Homework 2 - Introduction to Probabilistic Graphical Models

kipngeno koech - bkoech

March 13, 2025

1 Conditional Independence

1. (10 points) State True or False, and briefly justify your answer within 3 lines. The statements are either direct consequences of theorems in Koller and Friedman (2009, Ch. 3), or have a short proof. In the follows, P is a distribution and G is a BN structure.

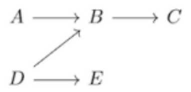

Figure 1: A Bayesian network.

$$\begin{tikzcd}
P \text{ factorizes over } \mathcal{G} \arrow[r, Rightarrow, "(1)"] & \mathcal{I}(\mathcal{G}) \subseteq \mathcal{I}(P) \arrow[r, Rightarrow, "(2)"] & \mathcal{I}_\ell(\mathcal{G}) \subseteq \mathcal{I}(P) \arrow[ll, Rightarrow, bend left, "(3)"]
\end{tikzcd}$$

Figure 2: Some relations in Bayesian networks.

Recall the definitions of local and global independences of G and independences of P.

$$I_l(G) = \{(X \perp \text{NonDescendants}_G(X) \mid \text{Parents}_G(X))\} \tag{1}$$

$$I(G) = \{(X \perp Y \mid Z) : \text{d-separated}_G(X, Y \mid Z)\} \tag{2}$$

$$I(P) = \{(X \perp Y \mid Z) : P(X, Y \mid Z) = P(X \mid Z)P(Y \mid Z)\} \tag{3}$$

$I_l(G)$ - is the local independence, where a node is independent of its non-descendants given its parents. $I(G)$ - is the global independence, where two nodes are independent given a set of nodes if they are d-separated. $I(P)$ - is the independence of a distribution, where two nodes are independent given a set of nodes if the joint distribution factorizes into the product of the marginals.

(a) In Figure 2, relation 1 is true.
True: d-separation is a necessary and sufficient condition for conditional independence in a BN. d-separation in G, implies conditional independence in any P that factorizes over G.

(b) In Figure 2, relation 2 is true.
True: Local independence describe a subset of the conditional independencies that hold in P.

(c) In Figure 2, relation 3 is true.
Trie: Global independence is a subset of the conditional independencies that hold in P.

(d) If G is an I-map for P, then P may have extra conditional independencies than G.
True: as an I-map only guarantees $I(G) \subseteq I(P)$, not equality.

(e) Two BN structures G_1 and G_2 are I-equivalent if they have the same skeleton and the same set of v-structures.
True: Two BN structures are I-equivalent if they have the same set of independencies, which is equivalent to having the same skeleton and the same set of v-structures.

(f) If G_1 is an I-map of distribution P, and G_1 has fewer edges than G_2, then G_2 is not a minimal I-map of P.
True: If G_1 is an I-map for P and has fewer edges than G_2, then G_2 is not minimal, as a minimal I-map has no unnecessary edges.

(g) The P-map of a distribution, if it exists, is unique.
True: If a perfect map (P-map) for a distribution exists, it is unique.

2 Exact Inference (Junction Tree a.k.a Clique Tree)

1. Consider the following Bayesian network G:

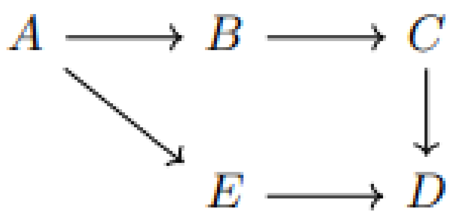

Figure 1: Caption for the image

2. We are going to construct a junction tree T from G. Please sketch the generated objects in each step.

   (a) (4 points) Moralize G to construct an undirected graph H.

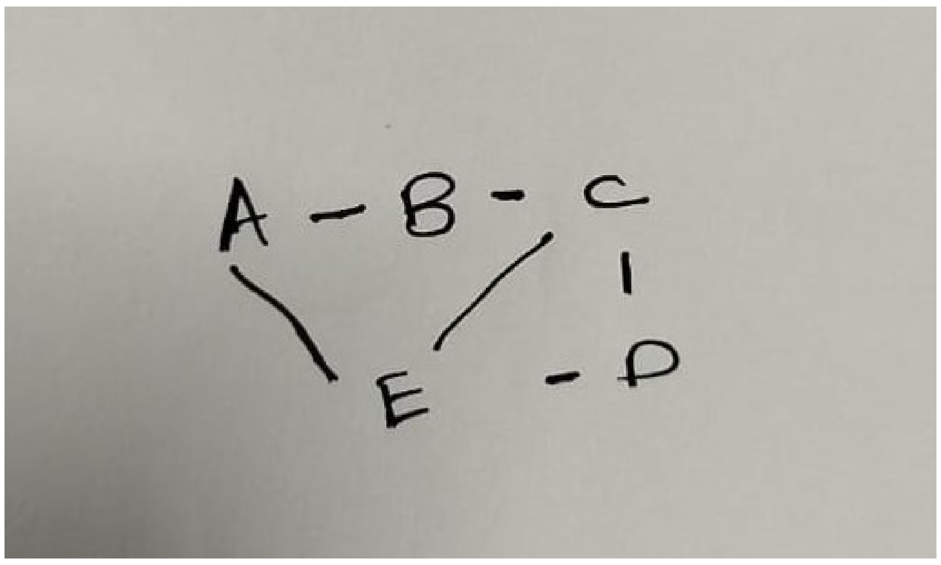

Figure 2: moralized graph

The original graph is moralized by adding edges between the parents who share a common child. For our case node E and C share a common child D, hence an edge is added between E and C.

