A Zero Knowledge Authentication Protocol Based on Novel Heuristic Algorithm of Dense Induced Subgraphs Isomorphism

Dr. N. M. G. Al-Saidi
Applied Sciences Department, University of Technology /Baghdad.
Email:nadiamg08@gmail.com
Dr. N.A. Rajab
Applied Sciences Department, University of Technology /Baghdad.
H. N. Abdul-Rahman
Email:haydernatiq86@ymail.com
Applied Sciences Department, University of Technology /Baghdad.

Received on: 5/8/2014 & Accepted on: 8/1/2015

ABSTRACT

Graphs provide an useful mathematical tool for modeling various real world phenomena. Dense graphs arise in many places of interest, for instance the internet and social networks to name just two. The density of a graph should be a real number reflecting just how many edges it contains. Many networks found in the real world share the so-called "small world" property that is organized into communities. These organizations rely on close relationships of people belonging to a same subgroup.

The term "network" is used to denote the real world entity that usually maps to a graph after it is modeled. Therefore, there is greater need to propose more efficient graph and subgraph match methods to decide if their structures are identical.

Reduce the search space in these networks motivate many researchers to give generously persevering attempt to propose a new efficient algorithm for that purpose. According to theoretical and practical interest in graph isomorphism, a new algorithm for determining graph isomorphism between two dense graphs is proposed. Furthermore, a new algorithm for determining an induced subgraph isomorphism between pattern and target graphs is proposed also.

Those algorithms are analyzed from complexity point of view to demonstrate its effectiveness after applying it to several types of graphs. It is demonstrated that subgraph isomorphism is an improvement over the use of graph isomorphism in the zero knowledge protocol. The improvement comes from subgraph isomorphism being an NP-complete problem, and therefore, more difficult for an unauthorized user to solve. Whereas, the graph isomorphism problem has been solved therefore, is vulnerable to attacks of malicious users. The algorithms have been applied using VB-language, with two easy to use interfaces to be helpful for the beneficiary.

Keywords: graph algorithms, subgraph isomorphism, Zero-knowledge

بروتوكولات التوثيق بالمعرفة الصفرية بالاعتماد على خوارزمية الكشف عن البيانات الكثيفة المقطعية المتشاكلة

الخلاصة

توفر البيانات أدوات مفيدة للنماذج الرياضية المختلفة التي تستخدم في التطبيقات الواقعية. ان للبيانات الكثيفة فوائد عديدة، على سبيل المثال شبكة الإنترنت والشبكات الاجتماعية. كما ان كثافة هذه البيانات تتحدد بعدد حقيقي من الحافات التي تحتويها.يستخدم مصطلح" الشبكة" للدلالة على كيان العالم الحقيقي الذي يحول الى بيان. يوجد في هذا العالم العديد من الشبكات التي تشترك في ما بينها لتجعل العالم اصغر.بالتالي فان هنالك حاجة كبيرة لاقتراح خوارزميات تتعامل مع البيانات والبيانات الجزئية لتحدد فيما إذا كانت هياكلها متطابقة. ان تقليل مساحة البحث في هذه الشبكات تحفز العديد من الباحثين لاقتراح خوارزمية جديدة فعالة لهذا الغرض. نظرا للفائده النظرية والعملية للبيانات المتشاكله، تم اقتراح خوارزمية جديدة لتحديد التشاكل بين اي اثنين من البيانات الكثيفة. وعلاوة على ذلك،تم اقتراح خوارزمية جديدة لتحديد البيانات الجزئية المتشاكلة،تعتمد على تحليل البيان إلى مجموعة من المسارات، من ثم تحديد اقصرالمسارات التي تربط بين اي رأسين من روؤس البيان. ان استخدام التشاكل الجزئي في البيانات هو تحسين على استخدام التشاكل في بروتوكولات المعرفة الصفرية(Zero-Knowledge). هذا التحسين يكمن في كون التشاكل الجزئي هو من المسائل المعقدة(NP-Complete) ، وبالتالي فانه من الصعب على الاشخاص غير المخولين حلها. في حين تم حل مسائل التشاكل في البيانات وبالتالي تعرضها لهجمات من قبل المستخدمين المتطفلين.تم تطبيقا لخوارزميات باستخدام لغة الـ VB، مع واجهتين سهلة الاستخدام،لكي تكون مفيدة للمستخدم المستفيد.

INTRODUCTION

Graphs are widely used in real-life applications to model structured objects, molecules, images, combinatorial and networks. In many of these applications, one has to compare graphs to decide if their structures are identical. This problem is known as the graph isomorphism. It tests whether there is a one-to-one mapping between the vertices of two graphs. At the theoretical level, its main theoretical interest is to know whether graph isomorphism is in P or NP-complete.

Different approaches to the problem of graph isomorphism are presented. We were attempted to give a sketch description for the main idea of each algorithm. Isomorphism algorithms practical are classified into two main categories; one of them uses direct approach as they take the two graphs to be compared, and try to find an isomorphism between them directly with a classical depth-first algorithm. The second class uses different approaches, they take a single graph G and compute canonical labeling of the graph, such that, for two graphs G and H, $C(G) = C(H)$ if and only if G and H are isomorphic. One of such algorithm is RW [1], which is based on Random walks. It uses the steady state probability distribution of the Random Walk for the nodes. A modified version, RW2 [2], for this algorithm has also peen proposed by the same authors with a better matching rate and the same time complexity.

Other non-traditional approaches have been explored by McGregor [3] into new domain as a constraint. This approach does not consider significant due to the fact that the constraint problem is an NP-hard problem. The weakness of non-traditional approaches is because of transforming the problem into another paradigm.

