The Ohio State University
Computer Science and Engineering

CSE 3521– Midterm
Survey of Artificial Intelligence I: Basic Techniques

Instructor: Wuwei Lan

2019/03/06

Student Name: ______________________________

LastName.Number: ______________________________

This exam contains 6 pages (including this cover page) and 4 questions. Total of points is 100. Good luck and Happy reading work!

Distribution of Marks

Question	Points	Score
1	60	
2	10	
3	10	
4	20	
Total:	100	

1. This part will test your basic knowledge about AI techniques, just give short and clear answer for each question.

   (a) (10 points) Please give your own definition for Artificial Intelligence (AI).

   (b) (10 points) We have BFS and DFS for uninformed search, what are the pros and cons for each strategy?

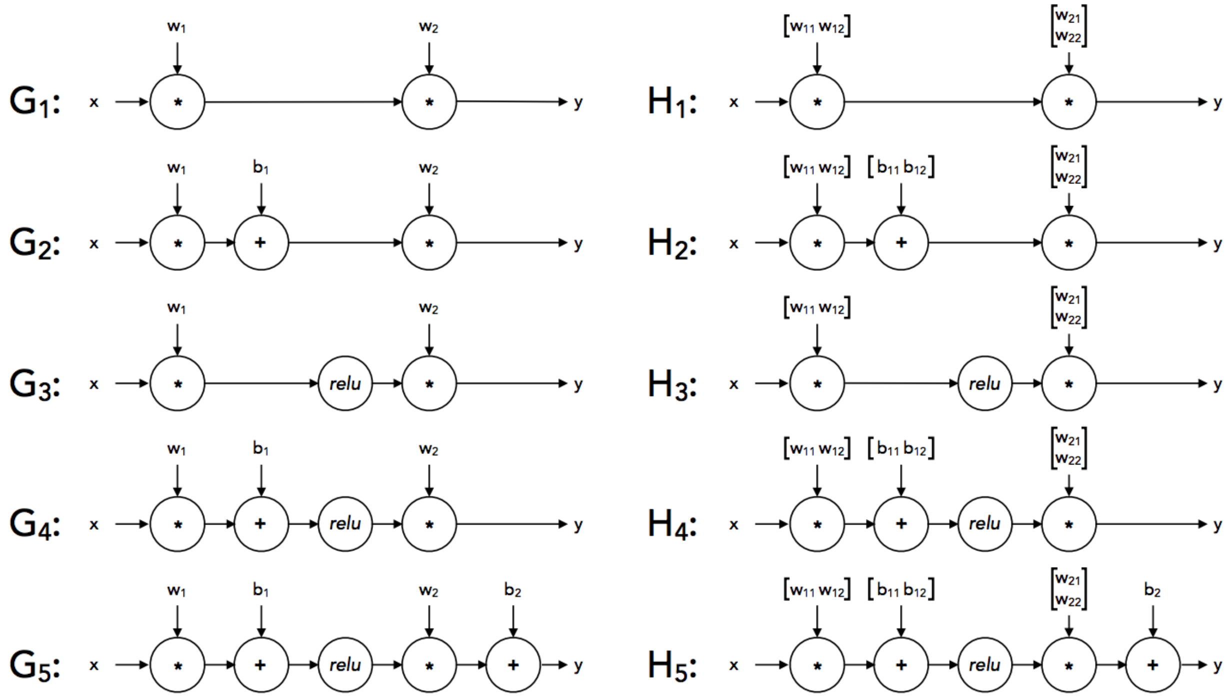
   (c) (10 points) What is the condition for heuristic function to make sure the optimality of A^*? How do we know our designed heuristic functions satisfy this criteria?

   (d) (10 points) Explain the difference between Underfitting and Overfitting. What can we do to deal with these two cases?

   (e) (10 points) We have non-linear activation functions for multi-layer neural networks, what will happen if we use linear functions? For example, identity function $g(x) = x$.

   (f) (10 points) Why do we introduce precision and recall for evaluation? What is the problem with accuracy?

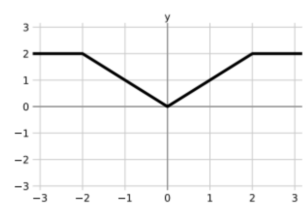
2. Neural Networks: Representation



For each of the piecewise-linear functions below, mark all networks from the list above that can represent the function **exactly** on the range $x \in (-\infty, \infty)$. In the networks above, $relu$ denotes the element-wise ReLU nonlinearity: $relu(z) = max(0, z)$. The networks G_i use 1-dimensional layers, while the networks H_i have some 2-dimensional intermediate layers.

