Artificial Intelligence: Search, CSPs and Logic Bayesian Networks, Probabilistic Inference

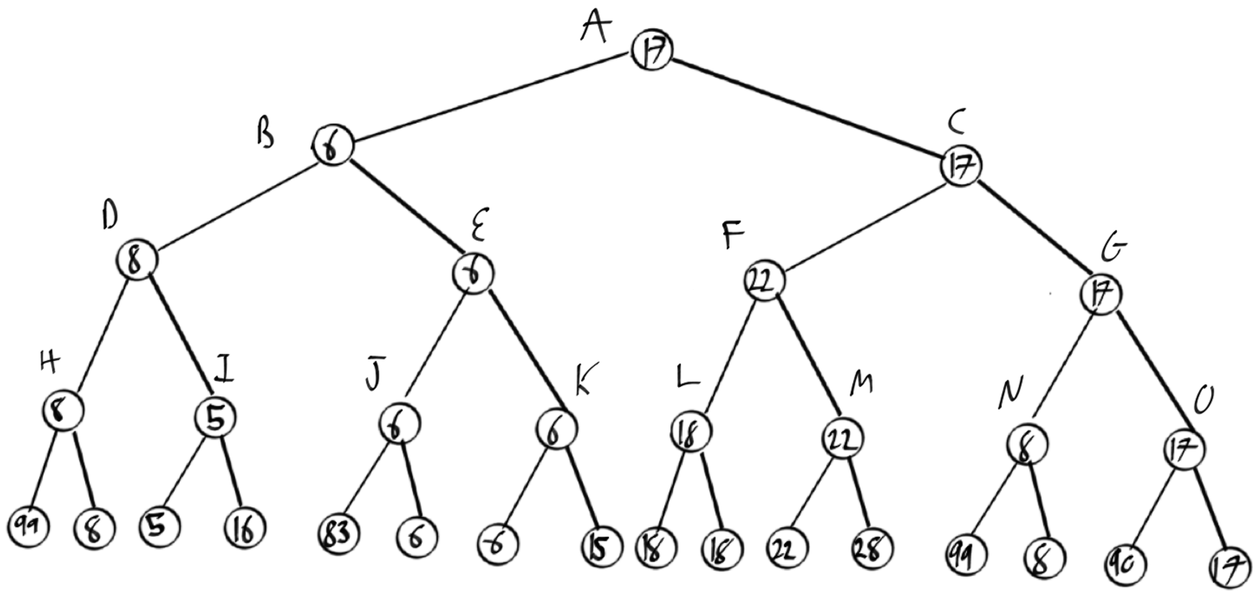
Nathan Morgenstern, Seo Bo Shim

December 11, 2017

Contents

Question 1

part a.



H: 8, I: 5, J: 6, K: 6, L: 18, M: 22, N: 8, O: 17

D: 8, E: 6, F: 22, G: 17

B: 6, C: 17

A: 17

Figure 1: Question 1: Part a

part b.

