Exercises for Chapter 3 - The Bayesian Network Representation

abc1306

October 25, 2025

Textbook: "Probabilistic Graphical Models - Principles and Techniques", Daphne Koller and Nir Friedman, Chapter 3 - Bayesian Network Representation.

Ex. 3.1 (p. 96)

Let X_1 and X_2 be independent fair coin flips (binary variables, value 0 or 1), so $P(X_1 = 0) = P(X_1 = 1) = 0.5$ and $P(X_2 = 0) = P(X_2 = 1) = 0.5$. Let X_3 be the XOR of X_1 and X_2, i.e., $X_3 = X_1 \oplus X_2$. This means $X_3 = 1$ if $X_1 \neq X_2$, and $X_3 = 0$ if $X_1 = X_2$.

The joint distribution is: $P(0, 0, 0) = 0.25$ $P(0, 1, 1) = 0.25$ $P(1, 0, 1) = 0.25$ $P(1, 1, 0) = 0.25$ All other combinations have probability 0.

Let's check the independencies:

1. $(X_1 \perp X_2)$**:** This holds by construction, as X_1 and X_2 are independent coin flips. $P(X_1, X_2) = P(X_1)P(X_2) = 0.25$ for all joint assignments.
2. $(X_1 \perp X_3)$**:** We need to check if $P(X_1, X_3) = P(X_1)P(X_3)$.
   - $P(X_3 = 0) = P(0, 0, 0) + P(1, 1, 0) = 0.25 + 0.25 = 0.5$.
   - $P(X_3 = 1) = P(0, 1, 1) + P(1, 0, 1) = 0.25 + 0.25 = 0.5$.
   - $P(X_1 = 0) = 0.5$.
   - $P(X_1 = 0, X_3 = 0) = P(0, 0, 0) = 0.25$. Compare to $P(X_1 = 0)P(X_3 = 0) = 0.5 \times 0.5 = 0.25$. They are equal.
   - $P(X_1 = 0, X_3 = 1) = P(0, 1, 1) = 0.25$. Compare to $P(X_1 = 0)P(X_3 = 1) = 0.5 \times 0.5 = 0.25$. They are equal.
   - By symmetry, this also holds for $X_1 = 1$. So, $(X_1 \perp X_3)$ holds.
3. $(X_2 \perp X_3)$**:** This holds by symmetry with the previous case.
4. $(X_1, X_2 \perp X_3)$**:** We need to check if $P(X_1, X_2, X_3) = P(X_1, X_2)P(X_3)$. Let's check for $X_1 = 0, X_2 = 0, X_3 = 1$.
   - $P(0, 0, 1) = 0$ according to our joint distribution.
   - $P(X_1 = 0, X_2 = 0) = P(X_1 = 0)P(X_2 = 0) = 0.5 \times 0.5 = 0.25$.
   - $P(X_3 = 1) = 0.5$.
   - $P(X_1 = 0, X_2 = 0)P(X_3 = 1) = 0.25 \times 0.5 = 0.125$.

- Since $0 \neq 0.125$, the independence $(X_1, X_2 \perp X_3)$ does *not* hold.

Thus, this distribution satisfies pairwise independence but not mutual independence.

Ex 3.2 (p.96)

a. We start with the chain rule for probability:

$$P(C, X_1, \ldots, X_n) = P(C)P(X_1|C)P(X_2|C, X_1)\ldots P(X_n|C, X_1, \ldots, X_{n-1})$$

The naive Bayes assumption is $(X_i \perp X_{-i}|C)$ for all i, where $X_{-i} = \{X_1, \ldots, X_n\} \setminus \{X_i\}$. A specific consequence of this assumption is that $(X_i \perp \{X_1, \ldots, X_{i-1}\}|C)$ for $i > 1$. Applying this conditional independence to each term $P(X_i|C, X_1, \ldots, X_{i-1})$ in the chain rule gives:

$$P(X_i|C, X_1, \ldots, X_{i-1}) = P(X_i|C)$$

Substituting this back into the chain rule expression yields:

$$P(C, X_1, \ldots, X_n) = P(C)P(X_1|C)P(X_2|C)\ldots P(X_n|C) = P(C)\prod_{i=1}^{n} P(X_i|C)$$

This is equation (3.7).

b. Using Bayes' theorem:

$$P(C = c|x_1, \ldots, x_n) = \frac{P(x_1, \ldots, x_n|C = c)P(C = c)}{P(x_1, \ldots, x_n)}$$

From the naive Bayes assumption (or factorization in eq. 3.7), we know $P(x_1, \ldots, x_n|C = c) = \prod_{i=1}^{n} P(x_i|C = c)$. Substituting this into Bayes' theorem:

$$P(C = c|x_1, \ldots, x_n) = \frac{\left(\prod_{i=1}^{n} P(x_i|C = c)\right) P(C = c)}{P(x_1, \ldots, x_n)}$$

Now form the ratio for two classes c^1 and c^2:

$$\frac{P(C = c^1|x_1, \ldots, x_n)}{P(C = c^2|x_1, \ldots, x_n)} = \frac{\frac{\left(\prod_{i=1}^{n} P(x_i|C=c^1)\right)P(C=c^1)}{P(x_1,\ldots,x_n)}}{\frac{\left(\prod_{i=1}^{n} P(x_i|C=c^2)\right)P(C=c^2)}{P(x_1,\ldots,x_n)}}$$

The denominator $P(x_1, \ldots, x_n)$ cancels out:

$$\frac{P(C = c^1|x_1, \ldots, x_n)}{P(C = c^2|x_1, \ldots, x_n)} = \frac{\left(\prod_{i=1}^{n} P(x_i|C = c^1)\right) P(C = c^1)}{\left(\prod_{i=1}^{n} P(x_i|C = c^2)\right) P(C = c^2)}$$

Rearranging the terms gives equation (3.8):

$$\frac{P(C = c^1|x_1, \ldots, x_n)}{P(C = c^2|x_1, \ldots, x_n)} = \frac{P(C = c^1)}{P(C = c^2)} \prod_{i=1}^{n} \frac{P(x_i|C = c^1)}{P(x_i|C = c^2)}$$

c. Let the binary values be $\{0, 1\}$. We are interested in the log odds ratio for $C = 1$ vs $C = 0$. Let x_i denote the observed value (0 or 1) for variable X_i. From part (b):

$$\log \frac{P(C = 1|x_1, \ldots, x_n)}{P(C = 0|x_1, \ldots, x_n)} = \log \frac{P(C = 1)}{P(C = 0)} + \sum_{i=1}^{n} \log \frac{P(x_i|C = 1)}{P(x_i|C = 0)}$$

Let's analyze the term inside the sum for a single X_i: $\log \frac{P(X_i=x_i|C=1)}{P(X_i=x_i|C=0)}$. Let $P(X_i = 1|C = 1) = \theta_{i|1}$ and $P(X_i = 1|C = 0) = \theta_{i|0}$. Then $P(X_i = 0|C = 1) = 1 - \theta_{i|1}$ and $P(X_i = 0|C = 0) = 1 - \theta_{i|0}$.

If $x_i = 1$, the term is $\log \frac{\theta_{i|1}}{\theta_{i|0}}$. If $x_i = 0$, the term is $\log \frac{1-\theta_{i|1}}{1-\theta_{i|0}}$.

We can express this using the observed value $x_i \in \{0, 1\}$:

$$\begin{aligned}\log \frac{P(x_i|C = 1)}{P(x_i|C = 0)} &= x_i \log \frac{\theta_{i|1}}{\theta_{i|0}} + (1 - x_i) \log \frac{1 - \theta_{i|1}}{1 - \theta_{i|0}} \\ &= x_i \left(\log \frac{\theta_{i|1}}{\theta_{i|0}} - \log \frac{1 - \theta_{i|1}}{1 - \theta_{i|0}} \right) + \log \frac{1 - \theta_{i|1}}{1 - \theta_{i|0}}\end{aligned}$$

Let $\alpha_i = \log \frac{\theta_{i|1}(1-\theta_{i|0})}{\theta_{i|0}(1-\theta_{i|1})}$ (the log odds ratio for X_i dependent on C). The term becomes $\alpha_i x_i + \log \frac{1-\theta_{i|1}}{1-\theta_{i|0}}$.

Substituting this back into the sum:

$$\begin{aligned}\log \frac{P(C = 1|x_1, \ldots, x_n)}{P(C = 0|x_1, \ldots, x_n)} &= \log \frac{P(C = 1)}{P(C = 0)} + \sum_{i=1}^{n} \left(\alpha_i x_i + \log \frac{1 - \theta_{i|1}}{1 - \theta_{i|0}} \right) \\ &= \left(\sum_{i=1}^{n} \alpha_i x_i \right) + \left(\log \frac{P(C = 1)}{P(C = 0)} + \sum_{i=1}^{n} \log \frac{1 - \theta_{i|1}}{1 - \theta_{i|0}} \right)\end{aligned}$$

This is a linear function of the form $\sum_{i=1}^{n} \alpha_i x_i + \alpha_0$, where α_0 is the constant term collecting all terms not dependent on the observed x_i values.

Ex 3.3 (p.96)

a. Define constraints on $P(A|B, E)$ that imply the explaining away property.

The "explaining away" property is a common form of **intercausal reasoning**. In the context of the alarm example ($B \to A \leftarrow E$), it means that given the alarm is on (a^1), observing one cause (e.g., Earthquake, e^1) *reduces* the probability of the other cause (e.g., Burglary, b^1) .

$$\left[P(b^1|a^1, e^1) < P(b^1|a^1) \right]$$

This "explaining away" occurs when the causes B and E are not synergistic. A common constraint is that the causes interact negatively (or sub-additively). This can be expressed by stating that the increase in probability for the alarm from one cause (Burglary) is *less* when the other cause (Earthquake) is already present:

$$\left[P(a^1|b^1, e^1) - P(a^1|b^0, e^1) \leq P(a^1|b^1, e^0) - P(a^1|b^0, e^0) \right]$$

This indicates that the causes compete to explain the effect.

b. Show that if the alarm always (deterministically) goes off whenever there is an earthquake, then $P(b^1|a^1, e^1) = P(b^1)$

We are given:

1. $P(a^1|b^1, e^1) = 1$
2. $P(a^1|b^0, e^1) = 1$

We also assume the causes are independent, so $(B \perp E)$, which means $P(b^1|e^1) = P(b^1)$ and $P(b^0|e^1) = P(b^0)$.

We want to prove $P(b^1|a^1, e^1) = P(b^1)$.

1. Start with Bayes' rule:
$$\left[P(b^1|a^1, e^1) = \frac{P(a^1|b^1, e^1)P(b^1|e^1)}{P(a^1|e^1)}\right]$$

2. Substitute the given information and the independence assumption:
$$\left[P(b^1|a^1, e^1) = \frac{1 \cdot P(b^1)}{P(a^1|e^1)}\right]$$

3. Now, we must find the denominator, $P(a^1|e^1)$, by marginalizing over B:
$$P(a^1|e^1) = P(a^1|b^0, e^1)P(b^0|e^1) + P(a^1|b^1, e^1)P(b^1|e^1)$$

4. Substitute the given information and the independence assumption into this equation:
$$P(a^1|e^1) = (1 \cdot P(b^0)) + (1 \cdot P(b^1))$$
$$P(a^1|e^1) = P(b^0) + P(b^1) = 1$$

5. Finally, substitute this result back into the equation from step 2:
$$P(b^1|a^1, e^1) = \frac{1 \cdot P(b^1)}{1}$$
$$P(b^1|a^1, e^1) = P(b^1)$$

This shows that observing the earthquake (e^1) **fully explains** the alarm (a^1). The alarm, therefore, provides no additional evidence about the burglary, and the probability of burglary reverts to its prior.

Ex 3.4 (p.96)

Provide a realistic example and construct a CPD $P(Z|X, Y)$ that exhibits the opposite interaction of explaining away

This type of intercausal reasoning occurs when the causes are **synergistic**. Observing the effect (z^1) and one cause (y^1) makes the other cause (x^1) *more* likely: $P(x^1|z^1) < P(x^1|y^1, z^1)$.

Realistic Example:
A good example, mentioned in the text, involves a murder investigation.

- X: Suspect had **Motive** (x^1).
- Y: Suspect had **Opportunity** (y^1).

- Z: A **Murder** occurred (z^1).
- **Structure:** $X \to Z \leftarrow Y$.

Justification:

1. Both motive and opportunity increase the probability of a murder: $P(z^1|x^1) > P(z^1)$ and $P(z^1|y^1) > P(z^1)$.
2. If a murder occurs (z^1), our belief in both motive (x^1) and opportunity (y^1) increases.
3. If we then discover the suspect *also* had opportunity (y^1), our belief in their motive (x^1) increases *even more*. The two causes together form a much more compelling explanation for the murder than either one alone. Thus, $P(x^1|z^1) < P(x^1|y^1, z^1)$.

