Artificial Intelligence

CS4365 --- Fall 2022

Final Exam Review

Instructor: Yunhui Guo

Final Exam

- Location: CR 1.202
- Time: 5:00PM - 7:45PM, Tuesday, Dec 13

- Topics:
  - State-space search
  - Propositional logic
  - First-order logic
  - Bayes Net

State-space Search

- Breadth first search
- Depth first search
- Greedy best-first search
- A* search

Generic Tree-Search Algorithm

Add initial state to the frontier

Loop

node = remove-frontier( ) -- and save in order to return as part of path to goal

if goal-test(node) = true return path to goal

S = successors(node)

Add S to frontier

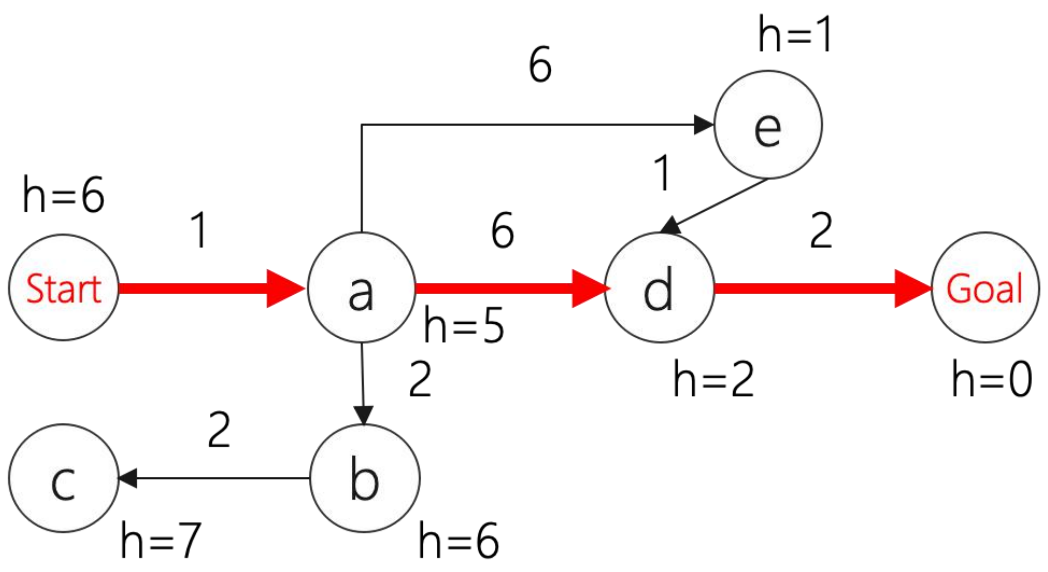
until frontier is empty
return failure

Final review

- Informed search
  - Greedy Best-First Search
    - Expands the node that is “closest” to the goal as measured by h(n)

  - A* search
    - Combining Uniform-Cost Search and Greedy Best-First Search
    - $f(n) = g(n) + h(n)$
      - $g(n)$: the path cost from the start node to node n
      - $h(n)$: the estimated cost of the cheapest path from node n to the goal node

Final review
A* Search



- Uniform-cost orders by path cost g(n)
- Greedy best first search orders by estimated goal proximity h(n)
- A* search combines g(n) and h(n)

Logic

- Logic:
  - defines a formal language for logical reasoning
- It gives us a tool that helps us to understand how to construct a valid argument
- Logic Defines:
  - the meaning of statements
  - the rules of logical inference

Logic as a Knowledge Representation

Model: a truth assignment to every propositonal symbol

Logic entailment:

A sentence follows logically from another sentence:

$$\alpha \models \beta$$

In every model in which α is true, β is also true.

KR Language: Propositional Logic

- Literal: an atomic formula or its negation
  - Positive literal: P, Q
  - Negative literal: ¬P, ¬Q

- Syntax: build sentences from atomic propositions, using connectives:
  - $\wedge$: and
  - $\vee$: or
  - ¬ : not
  - ⇒ : implies
  - ⇔ : equivalence (biconditional)

KR Language: Propositional Logic

Syntax: build sentences from atomic propositions, using connectives $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$

(and / or / not / implies / equivalence (biconditional))

E.g.: $\neg P$

$Q \wedge R$

$(\neg P \vee (Q \wedge R)) \Rightarrow S$

KR Language: Propositional Logic

- Clause: a disjunction of literals

  E.g.: $Q \lor R$

- Conjunctive normal form (CNF): a conjunction of clauses

  E.g.: $(Q \lor R) \land (P \lor R)$

- Every formula can be equivalently written as a formula in conjunctive normal form

  $(Q \land R) \lor P \quad \rightarrow \quad (Q \lor P) \land (R \lor P)$

Semantics

Semantics specifies what something means.

In propositional logic, the semantics (i.e., meaning) of a sentence is the set of interpretations (i.e., truth assignments) in which the sentence evaluates to True.

