Homework 4

Komal Niraula (kn2505)

Q1.

Answer:

Let D1, D2 and D3 be the three doors. D1 is the door being chosen, D3 is the door opened by the host which has nothing inside and D2 is the other unopened door.

- C1: The car is behind the initially chosen door D1
- C2: The car is behind the other unopened door D2
- H: The host opens a door which has nothing inside

P(C1 | H): Probability that the car is behind the initially chosen door after the host opens a door which has nothing inside

P(C2 | H): Probability that the car is behind the other unopened door after the host opens a door which has nothing inside

Here, the prior probabilities are:

- Probability that the car is behind the initially chosen door P(C1): 1/3
- Probability that the car is behind other unopened door P(C2): 2/3

If the car is behind the initially chosen door (C1):

P(H | C1) = 1/2 (The host can open either of the two remaining doors)

If the car is behind the other unopened door (C2):

P(H | C2) = 1 (The host has no choice and must open the door with a goat)

Using the law of probability:

P(H) = P(H | C1) × P(C1) + P(H | C2) × P(C2)

P(H) = (1/2 × 1/3) + (1 × 2/3) = 1/6 + 2/3 = 5/6

Now, applying Bayes theorem to compute P(C1 | H) and P(C2 | H):

$$P(\,C \mid H\,) = \frac{P(\,H \mid C\,) \cdot P(C)}{P(H)}$$

Probability of car being in the initially chosen door (C1):

P(C1 | H) = P(H | C1) × P(C1) / P(H)

= (1/2) × (1/3) / (5/6) = 1/3

Probability of car being in the other unopened door (C2):

P(C2 | H) = P(H | C2) × P(C2) / P(H)

$$= 1 \times (2/3) / (5/6) = 2/3$$

So, its better to change the door after the hosts opens the empty door. This is because the probability of car being in other unopened door is larger than the probability of car being in the initially chosen door $P(C2 \mid H) > P(C1 \mid H)$.

Q2.

Answer:
In the given network:

- x_1 : No parents (root node)
- x_2 : Parent is x_1
- x_3 : No parents (root node)
- x_4 : Parent are x_1 and x_3
- x_5 : Parents are x_2 and x_4

Using the chain rule for Bayesian networks:

$P(x_1, x_2, x_3, x_4, x_5) = P(x_1) \times P(x_2 \mid x_1) \times P(x_3) \times P(x_4 \mid x_1, x_3) \times P(x_5 \mid x_2, x_4)$

1. x_2 and x_4 are independent:

   False (They are conditionally independent given x1)

2. x_2 and x_4 are conditionally independent given x_1, x_3, and x_5:

   False

3. x_2 and x_4 are conditionally independent given x_1 and x_3:

   True (x_1, x_2 and x_4 are conditionally independent and not affected by adding x_3)

4. x_5 and x_3 are conditionally independent given x_4:

   False (x_3 influences x_4 which then influences x_5)

5. x_5 and x_3 are conditionally independent given x_1, x_2, and x_4:

   True (Given x_4, there is no direct path between x_5 and x_3)

6. x_1 and x_3 are conditionally independent given x_5:

   False (x_5 connects x_1 and x_3 indirectly through x_4)

7. x_1 and x_3 are conditionally independent given x_2:

   True (x_3 is connected to x_1 indirectly via x_4 and x_5)

8. x_2 and x_3 are independent:

   True (Paths connecting them are blocked)

9. x_2 and x_3 are conditionally independent given x_5:

   False (x_2 and x_3 share dependencies through x_4 even when x_5 is given)

10. x_2 and x_3 are conditionally independent given x_5 and x_4:

    False (x_2 depends on x_1)

Q3: Answer:

3)
⇒ At first draw undirected graphs

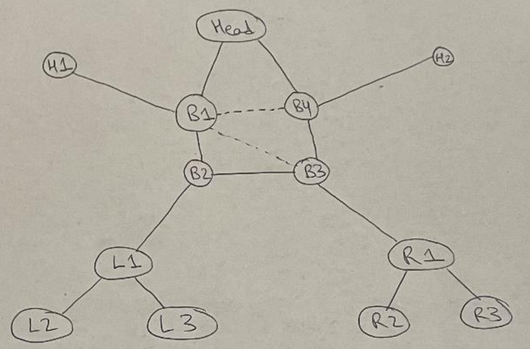

fig 1: Undirected graph with moralization & triangularization

In the above figure, ~~B1 & B4 are~~ an undirected edge is added between B1 & B4 as moralization. for triangularization an edge is added between B1 & B3

