6.034 Quiz 1

30 September 2020

Name	
Email	

Problem number	Maximum	Score	Grader
1 - Search	33		
2 - Games	33		
3 - Constraints	34		
Total	100		

There are 20 pages in this quiz, including this one, but not including tear-off sheets. Tear-off sheets with duplicate drawings and data are located after the final page of the quiz.

The quiz is open book, open notes, open just about everything, including a calculator, but no internet or other people.

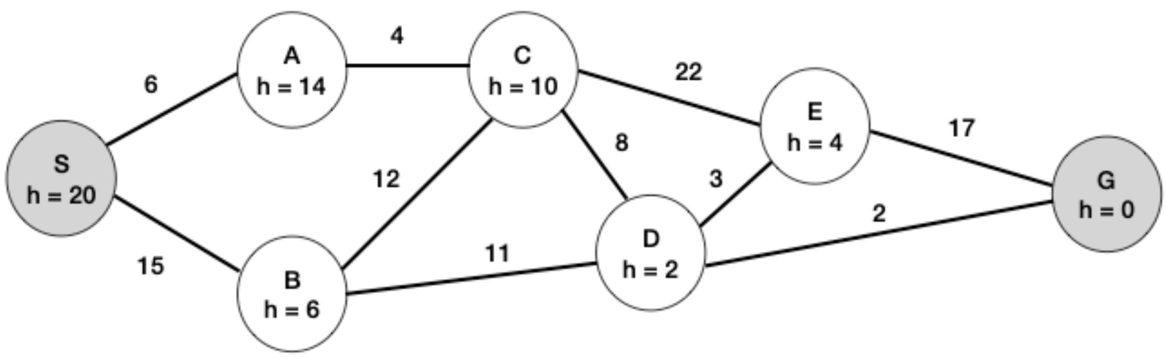
Problem 1: Search (33 points)

Part A: Your TikTok Obsessions (17 points)

You have decided to download TikTok during your quarantine boredom. You find, however, that TikTok has removed the search by username functionality, so now you can't directly find the user who created the annoying dance you've got stuck in your head. The only way to find a particular user is to search through TikTok profiles of accounts a user follows until you find the goal user.

For your convenience, a copy of the graph is provided on a tear-off sheet in the separate tear-off pdf file.

You create the following graph, with nodes representing users and edges representing the "follows" relationship. In the graph below, you start your search at user node S. You want to find a path from S to your goal user (G). Each edge is labeled with its length, representing the amount of time you spend getting distracted by videos before going to the next user. The start node (S) and goal node (G) are shown below in gray.



A1 (1 point) What is the shortest path, in terms of edge length, between S and G? Break ties lexicographically. You may do this by inspection.

A2 (2 points) You decide to use a **depth-first search**. In the space below, **draw the search tree** used to determine the path from S to G, and **state the path** by writing the nodes in order in the box provided. Break ties lexicographically.

Path:

A3 (2 points) You now decide to use **breadth-first search**. In the space below, **draw the search tree** used to determine the path from S to G, and **state the path** by writing the nodes in order in the box provided. Break ties lexicographically.

Path:

A4 (1 point) How is the queue updated differently when using breadth-first search instead of depth-first search? Answer this question for **graphs in general**, not for this specific question.

A5 (6 points) Due to your shortened attention span caused by excessive TikTok usage, you become frustrated that finding the goal user (G) is taking so long. You decide to try **branch and bound** (with extended set and no heuristic) to find the path from S to G that will take the shortest amount of time, i.e., shortest distance based on edge length.

In the space below, **draw the search tree** and **write out the queue** at each step in the search.

Be sure to:

a. Draw children and break ties lexicographically (e.g., A < B < C).
b. Clearly indicate the order in which nodes are extended by numbering the extended nodes in your search tree (①, ②, ③, …).
c. After each node extension, write the contents of the queue, numbering steps as in the tree.

We've started the tree and the queue for you.

Queue:

1. [S]

Tree:

① S

A6 (2 point) List the nodes in the **extended set**, in the order extended.