   (b) (7 points) Triangulate H to construct a chordal graph H*. (Although there are many ways to triangulate a graph, for the ease of grading, please try adding fewest additional edges possible.)

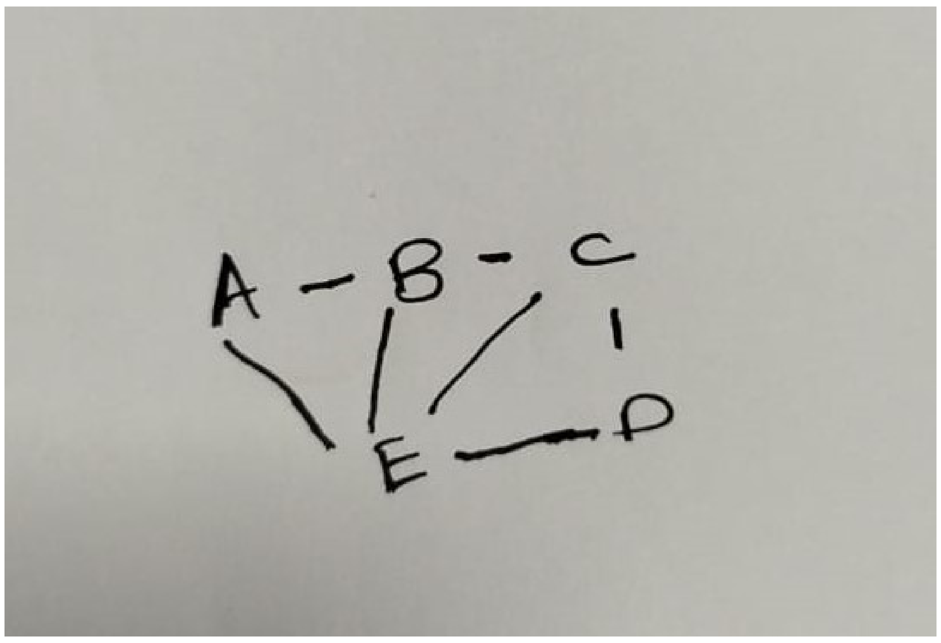

Figure 3: triangulated graph

The graph is triangulated by adding edges between the nodes that form a triangle with a chord. For our case, the graph is triangulated by adding an edge between B and E, so as to break the chord formed by the path A-B-C-D-E-A. When we are triangulating a graph, we are ensuring that no cycle of length four or more exists without a chord.

   (c) (7 points) Construct a cluster graph U where each node is a maximal clique Ci from H* and each edge is the sepset $S_{i,j} = C_i \cap C_j$ between adjacent cliques C_i and C_j.
The maximal cliques are the maximal complete subgraphs of the triangulated graph. The cluster graph is constructed by making each maximal clique a node and adding an edge between two nodes if the corresponding cliques share a variable. The sepset is the intersection of the two cliques.
The maximal cliques are:

- Clique 1: A, B, E
- Clique 2: B, C, E
- Clique 3: C, E, D

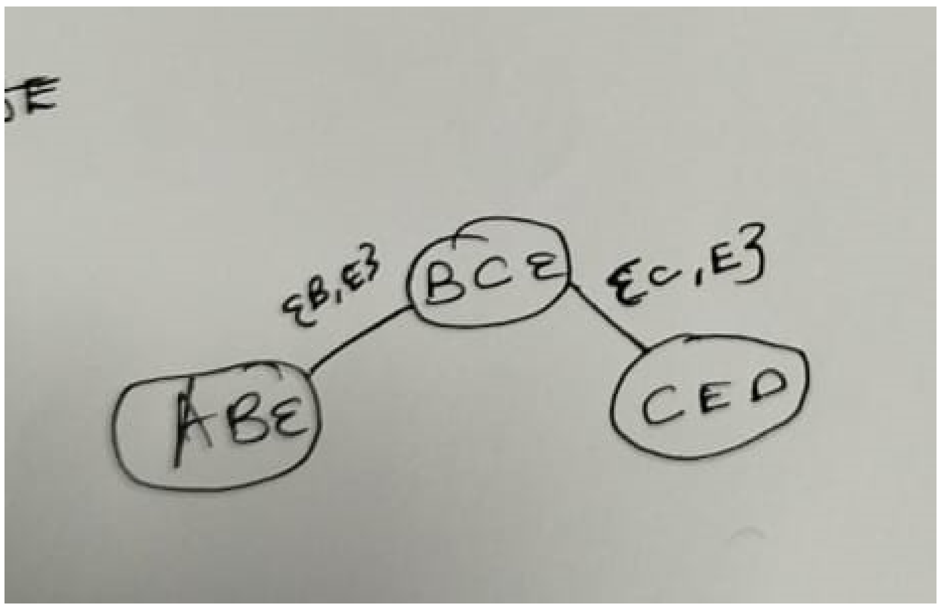

Figure 4: cluster graph

The cluster graph is shown in the figure above The edge between the nodes is the sepset between the two cliques. The sepsets are:

- Sepset 1: B, E
- Sepset 2: C, E

(d) (7 points) The junction tree T is the maximum spanning tree of U. (The cluster graph is small enough to calculate maximum spanning tree in one's head.)

