphism of order two contains transposing two vertices. So, we remove fixed points by compositions of automorphisms that contain transposing two vertices. Thus, we use Theorem II.2 and Corollary II.3 [11] to obtain the automorphisms that contain transposing two vertices of an input graph G using eigenvalue sets.

Theorem II.2. *Let $S_{v_i} = Sg(S, v_i, w)$ and $S_{v_j} = Sg(S, v_j, w)$ with $v_i, v_j \in V_{w0}(S)$, $v_i \neq v_j$ and $w > 0$. If $Ev(S_{v_i}) = Ev(S_{v_j})$, then S_{v_i} and S_{v_j} are isomorphic.*

Corollary II.3. *Let $S_{v_i} = Sg(S, v_i, w)$ and $S_{v_j} = Sg(S, v_j, w)$ with $v_i, v_j \in V_{w0}(S)$, $v_i \neq v_j$ and $w > 0$. If $Ev(S_{v_i}) \neq Ev(Sv_j)$, then S_{v_i} and S_{v_j} are not isomorphic.*

So, when $Ev(S_{v_i}) = Ev(S_{v_j})$ if, and only if, there is an automorphism in G that contain the transposition of v_i and v_j.

These proofs are reproduced in the appendix.

C. Detect whether there is a fixed-point-free automorphism

In this subsection, we provide the proofs for the results of detecting whether a given graph has a fixed-point-free automorphism from obtained the automorphisms.

Lemma II.4 and Theorem II.5 prove that it is possible to detect whether there is a fixed-point-free automorphism in G.

1) Composition of automorphisms does not increase the fixed points: We prove that the composition of automorphisms does not increase the fixed points.

Lemma II.4. *Suppose that a graph $G = (E, V)$ has nontrivial automorphisms $\psi_a, \psi_b \in Aut(G)$, where $\psi_a \neq \psi_b$. Let ψ_a have fixed points $V_{fixed,\psi_a} = \{v|\psi_a(v) = v, v \in V\}$. Now, ψ_b has the vertex transposition $\psi_b(v_a) = v_b$ and $\psi_b(v_b) = v_a$, $v_a \in V_{fixed,\psi_a}$. When we apply ψ_b following ψ_a, the set of fixed points becomes $V_{fixed,\psi_a} \cap V_{fixed,\psi_b}$.*

Proof. Suppose $\psi_a : V_{a,s} \mapsto V_{a,d}$, $(V_{a,s} \cup V_{a,d}) \oplus V_{fixed,\psi_a} = V$. When we apply ψ_b following ψ_a, we obtain $\psi_b \circ \psi_a : V_{a,s} \cup V_{fixed,\psi_a} \mapsto V_{a,s} \cup V_{fixed,\psi_a}$, $V_{a,d} \cup V_{fixed,\psi_a} \mapsto V_{a,d} \cup V_{fixed,\psi_a}$, so the vertices belonging to $V_{a,s}$ and $V_{a,d}$ are not returned to the original point by ψ_b. The automorphic transformation ψ_b maps at least one vertex v to another. Thus, Lemma II.4 holds. □

2) Detect whether there is a fixed-point-free automorphism: We prove how to detect whether there is a fixed-point-free automorphism.

Theorem II.5. *Consider a graph $G = (E, V)$. Let the vertex weighted graph $S = G$. We obtain the vertex sets $V_\lambda \subset V$ with the same $\lambda_v = Ev(Sg(S, v, w))$, $v \in V$, $w > 0$. $V_{\lambda_{v_i}} > 1$ for all vertex sets of λ_{v_i} if, and only if, G has a fixed-point-free automorphism.*

Proof. From Lemma II.4, applying composition of automorphic transformations to G does not increase the size of the set of fixed points. Suppose that the set of fixed points V_{fixed} exists after applying the composition of the automorphism $\psi_1 \cdots \psi_i$ to G. We can reduce the size of V_{fixed} by applying an automorphism ψ_{i+1} that contains the transposition of $v \in V_{fixed}$ and another vertex.

When $V_{\lambda_{v_i}} > 1$ for all vertex sets, there exists ψ such that $\psi(v) \neq v$ at every vertex v. On the other hand, suppose there is a set of vertices such that $|V_{\lambda_{v_j}}| = 1$. There is no ψ such that $\psi(v) \neq v$ at $v \in V_{\lambda_{v_j}}$. Then, v becomes a fixed point. □

III. Algorithm

In this section, we present a polynomial-time algorithm to determine whether a graph has a fixed-point-free automorphism. We assume that the number of vertices of the graph is n.