The subgraph isomorphism is an important problem and very general form of graph matching that finds practical applications in image processing, computer vision, computer

aided design, bio computing, graph grammars, graph transformation, search operation in chemical database, Hamiltonian paths, cliques, matching, girth, and shortest paths [4].It is also used for feature recognition to describe the topological shape of a part as well as that of a primitive feature. Feature recognition is performed by searching graph representation to identify a subgraph that matches the primitive using subgraph isomorphism [5].

The heuristic search techniques are the essential part that most of the recent research on subgraphs isomorphism algorithms is based on; it is described in [6, 7, 8, and 9]. Some special cases used a tree searching procedure that is work better for example, Akinniyi et al., in [6]. Cheng and Huang in [7] used bitwise parallelism during the resolution process even though a sequential computer is used.

Cortadella and Valiente [8] were handling the combinatorial explosion in the case of small pattern graphs. The notion of neighborhood constraints has been considered by Larrosa and Valiente [9]. In particular, an efficient algorithm is described by Ullmann [10]; this algorithm proposes the first practical algorithm for subgraph isomorphism search for graphs. It is a backtracking algorithm which finds solutions by incrementing partial solutions or abandoning them when it determines that they cannot be completed.A more recent algorithm, known as VF2, it is based on a depth-first search strategy with a set of rules to efficiently prune the search tree. Such rules in case of isomorphism are shown in [11].

The growth and prosperity of the society are performed through communication that allow for the transfer of data between two or more entities. Determining whether the data used to make decisions are trustworthy and valid is a big challenge prompts many researchers to be in a continuous seeking for efficient and secure methods.The both communicated sides have to trust the exchange of data for each other. Many protocols have been designed for this purpose, most of them required sufficient amount of time for new entity authentication. Minimizing the latency of authentication protocols is an important demand to protect the resources and network data [12]

A zero knowledge protocol is one of such authentication protocols.Networks and entity groupings requires entity authentication while preserving the privacy of the entity being authenticated. Zero-Knowledge Proof plays an important role in authentication without revealing secret information [13].

The zero knowledge protocol makes use of graph isomorphism, and subgraph isomorphism. Since the graph isomorphism problems have been considered as a P problem, therefore, it is vulnerable to attacks of malicious users because it can be solved easily when it is used in some authentication protocols. Subgraph isomorphism has been proved as an NP-complete problem [14], so, it is considered as an improvement over the graph isomorphism problem for such of these protocols, such as, zero knowledge protocol. In this paper, an induced subgraph isomorphism is used as a zero-knowledge protocols, it shows an improvement over graph isomorphism problem in these protocols.

Definition and notations

We present the mathematical terminology used in this paper. Graphs and concepts related to graphs are introduced, followed by the required concepts and basic results from matrix theory.

Definition

A graph $G = (V, E)$ or simply G consists of, a nonempty finite set called the vertices V and a finite collection of unordered pairs of vertices called the edges E. An edge $e = (u, v)$ connects or joins the vertices $u$and v, the vertices u and v are called the ends of e.The order of G is the number of vertices in V,denoted by $|V|$, and the size of G is the number of edges in Edenoted by $|E|$.

Definition

A graph $G = (V, E)$ is called vertex-labeled(or simply labeled) if a mapping $L: V \rightarrow N$ is given. $L(v)$is called a label of a vertex v.

Definition

A graph $G = (V, E)$ is called edge-labeled if a mapping $L: E \rightarrow N$ is given. $L(e)$is called a label of an edge e.

Definition

The degree of a vertex v is the number of edges incident to v, denoted by $\deg(v)$. The maximum degree of a graph G, denoted by $\Delta(G)$, is defined to be
$\Delta(G) = max\{deg(v) \mid v \in V\}$.
Similarly, the minimum *degree* is denoted by $\delta(G)$, and such that
$\delta(G) = min\{deg(v) \mid v \in V\}$.

Definition

In a graph $G = (V, E)$, the density is the ratio between the number of edges $|E|$ and the number of vertices $|V|$. A graph $G = (V, E)$ is said to be denseif $\delta(G) \geq \frac{1}{2}|V|$.

Definition 2.6 [15]: A regular graph is a graph that it is vertices has the same degree. A regular graph with vertices of degree k is called a k-regular graph or a regular graph of degree k.

Definition

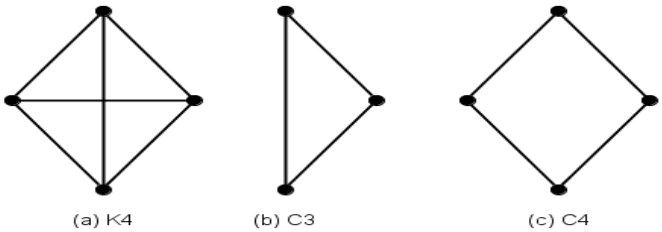
In a graph G, the distance from vertex u to vertex v denoted by $d(u, v)$ is the length of the shortest walk from u to v, (i.e. the number of edges in such a walk). If there is no walk from $u$to v, then $d(u, v) = \infty$. For any $u, v \hat{I} V$, the distance from v to u is defined as $d(u, v) = min\{d(u, v)\}$.

Definition

A subgraph H of a graph G is a graph with $V(H) \subseteq V(G)$, and $E(H) \subseteq E(G)$. It is denoted by $H \leq G$.