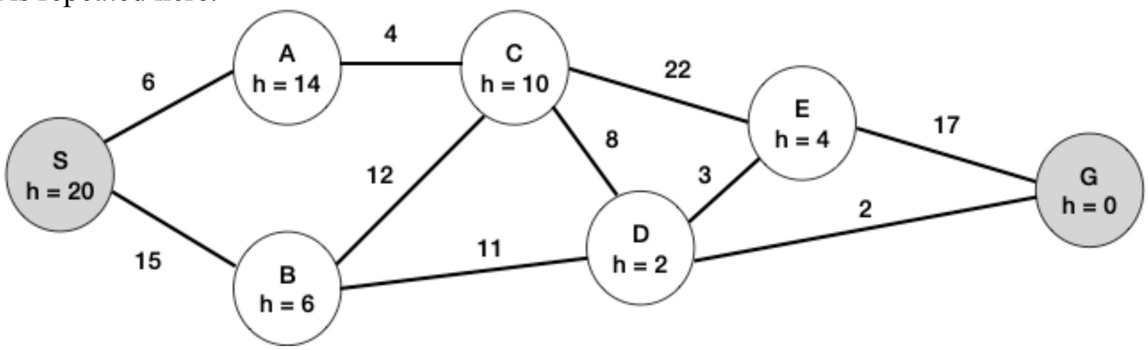
A7 (1 point) What is the **path** found from S to G? Write the nodes in order.

A8 (1 point) Is branch and bound considered an "informed" search? Explain why or why not.

A9 (1 point) Explain in your own words what it means to prune a search tree. Give one example of an algorithm covered in lecture that uses pruning.

Part B: Becoming A* on TikTok (16 points)

You decide there might be a better search to find your goal user on TikTok. The graph from part A is repeated here.



B1 (10 points)

In the space below, draw the search tree using **A* search** (with a heuristic and extended set) to find the path from S to G.

Be sure to:

a. Draw the children of each node in lexicographic order (e.g., A < B < C).
b. Break ties between paths lexicographically (e.g., S-D-Y would come before S-E-A).
c. Clearly indicate the order in which you extended nodes by numbering the extended nodes in your search tree ①, ②, ③, ...).
d. After each node extension, write the contents of the queue, numbering steps as in the tree.

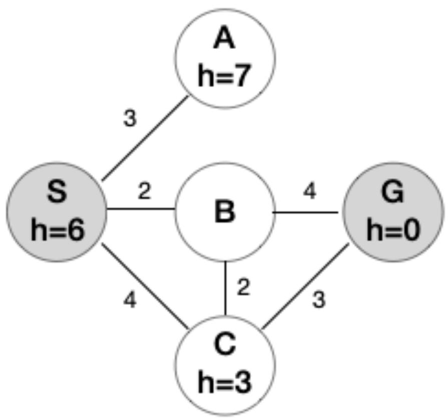
We've started the tree and the queue for you.

Queue:	Tree:
1. [S]	① S

B2 (2 point) List the nodes in the **extended set**, in the order extended.

B3 (1 points) What is the **path** found from S to G? Write the nodes in order.

B4 (3 points) Heuristic admissibility and consistency



For the graph above, find an integer value for the heuristic of node B, H(B), that would be both admissible and consistent. For full credit, show your work that proves it both admissible and consistent.

Problem 2: A Gaming Product Launch (33 points)