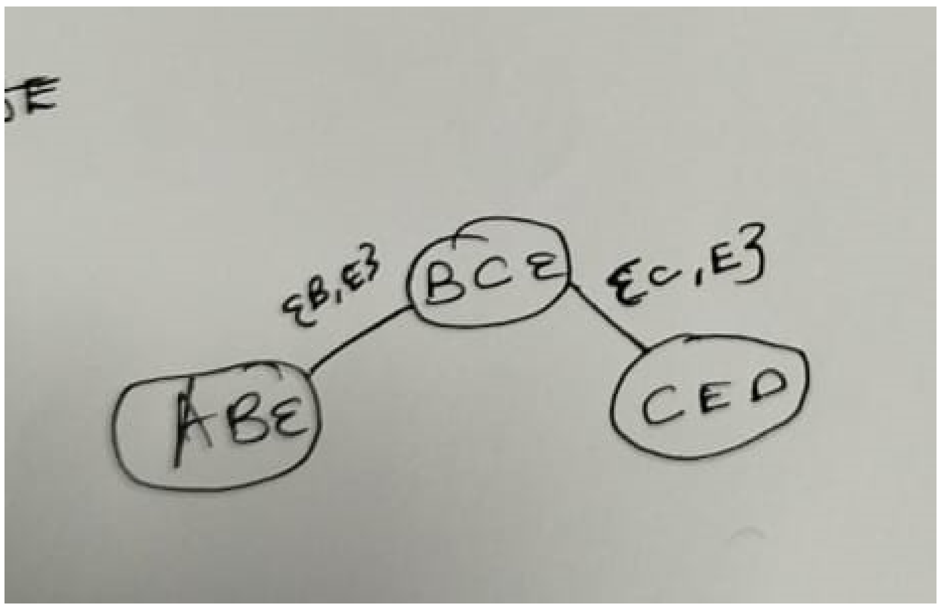

Figure 5: junction tree

The maximum spanning tree is the tree that connects all the nodes in the graph with the minimum possible number of edges. The junction tree is shown in the figure above. It is the same as the cluster graph because the cluster graph is already a tree and each edge has a weight of 2.

3 Parameter Estimation (HMM - EM Algorithm)

1. Consider an HMM with T time steps, M discrete states, and K-dimensional observations as in Figure 3, where $z_t \in \{0,1\}^M$, $\sum_s z_{ts} = 1$, $x_t \in \mathbb{R}^K$ for $t \in [T]$.

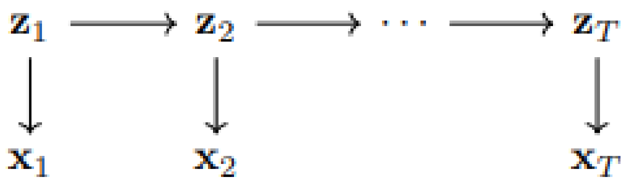

Figure 3: A hidden Markov model.

Figure 6: Caption for the image

2. The joint distribution factorizes over the graph:

$$p(x_{1:T}, z_{1:T}) = p(z_1) \prod_{t=2}^{T} p(z_t \mid z_{t-1}) \prod_{t=1}^{T} p(x_t \mid z_t). \tag{4}$$

Now consider the parameterization of CPDs. Let $\pi \in \mathbb{R}^M$ be the initial state distribution and $A \in \mathbb{R}^{M\times M}$ be the transition matrix. The emission density $f(\cdot)$ is parameterized by ϕ_i at state i. In other words,

$$p(z_{1i} = 1) = \pi_i, \quad p(z_1) = \prod_{i=1}^{M} \pi_i^{z_{1i}}, \tag{5}$$

$$p(z_{tj} = 1 \mid z_{t-1,i} = 1) = a_{ij}, \quad p(z_t \mid z_{t-1}) = \prod_{i=1}^{M}\prod_{j=1}^{M} a_{ij}^{z_{t-1,i}z_{tj}}, \quad t = 2, \ldots, T, \tag{6}$$

$$p(x_t \mid z_{ti} = 1) = f(x_t; \phi_i), \quad p(x_t \mid z_t) = \prod_{i=1}^{M} f(x_t; \phi_i)^{z_{ti}}, \quad t = 1, \ldots, T. \tag{7}$$

Let $\theta = (\pi, A, \{\phi_i\}_{i=1}^M)$ be the set of parameters of the HMM. Given the empirical distribution $\hat{p}$ of $x_{1:T}$, we would like to find the MLE of θ by solving the following problem:

$$\max_{\theta} \mathbb{E}_{x_{1:T}\sim\hat{p}}[\log p_\theta(x_{1:T})]. \tag{8}$$

However, the marginal likelihood is intractable due to summation over M^T terms:

$$p_\theta(x_{1:T}) = \sum_{z_{1:T}} p_\theta(x_{1:T}, z_{1:T}). \tag{9}$$

An alternative is to use the EM algorithm as we saw in the class.

