A Polynomial Time Algorithm for Undirected Graph Isomorphism

Aimin Hou, Zhifeng Hao

***Abstract*—Graph isomorphism is an important problem in graph theory. So far no polynomial time algorithms have been found for undirected graph isomorphism. A popular class of testing methods use necessary conditions to identify two non-isomorphic graphs. Unfortunately, there often exist some non-isomorphic graphs that satisfy these necessary conditions. Based on a necessary condition we proposed previously, this paper develops an $O(n^4)$ algorithm for undirected graph isomorphism using vertex partition and refinement. Also, our algorithm takes advantage of a recursive property that isomorphism of supergraphs will result in the isomorphism of subgraphs. Finally, the experiments on the Graph Database validated the correctness of this algorithm for graph isomorphism.**

***Index Terms*—necessary condition, supergraph isomorphism, subgraph isomorphism, algorithm, polynomial time complexity**

I. INTRODUCTION

GRAPH isomorphism is an important problem in graph theory. So far no polynomial time algorithms [1] have been found for undirected graph isomorphism. While there exist a few linear average-case time complexity algorithms (in the number of vertices of the graphs), the best algorithms known for this problem have exponential worst-case time complexity. In 1980, Babai et al. [2] used the degree results for the random graph to develop a fast algorithm which appeared to always work. Unfortunately, this canonical labeling algorithm can not work for those graphs which cannot be canonically labeled and has a rejection probability bounded by $n^{-1/7}$. Later some improved algorithms [3] were found but all of them had a non-zero rejection probability. It means that these improved algorithms can not work for all kinds of graphs.

The vertex partition is a useful method for graph isomorphism. For example, two popular methods of vertex partition use the degrees of the vertices and the degree sequence of the vertex neighborhood respectively. If the vertex partition can be refined iteratively with heuristics, the size of candidate isomorphism space will be reduced effectively. In fact, this idea is the basic strategy of some isomorphism algorithms [4-8]. Unfortunately, these algorithms may not refine sufficiently and need some backtracking operations so as to construct a search tree for testing and pruning. Therefore, they are not polynomial time algorithms in the worst case.

In our previous work [9] for necessary conditions for graph isomorphism, we introduced some new notions including row code XOR (i.e. exclusive-OR) distance, row code AOR (i.e. exclusive-NOR) distance, matrix with the entry of row code XOR distance, matrix with the entry of row code AOR distance, and row-row mapping between matrices. Based on those notions, we proposed a simple necessary condition: for two isomorphic graphs, there must exist an identical row-row mapping between their adjacency matrices, their row code XOR distance matrices, and their row code AOR distance matrices.

The vertex partition and refinement can be applied using this necessary condition. That is, the vertices belong to the same vertex partition set if they are the elements of the match set of a respective vertex. Besides, we observe a recursive property in undirected graphs: the isomorphism of supergraphs must result in the isomorphism of subgraphs. Thus the refinement operations can be iterated based on this recursive property. Also, the necessary condition we proposed previously can be exploited recursively for a sequence of subgraphs until the subgraphs of two vertices are handled. Then we can determine if the two original graphs are isomorphic or not according to the row-column elementary operations on the adjacency matrices based on the identical row-row mapping.

The rest of the paper is organized as follows. Section II introduces the necessary preliminaries. Section III presents the detailed algorithm with polynomial time complexity of $O(n^4)$. Section IV reports the test results on the Graph Database [10]. Finally, Section V concludes this paper.

II. PRELIMINARIES

We will present a necessary condition for undirected graph isomorphism introduced in [9] firstly.

Definition 1. Suppose that $\boldsymbol{G}=(\boldsymbol{V}, \boldsymbol{E})$ is an undirected pseudograph with parallel edges and loops where $|\boldsymbol{V}|=n$. The vertices of $\boldsymbol{G}$ are listed arbitrarily as $u_1, u_2, \cdots, u_n$. The adjacency matrix of $\boldsymbol{G}$ is $\boldsymbol{A_G}=(a_{ij})_{n\times n}$. Let $aa_{ij}=1$ if $a_{ij}\neq 0$, otherwise $aa_{ij}=0$. Then the matrix $\boldsymbol{AA_G}=(aa_{ij})_{n\times n}$ is called the zero-one matrix of adjacency matrix $\boldsymbol{A_G}$ of $\boldsymbol{G}$.

