Available online at www.worldscientificnews.com

World Scientific News

WSN 89 (2017) 71-81 EISSN 2392-2192

Application of Hamilton's graph theory in new technologies

Łucja Waligóra

Department of Human Resources Management, University of Economics in Katowice, Poland

E-mail address: lucja.waligora@ue.katowice.pl

ABSTRACT

There are many theories and articles about testing. People try to find the best ways to design computer systems for their later usability tests and functional tests. The article presents a somewhat mathematical approach to testing using graph theory and Hamilton's cycles. The inspiration for writing the article was the development of the text. Graphs as a decision support tool by Ewa Pospiech in the paper entitled " Elements of mathematics for economics and management students. Decisions, edited by J. Mika and A. Mastalerz – Kodzis.

***Keywords*:** graphs, Hamilton cycle, IT tests

1. INTRODUCTION

Testing your application is not an obvious and simple job. In practice, it is very time-consuming to "test" the application optimally before it hits the client. This problem is encountered by many IT companies and programmers. Recent software testing has become a very current topic. This is a problem that is trying to solve a powerful corporation. Obviously, no one can test the whole application from start to finish, because the test time will repeatedly exceed the life of the software or it would be unprofitable. Even automated tests are not able to provide full coverage of all possible combinations of functionality and program data. The question arises: How to design tests and how to write test cases to make the most optimally

(Received 14 October 2017; Accepted 02 November 2017; Date of Publication 24 November 2017)

sure that all features in the application work properly? How best to verify data transfer between states?

You can optimize testing when you limit yourself to designing a system and writing test cases in which we use the Hamilton's cycle in a data flow graph, state graph, or graph of data transfer between states defined and designed in a given IT system.

This approach can be applied at every level of the test and at every level of the system design, eg by using the V model (the test phase is related to the design and manufacturing phases).

2. HAMILTON'S GRAPHS OF THEIR PROPERTIES

The necessary information will be presented to you to understand the most important problem described in the next chapter, namely the use of Hamilton's graph theory when designing IT systems and test cases.

Graf is a basic object of the study of graph theory, a mathematical structure used to represent and study relations between objects. In simplified terms graph is a set of vertices that can be joined by edges so that each edge ends and starts at one of the vertices (Debnath, 2010).

The graph vertices can be numbered and sometimes represented as objects, while the edges can then represent relations between them. The vertices of the edge are called its ends. Edges may have a direction, and a graph containing such edges is called a directed graph or an orgraph (Ore, 1960). The edge of the graph may have a weight, i.e., the assigned number, which defines, for example, the distance between vertices (if, for example, graph is the representation of connections between cities). In the graph of the weight may be dependent on the direction of the edge (eg, if the graph represents the difficulty of moving around an area, the road will have higher weight than the hill) (Bryant, El-Zanati, Rodger, 2000).

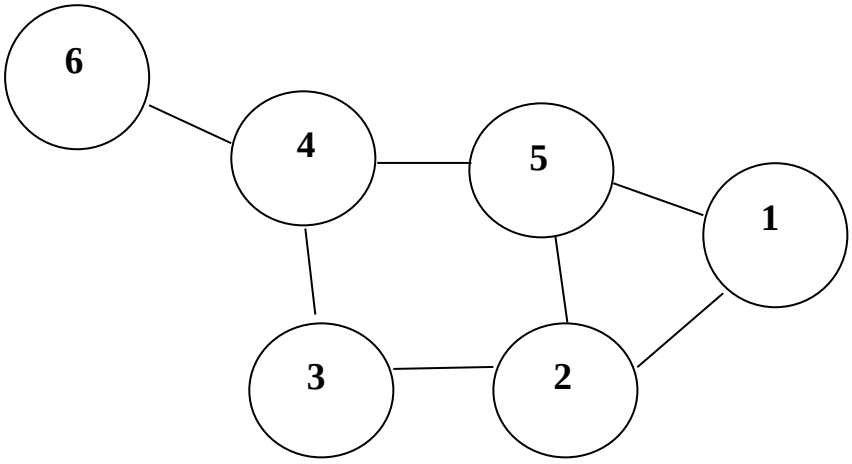
The first theorist and graphologist is Swiss physicist and mathematician Leonard Euler, who settled the question of the bridges of the Kings. The first use of the term "graph" is attributed to James Joseph Sylvester, a mathematician of English origin. There are several definitions of the Hamilton's cycle (Lesniak-Foster, 1977):