CPD Construction:
We need to construct the CPD $P(Z|X, Y)$ such that the "synergy" condition $P(z^1|x^1, y^1)$ is much greater than the individual contributions.

Let's define the CPD table for $P(z^1|X, Y)$:

X	Y	$P(z^1)$
x^0 (No Motive)	y^0 (No Opp.)	0.001
x^1 (Motive)	y^0 (No Opp.)	0.01
x^0 (No Motive)	y^1 (Opp.)	0.01
x^1 (Motive)	y^1 (Opp.)	**0.5**

The key is that $P(z^1|x^1, y^1) = 0.5$ is dramatically (synergistically) higher than $P(z^1|x^1, y^0) = 0.01$ or $P(z^1|x^0, y^1) = 0.01$. This CPD satisfies the required conditions.

Ex 3.5 (p.96)

Here is the qualitative comparison for each pair of queries, based on the network in Figure 3.14 and the provided rules .

(a) $P(t^1|d^1)$ **vs** $P(t^1)$

- **Active Trail:** $D \xrightarrow{-} C \xrightarrow{+} T$ (causal trail).
- **Reasoning:** d^1 (Good diet) makes c^1 (High cholesterol) *less* likely. Less likely c^1 makes t^1 (Test) *less* likely. The overall influence is negative.
- **Answer:** $P(t^1|d^1) < P(t^1)$.

(b) $P(d^1|t^0)$ **vs** $P(d^1)$

- **Active Trail:** $D \xrightarrow{-} C \xrightarrow{+} T$ (evidential trail).
- **Reasoning:** t^0 (Test is negative) makes c^1 (High cholesterol) *less* likely. The $D \xrightarrow{-} C$ link means d^1 makes c^1 *less* likely (and d^0 makes c^1 *more* likely). Therefore, c^0 (low cholesterol) is evidence for d^1 (good diet).
- **Answer:** $P(d^1|t^0) > P(d^1)$.

(c) $P(h^1|e^1, f^1)$ **vs** $P(h^1|e^1)$

- **Active Trail:** $H \to E \leftarrow F$ (v-structure, intercausal reasoning).

- **Reasoning:** We observe the effect e^1 (Exercise). h^1 (Health conscious) is a positive cause (+). f^1 (Little free time) is a negative cause (inhibitor, −). Observing e^1 *despite* the presence of the inhibitor f^1 requires a *stronger* explanation from the positive cause, h^1.
- **Answer:** $P(h^1|e^1, f^1) > P(h^1|e^1)$.

(d) $P(c^1|f^0)$ **vs** $P(c^1)$

- **Active Trail:** None.
- **Reasoning:** The trail $C \leftarrow D \leftarrow H \rightarrow E \leftarrow F$ is blocked at the unobserved v-structure E. The trail $C \leftarrow D \rightarrow W \leftarrow E \leftarrow F$ is blocked at the unobserved v-structure W. C and F are d-separated given the empty set.
- **Answer:** $P(c^1|f^0) = P(c^1)$.

(e) $P(c^1|h^0)$ **vs** $P(c^1)$

- **Active Trail:** $H \xrightarrow{+} D \xrightarrow{-} C$ (causal trail).
- **Reasoning:** h^0 (Not health conscious) makes d^1 (Good diet) *less* likely. d^0 makes c^1 (High cholesterol) *more* likely. The overall influence is positive.
- **Answer:** $P(c^1|h^0) > P(c^1)$.

(f) $P(c^1|h^0, f^0)$ **vs** $P(c^1|h^0)$

- **Active Trail:** None.
- **Reasoning:** We are testing if $(C \perp F|H)$. The trail $C \leftarrow D \leftarrow H \rightarrow E \leftarrow F$ is blocked by the observed node H (common cause). The trail $C \leftarrow D \rightarrow W \leftarrow E \leftarrow F$ is blocked by the unobserved v-structures W and E. C and F are d-separated given H.
- **Answer:** $P(c^1|h^0, f^0) = P(c^1|h^0)$.

(g) $P(d^1|h^1, e^0)$ **vs** $P(d^1|h^1)$

- **Active Trail:** None.
- **Reasoning:** We are testing if $(D \perp E|H)$. The trail $D \leftarrow H \rightarrow E$ is blocked by the observed common cause H. The trail $D \rightarrow W \leftarrow E$ is blocked by the unobserved v-structure W. D and E are d-separated given H.
- **Answer:** $P(d^1|h^1, e^0) = P(d^1|h^1)$.

(h) $P(d^1|e^1, f^0, w^1)$ **vs** $P(d^1|e^1, f^0)$

- **Active Trail:** Two trails are active between D and W given $\{e^1, f^0\}$.
  1. $D \xrightarrow{+} W$ (causal trail). The new evidence w^1 is a (positive) consequence of d^1. This trail provides *positive* influence for d^1.
  2. $D \rightarrow W \leftarrow E$ (v-structure). This trail is active because w^1 (the effect) is observed, and e^1 (an alternative cause) is also observed (in the conditioning set).
- **Reasoning:** The second trail is intercausal reasoning. We are given e^1 as an alternative cause for w^1. By the "explaining away" assumption, e^1 "explains away" w^1, which *reduces* the evidential support w^1 provides for d^1. This trail provides *negative* influence for d^1.
- **Answer:** The two active trails have conflicting influences. The net effect is **incomparable**.

(i) $P(t^1|w^1, f^0)$ **vs** $P(t^1|w^1)$

- **Active Trail:** $T \leftarrow C \leftarrow D \rightarrow W \leftarrow E \leftarrow F$.
- **Reasoning:** We are given w^1, which activates the v-structure $D \rightarrow W \leftarrow E$. The trail is now active. We are adding f^0.
  - f^0 (added evidence) $\implies$ e^1 is *more* likely (since $F \xrightarrow{-} E$).
  - e^1 is an alternative cause for w^1. By "explaining away", this makes d^1 *less* likely.
  - d^1 (Good diet) being *less* likely $\implies$ c^1 (High chol.) is *more* likely (since $D \xrightarrow{-} C$).
  - c^1 being *more* likely $\implies$ t^1 (Test) is *more* likely (since $C \xrightarrow{+} T$).

  The overall influence from f^0 to t^1 is positive.
- **Answer:** $P(t^1|w^1, f^0) > P(t^1|w^1)$.

Ex 3.6 (Page 96)

a. Upper bound on parameters in a BN with n nodes, l values each, and at most k parents.

- **BN Parameters:** A node X_i with k_i parents requires a conditional probability table (CPD). This CPD specifies a multinomial distribution over l values for each of the l^{k_i} possible assignments to its parents. A multinomial over l values has $l-1$ independent parameters.
  - Parameters for one node X_i: $(l-1) \cdot l^{k_i}$.
  - Since $k_i \leq k$, an upper bound for one node is $(l-1) \cdot l^k$.
  - For n nodes, a simple upper bound is $\mathbf{n \cdot (l-1) \cdot l^k}$.
- **Full Joint Parameters:** The full joint distribution over n variables with l values has l^n possible assignments. It requires $\mathbf{l^n - 1}$ independent parameters.

b. Parameters in a BN where X_i has parents $X_1, ..., X_{i-1}$.

- **BN Parameters:** We sum the parameters for each node:
  - X_1 (0 parents): $l-1$
  - X_2 (1 parent): $(l-1) \cdot l^1$
  - X_3 (2 parents): $(l-1) \cdot l^2$
  - ...
  - X_n ($n-1$ parents): $(l-1) \cdot l^{n-1}$
  - **Total:** $\sum_{i=0}^{n-1} (l-1) \cdot l^i = (l-1) \sum_{i=0}^{n-1} l^i$
  - This is a geometric series, which sums to $(l-1) \cdot \left(\frac{l^n-1}{l-1}\right) = \mathbf{l^n - 1}$.
- **Expressive Power:** This network has $\mathbf{l^n - 1}$ independent parameters, which is the *exact same number* as the full joint distribution. This network structure (a fully connected DAG) is an I-map for *any* distribution and can represent any joint distribution over these variables, as it does not encode any conditional independence assumptions.

c. Parameters in a naive Bayes model with class C (k values) and evidence $X_1, ..., X_n$ (l values).

- **Naive Bayes Parameters:** The model factorizes as $P(C)\prod_{i=1}^{n} P(X_i|C)$.
  - $P(C)$**:** A prior distribution over k values. This requires $\mathbf{k-1}$ independent parameters.
  - $P(X_i|C)$**:** For each evidence variable X_i, we need a CPD. This CPD is conditioned on k possible values of C. For each value c_j of C, $P(X_i|c_j)$ is a distribution over l values, requiring $l-1$ parameters.
  - Parameters for one X_i: $k \cdot (l-1)$.
  - Parameters for all n evidence variables: $n \cdot k \cdot (l-1)$.
  - **Total:** $(\mathbf{k-1}) + \mathbf{n \cdot k \cdot (l-1)}$. (The text gives $2n+1$ for the binary case $k=2, l=2$, which matches: $(2-1) + n \cdot 2 \cdot (2-1) = 1 + 2n$).
- **Full Joint Parameters:** The joint space includes $C, X_1, ..., X_n$.
  - Total assignments: $k \cdot l^n$.
  - Independent parameters: $\mathbf{k \cdot l^n - 1}$.

  The naive Bayes model's parameter count is **linear** in n, while the full joint's is **exponential** in n.

Ex 3.7 (page 96)

Let $\mathbf{x}_{-i}$ be the instantiation of all variables in the network *except* X_i. We want to compute $P(X_i|\mathbf{x}_{-i})$.

1. **Derivation:**
   By the definition of conditional probability, $P(X_i|\mathbf{x}_{-i}) = \frac{P(X_i,\mathbf{x}_{-i})}{P(\mathbf{x}_{-i})}$. The term $P(\mathbf{x}_{-i})$ is a normalization constant that does not depend on the value of X_i. We can write this as:

   $$[P(X_i|\mathbf{x}_{-i}) = \alpha \cdot P(X_i, \mathbf{x}_{-i})]$$

   The term $P(X_i, \mathbf{x}_{-i})$ is the probability of a full instantiation of the network, which can be computed using the chain rule for Bayesian networks:

   $$\left[P(X_i, \mathbf{x}_{-i}) = P(x_1, ..., x_n) = \prod_{j=1}^{n} P(x_j|Pa_{X_j}^{\mathcal{G}})\right]$$

   We can group this product into terms that depend on the value of X_i and terms that do not.
   - A term $P(x_j|Pa_{X_j})$ depends on X_i only if $j=i$ (the X_i term itself) or if X_i is a parent of X_j (i.e., X_j is a child of X_i).
   - All other terms are constant with respect to X_i and will be absorbed into the normalization constant α.

   Therefore, the computation simplifies to:

   $$\left[P(X_i|\mathbf{x}_{-i}) \propto P(X_i|Pa_{X_i}) \times \prod_{X_j \in Ch_{X_i}} P(x_j|Pa_{X_j})\right]$$

This shows that the probability of X_i given all other variables depends only on the values of its **Markov blanket**: its parents (Pa_{X_i}), its children (Ch_{X_i}), and its children's other parents (which are part of the Pa_{X_j} terms).

2. **Procedure:**
   To compute the distribution $P(X_i|\mathbf{x}_{-i})$:
   a. Initialize an unnormalized probability $U(x_i')$ for each value $x_i' \in Val(X_i)$.
   b. For each value $x_i' \in Val(X_i)$:
      (i) Find the probability $p_1 = P(X_i = x_i'|\mathbf{pa}_{X_i})$ from X_i's CPD, using the given values for X_i's parents from $\mathbf{x}_{-i}$.
      (ii) Initialize $p_2 = 1.0$.
      (iii) For each child $X_j \in Ch_{X_i}$:
         1. Get the given value x_j from $\mathbf{x}_{-i}$.
         2. Form the full parent assignment $\mathbf{pa}_{X_j}$ for X_j using $X_i = x_i'$ and the given values of X_j's other parents from $\mathbf{x}_{-i}$.
         3. Find the probability $p_{child} = P(x_j|\mathbf{pa}_{X_j})$ from X_j's CPD.
         4. $p_2 = p_2 \times p_{child}$.
      (iv) Set $U(x_i') = p_1 \times p_2$.
   c. Calculate the normalization constant $Z = \sum_{x_i' \in Val(X_i)} U(x_i')$.
   d. The final distribution is $P(X_i = x_i'|\mathbf{x}_{-i}) = U(x_i')/Z$ for all x_i'.

3. **Computational Complexity:**
   This procedure avoids constructing the joint distribution.
   - Let l be the maximum number of values for any variable.
   - Let d_{out} be the number of children of X_i.
   - Let k be the maximum number of parents for any node.

