IDC – Algorithms – Homework 2

NOTE: the general homework guidelines published in homework 1 are valid for all homework assignments in this course.

Due date: Sunday 20/11/2022

Problem 1 (16 pts.):

Find an Euler Path in the following undirected graph, represented by a matrix. Whenever you have multiple ways to advance, prefer vertices with lower labels. Describe the stages performed in the path construction (the sub-paths constructed in each iteration, and how they are merged).

	a	b	c	d	e	f
a	**0**	**1**	**1**	**0**	**0**	**0**
b	**1**	**0**	**1**	**0**	**0**	**0**
c	**1**	**1**	**0**	**0**	**1**	**1**
d	**0**	**0**	**0**	**0**	**1**	**1**
e	**0**	**0**	**1**	**1**	**1**	**0**
f	**0**	**0**	**1**	**1**	**0**	**0**

First iteration:
Start with vertex a. Trace edge (a,b), result in path a → b. Trace edge (b,c), results in a→b→c. Trace edge (c,a), results in cycle a→b→c→a. Ends the first iteration with cycle abca

Second Iteration:
Pick c as the starting vertex as it is the lowest labeled edge from the previous firs cycle which has untraced edges. Trace (c,e), results in path c→e. Trace (c,d), results in path c→e→d. Trace (d,f), results in path c→e→d→f. Trace (f,c), results in cycle c→e→d→f→c. Ends seconds iteration with cycle cedfc. We then merge the first cycle and second cycle to result in cycle abcedfca.

Third Iteration:
Pick vertex e. Trace (e,e), results in cycle e→e. Ends third iteration. Merge this cycle with previously merged cycle, to result in cycle abceedfca. This is the final euler cycle.

Problem 2 (20 pts.):

Let $G = (V, E)$ be a **directed** graph containing a Euler Cycle. Let $C = v_0 v_1 \cdots v_k$ be a cycle in G of length $k < |E|$ in which every edge appears at most once. Prove that if we remove the edges of C from G, there exists v_i, $0 \leq i < k$, such that $deg_{out}(v_i) > 0$.

G has an Euler cycle, and an Euler cycle can be constructed from merging sub-cycles such as C and other sub-cycles. Choose C to be one of the sub-cycles that can merge with other sub-cycles to construct the Euler cycle in G, and let one of the other sub-cycles be denoted by D. There must be a connecting (shared) vertex between cycle C and D which is where the merging is done. Lets denote this vertex by Vi.

If C is deleted from the euler cycle, D still exists. D is a full cycle with vertex Vi still a part of that cycle. Therefore out degree of Vi is bigger than zero.

Problem 3 (24 pts):

1. Prove that a graph G is bipartite only if every cycle has even length. (Note: This statement is actually an if and only if statement, but in this exercise you are only asked to prove the only if direction).

Assume every cycle in G has an even length. We will show that the following partition results in a valid bipartite graph for G. Choose an arbitrary vertex in G and denote it as v_i.

V1 = all v ∈ V s.t the shortest path from v to v_i is even

V2 = all v ∈ V s.t the shortest path from v to v_i is odd

ATC that V1 and V2 are not valid partitions that result in a bipartite graph. That implies that there exists an edge (x,y) s.t x,y ∈ V1 or x,y ∈ V2 (i.e they're from the same partition). Therefore a cycle is formed from the path from x to v_i, the path from y to v_i and the edge (x,y).

Case 1: x,y ∈ V1

The length of the path from x to v_i is even. The length of the path from y to v_i is even. The length of (x,y) is 1. Therefore, after adding up the lengths, the cycle length is odd. Contradiction

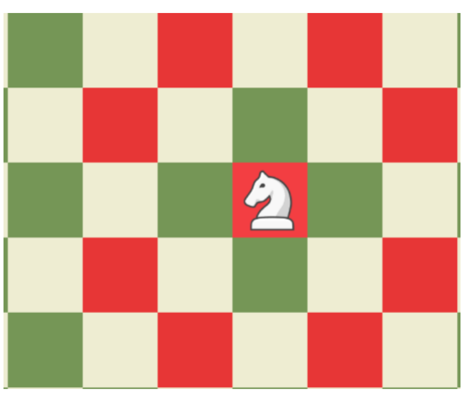
Case 2: x,y ∈ V2

The length of the path from x to v_i is odd. The length of the path from y to v_i is odd. The length of (x,y) is 1. Therefore, after adding up the lengths, the cycle length is odd. Contradiction.

2. Prove that every bipartite graph with an odd number of vertices does NOT contain a Hamiltonian cycle

Since G is a bipartite graph, then bu Q1 above, every cycle must be of even length. This implies that the number of nodes in each of those cycle must be even. Therefore since the graph has an odd number of nodes, there cannot exist a cycle in G that goes through all the nodes. Therefore no Hamiltonian cycle exists.

3. The knight tour problem on an n by n chess board asks if the knight can start at some square and then move to every square on the board, and return to its starting square in the end without landing on the same square twice. Below is a drawing of how the knight moves in chess. Prove that if n is odd then there does NOT exist any knight tour.