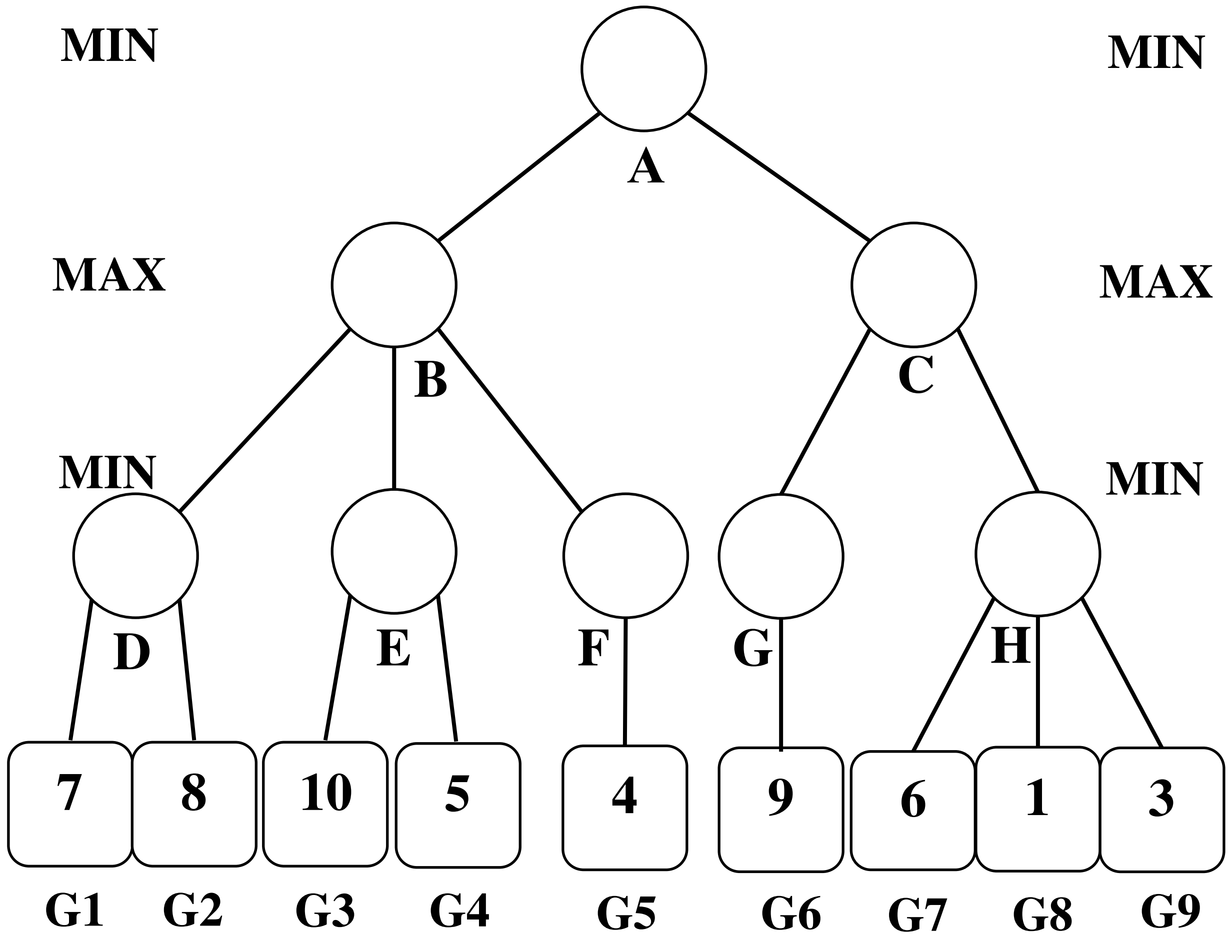
Part A: Maximizing Profits (12 points)

You wake up from a strange nightmare and realize that you are, in fact, NVIDIA CEO Jensen Huang's leather jacket. As his trusted leather jacket, you sit in on his board meetings, where you help plan the launch of their newest products: the fabled next-generation graphics processing units (GPUs).

You hear the rumors that rival AMD is planning a new product that will compete directly with NVIDIA products. In order to plan the launch of NVIDIA's new products, you use the skills you acquired in 6.034 to turn the product launch into a Minimax search tree as depicted on the following page.

Each **decision node** in the tree is displayed as a labeled circle. Each **leaf node**, labeled with the NVIDIA products to be launched (G0 through G9), is displayed as a square. The static evaluation at each leaf node represents the net revenue (in billions of dollars) that particular product would earn for NVIDIA in the following fiscal year.

For your convenience, a copy of the tree is provided on a tear-off sheet in the separate tear-off pdf file.



Your job as Jensen's leather jacket is to **maximize** the revenue NVIDIA makes from the launch of the new products. AMD seeks to **minimize** the revenue to keep it for themselves. Unfortunately, AMD made the first move.

A1 (8 points) Perform Minimax search (without alpha-beta pruning) on **the tree above**, using the static evaluation values given by the tree's leaf nodes (G0 – G9). **Write each decision node's value inside the node on the tree above**.

A2 (1 point) What is the Minimax value at node A?

A3 (1 point) Which leaf node does the Minimax algorithm choose? (Circle your answer.)

