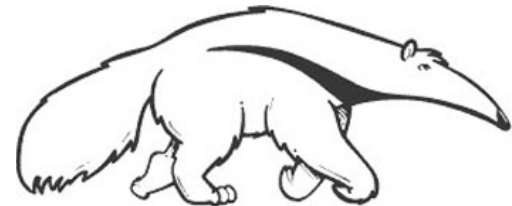


Introduction to Artificial Intelligence

CS171, Summer 1 Quarter, 2019
Introduction to Artificial Intelligence
Prof. Richard Lathrop



Read Beforehand: All assigned reading so far

CS-171 Midterm Review

- **Agents**

- (R&N Ch. 1-2, 26.preamble, 26.3-4, 27.4)

- **Propositional Logic**

- (R&N Ch. 7.1-7.5)

- **First-Order Logic**

- (R&N Ch. 8.1-8.5, 9.1-9.2)

- **Probability & Bayesian Networks**

- (R&N Ch. 13, 14.1-14.5)

- **Hidden Markov Models**

- (R&N Ch. 5.1-15.3)

- Questions on any topic

- Please review your quizzes & old test

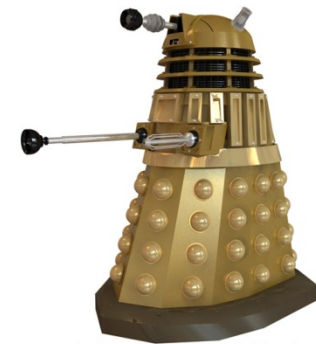
Review Agents

Chapter 2.1-2.3

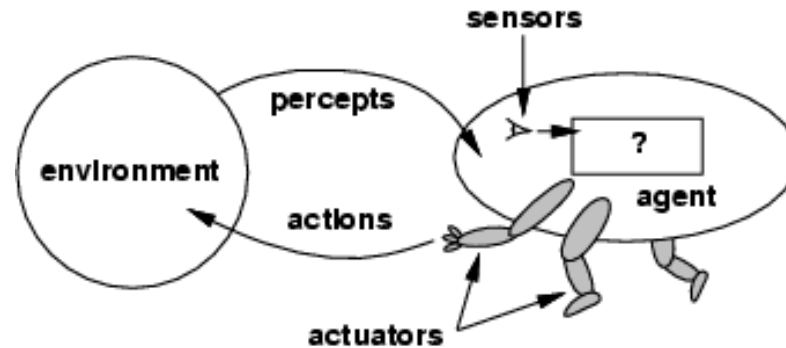
- Agent definition (2.1)
- Rational Agent definition (2.2)
 - Performance measure
- Task environment definition (2.3)
 - PEAS acronym
 - Properties of task environments

Agents

- An **agent** is anything that can be viewed as **perceiving** its **environment** through **sensors** and **acting** upon that environment through **actuators**
- Human agent:
 - Sensors: eyes, ears, ...
 - Actuators: hands, legs, mouth...
- Robotic agent
 - Sensors: cameras, range finders, ...
 - Actuators: motors



Agents and environments



- **Percept**: agent's perceptual inputs at an instant
- The **agent function** maps from percept sequences to actions: $[f: P^* \rightarrow \mathcal{A}]$
- The **agent program** runs on the physical **architecture** to produce f
- **agent = architecture + program**

Rational agents

- **Rational Agent:** For each possible percept sequence, a rational agent should select an action that is *expected* to maximize its *performance measure*, based on the evidence provided by the percept sequence and whatever built-in knowledge the agent has.
- **Performance measure:** An objective criterion for success of an agent's behavior (“cost”, “reward”, “utility”)
- **E.g.**, performance measure of a vacuum-cleaner agent could be amount of dirt cleaned up, amount of time taken, amount of electricity consumed, amount of noise generated, etc.

Task Environment

- Before we design an intelligent agent, we must specify its “task environment”:

PEAS:

Performance measure

Environment

Actuators

Sensors

Environment types

- **Fully observable** (vs. **partially observable**): An agent's sensors give it access to the complete state of the environment at each point in time.
- **Deterministic** (vs. **stochastic**): The next state of the environment is completely determined by the current state and the action executed by the agent. (If the environment is deterministic except for the actions of other agents, then the environment is **strategic**.)
- **Episodic** (vs. **sequential**): An agent's action is divided into atomic episodes. Decisions do not depend on previous decisions/actions.
- **Known** (vs. **unknown**): An environment is considered to be "known" if the agent understands the laws that govern the environment's behavior.