Definition 1. A Hamilton's cycle is a graph cycle in which every vertex of a graph is passed only once (except the first vertex). Hamilton's path is a graphical path that visits each vertex exactly once.

Finding a Hamilton's cycle with a minimum of edge weights is equivalent to solving the salesman problem. Hamilton's graphs are called Hamilton's. The Hamilton's graph is a graph discussed in graph theory, containing a path (path) passing through each vertex exactly once called the Hamilton's path. In particular, the Hamilton's graph is Hamilton's closed-loop graph (Harary, Palmer, 1973).

Definition 2. A coherent graph is a graph satisfying the condition that for each pair of vertices there exists a path that connects them (Example 1).

This graph is consistent, so as defined it has one consistent component. After removing edges 2-3 and 4-5, this graph is no longer coherent.

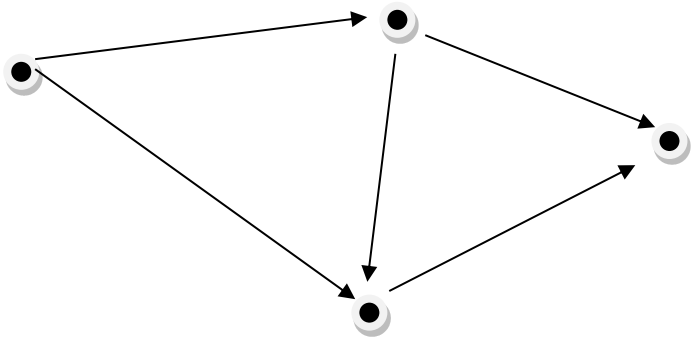


Example 1. Coherent graph

In order to better understand the following chapters, you need to know the following two definitions:

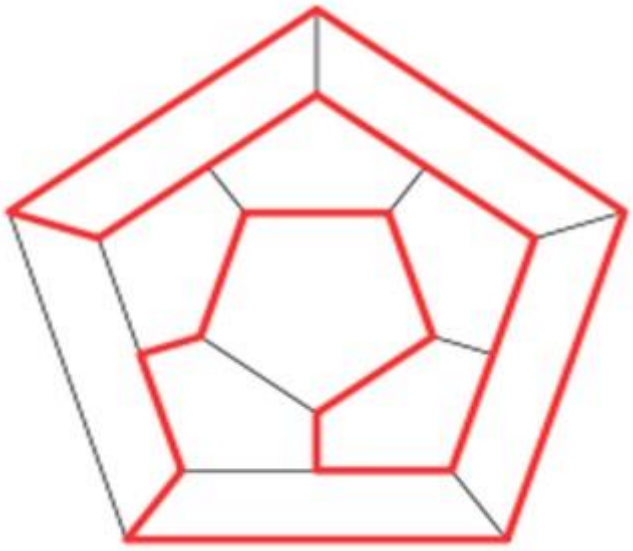
Definition 3. A coherent component of a non-directed G graph is a coherent graph G graph that is not included in a larger coherent graph G. In other words, a coherent graph component is a subgraph that can be "split" from the entire graph without removing the edges. The coherent graph has one consistent component (Lehmann, 2012).

Definition 4. A simple graph is called a graph without the own loop and multiple edges. Often the definition of graph (without adjectives) means simple graph (Gardner, 1957) (Example 2).