   The main loop (b) runs l times (once for each value of X_i).
   - Inside the loop, step (i) is one CPD lookup, which takes $O(k)$ time (to index into the table).
   - Step (iii) loops d_{out} times. Each lookup inside this loop (iii.3) also takes $O(k)$ time.

   The total complexity is $O(l \times (k + d_{out} \times k))$, which simplifies to $O(l \cdot k \cdot d_{out})$. This is highly efficient as it only depends on the local graph structure around X_i (its Markov blanket), not the exponential size of the full joint distribution.

Ex 3.8 (page 96)

1. **Setup:**
   - Let $\mathcal{B}$ be the network over all variables $\mathcal{X}$.
   - Let $\mathbf{Z} \subseteq \mathcal{X}$ be a set of variables.

- Let $\mathbf{X} \subseteq \mathcal{X}$ be the set of all ancestors of nodes in $\mathbf{Z}$. Note that $\mathbf{Z} \subseteq \mathbf{X}$.
- Let $\mathbf{Y} = \mathcal{X} - \mathbf{X}$ be the set of "barren nodes." These are all nodes in the graph that are *not* ancestors of any node in $\mathbf{Z}$.
- Let $\mathcal{B}'$ be the network over the nodes $\mathbf{X}$ using the induced subgraph and the original CPDs for nodes in $\mathbf{X}$.

2. **Goal:** Prove $P_{\mathcal{B}}(\mathbf{X}) = P_{\mathcal{B}'}(\mathbf{X})$.

3. **Proof:**

   a. First, consider the marginal distribution $P_{\mathcal{B}}(\mathbf{X})$ from the full network. We get this by marginalizing out the barren nodes $\mathbf{Y}$:

   $$\left[P_{\mathcal{B}}(\mathbf{X}) = \sum_{\mathbf{y}} P_{\mathcal{B}}(\mathbf{X}, \mathbf{y}) \right]$$

   b. Using the chain rule for $\mathcal{B}$, we split the joint into terms for $\mathbf{X}$ and $\mathbf{Y}$:

   $$\left[P_{\mathcal{B}}(\mathbf{X}, \mathbf{y}) = \prod_{X_i \in \mathbf{X}} P(x_i | Pa_{X_i}) \times \prod_{Y_j \in \mathbf{Y}} P(y_j | Pa_{Y_j}) \right]$$

   c. Substitute this back into the summation:

   $$\left[P_{\mathcal{B}}(\mathbf{X}) = \sum_{\mathbf{y}} \left[\prod_{X_i \in \mathbf{X}} P(x_i | Pa_{X_i}) \times \prod_{Y_j \in \mathbf{Y}} P(y_j | Pa_{Y_j}) \right] \right]$$

   d. **Key Property 1:** Can a barren node $Y_j \in \mathbf{Y}$ be a parent of a node $X_i \in \mathbf{X}$?
      - If $Y_j \in Pa_{X_i}$, then Y_j is an ancestor of X_i.
      - Since X_i is an ancestor of $\mathbf{Z}$ (by definition of $\mathbf{X}$), Y_j must also be an ancestor of $\mathbf{Z}$.
      - This would mean $Y_j \in \mathbf{X}$, which contradicts $Y_j \in \mathbf{Y}$.
      - Therefore, no node in $\mathbf{Y}$ can be a parent of any node in $\mathbf{X}$. All parents of nodes in $\mathbf{X}$ must also be in $\mathbf{X}$ (i.e., $Pa_{X_i} \subseteq \mathbf{X}$).

   e. Because of this property, the first product $\prod_{X_i \in \mathbf{X}} P(x_i | Pa_{X_i})$ does not depend on any variable in $\mathbf{y}$. We can pull it out of the summation:

   $$\left[P_{\mathcal{B}}(\mathbf{X}) = \left[\prod_{X_i \in \mathbf{X}} P(x_i | Pa_{X_i}) \right] \times \sum_{\mathbf{y}} \left[\prod_{Y_j \in \mathbf{Y}} P(y_j | Pa_{Y_j}) \right] \right]$$

   f. **Key Property 2:** Consider the term $\sum_{\mathbf{y}} \prod_{Y_j \in \mathbf{Y}} P(y_j | Pa_{Y_j})$.
      - The nodes $\mathbf{Y}$ are all non-ancestors of $\mathbf{Z}$. This means they cannot have descendants in $\mathbf{Z}$.
      - Can a node $Y_j \in \mathbf{Y}$ have a child $X_i \in \mathbf{X}$? No, this was proven in (d).
      - Therefore, all children of nodes in $\mathbf{Y}$ must also be in $\mathbf{Y}$.

- This means the set $\mathbf{Y}$ is "self-contained" with respect to descendants. We can find a topological ordering $Y_1, ..., Y_m$ of the nodes in $\mathbf{Y}$ such that $Pa_{Y_j} \subseteq \mathbf{X} \cup \{Y_1, ..., Y_{j-1}\}$.
- We can perform the summation $\sum_{\mathbf{y}}$ by summing out the nodes in $\mathbf{Y}$ in reverse topological order $(Y_m, ..., Y_1)$.
- $\sum_{y_1} ... \sum_{y_m} \prod_{j=1}^{m} P(y_j | Pa_{Y_j}) = \sum_{y_1} ... \sum_{y_{m-1}} \left[\prod_{j=1}^{m-1} P(y_j | Pa_{Y_j})\right] \times \sum_{y_m} P(y_m | Pa_{Y_m})$
- Since $P(Y_m | Pa_{Y_m})$ is a valid CPD, $\sum_{y_m} P(y_m | Pa_{Y_m}) = 1$ for any assignment to its parents.
- This process repeats, and the entire summation over $\mathbf{y}$ evaluates to 1.

g. Substituting this back into the equation from (e):

$$\left[P_{\mathcal{B}}(\mathbf{X}) = \left[\prod_{X_i \in \mathbf{X}} P(x_i | Pa_{X_i}) \right] \times 1 \right]$$

h. Now, consider the distribution $P_{\mathcal{B}'}(\mathbf{X})$. By definition, $\mathcal{B}'$ is a network over $\mathbf{X}$ with the *same CPDs*. The parents of any $X_i \in \mathbf{X}$ in $\mathcal{B}'$ are the same as in $\mathcal{B}$ (as shown in (d)). By the chain rule for $\mathcal{B}'$:

$$\left[P_{\mathcal{B}'}(\mathbf{X}) = \prod_{X_i \in \mathbf{X}} P(x_i | Pa_{X_i}) \right]$$

i. Comparing (g) and (h), we see that $P_{\mathcal{B}}(\mathbf{X}) = P_{\mathcal{B}'}(\mathbf{X})$.

Ex 3.9 (page 96)

1. **Goal:** We are given that P factorizes over $\mathcal{G}$: $P(X_1, ..., X_n) = \prod_{j=1}^{n} P(X_j | Pa_{X_j}^{\mathcal{G}})$. We must prove that $\mathcal{G}$ is an I-map for P, which means we must show that P satisfies the *local independencies* $\mathcal{I}_l(\mathcal{G})$.
2. **Local Independencies:** The local independencies $\mathcal{I}_l(\mathcal{G})$ state that for each variable X_i, X_i is conditionally independent of its non-descendants given its parents: $(X_i \perp \text{NonDescendants}_{X_i} | Pa_{X_i}^{\mathcal{G}})$.
3. **Proof:**
   a. Let X_i be an arbitrary variable. Let $\mathbf{Pa}_i = Pa_{X_i}^{\mathcal{G}}$.
   b. Let $\mathbf{ND}_i = \text{NonDescendants}_{X_i}$. Since the graph is acyclic, $\mathbf{Pa}_i \subseteq \mathbf{ND}_i$.
   c. Let $\mathbf{Z} = \mathbf{ND}_i - \mathbf{Pa}_i$. These are the non-descendants of X_i that are *not* its parents. The local independency assertion is $(X_i \perp \mathbf{Z} | \mathbf{Pa}_i)$.
   d. We need to prove that $P(X_i | \mathbf{Z}, \mathbf{Pa}_i) = P(X_i | \mathbf{Pa}_i)$.
   e. By definition of conditional probability:

$$\left[P(X_i | \mathbf{Z}, \mathbf{Pa}_i) = \frac{P(X_i, \mathbf{Z}, \mathbf{Pa}_i)}{P(\mathbf{Z}, \mathbf{Pa}_i)} \right]$$

   f. Let $\mathbf{W} = \{X_i\} \cup \mathbf{Z} \cup \mathbf{Pa}_i = \{X_i\} \cup \mathbf{ND}_i$. Let $\mathbf{D} = \text{Descendants}_{X_i}$. The set $\mathbf{W} \cup \mathbf{D}$ is all variables $\mathcal{X}$.

g. Let's compute the numerator $P(X_i, \mathbf{Z}, \mathbf{Pa}_i) = P(\mathbf{W})$. We get this by marginalizing the full joint $P(\mathcal{X})$ over the descendants $\mathbf{D}$:

$$\left[P(\mathbf{W}) = \sum_{\mathbf{d}} P(\mathbf{W}, \mathbf{d}) = \sum_{\mathbf{d}} \left[\prod_{W_k \in \mathbf{W}} P(w_k | Pa_{W_k}) \times \prod_{D_j \in \mathbf{D}} P(d_j | Pa_{D_j})\right]\right]$$

h. The parents of any node $W_k \in \mathbf{W}$ (a non-descendant of X_i) cannot be in $\mathbf{D}$ (descendants of X_i). If $D_j \in Pa_{W_k}$, then D_j would be an ancestor of W_k, which is impossible.

i. Therefore, $Pa_{W_k} \subseteq \mathbf{W}$. The first product does not depend on $\mathbf{d}$ and can be pulled out:

$$\left[P(\mathbf{W}) = \left[\prod_{W_k \in \mathbf{W}} P(w_k | Pa_{W_k})\right] \times \sum_{\mathbf{d}} \left[\prod_{D_j \in \mathbf{D}} P(d_j | Pa_{D_j})\right]\right]$$

j. As shown in Exercise 3.8 (step 3f), the summation term $\sum_{\mathbf{d}} \prod_{D_j \in \mathbf{D}} P(d_j | Pa_{D_j})$ evaluates to 1.

k. So, $P(\mathbf{W}) = P(X_i, \mathbf{Z}, \mathbf{Pa}_i) = \prod_{W_k \in \mathbf{W}} P(w_k | Pa_{W_k})$. We can separate the X_i term:

$$\left[P(X_i, \mathbf{Z}, \mathbf{Pa}_i) = P(X_i | \mathbf{Pa}_i) \times \prod_{W_k \in \mathbf{Z} \cup \mathbf{Pa}_i} P(w_k | Pa_{W_k})\right]$$

l. Now let's compute the denominator $P(\mathbf{Z}, \mathbf{Pa}_i)$ by marginalizing $P(\mathbf{W})$ over X_i:

$$P(\mathbf{Z}, \mathbf{Pa}_i) = \sum_{x_i' \in Val(X_i)} P(X_i = x_i', \mathbf{Z}, \mathbf{Pa}_i)$$

$$P(\mathbf{Z}, \mathbf{Pa}_i) = \sum_{x_i'} \left[P(X_i = x_i' | \mathbf{Pa}_i) \times \prod_{W_k \in \mathbf{Z} \cup \mathbf{Pa}_i} P(w_k | Pa_{W_k})\right]$$

m. The product term $\prod_{W_k \in \mathbf{Z} \cup \mathbf{Pa}_i} P(w_k | Pa_{W_k})$ does not depend on x_i', so we pull it out:

$$\left[P(\mathbf{Z}, \mathbf{Pa}_i) = \left[\prod_{W_k \in \mathbf{Z} \cup \mathbf{Pa}_i} P(w_k | Pa_{W_k})\right] \times \sum_{x_i'} P(X_i = x_i' | \mathbf{Pa}_i)\right]$$

n. The sum $\sum_{x_i'} P(X_i = x_i' | \mathbf{Pa}_i)$ is 1, as it's a sum over a valid CPD.

$$\left[P(\mathbf{Z}, \mathbf{Pa}_i) = \prod_{W_k \in \mathbf{Z} \cup \mathbf{Pa}_i} P(w_k | Pa_{W_k})\right]$$

o. Finally, we compute the conditional probability from (e):

$$P(X_i | \mathbf{Z}, \mathbf{Pa}_i) = \frac{P(X_i | \mathbf{Pa}_i) \times \prod_{W_k \in \mathbf{Z} \cup \mathbf{Pa}_i} P(w_k | Pa_{W_k})}{\prod_{W_k \in \mathbf{Z} \cup \mathbf{Pa}_i} P(w_k | Pa_{W_k})}$$

$$P(X_i | \mathbf{Z}, \mathbf{Pa}_i) = P(X_i | \mathbf{Pa}_i)$$

p. This proves $(X_i \perp \mathbf{Z} | \mathbf{Pa}_i)$, which is the local independency for X_i. Since X_i was arbitrary, this holds for all nodes. Therefore, P satisfies $\mathcal{I}_l(\mathcal{G})$ and $\mathcal{G}$ is an I-map for P.

