CS-171, Intro to A.I. — Final Exam — Summer Session 1, 2018

YOUR NAME: ____

YOUR ID: ___________ ID TO RIGHT: ___________ ROW: _________ SEAT: ________

Please turn off all cell phones now.

The exam will begin on the next page. Please, do not turn the page until told.

When told to begin, check first to ensure that your copy has all the pages, as numbered 1-18 in the bottom-right corner of each page. We will supply a new exam for any copy problems.

The exam is closed-notes, closed-book. No calculators, no cell phones, no electronics.

Clear your desk except for pen, pencil, eraser, & water bottle. Put backpacks under your seat. Please do not detach the provided scratch paper from the exam.

After you first stand up from your seat, your exam is over and must be turned in immediately. You may turn in your exam early and leave class when you are finished. IDs will be checked.

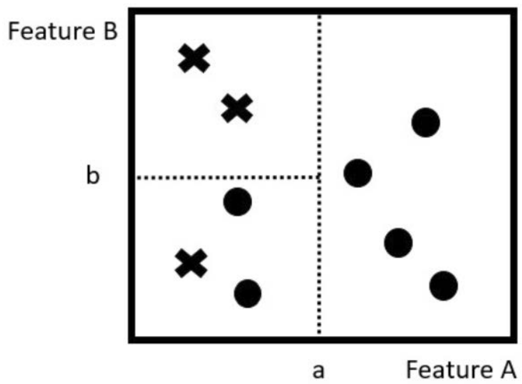
This page summarizes the points for each question, so you can plan your time.

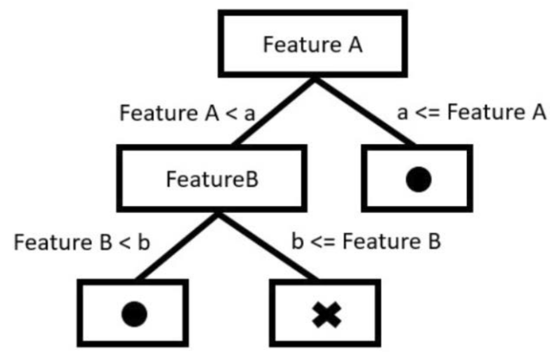
1. **(10 pts total) DECISION TREE LEARNING.**
2. **(10 pts total, 2 pts each) SEARCH STRATEGIES.**
3. **(10 pts total, -1 pt for each error, but not negative) MINI-MAX SEARCH/ALPHA-BETA PRUNING.**
4. **(5 pts total, 1 pt each) UNIFIERS AND UNIFICATION.**
5. **(5 points if correct, else partial credit up to 3 pts as 1 pt for each correct non-trivial derivation) PROPOSITIONAL LOGIC CONVERSION TO CNF.**
6. **(5 points if correct, else partial credit up to 3 pts as 1 pt for each correct non-trivial derivation) FIRST ORDER PREDICATE LOGIC CONVERSION TO CNF.**
7. **(10 pts total) PROBLEM SOLVING BY SEARCH/MINIATURE TOWER OF HANOI.**
8. **(10 pts total) BAYESIAN NETWORKS.**
9. **(7 pts total, 1 pt each) PROBABILITY FORMULAS.**
10. **(10 points total, 2 pts each) CONSTRAINT SATISFACTION PROBLEMS.**
11. **(3 pts if correct, else partial credit up to 2 pts as 1 pt for each correct non-trivial resolution) FOPL: PROVE THAT TWEETY IS A BIRD.**
12. **(10 pts total, 1 pt each) MACHINE LEARNING CONCEPTS.**
13. **(5 pts if correct, else partial credit up to 3 pts as 1 pt for each correct non-trivial resolution) RESOLUTION PROOF THAT CHARLIE DID IT.**

The Exam is printed on both sides to save trees! Work both sides of each page!

1. (10 pts total) DECISION TREE LEARNING.

1.a. (4 pts) Draw a decision tree from the decision boundaries shown at the left hand side. Label each branch node with the feature selected for the decision at that node. Label each arc with either “Feature X < x” or “x <= Feature X” as appropriate. Label each leaf node with either a circle or an X.



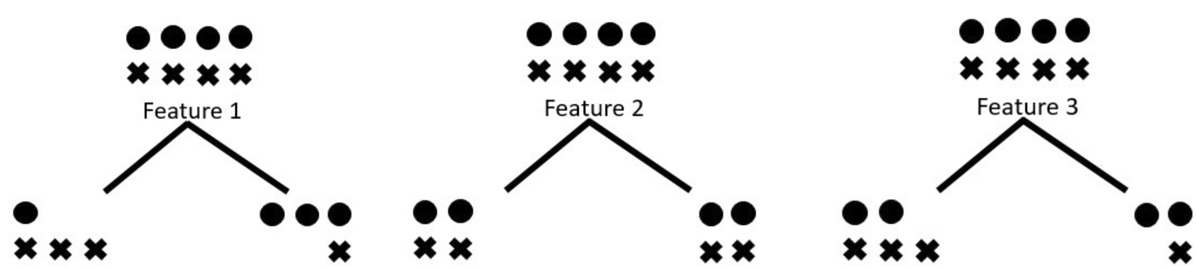


1.b. (3 pts) This problem is unrelated to 1.a. above. You are constructing a decision tree and have reached a branch node at which there are four circles and four Xs. You are considering which feature to select as the next ro … 2, and Featur … will be ask … heir desirability to be the next root of your decision sub-tree at this branch node. <u>(You should not need calculation to answer this question; the answer should be visually apparent based upon the material in your textbook and the lecture slides. However, for your convenience: $\log_2 1 = 0$; $\log_2 2 = 1$; $\log_2 3 \approx 1.7$; $\log_2 4 = 2$; $\log_2 5 \approx 2.3$.)</u>

You are not obliged to provide the numeric information gain, which is shown here for your benefit. You are only obliged to order the features by their desirability as the root.

