**CAP 6635 Artificial Intelligence**

**Homework 4 [12 pts, Due March 27 2023]**

[Homework solutions must be submitted through Canvas. Only pdf, word, and txt files are allowed. If you have multiple pictures, please include all pictures in one Word/pdf file. You can always update your submissions before due date, but only the latest version will be graded.]

*When traversing a game tree, use left-to-right policy (left child has a higher priority than the right child).*

**1. Question 1 [1.5 pts]:** Figure 1 shows a portion of an imperfect tic-tac-toe game tree using evaluation function. Assume X is the Max player and O is the Min player. Assume a heuristic function is defined as the number of X’s winning position subtract the number of O’s winning position. For example, for boardDiagram

Description automatically generated layout, X has 6 winning positions, and O has 5 winning positions. Therefore, the heuristic value of this board layout is 6-5=1.

* Using defined heuristic function, list the heuristic values for all leaf nodes. [0.5 pt]
* Assume X is the root player, applying α-β pruning to the game tree in Figure 1, determine α value for Max node, and β value for Min nodes. [0.25]
* Determine nodes which are pruned by α-β pruning [0.5], and also specify the best move for X. [0.25 pt]

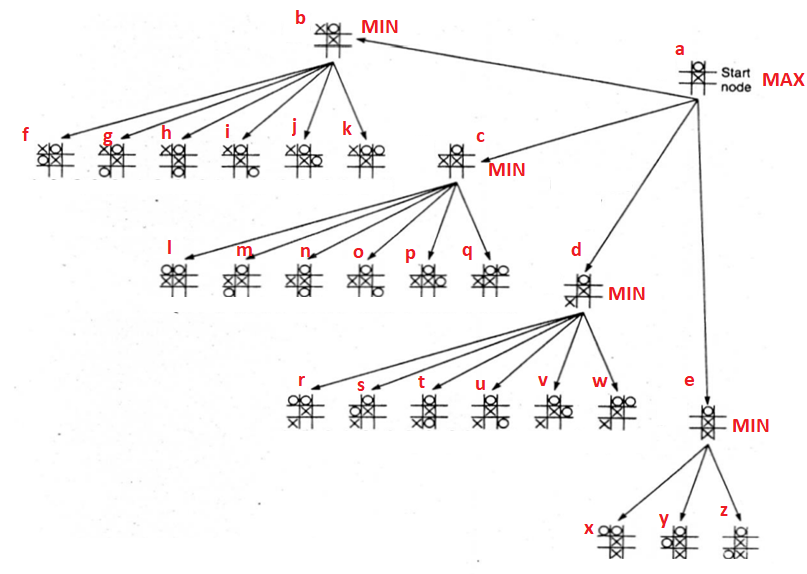


Figure 1



**Question 2 [1 pt]:** VariableWa,bdefines that a Wumpus is located as location [a,b], and Sx,y denotes that a stench is sensed at location [x,y].

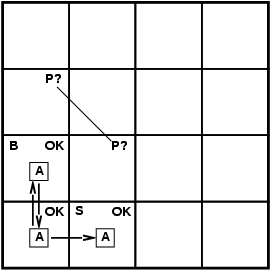
* Explain the semantic (meaning) of the following sentence?[0.5 pt]
* **S**how detailed steps to decompose sentence into CNF format [0.5 pt]

W1,3 ⇔ (S1,2 ∨ S1,4∨ S2,3)



**Question 3 [2.5 pts]** Figure 2 shows the Wumpus world game, where the agent starts from location [1,1], and does not sense breeze or stench. After that, the agent moves to location [1,2] and sense a Breeze (B). Then the agent moved back to [1,1], and further moved to location [2,1] and senses a Stench (S). Based on the above observations and the Wumpus world game rules, please use resolution algorithm to prove following entailment.