Example 2. Simple graph

Graf directed with the Hamilton path. The blue dots are the vertices of the graph, the arrows are the edges of the graph, and the Hamilton's line is marked with red (El-Zanati, Plantholt, Tipnis, 1995) (Example 3).



Example 3. Sample Hamilton's cycle in a regular dodecahedron graph

Let G be a graph, V (G) a set of vertices, E (G) a set of edges, | A | u_i single (in this case i-th) vertex of the graph a deg (v) degree of vertex (number of edges ending in it). Traditionally, V (G) = n and E (G) = m are written, and v {u, u} is a two-element set of vertices, used to denote the edges between v and u (DeLeon, 2000).

There are assertions based on the graph features available in linear time, and it is clear that the graph is Hamilton. Keep in mind that this is a one-sided implication - there are infinitely many Hamilton's graph graphs that do not have the following characteristics. These theorems are a mathematical picture of a fairly natural observation of the graph's properties - it is logical that the more edges in a graph, the "greater the chance" to find Hamilton's path among them. In short, the following statements say that the graph is Hamilton's if it has as many edges as the number of vertices. The most well-known theorems in this field are (Georges, Mauro, 1996):

Theorem 1 (Ore). If there is an inequality in the straight line G with n vertices n> 2, then: deg (v) + deg (u) ≥ n - 1, for each pair not connected directly to the vertex edges u and v, graph G has a Hamilton's cycle.

Theorem 2 (Dirac). If graph G has no loops or multiple edges (is a straight graph) and | V (G) | ≥ 3 and if deg (v) ≥ (| V (G) |) / 2 for every vertex in G, it is Hamilton's.

For a graph to have a Hamilton's cycle, it must be consistent - it is obvious. In the inconsistent graph, there are vertices between which there is no path, so you can not visit them. You can give a simple recursive algorithm to find all of Hamilton's cycles and paths, but it has the complexity of O (n!), Which makes it completely impractical for larger graphs (containing more than 30 vertices). The principle of our algorithm is as follows (Rubin, 1974): using a modified procedure we look at the graph starting from vertex 0 (the choice of vertex is irrelevant because the possible path or cycle of Hamilton must pass through all vertices of the graph). The processed tip is placed on the stack. It then checks to see if the

stack contains n vertices. If so, the Hamilton path is obtained. In this case, it is checked if the last vertex of the path is connected to the vertex with vertex 0. If so, the path creates the Hamilton's cycle - it outputs its contents from the stack (for the cycle additionally added at the end of the vertex 0), the last vertex is removed from and will retire to a higher level of recursion, where other Hamilton's paths or cycles will be considered (Debnath, 2010).

If the stack does not contain n vertices, the path finding procedure for all non-visited neighbors is recursively invoked. The vertex is removed from the stack and the information about its visit is deleted, and then it retires to the higher level of recursion.

The above algorithm produces Hamilton's paths and cycles in the opposite direction to their existence. For an undirected graph it does not matter. In the directed graph, this order should be reversed. The stack can be easily implemented in a dynamic array n element. It is then possible to read the vertices of the cycle in the order they are located.

3. HAMILTON SEARCH ALGORITHM

An example of an algorithm that finds the Hamilton's path in a graph may be (Rahman, Kaykobad, 2005) (Table 1):

1) Hamilton's recursive function (n, graph, v, visited, S)

2) Input: n - number of vertices in the graph, graph - given in any way chosen, the algorithm does not specify. v - current vertex of the graph, visited - n elementary visit table, S - stack of vertices,

3) Exit: All Hamilton cycles and paths in the graph,

4) Accessory: test - a logical variable for testing a Hamilton cycle or path, u - vertex of the graph.

Table 1. Number of steps in the algorithm (Watkins, 2004).