By **Definition 1**, we know that if the graph is an undirected

Manuscript received March 02, 2011; revised April 03, 2011. This work was supported by the Natural Science Foundation of Guangdong Province, P.R.C. (9251009001000005) and the Science and Technology Program of Guangdong Province, P.R.C. (2008B080701005).

Aimin Hou was with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, Guangzhou, P.R.C. He is now with the School of Computer, Dongguan University of Technology, Dongguan 523808, Guangdong, P.R.C. (phone: 86-13538377208; fax: 86-0769-22862362; e-mail: zhham@163.com).

Zhifeng Hao is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, Guangzhou, P.R.C. (e-mail: mazfhao@scut.edu.cn).

simple graph, then $\boldsymbol{AA_G}=\boldsymbol{A_G}$.

Definition 2. Given an undirected pseudograph $\boldsymbol{G}=(\boldsymbol{V}, \boldsymbol{E})$ and its adjacency matrix $\boldsymbol{A_G}$ of $\boldsymbol{G}$, a row code XOR distance between row i and row j, denoted by xord(i, j), is defined by the sum of entries which are exclusive of each other (i.e., one entry being 0 and the other being 1 or k) in the same columns of row i and row j of adjacency matrix $\boldsymbol{A_G}$.

Suppose that the adjacency matrix of $\boldsymbol{G}$ is $\boldsymbol{A_G}=(a_{ij})_{n\times n}$ and the zero-one matrix of adjacency matrix of $\boldsymbol{G}$ is $\boldsymbol{AA_G}=(aa_{ij})_{n\times n}$. We have that

$$\text{xord}(i,j)=(a_{i1}+a_{j1})\times(aa_{i1}\oplus aa_{j1})+(a_{i2}+a_{j2})\times(aa_{i2}\oplus aa_{j2}) +\ldots+(a_{in}+a_{jn})\times(aa_{in}\oplus aa_{jn}) \quad (1)$$

where + is add operation, × multiple operation, and ⊕ exclusive-OR operation (i.e., XOR).

We have that 0⊕0=0, 0⊕1=1, 1⊕0=1, 1⊕1=0.

Definition 3. Given an undirected pseudograph $\boldsymbol{G}=(\boldsymbol{V}, \boldsymbol{E})$ and its adjacency matrix $\boldsymbol{A_G}$ of $\boldsymbol{G}$, a row code AOR distance between row i and row j, denoted by aord(i, j), is defined by the sum of entries which are not exclusive of each other (i.e., both entries being 0 or both entries being 1 or k) in the same columns of row i and row j of adjacency matrix $\boldsymbol{A_G}$.

Suppose that the adjacency matrix of $\boldsymbol{G}$ is $\boldsymbol{A_G}=(a_{ij})_{n\times n}$ and the zero-one matrix of adjacency matrix of $\boldsymbol{G}$ is $\boldsymbol{AA_G}=(aa_{ij})_{n\times n}$. We have that

$$\text{aord}(i,j)=(a_{i1}+a_{j1})\times(aa_{i1}\otimes aa_{j1})+(a_{i2}+a_{j2})\times(aa_{i2}\otimes aa_{j2}) +\ldots+(a_{in}+a_{jn})\times(aa_{in}\otimes aa_{jn}) \quad (2)$$

where + is add operation, × multiple operation, and ⊗ exclusive-NOR operation (i.e., AOR).

We have that 0⊗0=1, 0⊗1=0, 1⊗0=0, 1⊗1=1.

Definition 4. Suppose that $\boldsymbol{G}=(\boldsymbol{V}, \boldsymbol{E})$ is an undirected pseudograph where $|\boldsymbol{V}|$=n and that the vertices of $\boldsymbol{G}$ are listed arbitrarily as $u_1, u_2, \ldots, u_n$. $\boldsymbol{A_G}$ is the adjacency matrix of $\boldsymbol{G}$. The row code XOR distance matrix, denoted by $\boldsymbol{BY_G}$, of $\boldsymbol{G}$ is $\boldsymbol{BY_G}=(by_{ij})_{n\times n}$ where by_{ij}=xord(i, j) if i ≠ j, otherwise $by_{ij}=a_{ii}\in\boldsymbol{A_G}$.