Fill in the blanks with Feature1, Feature2, and Feature3 in the order of largest information gain, i.e., in the order of their desirability to be the next root of your decision sub-tree at this branch node.

Feature 1 **(0.275 bits)** > Feature 3 **(0.0625 bits)** > Feature 2 **(0 bits)**



1.c. (3 pts, 1 pt each) Decision Tree concepts. Fill in the blanks below with T (= True) or F (= False).

<u>F(alse)</u> Decision tree classifiers can avoid overfitting if a sufficiently large training data set is provided.

<u>F(alse)</u> The prediction error on the test data set of a decision tree classifier is always less than a perceptron classifier if both are provided with the same training data set.

<u>T(rue)</u> A decision tree can learn and represent any Boolean function.

****** TURN PAGE OVER AND CONTINUE ON THE OTHER SIDE ******

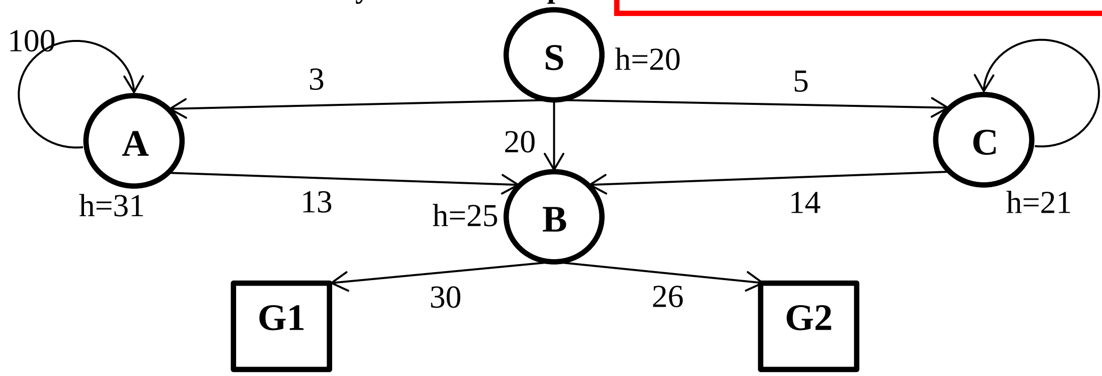
2. (10 pts total, 2 pts each) SEARCH STRATEGIES. Execute Tree Search through this graph (i.e., do not remember visited nodes, so repeated nodes are possible). It is not a tree, but pretend that you don’t know that it is not a tree. S is the start node and G1, G2 are the two goal nodes.

Step costs are given next to each arc. Heuristic values are given next to each node (as h=x). The successors of each node are indicated by the arrows out of that node. **Successors are returned in left-to-right order. Successors of S are A, B, C; successors of A are A, B; successors of B are G1, G2; successors of C are B, C; in those orders.**

For each search strategy below, show the order i children are generated) . If stuck in a loop, indicate the l write “None”. Your “Path found” will be scored based o

The first one is done for you as an example.

Please see the lecture slides for Uninformed Search, topic “When to do Goal-Test? When generated? When popped?” for clarification about exactly what to do in practical cases.



See Chapter 3.

2.a. (example) DEPTH FIRST SEARCH.

Order of node expansion: S A A A ...

Path found: None

DFS does the Goal-test iteratively on each child as generated, keeping the queue on the stack.

See Section 3.4.3 and Fig. 3.17.

2.b. (2 pts total) BREADTH FIRST SEARCH.

(1 pt) Order of node expansion: S A B (G1)

(1 pt) Path found: S B G1

BFS does the Goal-test before the child is pushed onto the queue. The goal is found when B is expanded.

See Section 3.4.1 and Fig. 3.11.

2.c. (2 pts total) UNIFORM COST SEARCH.

(1 pt) Order of node expansion: S A C B (G2)

(1 pt) Path found: S A B G2

UCS does goaltest when node is popped off queue.

See Section 3.4.2 and Fig. 3.14.

2.d. (2 pts total) GREEDY (BEST-FIRST) SEARCH.

(1 pt) Order of node expansion: S C C C ... etc.

(1 pt) Path found: None

C always has lower h (=21) than any other node on queue.

See Section 3.5.1 and Fig. 3.23.

2.e. (2 pts total) ITERATED DEEPENING SEARCH

(1 pt) Order of node expansion: S S A B (G1)

(1 pt) Path found: S B G1

IDS does the Goal-test iteratively on each child as generated, keeping the queue on the stack.

See Sections 3.4.4-5 and Figs. 3.18-19.

2.f. (2 pts total) A* SEARCH.

(1 pt) Order of node expansion: S C A B (G2)

(1 pt) Path found: S A B G2

A* does goaltest when node is popped off queue.

See Section 3.5.2 and Figs. 3.24-25.

3. (10 pts total, -1 pt for each error, but not negative) MINI-MAX SEARCH/ALPHA-BETA PRUNING. The game tree below illustrates a position reached in the game. Process the tree left-to-right. It is **Max**'s turn to move. At each leaf node (number at bottom) is the estimated score returned by the heuristic static evaluator.

