Solving graph isomorphism problem in polynomial time

Yasunori Ohto

Abstract

We show that the graph isomorphism problem is solvable in polynomial time. First, we define the following functions. Let S be 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 weight w is given to vertex v of S. Let $Ev(S)$ be the set of eigenvalues of the adjacency matrix of S. Next, we prove the following to obtain the automorphisms of S using eigenvalue sets. 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 construct an algorithm to determine whether two given graphs G_a and G_b are isomorphic using this result. Write n for the number of vertices of these graphs. Let $S_{a_0} = G_a$ and $S_{b_0} = G_b$. Consider a vertex weight $w_i > 0 \notin \{w_j | 0 \leq j < i\}$. Let $S_{a_{i+1}}$ be $Sg(S_{a_i}, v_{a_i}, w_i)$ with $v_{a_i} \in V_{w0}(S_{a_i})$. Let $S_{b_{i+1}}$ be $Sg(S_{b_i}, v_{b_i}, w_i)$ with $v_{b_i} \in V_{w0}(S_{b_i})$. Let $Ev(S_{a_{i+1}}) = Ev(S_{a_{j+1}})$. Then, we check the vertex mapping $\{v_{a_i} \mapsto v_{b_i} | 0 \leq i < n\}$ to determine whether G_a and G_b are isomorphic. The computational complexity to detect whether the two graphs are isomorphic is $\mathcal{O}(n^6)$.

Index Terms

graph isomorphism problem, graph spectrum, polynomial time computation.

I. Introduction

The graph isomorphism problem [1] is to determine whether two given graphs are isomorphic. This problem is one of the major problems in theoretical computer science, especially regarding the class of its computational complexity [2]. There are practical algorithms that can determine whether two graphs are isomorphic [3], [4], [5]. These methods can obtain correct results at a practical level. On the theoretical side, a quasi-polynomial algorithm has been proposed [6], [7]. On the other hand, polynomial-time isomorphism detection algorithms exist for special graphs [8], [9], [10].

The set of the eigenvalues of the adjacency matrix of a graph indicates the characteristics of the graph. However, if the two sets of eigenvalues are the same, such graphs are called cospectral graphs[11], the graphs might not be isomorphic. Therefore, we cannot determine the isomorphism of two graphs by whether their eigenvalue sets are only the same. When there two sets are the same, if the multiplicities of the eigenvalues are all 1, we can determine whether the graphs are isomorphic in $\mathcal{O}(n^3)$ time [12]. However, it is unclear whether a polynomial-time algorithm exists for general graphs.

In this paper, we show that the graph isomorphism problem is solvable in polynomial time. First, we define the following functions. Let S be 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 weight $w \in \mathbb{N}$ is given to vertex v of S. Let $Ev(S)$ be the set of eigenvalues of the adjacency matrix of S. Next, we prove the following Theorem II.1 to obtain the automorphisms of S using eigenvalue sets.

Theorem II.1. 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 construct an algorithm to determine whether two given graphs G_a and G_b are isomorphic using this result. Write n for the number of vertices of these graphs. Let $S_{a_0} = G_a$ and $S_{b_0} = G_b$. Consider a vertex weight $w_i > 0 \notin \{w_j | 0 \leq j < i\}$. Let $S_{a_{i+1}}$ be $Sg(S_{a_i}, v_{a_i}, w_i)$ with $v_{a_i} \in V_{w0}(S_{a_i})$. Let $S_{b_{i+1}}$ be $Sg(S_{b_i}, v_{b_i}, w_i)$ with $v_{b_i} \in V_{w0}(S_{b_i})$. Let $Ev(S_{a_{i+1}}) = Ev(S_{b_{i+1}})$. Then, we check the vertex mapping $\{v_{a_i} \mapsto v_{b_i} | 0 \leq i < n\}$ to determine whether G_a and G_b are isomorphic. 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 [13], [14] 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 the two graphs are isomorphic is $\mathcal{O}(n^6)$.

This paper is organized as follows. Section II provides the proofs used to determine whether two graphs are isomorphic. 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 about the determination of whether two given graphs are isomorphic.

A. Preparation

We define the following functions, which will be used in the proofs and the methods. Suppose S be 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

The following Theorem II.1 and Corollary II.2 prove that it is possible to obtain the automorphisms of S using eigenvalue sets.