K01:	S.push (v)	places v on the stack
K02:	If S contains less than n vertices, then go to K10	is Hamilton's path
K03:	test ← false	assumes no Hamilton cycle
K04:	Adjacent neighbor at vertex v, perform K05	looking at neighbors v
K05:	If u = 0, then test ← true and go to K06	
K06:	If test = true, then write "Hamilton cycle:" otherwise write "Hamiltonian Path:"	
K07:	Write content S without deleting data from S	

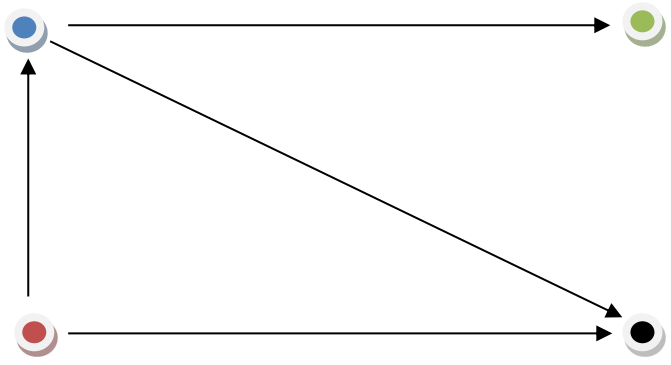
K08:	If test = true, then write 0	for the cycle to write the starting vertex
K09:	Go to K14	
K10:	Visited [v] ← true	means the vertex is visited
K11:	For each neighbor at vertex v, execute K12	looks at neighbor vertices v
K12:	If visited [u] = false, then Hamilton (n, graph, u, visited, S)	for non-visited neighbors perform a recursive call
K13:	Visited [v] ← false	retracts from vertex v
K14:	S. pop ()	vertex v is removed from the stack
K15:	Finish	

4. APPLICATION OF THEORY IN PRACTICE

4. 1. Use of Ore Theorem and Hamilton's Path in Test Design

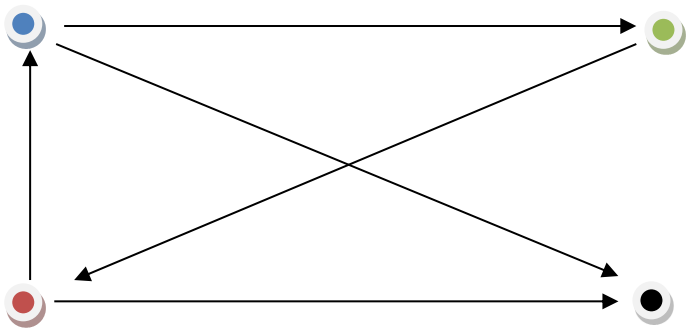
Testerzy.pl is a portal belonging to the 21CN Group, offering services and training in the area of software testing. This is a compendium of knowledge, industry know-how, a source of the latest news from the testing world, and an advisory center. This company operates in Katowice.

It is assumed that a function is designed which can be represented by the following graph (Example 4):



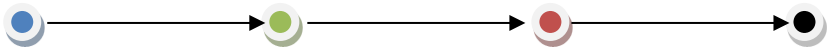
Example 4. This graph is not Hamilton's graph

This graph is not a Hamilton's graph. The number of vertices in this graph is equal to 4. You can use the Oreia theorem to add edges so that the assumption of the theorem is true. Just add one edge and the graph will become Hamilton's graph . The edge can be added as shown in the following graph (Example 5).



