SLAC-PUB-261
January 1967

A METHOD FOR FINDING HAMILTON PATHS AND KNIGHT'S TOURS*

by

Ira Pohl

Stanford Linear Accelerator Center
Stanford University, Stanford, California

ABSTRACT

The use of Warnsdorff's rule for finding a Knight's tour is generalized, and applied to the problem of finding a Hamilton path in a graph. A graph-theoretic justification for the method is given.

(To be submitted to Communications of the Association for Computing Machinery)

*Work supported by the U. S. Atomic Energy Commission.

1. Introduction

A path in a graph is a Hamilton path if and only if it goes through each node of the graph once and only once.[1] These paths were named for Sir William Hamilton who invented and analyzed a game to find these paths through the vertices of a regular dodecahedron. A problem of this type is the classic knight's tour problem (Fig. 1) on a chessboard. The knight is placed on a square and must cover the whole board, moving to each square once and only once.

Many mathematicians[2] proposed specialized methods for finding a knight's tour,[3] but general methods for Hamilton paths on graphs and similarly for knight's tours on any shape or size board are lacking. In 1823, H. Warnsdorff proposed the following rule for knight's tours:

> Select the move which connects with the fewest number of further moves, providing this number is not 0. If a tie occurs it may be broken arbitrarily.

This rule proved unusually successful and generally applicable[4] until a few carefully constructed counterexamples showed that in case of ties some of the options failed to find knight's tours. However, not much was done in analyzing the rule and its failures because of the large computational effort involved in using the rule. The rule was justified on the common sense grounds that it eliminated bad squares as quickly as possible, leaving many possibilities for success.

2. Problems and Background

The finding of a Hamilton path in a graph is generally attacked with combinatoric methods, which for large richly connected graphs are not feasible. Warnsdorff's rule is a simple computational rule for finding knight's tours. It presents an attractive method for finding Hamilton paths in richly connected graphs. In understanding the reasons for its successes and failures an improved generalization of the rule has been devised and implemented, and the general theory of the method has been examined.

The problem of a knight's tour can be displayed as a graph for which the connections (edges) are knight's moves. Consider a 3 by 3 board (Fig. 2). It is readily described as a graph with the center square unconnected. Therefore, there can be no tour even though the remainder of the graph is two-connected[5] and has a path.

The 4 by 4 board is more complicated, involving 16 squares and 48 connections. Although all the squares are at least two-connected (Fig. 3a) and the graph is richly connected, the graph has no Hamilton path. The graph may be broken into 4 factors:

$$\{(1,1),(2,3),(4,4),(3,2)\}, \quad \{(1,4),(3,3),(4,1),(2,2)\},$$
$$\{(4,2),(2,1),(1,3),(3,4)\} \text{ and } \{(2,4),(4,3),(3,1),(1,2)\}.$$

However, there are only enough transition squares to link three of them. The maximum path is 15 squares (Fig. 3b) and Warnsdorff's rule does indeed find it.

The first non-trivial square board to have a tour is a 5 by 5 on which the knight must start on a square such that

$$(\text{row number}) + (\text{file number}) = (\text{even integer})$$

because of parity.[6] Traditionally the 8 by 8 board or ordinary chessboard has gotten the most attention. First let us look at the number of connections from each square of the board (Fig. 4). They range from two to eight, totaling 336 connections, and giving an average of slightly over five connections per square. This average increases asymptotically to eight as the board size increases, but the corner squares remain two-connected.

Warnsdorff's rule maximizes the number of connections remaining at the current point in the path. A knight's tour is a path on the chessboard which maximizes the number of connections at the 63rd move, i.e., there is one remaining move and there can be at most one at this point. The move tree of the various paths of the knight is exponential in nature, and maximization at the 63rd level is not computationally possible. Maximizing the current level propagates down the move tree, and therefore is likely to maximize the later levels. Warnsdorff's rule is just an approximation to the algorithm of searching the complete tree for a connection at the 63rd level.

In implementing Warnsdorff's rule on a B5500 in extended Algol, the rule was tried starting from each square of an 8 by 8 board and found to fail at least once for each of five fixed orderings[7] of moves. In fact, the rule failed twelve times with a median of three failures for a given move ordering.[8] This high rate of failure was a disappointment as it seemed that for some fixed ordering of moves Warsndorff's rule could generate a knight's tour from each square of the board. Initially, the rule was investigated with a view to correcting these failures.