Theorem II.1. *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,

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

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,

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

Algorithm 1 A function to obtain a vertex v such that $Ev(Sg(S, v, w)) = \lambda$ with $v \in V_{w0}(S)$.

function OBTAIN_VERTEX(S, λ, w)
    for each $v \in V_{w0}(S)$ **do**
        if $Ev(Sg(S, v, w)) = \lambda$ **then**
            return v
        end if
    end for
    return null
end function

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

Corollary 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 $Ev(S_{v_i}) \neq Ev(Sv_j)$, 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. $\square$

C. Obtain vertex mapping and detect whether two graphs are isomorphic

The following Corollaries II.3 and II.4 prove that it is possible to obtain vertex mapping and detect whether two graphs are isomorphic.

Corollary II.3. *Suppose S_{a_i} and S_{b_i} are vertex-weighted graphs. Let $Ev(S_{a_i}) = Ev(S_{b_i})$. Let $v_{a_i} \in V_{w0}(S_{a_i})$ and $w > 0$. When $Ev(Sg(S_{a_i}, v_{a_i}, w)) \neq Ev(Sg(S_{b_i}, v_{b_i}, w))$ for any $v_{b_i} \in V_{w0}(S_{b_i})$, S_{a_i} and S_{b_i} are not isomorphic.*

Proof. Using a permutation matrix P, $A(Sg(S_{a_i}, v_{a_i}, w)) \neq P^t A(Sg(S_{b_i}, v_{b_i}, w))P$ for any $v_{a_i} \in V_{w0}(S_{a_i})$ and $v_{b_i} \in V_{w0}(S_{b_i})$. So, there is no bijection between $Sg(S_{a_i}, v_{a_i}, w))$ and $Sg(S_{b_i}, v_{b_i}, w))$. Therefore, S_{a_i} and S_{b_i} are not isomorphic. $\square$

Corollary II.4. *Suppose S_{a_i} and S_{b_i} are vertex-weighted graphs. Let $Ev(S_{a_i}) = Ev(S_{b_i})$ and $w > 0$. Let $S_{a_{i+1}} = Sg(S_{a_i}, v_{a_i}, w)$ with $v_{a_i} \in V_{w0}(S_{a_i})$. Let $S_{b_{i+1}} = Sg(S_{b_i}, v_{b_i}, w)$ with $v_{b_i} \in V_{w0}(S_{b_i})$. Let $Ev(S_{a_{i+1}}) = Ev(S_{b_{i+1}})$. Now, $S_{a_{i+1}}$ and $S_{b_{i+1}}$ are isomorphic if, and only if, S_{a_i} and S_{b_i} are isomorphic.*

Proof. Let V_a be the set of vertices of $S_{a_{i+1}}$ and V_b that of $S_{b_{i+1}}$. When $S_{a_{i+1}}$ and $S_{b_{i+1}}$ are not isomorphic, there if no bijection $V_a \to V_b$.

When $S_{a_{i+1}}$ and $S_{b_{i+1}}$ are isomorphic, there exists a bijection $V_a \to V_b$. From Theorem II.1 and Corollaries II.2, for any v_{a_i} and v_{b_i} such that $Ev(S_{a_{i+1}}) = Ev(S_{b_{i+1}})$, S_{a_i} and S_{b_i} are isomorphic. $\square$

III. Algorithm

In this section, we present a polynomial-time algorithm to determine whether two graphs G_a and G_b are isomorphic. We assume that the number of vertices of the graphs 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 calculate the Frobenius normal form to obtain the coefficients of the eigenequation of the adjacency matrix of a vertex-weighted graph without real number calculations. Then, we compare the coefficients to determine whether the eigenvalue sets are the same. The amount of computation required to convert an adjacency matrix into the Frobenius normal form is $\mathcal{O}(n^4)$. The amount of computation to compare the coefficients of the two characteristic equations is $\mathcal{O}(n)$.

We show a function 1 to obtain a vertex v such that $Ev(Sg(S, v, w)) = \lambda$ with $v \in V_{w0}(S)$. So, the amount of computation of this function is $\mathcal{O}(n^5)$.