(a) (10 points) Show that the EM updates can take the following form:

$$\theta^* \leftarrow \arg\max_{\theta} \mathbb{E}_{x_{1:T}\sim\hat{p}}[F(x_{1:T}; \theta)] \tag{10}$$

where

$$F(x_{1:T}; \theta) := \sum_{i=1}^{M} \gamma(z_{1i}) \log \pi_i + \sum_{t=2}^{T}\sum_{i=1}^{M}\sum_{j=1}^{M} \xi(z_{t-1,i}, z_{tj}) \log a_{ij}$$
$$+ \sum_{t=1}^{T}\sum_{i=1}^{M} \gamma(z_{ti}) \log f(x_t; \phi_i) \tag{11}$$

and γ and ξ are the posterior expectations over current parameters $\hat{\theta}$:

$$\gamma(z_{ti}) := \mathbb{E}_{z_{1:T}\sim p_{\hat{\theta}}(z_{1:T}|x_{1:T})}[z_{ti}] = p_{\hat{\theta}}(z_{ti} = 1 \mid x_{1:T}), \quad t = 1, \ldots, T \tag{12}$$

$$\xi(z_{t-1,i}, z_{tj}) := \mathbb{E}_{z_{1:T}\sim p_{\hat{\theta}}(z_{1:T}|x_{1:T})}[z_{t-1,i}z_{tj}] = p_{\hat{\theta}}(z_{t-1,i}z_{tj} = 1 \mid x_{1:T}), \quad t = 2, \ldots, T \tag{13}$$

Solution:
The EM algorithm is used to maximize the expected complete data log-likelihood w.r.t. the parameters θ. The expected complete data log-likelihood is given by the sum of the expected values of the complete data log-likelihood w.r.t. the posterior distribution of the latent variables. The expected complete data log-likelihood is given by equation (1). The complete data log-likelihood is given by equation (2). The joint distribution of latent variables and observations is given by equation (3). The expectation of the complete data log-likelihood w.r.t. the posterior distribution of latent variables is given by equation (1). To maximize the expected complete data log-likelihood, we need to maximize the function $F(x_{1:T}; \theta)$ w.r.t. the parameters θ. The joint distribution of latent variables $z_{1:T}$ and observations $x_{1:T}$ is given by

$$p_{\hat{\theta}}(z_{1:T}, x_{1:T}) = p_{\hat{\theta}}(z_1) \prod_{t=2}^{T} p_{\hat{\theta}}(z_t \mid z_{t-1}) \prod_{t=1}^{T} p_{\hat{\theta}}(x_t \mid z_t). \tag{3}$$

The complete data log-likelihood is

$$\log p_{\hat{\theta}}(z_{1:T}, x_{1:T}) = \sum_{i=1}^{M} z_{1i} \log \pi_i + \sum_{t=2}^{T}\sum_{i=1}^{M}\sum_{j=1}^{M} z_{t-1,i} z_{tj} \log a_{ij} + \sum_{t=1}^{T}\sum_{i=1}^{M} z_{ti} \log f(x_t; \phi_i). \tag{2}$$

This is equals to the function $F(x_{1:T}; \theta)$ given by equation (11). The expectation of the complete data log-likelihood w.r.t. the posterior distribution of latent variables is The expectation of the complete data log-likelihood w.r.t. the posterior distribution of latent variables is

$$\mathbb{E}_{z_{1:T} \sim p_{\hat{\theta}}(z_{1:T}|x_{1:T})}[\log p_{\hat{\theta}}(z_{1:T}, x_{1:T})] = \sum_{i=1}^{M} \gamma(z_{1i}) \log \pi_i + \sum_{t=2}^{T}\sum_{i=1}^{M}\sum_{j=1}^{M} \xi(z_{t-1,i}, z_{tj}) \log a_{ij} + \sum_{t=1}^{T}\sum_{i=1}^{M} \gamma(z_{ti}) \log f(x_t; \phi_i). \tag{1}$$

To maximize the expected complete data log-likelihood, we need to maximize the function $F(x_{1:T}; \theta)$ w.r.t. the parameters θ.

$$\theta^* = \arg\max_{\theta} \mathbb{E}_{x_{1:T} \sim \hat{p}}[F(x_{1:T}; \theta)]$$

(b) (0 points) (No need to answer.) Suppose γ and ξ are given, and we use isotropic Gaussian $x_t \mid z_{ti} = 1 \sim \mathcal{N}(\mu_i, \sigma_i^2 I)$ as the emission distribution. Then the parameter updates have the following closed form:

$$\pi_i^* \propto \mathbb{E}_{x_{1:T} \sim \hat{p}}[\gamma(z_{1i})] \tag{14}$$

$$a_{ij}^* \propto \mathbb{E}_{x_{1:T} \sim \hat{p}}\left[\sum_{t=2}^{T} \xi(z_{t-1,i}, z_{tj})\right] \tag{15}$$

$$\mu_{ik}^* = \frac{\mathbb{E}_{x_{1:T} \sim \hat{p}}\left[\sum_{t=1}^{T} \gamma(z_{ti}) x_t\right]}{\mathbb{E}_{x_{1:T} \sim \hat{p}}\left[\sum_{t=1}^{T} \gamma(z_{ti})\right]} \tag{16}$$

$$\sigma_i^{2*} = \frac{\mathbb{E}_{x_{1:T} \sim \hat{p}}\left[\sum_{t=1}^{T} \gamma(z_{ti}) \|x_t - \mu_i\|_2^2\right]}{\mathbb{E}_{x_{1:T} \sim \hat{p}}\left[\sum_{t=1}^{T} \gamma(z_{ti})\right] K} \tag{17}$$

