**CS375 Assignment 4 (Fall 2023)**

**Due on October 29 (by 11:59pm)**

Topics: graph, topological sorting, and linear programming.

**[Part A] Theory [74%]:**

1. (6%) The ***square*** of a directed graph G(V, E) is the graph G2 = (V, E2) such that (u, w) ∈ E2 if and only if for some v ∈ V, both (u, v) ∈ E and (v, w) ∈ E. That is, G2 contains an edge between u and w whenever G contains a path with exactly two edges between u and w. Describe an efficient algorithm for computing G2 from G for the adjacency-matrix representation of G. Give the running time of your algorithm.

Hint:

Adjacency matrix in In G2:

A2[i, j] = 1 if (A[i, 1] =1 && A[1, j] = 1) or (A[i, 2] =1 && A[2, j] = 1) or… or (A[i, n] =1 && A[n, j]=1)

Algorithm is defined in function CreateGsquare(A, n)

Input: Adjacency matrix A for graph G (n = |V|)

Output: Adjacency matrix Asquare for graph G2

**Answer:**

CreateGsquare(V) {

for i = 1 to V

     for j = 1 to V

           G^2[i][j] = 0

               for k = 1 to V

                       do if G[i][k] == 1 and  G[k][j] == 1

                               then G^2[i][j] == 1

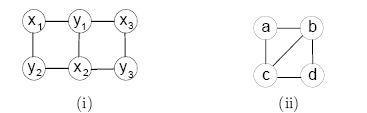
                               break

return G^2

}

The running time is V^3.

1. (a) [4%] Design an algorithm which can automatically indicate whether or not a graph *G* contain a cycle. Using the following two graphs as two examples to indicate what data structure to represent (i) and (ii), respectively.



**Answer:**

visit(node n) {  
      if n.color == grey: //if we're still visiting this node or its descendants  
        throw exception("Cycle found")

      n.color = grey //to indicate this node is being visited  
      for node child in n.children():  
        if child.color == white: //if the child is unexplored  
          visit(child)

      n.color = black //to show we're done visiting this node  
      return

}

Both (i) and (ii) can be represented by doubly linked lists, keeping each connected node to each of the node as head.

(b) [6%] Programming: implement your algorithm by using an appropriate data structure, and print out a cycle if the graph a cyclic. Print out the running time.

(c) [4%] Give the order that the nodes will be visited when doing a BFS starting at the node 1 of the following graph:

1

2

3

4

5

6

7

**Answer:**

1) visitOrder = []  
Stack = [1]  
  
2) visitOrder = [1]

Stack = [2, 4, 5]

3) visitOrder = [1, 2]  
Stack = [4, 5, 3, 7]  
  
4) visitOrder = [1, 2, 4]  
  
Stack = [5, 3, 7, 6]  
  
5) visitOrder = [1, 2, 4, 5]  
Stack = [3, 7, 6]  
  
6) visitOrder = [1, 2, 4, 5, 3]  
Stack = [7, 6]  
  
7) visitOrder = [1, 2, 4, 5, 3, 7]  
Stack = [6]  
  
8) visitOrder = [1, 2, 4, 5, 3, 7, 6]

1. [5%] Does an undirected graph with 5 vertices, each of degree 3 exist? If yes draw the graph, otherwise explain why there is no such graph.

**Answer:**

No, because the sum of the degrees of all the vertices in any undirected graph must be even; however, a graph with 5 vertices, each of degree 3, would have a sum of 15.

1. [5%]Apply depth first search to the following graph. Show the graph after a new edge has been identified as a tree, forward, or cross edge, back edge (indicate one of the four types). (NOTE: assume starting from node 1, and go to node 2 first).



A diagram of a new tree

Description automatically generated with medium confidence**Answer:**

1. [4%] Apply topological sort to the graph below. Show the sequence of the nodes found by your application. Include also the discovery and finishing time of each node. (Assume the starting node is 1, the second node to go is 7. Also node 2 will be selected before node 4.)