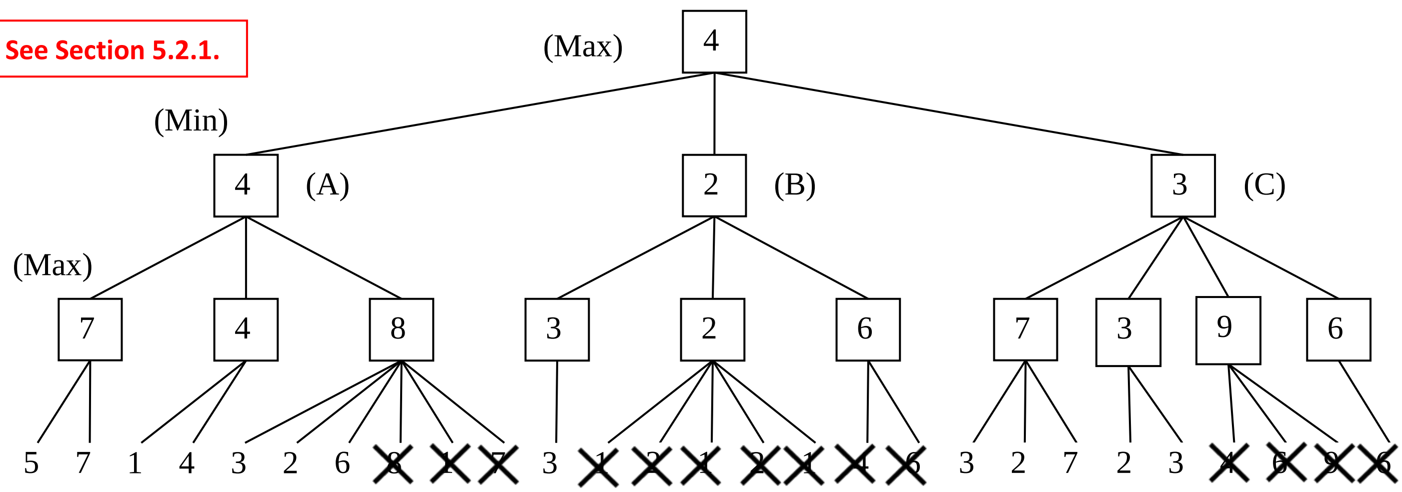
3.a. Fill in each blank square with the proper mini-max search value.

3.b. What is the best move for Max? (write A, B, or C) __A__

3.c. What score does Max expect to achieve? __4__

3.d. Draw X over each leaf node (number at bottom) that will be pruned by Alpha-Beta Pruning.

See Section 5.2.1.



4. (5 pts total, 1 pt each) Unifiers and Unification.

Write the **most general unifier** (or MGU) of the two terms given, or "None" if no unification is possible. Write your answer in the form of a substitution as given in your book, e.g., the substitution {*x / John, y / Mary, z / Bill*} means substitute *x* by *John,* substitute *y* by *Mary*, and substitute *z* by *Bill.*

The first one is done for you as an example.

4.a. (example) UNIFY(*Knows(John, x), Knows(John, Jane)*) __*{ x / Jane }*__

4.b. (1 pt) UNIFY(*Knows(John, x), Knows(y, Jane)*) __*{ x / Jane, y / John }*__

4.c. (1 pt) UNIFY(*Knows(John, x), Knows(y, Father (y))*) __{ y / John, x / Father (John) }__

4.d. (1 pt) UNIFY(*Knows(John, F(x)), Knows(y, F(F(z)))*) __*{ y / John, x / F (z) }*__

4.e. (1 pt) UNIFY(*Knows(John, F(x)), Knows(y, G(z))*) __None__

4.f. (1 pt) UNIFY(*Knows(John, F(x)), Knows(y, F(G(y)))*) __*{ y / John, x / G (John) }*__

****** TURN PAGE OVER AND CONTINUE ON THE OTHER SIDE ******

Problem 5 was canceled. Everyone gets it right, regardless of what you answered. The derivation shown below is correct.

5. (5 points if correct, e ... **ivation) Propositional Logic Conversion to CNF.** Convert this Propositional Logic wff (well-formed formula) to Conjunctive Normal Form and simplify. Show your work (correct result without work = 0 pts).

$$(P \Leftrightarrow Q) \Rightarrow (\neg Q \wedge R)$$

(one valid solution; if you failed to produce the correct answer, partial credit will be awarded up to 3 points, at 1 point per useful conversion.)

$[\,(P \Rightarrow Q) \wedge (Q \Rightarrow P)\,] \Rightarrow (\neg Q \wedge R)$
$= \neg\,[\,(\neg P \vee Q) \wedge (\neg Q \vee P)\,] \vee (\neg Q \wedge R)$
$= [\neg(\neg P \vee Q) \vee \neg(\neg Q \vee P)\,] \vee (\neg Q \wedge R)$
$= (P \wedge \neg Q) \vee (Q \wedge \neg P) \vee (\neg Q \wedge R)$
$= (P \vee Q \vee \neg Q) \wedge (P \vee Q \vee R) \wedge (P \vee \neg P \vee \neg Q) \wedge (P \vee \neg P \vee R) \wedge (\neg Q \vee Q \vee \neg Q) \wedge (\neg Q \vee Q \vee R)$
$\quad \wedge (\neg Q \vee \neg P \vee \neg Q) \wedge (\neg Q \vee \neg P \vee R)$
$= \text{TRUE} \wedge (P \vee Q \vee R) \wedge \text{TRUE} \wedge \text{TRUE} \wedge \text{TRUE} \wedge \text{TRUE} \wedge (\neg Q \vee \neg P) \wedge (\neg Q \vee \neg P \vee R)$
$= (P \vee Q \vee R) \wedge (\neg Q \vee \neg P) \wedge (\neg Q \vee \neg P \vee R)$
$= (P \vee Q \vee R) \wedge (\neg Q \vee \neg P) \wedge \text{TRUE}$
$= (P \vee Q \vee R) \wedge (\neg Q \vee \neg P)$

