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

When you are told to begin the exam, please check first to make sure that you have all seven pages, as numbered 1-7 in the bottom-left corner of each page.

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

Please clear your desk entirely, except for pen, pencil, eraser, a blank piece of paper (for scratch pad use), and an optional water bottle. Please write your name and ID# on the blank piece of paper and turn it in with your exam.

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

1. (10 pts total, 1 pt each) ENGLISH TO FOPC CONVERSION.
2. (10 pts total, -2 for each error, but not negative) FOPC RESOLUTION THEOREM PROVING.
3. (10 pts total, 2 pts each) SEARCH.
4. (10 pts total, 5 pts each) MATCHING THE HYPOTHESIS SPACE TO THE PROBLEM.
5. (10 pts total) DECISION TREE CLASSIFIER LEARNING.
6. (10 pts, 1 pt each) PROBABILITY.
7. (10 points total, 2 pts each) CONSTRAINT SATISFACTION PROBLEMS.
8. (10 pts total, -2 for each error, but not negative) BAYESIAN NETWORKS.
9. (10 pts total, -2 pt for each error, but not negative) ALPHA-BETA PRUNING.
10. (10 pts total, 2 pts each) FOPC KNOWLEDGE ENGINEERING IN THE BLOCKS WORLD.
1. (10 pts total, 1 pt each) ENGLISH TO FOPC CONVERSION.
For each English sentence below, write the FOPC sentence that best expresses its intended meaning.

1.a. (2pts) “All persons are mortal.”
[Use: Person(x), Mortal(x)]

\[ \forall x \text{ Person}(x) \Rightarrow \text{Mortal}(x) \]
\[ \forall x \neg \text{Person}(x) \lor \text{Mortal}(x) \]

1.b. (2pts) “Fifi has a sister who is a cat.”
[Use: Sister(Fifi, x), Cat(x)]

\[ \exists x \text{ Sister}(\text{Fifi}, x) \land \text{Cat}(x) \]

1.c. (2pts) “For every food, there is a person who eats that food.”
[Use: Food(x), Person(y), Eats(y, x)]

\[ \forall x \exists y \text{ Food}(x) \Rightarrow [\text{Person}(y) \land \text{Eats}(y, x)] \]
\[ \forall x \exists y \neg \text{Food}(x) \lor [\text{Person}(y) \land \text{Eats}(y, x)] \]
\[ \forall x \exists y [\neg \text{Food}(x) \lor \text{Person}(y)] \land [\neg \text{Food}(x) \lor \text{Eats}(y, x)] \]
\[ \forall x \exists y [\text{Food}(x) \Rightarrow \text{Person}(y)] \land [\text{Food}(x) \Rightarrow \text{Eats}(y, x)] \]

1.d. (2pts) “Every person eats every food.”
[Use: Person(x), Food(y), Eats(x, y)]

\[ \forall x \forall y \text{ Person}(x) \land \text{Food}(y) \Rightarrow \text{Eats}(x, y) \]
\[ \forall x \forall y \neg \text{Person}(x) \lor \neg \text{Food}(y) \lor \text{Eats}(x, y) \]
\[ \forall x \forall y \text{ Person}(x) \Rightarrow [\neg \text{Food}(y) \lor \text{Eats}(y, x)] \]
\[ \forall x \forall y \neg \text{Person}(x) \lor [\text{Food}(y) \Rightarrow \text{Eats}(x, y)] \]

1.e. (2 pts) “All greedy kings are evil.”
[Use: King(x), Greedy(x), Evil(x)]

\[ \forall x [\text{Greedy}(x) \land \text{King}(x)] \Rightarrow \text{Evil}(x) \]
\[ \forall x \neg \text{Greedy}(x) \lor \neg \text{King}(x) \lor \text{Evil}(x) \]
\[ \forall x \text{ Greedy}(x) \Rightarrow [\text{King}(x) \Rightarrow \text{Evil}(x)] \]

1.f. (2pts) “Everyone has a favorite food.”
[Use: Person(x), Food(y), Favorite(y, x)]

\[ \forall x \exists y \text{ Person}(x) \Rightarrow [\text{Food}(y) \land \text{Favorite}(y, x)] \]
\[ \forall x \exists y \neg \text{Person}(x) \lor [\text{Food}(y) \land \text{Favorite}(y, x)] \]
\[ \forall x \exists y [\neg \text{Person}(x) \lor \text{Food}(y)] \land [\neg \text{Person}(x) \lor \text{Favorite}(y, x)] \]
\[ \forall x \exists y [\text{Person}(x) \Rightarrow \text{Food}(y)] \land [\text{Person}(x) \Rightarrow \text{Favorite}(y, x)] \]