(c) (10 points) We will use the belief propagation algorithm (Koller and Friedman, 2009, Alg. 10.2) to perform inference for all marginal queries:

$$\gamma(z_t) = p_{\hat{\theta}}(z_t \mid x_{1:T}), \quad t = 1, \dots, T \tag{18}$$

$$\xi(z_{t-1}, z_t) = p_{\hat{\theta}}(z_{t-1}, z_t \mid x_{1:T}), \quad t = 2, \dots, T \tag{19}$$

For convenience, the notation $\hat{\theta}$ will be omitted from now on. Derive the following BP updates:

$$\gamma(z_t) = \frac{1}{Z(x_{1:T})} \cdot s(z_t) \tag{20}$$

$$\xi(z_{t-1}, z_t) = \frac{1}{Z(x_{1:T})} \cdot c(z_{t-1}, z_t) \tag{21}$$

where

$$s(z_t) = \alpha(z_t)\beta(z_t), \quad t = 1, \dots, T \tag{23}$$

$$c(z_{t-1}, z_t) = p(z_t \mid z_{t-1}) p(x_t \mid z_t) \alpha(z_{t-1}) \beta(z_t), \quad t = 2, \dots, T \tag{24}$$

$$Z(x_{1:T}) = \sum_{z_t} s(z_t) \tag{25}$$

and

$$\alpha(z_1) = p(z_1) p(x_1 \mid z_1) \tag{26}$$

$$\alpha(z_t) = p(x_t \mid z_t) \sum_{z_{t-1}} p(z_t \mid z_{t-1}) \alpha(z_{t-1}), \quad t = 2, \dots, T \tag{27}$$

$$\beta(z_{t-1}) = \sum_{z_t} p(z_t \mid z_{t-1}) p(x_t \mid z_t) \beta(z_t), \quad t = 2, \dots, T \tag{28}$$

$$\beta(z_T) = 1 \tag{29}$$

Solution:
To derive the belief propagation updates, we need to express the marginal queries in terms of the forward and backward messages. The marginal queries are given by

$$\gamma(z_t) = p(z_t \mid x_{1:T}), \quad t = 1, \dots, T \tag{18}$$

$$\xi(z_{t-1}, z_t) = p(z_{t-1}, z_t \mid x_{1:T}), \quad t = 2, \dots, T \tag{19}$$

The posterior marginals can be expressed in terms of the forward and backward messages. The forward message is given by $\alpha(z_t) = p(x_{1:t}, z_t)$
The backward message is given by

$\beta(z_t) = p(x_{t+1:T} \mid z_t)$ Let us derive the posterior marginal $\gamma(z_t)$ in terms of the forward and backward messages. The posterior marginal $\gamma(z_t)$ is given by

$$\begin{aligned}
\gamma(z_t) &= p(z_t \mid x_{1:T}) \\
&= \frac{p(x_{1:T}, z_t)}{p(x_{1:T})} \\
&= \frac{p(x_{1:t}, z_t) p(x_{t+1:T} \mid z_t)}{p(x_{1:T})} \\
&= \frac{\alpha(z_t)\beta(z_t)}{p(x_{1:T})} \\
&= \frac{\alpha(z_t)\beta(z_t)}{Z(x_{1:T})}
\end{aligned} \tag{22}$$

where $Z(x_{1:T}) = \sum_{z_t} \alpha(z_t)\beta(z_t)$. The posterior marginal $\xi(z_{t-1}, z_t)$ can be expressed in terms of the forward and backward messages as

$$\begin{aligned}
\xi(z_{t-1}, z_t) &= p(z_{t-1}, z_t \mid x_{1:T}) \\
&= \frac{p(x_{1:T}, z_{t-1}, z_t)}{p(x_{1:T})} \\
&= \frac{p(x_{1:t-1}, z_{t-1}) p(x_t \mid z_{t-1}) p(x_{t+1:T} \mid z_t)}{p(x_{1:T})} \\
&= \frac{\alpha(z_{t-1}) p(z_t \mid z_{t-1}) p(x_t \mid z_t) \beta(z_t)}{p(x_{1:T})} \\
&= \frac{\alpha(z_{t-1}) p(z_t \mid z_{t-1}) p(x_t \mid z_t) \beta(z_t)}{Z(x_{1:T})}
\end{aligned} \tag{23}$$

where $Z(x_{1:T}) = \sum_{z_t} \alpha(z_t)\beta(z_t)$. The belief propagation updates are given by

$$\gamma(z_t) = \frac{1}{Z(x_{1:T})} \cdot s(z_t) \tag{20}$$

$$\xi(z_{t-1}, z_t) = \frac{1}{Z(x_{1:T})} \cdot c(z_{t-1}, z_t) \tag{21}$$

where

$$s(z_t) = \alpha(z_t)\beta(z_t), \quad t = 1, \ldots, T \tag{23}$$

$$c(z_{t-1}, z_t) = p(z_t \mid z_{t-1}) p(x_t \mid z_t) \alpha(z_{t-1}) \beta(z_t), \quad t = 2, \ldots, T \tag{24}$$

$$Z(x_{1:T}) = \sum_{z_t} s(z_t) \tag{25}$$

The belief propagation updates are derived by expressing the posterior marginals in terms of the forward and backward messages. The posterior marginals are then expressed in terms of the forward and backward messages. The belief propagation updates are then derived by substituting the expressions for the posterior marginals in terms of the forward and backward messages into the belief propagation updates.