Example:

The semantics of the sentence $P \vee Q \Rightarrow R$ is

- P is True, Q is True, R is True
- P is True , Q is False, R is True
- P is False , Q is True , R is True
- P is False , Q is False , R is True
- P is False , Q is False , R is False

Evaluating a sentence under interpretation I

We can evaluate a sentence using a truth table

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

Evaluating a sentence under interpretation I

We can evaluate a sentence using a truth table

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

Note: $\Longrightarrow$ is somewhat counterintuitive

What's the true value of “5 is even implies Sam is smart”

If P is True, then I claim Q is True

Three Important Concepts

- Logic Equivalence

- Validity

- Satisfiability

Logic Equivalence

- Two sentences are equivalent if they are true in the same set of models.
- We write this as $\alpha \equiv \beta$. $\alpha \equiv \beta$ if and only if $\alpha \vDash \beta$ and $\beta \vDash \alpha$

For example:

I. If Lisa is in Denmark, then she is in Europe

II. If Lisa is not in Europe, then she is not in Denmark

Logic Equivalence

- $(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$ commutativity of $\wedge$
- $(\alpha \vee \beta) \equiv (\beta \vee \alpha)$ commutativity of $\vee$
- $((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$ associativity of $\wedge$
- $((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$ associativity of $\vee$
- $\neg(\neg \alpha) = \alpha$ double-negation
- $(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$ contraposition
- $(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$ implication elimination
- $(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$ biconditional elimination

Logic Equivalence

- $\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$ De Morgan
- $\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$ De Morgan
- $(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$ distributivity of $\wedge$ over $\vee$
- $(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$ distributivity of $\vee$ over $\wedge$

These equivalences play much the same role in logic as arithmetic identities do in ordinary mathematics.

Validity

- Some sentences are very true! For example

1) True 2) $P \Rightarrow P$ 3) $(P \wedge Q) \Rightarrow Q$

A valid sentence is one whose meaning includes every possible interpretation.

$$((P \vee H) \wedge (\neg H)) \Rightarrow P$$

P	H	$P \vee H$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	*False*	*False*	*False*	*True*
False	*True*	*True*	*False*	*True*
True	*False*	*True*	*True*	*True*
True	*True*	*True*	*False*	*True*

The truth table shows that $((P \vee H) \wedge (\neg H)) \Rightarrow P$ is valid

We write $\models ((P \vee H) \wedge (\neg H)) \Rightarrow P$

Satisfiability

- An unsatisfiable sentence is one whose meaning has no interpretation (e.g., $P \land \neg P$)

- A satisfiable sentence is one whose meaning has at least one interpretation.

- A sentence must be either satisfiable or unsatisfiable but it can’t be both.

- If a sentence is valid then it’s satisfiable.
- If a sentence is satisfiable then it may or may not be valid.

Convert to CNF

- Convert $(B_{1,1} \Leftrightarrow (P_{1,2} \lor P_{2,1})) \land \neg B_{1,1} \land P_{1,2}$ into CNF
- $((B_{1,1} \Rightarrow (P_{1,2} \lor P_{2,1})) \land ((P_{1,2} \lor P_{2,1}) \Rightarrow B_{1,1})) \land \neg B_{1,1} \land P_{1,2}$ (biconditional)
- $((\neg B_{1,1} \lor (P_{1,2} \lor P_{2,1})) \land (\neg(P_{1,2} \lor P_{2,1}) \lor B_{1,1}) \land \neg B_{1,1} \land P_{1,2}$ (Implication elimination)
- $(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land ((\neg P_{1,2} \land \neg P_{2,1}) \lor B_{1,1}) \land \neg B_{1,1} \land P_{1,2}$ (De Morgan)
- $(\neg B_{1,1} \lor P_{1,2} \lor P_{2,1}) \land (\neg P_{1,2} \lor B_{1,1}) \land (\neg P_{2,1} \lor B_{1,1}) \land \neg B_{1,1} \land P_{1,2}$ (Distri.)

The Resolution Rule (clausal form)

From $\alpha \vee p$ and $\neg p \vee \beta$, we can derive:

$\alpha \vee \beta$ (α and β are disjunctions of literals, where a literal is a propositional variable or its negation).

- Reason:
  - If p is true, β must be true
  - If p is false, α must be true
  - So β or α holds

The Resolution Rule (clausal form)

From $\alpha \lor p$ and $\neg p \lor \beta$, we can derive:

$\alpha \lor \beta$ (resolvent)

(α and β are disjunctions of literals, where a literal is a propositional variable or its negation).

Note: $\neg\alpha \Rightarrow p$ and $p \Rightarrow \beta$ gives $\neg\alpha \Rightarrow \beta$

It is a “chaining rule”.

General Resolution Rule

If $(L_1 \vee L_2 \vee ...L_k)$ is True,

and $(\neg L_k \vee L_{k+1} \vee ...L_m)$ True,

then we can conclude that

$(L_1 \vee L_2 \vee ...L_{k-1} \vee L_{k+1} \vee ... \vee L_m)$ is True.

A resolution-based theorem prover can, for any sentences α and β in propositional logic, decide whether $\alpha \vDash \beta$

General Resolution Rule

- We can derive the empty clause via resolution iff the set of clauses is inconsistent. ($\neg p \land p$)

- Method relies on property III. It's refutation complete. Note that method does not generate theorems from scratch

Algorithm: Resolution Proof

- Negate the theorem to be proved, and add the result to the list of sentences in the KB. We show that (KB $\wedge$ $\neg\alpha$) is unsatisfiable
- Put the list of sentences into conjunctive normal form.

- Until there is no resolvable pair of clauses,
  - Find resolvable clauses and resolve them.
  - Add the results of resolution to the list of clauses.
  - If NIL (empty clause) is produced, stop and report that the (original) theorem is true. (empty clause represents contradiction)
- Report that the (original) theorem is false

Example

1) ARM-OK

2) $\neg$MOVES

3) ARM-OK $\land$ LIFTABLE $\Rightarrow$ MOVES

4) $\neg$ARM-OK $\lor$ $\neg$LIFTABLE $\lor$ MOVES

Prove: $\neg$LIFTABLE

5) LIFTABLE (assert)

6) $\neg$ARM-OK $\lor$ MOVES (resolving 5 and 4)

7) $\neg$ARM-OK (from 6 and 2)

8) Nil (empty clause / contradiction, from 7 and 1).