3. Generalization and Analysis

To improve on Warnsdorff's arbitrary selection in case of ties, the following tie-breaking method was proposed and tested. For each tie move, sum the number of moves available to it at the next level and pick the one yielding a minimum. In theory this can be carried through as many levels as necessary for tie-breaking. For computational purposes the breaking of ties of the ordinary Warnsdorff's rule (first level minimization) was only carried one more level (second level tie-breaking). In symmetric positions, ties cannot be broken at any level. In unsymmetric positions, many-level tie-breaking methods are not computationally practical because of exponential growth of the move tree. The second level method always yielded a knight's tour from all squares of the 8 by 8 board. The generalized method was tried with several of the positions and move orderings which had previously failed and it was successful in each case. The ordinary Warnsdorff's rule on the chessboard had occasionally failed and the improvement brought complete success for the knight's tour problem.

This modification to Warnsdorff's rule is a generalization of the original concept. In terms of the connections of the unreached squares it is again a maximizing rule, with sufficient power always to work with a knight on a chessboard. The generalized rule (for method of order k) is:

> Consider all paths of k moves and count the remaining number of connections for each path. Select the first move of the path whose number is maximum, providing this path is not a dead end. Ties are broken by going to $k + 1$ moves.

In our case k has been 1.

The argument for Warnsdorff's rule has previously been an appeal to common sense. However, the maximization of connectivity at the first level as an approximation to a full search of the move tree is a firmer explanation and yet is not enough. While mathematicians have searched for a proof of Warnsdorff's conjecture, modified to account for any solution from a tie point, the rule may only be justified as above. A graph-theoretic proof would have to account for Warnsdorff's rule working in general, which it does not. A typical counter-example is Fig. 5, a graph with a Hamilton path which cannot be found by application of Warnsdorff's rule. To find it would require the generalized rule with $k = 9$. Certain nodes in a graph,[9] independently of their connectivity, play crucial roles as transition squares between otherwise independent components of the graph. (To see this, re-examine the 4 by 4 graph in Fig. 3a.)

4. Conclusions

Warnsdorff's generalized rule is a powerful yet practical method of finding a Hamilton path in a graph. In terms of a knight's tour on a chessboard a random generation of moves has virtually no chance of success, but the second level tie-breaking program was always successful. The program was also used to find several Hamilton paths in an especially tricky regular graph[10] of degree 3 with 46 nodes, proposed by W. Tutte. In Fig. 6 the nodes are labeled by their move number in a Hamilton path found by the program.

The time needed to find a path is directly proportional to the total number of edges in the graph. In contrast an enumeration procedure would be impossibly long and some recent methods[11] using Boolean connection matrices and linear systems of equations are useful only for sparse graphs. The modified Warnsdorff's method has proved so powerful in practice that it has generated solutions for 20 by 20 and 40 by 40 boards. The method may be viewed as an approximation to a full search on the move tree. The higher order the method, the better the approximation but the longer the computation.

There is much to be explored theoretically and empirically in using maximizing methods in graph theory. Certainly other tie-breaking modifications warrant attention. One could check the next level for the square with minimum connectivity instead of the sum as used in the program (see RecurNumofmov in appendix). Alternatively one could use a hybrid method in which the sum would be used unless a square is two-connected, in which case this path would be followed. A further possibility is to try Warnsdorff's rule and, if it fails, backtrack to where it is practical to derive the rest of the solution by using the Boolean system of equations for the graph. At this point the remaining unreached subgraph should be sparse. Some order of the generalized method outlined above is a useful basis method for finding Hamilton paths in graphs.

5. Acknowledgment

The author wishes to express his appreciation for the advice and encouragement of Professor William F. Miller of Stanford University and Stanford Linear Accelerator Center in preparing this paper.