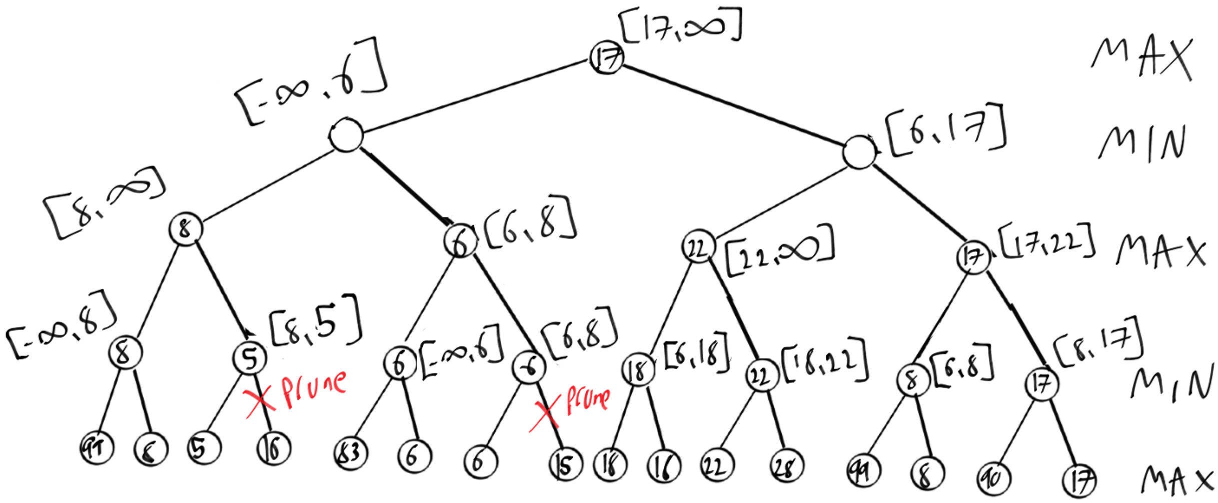


Figure 2: Question 1: Part b

part c.

The diagrams for parts a. and b. both show that the exhaustive minimax algorithm, and the minimax algorithm with alpha-beta pruning reach the same Max value for the root node: 17. This is not a coincidence and should happen in the general case, as alpha-beta pruning is simply meant to disregard the nodes that will not be worth considering. However, the underlying assumption is the opponent is playing optimally. If the opponent is playing suboptimal, the results of the minimax and alpha-beta pruning are not necessarily the best.

part d.

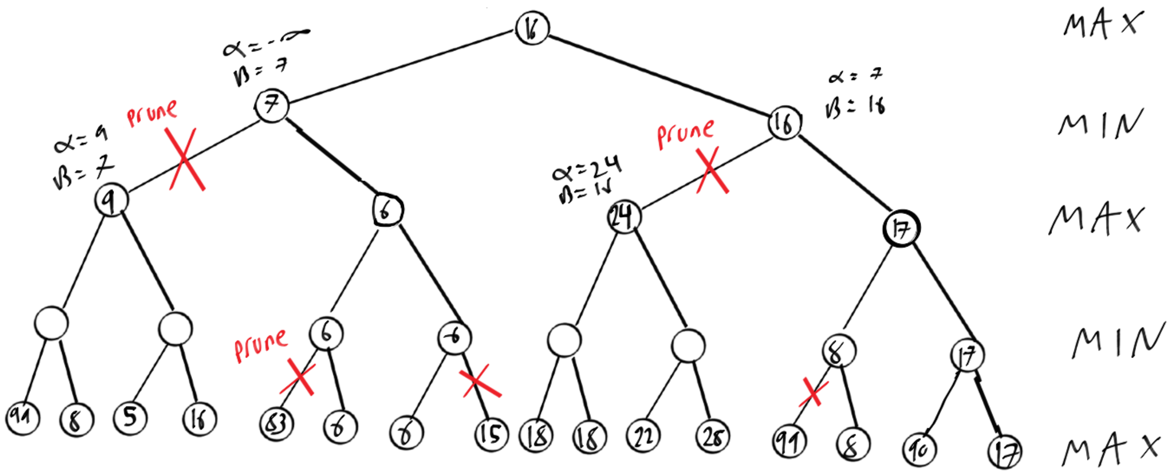


Figure 3: Question 1: Part d

First the minimax algorithm is run at nodes B,C,D,E backs up the value 7 to node b, and the value 16 to node c.

When the alpha beta pruning algorithm is ran

depth 1: alpha = - inf, beta = 7 depth 2: alpha = 9, beta = 7 [Prune Sub tree]

The left child of node b is pruned, and alpha beta continues down to depth 4.

depth 2: alpha = 7, beta = 7 depth 3: alpha = 7, beta = 7 depth 4: alpha = 83, beta = 6 [Prune] depth 4: alpha = 6, beta = 7

This will back up 6 to node J, the alpha beta pruning will now go to the right child of node b.

depth 3: alpha = 7, beta = 7 depth 4: alpha = 6, beta = 7 This will back up 6 to node K

The value 15 is pruned since its greater than the value backed up at its parent.