Predicate Logic

- Predicate logic (or first-order logic) is an extension of propositional logic that permits concisely reasoning about whole classes of entities and relations.

- **Propositional logic** treats simple propositions (sentences) as atomic entities

- In contrast, **predicate logic** distinguishes the subject of a sentence from its predicate.

Predicate Logic

- Plato is a philosopher
- Socrates is a philosopher

- Propositional logic:
  - P: Plato is a philosopher
  - Q: Socrates is a philosopher
- Predicate logic:
  - Variable: a
  - Predicate: “is a philosopher”

Quantifier Expressions

- Quantifiers provide a notation that allows us to quantify (count) how many objects in the u.d. satisfy a given predicate.

- "$\forall$" is the FOR ALL or **universal quantifier.**

$\forall x\, P(x)$ means for all x in the u.d., P holds

- "$\exists$" is the EXISTS or **existential quantifier.**

$\exists x\, P(x)$ means there exists an x in the u.d. (that is, 1 or more) such that P(x) is true.

The Universal Quantifier $\forall$

- Example:

Let the u.d. of x be parking spaces at UTD.

Let P(x) be the predicate "x is full".

Then the universal quantification of P(x), $\forall x\, P(x)$, is the proposition:

- "All parking spaces at UTD are full"
- i.e., "Every parking space at UTD is full."
- i.e., "For each parking space at UTD, that space is full."

The Universal Quantifier $\forall$

- Example
  - All kings are persons
    - $\forall x \ \ King(x) \Rightarrow Person(x)$
- Extended interpretation:
  - x → Richard the Lionheart
  - x → King John
  - x → Richard’s left leg
  - x → John’s left leg,
  - x → the crown.

The Universal Quantifier ∀

- ∀x King(x) ⇒ Person(x) is true if the sentence King(x) ⇒ Person(x) is true under each of the five extended interpretations

- Richard the Lionheart is a king ⇒ Richard the Lionheart is a person
- King John is a king ⇒ King John is a person.
- Richard’s left leg is a king ⇒ Richard’s left leg is a person.
- John’s left leg is a king ⇒ John’s left leg is a person
- The crown is a king ⇒ the crown is a person

The Universal Quantifier $\forall$

- A common mistake using $\forall$
  - $\forall x \;\; King(x) \land Person(x)$

- "Richard the Lionheart is a king $\land$ Richard the Lionheart is a person"
- "King John is a king $\land$ King John is a person"
- "Richard's left leg is a king $\land$ Richard's left leg is a person"
- ...

The Existential Quantifier ∃

- Example:

Let the u.d. of x be parking spaces at UTD.

Let P(x) be the predicate “x is full”.

Then the existential quantification of P(x), $\exists x\, P(x)$, is the proposition:

- “Some parking spaces at UTD are full”
- i.e., “There is a parking space at UTD is full.”
- i.e., “At least one parking space at UTD is full.”

The Existential Quantifier ∃

- Example
  - King John has a crown on his head
    - $\exists x\ Crown(x) \land OnHead(x, John)$
- At least one of the following is true:
  - Richard the Lionheart is a crown ∧ Richard the Lionheart is on John's head;
  - King John is a crown ∧ King John is on John's head
  - Richard's left leg is a crown ∧ Richard's left leg is on John's head;
  - John's left leg is a crown ∧ John's left leg is on John's head;
  - The crown is a crown ∧ the crown is on John's head.

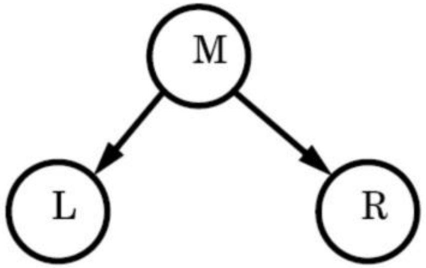
Quantifier Exercise

- If $R(x, y)$ = "x relies upon y", express the following in unambiguous English:
- $\forall x(\exists y\ R(x, y))$:
  - Everyone has someone to rely on
- $\exists y(\forall x\ R(x, y))$:
  - There's a poor overburdened soul whom everyone relies upon (including himself)!
- $\exists x(\forall y\ R(x, y))$:
  - There's some needy person who relies upon everybody (including himself).
- $\forall y(\exists x\ R(x, y))$:
  - Everyone has someone who relies upon them.
- $\forall x(\forall y\ R(x, y))$:
  - Everyone relies upon everybody, (including themselves)!

A Bayes Net

- Represents a set of variables and their conditional dependencies via a directed acyclic graph

- **Nodes** represent **variables**,
  - Can be assigned (observed) or unassigned (unobserved)
- **Edges** represent **conditional dependencies**
  - Similar to CSP constraints

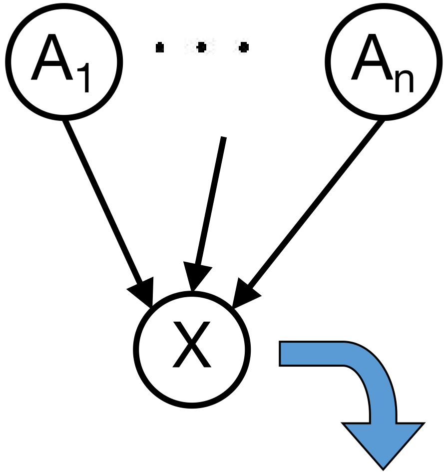


- Nodes that are not connected represent variables that are conditionally independent of each other