APPENDIX

```
procedure HAMILTONPATH (BOARD, ROW, FILE);
    value ROW, FILE; integer ROW, FILE;
    integer array BOARD;
    comment BOARD is a collection of nodes through which is generated
    a Hamilton path (connection of all the nodes passing through each
    node once and only once). The path is started at the node specified by
    BOARD [ROW, FILE] ;
begin
    integer i, j, nummov, move, min, T1, T2, T2L2; boolean flg;
    integer array NextR, NextF [1:8] ;
    comment The Hamilton paths to be found will be knight's tours where 8
    is the maximum number of moves (connections);
procedure Listofmov (CR, CF, XR, XF, II);
    value CR, CF; integer CR, CF, II;
    integer array XR, XF;
    comment From the current position specified by CR, CF this procedure
    generates in XR [i], XF [i] a list of the coordinates of the possible moves
    and their number II. The B5500 program used a CASE statement which for
    Algol 60 purposes is translated to a switch list;
begin
    integer i, rr, ff;
    switch case: = L1, L2, L3, L4, L5, L6, L7, L8;
    II:= 0;
    for i:=1 step 1 until 8 do
    begin
       go to case [i] ;
L1:    rr:=CR-1;  ff:=CF+2;  go to check;
L2:    rr:=CR-1;  ff:=CF-2;  go to check;
L3:    rr:=CR+1;  ff:=CF+2;  go to check;
L4:    rr:=CR+1;  ff:=CF-2;  go to check;
L5:    rr:=CR+2;  ff:=CF+1;  go to check;
L6:    rr:=CR+2;  ff:=CF-1;  go to check;
L7:    rr:=CR-2;  ff:=CF+1;  go to check;
L8:    rr:=CR-2;  ff:=CF-1;
```

```
check:  comment check whether a legal connection or move;
        if BOARD [rr, ff] = 0 then
        begin II : =II+1; XR[II] :=rr; XF [II] :=ff end
   end loop i
end procedure Listofmov;

integer procedure Numofmov (CR, CF);
        value CR, CF; integer CR, CF;
        comment This procedure is a simplification of Listofmov for efficiency.
        It is used only to obtain number of legal moves;
begin
        integer i, ii, rr, ff;
        switch case: = L1, L2, L3, L4, L5, L6, L7, L8;
        ii:= 0;
        for i:=1 step 1 until 8 do
        begin
           go to case [i] ;
L1:        rr:=CR-1;  ff:=CF+2;  go to check;
L2:        rr:=CR-1;  ff:=CF-2;  go to check;
L3:        rr:=CR+1;  ff:=CF+2;  go to check;
L4:        rr:=CR+1;  ff:=CF-2;  go to check;
L5:        rr:=CR+2;  ff:=CF+1;  go to check;
L6:        rr:=CR+2;  ff:=CF-1;  go to check;
L7:        rr:=CR-2;  ff:=CF+1;  go to check;
L8:        rr:=CR-2;  ff:=CF-1;
check:  if BOARD [rr, ff] =  then  ii:=ii+1
        end loop i;
        Numofmov:=ii
end procedure Numofmov;

integer procedure  RecurNumofmov (CR, CF, Level);
        value CR, CF, Level; integer CR, CF, Level;
        comment This is a recursive routine to the depth Level for counting
        the nodes of the move tree;
```

```
begin
      integer  tt, i, nn;
      integer array  ra, fa  [1:8] ;
      BOARD [CR, CF]  :=1;
      if  Level=1  then  RecurNumofmov:=Numofmov (CR, CF)
      else
      begin
         Listofmov (CR, CF, ra, fa, nn);
         tt:=0;
         for  i:=1  step  1  until  nn  do
             tt:=tt + RecurNumofmov (ra [i] , fa [i] , Level-1);
         RecurNumofmov:=tt
      end;  BOARD [CR, CF] :=0
end procedure  RecurNumofmov;

comment  The program deals with knight's tours on a chessboard.  Warnsdorff's
rule is applied and the improvement of reapplying the rule to resolve ties is used
and is sufficient for generating knight's tours on chessboards;
for  i:= -1, 0, 9, 10  do
begin  BOARD [i, j] :=BOARD [j, i] := -1 end;

comment  Initializing the boundaries of the BOARD to -1 prevents moving there.
The BOARD proper is initialized to 0;
for i:= 1  step  1  until  8  do
for j:= 1  step  1  until  8  do  BOARD [i, j] :=0;
comment  Initialize the starting position;
min:=0;  T2:=1;  BOARD [ROW, FILE] :=move:=1;
for move:=2  step 1  while  min ≠ 99  do
begin
      min:=99
      Listofmov (ROW, FILE, NextR, NextF,  nummov);
      for  i:=1  step  1  until  nummov  do
      begin
         T1:=Numofmov (NextR [i ], NextF [i] );
         if  (min    T1)    (T1≠0)  then
         begin  flg:=true; T2:=i; min:=T1  end
         else  comment  Above is Warnsdorff's rule;
```

```
        begin comment Here is the improvement;
          T2L1:=RecurNumofmov (NextR[i], NextF[i], 2);
          if flg then
            T2L2:=RecurNumofmov (NextR[i], NextF[i], 2);
          if (T2L2   T2L1)   (T2L1≠0) then
          begin T2L2:=T2L1; flg:=false; T2:=i end
        end tie breaking improvement;
        end loop i;
        if min≠99 then
    begin
        ROW:=NextR[T2]; FILE:=NextF[T2];
        BOARD[ROW, FILE]:=move; move:=move+1
        end
    end loop move;
    OUTPUT (BOARD);

    comment Use an output procedure to print results;
end procedure HAMILTONPATH alias the knight's tour;
```