* KB╞ ¬W1,3 (There is no Wumpus at location [1,3]) [0.5 pt]
* *KB* ╞ W3,1 (There is a Wumpus at location [3,1]) [0.5 pt]
* *KB* ╞ P1,3 (There is a Pit at location [1,3]) [0.5 pt]
* *KB* ╞ ¬ P2,2 (There is no Pit at location [2,2]) [0.5 pt]
* *KB* ╞ ¬ W2,2 (There is a no Wumpus at location [2,2]) [0.5 pt]

****

**Figure 2**

Text, letter

Description automatically generated

* + Let Pi,j be true if there is a pit in [i,j].
  + Let Bi,j be true if there is a breeze in [i,j].
  + Let Si,j be true if there is a stench in [i,j].
  + Bi,j ⬄ There is a pit in an adjacent cell.
  + Si,j ⬄ There is a wumpus in an adjacent cell.
* KB╞ ¬W1,3 (There is no Wumpus at location [1,3]) [0.5 pt]

R1: ¬ S1,2

R2: ¬ S1,2 ⬄ (¬ W1,3 ∧ ¬ W2,2)

KB = {R1, R2}

¬ S1,2 => (¬ W1,3 ∧ ¬ W2,2) ∧ (¬ W1,3 ∧ ¬ W2,2) => ¬ S1,2

¬ ¬ S1,2 ∨ (¬ W1,3 ∧ ¬ W2,2) ∧ ¬ (¬ W1,3 ∧ ¬ W2,2) ∨ ¬ S1,2

(S1,2 ∨ (¬ W1,3 ∧ ¬ W2,2)) ∧ ((W1,3 ∨ W2,2) ∨ ¬ S1,2)

(T OR (F AND F)) AND ((T OR T) OR F)

(T OR F) AND (T OR F)

T AND T = TRUE

You can plug these clauses into the algorithm, and this proves the entailment is True.

* *KB* ╞ W3,1 (There is a Wumpus at location [3,1]) [0.5 pt]

R1: S2,1

R2: B1,2

R3: S2,1 ⬄ (W2,2 ∨ W3,1)

R4: ¬ S1,2 ⬄ (¬ W1,3 ∧ ¬ W2,2)

(S2,1 ⬄ (W2,2 ∨ W3,1)) ^ (¬ S1,2 ⬄ (¬ W1,3 ∧ ¬ W2,2))

((S2,1 => (W2,2 ∨ W3,1)) ^ ((W2,2 ∨ W3,1)=> S2,1))^((¬ S1,2 =>(¬ W1,3 ∧ ¬ W2,2)) ^ ((¬ W1,3 ∧ ¬ W2,2)=> ¬ S1,2)

(¬ S2,1 v (W2,2 ∨ W3,1)) ^ (¬ (W2,2 ∨ W3,1) v S2,1)^ (¬ ¬ S1,2 ∨ (¬ W1,3 ∧ ¬ W2,2) ∧ ¬ (¬ W1,3 ∧ ¬ W2,2) ∨ ¬ S1,2 )

FROM ABOVE EXAMPLES WE CAN DERIVE: TRUE AND TRUE = TRUE

You can plug these clauses into the algorithm, and this proves the entailment is True.

* *KB* ╞ P1,3 (There is a Pit at location [1,3]) [0.5 pt]

R1: B1,2

R2: ¬ B2,1

R3: B1,2 ⬄ (P1,3 v P2,2)

R4: ¬ B2,1 ⬄ (¬ P2,2 v ¬ P3,1)

(B1,2 ⬄ (P1,3 v P2,2)) ^ (¬ B2,1 ⬄ (¬ P2,2 v ¬ P3,1))

((B1,2 => (P1,3 v P2,2))^((P1,3 v P2,2)=> B1,2))^ ((¬ B2,1 => (¬ P2,2 v ¬ P3,1))^((¬ P2,2 v ¬ P3,1)=>¬ B2,1))

((¬B1,2 v (P1,3 v P2,2))^( ¬ (P1,3 v P2,2) v B1,2)) ^ ((¬¬ B2,1 v (¬ P2,2 v ¬ P3,1))^( ¬ (¬ P2,2 v ¬ P3,1) v ¬ B2,1))

((¬B1,2 v (P1,3 v P2,2))^(( ¬P1,3 ^ ¬P2,2) v B1,2)) ^ ((B2,1 v (¬ P2,2 v ¬ P3,1))^((P2,2 ^ P3,1) v ¬ B2,1))

(F v T)^(F v T)^(T v F)^(T v F)

T ^ T ^ T ^ T = True

You can plug these clauses into the algorithm, and this proves the entailment is True.