Algorithm 2 A function that determines whether a map $v_a \mapsto v_b$ exists for the set of vertex pair (v_a, v_b) of vertices $v_a \in V_a$ and $v_b \in V_b$ of two graphs $G_a = (V_a, E_a)$ and $G_b = (V_b, E_b)$.

```
function TEST_ISOMORPHIC(G_a = (V_a, E_a), G_b = (V_b, E_b), h)
    for i ← 1 to |V_a| − 1 do
        v_i ← i-th vertex in V_a
        for j ← i + 1 to |V_a| do
            v_j ← j-th vertex in V_a
            if (v_i, v_j) ∈ E_a then
                if (h(v_i), h(v_j)) ∉ E_b then
                    return FALSE
                end if
            else
                if (h(v_i), h(v_j)) ∈ E_b then
                    return FALSE
                end if
            end if
        end for
    end for
    return TRUE
end function
```

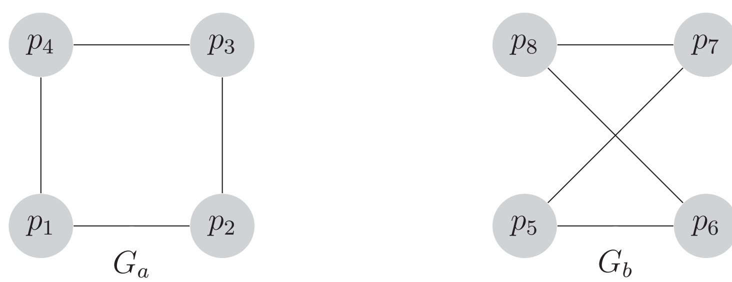

Fig. 1. Graphs G_a and G_b as an example to determine graph isomorphism.

Function 2 determines whether a map $v_a \mapsto v_b$ exists for the set of vertex pair (v_a, v_b) of vertices $v_a \in V_a$ and $v_b \in V_b$ of two graphs $G_a = (V_a, E_a)$ and $G_b = (V_b, E_b)$. The amount of computation of this function is $\mathcal{O}(n^2)$.

Function 3 determines whether two graphs G_a and G_b are isomorphic. From Theorem II.1 and Corollaries II.2, II.3, II.4, this function can detect whether two graphs are isomorphic. The amount of computation of this function is $\mathcal{O}(n^6)$.

Figure 1 shows that graphs $G_a = (V_a, E_a)$ and $G_b = (V_b, E_b)$ as an example to determine graph isomorphism. At first, we clear hash h. Let $S_{a_0} = G_a$ and $S_{b_0} = G_b$. So, all vertices in S_{a_0} and S_{b_0} have weights of 0. Next, let $w_0 = 2|V_a|$. Let $S_{a_1} = Sg(S_{a_0}, p_1, w_0)$. Obtain the vertex $v_0 \in V_{w0}(S_{b_0})$ such that $Ev(Sg(S_{b_0}, v_0, w_0)) = Ev(S_{a_1})$. Vertices $p_5, \ldots, p_8$ satisfy this condition. Since Theorem II.1 and Corollary II.2, we can select a vertex from any of then. So, we select p_5, then, let $S_{b_1} = Sg(S_{b_0}, p_5, w_0))$ and $h(p_1) = p_5$. Next, let $w_1 = w_0 + 2|V_a|$. Let $S_{a_2} = Sg(S_{a_1}, p_2, w_1)$. Obtain the vertex $v_1 \in V_{w0}(S_{b_1})$ such that $Ev(Sg(S_{b_1}, v_1, w_1)) = Ev(S_{a_2})$. Vertices p_6 and p_8 satisfy this condition. We can select a vertex from any of then. So, we select p_6, then, let $S_{b_2} = Sg(S_{b_1}, p_6, w_1))$ and $h(p_2) = p_6$. Next, let $w_2 = w_1 + 2|V_a|$. Let $S_{a_3} = Sg(S_{a_2}, p_3, w_2)$. Obtain the vertex $v_2 \in V_{w0}(S_{b_2})$ such that $Ev(Sg(S_{b_2}, v_2, w_2)) = Ev(S_{a_3})$. Vertex p_8 satisfies this condition. So, we use p_8, then, let $S_{b_3} = Sg(S_{b_2}, p_8, w_2)$ and $h(p_3) = p_8$. Next, let $w_3 = w_2 + 2|V_a|$. Let $S_{a_4} = Sg(S_{a_3}, p_4, w_3)$. Obtain the vertex $v_3 \in V_{w0}(S_{b_3})$ such that $Ev(Sg(S_{b_3}, v_3, w_3)) =$

Algorithm 3 A function that determines whether two graphs G_a and G_b are isomorphic.