Since the elements of an adjacency matrix of a vertex-weighted graph are all integers, the coefficients of the eigenequation of this matrix are all integers. We use the set of coefficients of the eigenequation of the adjacency matrix of a vertex-weighted graph instead of its set of eigenvalues. We calculate the Frobenius normal form to obtain the set of coefficients without real number calculations. The amount of computation required to convert an adjacency matrix into the Frobenius normal form is $\mathcal{O}(n^4)$.

Function 1 determines whether a graph G has a fixed-point-free automorphism. First, by adding a weight $w > 0$ to a vertex, we obtain a set of vertices V_λ with the same eigenvalue set. Thus, we obtain automorphisms of G from Theorem II.2 and Corollary II.3. Next, we check if the size of the vertex set V_λ is 1 or above to determine whether there is a fixed-point-free automorphism based on Theorem II.5. The computational complexity of this function is $\mathcal{O}(n^5)$.

Figure 1 shows an example of detecting a fixed point for the graph $G = (V, E)$. Let the vertex weighted graph $S = G$. We obtain the vertex sets $V_\lambda \subset V$ with the same $\lambda_v = Ev(Sg(S, v, w))$, $v \in V$, $w > 0$. Then, we obtain $V_{\lambda_1} = \{p_1, p_3\}$, $V_{\lambda_2} = \{p_2, p_4\}$ and $V_{\lambda_3} = \{p_5\}$. Thus, if $|V_{\lambda_3}| = 1$, then G has no fixed-point-free automorphism.

Since $V_{\lambda_1} = \{p_1, p_3\}$, there exists an automorphic transformation ψ_1 that includes the transposition of vertices p_1 and p_3. When we apply ψ_1, vertices p_2, p_4 and p_5 become fixed points. Now, since $V_{\lambda_2} =$

Algorithm 1 A function that determines whether a graph G has a fixed-point-free automorphism.

function HAS_FIXED_POINT_FREE_AUTOMORPHISM($G = (V, E)$)
    $S \leftarrow G$ with all vertex weights are 0
    $w \leftarrow 2|V|$
    clear hash h
    for each $v \in V$ **do**
        $\lambda \leftarrow Ev(Sg(S, v, w))$
        if $h(\lambda) = \emptyset$ **then** $h(\lambda) \leftarrow \{v\}$
        else$h(\lambda) \leftarrow h(\lambda) \cup \{v\}$
        end if
    end for
    for each $T \in h$ **do**
        if $|T| = 1$ **then**
            return FALSE
        end if
    end for
    return TRUE
end function

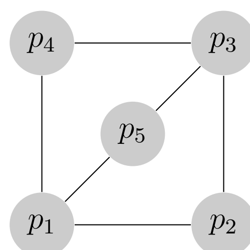

Fig. 1. An example of detecting a fixed point.

$\{p_2, p_4\}$, there exists an automorphic transformation ψ_2 that includes the transposition of vertices p_2 and p_4. Thus, applying ψ_2 after ψ_1 leaves p_5 as a fixed point. Since $V_{\lambda_3} = \{p_5\}$, there is no automorphic transformation that involves the transposition of vertex p_5 and other vertex. Therefore, vertex p_5 remains as a fixed point.

IV. Conclusion

In this paper, we have presented an algorithm to detect whether a given graph G has a fixed-point-free automorphism. It has polynomial time complexity. Since one of the NP-complete problems is solvable in polynomial time, the complexity classes P and NP are the same.

Note that this algorithm has a limitation in that it can only detect whether G has a fixed-point-free automorphism, not obtain a fixed-point-free automorphism even if it exists. When a fixed-point-free automorphism exists, it is unclear whether it can be obtained in polynomial time.

Appendix A
Definition

In this section, we give the definitions used in this paper.

Definition A.1. A graph $G = (V, E)$ is a pair consisting of a non-empty finite vertex set $V \neq \emptyset$ and an edge set E that is a subset of V^2. The graph's size is the number of its vertices $1 < n = |V|$. The number of vertices in a graph is assumed to be finite. In addition, we align the set V with $\{v_1, \ldots, v_n\}$.