- For example, a Bayes network could represent the probabilistic relationships between diseases and symptoms

Bayes' Net Semantics

- **Variables**: a set of nodes, one per variable X
- A directed, acyclic graph
- A conditional distribution for each node
  - A collection of distributions over X, one for each combination of parents' values
  - CPT: conditional probability table
  - Description of a noisy "causal" process



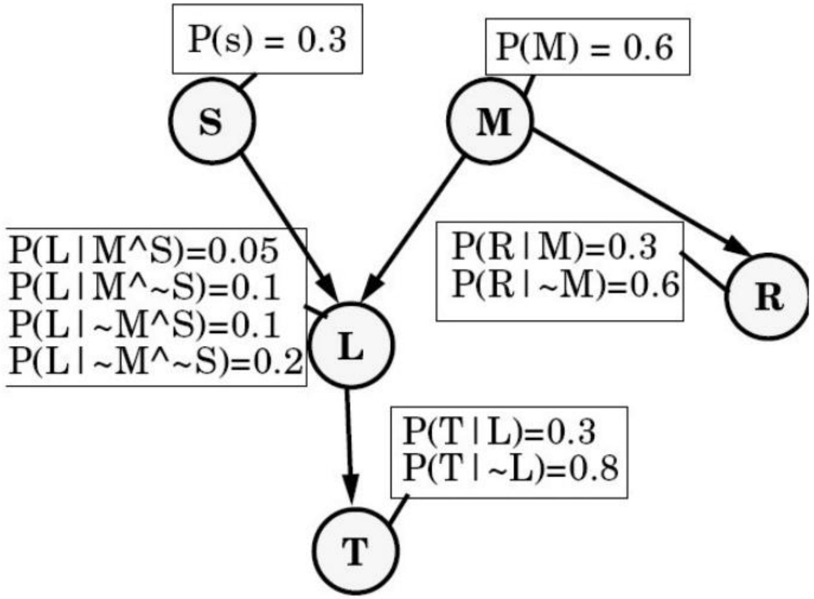
$P(X|A_1 \ldots A_n)$

- *A Bayes net = Topology (graph) + Local Conditional Probabilities*

What you should know

- Compute joint distribution
- Compute marginal distribution
- Compute conditional distritbution
- D-seperation

Computing with a Bayes Net



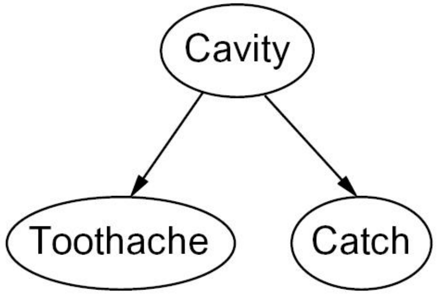
- The first thing we might want to do is compute an entry in a joint probability table
- Given an assignment of truth values to our variables, what is the probability? E.g., What is P(S, ~M, L, ~R, T)?

Probabilities in BNs

- Bayes' nets implicitly encode joint distributions
  - As a product of local conditional distributions
  - To see what probability a BN gives to a full assignment, multiply all the relevant conditionals together:

$$P(x_1, x_2, \ldots x_n) = \prod_{i=1}^{n} P(x_i|parents(X_i))$$

- Example:



$$P(+cavity, +catch, -toothache)$$

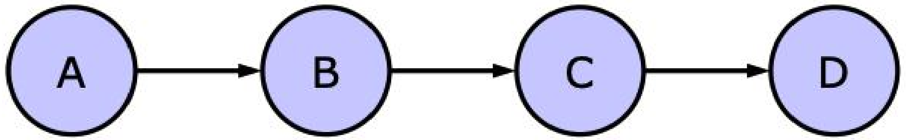
Compute Marginal Distribution

- To answer any query involving a conjunction of variables, sum over the variables not involved in the query

- Marginalization:

$$P(y) = \Sigma_{ABC} P(a,b,c,y),$$

Marginal distribution



- $P(d) = \Sigma_A \Sigma_B \Sigma_C P(a, b, c, d)$

  $= \Sigma_A \Sigma_B \Sigma_C P(d|c)P(c|b)P(b|a)P(a)$

  $= \Sigma_C P(d|c) \Sigma_B P(c|b) \Sigma_A P(b|a)P(a)$

Inference by Enumeration

- General case:
  - Evidence variables: $E_1 \dots E_k = e_1 \dots e_k$
  - Query* variable: Q
  - Hidden variables: $H_1 \dots H_r$
- We want:

$$P(Q|e_1 \dots e_k)$$

Inference by Enumeration

- Step 1:
  - Select the entries consistent with the evidence

- Step 2:
  - Sum out H to get joint of Query and evidence

$$P(Q, e_1 \ldots e_k) = \sum_{h_1 \ldots h_r} P(Q, h_1 \ldots h_r, e_1 \ldots e_k)$$

- Step 3:
  - Normalize

$$Z = \sum_q P(Q, e_1 \cdots e_k) \qquad P(Q|e_1 \cdots e_k) = \frac{1}{Z} P(Q, e_1 \cdots e_k)$$

Inference

- Given a Bayesian network, what questions might we want to ask?
- **Conditional probability query**: P(X = x | e)
  - Given instantiations for some of the variables (we'll use e here to stand for the values of all the instantiated variables; it doesn't have to be just one), what is the probability that node X has a particular value x?
- **Maximum a posteriori probability**: $\text{argmax}_q \; P(Q = q | E_1 = e_1 \ldots)$
- **General question**: What's the whole probability distribution over variable X given evidence e, P(X | e)?