Definition

induced subgraph is a subset of the vertices of a graph G together with all edges whose end points are both in this subset.An example of an induced subgraph of K_4 is C_3 obtained by selecting three vertices u, v, w and all edges between those vertices. Note that C_4 is not an induced subgraph of K_4 since all edges between the four vertices must be included. Figure 2.1, demonstrates an example of a subgraph, spanning subgraph, and an induced subgraph. Graph (b) represents both a subgraph and induced subgraph of (a), whereas (c), is a spanning subgraph but not an induced subgraph of (a).



Figure(2.1)An induced subgraph

Definition

Two graphs G_p = (V_p ,E_p) and G_t = (V_t,E_t) are isomorphic denoted by $G_p \cong G_t$, if there is a bijection $\emptyset$:$V_p \rightarrow V_t$ such that, for every pair of vertices v_i, $v_j \in V_p$, (v_i, v_j)$\in E_p$ if and only if ($\emptyset(v_i)$,$\emptyset(v_j)$) $\in E_t$. The Bijection $\emptyset$ is said to be an isomorphism between the two graphs. Figure 2.20 is an example of graph isomorphism.

Definition

An induced subgraph isomorphism is an isomorphism with an induced subgraph of a given graph, i.e., a graph G_p = (V_p,E_p) is isomorphic to an induced subgraph of a graph G_t = (V_t,E_t) if there exists an induced subgraph of G_t, say G_S, such that $G_P \cong G_S$. In this case a corresponding bijection between vertices of G_P and G_S is said to be an induced sub-isomorphism between two graphs.

Incidence matrix

The incidence matrix of an undirected graph with n vertices and m edges is a matrix $M_{n\times m} = [m_{i,j}]$,

where,

$$[m_{i,j}]=\begin{cases}1 & if\ v_i\ is\ incident\ with\ e_j\\ 0 & otherwise\end{cases}$$

Therefore, in addition to labeling the vertices, we must also label the edges. Hence, once the labeling is complete, the matrix is developed where, the rows represent the vertices and the columns represent the edges. If an edge is incident to a vertex in the graph, then the corresponding entry of the matrix is equal 1, otherwise the entry is zero.

Graph and induced subgraph isomorphism

The proposed method presents a heuristic algorithm for both graph and induced subgraphs isomorphism. It is based on the decomposition of the graph into components and refinements to find identification graph for G_1 and G_2 in graph isomorphism, and pattern graph in subgraph isomorphism. Each component is denoted by I_{vivj} that represents all sub-paths that contains minimum number of edges in the set of all paths between $v_i$and v_j.

Paths matrixI_G

In this section, a heuristic algorithm is given to obtain paths matrix; it is a symmetric matrix, with entries that represent all sub-paths contain minimum number of edges in the set of all paths between any two vertices.

Algorithm 3.1: (This algorithm is applied to connected dense graph to obtain paths matrix I_G).

Input: An incidence matrix $M_{n\times m} = [m_{ij}]$ of a connected dense graph $G = (V, E)$ with $|V| = n$and $|E| = m$.

Output: paths matrix $I_G$when its elements are I_{vivj} that represent all the paths from vertex v_i to v_j with minimum number of edges, where $v_i, v_j \in V$.

i. For $i = 1{:}n$.
ii. $j = i + 1$.
iii. Starting at vertex v_i, check for all $e_r{:}r = 1{:}m$ if $m_{ir} = m_{jr} = 1$ then the shortest path is found which is (v_i,e_r,v_j) and call it$I_{v_iv_j,x}$.
iv. Repeat to find other paths between $v_i$and v_j.
   a. Start at v_i, add e_r if $m_{ir} = 1$, after that add v_k.
   b. Continue until v_j is reached.
v. Among all paths of step iv, choose the paths of minimum number of edges in each sub-path and call it $I_{v_iv_j,y}$.
vi. From steps iii and v, construct $I_{v_iv_j}$=$I_{v_iv_j,x} \cup I_{v_iv_j,y}$.
vii. Return to step (ii) with $j = j + 1$ then continue until reaching $j = n$.
viii. End For.

Graph identification

We give two heuristic methods to represent the identification for regular and irregular graph. Depending on the paths matrix of graph, the identification is determined by finding a minimum number of $I_{v_iv_j}$ that contains all edges and vertices of graph.

Algorithm 3.2: (This algorithm is applied on a regular graph to find its identification).

Input: $I_{G(n\times n)}$

Output: identification $I_{G,I}$of G.

i. For $i = 1{:}n - 1$
ii. For $j = i + 1{:}n$
iii. Computes $I_{v_iv_j}$ and $I_{v_nv_1}$ to do the following:
   a. Check about the one that contains all edges and vertices, then choose $I_{v_iv_j}$ that contains minimum number of edges in each path. Call it $I_{G,I}$, the identification of G.
   b. Else, collect two or more$I_{v_iv_j}$, to determine the identification of Gthat contains all edges and vertices of G.
iv. End

Algorithm 3.3: (This algorithm is applied on irregular graph to find its identification).

Input: incidence matrix $M_{n\times m}$and $I_{G(n\times n)}$.