There is an edge between vertices v_a and v_b when (v_a, v_b) is an element of the set E. Also, edges have no direction. Moreover, the graph has no multiple edges between a pair of vertices, and there are no loops (i.e., (v_a, v_a) is never an edge).

Definition A.2. A vertex-weighted graph $S = (V, E, w)$ is a graph with a function $w : V \to \mathbb{N}$ that gives the weights of the vertices. Then, a graph is a vertex-weighted graph in which the weights of all its vertices are 0.

Definition A.3. The adjacency matrix A of a vertex-weighted graph $S = (V, E, w)$ with $n = |V|$ is an $n \times n$ symmetric matrix that is given as follows. The entries $a_{i,j}$, $v_i, v_j \in V$, $0 < i, j \le n$ of A satisfy:

$$\begin{cases} (v_i, v_j) \in E & \text{if} \quad a_{i,j} = a_{j,i} = 1, \\ (v_i, v_j) \notin E & \text{if} \quad a_{i,j} = a_{j,i} = 0, \\ a_{i,i} = w(v_i). \end{cases}$$

Definition A.4. Suppose that a graph $G = (E, V)$ has an automorphism. Let ψ be the automorphic transformation. A fixed-point-free automorphism is an automorphism such that $\psi(v) \neq v$ at all vertices $v \in V$.

Appendix B
Proof

This chapter reproduces the proofs from the reference [11].

The following Theorem II.2 and Corollary II.3 [11] prove that it is possible to obtain the automorphisms of S using eigenvalue sets.

Theorem II.2. Let $S_{v_i} = Sg(S, v_i, w)$ and $S_{v_j} = Sg(S, v_j, w)$ with $v_i, v_j \in V_{w0}(S)$, $v_i \neq v_j$ and $w > 0$. If $Ev(S_{v_i}) = Ev(S_{v_j})$, then S_{v_i} and S_{v_j} are isomorphic.

Proof. We show that if $Ev(S_{v_i}) = Ev(S_{v_j})$, then S_{v_i} and S_{v_j} are not cospectral but isomorphic.

Let $A(S_{v_i})$ and $A(S_{v_j})$ be A_{v_i} and A_{v_j}, respectively. When there exists a permutation matrix P such that $A_{v_i} = P^t A_{v_j} P$, S_{v_i} and S_{v_j} are isomorphic. Denote the eigenfunctions of A_{v_i} and A_{v_j} by f_{v_i} and f_{v_j}, respectively. When f_{v_i} and f_{v_j} are the same, the eigenvalue sets of A_{v_i} and A_{v_j} are the same. Therefore, we will prove that such a nontrivial permutation matrix exists when $f_{v_i} - f_{v_j} = 0$.

Without loss of generality, we may assume $i = 1$ and $j = 2$. We show the characteristic polynomials f_{v_1} and f_{v_2} as below.

$$\begin{aligned} f_{v_1} &= |A_{v_1} - \lambda I| \\ &= \begin{vmatrix} w-\lambda & a_{1,2} & a_{1,3} & a_{1,4} & \cdots & a_{1,n} \\ a_{2,1} & -\lambda & a_{2,3} & a_{2,4} & \cdots & a_{2,n} \\ a_{3,1} & a_{3,2} & w_3-\lambda & a_{3,4} & \cdots & a_{3,n} \\ a_{4,1} & a_{4,2} & a_{4,3} & \ddots & & \vdots \\ \vdots & \vdots & \vdots & & \ddots & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & \cdots & w_n-\lambda \end{vmatrix}, \\ f_{v_2} &= |A_{v_2} - \lambda I| \\ &= \begin{vmatrix} -\lambda & a_{1,2} & a_{1,3} & a_{1,4} & \cdots & a_{1,n} \\ a_{2,1} & w-\lambda & a_{2,3} & a_{2,4} & \cdots & a_{2,n} \\ a_{3,1} & a_{3,2} & w_3-\lambda & a_{3,4} & \cdots & a_{3,n} \\ a_{4,1} & a_{4,2} & a_{4,3} & \ddots & & \vdots \\ \vdots & \vdots & \vdots & & \ddots & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & \cdots & w_n-\lambda \end{vmatrix}. \end{aligned}$$