Compute Conditional Distribution

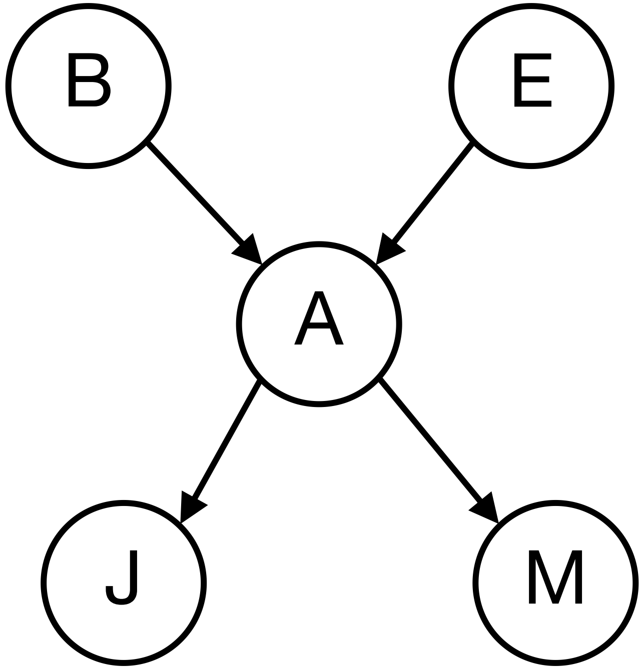
- To answer any query involving a conjunction of variables, sum over the variables not involved in the query

- $P(y|x) = P(x,y) / P(x)$

$$= \Sigma_{ABC} P(a,b,c,x,y) / \Sigma_{ABCY} P(a,b,c,x,y)$$

Inference by Enumeration in Bayes' Net

- Given unlimited time, inference in BNs is easy
- Inference by enumeration:



$$P(B \mid +j, +m) \propto_B P(B, +j, +m)$$

$$= \sum_{e,a} P(B, e, a, +j, +m)$$

$$= \sum_{e,a} P(B)P(e)P(a|B,e)P(+j|a)P(+m|a)$$

$$= P(B)P(+e)P(+a|B,+e)P(+j|+a)P(+m|+a) + P(B)P(+e)P(-a|B,+e)P(+j|-a)P(+m|-a)$$
$$P(B)P(-e)P(+a|B,-e)P(+j|+a)P(+m|+a) + P(B)P(-e)P(-a|B,-e)P(+j|-a)P(+m|-a)$$

D-Separation

- Query: $X_i \perp\!\!\!\perp X_j | \{X_{k_1}, ..., X_{k_n}\}$
- Check all (undirected!) paths between X_i and X_j
  - If one or more active, then independence not guaranteed

$$X_i \not\perp\!\!\!\perp X_j | \{X_{k_1}, ..., X_{k_n}\}$$

  - Otherwise (i.e. if all paths are inactive), then independence is guaranteed

$$X_i \perp\!\!\!\perp X_j | \{X_{k_1}, ..., X_{k_n}\}$$

Causal Chains

- This configuration is a "causal chain"

$$X \rightarrow Y \rightarrow Z \qquad P(x, y, z) = P(x)P(y|x)P(z|y)$$

X: Exam easy Y: Get A Z: Get recommended

- Guaranteed X independent of Z ? *No!*
  - One example set of CPTs for which X is not independent of Z is sufficient to show this independence is not guaranteed.
  - Example:
    - Eaxm easy causes Get A causes Get recommended.
    - In numbers:

      P(+y | +x) = 1, P(-y | -x) = 1,
      P(+z | +y) = 1, P(-z | -y) = 1

Causal Chains

- This configuration is a "causal chain"

$$X \rightarrow Y \rightarrow Z \qquad P(x,y,z) = P(x)P(y|x)P(z|y)$$

X: Exam easy   Y: Get A   Z: Get recommended

- Guaranteed X independent of Z given Y?

$$\begin{aligned} P(z|x,y) &= \frac{P(x,y,z)}{P(x,y)} \\ &= \frac{P(x)P(y|x)P(z|y)}{P(x)P(y|x)} \\ &= P(z|y) \end{aligned}$$

Yes!

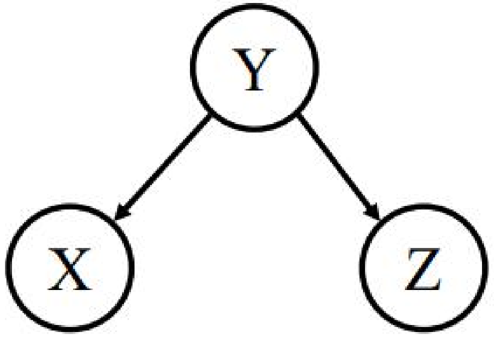
Evidence along the chain "blocks" the influence

Common Cause

- This configuration is a "common cause"

Y: Get A

X: Get reward



Z: Get recommended

- Guaranteed X independent of Z ? *No!*

$$P(x, y, z) = P(y)P(x|y)P(z|y)$$

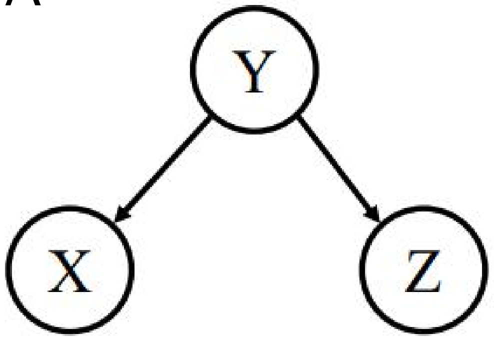
  - One example set of CPTs for which X is not independent of Z is sufficient to show this independence is not guaranteed.
  - Example:
    - Get A causes both Get reward and Get recommended
    - In numbers:
      - P(+x | +y) = 1, P(-x | -y) = 1,
      - P(+z | +y) = 1, P(-z | -y) = 1

Common Cause

- This configuration is a “common cause”

Y: Get A

X: Get reward



Z: Get recommended

$$P(x,y,z) = P(y)P(x|y)P(z|y)$$

- Guaranteed X and Z independent given Y?