* *KB* ╞ ¬ P2,2 (There is no Pit at location [2,2]) [0.5 pt]

R1: B1,2

R2: ¬ B2,1

R3: B1,2 ⬄ (P1,3 v P2,2)

R4: ¬ B2,1 ⬄ (¬ P2,2 v ¬ P3,1)

(B1,2 ⬄ (P1,3 v P2,2)) ^ (¬ B2,1 ⬄ (¬ P2,2 v ¬ P3,1))

((B1,2 => (P1,3 v P2,2))^((P1,3 v P2,2)=> B1,2))^ ((¬ B2,1 => (¬ P2,2 v ¬ P3,1))^((¬ P2,2 v ¬ P3,1)=>¬ B2,1))

((¬B1,2 v (P1,3 v P2,2))^( ¬ (P1,3 v P2,2) v B1,2)) ^ ((¬¬ B2,1 v (¬ P2,2 v ¬ P3,1))^( ¬ (¬ P2,2 v ¬ P3,1) v ¬ B2,1))

((¬B1,2 v (P1,3 v P2,2))^(( ¬P1,3 ^ ¬P2,2) v B1,2)) ^ ((B2,1 v (¬ P2,2 v ¬ P3,1))^((P2,2 ^ P3,1) v ¬ B2,1))

(F v T)^(F v T)^(T v F)^(T v F)

T ^ T ^ T ^ T = True

You can plug these clauses into the algorithm, and this proves the entailment is True.

* *KB* ╞ ¬ W2,2 (There is no Wumpus at location [2,2]) [0.5 pt]

R1: ¬ S1,2

R2: ¬ S1,2 ⬄ (¬ W1,3 ∧ ¬ W2,2)

KB = {R1, R2}

¬ S1,2 => (¬ W1,3 ∧ ¬ W2,2) ∧ (¬ W1,3 ∧ ¬ W2,2) => ¬ S1,2

¬ ¬ S1,2 ∨ (¬ W1,3 ∧ ¬ W2,2) ∧ ¬ (¬ W1,3 ∧ ¬ W2,2) ∨ ¬ S1,2

(S1,2 ∨ (¬ W1,3 ∧ ¬ W2,2)) ∧ ((W1,3 ∨ W2,2) ∨ ¬ S1,2)

(T OR (F AND F)) AND ((T OR T) OR F)

(T OR F) AND (T OR F)

T AND T = TRUE

You can plug these clauses into the algorithm, and this proves the entailment is True.

**Question 4 [2 pts]** The following table lists defined predicates.

|  |  |
| --- | --- |
| A(x) | x is an apple |
| B(x) | x is blue |
| T(x) | x is tasty |
| G(x) | x is green |
| R(x) | x is red |
| P(x) | x is people |
| F(x) | x is fruit |
| L(x,y) | x like y |

Using first order logic to express following sentences

* Apples are fruit [0.25 pt]
* Some apples are red [0.25 pt]
* No apple is blue [0.25 pt]
* Green apples are tasty [0.25 pt]
* Some people do not like apple [0.25 pt]
* Not all apples are green [0.25 pt]
* Some people like fruit, except apples [0.25 pt]
* No fruit is liked by every people [0.25]

****

**Question 5 [2 pts]** Figure 2shows the Wumpus world game, where the agent starts from location [1,1], and does not sense breeze or stench. After that, the agent moves to location [1,2] and sense a Breeze (B). Then the agent moves back to [1,1], and further moves to location [2,1] and senses a Stench (S).

|  |  |
| --- | --- |
| Wa,b | Wumpus is located at located [a,b] |
| Sx,y | A stench is sensed at location [x,y] |
| Pa,b | A pit is located at located [a,b] |
| Bx,y | A breeze is sensed at location [x,y] |
| Adjacent([a,b], [c,d]) | Cell located at [c,d] is adjacent to the cell located at [a,b] |

Using first order logic (FOL) to express following sentences and/or derive respective conclusion.

* Write an FOL sentence to express that “all cells next to the cell of Wumpus will sense a stench” [0.25 pt].

∀a,b,c,d: Wa,b ∧ Adjacent([a,b], [c,d] ⇒ Sx,y

* Write an FOL sentence to express that “all cells next to the cell of a pit will sense a breeze” [0.25 pt].

∀a,b,c,d: Pa,b ∧ Adjacent([a,b], [c,d]) ⇒ Bx,y

* Write an FOL sentence to express that “if there is no breeze sensed at a cell, there is no pit in adjacent cells” [0.25 pt].

∀a,b,c,d: ¬ Ba,b ∧ Adjacent([a,b], [c,d]) ⇒ ¬ Pc,d

* Write an FOL sentence to express that “if there is no stench sensed at a cell, there is no Wumpus in adjacent cells” [0.25 pt].