Ex 3.10 (page 96)

1. **Goal:** We need to prove that for any node X_i, X_i is d-separated from any node in $\text{NonDescendants}_{X_i} - Pa_{X_i}$ given the set $\mathbf{Z} = Pa_{X_i}^{\mathcal{G}}$.
2. **Proof:**
   a. Let X_i be an arbitrary node and $\mathbf{Z} = Pa_{X_i}$ be the conditioning set.
   b. Let N be an arbitrary non-descendant of X_i ($N \in \text{NonDescendants}_{X_i}$) such that $N \notin \mathbf{Z}$.
   c. We must show that *every* trail between X_i and N is **inactive** (blocked) given $\mathbf{Z}$.
   d. Consider any trail between X_i and N. This trail must start with an arrow either leaving X_i (to a child) or entering X_i (from a parent).
   e. **Case 1: The trail starts with an arrow entering X_i.**
      - The trail looks like $X_i \leftarrow P \rightleftharpoons \cdots \rightleftharpoons N$.
      - The node P must be a parent of X_i, so $P \in Pa_{X_i}$.
      - P is in the conditioning set $\mathbf{Z}$.
      - The node P is part of a two-edge segment on the trail. It is *not* a v-structure, because the arrow $X_i \leftarrow P$ points *into* X_i. The segment is either $X_i \leftarrow P \rightarrow \ldots$ (common cause) or $X_i \leftarrow P \leftarrow \ldots$ (evidential trail).
      - For both of these structures, the trail is active *if and only if* the middle node (P) is **not** observed.
      - Since $P \in \mathbf{Z}$, P *is* observed. Therefore, the trail is **blocked** at P.
   f. **Case 2: The trail starts with an arrow leaving X_i.**
      - The trail looks like $X_i \rightarrow Y \rightleftharpoons \cdots \rightleftharpoons N$.
      - The node Y must be a child of X_i, and therefore Y is a descendant of X_i.
      - The node N is a non-descendant of X_i.
      - Since the trail starts at a descendant (Y) and ends at a non-descendant (N), there *must* be a v-structure somewhere along the trail $Y \rightleftharpoons \cdots \rightleftharpoons N$. If there were no v-structures, the trail would be a simple chain, making N a descendant of Y and thus of X_i, which is a contradiction.
      - Let $W \rightarrow V \leftarrow U$ be the first v-structure on the trail after X_i. The trail looks like $X_i \rightarrow Y \rightarrow \cdots \rightarrow W \rightarrow V \leftarrow U \rightleftharpoons \cdots \rightleftharpoons N$.
      - For this trail to be active, the v-structure at V must be *activated*. This means V or one of V's descendants must be in the conditioning set $\mathbf{Z} = Pa_{X_i}$.
      - But V is a descendant of Y, which is a descendant of X_i. All of V's descendants are also descendants of X_i.
      - The conditioning set $\mathbf{Z} = Pa_{X_i}$ contains only parents of X_i, which are *non-descendants* of X_i.
      - Therefore, neither V nor any of its descendants can be in $\mathbf{Z}$.
      - The v-structure at V is not activated, and the trail is **blocked**.

g. **Conclusion:** In both possible cases, any trail from X_i to a non-descendant N is blocked by the parent set Pa_{X_i}. Therefore, $d\text{-sep}_{\mathcal{G}}(X_i; N|Pa_{X_i})$ holds, and the local independency is implied by d-separation.

Exercise 3.11

a. Construct a Bayesian network $\mathcal{B}'$ over all of the nodes except for `Alarm` that is a minimal I-map for the marginal distribution $P_{\mathcal{B}}(B, E, T, N, J, M)$.

To find the minimal I-map for the marginal distribution, we use a procedure based on a given variable ordering. Let's use the ordering: **B, E, T, N, J, M**. We test for conditional independencies in the original distribution $P_{\mathcal{B}}$ (represented by the graph in Figure 3.15). When we marginalize out `Alarm`, its parents (B, E) and its children (J, M) become coupled.

1. **Add B (Burglary):** $Pa(B) = \emptyset$.
2. **Add E (Earthquake):** We test $(E \perp B)$. In the original graph, the trail $E \rightarrow Alarm \leftarrow B$ is a v-structure. Marginalizing `Alarm` activates this path, making B and E dependent. Thus, $(E \perp B)$ does not hold. We must add an edge, e.g., $B \rightarrow E$. $Pa(E) = \{B\}$.
3. **Add T (TV):** We test $(T \perp \{B, E\})$. In the original graph, T is d-separated from B and E. The trail $T \rightarrow J \leftarrow Alarm \leftarrow B$ is blocked at the v-structure at J (since J is not observed). Thus, $(T \perp \{B, E\})$ holds. $Pa(T) = \emptyset$.
4. **Add N (Nap):** We test $(N \perp \{B, E, T\})$. N is d-separated from B, E, and T in the original graph. For example, $N \rightarrow M \leftarrow Alarm \leftarrow B$ is blocked at M. Thus, $(N \perp \{B, E, T\})$ holds. $Pa(N) = \emptyset$.
5. **Add J (JohnCall):** We test for parents from $\{B, E, T, N\}$.
   - The trail $J \leftarrow T$ is active. **T** is a parent.
   - The trail $J \leftarrow Alarm \leftarrow B$ is active (since `Alarm` is marginalized). **B** is a parent.
   - The trail $J \leftarrow Alarm \leftarrow E$ is active. **E** is a parent.

   Thus, $Pa(J) = \{B, E, T\}$.
6. **Add M (MaryCall):** We test for parents from $\{B, E, T, N, J\}$.
   - The trail $M \leftarrow N$ is active. **N** is a parent.
   - The trail $M \leftarrow Alarm \leftarrow B$ is active. **B** is a parent.
   - The trail $M \leftarrow Alarm \leftarrow E$ is active. **E** is a parent.
   - The trail $M \leftarrow Alarm \rightarrow J$ is active. **J** is a parent.
   - We test $(M \perp T|\{B, E, N, J\})$. The trail $M \leftarrow Alarm \rightarrow J \leftarrow T$ is active given J (v-structure at J is activated). **T** is a parent.

   Thus, $Pa(M) = \{B, E, T, N, J\}$.

The resulting minimal I-map $\mathcal{B}'$ for this ordering is:

- **Nodes:** {B, E, T, N, J, M}
- **Parents:**

- $Pa(B) = \emptyset$
- $Pa(E) = \{B\}$
- $Pa(T) = \emptyset$
- $Pa(N) = \emptyset$
- $Pa(J) = \{B, E, T\}$
- $Pa(M) = \{B, E, T, N, J\}$

b. Generalize the procedure... into a node elimination algorithm.

Here is a general algorithm to transform a graph $\mathcal{G}$ into a new graph $\mathcal{G}'$ that is an I-map for the marginal distribution over $\mathcal{X} \setminus \{X_i\}$.

Algorithm: Eliminate-Node($\mathcal{G}$, X_i)

1. **Identify Family:** Let Pa_i be the set of parents of X_i in $\mathcal{G}$, and let Ch_i be the set of children of X_i in $\mathcal{G}$.
2. **Initialize New Graph:** Create a new graph $\mathcal{G}'$ containing all nodes in $\mathcal{G}$ except X_i, and all edges from $\mathcal{G}$ that are not incident to X_i.
3. **Moralize and Connect:** Create a temporary "clique" set $C = Pa_i \cup Ch_i$.
4. **Add Edges:** For every pair of distinct nodes $U, V \in C$, add an undirected edge (U, V) to $\mathcal{G}'$ (if one does not already exist).
5. **Orient Edges (Optional):** To make $\mathcal{G}'$ a directed acyclic graph (DAG), orient all the newly added edges in a way that is consistent with a topological ordering (e.g., the original ordering of the nodes in $\mathcal{X} \setminus \{X_i\}$).
6. **Return:** The resulting graph $\mathcal{G}'$.

Exercise 3.12

a. Reverse the arc $B \to A$. What additional minimal modifications... are needed?

We want to reverse $B \to A$ to $A \to B$. Original parents:

- $Pa_B = \emptyset$
- $Pa_A = \{B, E\}$

The transformation is based on rewriting the factors: $P(B)P(A|B, E) = P(A|Pa'_A)P(B|Pa'_B)$

The new parent sets are:

1. **New Parents of A:** $Pa'_A = Pa_B \cup (Pa_A \setminus \{B\}) = \emptyset \cup \{E\} = \{E\}$.
2. **New Parents of B:** $Pa'_B = \{A\} \cup Pa_B \cup (Pa_A \setminus \{B\}) = \{A\} \cup \emptyset \cup \{E\} = \{A, E\}$.

The minimal modifications are:

1. **Reverse** the edge $B \to A$ to become $A \to B$.
2. **Remove** the edge $E \to A$.

3. **Add** the edge $E \to B$.

b. Define a general procedure for reversing the arc $X \to Y$.

To reverse an arc $X \to Y$ to $Y \to X$ and create a new graph $\mathcal{G}'$ that is an I-map for $P_{\mathcal{G}}$:

1. **Reverse Arc:** In $\mathcal{G}$, remove $X \to Y$ and add $Y \to X$.
2. **Update Y's Parents:** $Pa_Y^{\mathcal{G}'} = Pa_X^{\mathcal{G}} \cup (Pa_Y^{\mathcal{G}} \setminus \{X\})$. Y inherits all of X's original parents.
3. **Update X's Parents:** $Pa_X^{\mathcal{G}'} = \{Y\} \cup Pa_X^{\mathcal{G}} \cup (Pa_Y^{\mathcal{G}} \setminus \{X\})$. X gains Y as a parent and also inherits all of Y's original parents (except X).

This is based on rewriting the joint probability: $P(X, Y|Pa_X, Pa_Y \setminus \{X\}) = P(Y|X, Pa_Y \setminus \{X\})P(X|Pa_X)$ Using Bayes' rule, this becomes: $= P(X|Y, Pa_X, Pa_Y \setminus \{X\}) \cdot P(Y|Pa_X, Pa_Y \setminus \{X\})$ This gives the new CPDs and parent sets.

c. Are we guaranteed that the final network structure is equivalent to the original ($\mathcal{G} = \mathcal{G}''$)?

No, we are not guaranteed that $\mathcal{G} = \mathcal{G}''$.

Let's trace the operations from part (a):

1. **Start ($\mathcal{G}$):**
   - $Pa_A = \{B, E\}$
   - $Pa_B = \emptyset$
2. **Reverse $B \to A$ to get $\mathcal{G}'$:**
   - $Pa_A^{\mathcal{G}'} = \{E\}$
   - $Pa_B^{\mathcal{G}'} = \{A, E\}$
3. **Reverse $A \to B$ back to get $\mathcal{G}''$:**
   - We reverse the arc $A \to B$ in $\mathcal{G}'$. Let $X = A$ and $Y = B$.
   - $Pa_X^{\mathcal{G}'} = Pa_A^{\mathcal{G}'} = \{E\}$
   - $Pa_Y^{\mathcal{G}'} = Pa_B^{\mathcal{G}'} = \{A, E\}$
   - Apply the reversal procedure from (b):
   - **New Parents of A (X):** $Pa_A^{\mathcal{G}''} = \{Y\} \cup Pa_X^{\mathcal{G}'} \cup (Pa_Y^{\mathcal{G}'} \setminus \{X\}) = \{B\} \cup \{E\} \cup (\{A, E\} \setminus \{A\}) = \{B, E\}$.
   - **New Parents of B (Y):** $Pa_B^{\mathcal{G}''} = Pa_X^{\mathcal{G}'} \cup (Pa_Y^{\mathcal{G}'} \setminus \{X\}) = \{E\} \cup \{E\} = \{E\}$.

Comparing the final graph $\mathcal{G}''$ to the original $\mathcal{G}$:

- **$\mathcal{G}$:** $Pa_A = \{B, E\}$, $Pa_B = \emptyset$.
- **$\mathcal{G}''$:** $Pa_A = \{B, E\}$, $Pa_B = \{E\}$.

The graphs are **not equal**. $\mathcal{G}''$ contains an extra edge $E \to B$ that was not in $\mathcal{G}$.

Exercise 3.13

Show how $(X \perp Y|Z)$ translates into a set of polynomial equalities over the CPD parameters $\theta_{x|u}$.