(d) (0 points) (No need to answer.) Implemented as above, the (α, β)-recursion is likely to encounter numerical instability due to repeated multiplication of small values. One way to mitigate the numerical issue is to scale (α, β) messages at each step t, so that the scaled values are always in some appropriate range, while not affecting the inference result for (γ, ξ).

Recall that the forward message is in fact a joint distribution

$$\alpha(z_t) = p(x_{1:t}, z_t). \tag{30}$$

Define scaled messages by re-normalizing α w.r.t. z_t:

$$\hat{\alpha}(z_t) = \frac{1}{Z(x_{1:t})} \cdot \alpha(z_t), \tag{31}$$

$$Z(x_{1:t}) = \sum_{z_t} \alpha(z_t). \tag{32}$$

Furthermore, define

$$r_1 := Z(x_1), \tag{33}$$

$$r_t := \frac{Z(x_{1:t})}{Z(x_{1:t-1})}, \quad t = 2, \ldots, T. \tag{34}$$

Notice that $Z(x_{1:t}) = r_1 \cdot \ldots \cdot r_t$, hence

$$\hat{\alpha}(z_t) = \frac{1}{r_1 \cdot \ldots \cdot r_t} \cdot \alpha(z_t). \tag{35}$$

Plugging $\hat{\alpha}$ into forward messages, the new $\hat{\alpha}$-recursion is

$$\hat{\alpha}(z_1) = \frac{1}{r_1} \cdot p(z_1) p(x_1 \mid z_1), \tag{36}$$

$$\hat{\alpha}(z_t) = \frac{1}{r_t} \cdot p(x_t \mid z_t) \sum_{z_{t-1}} p(z_t \mid z_{t-1}) \hat{\alpha}(z_{t-1}), \quad t = 2, \ldots, T. \tag{37}$$

Since $\hat{\alpha}$ is normalized, each r_t serves as the normalizing constant:

$$r_t = \sum_{z_t} \tilde{\alpha}(z_t). \tag{38}$$

Now switch focus to β. In order to make the inference for (γ, ξ) invariant of scaling, β has to be scaled in a way that counteracts the scaling on α. Plugging $\hat{\alpha}$ into the marginal queries,

$$\gamma(z_t) = \frac{1}{Z(x_{1:T})} \cdot r_1 \cdot \ldots \cdot r_t \cdot \hat{\alpha}(z_t)\beta(z_t), \tag{39}$$

$$\xi(z_{t-1}, z_t) = \frac{1}{Z(x_{1:T})} \cdot p(z_t \mid z_{t-1})p(x_t \mid z_t) \cdot r_1 \cdot \ldots \cdot r_{t-1} \cdot \hat{\alpha}(z_{t-1})\beta(z_t). \tag{40}$$

Since $Z(x_{1:T}) = r_1 \cdot \ldots \cdot r_T$, a natural scaling scheme for β is

$$\hat{\beta}(z_{t-1}) = \frac{1}{r_t \cdot \ldots \cdot r_T} \cdot \beta(z_{t-1}), \quad t = 2, \ldots, T, \tag{41}$$

$$\hat{\beta}(z_T) := \beta(z_T). \tag{42}$$

which simplifies the expression for marginals (γ, ξ) to

$$\gamma(z_t) = \hat{\alpha}(z_t)\hat{\beta}(z_t), \tag{43}$$

$$\xi(z_{t-1}, z_t) = \frac{1}{r_t} \cdot p(z_t \mid z_{t-1})p(x_t \mid z_t)\hat{\alpha}(z_{t-1})\hat{\beta}(z_t). \tag{44}$$

The new $\hat{\beta}$-recursion can be obtained by plugging $\hat{\beta}$ into backward messages:

$$\hat{\beta}(z_{t-1}) = \frac{1}{r_t} \cdot \sum_{z_t} p(z_t \mid z_{t-1})p(x_t \mid z_t)\hat{\beta}(z_t), \quad t = 2, \ldots, T, \tag{45}$$

$$\hat{\beta}(z_T) = 1. \tag{46}$$

In other words, $\hat{\beta}(z_{t-1})$ is scaled by $1/r_t$, the normalizer of $\hat{\alpha}(z_t)$.
The full algorithm is summarized below.