Output: identification $I_{G,I}$of G.

i. Among all rows, identify the rows that contain vertices of largest degree.
ii. Repeat, for each row in step (i) to perform the following:
   a. Search in all $I_{v_i v_j}$ about the one that contains all edges and vertices, then choose $I_{v_i v_j}$ that contains minimum number of edges in each path. Call it $I_{G,\mathrm{I}}$, the identification of the G.
   b. Else, collect two or more $I_{v_i v_j}$ of the same row in step (i) to determine the identification of G.
iii. End

Graph isomorphism

The problem of deciding whether two graphs are isomorphic is fundamental in graph theory. Graphs G_1 and G_2 are said to be isomorphic if their vertices can be rearranged, so that, the corresponding edge structure is exactly the same. This problem can be very difficult to solve even for small graphs without a good algorithm. In this section a new algorithm is presented for determining whether two given graphs are isomorphic or not, as follows;

Algorithm 3.4: (This algorithm is applied to find an isomorphism between $G_1$and G_2).
Input: the identification $I_{G_1,I}$ and $I_{G_2,I}$
Output: isomorphism between G_1 and G_2
i. If the number of vertices, edges and the degree sequence of vertices are equal in G_1 and G_2 then continue. Else stop to conclude that G_1 and $G_2$are not isomorphic.
ii. If $\mathrm{I}_{G_1,\mathrm{I}}$ contains same number of $I_{v_i v_j}$, same number of edges in each paths corresponding to $\mathrm{I}_{G_2,\mathrm{I}}$ then stop, G_1 and G_2 are isomorphic.
iii. Else $G_1$and G_2 are not isomorphic.
iv. End

Example: Let G_1 and G_2 be two given regular graphs presented in Figure 3.1.

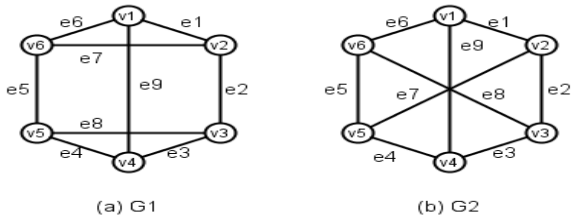


Figure 3.1 Two regular graphs

An algorithm 3.1 is applied to construct I_{G_1} as follows;
$I_{v_1 v_2}$= {(v_1, e_1, v_2), (v_1, e_6, v_6, e_7, v_2)}.
$I_{v_2 v_3}$= {(v_2, e_2, v_3), ($v_2, e_1, v_1, e_9, v_4, e_3, v_3$), ($v_2, e_7, v_6, e_3, v_5, e_8, v_3$)}.
$I_{v_3 v_4}$= {(v_3, e_3, v_4), (v_3, e_8, v_5, e_4, v_4)}.

$I_{v_4v_5}$= {(v_4, e_4, v_5), (v_4, e_3, v_3, e_8, v_5)}.
$I_{v_5v_6}$= {(v_5, e_5, v_6), ($v_5, e_4, v_4, e_9, v_1, e_6, v_6$), ($v_5, e_8, v_3, e_2, v_2, e_7, v_6$)}.
$I_{v_6v_1}$= {(v_6, e_6, v_1), (v_6, e_7, v_2, e_1, v_1)}.
Now, algorithm 3.2 is applied to determine $I_{G_1,I}$of G_1, such that,
($I_{v_2v_3}$ ∪
$I_{v_5v_6}$)={(v_2, e_2, v_3),($v_2, e_1, v_1, e_9, v_4, e_3, v_3$),($v_2, e_7, v_6, e_3, v_5, e_8, v_3$),(v_5, e_5, v_6),
($v_5, e_4, v_4, e_9, v_1, e_6, v_6$), ($v_5, e_8, v_3, e_2, v_2, e_7, v_6$)}.
Since ($I_{v_2v_3}$ ∪ $I_{v_5v_6}$) contains all edges and vertices of G_1, then ($I_{v_2v_3}$ ∪ $I_{v_5v_6}$) is the identification of G_1.
as well as, algorithm 3.1 is applied to construct I_{G_2}, as follows;
$I_{v_1v_2}$={(v_1, e_1, v_2),($v_1, e_9, v_4, e_3, v_3, e_2, v_2$),($v_1, e_6, v_6, e_8, v_3, e_2, v_2$),
($v_1, e_9, v_4, e_4, v_5, e_7, v_2$), ($v_1, e_6, v_6, e_5, v_5, e_7, v_2$)}.
$I_{v_2v_3}$={(v_2, e_3, v_3),($v_2, e_1, v_1, e_9, v_4, e_3, v_3$),($v_2, e_1, v_1, e_6, v_6, e_8, v_3$),
($v_2, e_7, v_5, e_6, v_6, e_8, v_3$), ($v_2, e_7, v_5, e_4, v_4, e_3, v_3$)}.
$I_{v_3v_4}$={(v_3, e_3, v_4),($v_3, e_2, v_2, e_1, v_1, e_9, v_4$),($v_3, e_8, v_6, e_5, v_5, e_4, v_4$),
($v_3, e_8, v_6, e_6, v_1, e_9, v_4$), ($v_3, e_2, v_2, e_7, v_5, e_5, v_4$)}.
$I_{v_4v_5}$={(v_4, e_4, v_5),($v_4, e_3, v_3, e_8, v_6, e_5, v_5$),($v_4, e_3, v_3, e_2, v_2, e_7, v_5$),
($v_4, e_9, v_1, e_1, v_2, e_7, v_5$), ($v_4, e_9, v_1, e_6, v_6, e_5, v_5$)}.
$I_{v_5v_6}$={(v_5, e_5, v_6),($v_5, e_7, v_2, e_1, v_1, e_6, v_6$),($v_5, e_7, v_2, e_2, v_3, e_8, v_6$),
($v_5, e_4, v_4, e_9, v_1, e_6, v_6$), ($v_5, e_4, v_4, e_3, v_3, e_8, v_6$)}.
$I_{v_6v_1}$={(v_6, e_6, v_1),($v_6, e_8, v_3, e_2, v_2, e_1, v_1$),($v_6, e_5, v_5, e_4, v_4, e_9, v_1$),
($v_6, e_5, v_5, e_7, v_2, e_1, v_1$), ($v_6, e_8, v_3, e_3, v_4, e_9, v_1$)}.
Now, algorithm 3.2 is applied to determine $I_{G_2,I}$of G_2, such that,
$I_{v_1v_2}$= {(v_1, e_1, v_2), ($v_1, e_9, v_4, e_3, v_3, e_2, v_2$), ($v_1, e_6, v_6, e_8, v_3, e_2, v_2$),
($v_1, e_9, v_4, e_4, v_5, e_7, v_2$),($v_1, e_6, v_6, e_5, v_5, e_7, v_2$)}.
Since $I_{v_1v_2}$ contains all edges and vertices of G_2, then $I_{v_1v_2}$is the identification of G_2.
G_1andG_2 are not isomorphic because the identification of these graphs are different,$I_{G_1,I} \neq I_{G_2,I}$