1.g. (2pts) “There is someone at UCI who is smart.”
[Use: Person(x), At(x, UCI), Smart(x)]

\[ \exists x \text{ Person}(x) \land \text{At}(x, \text{UCI}) \land \text{Smart}(x) \]

1.h. (2pts) “Everyone at UCI is smart.”
[Use: Person(x), At(x, UCI), Smart(x)]

\[ \forall x [\text{Person}(x) \land \text{At}(x, \text{UCI})] \Rightarrow \text{Smart}(x) \]
\[ \forall x \neg [\text{Person}(x) \land \text{At}(x, \text{UCI})] \lor \text{Smart}(x) \]
\[ \forall x \neg \text{Person}(x) \lor \neg \text{At}(x, \text{UCI}) \lor \text{Smart}(x) \]
1.i. (2pts) “Every person eats some food.”

[Use: Person (x), Food (y), Eats(x, y)]

∀x Ǝy Person(x) ⇒ [ Food(y) ∧ Eats(x, y) ]
∀x Ǝy ¬Person(x) ∨ [ Food(y) ∧ Eats(x, y) ]
∀x Ǝy [ ¬Person(x) ∨ Food(y) ] ∧ [ ¬Person(x) ∨ Eats(x, y) ]

1.j. (2pts) “Some person eats some food.”

[Use: Person (x), Food (y), Eats(x, y)]

Ǝx Ǝy Person(x) ∧ Food(y) ∧ Eats(x, y)
2. (10 pts total, -2 for each error, but not negative) FOPC RESOLUTION THEOREM PROVING.
You are a Knowledge Engineer for the genealogy domain. You interview an expert on genealogy who tells you, among other things, “One’s grandmother is defined as the mother of one’s parent.” You translate this into FOPC as “∀x ∀y Gmom(x, y) ⇔ ∃z Mom(x, z) ∧ Parent(z, y)” and into CNF as
( ¬Gmom(x1, y1) ∨ Mom(x1, F(x1, y1)) ) ( ¬Gmom(x2, y2) ∨ Parent(F(x2, y2), y2) )
( ¬Mom(x3, z3) ∨ ¬Parent(z3, y3) ∨ Gmom(x3, y3) )
where F(x, y) is a Skolem function.

The expert also says, “One’s parent is defined as one’s mother or one’s father.” You translate this into FOPC as “∀x ∀y Parent(x, y) ⇔ Mom(x, y) ∨ Dad(x, y)” and into CNF as
( ¬Mom(x4, y4) ∨ Parent (x4, y4) ) ( ¬Dad(x5, y5) ∨ Parent (x5, y5) )
( ¬Parent(x6, y6) ∨ Mom(x6, y6) ∨ Dad(x6, y6) )
You are told “Meg is Ann’s mother.” and “Ann is Rod’s mother.” You translate these into CNF as
( Mom(Meg, Ann) ) ( Mom(Ann, Rod) )
You are asked, “Is it true that Meg is the grandmother of Rod?” You form the negated goal sentence as
( ¬Gmom(Meg, Rod) )

Run resolution on this knowledge base until you produce the null clause, “( )”, thereby proving that the goal sentence is true. The shortest proof I know of is only four lines, including the first example line. It is OK to use more lines, if your proof is correct. SHOW YOUR WORK.
Repeatedly choose two clauses, write one in the first blank space, the other in the second, and their unifier in the third. Apply resolution to them. Write the resulting clause in the fourth blank space, and insert it into the knowledge base. Remember always to use different variable names in each wff. The first one is done for you as an example.
(For non-English speakers: “Mom” means “Mother” and “Dad” means “Father.”)

