THE PERMUTATIONS ALGORITHM TO SOLVE THE GRAPH ISOMORPHISM+

Hameed.H.Hameed*

Abstract

This research study the isomorphism problem of two simple planar graphs which they have the same number of vertices and edges, we used the permutations algorithm which generates a permutation from known permutation. In this paper a program in (Mat Lab) has been constructed to compare two graphs by generating all permutations of the vertices of the first graph and finding the adjacency matrix of it in each permutation then compare it with the adjacency matrix of the second graph . Finally the result discuses whether the two graphs are isomorphic or not.

المستخلص :

يقوم هذا البحث بدراسة تشاكل بيانين مستويين لهما نفس العدد من الرؤوس ونفس العدد من الحافات وذلك عن طريق استخدام خوارزمية التباديل والتي تولد تبديل من تبديل معطى مسبقاً، حيث قمنا بإنشاء برنامج بواسطة (Mat Lab) يقوم بمقارنة مصفوفة التجاور لأحد البيانين مع كل مصفوفات التجاور المستنتجة من تباديل رؤوس البيان الآخر ، فإذا تساوت المصفوفتين في احد التباديل يكون البيانين متشاكلين وعند عدم الحصول على التساوي ولكافة التباديل يكون البيانين غير متشاكلين .

1- Introduction

The graph isomorphism is one of the well-known long –standing open problem, some properties of identification has explored and investigated some uses of identification matrices in studying the graph isomorphism problem[1], for several special classes of graphs, polynomial-time algorithms are known , the most prominent being planar graphs[2] , an efficient parallel algorithm for planar graph isomorphism and several related problems has been presented[3] , the error-correcting graph isomorphism has been found useful in numerous pattern recognition applications[4] , in [5] the authors proposed an algorithm for solving the graph isomorphism problem. In this paper the permutation algorithm applied to solve the isomorphism problem.

2- Definition

Suppose $G_A = \langle V_A, E_A \rangle$ and $G_B = \langle V_B, E_B \rangle$ are two non weighted known oriented graphs. Where V_A, V_B are sets of their vertices and E_A, E_B are sets of their edges. $|V_A| = |V_B|$, $|E_A| = |E_B|$. Graphs $G_A = \langle V_A, E_A \rangle$ and $G_B = \langle V_B, E_B \rangle$ are said to be

+ **Received on 1/8/2007 , Accepted on 15/1/2009**

* **Assist Lecturer – Technical Institute of Al Suwarah**

isomorphism if there exist a bijection $\varphi : V_A \to V_B$ such that

$(i,j) \in E_A \Leftrightarrow (\varphi(i),\varphi(j)) \in E_B$ [5] .

Let us denote Isomorphism from graph G_A to G_B as $G_A \cong G_B$

Let A_0 be an adjacency matrix of graph G_A

i.e. $A_0 = (aij)$ where $aij = \begin{Bmatrix} 1 & if\,(aij) \in E_A \\ 0 & otherwise \end{Bmatrix}$

Let B_0 be an adjacency matrix of graph G_B .

3- Theorem[6] :

Two Graphs $G_A = \langle V_A, E_A \rangle$ and $G_B = \langle V_B, E_B \rangle$ with $|V_A| = |V_B|$ and $|E_A| = |E_B|$ are isomorphism $\Leftrightarrow$ for some ordering of their vertices their adjacency matrices are equal.

Proof: first, suppose $G_A \cong G_B$, that mean $A_O = B_O$ where A_O and B_O are adjacency matrices of G_A and G_B respectively.

Second, if $A_O = B_O$ that mean $G_A \cong G_B$. now if $A_O \neq B_O$, we can replace the second row of A_o with the first and replacing the second column with the first then compare A_o with B_o , if $A_O = B_O$, that our purpose, if not we replace the third row of A_o with the first row and replace third column with the first column then compare A_o with B_o and so on , finally if $A_O = B_O$ that implies $G_A \cong G_B$, Else. $G_A \neq G_B$ □

Note1 : there is n! different adjacency matrices of n vertices graph i.e. its equal to the number of permutations of n elements .