Definition 5. Suppose that $\boldsymbol{G}=(\boldsymbol{V}, \boldsymbol{E})$ is an undirected pseudograph where $|\boldsymbol{V}|$=n and that the vertices of $\boldsymbol{G}$ are listed arbitrarily as $u_1, u_2, \ldots, u_n$. The row code AOR distance matrix, denoted by $\boldsymbol{BT_G}$, of $\boldsymbol{G}$ is $\boldsymbol{BT_G}=(bt_{ij})_{n\times n}$ where bt_{ij}=aord(i, j) for all i and j.

Definition 6. The row-column elementary operation on an n×n matrix is defined by the elementary operation of interchanging simultaneously row i and row j as well as column i and column j.

Theorem 1. Suppose that $\boldsymbol{G}=(\boldsymbol{V}, \boldsymbol{E})$ is an undirected pseudograph. It holds that the row-column elementary operation on the adjacency matrix of $\boldsymbol{G}$ does not modify any entry of both the row code XOR distance matrix and the row code AOR distance matrix, and only performs the same operation on these two matrices.

Theorem 2. Suppose that $\boldsymbol{G}=(\boldsymbol{V_G}, \boldsymbol{E_G})$ and $\boldsymbol{H}=(\boldsymbol{V_H}, \boldsymbol{E_H})$ are two undirected pseudographs. If $\boldsymbol{G}\cong\boldsymbol{H}$, then after performing a sequence of row-column elementary operations, the adjacency matrices of $\boldsymbol{G}$ and $\boldsymbol{H}$ are the same, so are the row code XOR distance matrices of $\boldsymbol{G}$ and $\boldsymbol{H}$ and so are the row code AOR distance matrices of $\boldsymbol{G}$ and $\boldsymbol{H}$.

Definition 7. Suppose that $\boldsymbol{A}=(a_{ij})_{n\times n}$ and $\boldsymbol{B}=(b_{ij})_{n\times n}$ are matrices and the labeling of the rows of $\boldsymbol{A}$ and $\boldsymbol{B}$ are listed as $u_1, u_2, \ldots, u_n$ and $v_1, v_2, \ldots, v_n$ respectively. If there exists a correspondence $[u_1\leftrightarrow v_1', u_2\leftrightarrow v_2', \ldots, u_n\leftrightarrow v_n']$, where $(v_1', v_2', \ldots, v_n')$ is one of the permutations of $(v_1, v_2, \ldots, v_n)$, such that all entries of row u_i of $\boldsymbol{A}$ and all entries of row v_i' of $\boldsymbol{B}$ are identical (without distinguishing the ordering list of entries), then the correspondence $[u_1\leftrightarrow v_1', u_2\leftrightarrow v_2', \ldots, u_n\leftrightarrow v_n']$ is called a row-row mapping between $\boldsymbol{A}$ and $\boldsymbol{B}$.

Theorem 3. Suppose that $\boldsymbol{G}=(\boldsymbol{V_G}, \boldsymbol{E_G})$ and $\boldsymbol{H}=(\boldsymbol{V_H}, \boldsymbol{E_H})$ are two undirected pseudographs. If $\boldsymbol{G}\cong\boldsymbol{H}$, then there must exist an identical row-row mapping between the adjacency matrix, the row code XOR distance matrix, the row code AOR distance matrix of $\boldsymbol{G}$ and the adjacency matrix, the row code XOR distance matrix, the row code AOR distance matrix of $\boldsymbol{H}$, respectively.

Theorem 4. Suppose that $\boldsymbol{G}=(\boldsymbol{V_G}, \boldsymbol{E_G})$ and $\boldsymbol{H}=(\boldsymbol{V_H}, \boldsymbol{E_H})$ are two undirected pseudographs. If $\boldsymbol{G}\cong\boldsymbol{H}$, then there must exist a subgraph $\boldsymbol{G_i}$ of $\boldsymbol{G}$ and a subgraph $\boldsymbol{H_i}$ of $\boldsymbol{H}$ such that the two subgraphs $\boldsymbol{G_i}$ and $\boldsymbol{H_i}$ are isomorphic. For any two isomorphic subgraphs $\boldsymbol{G_i}$ and $\boldsymbol{H_i}$, there must exist an identical row-row mapping between the adjacency matrix, the row code XOR distance matrix, the row code AOR distance matrix of $\boldsymbol{G_i}$ and the adjacency matrix, the row code XOR distance matrix, the row code AOR distance matrix of $\boldsymbol{H_i}$, respectively. This relationship holds until the subgraphs have two vertices.