1. **Definition of Independence:** The statement $(X \perp Y|Z)$ is equivalent to $P(X, Y|Z) = P(X|Z)P(Y|Z)$ for all instantiations x, y, z.
2. **Equivalent Form:** Assuming positive probabilities, this is equivalent to:

$$P(X = x, Y = y, Z = z) \cdot P(Z = z) = P(X = x, Z = z) \cdot P(Y = y, Z = z)$$

3. **Probabilities as Polynomials:** In a BN, the joint probability of any full instantiation $\xi = (x_1, \ldots, x_n)$ is a product (a monomial) of the θ parameters:

$$P(\xi) = \prod_{i=1}^{n} P(x_i|pa_{X_i}) = \prod_{i=1}^{n} \theta_{x_i|pa_{X_i}}$$

4. Any marginal probability is a sum of these joint probabilities, which is a sum of products of θ parameters, i.e., a **polynomial**.

$$P(x, y, z) = \sum_{w \in Val(W)} P(x, y, z, w) = \sum_{w \in Val(W)} \left(\prod_{V \in \mathcal{X}} \theta_{v|pa_V} \right)$$

5. **The Equality:** Let $f_{x,y,z}(\Theta)$ be the polynomial for $P(X = x, Y = y, Z = z)$. Then:
   - $P(Y = y, Z = z)$ is the polynomial $g_{y,z}(\Theta) = \sum_{x'} f_{x',y,z}(\Theta)$.
   - $P(X = x, Z = z)$ is the polynomial $h_{x,z}(\Theta) = \sum_{y'} f_{x,y',z}(\Theta)$.
   - $P(Z = z)$ is the polynomial $k_z(\Theta) = \sum_{x'} h_{x',z}(\Theta)$.
6. **Conclusion:** The conditional independence statement $(X \perp Y|Z)$ translates into a set of polynomial equalities, one for each (x, y, z):

$$f_{x,y,z}(\Theta) \cdot k_z(\Theta) - h_{x,z}(\Theta) \cdot g_{y,z}(\Theta) = 0$$

   This is a set of polynomial constraints on the parameters Θ.

Exercise 3.14

Prove Theorem 3.6: The algorithm `Reachable` (Alg 3.1) returns the set of all nodes reachable from X via active trails.

Let A be the set of nodes computed in Phase I. $A = \{Y|Y \in Z$ or Y is an ancestor of some $W \in Z\}$. This set A correctly identifies all nodes that can "activate" a v-structure. Let R be the set of nodes returned by Phase II.

1. Soundness (If $Y \in R$, there is an active trail from X to Y given Z)

We prove by induction that any pair (Y, d) added to L corresponds to a node Y that is the endpoint of an active trail from X.

- **Base Case:** L is initialized with $(X, \dagger)$. The trail X is a trivial active trail to X.
- **Inductive Step:** Assume (Y, d) was popped from L, corresponding to an active trail $T_{X \rightsquigarrow Y}$. Y is added to R if $Y \notin Z$. If $Y \notin Z$, $T_{X \rightsquigarrow Y}$ is an active trail *to* Y, so Y is reachable.

  Now consider the neighbors added to L:

  - **Case 1:** $d = \dagger$ **and** $Y \notin Z$**.** (Arrived from below, Y not observed).
    - Add $(P, \dagger)$ for $P \in Pa_Y$: The trail $T_{X \rightsquigarrow Y \leftarrow P}$ is active because Y is a non-observed, non-v-structure node.
    - Add $(C, \perp)$ for $C \in Ch_Y$: The trail $T_{X \rightsquigarrow Y \rightarrow C}$ is active because Y is a non-observed, non-v-structure node.
  - **Case 2:** $d = \perp$**.** (Arrived from above).
    - **Subcase 2a:** $Y \notin Z$**.** Add $(C, \perp)$ for $C \in Ch_Y$: The trail $T_{X \rightsquigarrow Y \rightarrow C}$ is active because Y is a non-observed, non-v-structure node.
    - **Subcase 2b:** $Y \in A$**.** Add $(P, \dagger)$ for $P \in Pa_Y$: The trail ... $\rightarrow Y \leftarrow P$ is a v-structure. It is **active** because $Y \in A$ (meaning Y or its descendant is in Z).

  In all cases, any new pair added to L represents an active trail. When Y is added to R, it's because $Y \notin Z$, guaranteeing it's reachable by an active trail.

2. Completeness (If an active trail $X \rightleftharpoons \cdots \rightleftharpoons Y$ exists, $Y \in R$)

Let $T = (X_1 \rightleftharpoons X_2 \rightleftharpoons \cdots \rightleftharpoons X_k)$ be an active trail, with $X_1 = X$ and $X_k = Y$. We prove by induction on i that (X_i, d_i) will be added to L, where d_i is the direction of arrival.

- **Base Case ($i = 1$):** $(X_1, \dagger) = (X, \dagger)$ is added to L.
- **Inductive Step:** Assume (X_i, d_i) is added to L. Since T is active, $X_i \notin Z$ (for $i < k$). When (X_i, d_i) is popped, we show (X_{i+1}, d_{i+1}) is added:
  - **Case 1:** $X_{i-1} \rightarrow X_i \rightarrow X_{i+1}$ (Serial) $d_i = \perp$. $X_i \notin Z$. Line 30 adds $(X_{i+1}, \perp)$.
  - **Case 2:** $X_{i-1} \leftarrow X_i \leftarrow X_{i+1}$ (Serial) $d_i = \dagger$. $X_i \notin Z$. Line 24 adds $(X_{i+1}, \dagger)$.
  - **Case 3:** $X_{i-1} \leftarrow X_i \rightarrow X_{i+1}$ (Common Cause) $d_i = \dagger$. $X_i \notin Z$. Line 26 adds $(X_{i+1}, \perp)$.
  - **Case 4:** $X_{i-1} \rightarrow X_i \leftarrow X_{i+1}$ (v-structure) $d_i = \perp$. Since trail is active, $X_i \in A$. Line 34 adds $(X_{i+1}, \dagger)$.

  By induction, $(X_k, d_k) = (Y, d_k)$ will be added to L. When it's popped, since $Y \notin Z$ (by definition of active trail end), Y will be added to R.

Exercise 3.15

We use **Theorem 3.8**: Two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ are I-equivalent if and only if they have the same **skeleton** and the same set of **immoralities** (v-structures $X \rightarrow Z \leftarrow Y$ where X and Y are not adjacent).

Network (a): $A \rightarrow B \leftarrow C, B \rightarrow D$

- **Skeleton:** A-B, B-C, B-D.

- **v-structures:** $A \to B \leftarrow C$.
- **Immoralities:** $A \to B \leftarrow C$ is an immorality (no edge A-C). This is the only immorality.

Any I-equivalent graph $\mathcal{G}'$ must have the same skeleton and this single immorality.

1. The immorality $A \to B \leftarrow C$ forces the edges $A \to B$ and $C \to B$.
2. This leaves the edge B-D. Can it be $D \to B$?
3. Let's test $\mathcal{G}' = (A \to B \leftarrow C, D \to B)$.
    - **Skeleton:** Same.
    - **Immoralities:** $A \to B \leftarrow C$, $A \to B \leftarrow D$, $C \to B \leftarrow D$.
4. $\mathcal{G}'$ has a different set of immoralities than (a). It is not I-equivalent.

The orientation of B-D is forced to be $B \to D$ (to avoid creating new immoralities). Since all edges have a forced orientation, **no other Bayesian network can be I-equivalent to (a).**

Network (b): $A \to B, B \to C, B \to D$

- **Skeleton:** A-B, B-C, B-D.
- **v-structures:** None.
- **Immoralities:** None.

Yes, other I-equivalent networks exist. Any graph $\mathcal{G}'$ with the same skeleton (A-B, B-C, B-D) and no immoralities is I-equivalent.

Example $\mathcal{G}'$: $B \to A, B \to C, B \to D$

- **Skeleton:** A-B, B-C, B-D. (Same)
- **Immoralities:** None.

Since $\mathcal{G}'$ has the same skeleton and the same (empty) set of immoralities, $\mathcal{G}'$ **is I-equivalent to (b).**

Exercise 3.16

Prove Theorem 3.7: Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two graphs over $\mathcal{X}$. If $\mathcal{G}_1$ and $\mathcal{G}_2$ have the same skeleton and the same set of v-structures then they are I-equivalent.

Proof. To prove I-equivalence, we must show that $\mathcal{I}(\mathcal{G}_1) = \mathcal{I}(\mathcal{G}_2)$. This means that for any three disjoint sets of nodes $X, Y, Z \subseteq \mathcal{X}$, $d - sep_{\mathcal{G}_1}(X; Y|Z)$ if and only if $d - sep_{\mathcal{G}_2}(X; Y|Z)$. This holds if and only if every trail T between a node $x \in X$ and $y \in Y$ is active given Z in $\mathcal{G}_1$ iff it is also active given Z in $\mathcal{G}_2$.

1. **Skeletons:** Since $\mathcal{G}_1$ and $\mathcal{G}_2$ have the same skeleton, they have the exact same set of trails. Let T be an arbitrary trail between x and y.
2. **Active Trail Definition:** A trail T is active given Z if:
    - (a) Every *non-collider* (non-v-structure) node W on T is *not* in Z ($W \notin Z$).

- (b) Every *collider* (v-structure) node C on T is "activated," meaning $C \in Z$ or one of C's descendants is in Z ($Desc(C) \cap Z \neq \emptyset$).

3. **Applying the Premise:**
   - By premise, $\mathcal{G}_1$ and $\mathcal{G}_2$ have the same set of v-structures.
   - This means that for the trail T, a node W is a non-collider in $\mathcal{G}_1$ if and only if it is a non-collider in $\mathcal{G}_2$.
   - Similarly, a node C is a collider in $\mathcal{G}_1$ if and only if it is a collider in $\mathcal{G}_2$.
   - Therefore, the set of nodes on T to which condition (a) applies is identical for $\mathcal{G}_1$ and $\mathcal{G}_2$.
   - The set of nodes on T to which condition (b) applies is identical for $\mathcal{G}_1$ and $\mathcal{G}_2$.
4. **Analyzing Potential Differences:**
   - Condition (a) is identically met or violated in both graphs, as it only depends on T and Z.
   - Condition (b) is the only place a difference could arise. The trail T could be active in $\mathcal{G}_1$ but blocked in $\mathcal{G}_2$ only if there is a collider C on T such that $C \notin Z$, and:
     - $Desc_1(C) \cap Z \neq \emptyset$ (activated in $\mathcal{G}_1$)
     - $Desc_2(C) \cap Z = \emptyset$ (blocked in $\mathcal{G}_2$)
   - This would imply that the set of descendants of C is different in the two graphs. This means there must be some directed path $C \to D \dots$ in $\mathcal{G}_1$ that does not exist in $\mathcal{G}_2$. This implies some edge $W \to V$ on that path in $\mathcal{G}_1$ is reversed in $\mathcal{G}_2$ (i.e., $W \leftarrow V$).
   - Let's analyze the first such reversed edge $C \to D$ in $\mathcal{G}_1$ vs. $C \leftarrow D$ in $\mathcal{G}_2$. C is a collider on trail T, so T looks like $\dots \to C \leftarrow \dots$.
   - In $\mathcal{G}_1$, the triplet is $(\dots \to C \to D)$. This is not a v-structure.
   - In $\mathcal{G}_2$, the triplet is $(\dots \to C \leftarrow D)$. This is a *new* v-structure in $\mathcal{G}_2$.
   - This contradicts the premise that $\mathcal{G}_1$ and $\mathcal{G}_2$ have the same set of v-structures.
5. **Conclusion:** The set of descendants for any collider C on T must be identical in $\mathcal{G}_1$ and $\mathcal{G}_2$. Therefore, condition (b) is also identically met or violated in both graphs.

Since any trail T is active in $\mathcal{G}_1$ given Z if and only if it is active in $\mathcal{G}_2$ given Z, the graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ are I-equivalent. ■

Exercise 3.17

Prove Theorem 3.8: Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two graphs over $\mathcal{X}$. Then $\mathcal{G}_1$ and $\mathcal{G}_2$ have the same skeleton and the same set of immoralities if and only if they are I-equivalent.

Part 1: (I-equivalence) $\implies$ (Same Skeleton + Same Immoralities)

We prove the contrapositive: If $\mathcal{G}_1$ and $\mathcal{G}_2$ do *not* have the same skeleton or do *not* have the same immoralities, then they are *not* I-equivalent.