The left traversal of the alpha-beta pruning is done, now it starts doing right traversal.

Nodes Examined: Nodes Not Examined: 14

Similarly, to the left side alpha beta pruning cuts off node e

part e.

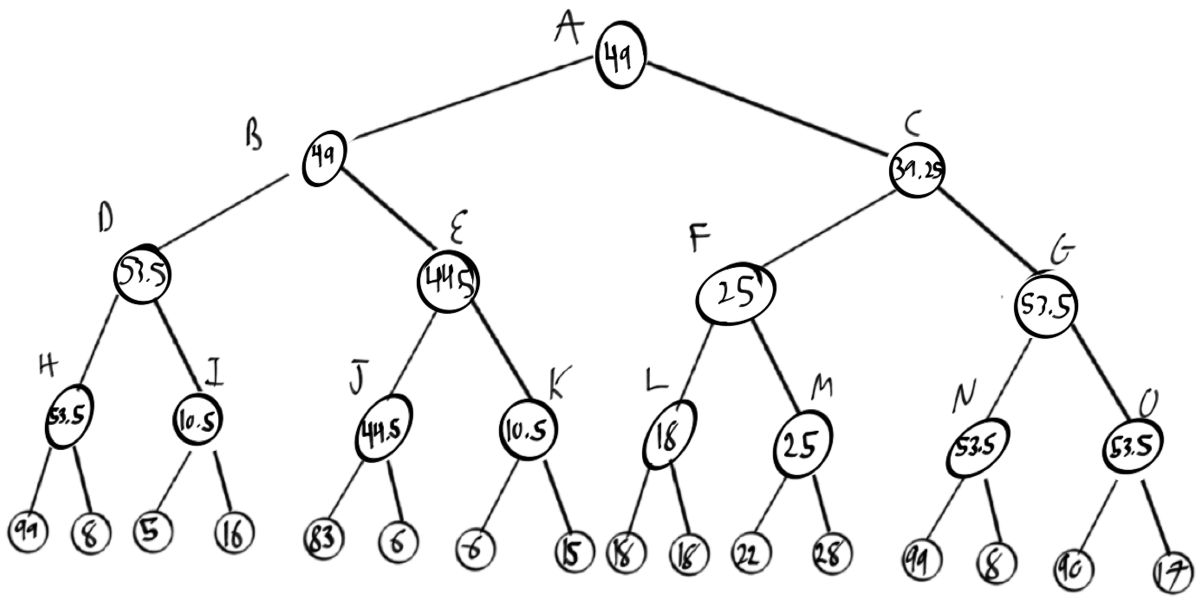


Figure 4: Question 1: Part e

Applying the minimax algorithm using expected values...

Expected Values of Nodes H,I,J,K,L,N,N,O...

Node H: 0.5 * (99 + 8) = 53.5	Node I: 0.5 * (5 + 16) = 10.5
Node J: 0.5 * (83 + 6) = 44.5	Node K: 0.5 * (6 + 15) = 10.5
Node L: 0.5 * (18 + 18) = 18	Node M: 0.5 * (22 + 28) = 25
Node N: 0.5 * (99 + 8) = 53.5	Node O: 0.5 * (90 + 17) = 53.5

Nodes D, E, F, G choose the Max

Node D: 53.5 Node E: 44.5

Node F: 25 Node G: 53.5

Expected values of Nodes B, C

Node B: 0.5 * (53.5 + 44.5) = 49 Node C: 0.5 * (25 + 53.5) = 39.25

Node A: 49

The root node value has a value of 49, which is the expected value for the scenario of the opponent choosing a random node.