∀a,b,c,d: ¬ Sa,b ∧ Adjacent([a,b], [c,d]) ⇒ ¬ Wc,d

* Based on the above settings, derive a conclusion that there is no Wumpus at [1,3] [0.5 pt]
  + ¬S1,2 ∧ Adjacent([1, 2], [1, 3])
  + Using contraposition we can say:
  + Wa,b ∧ Adjacent([a, b], [x, y]) ⇒ Sx,y  ≡ ¬Sx,y ⇒ ¬(Wa,b ∧ Adjacent([a, b], [x, y]))
  + ¬S1,2 ⇒ ¬(W1,3 ∧ Adjacent([1, 3], [1, 2]))
  + Now we can convert to CNF format
  + S1,2 ∨ ¬W1,3 ∨ ¬Adjacent([1, 3], [1, 2]))
  + S1,2 ∨ ¬W1,3 ∨ ¬Adjacent([1, 3], [1, 2])) ; ¬S1,2
  + ¬W1,3 ∨ ¬Adjacent([1, 3], [1, 2])) ; Adjacent([1, 3], [1, 2])
  + ¬W1,3
  + Since S1,2 ∨ ¬W1,3 ∨ ¬Adjacent([1, 3], [1, 2])) is true and S1,2 and ¬Adjacent([1, 3], [1, 2]) are false, ¬W1,3 must be true meaning there is no Wumpus at [1,3].
* Based on the above settings, derive a conclusion that there is no pit at [2,2] [0.5 pt]
  + ¬B2,1 ∧ Adjacent([2, 1], [2, 2])
  + Using contraposition we can say:
  + Pa,b ∧ Adjacent([a, b], [x, y]) ⇒ Bx,y  ≡ ¬Bx,y ⇒ ¬(Pa,b ∧ Adjacent([a, b], [x, y]))
  + ¬B2,1 ⇒ ¬(P2,2 ∧ Adjacent([2, 2], [2, 1]))
  + Now we can convert to CNF format
  + B2,1 ∨ ¬P2,2 ∨ ¬Adjacent([2, 2], [2, 1]))
  + B2,1 ∨ ¬P2,2 ∨ ¬Adjacent([2, 2], [2, 1])) ; ¬B2,1
  + ¬P2,2 ∨ ¬Adjacent([2, 2], [2, 1])) ; Adjacent([2, 2], [2, 1])
  + ¬P2,2
  + Since B2,1 ∨ ¬P2,2 ∨ ¬Adjacent([2, 2], [2, 1])) is true and B2,1 and ¬Adjacent([2, 2], [2, 1]) are false, ¬P2,2 must be true meaning there is no Pit at [2,2].

**Question 6 [2 pts]** The following sentences describe interesting behaviors of a Hoffer club members.

* + Tony, Shi-Kuo and Ellen belong to the Hoofers Club.
  + Every member of the Hoofers Club is either a skier or a mountain climber or both.
  + No mountain climber likes rain, and all skiers like snow.
  + Ellen dislikes whatever Tony likes and likes whatever Tony dislikes.
  + Tony likes rain and snow.

Query: Does Hoffer club has a member who is a mountain climber but not a skier?

1. Define predicates and relations of the Hoffer using first order logic [0.5 pt]

S(x):skier M(x): mountain climber

L(x,y): x likes y

1. Using first order logic to express each sentence (including query) [0.5 pt]

¬ ∃x M(x) ∧ ¬ S(x)

1. Converting each sentence to clause format [0.5 pt]
2. Tony, Shi-Kuo and Ellen belong to the Hoofers Club.

Hoofers\_club(Tony, Ski-Kuo, Ellen)

1. Every member of the Hoofers Club is either a skier or a mountain climber or both.

∀xS(x)∨M(x); ¬ S(x1) ∨ M(x1)

1. No mountain climber likes rain, and all skiers like snow.

¬ ∃x M(x) ∧L(x, Rain); ¬M(x2) ∨ ¬ L(x2, Rain)

∀S(x) ⇒L(x, Snow); ¬S(x) ∨ L(x3, snow)

1. Ellen dislikes whatever Tony likes and likes whatever Tony dislikes.

∀y L(Ellen,y) ⇔ ¬ L (Tony, y)

¬L(Ellen, x4) ∨ ¬ (Tony, x4); L(Ellen, x5) ∨ L(Tony, x5)

1. Tony likes rain and snow

L(Tony, Rain); L(Tony, Rain)

L(Tony, Snow); L (Tony, Snow)

1. Using Unification to answer the query [0.5]

Ellen is a mountain biker but not a skier.