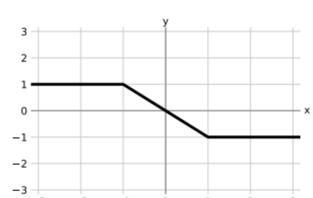
(a) (5 points) Mark your answers below:

☐ G_1 ☐ H_1
☐ G_2 ☐ H_2
☐ G_3 ☐ H_3
☐ G_4 ☐ H_4
☐ G_5 ☐ H_5
○ None of the above



☐ G_1 ☐ H_1
☐ G_2 ☐ H_2
☐ G_3 ☐ H_3
☐ G_4 ☐ H_4
☐ G_5 ☐ H_5
○ None of the above

(b) (5 points) Mark your answers below:

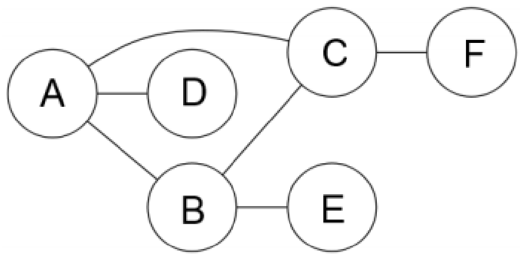


☐ G_1 ☐ H_1
☐ G_2 ☐ H_2
☐ G_3 ☐ H_3
☐ G_4 ☐ H_4
☐ G_5 ☐ H_5
○ None of the above

☐ G_1 ☐ H_1
☐ G_2 ☐ H_2
☐ G_3 ☐ H_3
☐ G_4 ☐ H_4
☐ G_5 ☐ H_5
○ None of the above

3. The graph below is a constraint graph for a CSP that has only binary constraints. Initially, no variables have been assigned.

   For each of the following scenarios, mark all variables for which the specified filtering might result in their domain being changed.



   (a) (2 points) A value is assigned to A. Which domains might be changed as a result of running forward checking for A?

   ○ A ○ B ○ C ○ D ○ E ○ F

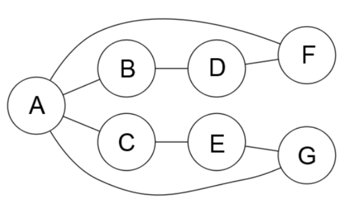
   (b) (2 points) A value is assigned to A, and then forward checking is run for A. Then a value is assigned to B. Which domains might be changed as a result of running forward checking for B?

   ○ A ○ B ○ C ○ D ○ E ○ F

   (c) (2 points) A value is assigned to A. Which domains might be changed as a result of enforcing arc consistency after this assignment?

   ○ A ○ B ○ C ○ D ○ E ○ F

   (d) (4 points) You decide to try a new approach to using arc consistency in which you initially enforce arc consistency, and then enforce arc consistency every time you have assigned an even number of variables. You have to backtrack if, after a value has been assigned to a variable, X, the recursion returns at X without a solution. Concretely, this means that for a single variable with d values remaining, it is possible to backtrack up to d times. For the following constraint graph, if each variable has a domain of size d, how many times would you have to backtrack in the worst case for each of the specified orderings?



   A-B-C-D-E-F-G: __________

   F-D-B-A-C-G-E: __________

   C-A-F-E-B-G-D: __________

4. Python Programming: write down python code for the following Naive Bayes classifier.

From training corpus, extract *Vocabulary*

Calculate $P(c_j)$ terms
- For each c_j in C do
  $docs_j \leftarrow$ all docs with class $= c_j$

$$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

- Calculate $P(w_k \mid c_j)$ terms
  - $Text_j \leftarrow$ single doc containing all $docs_j$
  - For each word w_k in *Vocabulary*
    $n_k \leftarrow$ # of occurrences of w_k in $Text_j$

$$P(w_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha |Vocabulary|}$$

$$P(c_j|doc) \propto P(c_j) \prod_k P(w_k|c_j)$$

(a) (10 points) Python code for training Naive Bayes classifier.

(b) (10 points) Python code for testing Naive Bayes classifier.

This page is intentionally left blank to accommodate work that wouldn't fit elsewhere and/or scratch work.