Algorithm 1: Exact inference for (γ, ξ)

i. Scaled forward message for $t = 1$:

$$\tilde{\alpha}(z_1) = p(z_1)p(x_1 \mid z_1), \tag{47}$$

$$r_1 = \sum_{z_1} \tilde{\alpha}(z_1), \tag{48}$$

$$\hat{\alpha}(z_1) = \frac{1}{r_1} \cdot \tilde{\alpha}(z_1). \tag{49}$$

ii. Scaled forward message for $t = 2, \ldots, T$:

$$\tilde{\alpha}(z_t) = p(x_t \mid z_t) \sum_{z_{t-1}} p(z_t \mid z_{t-1})\hat{\alpha}(z_{t-1}), \tag{50}$$

$$r_t = \sum_{z_t} \tilde{\alpha}(z_t), \tag{51}$$

$$\hat{\alpha}(z_t) = \frac{1}{r_t} \cdot \tilde{\alpha}(z_t). \tag{52}$$

iii. Scaled backward message for $t = T + 1$:

$$\hat{\beta}(z_T) = 1. \tag{53}$$

iv. Scaled backward message for $t = T, \ldots, 2$:

$$\hat{\beta}(z_{t-1}) = \frac{1}{r_t} \cdot \sum_{z_t} p(z_t \mid z_{t-1})p(x_t \mid z_t)\hat{\beta}(z_t). \tag{54}$$

v. Singleton marginal for $t = 1, \ldots, T$:

$$\gamma(z_t) = \hat{\alpha}(z_t)\hat{\beta}(z_t). \tag{55}$$

vi. Pairwise marginal for $t = 2, \ldots, T$:

$$\xi(z_{t-1}, z_t) = \frac{1}{r_t} \cdot p(z_t \mid z_{t-1})p(x_t \mid z_t)\hat{\alpha}(z_{t-1})\hat{\beta}(z_t). \tag{56}$$

(e) (15 points) We will implement the EM algorithm (also known as Baum-Welch algorithm), where the E-step performs exact inference and the M-step updates parameter estimates. Please complete the TODO blocks in the provided template `baum_welch.py` and submit it to Gradescope. The template contains a toy problem to play with. The submitted code will be tested against randomly generated problem instances.

4 D-Separation

1. (5 points) Given that the gray nodes are observed, are the variables in A independent to those in B?

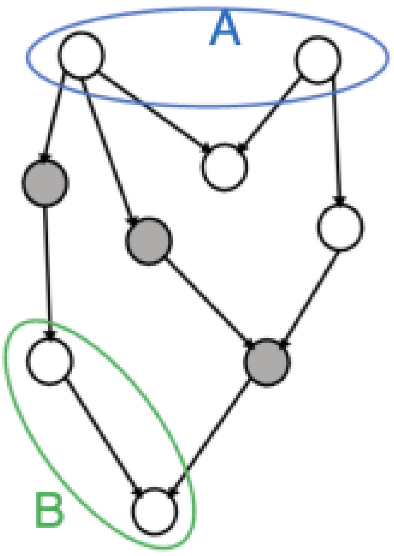

Figure 4: A Bayesian network.