(1) Resolve    ( ¬Gmom(Meg, Rod) )
and    ( ¬Mom(x3, z3) ∨ ¬Parent(z3, y3) ∨ Gmom(x3, y3) )
with unifier    { x3 / Meg, y3 / Rod }
to give    ( ¬Mom(Meg, z7) ∨ ¬Parent(z7, Rod) )

(2) Resolve    ( ¬Mom(Meg, z7) ∨ ¬Parent(z7, Rod) )
and    ( Mom(Meg, Ann) )
with unifier    { z7 / Ann }
to give    ( ¬Parent(Ann, Rod) )

(3) Resolve    ( ¬Parent(Ann, Rod) )
and    ( ¬Mom(x4, y4) ∨ Parent (x4, y4) )
with unifier    { x4 / Ann, y4 / Rod }
to give    ( ¬Mom(Ann, Rod) )
(4) Resolve \( \neg \text{Mom}(\text{Ann}, \text{Rod}) \) and \( \text{Mom}(\text{Ann}, \text{Rod}) \) with unifier \{ \} to give \( \) 

(5) Resolve and with unifier to give 

(6) Resolve and with unifier to give 

(7) Resolve and with unifier to give 

(8) Resolve and with unifier to give 

(9) Resolve and with unifier to give
3. (10 pts total, 2 pts each) SEARCH. 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.

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. **(Note: C is a successor of itself).**

For each search strategy below, show the order in which nodes are expanded (i.e., to expand a node means that its children are generated), ending with the goal node that is found. Show the path from start to goal, or write “None”. The first one is done for you as an example.

3.a. (2 pts) DEPTH FIRST SEARCH.

Order of node expansion: S A D G1

Path found: S A D G1

3.b. (2 pts) UNIFORM COST SEARCH.

Order of node expansion: S C B A F C E D F C G1

Path found: S A D G1

3.c. (2 pts) GREEDY (BEST-FIRST) SEARCH.

Order of node expansion: S C C C C C C C etc.

Path found: None

3.d. (2 pts) ITERATED DEEPENING SEARCH.

Order of node expansion: S S A B C S A D B E C F C S A D G1

Path found: S A D G1

3.e. (2 pts) A* SEARCH.

Order of node expansion: S C B A F C E G2

Path found: S B E G2
4. (10 pts total, 5 pts each) MATCHING THE HYPOTHESIS SPACE TO THE PROBLEM.
For each set of training data shown below, write the key corresponding to the machine learning classifier that best matches the problem, i.e., that best can separate “X” from “O” in the graphs.

“LC” = Linear Classifier  “DT” = Decision Tree  “B” = Both  “N” = None
Then draw the decision boundaries for that classifier that provide the best classification.
The first one is done for you as an example.

4.a. **LC**

![Diagram](image1)

4.b. (5 pts) **DT**

![Diagram](image2)

4.c. (5 pts) **B**

![Diagram](image3)
5. (10 pts total) DECISION TREE CLASSIFIER LEARNING. You are a robot in a lumber yard, and must learn to discriminate Oak wood from Pine wood. You choose to learn a Decision Tree classifier. You are given the following examples:

<table>
<thead>
<tr>
<th>Example</th>
<th>Density</th>
<th>Grain</th>
<th>Hardness</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example #1</td>
<td>Heavy</td>
<td>Small</td>
<td>Hard</td>
<td>Oak</td>
</tr>
<tr>
<td>Example #2</td>
<td>Heavy</td>
<td>Large</td>
<td>Hard</td>
<td>Oak</td>
</tr>
<tr>
<td>Example #3</td>
<td>Heavy</td>
<td>Small</td>
<td>Hard</td>
<td>Oak</td>
</tr>
<tr>
<td>Example #4</td>
<td>Light</td>
<td>Large</td>
<td>Soft</td>
<td>Oak</td>
</tr>
<tr>
<td>Example #5</td>
<td>Light</td>
<td>Large</td>
<td>Hard</td>
<td>Pine</td>
</tr>
<tr>
<td>Example #6</td>
<td>Heavy</td>
<td>Small</td>
<td>Soft</td>
<td>Pine</td>
</tr>
<tr>
<td>Example #7</td>
<td>Heavy</td>
<td>Large</td>
<td>Soft</td>
<td>Pine</td>
</tr>
<tr>
<td>Example #8</td>
<td>Heavy</td>
<td>Small</td>
<td>Soft</td>
<td>Pine</td>
</tr>
</tbody>
</table>

If root is Density:
- Heavy = OOPPP, Light = OP
If root is Grain:
- Small = OOPP, Large = OOPP
If root is Hardness:
- Hard = OOOP, Soft = OPPP

(O = Oak, P = Pine)

5a. (2 pts) Which attribute would information gain choose as the root of the tree?

Hardness

5b. (4 pts) Draw the decision tree that would be constructed by recursively applying information gain to select roots of sub-trees, as in the Decision-Tree-Learning algorithm.

Half credit for the correct root; half credit for wrong root but correct classification; full credit for the correct tree.