We can apply alpha beta pruning algorithm, although first the minimax algorithm will need to find the expected values at each of the nodes. Using the algorithm after this step begs the question of why should we use the alpha beta pruning algorithm in the first place. If we use all of the computational resources to calculate those expected values then using the alpha beta pruning algorithm is redundant and doesn't provide any utility.

This shows that alpha beta pruning may not be the most efficient algorithm in the case of a suboptimal opponent, as there are better choices to choose from.

NOTCOMPLETE

NOT DONE: CAN WE APPLY ALPHA BETA PRUNING IN THIS CASE

Question 2

part a.

The set of variables is the combination of the row and column of the grid.

X = [1,1], [1,2], ... , [i,j-1], [i, j]

The domain is the possible values that each variable can hold: 1 to 9.

D = 1, 2, 3, 4, 5, 6, 7, 8, 9

The constraint is that each row, column, and neighboring box must hold unique values in each of the variables.

i, j = fixed variable I, J = random variable C = (X(1,1), X(1,2), X(1,3), X(1,4), X(1,5), X(1,6), X(1,7), X(1,8), X(1,9)), X(1,1) != X(1,2) != X(1,3) != X(1,4) != X(1,5) != X(1,6) != X(1,7) != X(1,8) != X(1,9)

part b.

Start state: initial arrangement of values 1-9 on the Sudoku board in random locations and such that the constraints are not broken. No additional values on the board

Successor function: add

Goal test: all locations on the board has a value between 1 - 9.

Path cost: If the action leads to a state where one of the constraints are broken, then the path cost should be infinite. Path cost should depend on how many

part c.

Easy problems can be solved by choosing actions in the coordinates that are most constrained, and continue choosing actions this way until the board is filled. The clues are given in such a way that every action the user takes can be deterministic and the user does not have to make any choices.

Difficult problems most likely will require backtracking, because the solution can require actions that aren't intuitive and may require the user to make approximations. The puzzle may reach an equilibrium point where the next action is not obvious, as multiple actions can be taken and still be valid until a much later state.

part d.

NOT DONE

Question 3

part a.

S = superman is defeated

O = facing opponent alone

OK = opponent is carrying kryptonite

For superman to be defeated, it has to be that he is facing an opponent alone and his opponent is carrying kryptonite.

S $\iff (O \wedge OK)$

$(S \rightarrow (O \wedge OK)) \wedge ((O \wedge OK) \rightarrow S)$(applied biconditional elimination)

$(S \rightarrow (\neg O \vee \neg OK)) \wedge ((O \wedge OK) \rightarrow S)$(DeMorgans)

$(\neg S \vee (\neg O \vee \neg OK)) \wedge (\neg(O \wedge OK) \vee S)$(applied implication elimination)

$(\neg S \vee \neg O \vee \neg OK) \wedge (\neg O \vee \neg OK \vee S)$

part b.

Turning the above statements into 3-cnf form, our KB can be represented as follows...

(¬S ∨¬O ∨¬OK) ∧(¬O ∨¬OK ∨S) ∧(¬AK ∨¬BC ∨¬BA) ∧(¬BC ∨WU ∨WS)

part c.

To show that batman cannot defeat superman, we can show that (KB $\wedge\alpha$) is unsatisfiable(The knowledge base entails that superman can be defeated), which means every assignment of variables does not satisfy the sentence. This is known as proof by contradiction α = S, or the value of superman being defeated by batman.

Taking the disjunction of KB and $\neg\alpha$...

$(\neg S \vee \neg O \vee \neg OK) \wedge (\neg O \vee \neg OK \vee S) \wedge (\neg AK \vee \neg BC \vee \neg BA) \wedge (\neg BC \vee WU \vee WS)$

$\wedge S$

Question 4

part a.

$$(\neg P_1 \vee ... \vee \neg P_m \vee Q)$$

Using DeMorgan's theorem, this term is equivalent to:

$$\neg(P_1 \wedge ... \wedge P_m \wedge \neg Q)$$

Using the Associative property

$$\neg((P_1 \wedge ... \wedge P_m) \wedge \neg Q)$$

Using the conditional equivalence: $\neg(P \rightarrow Q) = (P \wedge \neg Q)$

$$\neg(\neg(P_1 \wedge ... \wedge P_m \rightarrow Q))$$

Finally,

$$(P_1 \wedge ... \wedge P_m \rightarrow Q)$$

part b.