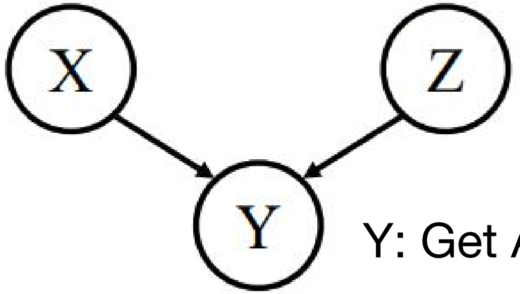
$$P(z|x,y) = \frac{P(x,y,z)}{P(x,y)} = \frac{P(y)P(x|y)P(z|y)}{P(y)P(x|y)} = P(z|y)$$

- Observing the cause blocks influence between effects.

Common Effect

- Last configuration: two causes of one effect (v-structures)

X: Exam easy



Z: Study hard

Y: Get A

- Are X and Z independent?
  - Yes: Exam easy and Study hard cause Get A, but they are not correlated
- Are X and Z independent given Y?
  - No: seeing Get A puts Exam easy and Study hard in competition as explanation.
- This is backwards from the other cases
  - Observing an effect activates influence between possible causes.

The General Case

- General question: in a given BN, are two variables independent (given evidence)?

- Solution: analyze the graph

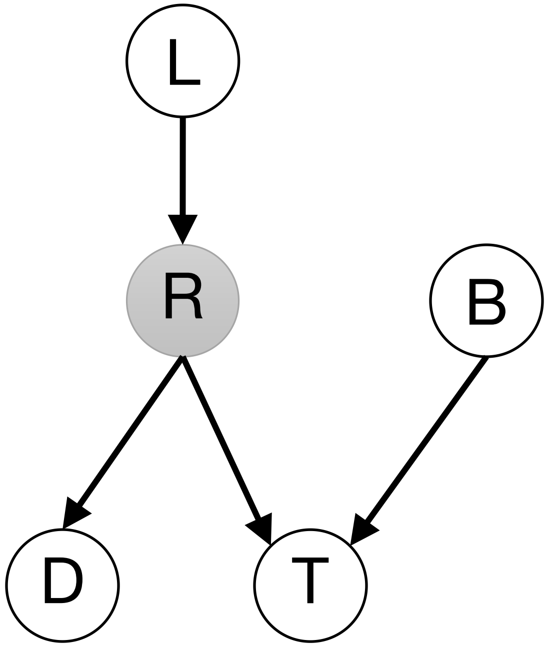
- Any complex example can be broken into repetitions of the three canonical cases

Reachability

- Recipe: shade evidence nodes, look for paths in the resulting graph

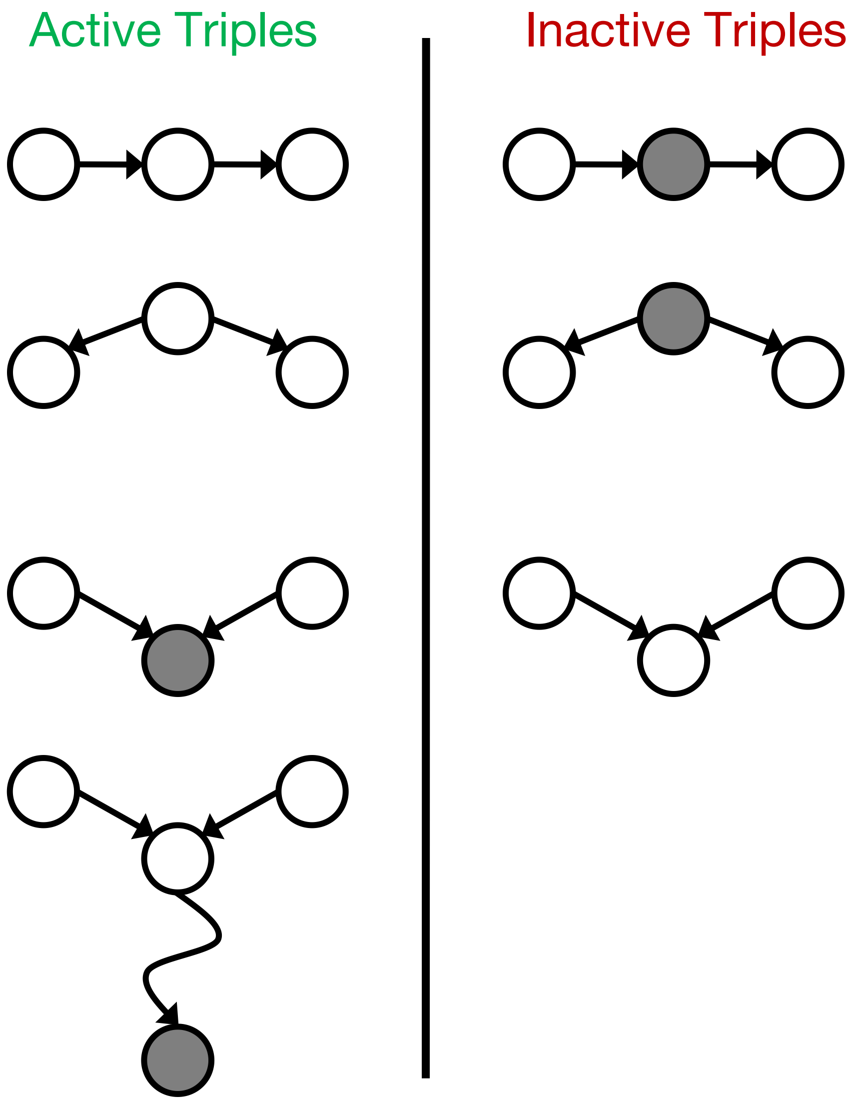
- Place balls on one of the variables

- If any ball can reach another random variable, then they are not conditionally independent, otherwise they are