1. **Assume Skeletons Differ:**
   - Suppose $\mathcal{G}_1$ has an edge $X - Y$ that $\mathcal{G}_2$ does not.
   - In $\mathcal{G}_1$, X and Y are adjacent. By Lemma 3.1, they cannot be d-separated by any set Z. Thus, $(X \perp\!\!\!\perp Y|Z) \notin \mathcal{I}(\mathcal{G}_1)$ for any Z.
   - In $\mathcal{G}_2$, X and Y are not adjacent. By Lemma 3.2, there *exists* a set Z (e.g., $Pa_X^{\mathcal{G}_2}$) such that $d - sep_{\mathcal{G}_2}(X; Y|Z)$ holds. Thus, $(X \perp\!\!\!\perp Y|Z) \in \mathcal{I}(\mathcal{G}_2)$.
   - Since $\mathcal{I}(\mathcal{G}_1) \neq \mathcal{I}(\mathcal{G}_2)$, they are not I-equivalent.
2. **Assume Skeletons are Same, but Immoralities Differ:**
   - Suppose $X \to Z \leftarrow Y$ is an immorality in $\mathcal{G}_1$ but not in $\mathcal{G}_2$.
   - "Immorality" means X and Y are not adjacent.
   - In $\mathcal{G}_1$: The trail $X \to Z \leftarrow Y$ is a v-structure. It is *blocked* given $Z = \emptyset$. Thus, $d - sep_{\mathcal{G}_1}(X; Y|\emptyset)$ holds.
   - In $\mathcal{G}_2$: The skeleton $X - Z - Y$ is the same. Since it's not an immorality, the structure must be one of: $X \to Z \to Y$, $X \leftarrow Z \leftarrow Y$, or $X \leftarrow Z \to Y$.
   - In all three of these cases, the trail is *active* given $Z = \emptyset$ (since Z is a non-collider and $Z \notin \emptyset$).
   - Thus, $d - sep_{\mathcal{G}_2}(X; Y|\emptyset)$ does *not* hold.
   - Since $\mathcal{I}(\mathcal{G}_1) \neq \mathcal{I}(\mathcal{G}_2)$, they are not I-equivalent.

Part 2: (Same Skeleton + Same Immoralities) $\implies$ (I-equivalence)

This proof relies on the concept of a **minimal active trail**.

1. Assume $\mathcal{G}_1$ and $\mathcal{G}_2$ have the same skeleton and immoralities. Assume for contradiction they are *not* I-equivalent.
2. Then there must exist X, Y, Z such that (WLOG) $d{-}sep_{\mathcal{G}_1}(X; Y|Z)$ holds, but $\neg d{-}sep_{\mathcal{G}_2}(X; Y|Z)$ holds.
3. $\neg d - sep_{\mathcal{G}_2}(X; Y|Z)$ means there exists at least one active trail T between X and Y given Z in $\mathcal{G}_2$.
4. Let T be a *minimal* active trail in $\mathcal{G}_2$.
5. **Key Lemma (from prompt):** The only v-structures (colliders) that can exist on a minimal active trail are *immoralities*. Any covered v-structure $A \to B \leftarrow C$ (with edge $A - C$) would allow a "shortcut" $A - C$, meaning T was not minimal.
6. Therefore, all colliders on T are immoralities.
7. Since T is active in $\mathcal{G}_2$ given Z:
   - (a) Every non-collider W on T is $W \notin Z$.
   - (b) Every collider C on T (which must be an immorality) is "activated" ($C \in Z$ or $Desc_2(C) \cap Z \neq \emptyset$).
8. Now consider T in $\mathcal{G}_1$.

- T exists in $\mathcal{G}_1$ (same skeleton).
- Every collider C on T in $\mathcal{G}_2$ is an immorality. By premise, $\mathcal{G}_1$ has the same immoralities. So C is also an immorality (and thus a collider) in $\mathcal{G}_1$.
- Every non-collider W on T in $\mathcal{G}_2$ is not an immorality. It cannot be an immorality in $\mathcal{G}_1$ either. Thus W is a non-collider in $\mathcal{G}_1$.
- This means T has the *exact same set of colliders and non-colliders* in $\mathcal{G}_1$ as in $\mathcal{G}_2$.

9. Condition (a) is identical for T in $\mathcal{G}_1$.
10. The *only* way T could be *blocked* in $\mathcal{G}_1$ (as assumed) is if condition (b) fails for some collider C on T. This means:
    - $C \notin Z$ AND $Desc_2(C) \cap Z \neq \emptyset$ (Active in $\mathcal{G}_2$)
    - $C \notin Z$ AND $Desc_1(C) \cap Z = \emptyset$ (Blocked in $\mathcal{G}_1$)
11. This implies $Desc_1(C) \neq Desc_2(C)$, which requires an edge reversal on a descendant path from C.
12. As shown in the proof for Ex 3.16, any such edge reversal that changes descendant status must also change the set of v-structures (or immoralities, in this minimal case). This contradicts the premise.
13. Therefore, T must also be active in $\mathcal{G}_1$. This contradicts the assumption that $d\text{-}sep_{\mathcal{G}_1}(X; Y|Z)$ holds.
14. Thus, $\mathcal{G}_1$ and $\mathcal{G}_2$ must be I-equivalent.

Exercise 3.18

Prove Theorem 3.9: Two graphs $\mathcal{G}$ and $\mathcal{G}'$ are I-equivalent if and only if there exists a sequence of ... single reversal[s] of a covered edge.

a. Prove that reversing a covered edge preserves I-equivalence.

Proof. Let $\mathcal{G}$ be a graph with a covered edge $X \to Y$. By Definition 3.12, this means $Pa_Y^{\mathcal{G}} = Pa_X^{\mathcal{G}} \cup \{X\}$. Let $\mathcal{G}'$ be the graph identical to $\mathcal{G}$ but with the edge $Y \to X$.

By Theorem 3.8, we prove I-equivalence by showing they have the same skeleton and same immoralities.

1. **Skeleton:** Reversing an edge does not change the skeleton.
2. **Immoralities:** We must show the reversal neither creates nor destroys immoralities.
    - **Destroying an Immorality:** Could $X \to Y$ be part of an immorality $A \to Y \leftarrow X$ in $\mathcal{G}$?
        - This requires $A \in Pa_Y$ and A, X to be non-adjacent.
        - If $A \in Pa_Y$ and $A \neq X$, then by the covered edge premise, $A \in Pa_X$.
        - This means there is an edge $A \to X$.
        - Since A and X are adjacent, $A \to Y \leftarrow X$ is *not* an immorality.

- No immorality is destroyed.

- **Creating an Immorality:** Could $Y \to X$ create an immorality $Y \to X \leftarrow A$ in $\mathcal{G}'$?
  - This requires $A \in Pa_X^{\mathcal{G}'}$ and Y, A to be non-adjacent.
  - The new parents are $Pa_X^{\mathcal{G}'} = Pa_X^{\mathcal{G}} \cup \{Y\}$.
  - So, A must be in $Pa_X^{\mathcal{G}}$.
  - If $A \in Pa_X^{\mathcal{G}}$, then by the covered edge premise, $A \in Pa_Y^{\mathcal{G}}$.
  - This means there is an edge $A \to Y$ in $\mathcal{G}$, and thus also in $\mathcal{G}'$.
  - Since A and Y are adjacent, $Y \to X \leftarrow A$ is *not* an immorality.
  - No new immorality is created.

Since the skeleton and immoralities are unchanged, $\mathcal{G}$ and $\mathcal{G}'$ are I-equivalent. ■

b. Provide a counterexample if $X \to Y$ is not a covered edge.

Counterexample: Let $\mathcal{G} = A \to X \to Y$. The edge $X \to Y$ is **not covered**: $Pa_Y = \{X\}$ and $Pa_X = \{A\}$. $Pa_X \cup \{X\} = \{A, X\} \neq Pa_Y$.

Let $\mathcal{G}'$ be the graph with the edge reversed: $\mathcal{G}' = A \to X \leftarrow Y$.

- In $\mathcal{G}$: The trail $A \to X \to Y$ is active given $\emptyset$. Thus, $(A \perp\!\!\!\perp Y|\emptyset)$ does *not* hold.
- In $\mathcal{G}'$: The trail $A \to X \leftarrow Y$ is a v-structure. It is *blocked* given $\emptyset$. Thus, $(A \perp\!\!\!\perp Y|\emptyset)$ *does* hold.

Since $\mathcal{I}(\mathcal{G}) \neq \mathcal{I}(\mathcal{G}')$, the graphs are **not** I-equivalent.

c. Prove that I-equivalent networks are connected by covered edge reversals.

This is the "only if" part.

1. If $\mathcal{G}$ and $\mathcal{G}'$ are I-equivalent, by Theorem 3.8 they have the same skeleton and immoralities.
2. They belong to the same I-equivalence class, represented by a single PDAG (class graph) K.
3. $\mathcal{G}$ and $\mathcal{G}'$ are "completions" of K, formed by orienting the undirected edges of K.
4. We need to show that any undirected edge $X - Y$ in K corresponds to a covered edge in any valid orientation.
5. An edge $X - Y$ remains undirected in K *only if* neither orientation ($X \to Y$ or $Y \to X$) creates a new immorality or a cycle.
6. Let $X - Y$ be an undirected edge in K. If we orient it $X \to Y$ to create $\mathcal{G}_1$, for this to be covered, we need $Pa_Y^{\mathcal{G}_1} = Pa_X^{\mathcal{G}_1} \cup \{X\}$.
7. The rules R1-R3 find all *compelled* edges. An edge $X - Y$ remains undirected only if these rules do not apply.
8. Consider rule R1: If there was a node A such that $A \to X - Y$ (and A, Y not adjacent), R1 would compel $X \to Y$. Since $X - Y$ is undirected, this configuration must not exist.
9. This means any parent A of X ($A \to X$ in K) must either be adjacent to Y (which would create a covered v-structure, not an immorality) or must also be a parent of Y ($A \to Y$ in K).

A detailed analysis shows that for $X - Y$ to remain undirected, we must have $Pa_X^K = Pa_Y^K$.

10. If we orient this as $\mathcal{G}_1 = K \cup \{X \to Y\}$, then $Pa_Y^{\mathcal{G}_1} = Pa_Y^K \cup \{X\}$ and $Pa_X^{\mathcal{G}_1} = Pa_X^K$.
11. Since $Pa_X^K = Pa_Y^K$, we have $Pa_Y^{\mathcal{G}_1} = Pa_X^{\mathcal{G}_1} \cup \{X\}$. The edge $X \to Y$ is **covered**.
12. Any two I-equivalent graphs $\mathcal{G}$ and $\mathcal{G}'$ are two different valid orientations of K. We can transform $\mathcal{G}$ to $\mathcal{G}'$ by finding an edge $X \to Y$ in $\mathcal{G}$ that is $Y \to X$ in $\mathcal{G}'$. This edge must be undirected in K. By the logic above, $X \to Y$ is a covered edge in $\mathcal{G}$. Reversing it (which preserves I-equivalence by part (a)) gets us one step closer to $\mathcal{G}'$. We repeat this sequence until $\mathcal{G}$ is transformed into $\mathcal{G}'$.

Exercise 3.19

Prove Lemma 3.2: Let $\mathcal{G}^*$ be an I-map of a distribution P, and let X and Y be two variables that are not adjacent in $\mathcal{G}^*$. Then either $P \models (X \perp\!\!\!\perp Y | Pa_X^{\mathcal{G}^*})$ or $P \models (X \perp\!\!\!\perp Y | Pa_Y^{\mathcal{G}^*})$.

Proof.

1. A Bayesian network structure $\mathcal{G}^*$ is a directed acyclic graph (DAG).
2. By definition, a DAG cannot contain a directed cycle. This means it is impossible for X to be an ancestor of Y *and* Y to be an ancestor of X.
3. Therefore, at least one of the following must be true:
   - (a) Y is a non-descendant of X ($Y \in NonDescendants_X$).
   - (b) X is a non-descendant of Y ($X \in NonDescendants_Y$).
4. The graph $\mathcal{G}^*$ encodes a set of local independencies, $\mathcal{I}_l(\mathcal{G}^*)$. By Definition 3.1, this set contains the assertion:
   - For each variable X_i: $(X_i \perp\!\!\!\perp NonDescendants_{X_i} | Pa_{X_i}^{\mathcal{G}^*})$.
5. If (a) is true ($Y \in NonDescendants_X$), then the local independence for X implies $(X \perp\!\!\!\perp Y | Pa_X^{\mathcal{G}^*})$ by the decomposition property.
6. If (b) is true ($X \in NonDescendants_Y$), then the local independence for Y is $(Y \perp\!\!\!\perp NonDescendants_Y | Pa$ This implies $(Y \perp\!\!\!\perp X | Pa_Y^{\mathcal{G}^*})$, which is equivalent to $(X \perp\!\!\!\perp Y | Pa_Y^{\mathcal{G}^*})$.
7. The premise states that $\mathcal{G}^*$ is an I-map for P. By definition, P must satisfy all independencies encoded by $\mathcal{G}^*$. This includes the local independencies: $\mathcal{I}_l(\mathcal{G}^*) \subseteq \mathcal{I}(P)$.
8. Since at least one of the independencies in (5) or (6) must be in $\mathcal{I}_l(\mathcal{G}^*)$, P must satisfy at least one of them.
9. Therefore, either $P \models (X \perp\!\!\!\perp Y | Pa_X^{\mathcal{G}^*})$ or $P \models (X \perp\!\!\!\perp Y | Pa_Y^{\mathcal{G}^*})$ must hold.