The weights of the vertices are w, $w_3, and \dots w_n$, all of which are integers. Then,

$$f_{v_1} - f_{v_2} = w \begin{vmatrix} 0 & a_{2,3} & a_{2,4} & \cdots & a_{2,n} \\ a_{3,2} & w_3 - \lambda & a_{3,4} & \cdots & a_{3,n} \\ a_{4,2} & a_{4,3} & \ddots & & a_{3,n} \\ \vdots & \vdots & & \ddots & \vdots \\ a_{n,2} & a_{n,3} & \cdots & \cdots & w_n - \lambda \end{vmatrix} - w \begin{vmatrix} 0 & a_{1,3} & a_{1,4} & \cdots & a_{1,n} \\ a_{3,1} & w_3 - \lambda & a_{3,4} & \cdots & a_{3,n} \\ a_{4,1} & a_{4,3} & \ddots & & a_{3,n} \\ \vdots & \vdots & & \ddots & \vdots \\ a_{n,1} & a_{n,3} & \cdots & \cdots & w_n - \lambda \end{vmatrix} \tag{1}$$
$$= 0.$$

If $n = 2$, f_{v_1} and f_{v_2} are the same. Hence, in this case, S_{v_1} and S_{v_2} are isomorphic.

We treat the case of $n = 3$ as follows. Equation 1 becomes

$$\begin{aligned} f_{v_1} - f_{v_2} &= w \begin{vmatrix} 0 & a_{2,3} \\ a_{3,2} & w_3 - \lambda \end{vmatrix} - w \begin{vmatrix} 0 & a_{1,3} \\ a_{3,1} & w_3 - \lambda \end{vmatrix} \\ &= w(a_{2,3}a_{3,2} - a_{1,3}a_{3,1}) \\ &= 0. \end{aligned}$$

So, when $a_{2,3} = a_{1,3}$, f_{v_1} and f_{v_2} are the same. For this case, then, S_{v_1} and S_{v_2} are isomorphic.

Let $n > 3$. Suppose the matrix A' is as follows.

$$A' = \begin{pmatrix} w_3 & a_{3,4} & \cdots & a_{3,n} \\ a_{4,3} & \ddots & & a_{3,n} \\ \vdots & & \ddots & \vdots \\ a_{n,3} & \cdots & \cdots & w_n \end{pmatrix}.$$

Let vertex $u_1 = (a_{1,3}, a_{1,4}, \dots, a_{1,n})^t$ and $u_2 = (a_{2,3}, a_{2,4}, \dots, a_{2,n})^t$. Then, Equation 1 becomes as follows.

$$\begin{aligned} f_{v_1} - f_{v_2} &= w \begin{vmatrix} 0 & u_2^t \\ u_2 & A' - \lambda I \end{vmatrix} - w \begin{vmatrix} 0 & u_1^t \\ u_1 & A' - \lambda I \end{vmatrix} \\ &= 0. \end{aligned}$$

In order for f_{v_1} and f_{v_2} to be the same, it is necessary that $f_{v_1} - f_{v_2} = 0$ for all λ. So, we assume $|A' - \lambda I| \neq 0$. Then,

$$\begin{aligned} f_{v_1} - f_{v_2} \quad &= \quad w|A' - \lambda I||0 - u_2^t(A' - \lambda I)^{-1}u_2| \\ &- \quad w|A' - \lambda I||0 - u_1^t(A' - \lambda I)^{-1}u_1| \\ &= \quad w|A' - \lambda I|(u_2 - u_1)^t(A' - \lambda I)^{-1}(u_2 - u1) \\ &= \quad 0. \end{aligned}$$

When $u_1 = u_2$, f_{v_1} and f_{v_2} are the same. In this case, then, S_{v_1} and S_{v_2} are isomorphic.

Let $u_2 \neq u_1$. When $(u_2 - u_1)^t(A' - \lambda I)^{-1}(u_2 - u_1) = 0$, $u_2 - u_1$ and $(A' - \lambda I)^{-1}(u_2 - u_1)$ are orthogonal. So,

$$\begin{aligned} (u_2 - u_1)^t(A' - \lambda I)(u_2 - u_1) &= u_2^t A' u_2 - u_1^t A' u_1 - u_2^t \lambda I u_2 + u_1^t \lambda I u_1 \\ &= 0. \end{aligned}$$

In order for f_{v_1} and f_{v_2} to be the same, it is necessary that $f_{v_1} - f_{v_2} = 0$ for all λ. So, the number of elements with value 1 in u_2 and u_1 is the same.