Subgraph isomorphism

This section presents the main algorithm, which finds the subgraphs and all induced subgraphs of a given target graph G_T isomorphic to a pattern graph G_P. This method essentially improves the run time when the algorithm was applied on dense graphs.

Algorithm 3.5 (this algorithm is applied on G_T to obtain all induced subgraph isomorphism).
Input: I_{G_T}.
Output: all induced subgraph isomorphism between $G_T$and G_P.

i. Depending on the type of graph, determine the identification of pattern graph $I_{Gp,I}$ by algorithms 3.2 or 3.3.
ii. Repeat for each row in step (ii) to check the following.

a. If any entry contains equal number of paths, equal number of edges and vertices in each I_{vivj} corresponding to $I_{Gp,\mathrm{I}}$ then choose this entry.

b. For each row, find the number of edges in each entry $I_{v_iv_j}$, if the number of paths is greater than those of $I_{Gp,\mathrm{I}}$ then remove some of them in a permutation way to obtain new collection of paths to be compared to $I_{Gp,\mathrm{I}}$.

iii. If $I_{Gp,\mathrm{I}}$ consists of two or more of $I_{v_iv_j}$, then collect two or more $I_{v_iv_j}$of step a or b in the same row as $I_{GP,I}$.

iv. End

Example: Let G_1 and G_2 be two given graphs presented in Figure 3.2**.**

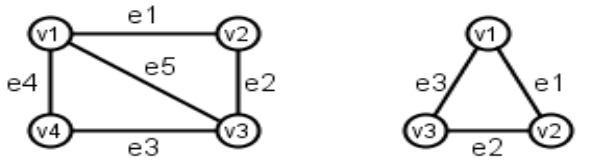


Figure 3.2 Target and pattern graph

The incidence matrix of pattern graph $G_P(M_{3\times3})$ is as follows;

$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array}\begin{array}{c} \begin{array}{ccc} e_1 & e_2 & e_3 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \end{array}$$

Algorithm 3.1 is applied to construct I_G of pattern graph; all paths between any two vertices are as follows

$\boldsymbol{v_1 \rightarrow v_2}$

(v_1, e_1, v_2), $(v_1, e_3, v_3, e_2, v_2)$

$I_{v_1v_2,x} = (v_1, e_1, v_2)$

$I_{v_1v_2,y} = (v_1, e_3, v_3, e_2, v_2)$

$I_{v_1v_2} = \{(v_1, e_1, v_2), (v_1, e_3, v_3, e_2, v_2)\}$

$\boldsymbol{v_1 \rightarrow v_3}$

$I_{v_1v_3} = \{(v_1, e_3, v_3), (v_1, e_1, v_2, e_2, v_3)\}$

$\boldsymbol{v_2 \rightarrow v_3}$

$I_{v_2v_3} = \{(v_2, e_2, v_3), (v_2, e_1, v_1, e_3, v_3)\}$

The paths matrix for pattern graph denoted I_{GP} is a symmetric matrix, where $I_{v_iv_j} = I_{v_jv_i}$ so, it is enough to find the upper or lower triangle.
Table: The paths matrix of pattern graph G_P

	v_1	v_2	v_3
v_1	0	$I_{v_1v_2}$	$I_{v_1v_3}$
v_2	$I_{v_2v_1}$	0	$I_{v_2v_3}$
v_3	$I_{v_3v_1}$	$I_{v_3v_2}$	0

Similarly: algorithm 3.1 is applied to construct I_{GT} of target graph G_T, as follows;
The incidence matrix of target graph $G_T(M_{4\times5})$ is as follows;

$I_{v_1v_2}$= {(v_1, e_1, v_2), (v_1, e_5, v_3, e_2, v_2)}
$I_{v_1v_3}$= {(v_1, e_5, v_3), (v_1, e_1, v_2, e_2, v_3), (v_1, e_4, v_4, e_3, v_3)}
$I_{v_1v_4}$= {(v_1, e_4, v_4), (v_1, e_5, v_3, e_3, v_4)}.
$I_{v_2v_3}$= {(v_2, e_2, v_3), (v_2, e_1, v_1, e_5, v_3)}.
$I_{v_2v_4}$= {(v_2, e_1, v_1, e_4, v_4), (v_2, e_2, v_3, e_3, v_4)}.
$I_{v_3v_4}$= {(v_3, e_3, v_4), (v_3, e_5, v_1, e_4, v_4)}.