We can repeat the above proof by replacing Q with an expression $(Q_1 \vee ... \vee Q_n)$. A literal can be replaced with an expression.

$$(\neg P_1 \vee ... \vee \neg P_m \vee \neg(Q_1 \vee ... \vee Q_n))$$

Using DeMorgan's theorem, this term is equivalent to:

$$\neg(P_1 \wedge ... \wedge P_m \wedge \neg(Q_1 \vee ... \vee Q_n))$$

Using the Associative property

$$\neg((P_1 \wedge ... \wedge P_m) \wedge \neg(Q_1 \vee ... \vee Q_n))$$

Using the conditional equivalence: $\neg(P \rightarrow Q) = (P \wedge \neg Q)$

$$\neg(\neg((P_1 \wedge ... \wedge P_m) \rightarrow (Q_1 \vee ... \vee Q_n)))$$

Finally,

$$(P_1 \wedge ... \wedge P_m) \rightarrow (Q_1 \vee ... \vee Q_n)$$

part c.

To complete the full resolution and find the resolvent, start with the two clauses:

$$(l_1 \vee ... \vee l_k \vee m_1 \vee ... \vee m_n)$$

We can split this expression up using the associative property:

$$(l_i \vee m_j) \vee (l_1 \vee ... \vee l_{i-1} \vee l_{i+1} \vee ... \vee l_k) \vee (m_1 \vee ... \vee m_{j-1} \vee m_{j+1} \vee ... \vee m_n)$$

Using the Conditional equivalence property, similar to what we derived in part A, this term is equivalent to:

$$\neg(l_i \vee m_j) \rightarrow (l_1 \vee ... \vee l_{i-1} \vee l_{i+1} \vee ... \vee l_k) \vee (m_1 \vee ... \vee m_{j-1} \vee m_{j+1} \vee ... \vee m_n)$$

Because l_i and m_j are complimentary: $(l_i \vee m_j) = True$

$$True \rightarrow (l_1 \vee ... \vee l_{i-1} \vee l_{i+1} \vee ... \vee l_k) \vee (m_1 \vee ... \vee m_{j-1} \vee m_{j+1} \vee ... \vee m_n)$$

Therefore it is inferred that

$$(l_1 \vee ... \vee l_{i-1} \vee l_{i+1} \vee ... \vee l_k \vee m_1 \vee ... \vee m_{j-1} \vee m_{j+1} \vee ... \vee m_n)$$

Question 5

part a.

P(A,B,C,D,E) = P(A) * P(D | A, B) * P(B) * P(E | B, C) * P(C)

P(A,B,C,D,E) = 0.2 * 0.1 * 0.5 * 0.3 * 0.8

P(A,B,C,D,E) = 0.0024

part b.

P(¬A,¬B,¬C,¬D,¬E) = P(¬A) * P(¬D | ¬A, ¬B) * P(¬B) * P(¬E | ¬B, ¬C) * P(¬C)

P(¬A,¬B,¬C,¬D,¬E) = 0.8 * 0.9 * 0.5 * 0.7 * 0.2

P(¬A,¬B,¬C,¬D,¬E) = 0.0504

part c.

$$P(\neg A|B,C,D,E) = \frac{P(\neg A,B,C,D,E)}{\sum_A P(B,C,D,E)}$$

$$P(\neg A|B,C,D,E) = \alpha P(\neg A,B,C,D,E)$$

$$P(\neg A|B,C,D,E) = \alpha P(\neg A) * P(B) * P(C) * P(D|\neg A,B) * P(E|B,C)$$

$$P(\neg A|B,C,D,E) = \alpha 0.8 * 0.5 * 0.8 * 0.6 * 0.3$$

$$P(\neg A|B,C,D,E) = \alpha 0.0576$$

Solving for α...