Another derivation is:
$[\,(P \wedge Q) \vee (\neg P \wedge \neg Q)\,] \Rightarrow (\neg Q \wedge R)$
$= \neg\,[\,(P \wedge Q) \vee (\neg P \wedge \neg Q)\,] \vee (\neg Q \wedge R)$
$= [\,\neg(P \wedge Q) \wedge \neg(\neg P \wedge \neg Q)\,] \vee (\neg Q \wedge R)$
$= [\,(\neg P \vee \neg Q) \wedge (P \vee Q)\,] \vee (\neg Q \wedge R)$
$= (\neg P \vee \neg Q \vee \neg Q) \wedge (P \vee Q \vee \neg Q) \wedge (\neg P \vee \neg Q \vee R) \wedge (P \vee Q \vee R)$
$= (\neg P \vee \neg Q) \wedge \text{TRUE} \wedge (\neg P \vee \neg Q \vee R) \wedge (P \vee Q \vee R)$
$= (\neg P \vee \neg Q) \wedge (\neg P \vee \neg Q \vee R) \wedge (P \vee Q \vee R)$
$= (\neg P \vee \neg Q) \wedge \text{TRUE} \wedge (P \vee Q \vee R)$
$= (\neg P \vee \neg Q) \wedge (P \vee Q \vee R)$

Problem 6 was canceled. Everyone gets it right, regardless of what you answered. The derivation shown below is correct.

6. (5 points if correct, else part ... **First Order Predicate Logic Conversion to CNF.** Convert this FOPL wff (well-formed formula) to Conjunctive Normal Form and simplify. Show your work (correct result without work = 0 pts).

"Everyone who loves all animals is loved by someone."

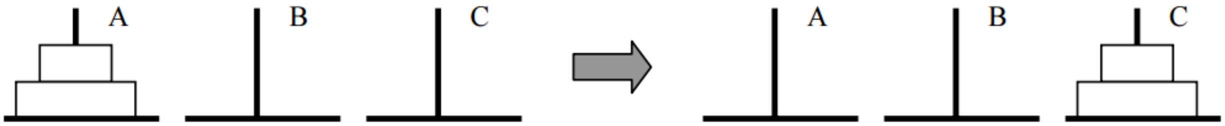
$$\forall x\,[\,\forall y\ Animal(y) \Rightarrow Loves(x,y)\,] \Rightarrow [\,\exists y\ Loves(y,x)\,]$$

(one valid solution; if you failed to produce the correct answer, partial credit will be awarded up to 3 points, at 1 point per useful conversion.)

$\forall x\ \neg\ [\,\forall y\ \neg Animal(y) \vee Loves(x,y)\,] \vee [\,\exists y\ Loves(y,x)\,]$
$= \forall x\,[\,\exists y\ \neg(\neg Animal(y) \vee Loves(x,y))\,] \vee [\,\exists y\ Loves(y,x)\,]$
$= \forall x\,[\,\exists y\ \neg\neg Animal(y) \wedge \neg Loves(x,y)\,] \vee [\,\exists y\ Loves(y,x)\,]$
$= \forall x\,[\,\exists y\ Animal(y) \wedge \neg Loves(x,y)\,] \vee [\,\exists y\ Loves(y,x)\,]$
$= \forall x\,[\,\exists y\ Animal(y) \wedge \neg Loves(x,y)\,] \vee [\,\exists z\ Loves(z,x)\,]$
$= \forall x\,[\,Animal(\,F(x)\,) \wedge \neg Loves(x,F(x)\,)\,] \vee Loves(G(x),x)$
$= [\,Animal(F(x)\,) \wedge \neg Loves(x,F(x)\,)\,] \vee Loves(G(x),x)$
$= (Animal(F(x)) \vee Loves(G(x),x)\,) \wedge (\,\neg Loves(x,F(x)) \vee Loves(G(x),x)\,)$

7. (10 pts total) PROBLEM SOLVING BY SEARCH/MINIATURE TOWER OF HANOI.
You are a robot tasked with solving the **MINIATURE TOWER OF HANOI** problem. There are 3 towers (A, B, C), and 2 disks (small one and large one). The goal is to move both disks from tower A to tower C (as illustrated in the figure below), subject to following three conditions:

- You can move only one disk at a time.
- You can move only the top disk in a stack of disks.
- You cannot put the large disk on top of the small disk.



The possible states can be denoted as follows:
x:(y, z) where **x** is the state number (below), **y** is the tower letter for the large disk, and **z** is the tower letter for the small disk. E.g., 3:(A C) says that in state 3 the large disk is on tower A and the small disk is on tower C.

If we use this notation, the following nine states are possible.
1:(A A), 2:(A B), 3:(A C), 4:(B A), 5:(B B), 6:(B C), 7:(C A), 8:(C B), 9:(C C)

7.a. (2 pts total, 1 pt each) Which states are the initial state and goal state?

7.a.1. (1 pt) Initial state: ___1 or 1:(A A)__

7.a.2. (1pt) Goal state: _____9 or 9:(C C)__

7.b. (4 pts total, -1 for each error, but not negative) Enumerate all one-move transitions between states.
For example, 1->2, 3 means that it is possible to make a transition from state 1 to state 2 or state 3 in one move. Include "back transitions;" e.g., since you are given the example 1->2, 3, necessarily states 2 and 3 must have a transition back to state 1.

The first one is done for you as an example.