Environment types

- **Static** (vs. **dynamic**): The environment is unchanged while an agent is deliberating. (The environment is **semidynamic** if the environment itself does not change with the passage of time but the agent's performance score does)
- **Discrete** (vs. **continuous**): A limited number of distinct, clearly defined percepts and actions.
 - How do we **represent** or **abstract** or **model** the world?
- **Single agent** (vs. **multi-agent**): An agent operating by itself in an environment. Does the other agent interfere with my performance measure?

CS-171 Midterm Review

- **Agents**
 - (R&N Ch. 1-2, 26.preamble, 26.3-4, 27.4)
- **Propositional Logic**
 - (R&N Ch. 7.1-7.5)
- **First-Order Logic**
 - (R&N Ch. 8.1-8.5, 9.1-9.2)
- **Probability & Bayesian Networks**
 - (R&N Ch. 13, 14.1-14.5)
- **Hidden Markov Models**
 - (R&N Ch. 5.1-15.3)
- Questions on any topic
- Please review your quizzes & old test

Review Propositional Logic

Chapter 7.1-7.5; Optional 7.6-7.8

- Definitions:
 - Syntax, Semantics, Sentences, Propositions, Entails, Follows, Derives, Inference, Sound, Complete, Model, Satisfiable, Valid (or Tautology)
- Syntactic & Semantic Transformations:
 - E.g., $(A \Rightarrow B) \Leftrightarrow (\neg A \vee B)$
 - E.g., $(KB \models \alpha) \equiv (\models (KB \Rightarrow \alpha))$
- Truth Tables:
 - Negation, Conjunction, Disjunction, Implication, Equivalence (Biconditional)
- Inference:
 - By Resolution (CNF)
 - By Backward & Forward Chaining (Horn Clauses)
 - By Model Enumeration (Truth Tables)

Recap propositional logic: **Syntax**

- Propositional logic is the simplest logic – illustrates basic ideas
- The proposition symbols P_1, P_2 etc are sentences
 - If S is a sentence, $\neg S$ is a sentence (**negation**)
 - If S_1 and S_2 are sentences, $S_1 \wedge S_2$ is a sentence (**conjunction**)
 - If S_1 and S_2 are sentences, $S_1 \vee S_2$ is a sentence (**disjunction**)
 - If S_1 and S_2 are sentences, $S_1 \Rightarrow S_2$ is a sentence (**implication**)
 - If S_1 and S_2 are sentences, $S_1 \Leftrightarrow S_2$ is a sentence (**biconditional**)

Recap propositional logic:

Semantics

Each model/world specifies true or false for each proposition symbol

E.g., $P_{1,2}$ $P_{2,2}$ $P_{3,1}$
 false true false

With these symbols, 8 possible models can be enumerated automatically.

Rules for evaluating truth with respect to a model m :

$\neg S$	is true iff	S is false	
$S_1 \wedge S_2$	is true iff	S_1 is true and	S_2 is true
$S_1 \vee S_2$	is true iff	S_1 is true or	S_2 is true
$S_1 \Rightarrow S_2$	is true iff	S_1 is false or	S_2 is true
(i.e.,	is false iff	S_1 is true and	S_2 is false
$S_1 \Leftrightarrow S_2$	is true iff	$S_1 \Rightarrow S_2$ is true and	$S_2 \Rightarrow S_1$ is true

Simple recursive process evaluates an arbitrary sentence, e.g.,

$$\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1}) = \text{true} \wedge (\text{true} \vee \text{false}) = \text{true} \wedge \text{true} = \text{true}$$

Recap propositional logic:

Truth tables for connectives

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>true</i>
<i>false</i>	<i>true</i>	<i>true</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>false</i>
<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>
<i>true</i>	<i>true</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>true</i>

OR: P or Q is true or both are true.
XOR: P or Q is true but not both.


Implication is always true
when the premises are False!