■

Exercise 3.20

Show that Z is a requisite CPD for $P(X|Y)$ if and only if (in a modified graph $\mathcal{G}'$ with a new "dummy" parent $\hat{Z}$) $\hat{Z}$ has an active trail to X given Y.

Proof. Let $\mathcal{G}'$ be the graph $\mathcal{G}$ with the addition of a new root node $\hat{Z}$ and a single edge $\hat{Z} \to Z$. Z

is a requisite CPD for $P(X|Y)$ if changing $P(Z|Pa_Z)$ can change the value of $P(X|Y)$. Changing $P(Z|Pa_Z)$ is analogous to providing new "information" or "influence" at node Z. The query $P(X|Y)$ will change *if and only if* this influence can propagate from Z to X given the evidence Y.

The d-separation test $d - sep_{\mathcal{G}'}(\hat{Z}; X|Y)$ checks exactly for this flow of influence.

1. **Test:** $\neg d - sep_{\mathcal{G}'}(\hat{Z}; X|Y)$ **(Trail is Active)**
   - This means there is an active trail T from $\hat{Z}$ to X given Y.
   - This trail must be of the form $\hat{Z} \to Z \rightleftharpoons \cdots \rightleftharpoons X$.
   - For this trail to be active, the first link $\hat{Z} \to Z$ must be active. This means $Z \notin Y$ (as Z is not a v-structure on this link).
   - The rest of the trail $T' = Z \rightleftharpoons \cdots \rightleftharpoons X$ must also be active given Y, when entered from Z's parent ($\hat{Z}$).
   - This is precisely the condition for "influence" to flow from Z to X given Y.
   - By the completeness of d-separation (Theorem 3.4), if Z and X are not d-separated given Y (in this manner), there exists some distribution P where Z and X are dependent given Y.
   - $P(X|Y) = \sum_z P(X|Y, z) \cdot P(z|Y)$.
   - The term $P(z|Y)$ is computed using $P(Z|Pa_Z)$.
   - If X and Z are dependent given Y, $P(X|Y, z)$ changes for different z.
   - Since $Z \notin Y$ and there is an active trail, $P(z|Y)$ is non-trivial and depends on $P(Z|Pa_Z)$. By changing $P(Z|Pa_Z)$, we change $P(z|Y)$, which changes the weighted average $P(X|Y)$.
   - Therefore, Z is a **requisite CPD**.
2. **Test:** $d - sep_{\mathcal{G}'}(\hat{Z}; X|Y)$ **(Trail is Blocked)**
   - This means *all* trails from $\hat{Z}$ to X are blocked by Y.
   - **Case (a):** $Z \in Y$**.** The trail $\hat{Z} \to Z$ is blocked at Z. $d - sep$ holds.
     - When $Z \in Y$, the query is $P(X|Y) = P(X|Y \setminus \{Z\}, Z = z)$. The CPD $P(Z|Pa_Z)$ is not used to compute this; we use the *evidence* $Z = z$. Therefore, Z is **not requisite**.
   - **Case (b):** $Z \notin Y$**.** $d - sep$ holds, meaning every trail $Z \rightleftharpoons \cdots \rightleftharpoons X$ is blocked by Y (when entered from Z's parent).
     - This implies $(X \perp\!\!\!\perp Z|Y, \dots)$.
     - Because all paths of influence from Z to X are blocked by Y, the $P(Z|Pa_Z)$ term will factor out and cancel in the computation of $P(X|Y) = P(X, Y)/P(Y)$.
     - Therefore, Z is **not requisite**.

The test $\neg d - sep_{\mathcal{G}'}(\hat{Z}; X|Y)$ holds if and only if Z is a requisite CPD. ■

Exercise 3.21

a. Prove that $(Z \perp\!\!\!\perp U|Y)$ for a self-contained set Z.

Proof. We must show $d - sep(z; u|Y)$ for any $z \in Z$ and $u \in U$.

- Z is self-contained: any directed path between two nodes in Z is fully in Z.
- $Y = Pa(Z) \setminus Z$: parents of Z that are not in Z.
- $U = Ancestors(Z) \setminus (Y \cup Z)$: other ancestors of Z.
- $Z_{obs} = Y$.

Let T be any trail between $u \in U$ and $z \in Z$. We must show T is blocked by Y.

1. The trail must "enter" the set $Y \cup Z$.
2. Let w be the first node on T (starting from u) that is in $Y \cup Z$.
3. Let v be the node just before w on T. $v \in U$ (or $v = u$). $T = u \dots v \rightleftharpoons w \dots z$.
4. **Case A:** $w \in Y$**.**
   - The trail T hits $w \in Y = Z_{obs}$.
   - If w is a non-collider on T (e.g., $v \to w \to \dots$ or $v \leftarrow w \dots$), the trail is **BLOCKED** by $w \in Y$.
   - If w is a collider ($v \to w \leftarrow \dots$), the trail is **ACTIVE** at w.
   - Let's check if $v \to w \leftarrow z'$ ($z' \in Z$) is possible. $v \in U$. $w \in Y$. w is a descendant of z'. But $w \in Y = Pa(Z)$, so w is an ancestor of Z. This implies a cycle. Impossible.
   - Thus, any trail entering Y must be blocked.
5. **Case B:** $w \in Z$**.**
   - The segment is $v \rightleftharpoons w$, with $v \in U$ and $w \in Z$.
   - If $v \to w$: $v \in Pa(w)$. Since $w \in Z$ and $v \notin Z$, v must be in $Pa(Z) \setminus Z$. This means $v \in Y$. This contradicts $v \in U$ (since $U \cap Y = \emptyset$). Impossible.
   - If $v \leftarrow w$: $w \in Pa(v)$. $w \in Z$, $v \in U$. Z is an ancestor of U. U is an ancestor of Z. This implies a cycle. Impossible.
6. Since all possible trails from U to Z must be blocked, $d - sep(U; Z|Y)$ holds. Therefore, $P \models (Z \perp\!\!\!\perp U|Y)$.

■

b. Provide a counterexample if Z is not self-contained.

Counterexample Graph:

- Nodes: A, B, C, U
- Edges: $U \to C, A \to C, C \to B$.

Definitions:

- Let $Z = \{A, B\}$.

- Z is **not self-contained** because the directed path $A \to C \to B$ exists, but the intermediate node C is not in Z.
- $Y = Pa(Z) \setminus Z = (Pa(A) \cup Pa(B)) \setminus Z = (\emptyset \cup \{C\}) \setminus \{A, B\} = \{C\}$.
- $Ancestors(Z) = Ancestors(A) \cup Ancestors(B) = \{A\} \cup \{B, C, U, A\} = \{A, B, C, U\}$.
- $U = Ancestors(Z) \setminus (Y \cup Z) = \{A, B, C, U\} \setminus (\{C\} \cup \{A, B\}) = \{U\}$.
- $Z_{obs} = Y = \{C\}$.

Test the Independence: We must test $(Z \perp\!\!\!\perp U|Y)$, which is $(\{A, B\} \perp\!\!\!\perp \{U\}|\{C\})$.

- Test $d - sep(A; U|\{C\})$:
- The trail is $A \to C \leftarrow U$.
- C is a v-structure.
- C is in the conditioning set $Z_{obs} = \{C\}$.
- The v-structure is activated. The trail is **ACTIVE**.

Since d-separation fails, the independence $(Z \perp\!\!\!\perp U|Y)$ does not hold.

Exercise 3.22

Consider what happens if P does not have a P-map. For each of the two types of errors, either prove that they cannot happen, or provide a counterexample.

The algorithm `Build-PMap-Skeleton` (Algorithm 3.3) removes an edge $X - Y$ if it finds a witness set U (of size $\leq d$) such that $P \models (X \perp\!\!\!\perp Y|U)$. It keeps the edge otherwise.

1. **Missing edges: CANNOT HAPPEN.**
   - A "missing edge" error means the algorithm *removes* an edge $X - Y$ that appears in *all* minimal I-maps of P.
   - If the algorithm removes $X - Y$, it must have found a witness set U such that $P \models (X \perp\!\!\!\perp Y|U)$.
   - By definition, *any* I-map $\mathcal{G}$ for P must satisfy all independencies in P, so $\mathcal{I}(\mathcal{G}) \subseteq \mathcal{I}(P)$.
   - Therefore, *all* minimal I-maps $\mathcal{G}_{min}$ for P must satisfy $(X \perp\!\!\!\perp Y|U)$.
   - By the soundness of d-separation (Theorem 3.1), if $\mathcal{G}_{min}$ satisfies this independence, it must be that $d - sep_{\mathcal{G}_{min}}(X; Y|U)$ holds.
   - An edge $X - Y$ is a trail of length 1. Such a trail cannot be blocked by any U. If X and Y are adjacent, they cannot be d-separated.
   - Therefore, X and Y cannot be adjacent in $\mathcal{G}_{min}$.
   - This contradicts the premise that the edge $X - Y$ appears in $\mathcal{G}_{min}$.
   - Thus, the algorithm can never remove an edge that is present in all minimal I-maps.
2. **Spurious edges: CAN HAPPEN.**
   - A "spurious edge" error means the algorithm *keeps* an edge $X - Y$ that does *not* appear in all minimal I-maps.

- This happens if the algorithm *fails* to find a witness U (with $|U| \leq d$) for $X - Y$, but a minimal I-map constructed with a specific ordering *can* remove the edge $X - Y$.
- **Counterexample:** Consider the distribution P from Example 3.8 (the "Misconception" diamond), which does not have a P-map. This distribution P has the independencies $(A \perp\!\!\!\perp C|\{B, D\})$ and $(B \perp\!\!\!\perp D|\{A, C\})$.
- Assume we run `Build-PMap-Skeleton` with a bound of $d = 1$.
- The algorithm will test $P \models (A \perp\!\!\!\perp C|U)$ for all $|U| \leq 1$. It will *fail* to find the true independence, which requires a witness set of size 2 ($U = \{B, D\}$).
- Therefore, the algorithm *keeps* the edge $A - C$.
- Now, we show $A - C$ is *not* in all minimal I-maps.
- Let's construct a minimal I-map (Algorithm 3.2) using the ordering: B, D, A, C.
  (a) Add B: $\emptyset$
  (b) Add D: $P \not\models (D \perp\!\!\!\perp B)$, so add $B \to D$.
  (c) Add A: $P \not\models (A \perp\!\!\!\perp B|D)$ and $P \not\models (A \perp\!\!\!\perp D|B)$, so add $B \to A$ and $D \to A$.
  (d) Add C: We test for parents from $\{B, D, A\}$. We know $P \models (C \perp\!\!\!\perp A|\{B, D\})$. This independence is of the form $(X_i \perp\!\!\!\perp \{X_1, \ldots, X_{i-1}\} - U|U)$, with $X_i = C$ and $U = \{B, D\}$.
  (e) This means A is *not* a parent of C in this minimal I-map.
- The resulting minimal I-map $\mathcal{G}_{B,D,A,C}$ does *not* contain the edge $A - C$.
- The algorithm *kept* $A - C$, but we found a minimal I-map that *omits* $A - C$.
- Therefore, the algorithm made a "spurious edge" error.

Exercise 3.23

Prove Proposition 3.3 by induction, using the definition of a partial class graph.
Proposition 0.1. *Let P be a distribution that has a P-map $\mathcal{G}^*$, and let K be the graph returned by `Build-PDAG(X, P)`. Then, if $X \to Y \in K$ then $X \to Y$ appears in all DAGs in the I-equivalence class of $\mathcal{G}^*$.*

Proof. We prove by induction that at every step i of the `Build-PDAG` algorithm, the graph K_i is a **partial class graph** for $\mathcal{G}^*$. A graph K is a partial class graph for $\mathcal{G}^*$ if:

- (a) K has the same skeleton as $\mathcal{G}^*$.
- (b) K has the same immoralities as $\mathcal{G}^*$.
- (c) If $X \to Y \in K$, then $X \to Y$ appears in all DAGs in $\mathcal{G}^*$'s I-equivalence class (i.e., $X \to Y$ is *compelled*).

1. **Base Case ($i = 0$):** Let K_0 be the graph returned by `Mark-Immoralities`.
   - (a) `Build-PMap-Skeleton` correctly identifies the skeleton (since a P-map exists).
   - (b) `Mark-Immoralities` correctly identifies all immoralities (by Prop 3.1 and 3.2).