III. ISOMORPHIC ALGORITHM

In this section, we first describe the general strategy employed for our algorithm. Next, we will present the three detailed parts, which together make up an $O(n^4)$ algorithm for graph isomorphism.

Since in undirected graphs the recursive property holds that the isomorphism of supergraphs must result in the isomorphism of subgraphs, the necessary condition we proposed previously can be exploited recursively for a sequence of subgraphs until the subgraphs of two vertices are handled. For two graphs $\boldsymbol{G}$ and $\boldsymbol{H}$, when this iterative process fails in determining two subgraphs of two vertices being not non-isomorphic, we will obtain an identical row-row mapping between a pair of adjacency matrices, a pair of row code XOR distance matrices, a pair of row code AOR distance matrices of $\boldsymbol{G}$ and $\boldsymbol{H}$ as well as their all subgraphs. We can determine if the two original graphs are isomorphic or not according to the row-column elementary operations on the adjacency matrices based on the identical row-row mapping. If they are isomorphic, the identical row-row mapping is an isomorphism. The following is the detailed algorithm.

A. Row-row Mapping

Algorithm Ⅰ: ComputeRRM

Input: $\boldsymbol{A_G}$— an adjacency matrix of graph $\boldsymbol{G}$
$\boldsymbol{A_H}$— an adjacency matrix of graph $\boldsymbol{H}$
n— the number of vertices of graph $\boldsymbol{G}$

Output: $\boldsymbol{BY_G}$—a row code XOR distance matrix of graph $\boldsymbol{G}$
$\boldsymbol{BT_G}$—a row code AOR distance matrix of graph $\boldsymbol{G}$
$\boldsymbol{BY_H}$—a row code XOR distance matrix of graph $\boldsymbol{H}$
$\boldsymbol{BT_H}$—a row code AOR distance matrix of graph $\boldsymbol{H}$
$\boldsymbol{MS}$ — a vertex match set
$\boldsymbol{M}$ — a row-row mapping

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Step 1: By **Definition 1**, we can computer the zero-one matrix $\boldsymbol{AA_G}$ of adjacency matrix $\boldsymbol{A_G}$ of $\boldsymbol{G}$ and the zero-one matrix $\boldsymbol{AA_H}$ of adjacency matrix $\boldsymbol{A_H}$ of $\boldsymbol{H}$, respectively.

Step 2: By **formula (1)** and **(2)**, **Definition 4** and **5**, we can obtain the row code XOR distance matrix $\boldsymbol{BY_G}$ and the row code AOR distance matrix $\boldsymbol{BT_G}$ of graph $\boldsymbol{G}$, respectively.

Step 3: By **formula (1)** and **(2)**, **Definition 4** and **5**, we can obtain the row code XOR distance matrix $\boldsymbol{BY_H}$ and the row code AOR distance matrix $\boldsymbol{BT_H}$ of graph $\boldsymbol{H}$, respectively.

Step 4: We compare every row, labeled by u_i for $1\le i\le n$, of $\boldsymbol{BY_G}$ with some rows of $\boldsymbol{BY_H}$ to seek a possible match, that is, all the entries of the row, labeled by u_i, of $\boldsymbol{BY_G}$ and all the entries of some corresponding row of $\boldsymbol{BY_H}$ are identical (irrespective of the ordering list of the entries). A match set of u_i is the set consisting of all the labels of the corresponding rows of $\boldsymbol{BY_H}$, denoted by S_u_i.

Step 5: We examine whether these match sets S_u_i for $1\le i\le n$ still hold between a pair of $\boldsymbol{A_G}$ and $\boldsymbol{A_H}$, as well as between a pair of $\boldsymbol{BT_G}$ and $\boldsymbol{BT_H}$.

Step 6: If it is true, then the match set S_u_i of u_i is updated by the intersection set of the S_u_i obtained in this iteration and the S_u_i obtained in previous iteration for $1\le i\le n$. The vertex match set $\boldsymbol{MS}$ is the set consisting of all the S_u_i for $1\le i\le n$. If it is false, then let the row-row mapping $\boldsymbol{M}$ be an empty set. Furthermore, if, after the intersection operation, some S_u_i is an empty set, then let the row-row mapping $\boldsymbol{M}$ be an empty set, too. Otherwise, an one-to-one correspondence between the vertex set of graph $\boldsymbol{G}$ and the vertex set of graph $\boldsymbol{H}$ constitutes the row-row mapping $\boldsymbol{M}$=[$u_1\leftrightarrow v_1'$, $u_2\leftrightarrow v_2'$,…, $u_n\leftrightarrow v_n'$], where $v_i'\in S\_u_i$ for $1\le i\le n$. If such an one-to-one correspondence does not exist, then the graph $\boldsymbol{G}$ and $\boldsymbol{H}$ are non-isomorphic and let the row-row mapping $\boldsymbol{M}$ be an empty set.