G1 G2 G3 G4 G5 G6 G7 G8 G9

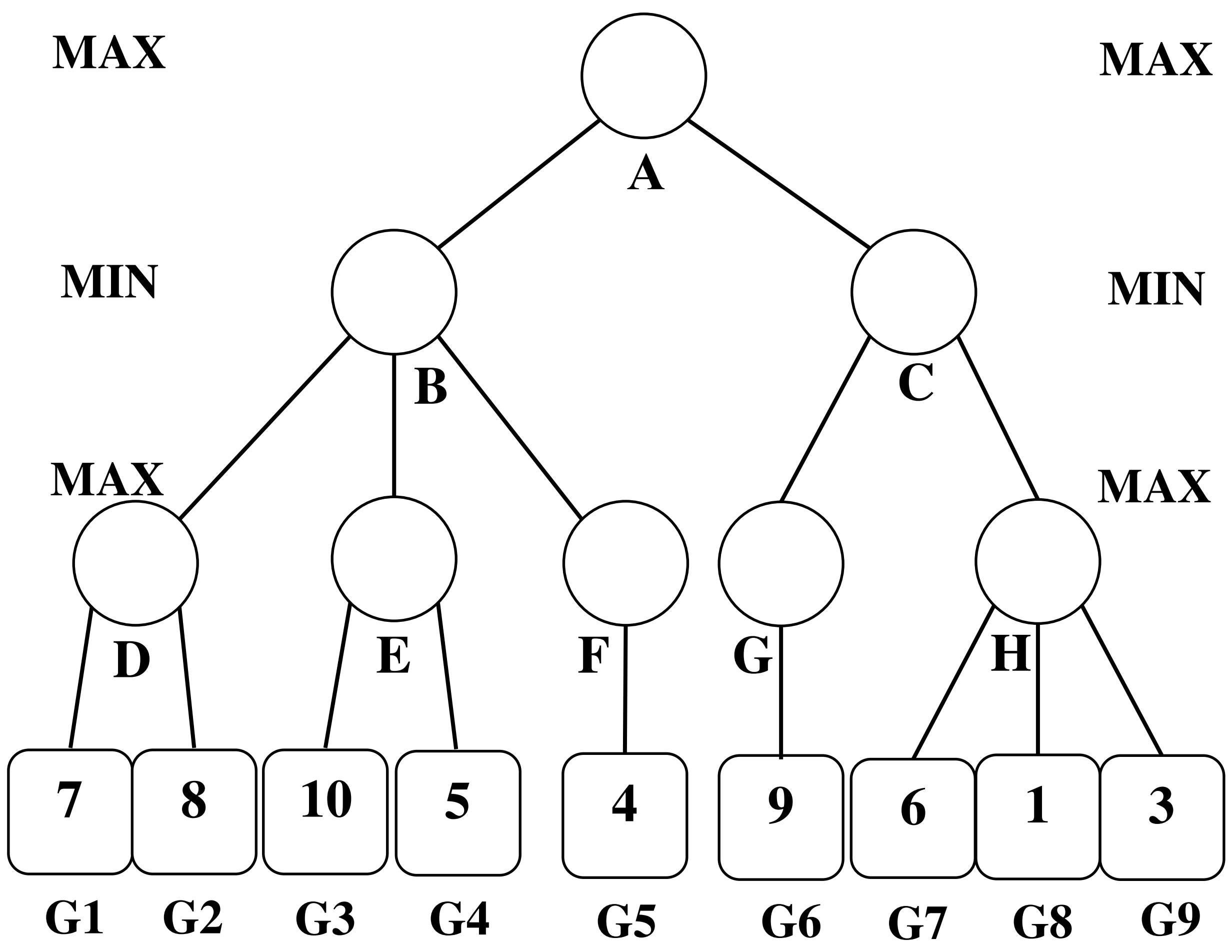
A4 (2 points) What is the Minimax path? Write the node labels in order from top to bottom.

Part B: The Leaks Keep Coming (13 points)

The oncoming product launch has prompted AMD leakers to release information suggesting their products will beat NVIDIA's in performance, making Jensen Huang sweat nervously. Being the faithful leather jacket that you are, you decide to take advantage of the leaks to **make the first move in the search tree from part A and to use alpha-beta pruning**.