Recap propositional logic:

Logical equivalence and rewrite rules

- To manipulate logical sentences we need some rewrite rules.
- Two sentences are **logically equivalent** iff they are true in same models: $\alpha \equiv \beta$ iff $\alpha \models \beta$ and $\beta \models \alpha$

$$\begin{aligned}(\alpha \wedge \beta) &\equiv (\beta \wedge \alpha) && \text{commutativity of } \wedge \\(\alpha \vee \beta) &\equiv (\beta \vee \alpha) && \text{commutativity of } \vee \\((\alpha \wedge \beta) \wedge \gamma) &\equiv (\alpha \wedge (\beta \wedge \gamma)) && \text{associativity of } \wedge \\((\alpha \vee \beta) \vee \gamma) &\equiv (\alpha \vee (\beta \vee \gamma)) && \text{associativity of } \vee \\ \neg(\neg\alpha) &\equiv \alpha && \text{double-negation elimination} \\(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) && \text{contraposition} \\(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \vee \beta) && \text{implication elimination} \\(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) && \text{biconditional elimination} \\ \neg(\alpha \wedge \beta) &\equiv (\neg\alpha \vee \neg\beta) && \text{de Morgan} \\ \neg(\alpha \vee \beta) &\equiv (\neg\alpha \wedge \neg\beta) && \text{de Morgan} \\(\alpha \wedge (\beta \vee \gamma)) &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) && \text{distributivity of } \wedge \text{ over } \vee \\(\alpha \vee (\beta \wedge \gamma)) &\equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) && \text{distributivity of } \vee \text{ over } \wedge\end{aligned}$$



You need to know these !

Entailment

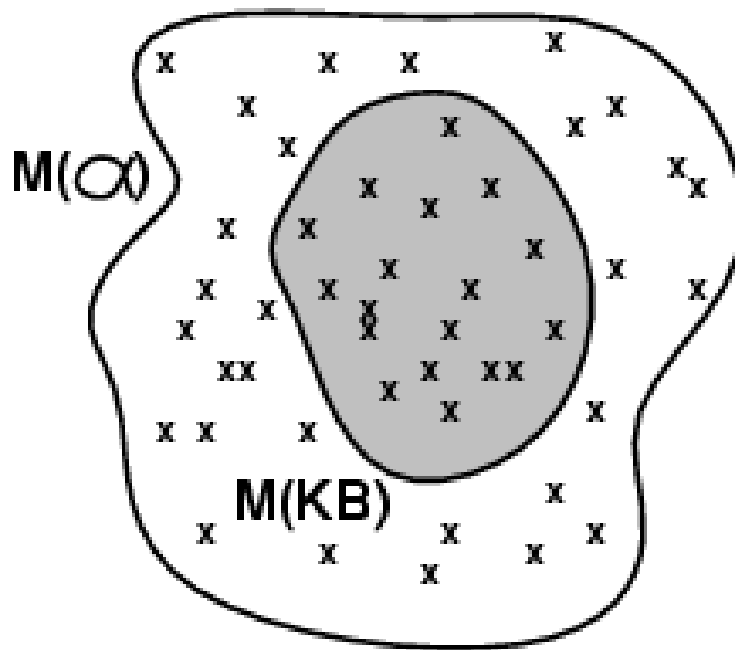
- **Entailment** means that one thing **follows from** another set of things:

$$KB \models \alpha$$

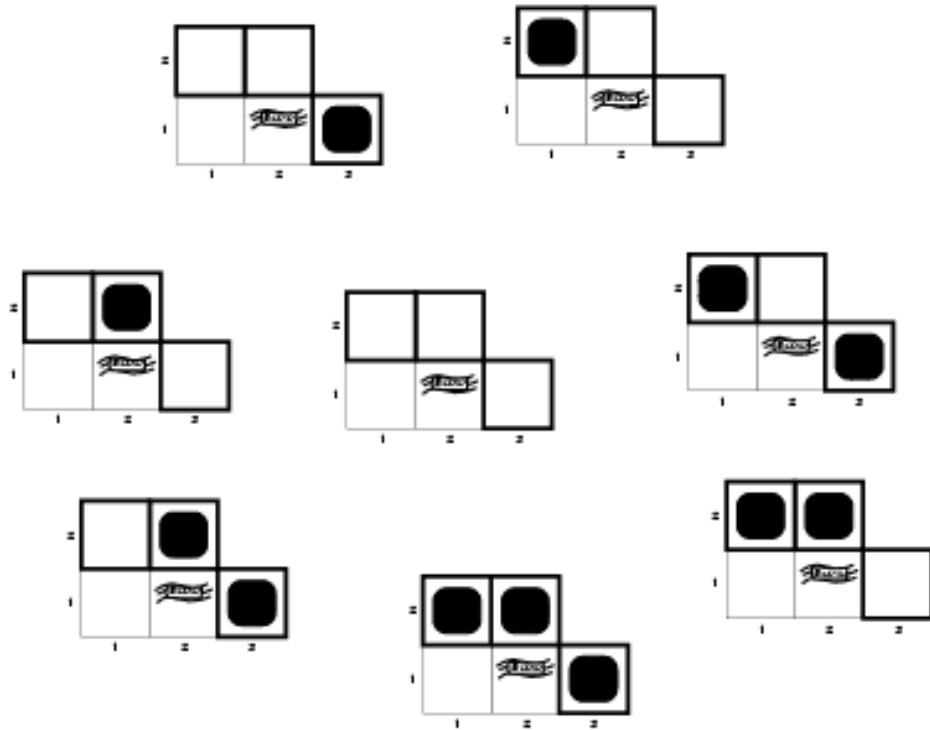
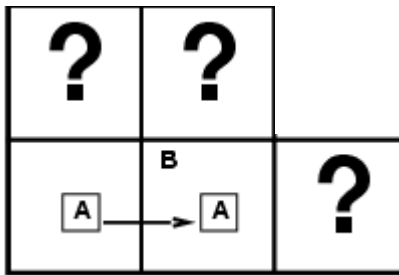
- Knowledge base *KB* entails sentence α if and only if α is true in **all worlds** wherein *KB* is true
 - E.g., the *KB* = “the Giants won and the Reds won” entails α = “The Giants won”.
 - E.g., *KB* = “ $x+y = 4$ ” entails α = “ $4 = x+y$ ”
 - E.g., *KB* = “Mary is Sue’s sister and Amy is Sue’s daughter” entails α = “Mary is Amy’s aunt.”
- The entailed α MUST BE TRUE in ANY world in which KB IS TRUE.

Review: Models (and in FOL, Interpretations)

- **Models** are formal worlds in which truth can be evaluated
- We say m is a **model of** a sentence α if α is true in m
- $M(\alpha)$ is the set of all models of α
- Then $KB \models \alpha$ iff $M(KB) \subseteq M(\alpha)$
 - E.g. KB , = “Mary is Sue’s sister and Amy is Sue’s daughter.”
 - α = “Mary is Amy’s aunt.”
- Think of KB and α as constraints, and of models m as possible states.
- $M(KB)$ are the solutions to KB and $M(\alpha)$ the solutions to α .
- Then, $KB \models \alpha$, i.e., $\models (KB \Rightarrow \alpha)$, when all solutions to KB are also solutions to α .