$$\alpha = \frac{1}{P(\neg A, B, C, D, E) + P(A, B, C, D, E)}$$

$$\alpha = \frac{1}{0.0576 + 0.0024}$$

$$\alpha = \frac{1}{0.06}$$

Resulting...

$$\text{P}(\neg\text{A} \mid \text{B,C,D,E}) = \frac{0.0576}{0.06}$$

$$P(\neg A|B, C, D, E) = 0.96$$

Question 6

part a.

B = burglary E = earthquake

A = alarm

J = John calls

M = Mary calls

$$P(B) = 0.001$$

$$P(E) = 0.002$$

$$P(B|J, M) = \alpha * P(B) * \sum_e P(e) * \sum_a P(a|B, e) * P(J|a) * P(M|a)$$

Compute the sum over A, which results in summing two terms as alarm is either true or false.

$$\sum_a P(a|B, e) * P(J|a) * P(M|a) = P(a|B, E) * P(j|a) * P(m|a) + P(\neg a|B, E) * P(j|\neg a) * P(m|\neg a)$$

$$P(b, e) = .95 * .9 * .7 + .05 * .05 * .01 = .598525$$

For the next summation we will also need the case where the earthquake event does not occur:

$$P(b, \neg e) = .94 * .9 * .7 + .06 * .05 * .01 = .592230$$

Then the rest of the calculations:

$$P(\neg b, e) = .29 * .9 * .7 + .71 * .05 * .01 = .183055$$

$$P(\neg b, \neg e) = .001 * .9 * .7 + .999 * .05 * .01 = .001130$$

Including these calculations into the matrix form:

$$f_6(B,E) = \begin{Bmatrix} P(b,e) & P(b,\neg e) \\ P(\neg b,e) & P(\neg b,\neg e) \end{Bmatrix}$$

$$f_6(B,E) = \begin{Bmatrix} .598525 & .592230 \\ .183055 & .001130 \end{Bmatrix}$$

Then sum over E, whether the event of an earthquake is true or false.

$$P(B|J,M) = \alpha * P(B) * \sum P(e) * f_6(B,e)$$

$$\sum_e P(e) * P(B,e) = P(e) * f_6(B,e) + P(\neg e) * f_6(B,\neg e)$$

$$\begin{Bmatrix} P(b,e) \\ P(\neg b,e) \end{Bmatrix} = .002 * \begin{Bmatrix} .598525 \\ .183055 \end{Bmatrix} + 0.998 * \begin{Bmatrix} .592230 \\ .001130 \end{Bmatrix} = \begin{Bmatrix} .590466 \\ .001494 \end{Bmatrix}$$

$$f_7(B) = \begin{Bmatrix} .590466 \\ .001494 \end{Bmatrix}$$

$$f_7(b) = .590466$$

$$f_7(\neg b) = .001494$$

Finally

$$P(B|J,M) = \alpha * P(B) * f_7(B))$$

$$P(b|J,M) = \alpha * .001 * .590466$$

$$P(\neg b|J,M) = \alpha * .999 * .001494$$

We sum up the two probabilities to find alpha

$$P(b|J,M) + P(\neg b|J,M) = 0.0020830$$

$$\alpha = 1/0.0020830 = 480.$$

$$\boldsymbol{P(b|J,M) = 480 * .001 * .590466 = 0.28347}$$

$$\boldsymbol{P(\neg b|J,M) = 480 * .999 * .001494 = 0.71653}$$

part b.

Operations in Variable Elimination

4*(4 mult 1 add) + 2*(2 mult 1 add) + 2*(2 mult):

+ 1 division and 1 addition for finding the α value:

7 additions

24 multiplications

1 division

32 total operations

Operations in tree enumeration algorithm

For each probability in the final term $P(B|j,m)$ we have three additions, 14 multiplications, and 1 division. The division is from the normalization factor.