B1 (11 points) Perform Minimax search with alpha-beta pruning on the tree below. Notice that the order of the players, MAX and MIN, is different from the search performed in Part A.

a. Indicate on the tree below which leaf nodes are pruned, i.e., **not** evaluated, by placing an "X" beneath the nodes.
b. Indicate which branches are pruned by drawing a zigzag line across the branches.



B2 (1 point) Which leaf nodes are pruned, i.e., not evaluated? (Circle **all** that apply. If no leaf nodes are pruned, circle NONE.)

G1 G2 G3 G4 G5 G6 G7 G8 G9 NONE

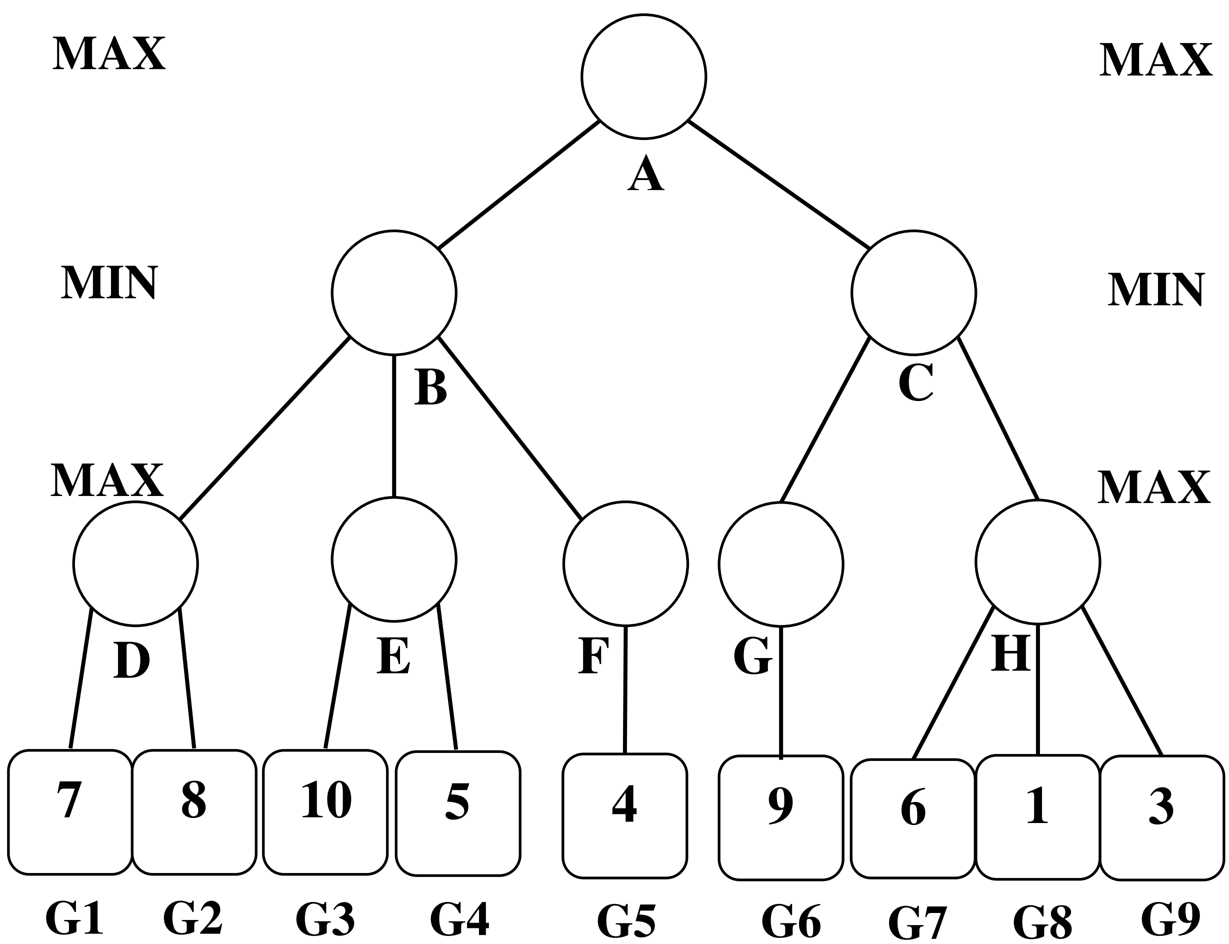
B3 (1 point) Which decision nodes are pruned, i.e., not assigned values? (Circle **all** that apply. If no decision nodes are pruned, circle NONE.)