Figure 7: Unlabeled image

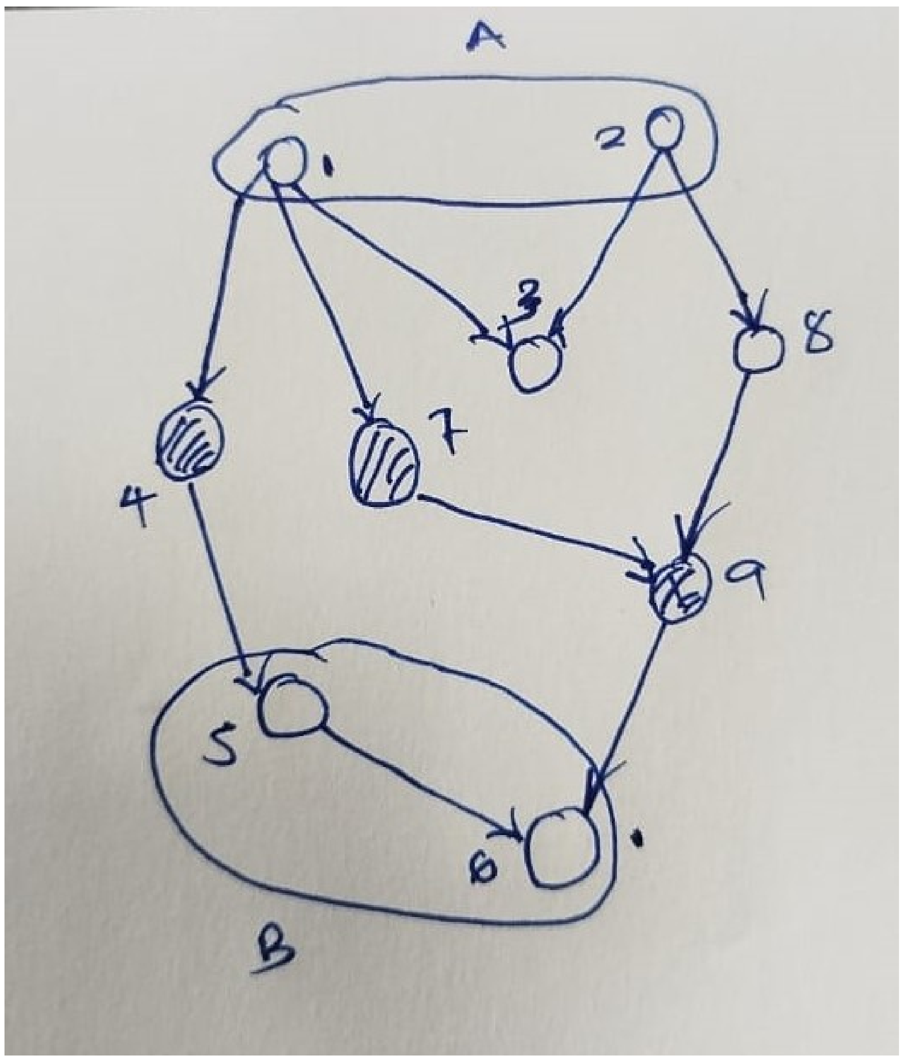

Figure 8: Labeled image

To check if the variables in A are independent of those in B, we need to check if there is a path between the two sets of variables that is not blocked by any of the observed variables.

- path 1: 1 → 4 → 5 is blocked by 4 because 4 is observed. **(Markov property)**
- path 2: 1 → 7 → 9 is blocked by 7 because 7 is observed. **(Markov property)**
- path 3: 1 → 3 → 2 is blocked by 3 because 3 is not observed. **(common effect property)**
- path 4: 8 → 9 → 6 is blocked by 9 because 9 is observed. **(Markov property)**

Since all paths are blocked, the variables in A are independent of those in B.

2. (5 points) Given that the gray nodes are observed, are the nodes 2 and 3 d-separated?

- path 1: 2 → 1 → 3 is blocked by 1 because 1 is observed. **(common cause property)**
- path 2: 2 → 4 → 3 is active because 7 which is a child of 4 is observed. **(common effect property)**

Since there is an active path, the nodes 2 and 3 are not d-separated.

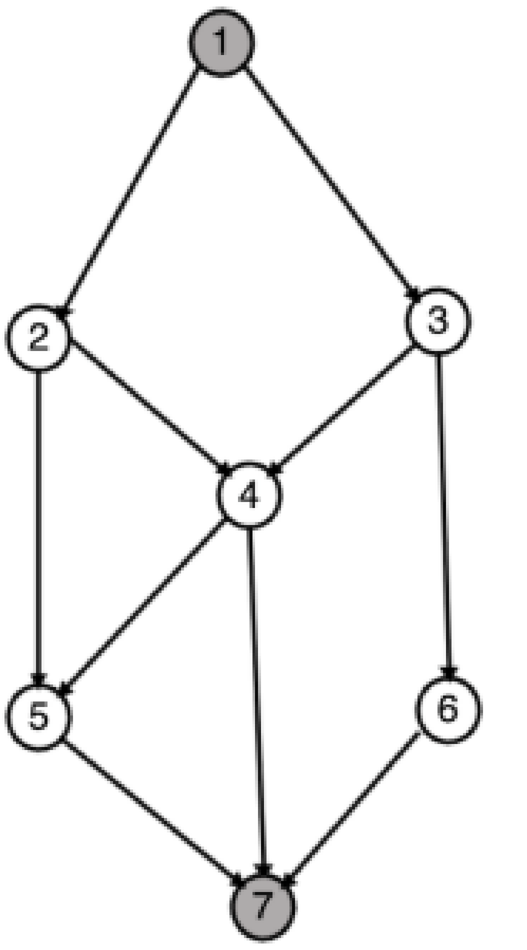

Figure 5: A Bayesian network.

3. (5 points) Given that the gray nodes are observed, are the nodes 5 and 6 d-separated?

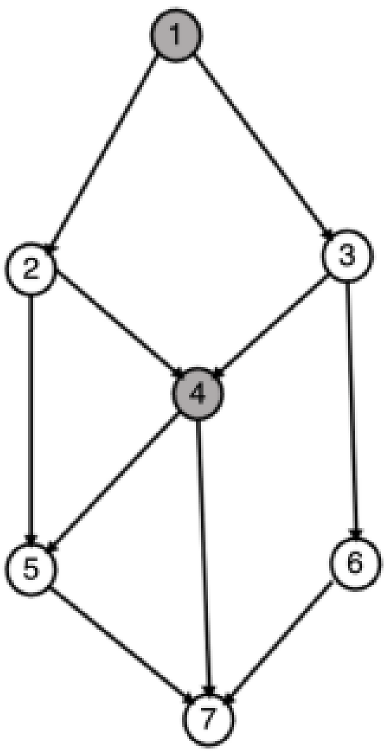

Figure 6: A Bayesian network.

- path 1: 5 → 4 → 3 is blocked by 3 because 3 is observed. **(Markov property)**
- path 2: 5 → 7 → 6 is blocked by 7 because 7 is not observed. **(common effect property)**
- path 3: 5 → 2 → 1 → 3 → 6 is blocked by 1 because 1 is observed. **(common cause property)**
- path 4: 5 → 2 → 4 → 3 → 6 is an active path because 4 is observed. **(common effect property)**

Since there is an active path, the nodes 5 and 6 are not d-separated.

4. (15 points) Write a python program to check d-separation. Three files have been provided. You have to modify only the `BN.py` file. The instructions on how to run the code are in the `check_dsep.py` file. All the files are provided as a zip file named `code_files.zip`.