Footnotes and References

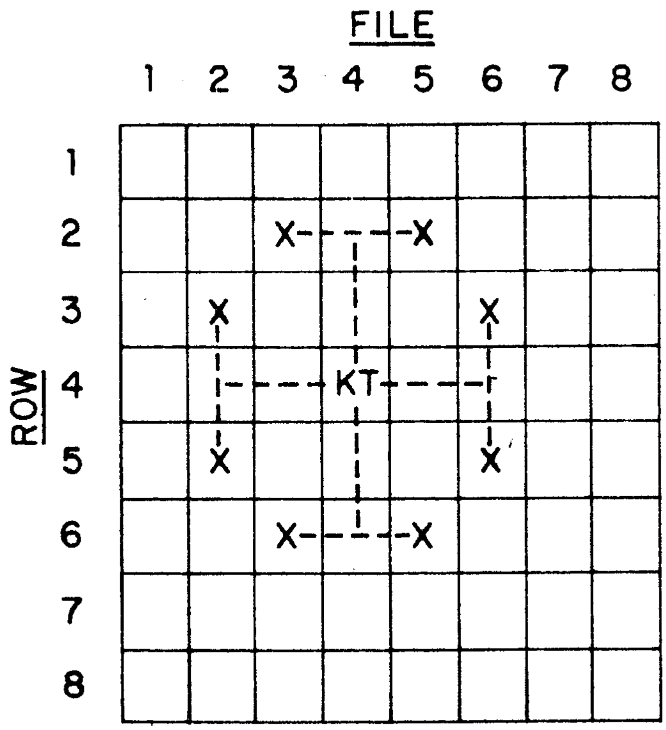
1. C. Berge, The Theory of Graphs and Its Applications, pp. 107-118 (John Wiley and Sons, New York, 1962). This book is useful for the general theory of graphs.
2. Euler, Vandermonde, de Moivre, Roget, and others.
3. For general material on the knight's tour problem see the entertaining book of W.W.R. Ball.

   W.W.R. Ball, Mathematical Recreations and Essays, pp. 174-184 (McMillan Co., New York, 1947).
4. It was called inelegant by many mathematicians because of its computational nature.
5. Two-connected means that in one move from a square, two other squares may be reached. It correxponds to the degree of a node in an undirected graph without multiple edges (see Berge, Theory).
6. A board is ordinarily thought of as having white and black squares. On boards of odd length one color has to have an extra square. Knight moves alternate between colors, and therefore on a board with an odd number of squares the knight must start on a square of the majority color to be able to complete a tour.
7. In cases where more than one minimum exists, the order in which the squares are evaluated determines the next move. The first square encountered with the minimum value is used.
8. Ball remarks, "it would require exceptionally bad luck to happen accidently, [failures of Warnsdorff's rule] on such a route." Ball, Recreations, p 181.

9. Certain nodes in a graph may be articulation points. These are nodes that, if removed from the graph, leave an unconnected subgraph. In Fig. 5, node 9 is an articulation point.

10. A regular graph is a graph where all the nodes have the same degree. In terms of terms of squares, each square has the same number of connections.