There are 8 different cases in $P(B|j,m)$, so in total:

24 additions

112 multiplications

8 divisions

144 total operations

Variable elimination uses much less operations than the tree enumeration algorithm. It uses 22 % of the operations used by the tree enumeration algorithm.

part c.

where n is the number of boolean variables

Using variable elimination: The time and space complexity are dominated by the highest factor constructed. The largest factor is comprised of the number of a children a node in a Bayesian Network may have. If there are n boolean variables, then it is possible for n-1 boolean variables to all have the same parent node. So the largest factor is O(n-1), but this is in fact the worst case scenario. On average it will be less than n as the Bayesian Network will not always be in this configuration.

The space complexity is $0(2^{(n-1)})$ The time complexity is $O(2^{(n-1)})$

the worst case complexity is worse than the tree enumeration, however on average the time complexity is better. There is higher potential space complexity to save states so they do not need to be calculated again (reduce time complexity).

Using enumeration: The space complexity is $O(n)$ as the only requirement is to store each boolean variable The time complexity is $O(2^(n))$, as each combination of the boolean variable must be calculated.

Question 7

part a.

Prove that $P(X|MB(X)) = \alpha P(X|U_1, ..., U_m) \prod_{Y_i} \mathrm{P}(\mathrm{Y}_i|Z_1...)$

The markov blanket covers the parents of X as well as its children and children's parents.

$MB(X) = Parents(X), Y_i, Z_{1i}, ..., Z_{nj}$ Also, note $Parents(X) = U_1, ..., U_m$

$MB(X) = (U_1, ..., U_m), Y_i, (Z_{1i}, ..., Z_{nj})$

So, we have...

$P(X|MB(X)) = P(X|(U_1, ..., U_m), Y, Z_{1i}, ..., Z_{nj}) = \alpha P(X, (U_1, ..., U_m), Y_i, (Z_{1i}, ..., Z_{nj}))$

$P(X|MB(X)) = \alpha P(X|U_1, ..., U_m) P(Y_i|(Z_{1i}, ..., Z_{nj}))$

$P(X|MB(X)) = \alpha P(X|U_1, ..., U_m) P(Y_i|(Z_{1i}) * P(Y_i|...) * P(Y_i|Z_{nj})$

$P(X|MB(X)) = \alpha P(X|U_1, ..., U_m) \prod_{Y_i} \mathrm{P}(\mathrm{Y}_i|Z_1...)$ Q.E.D.

part b.

The Markov Chain Monte Carlo algorithm will begin with an initial state [sprinkler, wetgrass, cloudy, rain] The evidence variables: sprinkler, and wetgrass are initialized to their observed values while the nonobserved values: cloudy, rain are randomized. Initial State: <cloudy, sprinkler, rain, wetgrass> = <true, true, false, true>

After initializing, the non evidence variables are sampled in an arbitrary order for N iterations, while the evidence variables are fixed. For example, the non evidence variable cloudy is sampled, and a result of cloudy = false is returned giving: <false, true, false, true>. This process will finish until all of the iterations are complete.

In total, there are 4 states since the evidence variables sprinkler and wetgrass will not be changing. There are 2 non evidence variables cloudy and rain $2^2 = 4$states

R_C = P(Rain = true | sprinkler = true, wetgrass = true, cloudy = true)

R_¬C = P(Rain = true | sprinkler = true, wetgrass = true, cloudy = false)

C_R = P(Cloudy = true | sprinkler = true, wetgrass = true, rain = true)

C_¬R = P(Cloudy = true | sprinkler = true, wetgrass = true, rain = false)

$$\begin{array}{c c}
\begin{array}{ccccc} R & \neg R & C & \neg C & \end{array} & \\
\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} & \begin{array}{c} R \\ \neg R \\ C \\ \neg C \end{array}
\end{array}$$

part c.

NOT DONE

Question 8

part a.