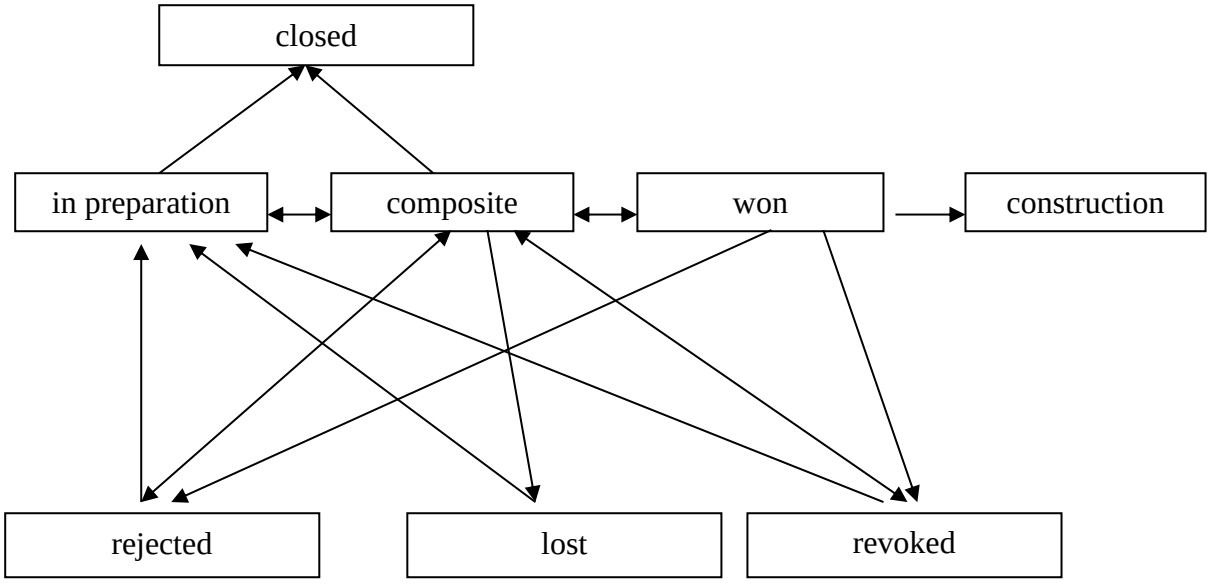
Example 5. Hamilton Graf

This is possible if adding a new track does not interfere with the design of the system or is not in conflict with the logic of the system. The algorithm that determines the Hamilton's cycle will indicate the following path (Example 6).



Example 6. The Hamilton cycle

The following data transfer problem will now be resolved. The graph below shows how the status of tender projects in construction companies can change (Example 7).

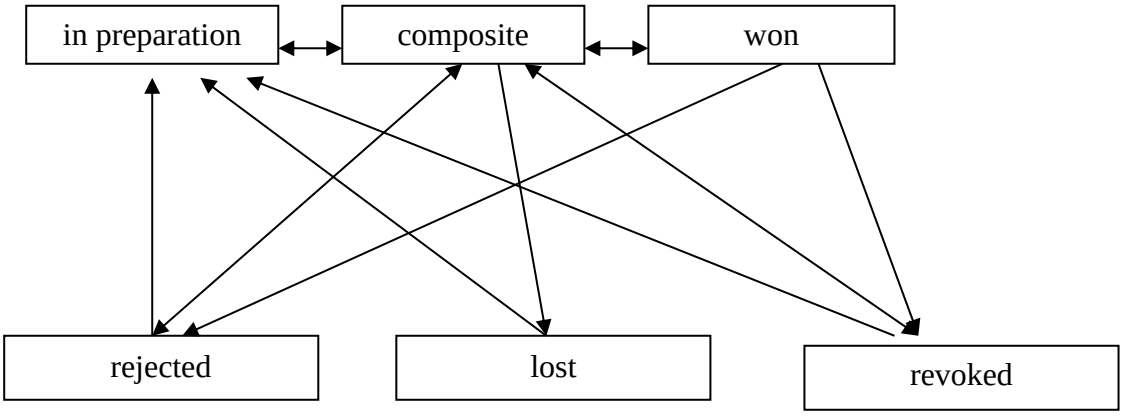


Example 7. Status of tender projects in construction companies

This graph is not Hamilton's graph. Graph states can be treated as graph vertices. To test data transfer between different statuses, you need to pass through some nodes several times. However, one may wonder whether it is worth adding one more path so that it is not incompatible with the logic of construction companies. You can propose this approach.