11. Berge, Theory, p. 115.



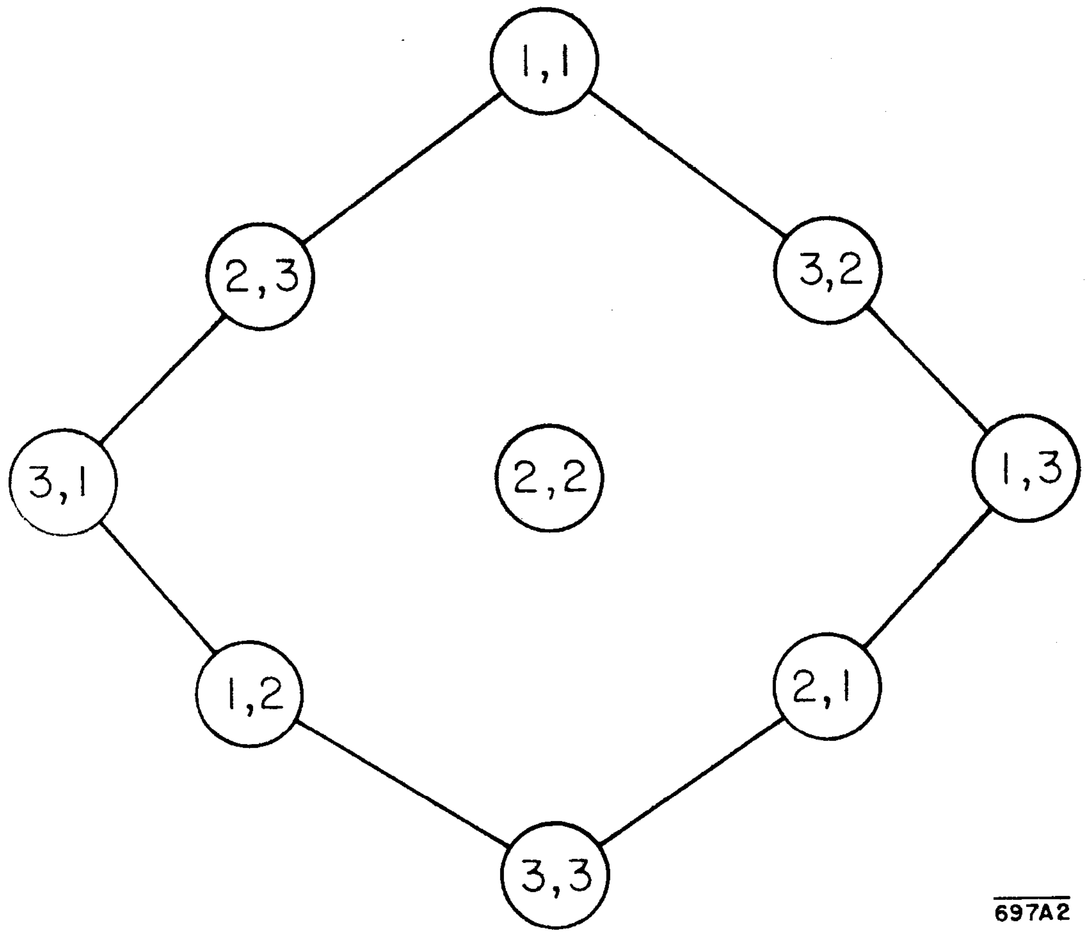
LEGAL KNIGHT MOVES

3	12	51	28	5	10	63	32
52	27	4	11	64	31	6	9
13	2	57	50	29	8	33	62
26	53	42	1	56	59	30	7
43	14	55	58	49	20	61	34
54	25	46	41	60	35	38	19
15	44	23	48	17	40	21	36
24	47	16	45	22	37	18	39