Space bound: $O(n^2)$.

Running time: $O(n^3)$.

B. Row-row Mapping of Subgraphs

Algorithm Ⅱ: DetermineSubgraphRRM

Input: $\boldsymbol{A_G}$— an adjacency matrix of graph $\boldsymbol{G}$
$\boldsymbol{A_H}$— an adjacency matrix of graph $\boldsymbol{H}$
n— the number of vertices of graph $\boldsymbol{G}$

Output: $\boldsymbol{BY_G}$—a row code XOR distance matrix of graph $\boldsymbol{G}$
$\boldsymbol{BT_G}$—a row code AOR distance matrix of graph $\boldsymbol{G}$
$\boldsymbol{BY_H}$—a row code XOR distance matrix of graph $\boldsymbol{H}$
$\boldsymbol{BT_H}$—a row code AOR distance matrix of graph $\boldsymbol{H}$
$\boldsymbol{MS}$ — a vertex match set
$\boldsymbol{M}$ — a row-row mapping

Step 1: We make some initialization as follows. Let the graph $\boldsymbol{SG}$ be the original graph $\boldsymbol{G}$ and the graph $\boldsymbol{SH}$ be the original graph $\boldsymbol{H}$. Let the matrix $\boldsymbol{A_{SG}}$ be the adjacency matrix $\boldsymbol{A_G}$ and the matrix $\boldsymbol{A_{SH}}$ be the adjacency matrix $\boldsymbol{A_H}$. Let $S\_u_i=\{v_1, v_2,\cdots, v_n\}$ for $1\le i\le n$.

Step 2: Then, we consider the original graphs $\boldsymbol{G}$ and $\boldsymbol{H}$ in the first iteration. We run the algorithm **ComputeRRM** by the adjacency matrix $\boldsymbol{A_G}$ of $\boldsymbol{G}$ and the adjacency matrix $\boldsymbol{A_H}$ of $\boldsymbol{H}$ in order to obtain a vertex match set $\boldsymbol{MS}$ and a row-row mapping $\boldsymbol{M}$ between $\boldsymbol{A_G}$ and $\boldsymbol{A_H}$. If there does not exist such a row-row mapping $\boldsymbol{M}$ (i.e., the row-row mapping $\boldsymbol{M}$ is an empty set), we can determine that the original graphs $\boldsymbol{G}$ and $\boldsymbol{H}$ are non-isomorphic and terminate the algorithm. Otherwise, we save the $\boldsymbol{BY_G}$, $\boldsymbol{BT_G}$, $\boldsymbol{BY_H}$ and $\boldsymbol{BT_H}$ as the return values.

Step 3: In this step, we do some auxiliary work before the next iteration. Since if there exists one vertex u_i of the original graphs $\boldsymbol{G}$ such that $\boldsymbol{G}$–u_i and $\boldsymbol{H}$–v_j for $1\le j\le n$ are non-isomorphic, we can determine that the original graphs $\boldsymbol{G}$ and $\boldsymbol{H}$ are non-isomorphic. Therefore, we will consider the isomorphism of the subgraphs $\boldsymbol{G}$–u_i and $\boldsymbol{H}$–v_j for $1\le i\le n$ and $1\le j\le n$. In order to enhance the efficiency, we can only consider the subgraphs $\boldsymbol{G}$–u_i and $\boldsymbol{H}$–v_j where v_j is the element of S_u_i for $1\le i\le n$. If the hypothesis is true, we can determine that the original graphs $\boldsymbol{G}$ and $\boldsymbol{H}$ are non-isomorphic and terminate the algorithm. Otherwise, we go to **Step 4**.