**Answer:** The topological order is 1, 6, 5, 2, 3, 8, 7, 3.

A diagram of a diagram

Description automatically generated with medium confidence

[8%] Programming: Implement the algorithm and print out the topological sorted nodes, and print out the edge type for each edge (e.g., T – tree edge; F – forward edge; B – backward edge; C – cross edge). What’s the time complexity. Print out the running time.



1. [8%] A bipartite graph G(V, E) is an undirected graph whose vertices can be partitioned into two disjoint sets V1 and V2 = V-V1 with the properties that no two vertices inV1 are adjacent in G and no two vertices in V2 are adjacent in G. All edges go between the two sets V1 and V2. Is the following graph G a bipartite graph? Write your algorithm to determine whether the graph G is bipartite and the two disjoint sets V1 and V2 if it is a bipartite. What’s the time complexity of your algorithm.

(Optional [extra 8%]: Programming: implement your algorithm, print out set V1 and set V2, and print out the running time).

1

2

3

4

5

6

7

8

9

10

11

**Answer:** 2-colorable graphs are also known as bipartite graphs, as we can separate the vertices into two sets such that all edges only connect vertices from different sets. A 2-colorable graph has vertices colored in two colors in such a way that no two adjacent vertices have the same color.

Graph G is a bipartite graph.

A diagram of a network

Description automatically generated

Based on this, we can see that the graph is 2-colorable, so it is a bipartite graph. Let R vertices be in V1, and B vertices be in V2. Then, V1 = {1, 4, 5, 6, 7, 11} and V2 = {2, 3, 8, 9, 10}. The time complexity is O(V^2 + E), where V = # of vertices and E = # of edges.

7. [8%]

One can model a maze by having a vertex for a starting point, a finishing point, dead ends, and all the points in the maze where more than one path can be taken, and then connecting the vertices according to the paths in the maze.

1. Construct such a graph for the following maze.

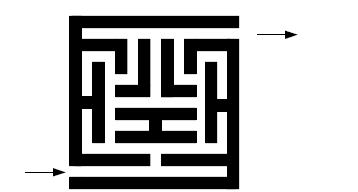
**Answer:**

A diagram of a network

Description automatically generated

1. Which traversal–DFS or BFS–would you use if you found yourself in a maze and why?

**Answer:** DFS because there are more possibilities of finding a path by exploring each node rather than visiting each adjacent node once and then returning to the previous node. When DFS traverses the next node, it is connected to the current node by an edge, whereas BFS does not.



8. [8%] A merchant plans to manufacture two models of home computers at costs of $250 and $400, respectively. To sell the computers, the $250 model yields a profit of $45 and the $400 model yields a profit of $50. The merchant estimates that the total monthly demand will not exceed 250 units. Using Graphical Method, find the number of units of each model that should be stocked in order to maximize profit. Assume that the merchant does not want to invest more than $70,000 in computer inventory.

**Answer:**

Cost: 250x + 400y <= 70, 250x + 400y = 70,000

Y <= (70,000 – 250x) / 400

Demand: x + y <= 250

Y <= 250 – x

Non-negative: x >= 0, y >= 0

Z = 45x + 50y

Graphs meet at (200,50), so solution is $11,500.

9. [8%] Using the Simplex Algorithm to solve the three variables linear programming problem

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Maximize** | P | = | 20x1 | + | 10x2 | + | 15x3 |  |  |
| **Subject to:** |  |  | 3x1 | + | 2x2 | + | 5x3 | ≤ | 55 |
|  |  |  | 2x1 | + | x2 | + | x3 | ≤ | 26 |
|  |  |  | x1 | + | x2 | + | 3x3 | ≤ | 30 |
|  |  |  | 5x1 | + | 2x2 | + | 4x3 | ≤ | 57 |
|  |  |  | x1 | , | x2 | , | x3 | ≥ | 0 |

Show the results through the pivot operations and linear program (show the table step by step).