Since $u_2 - u_1$ and $(A' - \lambda I)(u_2 - u_1)$ are orthogonal,

$$\begin{aligned} (u_2 - u_1)^t A'(u_2 - u_1) &= (u_2 - u_1)^t P'^t A' P'(u_2 - u_1) \\ &= (u_1 - u_2)^t P'^t A' P'(u_1 - u_2) \\ &= 0 \end{aligned}$$

with P' a liner operator. When A_1 and A_2 have the same eigenvalue set, there exists a set of nontrivial permutation matrices $\{P' | P'^t A' P' = A' \land (u_2 - u_1) = P'(u_1 - u_2))\}$. So, S_{v_1} and S_{v_2} are isomorphic. $\square$

Corollary II.3. Let $S_{v_i} = Sg(S, v_i, w)$ and $S_{v_j} = Sg(S, v_j, w)$ with $v_i, v_j \in V_{w0}(S)$, $v_i \neq v_j$ and $w > 0$. If $Ev(S_{v_i}) \neq Ev(Sv_j)$, then S_{v_i} and S_{v_j} are not isomorphic.

Proof. Using a permutation matrix P, $A(S_{v_i}) \neq P^t A(S_{v_j}) P$. So, there is no bijection between S_{v_i} and S_{v_j}. Therefore, S_{v_i} and S_{v_j} are not isomorphic. $\square$

REFERENCES

[1] S. A. Cook, "The complexity of theorem-proving procedures," in *Logic, Automata, and Computational Complexity: The Works of Stephen A. Cook*. New York and NY and USA: Association for Computing Machinery, 2023, pp. 143–152.

[2] L. Fortnow, "The status of the p versus np problem," *Communications of the ACM*, vol. 52, no. 9, pp. 78–86, 2009.

[3] T. Baker, J. Gill, and R. Solovay, "Relativizations of the p = np question," *SIAM Journal on Computing*, vol. 4, no. 4, pp. 431–442, 1975. [Online]. Available: https://doi.org/10.1137/0204037

[4] A. A. Razborov and S. Rudich, "Natural proofs," in *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*. New York and NY and USA: Association for Computing Machinery, 1994, pp. 204–213.

[5] S. Aaronson and A. Wigderson, "Algebrization: A new barrier in complexity theory," *ACM Transactions on Computation Theory (TOCT)*, vol. 1, no. 1, pp. 1–54, 2009.

[6] J. Gu, P. W. Purdom, J. V. Franco, and B. W. Wah, "Algorithms for the satisfiability (sat) problem: A survey." *Satisfiability problem: Theory and applications*, vol. 35, pp. 19–152, 1996.

[7] G. J. Woeginger, "Exact algorithms for np-hard problems: A survey," in *Combinatorial Optimization—Eureka, You Shrink! Papers Dedicated to Jack Edmonds 5th International Workshop Aussois, France, March 5–9, 2001 Revised Papers*. Springer, 2003, pp. 185–207.

[8] P. Prosser, "Exact algorithms for maximum clique: A computational study," *Algorithms*, vol. 5, no. 4, pp. 545–587, 2012.

[9] M. R. Garey, D. S. Johnson, and L. Stockmeyer, "Some simplified np-complete problems," in *Proceedings of the sixth annual ACM symposium on Theory of computing*. New York and NY and USA: Association for Computing Machinery, 1974, pp. 47–63.

[10] A. Lubiw, "Some np-complete problems similar to graph isomorphism," *SIAM Journal on Computing*, vol. 10, no. 1, pp. 11–21, 1981.

[11] Y. Ohto, "Solving graph isomorphism problem in polynomial time," *FOCS*, submitted.

[12] J. A. Howell, "An algorithm for the exact reduction of a matrix to Frobenius form using modular arithmetic. I," *Mathematics of Computation*, vol. 27, no. 124, pp. 887–904, 1973.

[13] ——, "An algorithm for the exact reduction of a matrix to Frobenius form using modular arithmetic. II," *Mathematics of Computation*, vol. 27, no. 124, pp. 905–920, 1973.