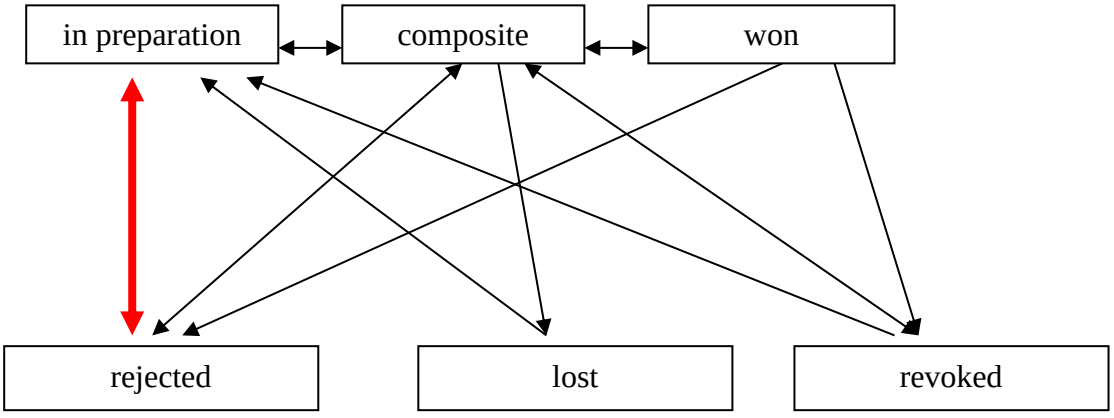
1. The premise and the winning tender can be treated as a single node, since documents related to the construction itself are completely different documents than the tender project and are no longer the status of the tender. At the moment of construction, a new blank document is created that needs to be filled out and saved. With this approach, graphs are reduced to seven vertices.
2. Closed tenders are omitted, because it is a tender from which the company resigns. It only stays in the database and nothing is going on with it (data from such a tender is no longer transferred after closing).

Thinking in this way, you can reduce the graph to six vertices. The data flow graph will assume the following form (Example 8):



Example 8. Graph of data flow in tender projects in construction companies

Now just add the edge as shown below in the image and graph is already Hamilton's graph (Example 9).



Example 9. Status of tender projects in construction companies - graphic Hamilton's

Added a possibility to reject the project in preparation. The Hamilton's path for this graph looks like this:

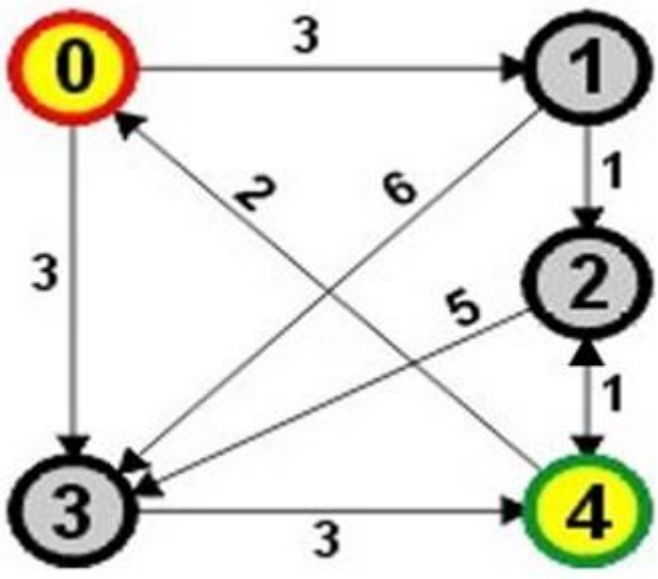
Lost → In preparation → Rejected → Composite → Win → Revoked

If you want to test the construction and close the tender, you will need to write test cases other than the Hamilton cycle. The closing of the tender can be solved in such a way that the edge from the closed tender is closed and the graph will be Hamilton's graph at seven points. This approach can sometimes be very supportive of testing, but it is always a good idea to consider whether it is profitable - since adding a path is a new feature.

4. 2. Weighted Hamilton's graph

We should consider what would happen if there were more than one Hamilton graphs in the states in the system or between data flows. In practice, there is always a path in the application that the client performs more often than other activities.