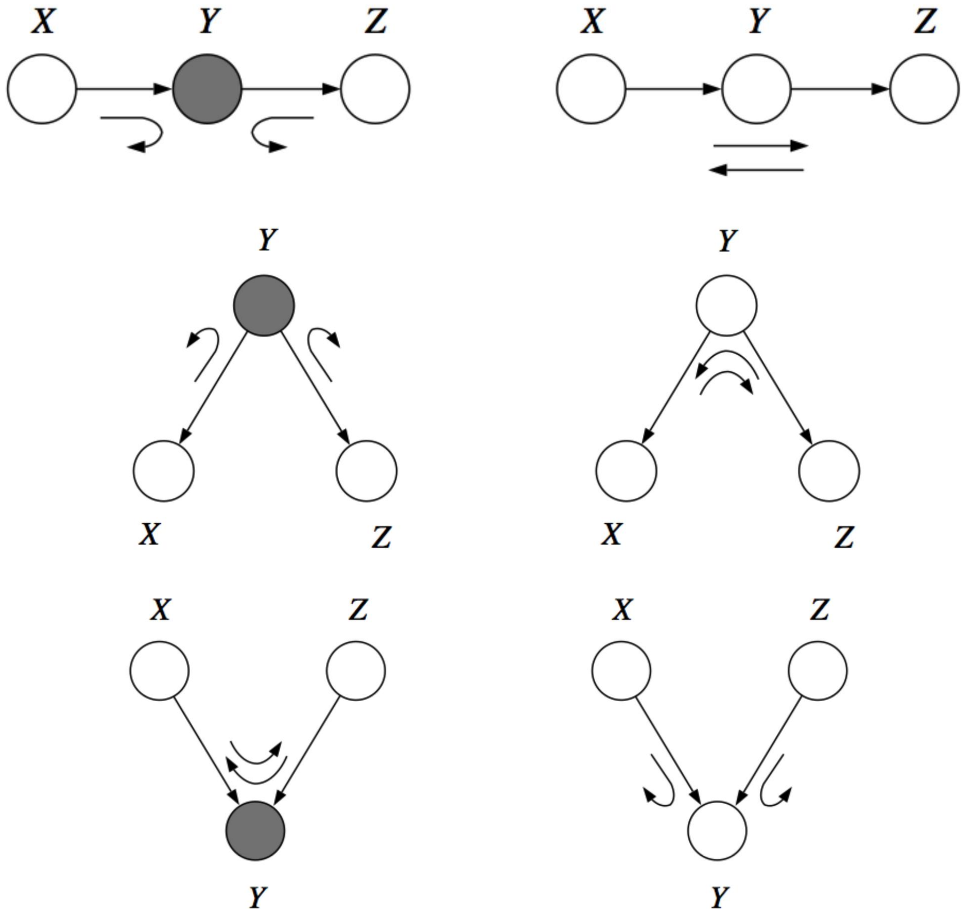


Active / Inactive Paths

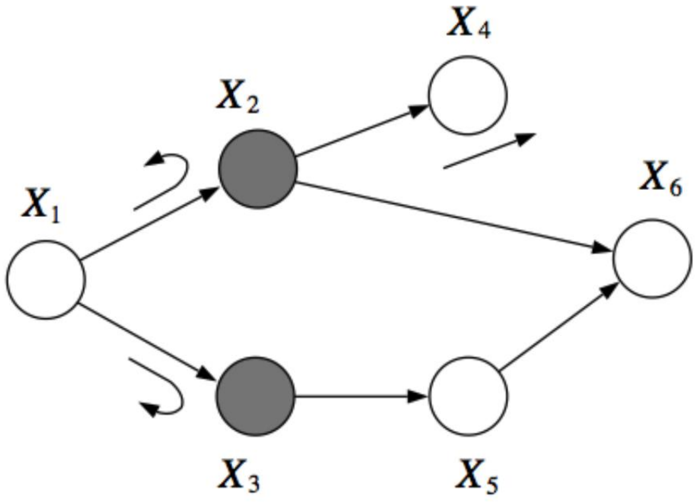
- **Question**: Are X and Y conditionally independent given evidence variables {Z}?
  - Yes, if X and Y "d-separated" by Z
  - Consider all (undirected) paths from X to Y
  - No active paths = independence!
- A path is active if each triple is active:
  - Causal chain A → B → C where B is unobserved (either direction)
  - Common cause A ← B → C where B is unobserved
  - Common effect (aka v-structure)
    - A → B ← C where B or one of its descendents is observed
- All it takes to block a path is a single inactive segment