Classify these new examples as Oak or Pine using your decision tree above.

5c. (2 pts) What class is [Density=Light, Grain=Small, Hardness=Hard]?  Pine

5d. (2 pts) What class is [Density=Light, Grain=Small, Hardness=Soft]?  Oak

6. (10 pts, 1 pt each) PROBABILITY. Let $\alpha$, $\beta$, and $\gamma$ be any three events. Mark each of the following statements as “T” (= true) or “F” (= false) in the general case, subject to any conditions noted.

<table>
<thead>
<tr>
<th>Statement</th>
<th>True/False</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0 \leq P(\alpha) \leq 1$</td>
<td>T</td>
</tr>
<tr>
<td>$P(\alpha</td>
<td>\beta) = P(\alpha) P(\beta)$</td>
</tr>
<tr>
<td>$P(\alpha \lor \beta) = P(\alpha) + P(\beta)$</td>
<td>; instead, $P(\alpha \lor \beta) = P(\alpha) + P(\beta) - P(\alpha \land \beta)$</td>
</tr>
<tr>
<td>$P(\alpha \land \beta</td>
<td>\gamma) = P(\alpha</td>
</tr>
<tr>
<td>$P(\alpha \lor \beta) = P(\alpha) + P(\beta)$</td>
<td>; when $\alpha$ and $\beta$ are independent</td>
</tr>
<tr>
<td>$P(\alpha \land \beta) = P(\alpha) + P(\beta)$</td>
<td>; Bayes’ Rule --- memorize!!</td>
</tr>
<tr>
<td>$P(\alpha</td>
<td>\beta) = P(\alpha</td>
</tr>
<tr>
<td>$P(\alpha \land \beta \land \gamma) = P(\alpha</td>
<td>\beta \land \gamma) P(\beta</td>
</tr>
</tbody>
</table>

See Section 18.3.
7. (10 points total, 2 pts each) CONSTRAINT SATISFACTION PROBLEMS.

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

7a. (2pts total, -1 each wrong answer, but not negative) FORWARD CHECKING.
Cross out all values that would be eliminated by Forward Checking, after variable MA has just been assigned value R as shown:

<table>
<thead>
<tr>
<th>CT</th>
<th>RI</th>
<th>MA</th>
<th>VT</th>
<th>NH</th>
<th>ME</th>
</tr>
</thead>
<tbody>
<tr>
<td>G B</td>
<td>G B</td>
<td>R</td>
<td>G B</td>
<td>R</td>
<td>R G B</td>
</tr>
</tbody>
</table>

7b. (2pts total, -1 each wrong answer, but not negative) ARC CONSISTENCY.
CT and RI have been assigned values, but no constraint propagation has been done. Cross out all values that would be eliminated by Arc Consistency (AC-3 in your book).

<table>
<thead>
<tr>
<th>CT</th>
<th>RI</th>
<th>MA</th>
<th>VT</th>
<th>NH</th>
<th>ME</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>G</td>
<td>X B</td>
<td>R</td>
<td>G B</td>
<td>R</td>
</tr>
</tbody>
</table>

7c. (2pts total, -1 each wrong answer, but not negative) MINIMUM-REMAINING-VALUES HEURISTIC. Consider the assignment below. RI is assigned and constraint propagation has been done. List all unassigned variables that might be selected by the Minimum-Remaining-Values (MRV) Heuristic: __________ CT, MA __________.

<table>
<thead>
<tr>
<th>CT</th>
<th>RI</th>
<th>MA</th>
<th>VT</th>
<th>NH</th>
<th>ME</th>
</tr>
</thead>
<tbody>
<tr>
<td>R B</td>
<td>G</td>
<td>R B</td>
<td>R</td>
<td>G B</td>
<td>R</td>
</tr>
</tbody>
</table>

7d. (2pts total, -1 each wrong answer, but not negative) DEGREE HEURISTIC.
Consider the assignment below. (It is the same assignment as in problem 7c above.) RI is assigned and constraint propagation has been done. List all unassigned variables that might be selected by the Degree Heuristic: ______________ MA, NH ______________.