function IS_ISOMORPHIC($G_a = (V_a, E_a)$, $G_b = (V_b, E_b)$)
    $S_a \leftarrow G_a$ with all vertex weights equal to 0
    $S_b \leftarrow G_b$ with all vertex weights equal to 0
    if $Ev(S_a) \neq Ev(S_b)$ **then**
        return FALSE
    end if
    $w \leftarrow 2|V_a|$
    clear hash h
    while $V_{w0}(S_a) \neq \emptyset$ **do**
        $v_a \leftarrow$ one vertex in $V_{w0}(S_a)$
        $\lambda \leftarrow Ev(Sg(S_a, v_a, w))$
        $v_b \leftarrow$ OBTAIN_VERTEX(S_b, λ, w)
        if $v_b =$ null **then**
            return FALSE
        end if
        $h(v_a) \leftarrow v_b$
        $w \leftarrow w + 2|V_a|$
        $S_a \leftarrow Sg(S_a, v_a, w)$
        $S_b \leftarrow Sg(S_b, v_b, w)$
    end while
    return TEST_ISOMORPHIC(G_a, G_b, h)
end function

$Ev(S_{a_4})$. Vertex p_7 satisfies this condition. So, we use p_7, then, let $S_{b_4} = Sg(S_{b_3}, p_7, w_3))$ and let $h(p_4) = p_7$. Finally, using the stack h that stored the map between vertices of G_a and G_b, we check whether a bijection exists between G_a and G_b.

IV. Conclusion

In this paper, we have presented an algorithm to detect whether two given graphs are isomorphic. It has polynomial time complexity. Note that this algorithm has a limitation in that it can only obtain one of the isomorphisms.

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 $a_{i,j}$, $v_i, v_j \in V$, $0 < i, j \leq n$ 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. An isomorphism of graphs $G_a = (V_a, E_a)$ and $G_b = (V_b, E_b)$ is a bijection between the vertex sets of G_a and G_b $f : V_b \to V_a$ such that two vertices v_i and v_j of G_a are adjacent in G_a if and only if $f(v_a)$ and $f(v_b)$ are adjacent in G_b.

References

[1] M. Grohe and P. Schweitzer, "The graph isomorphism problem." *Commun. ACM*, vol. 63, no. 11, pp. 128–134, 2020. [Online]. Available: http://dblp.uni-trier.de/db/journals/cacm/cacm63.html\#GroheS20

[2] S. Fortin, "The graph isomorphism problem," Edmonton, Alberta, Canada, Tech. Rep., 1996. [Online]. Available: https://api.semanticscholar.org/CorpusID:6986595

[3] B. McKay, "Practical graph isomorphism," *Congressus Numerantium*, vol. 30, pp. 45–87, 1981.

[4] B. D. McKay and A. Piperno, "Practical graph isomorphism, ii," *Journal of Symbolic Computation*, vol. 60, no. 0, pp. 94 – 112, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0747717113001193

[5] D. C. Schmidt and L. E. Druffel, "A fast backtracking algorithm to test directed graphs for isomorphism using distance matrices." *J. ACM*, vol. 23, no. 3, pp. 433–445, 1976. [Online]. Available: http://dblp.uni-trier.de/db/journals/jacm/jacm23.html\#SchmidtD76

[6] L. Babai, "Graph isomorphism in quasipolynomial time." *CoRR*, vol. abs/1512.03547, 2015. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1512.html\#Babai15

[7] ——, "Graph isomorphism in quasipolynomial time [extended abstract]." in *STOC*, D. Wichs and Y. Mansour, Eds. ACM, 2016, pp. 684–697. [Online]. Available: http://dblp.uni-trier.de/db/conf/stoc/stoc2016.html\#Babai16

[8] J. E. Hopcroft and J. K. Wong, "Linear time algorithm for isomorphism of planar graphs (preliminary report)," in *STOC*, R. L. Constable, R. W. Ritchie, J. W. Carlyle, and M. A. Harrison, Eds. ACM, 1974, pp. 172–184. [Online]. Available: http://dblp.uni-trier.de/db/conf/stoc/stoc74.html\#HopcroftW74

[9] G. S. Lueker and K. S. Booth, "A linear time algorithm for deciding interval graph isomorphism." *J. ACM*, vol. 26, no. 2, pp. 183–195, 1979. [Online]. Available: http://dblp.uni-trier.de/db/journals/jacm/jacm26.html\#LuekerB79

[10] M. Muzychuk, "A solution of the isomorphism problem for circulant graphs," *Proceedings of the London Mathematical Society*, vol. 88, no. 1, pp. 1–41, 2004.

[11] F. Harary, C. King, A. Mowshowitz, and R. C. Read, "Cospectral graphs and digraphs," *Bulletin of the London Mathematical Society*, vol. 3, no. 3, pp. 321–328, 1971.

[12] F. T. Leighton and G. l. Miller, "Certificates for graphs with distinct eigen values," 1979, orginal Manuscript.

[13] 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.

[14] ——, "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.