Reachability

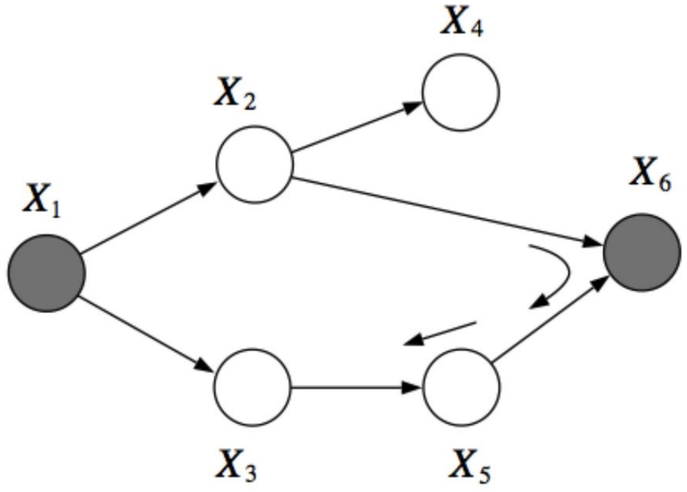


Reachability



- $(X_1 \perp\!\!\!\perp X_4 \mid X_2, X_3)$?

Reachability



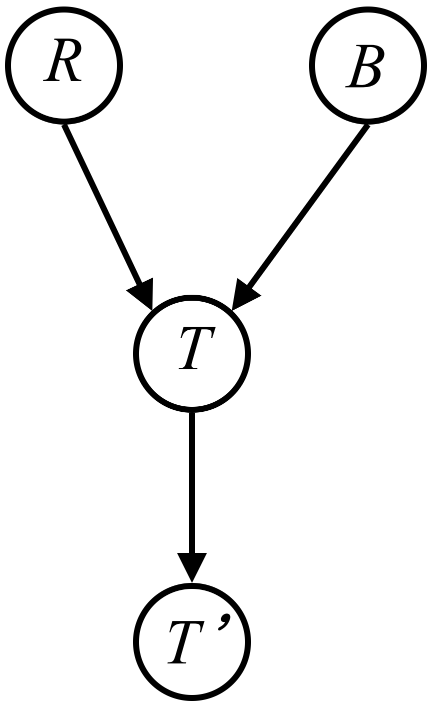
- $(X_2 \perp\!\!\!\perp X_3 \mid X_1, X_6)$?

Example

$R \perp\!\!\!\perp B$ Yes

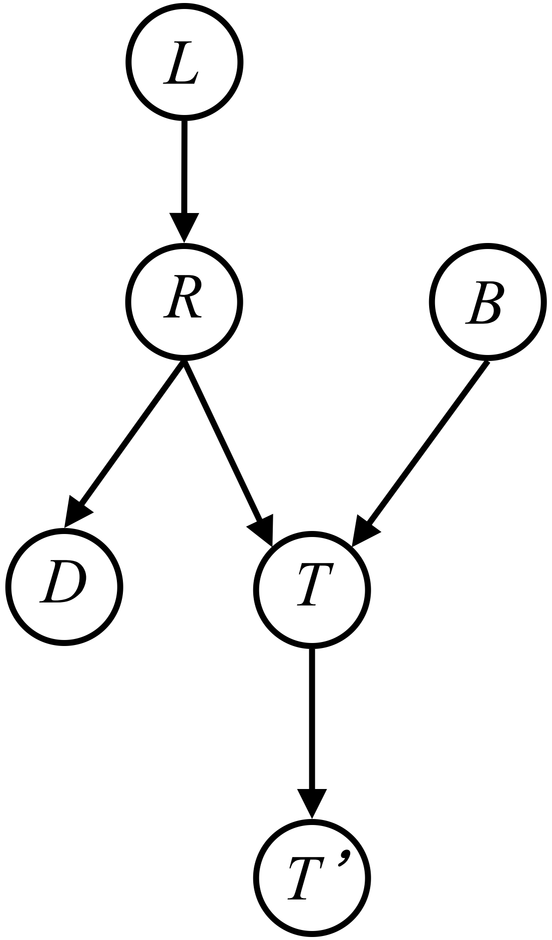
$R \perp\!\!\!\perp B | T$ No

$R \perp\!\!\!\perp B | T'$ No



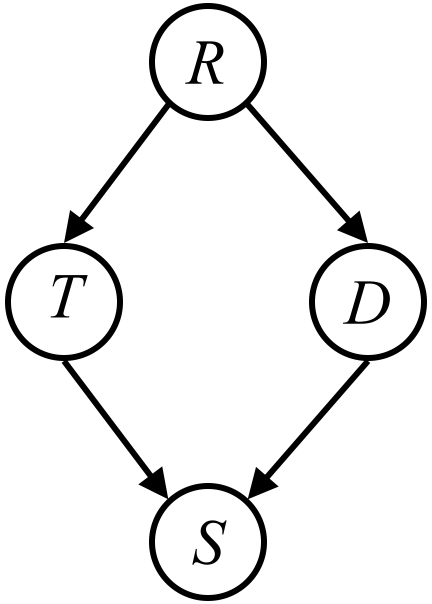
Example

$L \perp\!\!\!\perp T' \mid T$	Yes
$L \perp\!\!\!\perp B$	Yes
$L \perp\!\!\!\perp B \mid T$	No
$L \perp\!\!\!\perp B \mid T'$	No
$L \perp\!\!\!\perp B \mid T, R$	Yes



Example

- Variables:
  - R: Raining
  - T: Traffic
  - D: Roof drips
  - S: I'm sad



- Questions:

$T \perp\!\!\!\perp D$	No
$T \perp\!\!\!\perp D \| R$	Yes
$T \perp\!\!\!\perp D \| R, S$	No

Thank You!