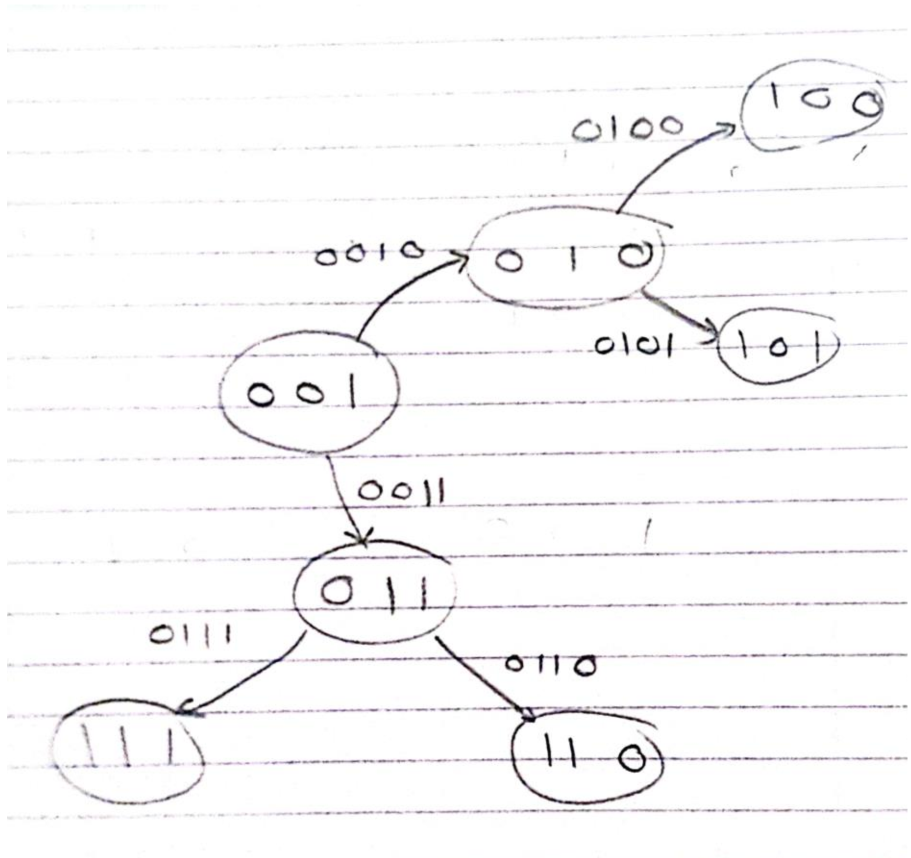
We look at the chess board as a graph with each square on the board being a vertex and every move the knight can make from one square to another as an edge between those two correlating vertices of the graph. There are nxn squares and since n is odd, then nxn is odd (odd times an odd is an odd). Therefore there is an odd number of vertices in the graph. We can partition the graph into 2 groups, white and green squares (where each square is a vertex). By the nature of a knight in chess, it can only move from a green square to a white square and vice versa. Therefore there are only edges going from one partition to the other =, and therefore the graph is a bipartite. Therefore, by Q2, this graph does not contain a Hamiltonian cycle. Therefore there does not exist a cycle that goes through every vertex. Therefore there does not exist a knight tour

Problem 4: (20 pts)

1. Given is the following binary sequence 01000100, which is a part of De-Bruijn sequence over the alphabet {0,1} and n=5.
   Is it possible to know what the next character is? Justify your answer.

Yes it is possible to know. The next character is a 1. The reason being is that if the next character was a 0, then there would be a repetition of the sequence 01000 (010001000) which is a cannot happen in a De-Bruijn sequence and would be a contradiction.

2. Draw a sub-graph of the De-Bruijn graph corresponding to Σ={0,1} and n=4. The subgraph to be drawn should include the vertex '001' and all the edges and vertices reachable by a path of length at most 2 from '001'



Problem 5 (20 pts)

Prove (in the next page) the following lemma:
Lemma: Let $G = (V, E)$ be a DAG (directed acyclic graph).
G has a **unique** topological sort if and only if it has a Hamiltonian path.
(note that you need to prove both directions in order to prove correctness)

→ If G has a unique topological sort it has a Hamiltonian path:

Let S= $v_0 v_1 \cdots v_k$ be the unique sort of G. Note that for every i ∈ {0,1,2,…,k-1}, we have that there is an edge going from v_i to v_{i+1}, as if it didn't these two vertices could be switched in the sort and make a new valid topological sort. Therefore a valid Hamiltonian path exists, by iterating through each vertex and using the edges that connects it to the next vertex in the sort (i.e go from v_0 to v_1, from v_1 to v_2, from v_2 to v_3 etc until reaching v_k). This path would form a valid Hamiltonian path.

← If G has a Hamiltonian path it has a unique topological sort.

Let C= $v_0 v_1 \cdots v_k$ be the Hamiltonian path of G. Note that for every i ∈ {0,1,2,…,k-1}, we have that there is an edge going from v_i to v_{i+1} (by definition of Hamiltonian path), therefore in a valid topological sort v_i would have to be before v_{i+1} for every i. This results in only one possible valid topological sort, which is C.