<table>
<thead>
<tr>
<th>CT</th>
<th>RI</th>
<th>MA</th>
<th>VT</th>
<th>NH</th>
<th>ME</th>
</tr>
</thead>
<tbody>
<tr>
<td>R B</td>
<td>G</td>
<td>R B</td>
<td>R</td>
<td>G B</td>
<td>R</td>
</tr>
</tbody>
</table>

7e. (2pts total) MIN-CONFLICTS HEURISTIC. Consider the complete but inconsistent assignment below. MA has just been selected to be assigned a new value during local search for a complete and consistent assignment. What new value would be chosen below for MA by the Min-Conflicts Heuristic? ______________ R ______________.

<table>
<thead>
<tr>
<th>CT</th>
<th>RI</th>
<th>MA</th>
<th>VT</th>
<th>NH</th>
<th>ME</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>G</td>
<td>?</td>
<td>G</td>
<td>G</td>
<td>B</td>
</tr>
</tbody>
</table>
8. (10 pts total, -2 for each error, but not negative) BAYESIAN NETWORKS. Shown below is the Bayesian network corresponding to the Burglar Alarm problem, \( P(J \mid A) \, P(M \mid A) \, P(A \mid B, E) \, P(B) \, P(E) \).

\[
\begin{array}{c}
\text{(Burglary)} \\
B \\
\text{(Earthquake)} \\
E \\
\text{(Alarm)} \\
A \\
\text{(John calls)} \\
J \\
\text{(Mary calls)} \\
M
\end{array}
\]

Write down an expression that will evaluate to \( P(j=F \land m=T \land a=T \land b = T \land e = F) \). Express your answer as a series of numbers (numerical probabilities) separated by multiplication symbols. You do not need to carry out the multiplication to produce a single number (probability). **SHOW YOUR WORK.**

\[
P(j=F \land m=T \land a=T \land b = T \land e = F) \\
= P(j=F \mid a=T) * P(m=T \mid a = T) * P(a=T \mid b=T \land e=F) * P(b=T) * P(e=F) \\
= .10 * .70 * .94 * .001 * .998
\]

9. (10 pts total, -2 pt for each error, but not negative) 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 is the estimated score of the heuristic static evaluator.

Cross out each leaf node that will be pruned by Alpha-Beta Pruning.

Red lines indicate where in the tree pruning occurred. You are not obliged to provide the red lines — only to cross out pruned leaf nodes.

See Section 5.3.
10. (10 pts total, 2 pts each) FOPC KNOWLEDGE ENGINEERING IN THE BLOCKS WORLD.
You are a Knowledge Engineer assigned to the Blocks World, which involves controlling a robot arm that stacks children’s blocks one atop another, or on a table, into a desired configuration. For example, you are concerned with configurations such as:

Here we wish only to describe static (unmoving) configurations that will become goals (targets) in the Blocks World. A separate module, not your concern, will move the robot arm to achieve these goals. You need only to describe correctly in FOPC the static goal (target) configurations.

Write the FOPC sentence that best represents the corresponding English sentence. Assume that all objects in the world are blocks, i.e., there is no need for Block(x) guard predicates. Use “Stacked(x, y)” to mean that “Block x is stacked directly on top of block y.”
The first one is done for you as an example.

10.a. Assert that “Block x is stacked on block y” implies “Block y is not stacked on block x.”

\[ ∀x, y \text{ Stacked}(x, y) ⇒ \neg \text{Stacked}(y, x) \]

10.b. (2pts) Define a predicate “Clear(x)” to mean that no block y is stacked on block x.

\[ ∀x, y \text{ Clear}(x) ⇔ \neg \text{Stacked}(y, x) \]

10.c. (2pts) Define a predicate “OnTable(x)” to mean that block x is on the table, i.e., not on any block.

\[ ∀x, y \text{ OnTable}(x) ⇔ \neg \text{Stacked}(x, y) \]

10.d. (2pts) Define a predicate “Above(x, y)” to mean that x is above y in a stack including both x and y.

\[ ∀x, y \text{ Above}(x, y) ⇔ [ \text{Stacked}(x, y) \lor ( \exists z \text{Stacked}(z, y) \land \text{Above}(x, z) ) ] \]

10.e. (2pts) State that at least one block must have no other block stacked upon it. You may use the Clear(x) predicate that you defined in (10.b) above

\[ \exists x \text{Clear}(x) \]
\[ \exists x \forall y \neg \text{Stacked}(y, x) \]

10.f. State that at least one block must be on the table. You may use the OnTable(x) predicate that you defined in (10.c) above

\[ \exists x \text{OnTable}(x) \]
\[ \exists x \forall y \neg \text{Stacked}(x, y) \]

**** THIS IS THE END OF THE FINAL EXAM ****