1-> ___2, 3__________

2-> ___1, 3, 8________

3-> ___1, 2, 6________

4-> ___5, 6, 7________

5-> ___4, 6__________

6-> ___3, 4, 5________

7-> ___4, 8, 9________

8-> ___2, 7, 9________

9-> ___7, 8__________

[PROBLEM CONTINUES ON THE NEXT PAGE]

****** TURN PAGE OVER AND CONTINUE ON THE OTHER SIDE ******

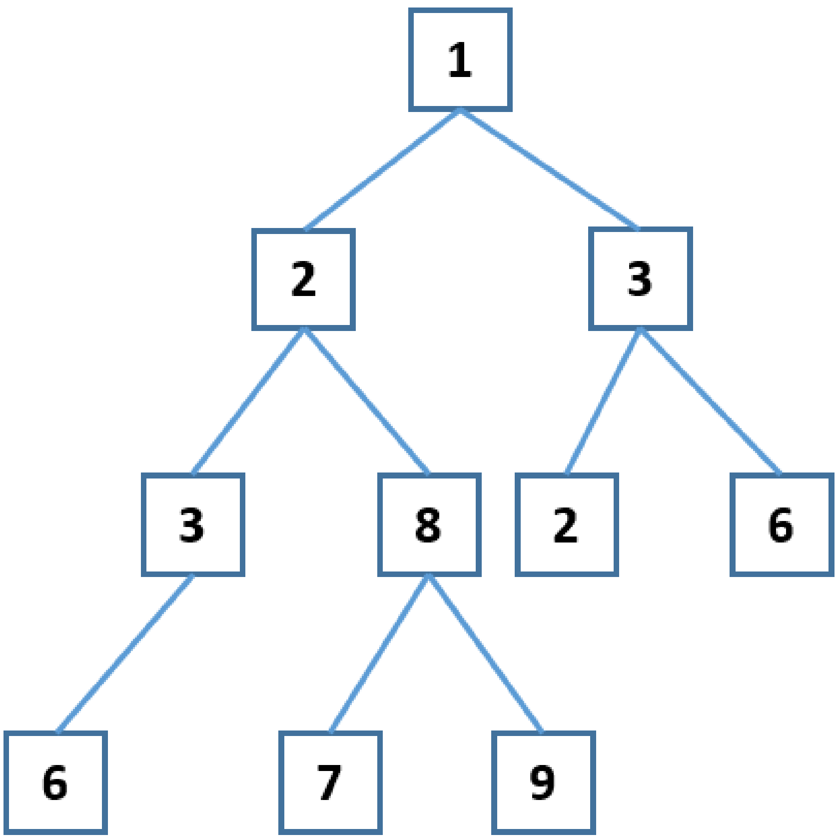
7.c. (4 pts total, -1 for each error, but not negative) PROBLEM SOLVING BY SEARCH (continued). Find a solution using breath-first search and draw the search tree. Stop when you reach goal state 9.

Use Tree Search, i.e., do not remember visited nodes. Children of a node are returned in increasing numerical order by state; i.e., the children of a node are ordered left-to-right by increasing state number.

Assume that cycles are detected and eliminated by never expanding a node containing a state that is repeated on the path back to the root. That is, if a newly-generated child node state also occurs on the path from the parent back to the root, then that child is pruned immediately. You do not need to write down nodes that are pruned in this way; you may assume that they are simply discarded and the search back-tracks immediately.

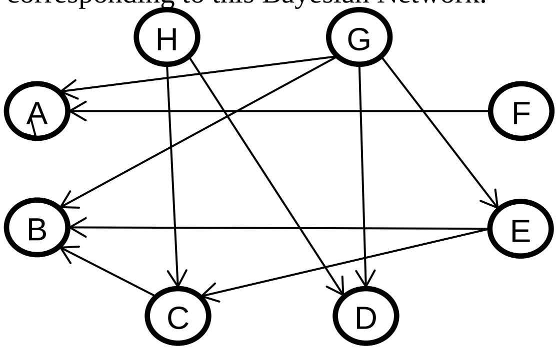
This pruning strategy is a "light-weight" way to avoid many cycles and repeated states in Tree Search without the full memory burden of remembering all visited nodes. However, it is an approximate strategy and does not fully solve the repeated node problem. (See the lecture notes on this topic.)

The first expansion is done for you as an example.



8. (10 pts total) BAYESIAN NETWORKS.

8.a. (3 pts total, -1 for each error, but not negative) Write down the factored conditional probability expression corresponding to this Bayesian Network:

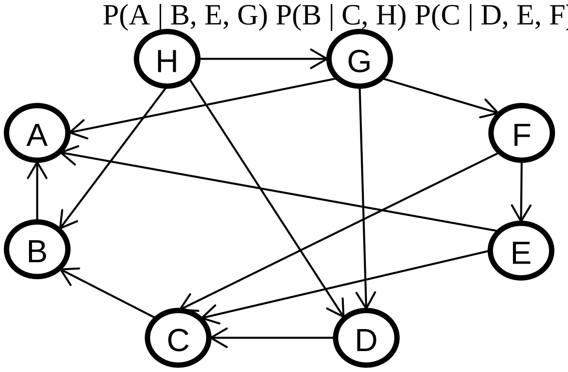


See Section 14.1-4.

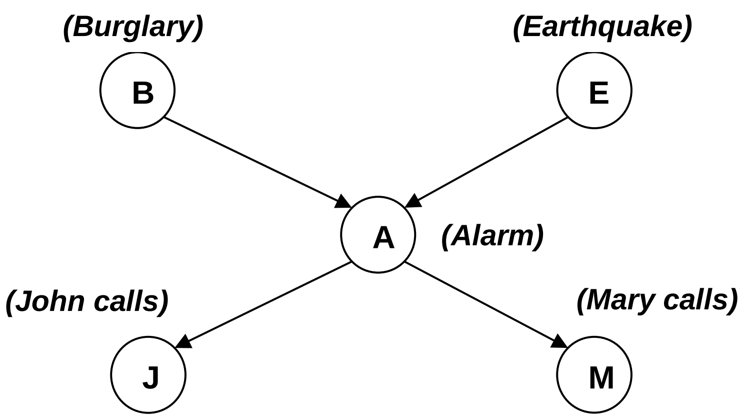
P(A | F, G) P(B | C, E, G) P(C | E, H) P(D | G, H) P(E | G) P(F) P(G) P(H)