Step 4: From then on, we will consider a sequence of subgraphs in the next iteration. For every element S_u_i, in which the vertex u_i has not been signed by "visited", of the vertex match set $\boldsymbol{MS}$, if there exists some element S_u_i such that $|S\_u_i|=1$, then we will consider the subgraphs $\boldsymbol{SG}=\boldsymbol{SG}$–$u_i$ and $\boldsymbol{SH}=\boldsymbol{SH}$–$v_j$ firstly, where v_j is the only element of S_u_i. Otherwise we will select arbitrarily one element S_u_i, in which the vertex u_i has not been signed by "visited", of $\boldsymbol{MS}$ and any one element v_j of S_u_i so as to obtain the subgraphs $\boldsymbol{SG}=\boldsymbol{SG}$–$u_i$ and $\boldsymbol{SH}=\boldsymbol{SH}$–$v_j$. We update the $\boldsymbol{A_{SG}}$ and the $\boldsymbol{A_{SH}}$ by deleting the row and the column, labeled by u_i, of the old $\boldsymbol{A_{SG}}$ as well as the row and the column, labeled by v_j, of the old $\boldsymbol{A_{SH}}$, respectively.

Step 5: We consider the subgraphs $\boldsymbol{SG}$ and $\boldsymbol{SH}$ in the next iteration. We run the algorithm **ComputeRRM** by the adjacency matrix $\boldsymbol{A_{SG}}$ of $\boldsymbol{SG}$ and the adjacency matrix $\boldsymbol{A_{SH}}$ of $\boldsymbol{SH}$ in order to obtain a vertex match set $\boldsymbol{MS}$ and a row-row mapping $\boldsymbol{M}$ between $\boldsymbol{A_{SG}}$ and $\boldsymbol{A_{SH}}$.

Step 6: If there exists such a row-row mapping $\boldsymbol{M}$ (i.e., the row-row mapping $\boldsymbol{M}$ is not an empty set), we sign the vertex u_i by "visited". Otherwise we consider another element v_j of S_u_i in **Step 4**. If all the elements of S_u_i are considered and no a row-row mapping is obtained, we can determine that the original graphs $\boldsymbol{G}$ and $\boldsymbol{H}$ are non-isomorphic and terminate the algorithm.

Step 7: Repeat **Step 4**, **5** and **6** iteratively until the subgraph $\boldsymbol{SG}$ has only two vertices. If there exists a row-row mapping $\boldsymbol{M}$, we save the $\boldsymbol{MS}$ and the $\boldsymbol{M}$ as the return values. Otherwise, we can determine that the original graphs $\boldsymbol{G}$ and $\boldsymbol{H}$ are non-isomorphic and terminate the algorithm.

Space bound: $O(n^2)$.

Running time: $O(n^4)$.

C. Isomorphism Function

Algorithm Ⅲ: DetermineIsomorphismFunction

Input: $\boldsymbol{A_G}$— an adjacency matrix of graph $\boldsymbol{G}$
$\boldsymbol{BY_G}$—a row code XOR distance matrix of graph $\boldsymbol{G}$
$\boldsymbol{BT_G}$—a row code AOR distance matrix of graph $\boldsymbol{G}$
$\boldsymbol{A_H}$—an adjacency matrix of graph $\boldsymbol{H}$
$\boldsymbol{BY_H}$—a row code XOR distance matrix of graph $\boldsymbol{H}$
$\boldsymbol{BT_H}$—a row code AOR distance matrix of graph $\boldsymbol{H}$
$\boldsymbol{M}$ — a row-row mapping
n— the number of vertices of graph $\boldsymbol{G}$

Output: $\boldsymbol{F}$— an isomorphism function

ISBN: 978-988-19251-4-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Step 1: After the algorithm **DetermineSubgraphRRM** has been executed successfully, we will run this algorithm **DetermineIsomorphismFunction** in order to find the isomorphism function by performing some row-column elementary operations based on the row-row mapping $\boldsymbol{M}$. We interchange the corresponding row and column of $\boldsymbol{A_H}$ according to every element of $\boldsymbol{M}$ as follows. For every element $u_i \leftrightarrow v_j$ of $\boldsymbol{M}$, we interchange the row labeled by v_i and the row labeled by v_j of $\boldsymbol{A_H}$, as well as the column labeled by v_i and the column labeled by v_j of $\boldsymbol{A_H}$.

Step 2: The same interchanging operations are performed on $\boldsymbol{BY_H}$ and $\boldsymbol{BT_H}$ respectively.