A B C D E F G H NONE

Part C: Faster than a Teraflop (8 points)

You are putting the final touches on your presentation, and you decide to run an alpha-beta pruning benchmark to show NVIDIA products beating AMD's. Using the AMD product, you will run your search routine on a version of the tree **modified such that alpha-beta pruning does no pruning, i.e., all leaf nodes are statically evaluated.** Using the NVIDIA product, you will run your search routine on a version of the tree **modified such that alpha-beta pruning reduces the number of static evaluations as much as possible.**

Here's the original tree from Part B, which you'll modify to affect the number of static evaluations.



C1 (4 points) Modify the tree used in part B, pictured on the previous page and on a tear-off sheet, by **reordering the nodes in such a way that no leaf nodes would be pruned.** Keep in mind that **each node must remain connected to its original parent and children** in order for the new tree to be a valid reordering. Express your answer by redrawing the modified tree in the box below. If no nodes were pruned in the original tree from part B, write N/A in the space provided.

.

C2 (4 points) Modify the tree used in part B by **reordering the nodes in order to maximize the number of pruned nodes.** Keep in mind that **each node must remain connected to its original parent and children** in order for the new tree to be a valid reordering. Express your answer by redrawing the modified tree in the box below. If the number of pruned nodes was already optimal, write N/A in the space provided.

Problem 3: Constraint Satisfaction (34 points)

Part A: Strut the Pup (18 points)

Kyla is an elite software developer for Strut the Pup, a "walk-share" service for good girl and boy dogs and their owners to request walks wherever, whenever. To attract investors, UberDog is augmenting their pairing algorithm with artificial intelligence. Kyla is tasked with assigning walking routes to each dog and decides to implement a simple constraint satisfaction system.

For your convenience, a copy of the constraints and constraint graph are provided on a tear-off sheet in the separate tear-off pdf file.

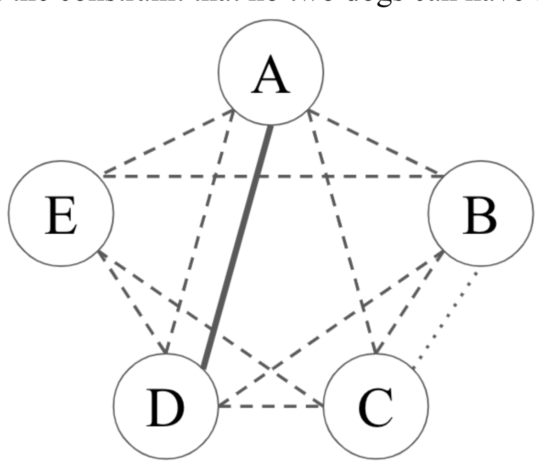
Six dogs have requested walks and should be assigned routes in order:
Ada **(A)**, Balto **(B)**, Charley **(C)**, Doug **(D)**, Ellie **(E)**

Pairing constraints:

- Each of five routes **(1, 2, 3, 4, 5)** must be assigned exactly one dog.
- Ada **(A)** must go on a slow route **(4, 5)**. She wants time to stop to sniff the roses and eat the leaves during her walk.
- Balto **(B)** must go on a fast route **(1, 3)**. Balto is a Big Business Dog doing Big Dog Business for Strut the Pup and has dog places to be.
- Charley **(C)** and Balto's **(B)** route numbers **must be adjacent**. Charley works for a competitor, Paws to Pavement and is trying to eavesdrop for trade secrets during Balto's walk.
- Doug **(D)** and Ada's **(A)** route numbers **must not be adjacent**. Doug is scared of Ada.
- Ellie **(E)** needs to go on routes that pass by the house of a certain cute dog **(2, 4)**.

The constraint graph is below.

- The dotted line from **B** to **C** represents the constraint for those dogs whose route numbers **must** be adjacent to each other.
- The solid line from **A** to **D** represents the constraint for those dogs whose route numbers **must not** be adjacent to each other.
- Dashed lines represent the constraint that no two dogs can have the same route.