Negation Query:

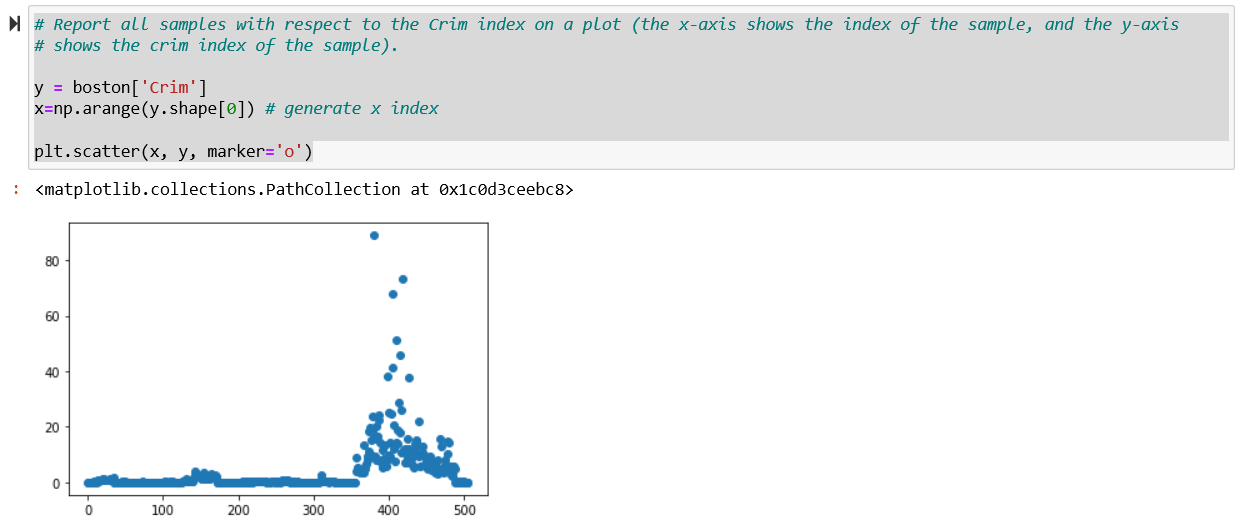
¬ ∃x M(x) ∧ ¬ S(x)

¬M(x6) ∨ S (x6)?

|  |  |  |  |
| --- | --- | --- | --- |
| Clause 1 | Clause 2 | Resolvent | MGU |
| ¬M(x6) ∨ S (x6) | S(x1) ∨ M(x1) | S(x1) | {X6/x1} |
| S(x1) | ¬S(x3) ∨ Like (x3, Snow) | Like (x1, Snow) | {x3/x1} |
| Like(x1, snow) | ¬ L (Ellen, x4) ∨ ¬L(Tony, x4) | ¬ L(Tony, Snow) | {x4/snow, x1/Ellen} |
| ¬L(Tony, Snow) | L(Tony, snow) | False | {} |

**For all programming tasks, please submit the Notebook as an html notebook file or a pdf notebook file for grading (your submission must show scripts/code and the running results of the script). The code running results must reflect/answer the task requirements.**

For each subtask, please use task description (requirement) as comments, and report your coding and results in following format:



**Question 7 [1 pt]** The Minmax Decision Playing Tictactoe [Notebook, html] and Alphabeta Pruning Playing Tictactoe [Notebook, html] posted on Canvas show two programs playing Tic-Tac-Toe game against a computer agent. Use Notebook as the skeleton code, validate and compare following settings and results.

* 1. Play Tic Tac Toe Minmax for 10 times. Report average computer thinking time (which is the time required for Minmax algorithm to find solutions) [0.25 pt]. Explain why it is hard to win the game against computer [0.25 pt]
  2. Play Tic Tac Toe Alphabeta for five time. Report average computer thinking time (which is the time required for Alpha Beta Pruning to find solutions) [0.25 pt]. Compare time required for Minmax and Alphabeta pruning, explain why Alphabeta pruning is quicker [0.25 pt].

I included a table below that displays the different attempts for each version of the game and the average computer thinking times.

It is hard to win the game against the computer because the computer is evaluating every possible state after each move and choosing the move that is most optimal for it to win.

After comparing the computing time for each algorithm, it is evident that Alphabeta is much faster. This is because the computer is “pruning away” states that it does not need to evaluate. It is only focusing on those necessary to win.

A piece of paper with writing

Description automatically generated with medium confidence