Step 3: After performing these row-column elementary operations, if $\boldsymbol{A_G}=\boldsymbol{A_H}$, $\boldsymbol{BY_G}=\boldsymbol{BY_H}$, and $\boldsymbol{BT_G}=\boldsymbol{BT_H}$, then the original graphs $\boldsymbol{G}$ and $\boldsymbol{H}$ are isomorphic. All the correspondences between u_i and v_j constitute the isomorphism function. That is, the isomorphism function $\boldsymbol{F}$ is the row-row mapping $\boldsymbol{M}$. Otherwise the original graphs $\boldsymbol{G}$ and $\boldsymbol{H}$ are non-isomorphic.

Space bound: $O(n^2)$.

Running time: $O(n^3)$.

IV. TESTING RESULTS

To test performance of our algorithm, a random graph generating program was used. All randomly generated pairs of graphs including regular graphs and irregular graphs were rejected as non-isomorphic or proved as isomorphic. Furthermore, our algorithm was tested on 11900 graphs from the Graph Database [10].

TABLE I
TESTING RESULTS OF THE FIVE CATEGORIES BY THE GRAPH ISOMORPHISM ALGORITHM

Graph Type	Numbers of Sub Categories	Graph Size	Numbers of Graph Pairs
Bounded-Valence	40	20～200	4000
Bi-Dim Mesh	38	16～196	3800
Random	20	20～200	2000
Tri-Dim Mesh	14	27～216	1400
Quad-Dim Mesh	7	16～81	700

The Graph Database consisted of five types of graphs. In each category there are further sub-categories based on size and other parameters of the main category. For each of these tests we have validated the identical row-row mapping returned by the isomorphism algorithm and have not found a counterexample. The programs were written in the C language and the experiments were carried out with a computer with AMD Athlon(tm) 64×2 Dual Core Processor 3600Hz/1GB. The testing results are summarized in Table I.

V. CONCLUSION

As an NP-hard problem, so far no polynomial time (in the number of vertices of the graphs) algorithms have been known for undirected graph isomorphism. Although a few linear average-case time complexity algorithms were found to solve this problem, the best algorithms still have exponential worst-case time complexity.

In this paper, based on a necessary condition we proposed previously, we developed an $O(n^4)$ algorithm for undirected graph isomorphism using vertex partition and refinement. In particular, our algorithm takes advantage of a recursive property that isomorphism of supergraphs will result in the isomorphism of subgraphs. The extensive experimental results on the Graph Database validated the correctness of this algorithm for testing graph isomorphism. As for future work, we plan to investigate more efficient implementation for our algorithm.

ACKNOWLEDGMENT

This research was supported by the Natural Science Foundation of Guangdong Province, P.R.C. (9251009001000005) and the Science and Technology Program of Guangdong Province, P.R.C. (2008B080701005).

REFERENCES

[1] Douglas B. West, “Introduction to graph theory,” second edition, Pearson Education Asia Ltd., 2004, pp.438-439.
[2] Babai L., P. Erdős, and S.M. Selkow, “Random graph isomorphism”, SIAM Journal of Computing, vol. 9, pp.628-635, 1980.
[3] Tomek Czajka and Gopal Pandurangan, “Improved random graph isomorphism”, Journal of Discrete Algorithm, vol. 6, pp.85-92, 2008.
[4] Ullmann J. R., “An algorithm for subgraph isomorphism”, Journal of the Association for Computer Machinery, vol. 23(1), pp.31-42, 1976.
[5] Schmidt D. C., Druffel L. E., “A fast backtracking algorithm to test directed graphs for isomorphism using distance matrices”, Journal of the Association for Computer Machinery, vol. 23(3), pp.433-445, 1976.
[6] McKay B. D., “Practical graph isomorphism”, Congressus Numberantium, vol. 30, pp.45-87, 1981.
[7] Cordella L. P., Foggia P., Sansone C., et al., “An improved algorithm for matching large graphs”, International Workshop on Graph-based Representation in Pattern Recognition, Ischia, Italy, pp.149-159, May 2001.
[8] Cordella L. P., Foggia P., Sansone C., et al., “Subgraph transformations for the inexact matching of attributed relational graphs”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26(10), pp.1367-1372, 2004.
[9] Aimin Hou, “Elementary operations on a matrix to determine the isomorphism of graphs”, Journal of Computer Engineering and Applications, vol. 42(20), pp.51-54, 2006. (in Chinese).
[10] The Graph Data, collection of isomorphic graphs, http://amalfi.dis.unina.it/graph/.