A KNIGHT'S TOUR

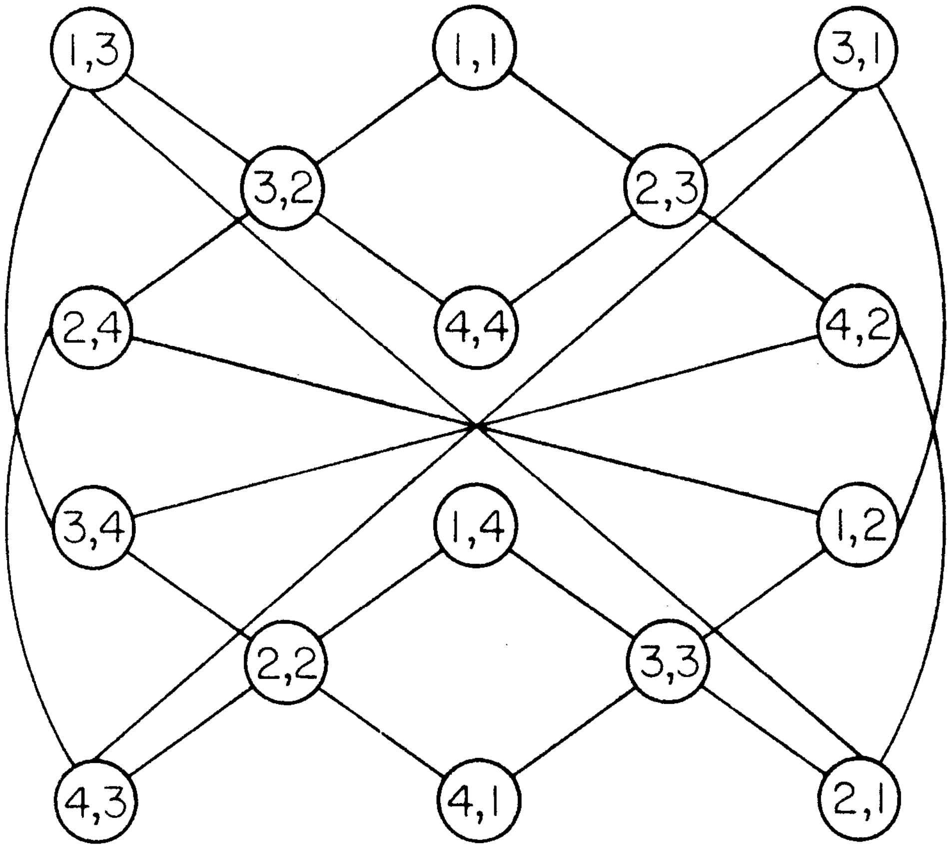
697A1

Fig. 1



Graph of a 3 by 3 Board

FIG. 2



a

Graph of a 4 by 4 Board

FILE

ROW	1	2	3	4
1	1	14	7	10
2	8	11	4	13
3	15	2	9	6
4		5	12	3

b.

679A3

15 Move Path

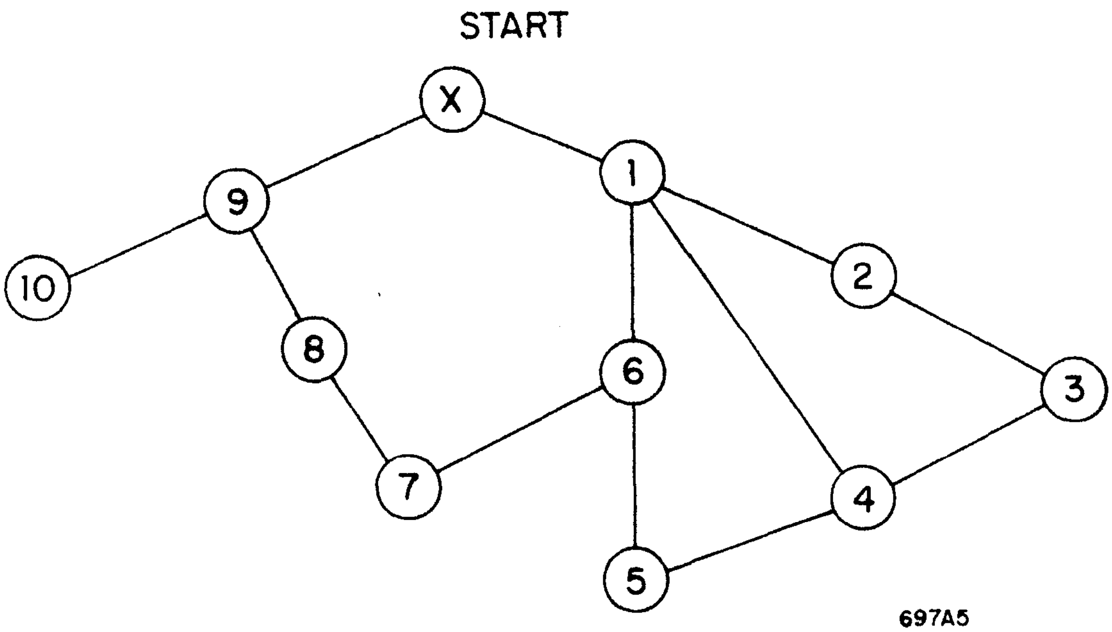
FIG. 3

2	3	4	4	4	4	3	2
3	4	6	6	6	6	4	3
4	6	8	8	8	8	6	4
4	6	8	8	8	8	6	4
4	6	8	8	8	8	6	4
4	6	8	8	8	8	6	4
3	4	6	6	6	6	4	3
2	3	4	4	4	4	3	2