The path selection problem can be solved by introducing the scales for the edges. The question of whether the sum of scales is the largest or the smallest depends on the context. This is the following example (Example 10).



Example 10. Weights in Hamilton's graph

a) If you want the smallest amount of scales, then choose the path:

1 → 2 → 4 → 0 → 3,

where the sum of weights is 7,

b) If you want the maximum weight, then the path is:

0 → 1 → 3 → 4 → 2,

where: the sum of the weights is equal to 13.

An algorithm that checks the sum of weights can be analogous to an algorithm presented earlier, except that it must have an additional condition that checks the sum of the paths after finding the Hamilton's path.

5. CONCLUSION

Graf is a basic object of the study of graph theory, a mathematical structure that serves to represent and study relations between objects. In simplified terms, a graph is a set of vertices that can be joined by edges so that each edge ends and starts at one of the vertices.

The task of the tester (according to the ISTQB ideology) is to say that it "works" rather than pessimistic approach to the application, in a way that nothing works well. So, designing tests and applications for Hamilton's existence in it will save time when testing data transfer between states in an application. If the test results prove positive, then the data is moving correctly and there are no gaps or defects between transition states.

The existence of the Hamilton's cycle (as in the tendering scheme) will also allow the design of an automated test so that once the data is entered and finally the test conditions are created.

References

[1] F. Harary, E. M. Palmer, Graphical Enumeration. New York: Academic Press, 1973, p. 219.

[2] O. Ore, A Note on Hamiltonian Circuits. *Amer. Math. Monthly* 67 (1960) 55.

[3] F. Rubin, A Search Procedure for Hamilton Paths and Circuits. *Journal of the ACM* 21 (1974) 576-580.

[4] N. L. Biggs, T. P. Kirkman, *The Bulletin of the London Mathematical Society* 13 (2) (1981) 97-120.

[5] J. J. Watkins, Chapter 2: Knight's Tours, Across the Board: The Mathematics of Chessboard Problems, Princeton University Press, 2004, pp. 25-38.

[6] M. Gardner, Mathematical Games: About the Remarkable Similarity between the Icosian Game and the Towers of Hanoi. *Sci. Amer.* 196 (1957) 150-156.

[7] M. S Rahman, M. Kaykobad, On Hamiltonian cycles and Hamiltonian paths. *Information Processing Letters* 94 (2005) 37-41.

[8] J. Moon, L. Moser, On Hamiltonian bipartite graphs, *Israel Journal of Mathematics* 1 (1963) 163-165.

[9] M. DeLeon, A study of sufficient conditions for Hamiltonian cycles, *Rose-Hulman Undergraduate Math Journal*, 1, 1, (2000) 56-62

[10] L. Debnath, A Brief Historical Introduction to Euler's Formula for Polyhedra, Topology, Graph Theory and Networks. *International Journal of Mathematical Education in Science and Technology*, 41, 6, (2010) 769-785

[11] D. E. Bryant, S. El-Zanati, C. A. Rodger, Maximal sets of Hamilton cycles in $K_{n,n}$**.** *Journal of Graph Theory* 33, 1, (2000) 25-31

[12] L. Lesniak-Foster, Some recent results in hamiltonian graphs**.** *Journal of Graph Theory*, 1, 1, (1977) 27-36

[13] L. Lehmann, The stationary distribution of a continuously varying strategy in a class-structured population under mutation-selection-drift balance**.** *Journal of Evolutionary Biology* 25, 4, (2012) 770-787

[14] S. I. El-Zanati, M. J. Plantholt, S. K. Tipnis, Factorization of regular multigraphs into regular graphs**.** *Journal of Graph Theory* 19, 1, (1995) 93-105

[15] J. P. Georges, D. W. Mauro, On the size of graphs labeled with a condition at distance two**.** *Journal of Graph Theory* 22, 1, (1996) 47-57