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | X1 | X2 | X3 | S1 | S2 | S3 | S4 |  |
| P | -20 | -10 | -15 | 0 | 0 | 0 | 0 | 0 |
| S1 | 3 | 2 | 5 | 1 | 0 | 0 | 0 | 55 |
| S2 | 1 | 1 | 3 | 0 | 0 | 1 | 0 | 30 |
| S3 | 1 | 1 | 3 | 0 | 0 | 1 | 0 | 30 |
| S4 | 5 | 2 | 4 | 0 | 0 | 0 | 1 | 57 |

Min(55/3, 26/2, 30/1, 57/5) = 57/7

4th row is pivot row.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | X1 | X2 | X3 | S1 | S2 | S3 | S4 |  |
| P | 0 | 0 | 5 | 0 | -1 | 0 | 0 | 260 |
| X2 | 3 | 1 | 1 | 0 | -1 | 0 | 0 | 26 |
| S3 | -1 | 0 | 2 | 0 | -1 | 1 | 0 | 4 |
| S4 | 3 | 0 | 3 | 0 | -2 | 0 | 1 | 51 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | X1 | X2 | X3 | S1 | S2 | S3 | S4 |  |
| P | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 310 |
| X2 | 1 | 1 | 0 | 0 | 2 | 1 | 0 | 30 |
| X1 | 0 | -1 | 1 | 1 | -1 | -1 | 0 | 2 |
| S4 | 0 | -3 | 0 | 0 | -3 | 3 | 1 | 45 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | X1 | X2 | X3 | S1 | S2 | S3 | S4 |  |
| P | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 370 |
| X2 | 1 | 1 | 0 | 0 | 2 | 1 | 0 | 30 |
| X1 | 0 | -1 | 1 | 1 | -1 | -1 | 0 | 2 |
| X3 | 0 | -5 | 0 | 0 | -5 | -5 | 1 | 45 |

P = 0 when x1 = 2, x2 = 30, x3 = 0. Optimal value = 370.

**[Part B]: Programming Part (26%)**

**B.1 Graph Algorithm (16%)**

Suppose a CS curriculum consists of n courses, all of them mandatory. The prerequisite graph G has a node for each course, and an edge from course v to course w if and only if v is a prerequisite for w. Find and algorithm that works directly with this graph representation, and computes the minimum number of semesters necessary to complete the curriculum (Assume that a student can take any number of courses in one semester).

Using following example to justify your answer:

The CS Department requires fifteen one-semester courses with the prerequisites shown below:

cs1

cs2

cs3

cs4 requires cs2

cs5 requires cs4

cs6 requires cs1 and cs3

cs7 requires cs4

cs8 requires cs5 and cs6

cs9 requires cs7

cs10 requires cs9

cs11 requires cs8

cs12 requires cs3

cs13 requires cs6

cs14 requires cs4 and cs6

cs15 requires cs14

*Your task is to determine the minimum number of semesters needed to finish the degree.*

(Hint: Represent the courses and their prerequisites as a DAG. Finding the minimum number of semesters translates to a simple graph problem, e.g., Using adjacency-matrix or linked-list representation in BFS).

**Please provide:**

1. Manually plot the DAG (3%)
2. Explain the algorithm that you are going to implement by the Pseudo-code, and indicate the minimum number of semesters necessary to finish the degree. Indicate the time complexity and space complexity (3%)

**Answer:** I used topological sort, which has a time complexity of O(V+E) and a space complexity of O(V^2).

1. Write and run your program to print out the result for verification (i.e., minimum number of semesters, and print out the running time) (10%)

**B.2. Linear Programming (10%)**

**Design and implement a linear programming algorithm to solve the following minimum cost problem**

The liquid portion of a diet is to provide at least 300 calories, 36 units of vitamin A, and 90 units of vitamin C daily. A cup of dietary drink X provides 60 calories, 12 units of vitamin A, and 10 units of vitamin C. A cup of dietary drink Y provides 60 calories, 6 units of vitamin A, and 30 units of vitamin C. Now, suppose that dietary drink X costs $0.12 per cup and drink Y costs $0.15 per cup. How many cups of each drink should be consumed each day to minimize the cost and still meet the stated daily requirements?

Print out the minimum cost, and the number cups of drink X and number of cups of drink Y; Print out the running time.

**B.3. [Optional Extra point 10%]**

**To answer the Question #9 (Part A), implement the Simplex Algorithm and display the results by your program, and print out the running time.**