697A4

Connectivity of a Chess Board
for a Knight

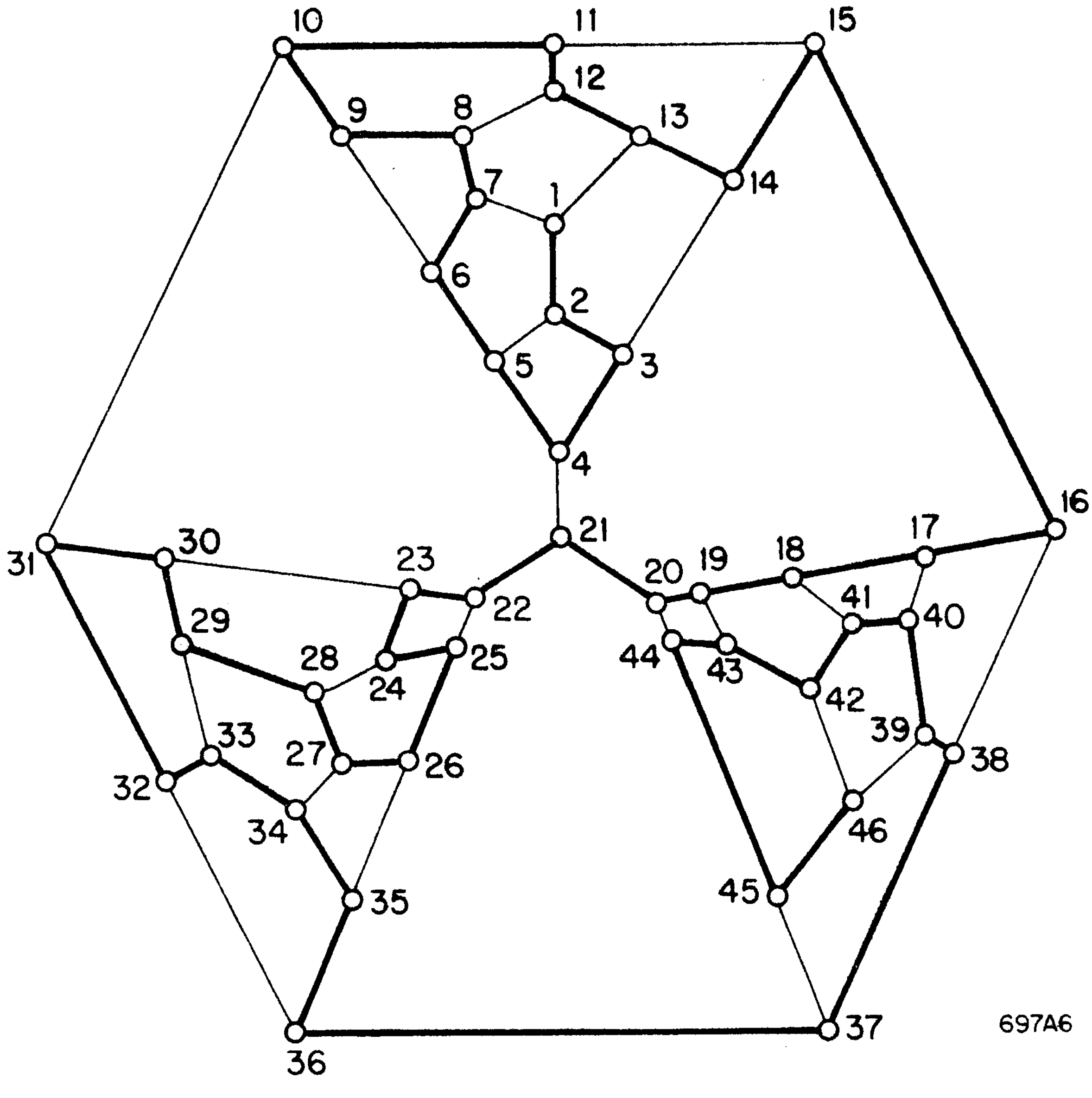
FIG. 4



Counterexample for the generalized graph application of Warnsdorff's rule.

Warnsdorff's rule selects node 9 which would make a Hamilton Path impossible.

FIG. 5



Tutte's Graph Planar of Degree 3

FIG. 6