Now, algorithm 3.2 is applied to determine the identification of pattern graph$I_{G_p,I}$
Since ($I_{v_1v_2} or\ I_{v_1v_3}\ or\ I_{v_2v_3}$) contains all edges and vertices of G_p, then ($I_{v_1v_2} or\ I_{v_1v_3}\ or\ I_{v_2v_3}$) is the identification of G_p.
An algorithm 3.5 is applied on G_T to find subgraph and all induced subgraph of $G_P$that is isomorphism to G_T.
$I_{v_1v_2}$= {(v_1, e_1, v_2), (v_1, e_5, v_3, e_2, v_2)}
$I_{v_1v_4}$= {(v_1, e_4, v_4), (v_1, e_5, v_3, e_3, v_4)}.
These paths represent subgraph and all induced subgraph of a given target graph G_T isomorphic to a pattern graph G_P.

Software implementation

We have proposed in previous section algorithms on graph and subgraph isomorphism. The application of graph and subgraph isomorphism provides faster heuristic algorithms for finding, not one, but all induced subgraphs of the database graph isomorphic to a given pattern graph. This helps in to compute the measure of statistical significance of that pattern graph in the database and match between them in graph isomorphism.

This section is started off by describing how the implemented code operates. The second part of this section is analyzed the complexity of each algorithm to investigate the performance of the whole algorithm. In other hand, the third part deals with presenting and analyzing of some experimental results. Finally, a zero knowledge protocol which based on of induced subgraph isomorphism is described in section 5.

The proposed algorithms were implemented on a computer with the specification, CPU CORi3 with 2 GB Ram under the operating system Windows 7 using Visual Basic NET 2012.

Graph isomorphism

In this part, the first step is to input the elements of incidence matrix for the given graphs $G_1$and G_2. The second part is to find the identification of two graphs G_1 and G_2 to be used for comparison between them. The final step is to check whether they are isomorphism or not. This is performed through the key (find graph isomorphism) as shown in Figure 4.1.

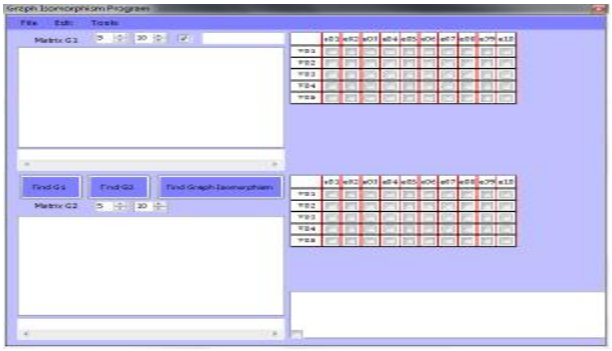


Figure 4.1 Graph isomorphism user interface

Subgraph isomorphism:

In this part, the first step is to input the elements of incidence matrix for the pattern graph and the target graph. The identification of pattern graph and the paths matrix of target graph are calculated. Finally checking for subgraph isomorphism is accomplished by using the key (find subgraph isomorphism) as shown in Figure 4.2.

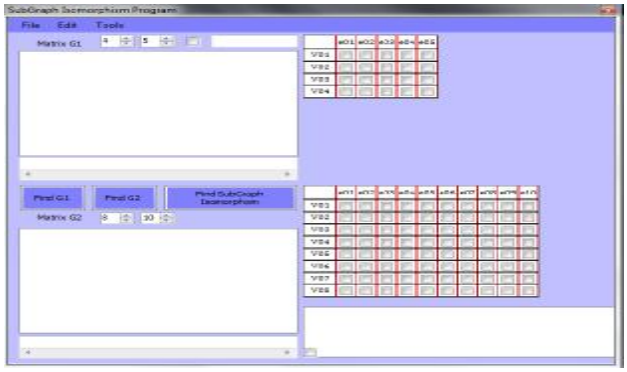


Figure 4.2 Subgraph isomorphism form

Experimental results

In order to verify the effectiveness of the proposed algorithm, some tests have been performed on dense graphs database. On these graphs, we have measured the time required for finding all the subgraph isomorphism or isomorphism between G_1 and G_2 using proposed algorithms.

For the synthetic case, we generate random graphs of sizes (3. 9) vertices with various numbers of edges and labels to serve as database graphs. We used random graphs in graph isomorphism: graphs with $deg(v) \geq {}^{n}/_{2}$, where $v \in V$. For subgraph isomorphism, we use two types of random graphs are used: a target graph with $deg(v) \geq {}^{n}/_{2}$, where $v \in V$, and pattern graph.

target graph	Pattern graph	Identification time of pattern graph	Paths matrix time of target graph	Number of matches
4 nodes	3 nodes	0.006	0.006	2
5 nodes	3 nodes	0.006	0.006	4
5 nodes	4 nodes	0.009	0.009	3
6 nodes	4 nodes	0.009	0.009	8
7 nodes	3 nodes	0.451	0.006	4
7 nodes	4 nodes	0.451	0.009	13
7 nodes	5 nodes	0.451	0.022	9
8 nodes	3 nodes	0.380	0.006	16
8 nodes	4 nodes	0.380	0.009	11
8 nodes	5 nodes	0.380	0.022	26
9 nodes	3 nodes	0.343	0.006	18
9 nodes	4 nodes	0.343	0.009	18

After applying this algorithm on different cases, some of the concluding remarks are abstracted as follows:

- This algorithm can be used for graph isomorphism and induced subgraph isomorphism.
- Choosing the maximum degree vertex helps to reduce the search space that is better than random search.
- Using this algorithm for finding all induced subgraph has an advantage over some well-known algorithms designated to find only one graph or subgraph isomorphism.

- Based on minimal path search, it is better and faster than other types of search based on a chosen starting point randomly. This is because the determination of start and end vertices will facilitate the search space and improve the complexity of the search algorithm.
- The density condition on the graph being used for search is another advantage that leads to a wide area of application of the proposed algorithm over some known algorithms suitable to be applied for low connected graphs.
- The worst case arises when the vertices of the given graph are complete, so this case is overtaken in our algorithm.