8.b. (3 pts, -1 for each error, but not negative) Draw the Bayesian Network corresponding to this factored conditional probability expression:

P(A | B, E, G) P(B | C, H) P(C | D, E, F) P(D | G, H) P(E | F) P(F | G) P(G | H) P(H)



8.c. (4 pts, -1 for each error, but not negative) Shown below is the Bayesian network corresponding to the Burglar Alarm problem, i.e., P(J,M,A,B,E) = P(J | A) P(M | A) P(A | B, E) P(B) P(E). This is Fig. 14.2 in your R&N textbook. The probability tables show the probability that a boolean random variable is true. For example, P(J=t | A=t) is 0.9 and so P(J=f | A=t) is 1 – 0.9 = 0.1.



P(B)
.001

P(E)
.002

B	E	P(A)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

A	P(J)
t	.90
f	.05

A	P(M)
t	.70
f	.01

Write down an expression that will evaluate to P(J=f ∧ M=t ∧ A=t ∧ B=t ∧ E=f). **<u>Express your answer as a series of numbers (numerical probabilities) separated by multiplication symbols.</u>** You do not need to carry out the multiplication to produce a single number (probability).

P(J=f ∧ M=t ∧ A=t ∧ B=t ∧ E=f) = .10 * .70 * .94 * .001 * .998

****** TURN PAGE OVER AND CONTINUE ON THE OTHER SIDE ******

9. (7 pts total,
Below, "in ter

If your answer is mathematically correct and showed that you knew what you were doing and had mastered the material, then you get full credit even if it is not in one of the forms below. If your answer showed that you knew what you were doing and had mastered the material but had a very minor mistake, then you are likely to get partial credit. If your answer did <u>not</u> show that you knew what you were doing and had mastered the material, then you are likely to get little or no credit.

9.a. (1 pt) Wri

$P(A \wedge B)$ = $P(A) + P(B) - P(A \vee B)$

Other answers get full credit if they are mathematically correct. E.g., $P(A \wedge B) = P(A \vee B) - P(A \wedge \neg B) - P(B \wedge \neg A)$ is creative, but it gets full credit because it is mathematically correct & responsive to the problem.

9.b. (1 pt) Write the formula for the conditional probability $P(A \mid B)$.

$P(A \mid B)$ = $P(A \wedge B) / P(B)$

9.c. (1 pt) Factor $P(A \wedge B \wedge C)$ completely using the Product Rule (or Chain Rule). You may use any variable ordering you wish.

$P(A \wedge B \wedge C)$ = $P(A \mid B \wedge C) * P(B \mid C) * P(C)$

Other variable orderings are OK iff correct, e.g.,
$P(A \wedge B \wedge C) = P(C \mid A \wedge B) * P(B \mid A) * P(A)$
$= P(B \mid A \wedge C) * P(C \mid A) * P(A)$, etc.

9.d. (1 pt) Given a joint probability distribution $P(A \wedge B \wedge C)$, use the Sum Rule (or Law of Total Probability) to write the marginal probability of $P(A)$.

$P(A)$ = $\Sigma_{B,C}\, P(A \wedge B \wedge C)$

All are correct:
$P(A) = \Sigma_B \Sigma_C P(A \wedge B \wedge C)$
$= \Sigma_{b \in B} \Sigma_{c \in C} P(A \wedge b \wedge c)$
$= \Sigma_{b \in B, c \in C} P(A \wedge b \wedge c)$

9.e. (1 pt) Write Bayes' Rule (or Bayes' Theorem).

$$P(A \mid B) = P(B \mid A) * P(A) / P(B) = \frac{P(B \mid A) * P(A)}{\Sigma_{a \in A} P(B \mid a) * P(a)} = \frac{P(B \mid A) * P(A)}{P(B \mid a) * P(a) + P(B \mid \neg a) * P(\neg a)}$$

Bayes' Rule is written in several different forms in different places, any of which gets full credit if mathematically correct.

Iff A is Boolean.

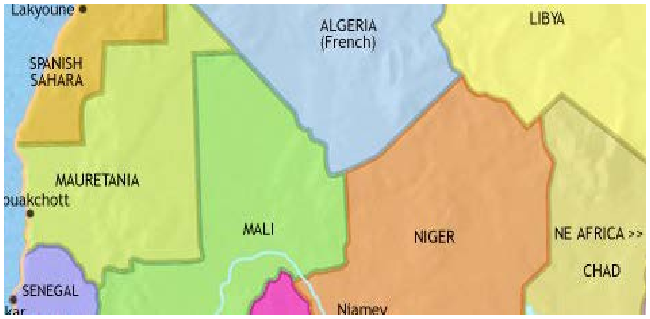
9.f. (1 pt) Assume that A and B [...] of $P(A)$ and $P(B)$ and possibly other terms.

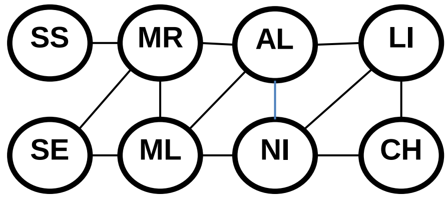
$P(A \wedge B)$ = $P(A) * P(B)$

9.g. (1 pt) Assume that A and B are conditionally independent given C. Write $P(A \wedge B \mid C)$ in terms of $P(A \mid C)$ and $P(B \mid C)$ and possibly other terms.