4- The Permutations Algorithm[7] :

this algorithm cans generate a permutation from given permutation .

Let A(n) be a vector of n positive integer , $n \geq 2$ (for example $A = [1 \quad 2 \quad 3 \quad 4]$) then we can generate a permutation from given permutation as the following

- *a- k=n-1*
- *b- if $A(k) < A(k+1)$ go to step d*
- *c- k=k-1 go to step b*
- *d- m=n*
- *e- if $A(m) > A(k)$ go to step g*
- *f- m=m-1 go to step e*
- *g- replace $A(k)$ by $A(m)$*
- *h- reserve the arrange of number from $A(k+1)$ to $A(n)$*
- *i- end*

5- example 1 : use the above Algorithm to generate a permutation from the following permutation:

$A = [7 \quad 3 \quad 6 \quad 5 \quad 4 \quad 2 \quad 1]$

solution

$n = 7$

- a- k=6
- b- k=2 since $A(2) = 3$ that is the first entry less than the next entry.

c- m=7

d- m=5 since $A(5)=4$ that is the first entry is greater than $A(2)$

e- the permutation become $\begin{bmatrix}7 & 4 & 6 & 5 & 3 & 2 & 1\end{bmatrix}$

f- the permutation become $\begin{bmatrix}7 & 4 & 1 & 2 & 3 & 5 & 6\end{bmatrix}$

6- the program of above algorithm :

the following program constructed in Matlab to list all permutation of vertices of any graph according to the above algorithm

```
L=input('input the number of vertices  L=')% this step to inter the number of vertices
g=1;
for i= L:-1:1 % this loop to compute g=L!
   a(i)=i;
   g=g*i;
end
a
for j= g-1:-1:1   % this loop to compare a(k) with a(k+1) according the Algorithm
   for i= L-1:-1:1
      k=i;
      if a(k)<a(k+1) break
   end
end
for i=L-1:-1:1  % this loop to compare a(m) with a(k) according the    Algorithm
   m=i+1;
   if a(m)>a(k) break
   end
end
r=a(k) ; z=a(m ) ; a(k)=z; a(m)=r; % this step to replace the number in the %location
                                     a(k)replace with the  number in %the location
                                     a(m) according to the %algorithm
for i=k+1:L  % this loop to compute the items of vector L from k+1 to L
   b(i)=a(i);
end
for i= (k+1):L
   r=(L-(i-(k+1)));
    a(i)=b(r);
end
a
end
```

7- Example : use the above program to list all permutations of graphs with four vertices

Solution : by using above program we obtain the following

L = 4

1	**2**	**3**	**4**
1	**2**	**4**	**3**
1	**3**	**2**	**4**
1	**3**	**4**	**2**
1	**4**	**2**	**3**
1	**4**	**3**	**2**

2	**1**	**3**	**4**
2	**1**	**4**	**3**
2	**3**	**1**	**4**
2	**3**	**4**	**1**
2	**4**	**1**	**3**
2	**4**	**3**	**1**
3	**1**	**2**	**4**
3	**1**	**4**	**2**
3	**2**	**1**	**4**
3	**2**	**4**	**1**
3	**4**	**1**	**2**
3	**4**	**2**	**1**
4	**1**	**2**	**3**
4	**1**	**3**	**2**
4	**2**	**1**	**3**
4	**2**	**3**	**1**
4	**3**	**1**	**2**
4	**3**	**2**	**1**

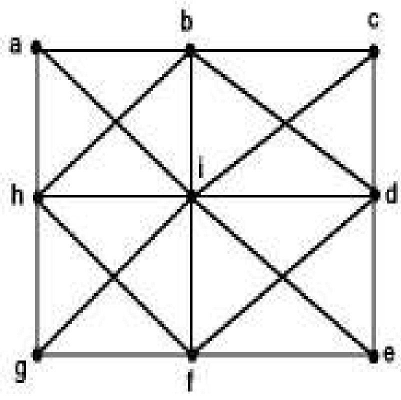
8- the program to calculate the isomorphism problem :

the following program constructed in Matlab to calculate the isomorphism problem according to the above algorithm