Induced subgraph isomorphism as Zero-Knowledge Protocols

Zero knowledge protocols enable one party called the prover to convince another party called the verifier the validity of a mathematical statement, such that, the verifier learns nothing other than the fact that the proven statement is true.

- To use an induced subgraph isomorphism in zero knowledge protocols we should start with two public graphs, $G_1$and G_2. The isomorphism between graph G_1 and a subgraph H of $G_2$is the private key for the prover that should be kept secret from all parties. Whereas, the verifier sends the prover either 0 or 1,if 0 is sent, the prover will send G_3, if 1 is sent, the prover will send G_4.
- During the authentication exchange, the secret $H_i$of the Prover will never compromise. In each round, one of two cases is proved. Either a graph that is isomorphic to graph $G_2$is sent by the Prover, or a graph containing a subgraph isomorphic to graph $G_1$is sent by the Prover.
- In this protocol, the Verifier has no method to extract the Prover'ssecret H_i. The verifier can never test the Prover's secret, so, he will always have doubt that the Prover is a valid user, which is a part of the risk of using zero knowledge protocols.
- By repeating the above verification steps based on subgraph isomorphism protocol, the Verifier doubt is reduced. The answer for both 0 and 1by the Prover should be prepared as long as he doesn't anticipate the Verifier response. The chance for the prover to guess the Verifier response is reduced as long as the multiple verification rounds is performed, and he will gradually win the Verifier's trust.
- For n running time verification process, the trust is equal to 1-(½)^n.The Verifier's trust of the prover moves closer, but never to 1with each run. Once the trust value has reached a specific threshold such as 95%, the protocol can be set to completely trust a prover.

Design of a parallel subgraph isomorphism protocol.

In the beginning, we have to create two public graphs known as $G_1$and G_2. The prover must create secret array $H_1, \dots, H_k$ of G_2 by algorithm (4.5) to find all subgraph isomorphism; this is kept by the prover and never shared.

1. At the beginning of a verification round the prover will create an array called IS to be applied on G_2 in order to obtain the graph G_3.
2. For each verification round between the prover and the verifier, the prover will create array IS,and therefore, a new G_3.

3. The prover chooses random H_j, $j = 1, \dots, k$ from G_3, depending on the vertex of secret array of the prover H_i, $i = 1, \dots, k$.
4. The verifier sends the prover either 0 or 1. If 0 received, then the Prover will send $G_2$and G_3. If 1 is received then the Prover will send $G_1$and G_4.
5. The verifier must be able to verify the prover is truthful during each round, the verifier will do this by testing that

a. Either G_2 and G_3 is isomorphic by algorithm (4.4).
b. Or by testing for a subgraph isomorphism G_4 of G_3 if it is isomorphic to G_1 by algorithm (3.4).

Example: Given two public graphs, pattern graph G_1, and target graph G_2, shown in Figure5.1.

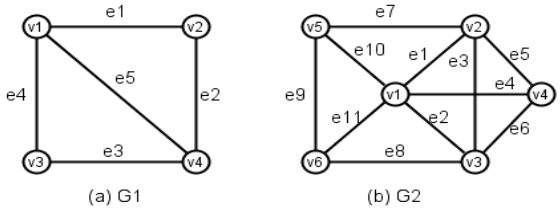


Figure 5.1 Public graphs

In the beginning, the prover creates random array to be applied on the adjacent matrix of the target graph G_2 to obtain G_3.
The adjacent of G_2

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	0	1	1	1	1	1
v_2	1	0	1	1	1	0
v_3	1	1	0	1	0	1
v_4	1	1	1	0	0	0
v_5	1	1	0	0	0	1
v_6	1	0	1	0	1	0

The vertex of G_2	v_1	v_2	v_3	v_4	v_5	v_6
IS	v_3	v_1	v_5	v_6	v_2	v_4

Now, the adjacent matrix of $G_3$is

	v_1	v_2	v_3	v_4	v_5	v_6
v_1	0	1	0	1	1	1
v_2	1	0	1	1	1	1
v_3	0	1	0	1	1	0
v_4	1	1	1	0	0	0
v_5	1	1	1	0	0	1
v_6	1	1	0	0	1	0

The new graph is G_3 obtained by applied ISon G_2, as shown in Figure 5.2.

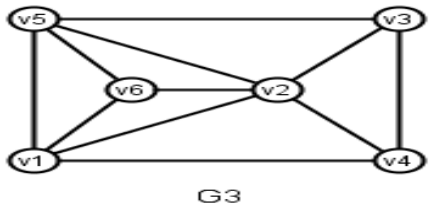


Figure 5.2A graph G_3

G_2Contains seven subgraph isomorphism of G_1, they are $(H_1, \dots, H_7)$, this kept by the prover and never shared.
$(H_1 = v_1, v_2, v_4, v_5)$, $(H_2 = v_1, v_2, v_3, v_5)$, $(H_3 = v_1, v_2, v_3, v_4)$, $(H_4 = v_1, v_3, v_4, v_6)$, $(H_5 = v_1, v_2, v_3, v_6)$, $(H_6 = v_1, v_2, v_5, v_6)$, $(H_7 = v_1, v_3, v_5, v_6)$.