- (c) The only directed edges in K_0 are the immoralities. By Theorem 3.8, all DAGs in an I-equivalence class share the same immoralities. Therefore, these edges are compelled.

Thus, K_0 is a partial class graph.

2. **Inductive Step:** Assume K_i is a partial class graph. Let K_{i+1} be the graph after applying one rule (R1, R2, or R3) that orients a new edge, say $A \to B$. We must show K_{i+1} is also a partial class graph.
   - (a) Skeletons are the same (rules only add directionality).
   - (b) Immoralities are the same. The rules are designed to orient edges *specifically to avoid* creating new immoralities (R1, R3) or cycles (R2).
   - (c) We must show the new edge $A \to B$ is compelled.
     - **If R1 applied:** $X \to Y - A$ (no $X - A$) $\implies$ $Y \to A$ (so $A = Z$ in rule). We have $X \to Y \in K_i$. By inductive hypothesis (c), $X \to Y$ is in all $\mathcal{G}_{eq}$ (I-equivalent DAGs). In any $\mathcal{G}_{eq}$, the edge $Y - A$ must be oriented. If it were $Y \leftarrow A$, $\mathcal{G}_{eq}$ would contain $X \to Y \leftarrow A$. Since X, A are not adjacent, this is a *new immorality* not in K_i. This contradicts $\mathcal{G}_{eq}$ being in the I-equivalence class. Therefore, $\mathcal{G}_{eq}$ *must* contain $Y \to A$. The new edge is compelled.
     - **If R2 applied:** $X \to Y \to Z$ and $X - A$ (with $A = Z$) $\implies$ $X \to A$. By hypothesis (c), $X \to Y \to Z$ is in all $\mathcal{G}_{eq}$. In any $\mathcal{G}_{eq}$, the edge $X - A$ must be oriented. If it were $X \leftarrow A$, $\mathcal{G}_{eq}$ would contain a cycle $X \to Y \to Z \to X$. Impossible for a DAG. Therefore, $\mathcal{G}_{eq}$ *must* contain $X \to A$. The new edge is compelled.
     - **If R3 applied:** $Y_1 \to A \leftarrow Y_2$ (no $Y_1 - Y_2$), $X - Y_1$, $X - Y_2$, $X - A$ $\implies$ $X \to A$ (with $A = Z$). By hypothesis (c), $Y_1 \to A \leftarrow Y_2$ is in all $\mathcal{G}_{eq}$. Assume $X - A$ is $X \leftarrow A$. Consider $X - Y_1$. If $X \to Y_1$, we have a cycle $X \to Y_1 \to A \to X$. Impossible. So it must be $Y_1 \to X$. Similarly, $Y_2 \to X$. But this creates $Y_1 \to X \leftarrow Y_2$. Since Y_1, Y_2 are not adjacent, this is a *new immorality* not in K_i. This is a contradiction. $\mathcal{G}_{eq}$ *must* contain $X \to A$. The new edge is compelled.
3. **Conclusion:** By induction, the final graph K returned by `Build-PDAG` is a partial class graph. Therefore, any directed edge $X \to Y \in K$ is compelled to be in all DAGs in the I-equivalence class of $\mathcal{G}^*$.

■

Exercise 3.24

Prove Proposition 3.4: Let K be the graph returned by `Build-PDAG`. Then, if $X \to Y \in K$ and $Y - Z \in K$, then $X \to Z \in K$.

Proof. (Note: The proposition as stated in the text is $X \to Z \in K$. This is what is used in the proof of Prop 3.5.)

1. Let K be the final, converged graph returned by `Build-PDAG`.
2. We are given $X \to Y \in K$ and $Y - Z \in K$.
3. Since K has converged, the edge $Y - Z$ was *not* oriented by any of the rules R1, R2, or R3.

4. Consider rule **R1**: $(X \to Y - Z, \text{no } X - Z) \implies Y \to Z$.
5. The premise $X \to Y$ is met. The premise $Y - Z$ is met.
6. However, the *consequence* $Y \to Z$ did *not* happen (we still have $Y - Z$).
7. This means the rule *must not have applied*. The only way the rule could fail to apply is if its third premise, "no $X - Z$", is **false**.
8. Therefore, there *must* be an edge between X and Z in the skeleton of K.
9. So, K contains $X \to Y$, $Y - Z$, and an edge $X - Z$ (which is $X - Z$, $X \to Z$, or $X \leftarrow Z$).
10. Can the edge be $X \leftarrow Z$?
    - If $X \leftarrow Z$, the graph contains the path $Z \to X \to Y$.
    - This path, combined with the edge $Y - Z$, forms a cycle ($Z \to X \to Y - Z$).
    - This configuration ($Z \to X \to Y$, $Z - Y$) matches rule **R2**.
    - Rule R2 would compel $Z \to Y$.
    - This contradicts our premise that $Y - Z \in K$ (i.e., that the graph K has converged).
    - Therefore, the orientation $X \leftarrow Z$ is impossible.
11. Can the edge be $X - Z$?
    - If $X - Z$, K contains $X \to Y$, $Y - Z$, and $X - Z$.
    - This configuration has not converged. It is not stable. For example, rule R2 could apply if $Y - Z$ were oriented $Y \to Z$, or R1 could apply.
    - A converged graph cannot contain this structure. (This is a more subtle point, but the configuration is not "stuck".)
12. The only remaining possibility is $X \to Z$.
    - If K contains $X \to Y, Y - Z, X \to Z$, this is a stable, converged state. R1 does not apply (since $X - Z$ is an edge). R2 does not apply (since $Y - Z$ is not $Y \to Z$).
    - Therefore, for the graph K to have converged with $X \to Y$ and $Y - Z$, it *must* be the case that $X \to Z$ is also in K.

■

Exercise 3.25

Prove Proposition 3.6: The PDAG K returned by `Build-PDAG` is necessarily chordal.

Proof. (Note: This proof is non-trivial and relies on properties of faithful distributions not fully covered in the chapter. The hint in the text is insufficient.)

1. Let K_0 be the graph returned by `Mark-Immoralities`, which consists of the skeleton S and directed edges for immoralities.
2. A graph is **chordal** if every cycle of length ≥ 4 has a **chord** (an edge between non-adjacent nodes in the cycle).

3. Let S be the skeleton of K_0. Assume S contains a chordless cycle $C = X_1 - X_2 - \cdots - X_k - X_1$ with $k \geq 4$.
4. For K to be chordal, this cycle must be "broken" by the orientation rules.
5. A key property of distributions P that have a P-map $\mathcal{G}^*$ (i.e., are faithful to $\mathcal{G}^*$) is that their skeleton S must be chordal.
6. *Sketch of proof for this property:* Assume the skeleton S has a chordless cycle C of length $k \geq 4$. No triplet $X_{i-1} - X_i - X_{i+1}$ in this cycle can be an immorality in $\mathcal{G}^*$, because that would require X_{i-1} and X_{i+1} to be non-adjacent, which they are (by chordless property). However, other triplets $X_i - X_j - X_l$ in the cycle *could* be immoralities.
7. A theorem by Verma and Pearl (related to faithfulness) shows that if P is faithful to $\mathcal{G}^*$, then the skeleton of $\mathcal{G}^*$ must be chordal.
8. Since `Build-PMap-Skeleton` correctly identifies the skeleton S of $\mathcal{G}^*$ (when P has a P-map), the skeleton S must be chordal.
9. The graph K returned by `Build-PDAG` has the same skeleton S.
10. Since S is chordal, K is also chordal (as chordality is a property of the undirected skeleton).

■

Exercise 3.26

Implement an efficient algorithm that takes a Bayesian network ... and a full instantiation ξ to $\mathcal{X}$, and computes the probability of ξ.

This algorithm implements the **Chain Rule for Bayesian Networks**.

```
Procedure Compute-Probability(B, xi):
  // B is the Bayesian Network (Graph G, CPDs P)
  // xi is a map of {Variable: value}

  P_total = 1.0

  // Iterate over all variables in the graph
  for X_i in G.Variables:
    // 1. Get the value of X_i from the instantiation
    x_i = xi[X_i]

    // 2. Get the parents of X_i from the graph
    Parents = G.Parents(X_i)

    // 3. Get the instantiated values of the parents from xi
    parent_values = {}
    for P_j in Parents:
      parent_values[P_j] = xi[P_j]

    // 4. Look up the conditional probability in the CPD for X_i
    // P(X_i = x_i | Parents = parent_values)
```

```
    prob = B.CPD(X_i).GetProbability(x_i, parent_values)

    // 5. Multiply this probability into the total
    P_total = P_total * prob

  return P_total
```

Exercise 3.27

Implement `Reachable` of algorithm 3.1.

```
Procedure Reachable(G, X, Z):
  // G: Bayesian network graph
  // X: Source variable
  // Z: Set of observation variables

  // Phase I: Find all ancestors of Z (nodes that can activate v-structures)
  L_ancestors = Z
  A = empty_set
  while L_ancestors is not empty:
    Select some Y from L_ancestors
    L_ancestors.remove(Y)
    if Y is not in A:
      L_ancestors.add(G.Parents(Y))
      A.add(Y)

  // Phase II: Traverse active trails starting from X
  // L: List of (Node, direction) to visit.
  // d = 'up' (arrived from a child)
  // d = 'down' (arrived from a parent)
  L = {(X, 'up')}

  // V: Set of (Node, direction) marked as visited
  V = empty_set
  // R: Set of nodes reachable via active trail
  R = empty_set

  while L is not empty:
    Select some (Y, d) from L
    L.remove((Y, d))

    if (Y, d) is not in V:
      if Y is not in Z:
        R.add(Y) // Y is reachable
      V.add((Y, d))

      if d == 'up' and Y is not in Z:
        // Trail up through Y (serial or common cause) is active
```

```
      for P in G.Parents(Y):
        L.add((P, 'up')) // Visit parents from bottom
      for C in G.Children(Y):
        L.add((C, 'down')) // Visit children from top

    else if d == 'down':
      if Y is not in Z:
        // Downward serial trail is active
        for C in G.Children(Y):
          L.add((C, 'down')) // Visit children from top

      if Y is in A:
        // v-structure trail is active
        for P in G.Parents(Y):
          L.add((P, 'up')) // Visit parents from bottom

  return R
```

Exercise 3.28

Implement an efficient algorithm that determines, for a given set Z of observed variables and all pairs of nodes X and Y, whether X, Y are d-separated in G given Z.

We can exploit the symmetry of d-separation ($d - sep(X; Y|Z) \iff d - sep(Y; X|Z)$) to cut the number of calls to `Reachable` roughly in half.

```
Procedure All-Pairs-D-Separation(G, Z):
  // G: Bayesian network graph (with n nodes)
  // Z: Set of observation variables
  // Output: An n x n matrix D, where D[i, j] = true if X_i, X_j d-separated

  // D[i, j] = null (uncomputed), true (d-separated), false (d-connected)
  D = new n x n matrix initialized to null

  // Pre-compute the set A of nodes that activate v-structures
  // This is Phase I of the Reachable algorithm
  L_ancestors = Z
  A = empty_set
  while L_ancestors is not empty:
    Select some Y from L_ancestors
    L_ancestors.remove(Y)
    if Y is not in A:
      L_ancestors.add(G.Parents(Y))
      A.add(Y)

  for i = 1 to n:
    // If D[i, i] is null, we have not computed reachability from X_i
    if D[i, i] is null:
```

```
      // Run Reachable's Phase II starting from X_i
      // We pass pre-computed A
      Reachable_from_i = Reachable_Phase_II(G, X_i, Z, A)

      // Now, fill in the matrix based on this run
      for j = 1 to n:
        if X_j is in Reachable_from_i:
          // X_i and X_j are d-connected
          D[i, j] = false
          D[j, i] = false // Exploit symmetry
        else:
          // X_i and X_j are d-separated
          D[i, j] = true
          D[j, i] = true // Exploit symmetry

  return D

// Helper function: Just Phase II of Algorithm 3.1
Procedure Reachable_Phase_II(G, X, Z, A):
  L = {(X, 'up')}
  V = empty_set
  R = empty_set

  while L is not empty:
    Select some (Y, d) from L
    L.remove((Y, d))

    if (Y, d) is not in V:
      if Y is not in Z:
        R.add(Y)
      V.add((Y, d))

      if d == 'up' and Y is not in Z:
        for P in G.Parents(Y):
          L.add((P, 'up'))
        for C in G.Children(Y):
          L.add((C, 'down'))

      else if d == 'down':
        if Y is not in Z:
          for C in G.Children(Y):
            L.add((C, 'down'))
        if Y is in A:
          for P in G.Parents(Y):
            L.add((P, 'up'))

  return R
```