$P(A \wedge B \mid C)$ = $P(A \mid C) * P(B \mid C)$

10. (10 points total, 2 pts each) Constraint Satisfaction Problems





AL = Algeria
CH = Chad
LI = Libya
ML = Mali
MR = Mauretania
NI = Niger
SE = Senegal
SS = Spanish Sahara

You are a map-coloring robot assigned to color this map of west African countries. Adjacent regions must be colored a different color (R=Red, G=Green, B=Blue). The constraint graph is shown.

10.a. (2 pts total, -1 each w See Section 6.3.2. **not negative) FORWARD CHECKING.** NI has been assigned the value B, as shown; but no constraint propagation has been done. Cross out all values that would be eliminated by Forward Checking (FC):

AL	CH	LI	ML	MR	NI	SE	SS
R G ~~B~~	R G ~~B~~	R G ~~B~~	R G ~~B~~	R G B	B	R G B	R G B

10.b. (2 pts total, -1 each w See Section 6.3.2. **t not negative) ARC CONSISTENCY.**
NI has been assigned B and AL has been assigned R, as shown; but no constraint propagation has been done. Cross out all values that would be eliminated by Arc Consistency (AC-3 in your book).

AL	CH	LI	ML	MR	NI	SE	SS
R	R ~~G~~ ~~B~~	~~R~~ G ~~B~~	~~R~~ G ~~B~~	~~R~~ ~~G~~ B	B	R ~~G~~ ~~B~~	R G ~~B~~

10.c. (2 pts total, -1 each w See Section 6.3.1. **not negative) MINIMUM-REMAINING-VALUES (MRV) HEURISTIC.** Consider the assignment below. NI has been assigned B and constraint propagation has been done, as shown. List all unassigned variables (in any order) that might be selected now by the Minimum-Remaining-Values (MRV) Heuristic: **AL, CH, LI, ML** .

AL	CH	LI	ML	MR	NI	SE	SS
R G	R G	R G	R G	R G B	B	R G B	R G B

10.d. (2 pts total, -1 each w See Section 6.3.1. **not negative) DEGREE HEURISTIC (DH).** Consider the assignment below. (It is the same assignment as in problem 10.c above.) NI has been assigned B and constraint propagation has been done, as shown. **Ignoring the MRV heuristic,** list all unassigned variables (in any order) that might be selected now by the Degree Heuristic (DH):
MR .

AL	CH	LI	ML	MR	NI	SE	SS
R G	R G	R G	R G	R G B	B	R G B	R G B

10.e. (2 pts total, -1 each w See Section 6.3.1. **not negative) LEAST-CONSTRAINING-VALUE (LCV) HEURISTIC.** Consider the assignment below. (It is the same assignment as in problem 10.c above.) NI has been assigned B and constraint propagation has been done, as shown. **MR has been chosen as the next variable to explore.** List all values for MR that could be explored first by the Least-Constraining-Value Heuristic (LCV). **B** .

AL	CH	LI	ML	MR	NI	SE	SS
R G	R G	R G	R G	R G B	B	R G B	R G B

11. (3 pts if correct, else partial credit up to 2 pts as 1 pt for each correct non-trivial resolution) FOPL: Prove that Tweety is a bird. (Adapted from //www.cs.cornell.edu/courses/cs472/2007fa/lectures/17-kb-systems_fol.pdf)

You are a logic robot given the Knowledge Base below. Produce a FOPL resolution proof that Tweety is a bird.

Category	**FOPL wff**	**CNF clausal form**
General Knowledge about the domain (Laws of Physics)	∀x Feathers(x) ⇒ Bird(x)	¬ Feathers(x) ∨ Bird(x)
Description of a specific problem instance	Feathers(Tweety)	Feathers(Tweety)
Goal sentence	Bird(Tweety)	Bird(Tweety)
Negated Goal sentence	¬ Bird(Tweety)	¬ Bird(Tweety)

The KB sentences have been labeled Si in order to make them easy to refer to in your proof. Your resulting KB plus the negated goal sentence is as shown:

S1: ¬ Feathers(x) ∨ Bird(x)
S2: Feathers(Tweety)
S3: ¬ Bird(Tweety)

Produce a FOPL resolution proof that Tweety is a bird. On each line below, enter the two clauses in your KB that you wish to resolve (denoted as Si for ease in identification), the Most General Unifier (MGU) of those two clauses, and the resolvent clause that results from the unification and resolution. Insert the resolvent clause that results back into your KB, and repeat. The shortest proof that I know of is only two lines long. Longer proofs are OK as long as they are mathematically correct. Add extra lines if needed.

Below, fill in the blanks of "Resolve ________ with ________" with Si to designate the sentences to be resolved.

Fill in the blanks of "using MGU __________" with the substitution that makes the two resolved literals syntactically identical. Recall that an MGU substitution is written as {x/y, z/A}, which means that y is substituted for (= replaces) x and A is substituted for (= replaces) z. Write "None" if no substitution is needed.

End with "{ }" to indicate that you have produced the empty clause (a contradiction, proving the goal sentence).

Resolve ___S1___ with ___S2___ using MGU ___{x/Tweety}___ to yield S4 = ___Bird(Tweety)___

Resolve ___S3___ with ___S4___ using MGU ___None___ to yield S5 = ___{ }___

Resolve ___________ with ________ using MGU ______________ to yield S6 = ____________________

Resolve ___________ with ________ using MGU ______________ to yield S7 = ____________________

Resolve ___________ with ________ using MGU ______________ to yield S8 = ____________________

Resolve ___________ with ________ using MGU ______________ to yield S9 = ____________________

Resolve ___________ with ________ using MGU ______________ to yield S10 = ___________________

12. (10 pts total, 1 pt each) Machine Learning concepts.
For each of the following items on the left, write in the letter corresponding to the best answer or the correct definition on the right. The first one is done for you as an example.