If the prover chooses random H_6 and applied IS on it to obtain G_4.
The adjacent matrix of G_6 is as follows

	v_1	v_2	v_5	v_6
v_1	0	1	1	1
v_2	1	0	1	0
v_5	1	1	0	1
v_6	1	0	1	0

H_6	v_1	v_2	v_5	v_6
IS	□3	□1	□2	□4

Now, the adjacent matrix of □4is

	$\square_1$	$\square_2$	$\square_3$	$\square_4$
$\square_1$	0	1	0	1
$\square_2$	1	0	1	1
$\square_3$	0	1	0	1
$\square_4$	1	1	1	0

Outline of a Parallel subgraph isomorphism Protocol is as shown in Figure 5.3.

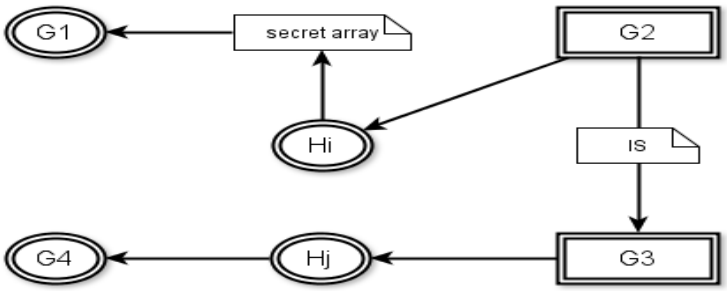


Figure 5.3: Components of the protocols and how they relate to each other.

CONCLUSIONS

The application of graph and subgraph isomorphism provides faster heuristic algorithms for finding, not one, but all induced subgraphs of the database graph isomorphic to a given pattern graph. New algorithms for graph and subgraph isomorphism are proposed in this paper. This helps to compute the measure of statistical significance of that pattern graph in the database and match between them in graph isomorphism. The graph isomorphism problem has been solved and therefore is vulnerable to attacks of malicious users. Therefore, subgraph isomorphism that is NP-complete problem is an improvement over the use of graph isomorphism in the zero knowledge protocol. The possibility of applying the proposed algorithm on a highly connected dense graph helps to expand fields of application that many traditional algorithms failed to cover.

REFERENCE

[1].M. Gori, M. Maggini, and L. Sarti. “Graph matching using random walks”. International Conference on Pattern Recognition, vol. 3, pp: 394–397, 2004.

[2]. M. Gori, M. Maggini, and L. Sarti.“The RW2 algorithm for exact graph matching”. International Conference on Advances in Pattern Recognition, vol. 3686 of Lecture Notes in Computer Science, pp: 81–88, 2005.

[3]. J. J. McGregor. “Relational consistency algorithms and their application in finding subgraph and graph isomorphisms”. Information sciences, vol. 19, no. 3, pp: 229–250, 1979.

[4]. V. Lipets, N. Vanetik, and E. Gudes. "Subsea: An efficient heuristic algorithm for subgraph isomorphism". Data Mining and Knowledge Discovery, vol. 19, pp: 320-350, 2009.
[5].H. S. Ketan. "Development Approach of Automated Recognition for Isolated and Intersection (Complex) Manufacturing Features for Prismatic Mechanical Parts". Eng. & Tech. Journal, vol. 28, no. 21, 2010.
[6]. F. A. Akinniyi, A. K. C. Wong, and D. A. Stacey. "A new algorithm for graph monomorphism based on the projections of the product graph". Trans. Systems, Man and Cybernetics, vol. 16, pp: 740–751, 1986.
[7]. J. K. Cheng, and T. S. Huang. "A subgraph isomorphism algorithm using resolution". The Journal of the Pattern Recognition Society, vol. 13, pp: 371–379, 1981.
[8]. J. Cortadella, and G. Valiente. "A relational view of subgraph isomorphism". In Proc. Fifth Int. Seminar on Relational Methods in Computer Science, pp: 45–54, 2000.
[9]. J. Larrosa, and G. Valiente. "Graph pattern matching using constraint satisfaction". Mathematical structures in computer science, vol. 12, pp: 403-422, 2002.
[10]. J. R. Ullmann. "An algorithm for subgraph isomorphism". J. Assoc. Comput. Mach., vol. 23, pp: 31–42, 1976.
[11]. L. P. Cordella, P. Foggia, C. Sansone, and M. Vento. "A (sub)graph isomorphism algorithm for matching large graphs". IEEE Trans. Pattern Anal. Mach. Intell., vol. 26, no. 10, pp: 1367–1372, 2004.
[12]. J. Kizza, L. Bramlett, and E. Morgan. "Using subgraph isomorphism as a zero knowledge proof authentication in timed wireless mobile networks". International Journal of Computing and ICT Research, vol. 4, no. 1, pp: 81-9, 2010.
[13].M. K. Ibrahim. "Modification of Diffie–Hellman Key Exchange Algorithm for Zero Knowledge Proof". Eng. & Tech. Journal, vol. 30, no. 3, 2012.
[14]. M. R. Garey, and D. S. Johnson. "Computers and Intractability: A guide to the theory of NP-completeness". W. H. Freeman and company New York, 1979.
[15]. S. Pejić. "Algebraic graph theory in the analysis of frequency assignment problems". PhD thesis, The London School of Economics and Political Science, 2008.
[16]. V. Lipets, N. Vanetik, and E. Gudes. "Subsea: An efficient heuristic algorithm for subgraph isomorphism". Data Mining and Knowledge Discovery, vol. 19, pp: 320-350, 2009.
[17]. Frank Harary. "Graph theory. Addison-Wesley Publishing" Co., Reading, Mass.-Menlo Park, Calif.-London, 1969.
[18]. A.V. Goldberg. "Finding a maximum density subgraph". Technical Report UCB/CSD-84-171, EECS Department, University of California, Berkeley, 1984.
[19]. S. F. Florkowski. "Spectral graph theory of the hypercube".Nova science publishers, 2008.