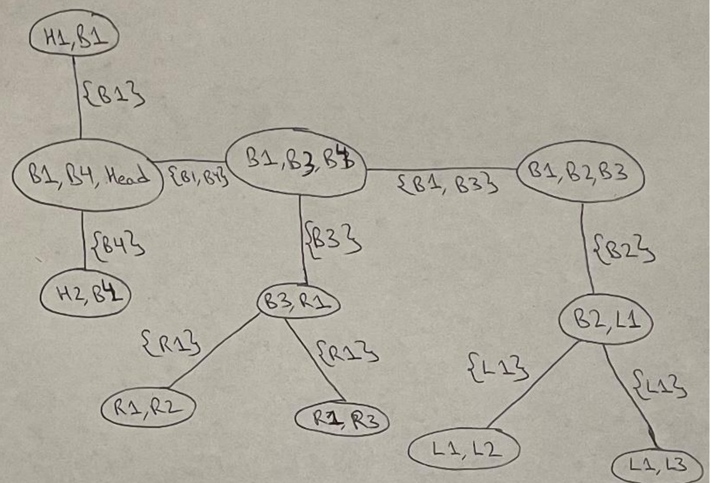

fig 2: Junction-tree graph

Q4.

Answer:

We follow the collect and distribute paradigm for a junction tree structure. The collect step (forward) involves sending messages sequentially from left to right, i.e., from x_1, x_2, to x_{n-1}, x_n. After reaching the root, the distributing step (backward) propagates messages in reverse. Once all messages are passed, the marginals are normalized to ensure they represent valid probability distributions.

Implemented JTA, following pairwise marginals were obtained:

```
Marginal 1:
    0.0405    0.4451
    0.3237    0.1908

Row sums for Marginal 1: [0.4855, 0.5145]
Column sums for Marginal 1: [0.3642, 0.6358]
Total sum for Marginal 1: 1.0000

Marginal 2:
    0.2601    0.1040
    0.0578    0.5780

Row sums for Marginal 2: [0.3642, 0.6358]
Column sums for Marginal 2: [0.3179, 0.6821]
Total sum for Marginal 2: 1.0000

Marginal 3:
    0.1192    0.1987
    0.6395    0.0426

Row sums for Marginal 3: [0.3179, 0.6821]
Column sums for Marginal 3: [0.7587, 0.2413]
Total sum for Marginal 3: 1.0000

Marginal 4:
    0.5690    0.1897
    0.0603    0.1810

Row sums for Marginal 4: [0.7587, 0.2413]
Column sums for Marginal 4: [0.6293, 0.3707]
Total sum for Marginal 4: 1.0000
```

Each marginal distribution for (x_i, x_{i+1}) is a 2×2 matrix summing to 1. The marginals are also consistent across overlapping variables. The implementation demonstrates how local interactions (potentials) influence the marginals.

Matlab code:

```
% Main function
n = 5; % Number of variables
psis = cell(n-1, 1); % Initialize potentials

% Define the given potentials
psis{1} = [0.1, 0.7; 0.8, 0.3];
psis{2} = [0.5, 0.1; 0.1, 0.5];
psis{3} = [0.1, 0.5; 0.5, 0.1];
psis{4} = [0.9, 0.3; 0.1, 0.3];

% Compute marginals using JTA
[marginals] = problem4_JTAMC(psis);

% Display results
disp('Pairwise Marginals and Sums:');
for i = 1:length(marginals)
    fprintf('Marginal %d:\n', i);
    disp(marginals{i});

    % Compute and display row sums
    rowSums = sum(marginals{i}, 2);
    fprintf('Row sums for Marginal %d: [%.4f, %.4f]\n', i, rowSums(1), rowSums(2));

    % Compute and display column sums
    colSums = sum(marginals{i}, 1);
    fprintf('Column sums for Marginal %d: [%.4f, %.4f]\n', i, colSums(1), colSums(2));

    % Compute and display total sum
    totalSum = sum(marginals{i}(:));
    fprintf('Total sum for Marginal %d: %.4f\n\n', i, totalSum);
end
```

```
function [marginals] = problem4_JTAMC(psis)
    % Number of variables
    n = length(psis) + 1;

    % Initialize forward and backward messages
    m_forward = cell(n, 1);
    m_backward = cell(n, 1);
    marginals = cell(n-1, 1);

    % Forward message passing
    m_forward{1} = ones(2, 1);
    for i = 1:n-1
        m_forward{i+1} = zeros(2, 1);
        for x_next = 1:2
            m_forward{i+1}(x_next) = sum(psis{i}(:, x_next) .* m_forward{i});
        end
    end

    % Backward message passing
    m_backward{n} = ones(2, 1);
    for i = n-1:-1:1
        m_backward{i} = zeros(2, 1);
        for x_prev = 1:2
            m_backward{i}(x_prev) = sum(psis{i}(x_prev, :)' .* m_backward{i+1});
        end
    end

    % Compute pairwise marginals
    for i = 1:n-1
        marginals{i} = zeros(2, 2);
        for x_i = 1:2
            for x_j = 1:2
                marginals{i}(x_i, x_j) = psis{i}(x_i, x_j) * m_forward{i}(x_i) * m_backward{i+1}(x_j);
            end
        end
        % Normalize the marginal
        marginals{i} = marginals{i} / sum(marginals{i}(:));
    end
end
```