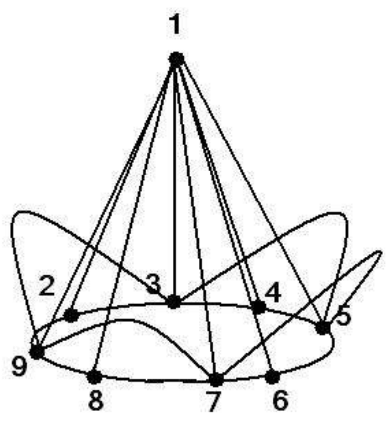
```
 L=input(' input the number of vertices of two graphs  L= ') % this for input the number
          of vertices
 for i=1:L  % this loop to input the elements
for j=1:L  % of two adjacency matrices C
if i<=j
    c(i,j)=input('input the elements of diagonal & the elements above its matrix C  ');
elseif i>j
    c(i,j)=c(j,i);
end
end
end
for i=1:L  % this loop to input the elements
for j=1:L  % of two adjacency matrices  D
if i<=j
d(i,j)=input('input the elements of diagonal & the elements above its matrix D  ');
elseif i>j
      d(i,j)=d(j,i);
end
end
end
g=1;
for i=L:-1:1
   a(i)=i;
    g=g*i;
end
for i=g:-1:1 % this loop for list all permutations L!
i=1:L; % this loop to compute the matrix  c associated one permutation
j=1:L;
```

```
a  % to list vector a
m=c(a(i),a(j))' % the matrix associated with permutation a
   d
if m==d  'the two graphs are isomorphism'
   break
else
     ' the two grapes are not isomorphism'
    end

   for i=L-1:-1:1   % this loop for compare a(k) with a(k+1) according to the Algorithm
      k=i;
  if a(k)< a(k+1) break
  end
end
for i=L-1:-1:1 % this loop for compare a(m) with a(k) according to the Algorithm
   m=i+1;
   if a(m) > a(k) break
   end
end
r=a(k);z=a(m);a(k)=z;a(m)=r; % this form to replace a(m) with a(k)
for i=k+1:L % this loop for compute the items of vector L from k+1 to L
   b(i)=a(i);
end
for i=(k+1):L % this loop to reciprocal the items from k+1 to L
   r=(L-(i-(k+1))); % if the items was 1234 so it become 4321 according to %the
                            Algorithm
   a(i)=b(r);
end

end
' the two graphs are not isomorphism'
  end
```

9- example : use the above program to determine $A \cong B$ or not .Where A and B are two graphs as show below .



A



B

solution : the adjacency matrices of graphs A and B as the following respectively :

$$A_o = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} , B_o = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

we observe that $A_o \neq B_o$, but making permutation of B_o by the program we obtain $A_o = B_o$ then $A \cong B$,

10- conclusion :

1- the algorithm and constructed program are good way to calculate the isomorphism problem .
2- the program need a long time to be performed when n becomes a large number because it depends on n! when it calculates the isomorphism problem .
3- we can develop the program to solve the isomorphism problem of multiple non planar graphs .

References

1- Lin C.,"Graph Isomorphism Detection with Identification matrices" ,*IEEE Transaction on Parallel and Distributed System* ,Vol.7,No.3 ,PP.308-319, 1996.
2- John E. H. ; J.K.W., " Linear time algorithm for isomorphism of planar graphs", *IEEE J* ,Vol 32,No.2,PP.95-109,1996
3- Joseph J ; Rao K., " Parallel Algorithm for planar Graph Isomorphism and Related problem " , *IEEE Transactions on circuts and systems* , Vol.35,No.3,PP.304-311,1988
4- Yuan K. ; Kuo Ch. ; Jorng T.," Genetic –Based search for Error –correcting graph Isomorphism " ,*IEEE J*, Vol.27,No.4,PP.588-597,1997.
5- Rashit T. ; Alexander V.,"The Direct algorithm for solving of the Graph Isomorphism Problem" , *AMS J*, Vol.32, No.1,PP.105-114, 2005
6- Wictor .Adamachik , *Graph Theory* , CRC Press ,London, 2005
7- James B. , *Introduction to Discrete Mathematics* , Wiley Publishing Company, 2002