A1 (2 points) Fill in the table below with the initial domain values, i.e., route numbers, for each of the variables A through E.

	Domain
A	
B	
C	
D	
E	

A2 (14 points) Perform **<u>Depth-First Search with Forward Checking and no propagation</u>** (DFS+FC) to generate a matching of dogs to routes. **Draw your search tree in box below, using the order of variables provided.** *(If you want to start over, use the next page. If you write on both pages, <u>clearly</u> mark which page we are to grade.)*

A
B
C
D
E

A3 (2 point) The original ordering of assignments was A-B-C-D-E. List two alternate orderings of dog assignments that would **not** result in backtracking.

Ordering 1:

Ordering 2:

This is a duplicate for part A2 (DFS + FC). If you want this copy graded instead, check the box.

☐ **I want to start over; grade this copy.**

A
B
C
D
E

A3 (2 point) The original ordering of assignments was A-B-C-D-E. List two alternate orderings of dog assignments that would **not** result in backtracking.

Ordering 1:
Ordering 2:

Part B: Who's holding the leash? (16 points)

Kyla forgot that (most) dogs can't walk themselves! She revises the system to match both humans and dogs to a route. This time she's rolling out the system in a new neighborhood with different dogs and routes. She also includes herself because she wants some relaxation after a long day at work.

For your convenience, a copy of the constraints and constraint graph are provided on a tear-off sheet in the separate tear-off pdf file.

Humans: Andrew **(A)**, Bea **(B)**, Charlotte **(C)**
Dogs: Jack **(J)**, Kyla **(K)**, Lucky **(L)**
Routes: **1, 2, 3**
Make assignments in the order: **A, B, C, J, K, L**

Pairing constraints:

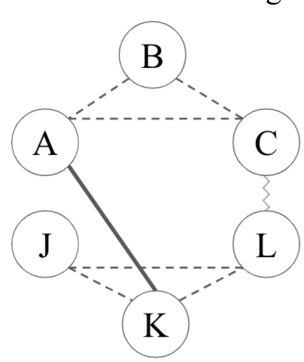
- Each route **(1, 2, 3)** must be assigned exactly one human **(A, B, C)** and one dog **(J, K, L)**.
- Andrew **(A) must not** be paired with Kyla **(K)**. Andrew visits Kyla too much already.
- Lucky **(L)** needs to go on route **3**. Her favorite park is along that route.
- Charlotte **(C) must** be paired with Lucky **(L)**. Lucky personally requested Charlotte.
- Kyla **(K)** needs to go on route **1**. She needs to stop by campus for her therapy dog duties.

Initial Domains:

A	1, 2, 3
B	1, 2, 3
C	1, 2, 3
J	1, 2, 3
K	1
L	3

The constraint graph is below.

- The zigzag line from **C** to **L** represents the constraint that they **must** have the same route.
- The solid line from **A** to **K** represents the constraint that they **must not** have the same route.
- Dashed lines represent the constraint that no two dogs and no two humans can have the same route.



B1 (16 points) Perform **Domain Reduction propagating on reduced neighbors only** to reduce the potential schedule choices by filling out the table below. There may be more rows than you need. **<u>If a user is already in the queue, do not add them into the queue again.</u>**

Initialize your propagation queue in the order **A, B, C, J, K, L.**

Pairing constraints and initial domains are listed again for your convenience.

Pairing constraints:

- Each route **(1, 2, 3)** must be assigned exactly one human **(A, B, C)** and one dog **(J, K, L)**.
- Andrew **(A) must not** be paired with Kyla **(K)**. Andrew visits Kyla too much already.
- Lucky **(L)** needs to go on route **3**. Her favorite park is along that route.
- Charlotte **(C) must** be paired with Lucky **(L)**. Lucky personally requested Charlotte.
- Kyla **(K)** needs to go on route **1**. She needs to stop by campus for her therapy dog duties.

Initial Domains:

A	1, 2, 3
B	1, 2, 3
C	1, 2, 3
J	1, 2, 3
K	1
L	3

Note: You don't need to do assignments; just reduce the domains.

	Variable de-queued	List all values just eliminated from neighboring variables or NONE
1	**A**	
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		

For partial credit, show your work here.