A .	Learning	A	Improves performance of future tasks See the first sentence of Chapter 18.
J	Information Gain	B	Fixed set, list, or vector of features/attributes paired wit See Section 18.1.
H	Decision Boundary	C	Tests P (C) Π_i P(X_i | C), where C is a class label and X_i are See Section 20.2.2.
G	Cross-validation	D	Tests **w·f**>0, where **w** is a weight vector and **f** is a feature See Section 18.7.
D	Linear Classifier (Perceptron)	E	Example input-output pairs, from which to discover a hypothesis
B	Factored Representation (Feature Vector)	F	Examples distinct from training set, used to estimate accu See Section 18.2.
K	Overfitting	G	Randomly split the data into a training set and a test set See Section 18.4.1.
F	Test Set	H	Surface in a high-dimensional space that separates the clas See Section 18.6.3.
C	Naïve Bayes Classifier	I	Internal nodes test a value of an attribute, leaf nodes=class See Section 18.3.
E	Training Set	J	Expected reduction in entropy from testing an attribute v See Section 18.3.4.
I	Decision Tree	K	Choose an over-complex model based on irrelevant data See Section 18.3.5.

13. (5 pts if correct, else partial credit up to 3 pts as 1 pt for each correct non-trivial resolution) RESOLUTION PROOF THAT CHARLIE DID IT. (Adapted from http://www.braingle.com)

See Section 7.5.2

Detective Dorothy interviewed four local burglars to identify who stole Lady Diva's teapot.
It was well known that each burglar told exactly one lie:

Arnold: *I didn't do it. Brian did it.*
Brian: *I didn't do it. Derek did it.*
Charlie: *I didn't do it. Brian is lying when he says Derek did it.*
Derek: *I didn't do it. If Arnold didn't do it, then Brian did it.*

Use these propositional variables:

A= Arnold did it. **B**= Brian did it. **C**= Charlie did it. **D**= Derek did it.

You translate the evidence into propositional logic (recall that each suspect told exactly one lie):

Arnold: $(A \wedge B) \vee (\neg A \wedge \neg B)$ ***Brian:*** $(B \wedge D) \vee (\neg B \wedge \neg D)$
Charlie: $(C \wedge \neg D) \vee (\neg C \wedge D)$ ***Derek:*** $(D \wedge (\neg A \Rightarrow B)) \vee (\neg D \wedge \neg (\neg A \Rightarrow B))$

At most one burglar stole the teapot:

$(A \Rightarrow \neg B \wedge \neg C \wedge \neg D)$ $(B \Rightarrow \neg A \wedge \neg C \wedge \neg D)$ $(C \Rightarrow \neg A \wedge \neg B \wedge \neg D)$ $(D \Rightarrow \neg A \wedge \neg B \wedge \neg C)$

After converting to Conjunctive Normal Form, your Knowledge Base (KB) consists of:

Arnold: $(A \vee \neg B)$ $(\neg A \vee B)$ ***Brian:*** $(B \vee \neg D)$ $(\neg B \vee D)$
Charlie: $(C \vee D)$ $(\neg C \vee \neg D)$ ***Derek:*** $(\neg A \vee D)$ $(\neg B \vee D)$ $(A \vee B \vee \neg D)$
At most one: $(\neg A \vee \neg B)$ $(\neg A \vee \neg C)$ $(\neg A \vee \neg D)$ $(\neg B \vee \neg C)$ $(\neg B \vee \neg D)$ $(\neg C \vee \neg D)$

(Side note: Normally, you would start four proofs, one for each goal sentence: A, B, C, D. Only the proof of C would succeed, and you would know Charlie did it. For this timed test, you will do only one proof.)

You will be asked to prove, **"Charlie did it."** The goal is (C). You adjoin the negated goal to your KB:

$(\neg C)$

Other proofs are OK as long as they are correct. For example, a three-line proof is:
Resolve $(B \vee \neg D)$ and $(\neg B \vee \neg D)$ to give $(\neg D)$.
Resolve $(C \vee D)$ and $(\neg D)$ to give (C).
Resolve (C) and $(\neg C)$ to give $( )$.

Produce a resolution p

Repeatedly choose two ... second. Apply resolution to the ... ge base. Continue until you produce ( ). If you cannot produce ( ), then you have made a mistake. The shortest proof I know is only three lines. It is OK to use more lines, if your proof is correct. It is OK to use abbreviated CNF, i.e., $(\neg A \ \neg B)$ instead of $(\neg A \vee \neg B)$. It is OK to omit the parentheses.

Resolve $(C \vee D)$ and $(\neg C)$ to give (D).

Resolve $(B \vee \neg D)$ and (D) to give (B).

Resolve $(A \vee \neg B)$ and (B) to give (A).

Resolve $(\neg A \vee \neg B)$ and (B) to give $(\neg A)$.

Resolve $(\neg A)$ and (A) to give $( )$.

Resolve ______

Resolve ______

Other proofs are OK as long as they are correct. For example, another proof is:
Resolve $(\neg C)$ and $(C \vee D)$ to give (D).
Resolve (D) and $(\neg B \vee \neg D)$ to give $(\neg B)$.
Resolve $(\neg B)$ and $(B \vee \neg D)$ to give $(\neg D)$.
Resolve $(\neg D)$ and (D) to give $( )$.

Extra lines or steps are OK as long as your proof is correct.

It is OK (even expected) to simplify expressions. E.g., if you resolved $(\neg B \vee \neg D)$ and $(B \vee \neg D)$ to give $(\neg D \vee \neg D)$, of course you would simplify it to $(\neg D)$. It is OK to simplify as you go, i.e., you don't need a separate step.