Cost of good quality car:

$$C(q^{+}(c_1)) = \$4000 - (3000) = \$1000. \tag{1}$$

Cost of bad quality car:

$$C(q^{-}(c_1)) = \$4000 - (3000 + 1400) = -\$400. \tag{2}$$

Assuming that repairs can be made without taking it to the mechanic - perhaps the $1400 is a separate cost than the $100 needed to check the quality of the car.

Probability of good quality car:

$$P(q^{+}(c_1)) = 0.7 \tag{3}$$

Probability of bad quality car:

$$P(q^{-}(c_1)) = 0.3 \tag{4}$$

Expected net gain:

$$E(c_1) = C(q^{+}(c_1)) * P(q^{+}(c_1)) + C(q^{-}(c_1)) * P(q^{-}(c_1))$$

$$\mathbf{E}(\mathbf{c_1}) = \$\mathbf{580} \tag{5}$$

part b.

$$P(Pass|q^{+}) = 0.8 \tag{6}$$

$$P(Pass|q^{-}) = 0.35 \tag{7}$$

Using 6 and 7:

$$P(\neg Pass|q^{+}) = 1 - P(Pass|q^{+}) = 0.2 \tag{8}$$

$$P(\neg Pass|q^-) = 1 - P(Pass|q^-) = 0.65 \tag{9}$$

Using 3, 4, 6, 7:

$$P(Pass) = P(Pass|q^+) * P(q^+) + P(Pass|q^-) * P(q^-) = 0.665$$

$$\mathbf{P(Pass) = 0.665} \tag{10}$$

Using 10:

$$P(\neg Pass) = 1 - P(Pass) = 0.335$$

$$\mathbf{P(\neg Pass) = 0.335} \tag{11}$$

Using 6, 3, and 10:

$$\mathbf{P(q^+|Pass) = \frac{P(Pass|q^+) * P(q^+)}{P(Pass)} = 0.842} \tag{12}$$

Using 7, 4, and 10:

$$\mathbf{P(q^-|Pass) = \frac{P(Pass|q^-) * P(q^-)}{P(Pass)} = 0.158} \tag{13}$$

Using 8, 3, and 11:

$$\mathbf{P(q^+|\neg Pass) = \frac{P(\neg Pass|q^+) * P(q^+)}{P(\neg Pass)} = 0.418} \tag{14}$$

Using 9, 4, and 11:

$$\mathbf{P(q^-|\neg Pass) = \frac{P(\neg Pass|q^-) * P(q^-)}{P(\neg Pass)} = 0.582} \tag{15}$$

part c.

Paying for the test with the mechanic, the new costs are:

$$C'(q^+(c_1)) = C(q^+(c_1)) - \$100 = \$900$$

$$C'(q^-(c_1)) = C(q^-(c_1)) - \$100 = -\$500$$

Given a pass:

$$E(c_1|Pass) = C'(q^+(c_1)) * P(q^+(c_1)|Pass) + C'(q^-(c_1)) * P(q^-(c_1)|Pass)$$

$$\mathbf{E(c_1|Pass) = \$678.8}$$

Given a failure:

$$E(c_1|\neg Pass) = C'(q^+(c_1)) * P(q^+(c_1)|\neg Pass) + C'(q^-(c_1)) * P(q^-(c_1)|\neg Pass)$$

$$\mathbf{E(c_1|\neg Pass) = \$85.2}$$

Regardless of a pass or a failure, the best decision is the sell the car as there will be a net gain.

part d.

Without the mechanic's test, the expected gain from selling the car will be $\mathbf{E(c_1) = \$580}$

With the test, the expected gain is $\mathbf{\$678.80}$. The value of the optimal information is the difference between the expected gain with the information and the expected gain without the information. The optimal information value is $\mathbf{\$98.80}$. I should take C1 to the mechanic.

Question 9

part a.

NOT DONE

part b.

NOT DONE

part c.

NOT DONE

Question 10

NOT DONE