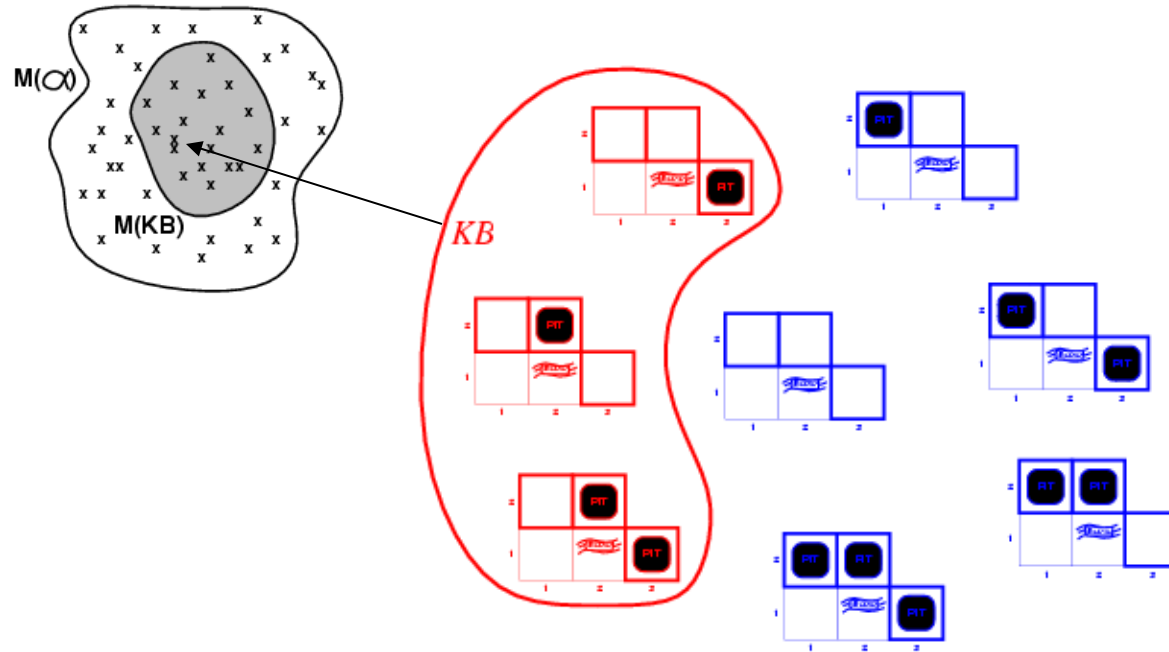


Wumpus models



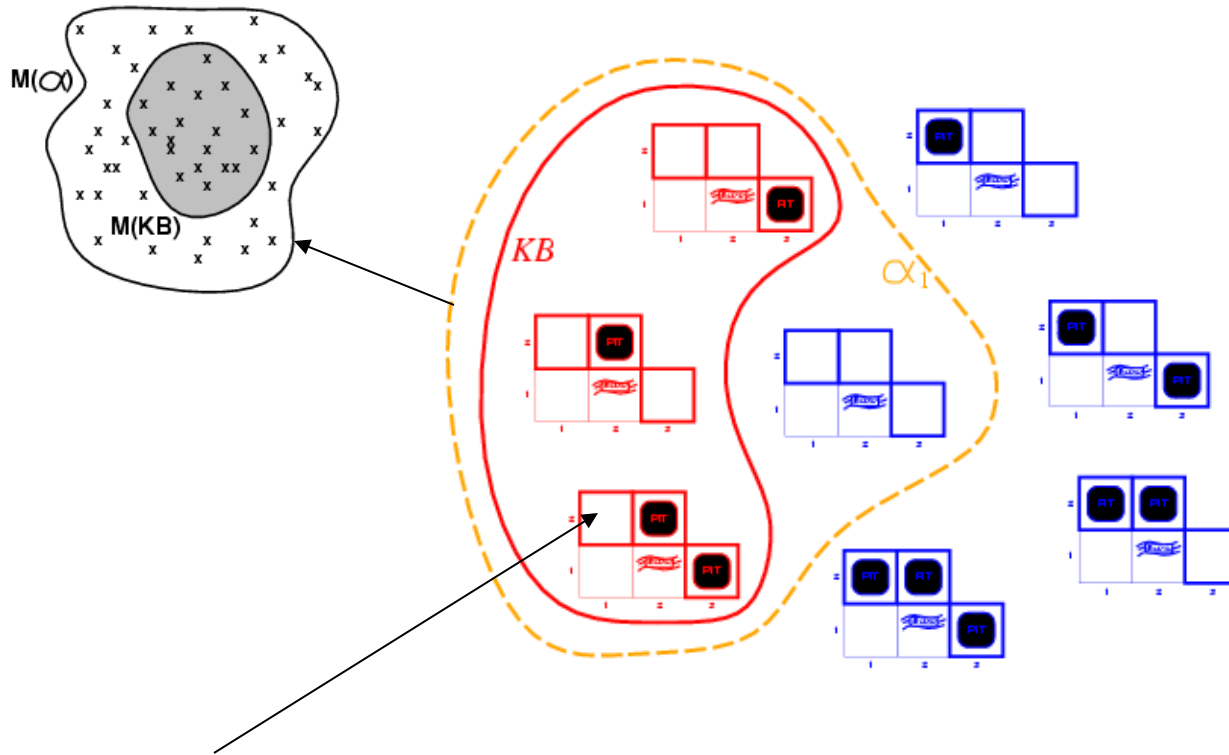
All possible models in this reduced Wumpus world. What can we infer?

Review: Wumpus models



- KB = all possible wumpus-worlds consistent with the observations and the “physics” of the Wumpus world.

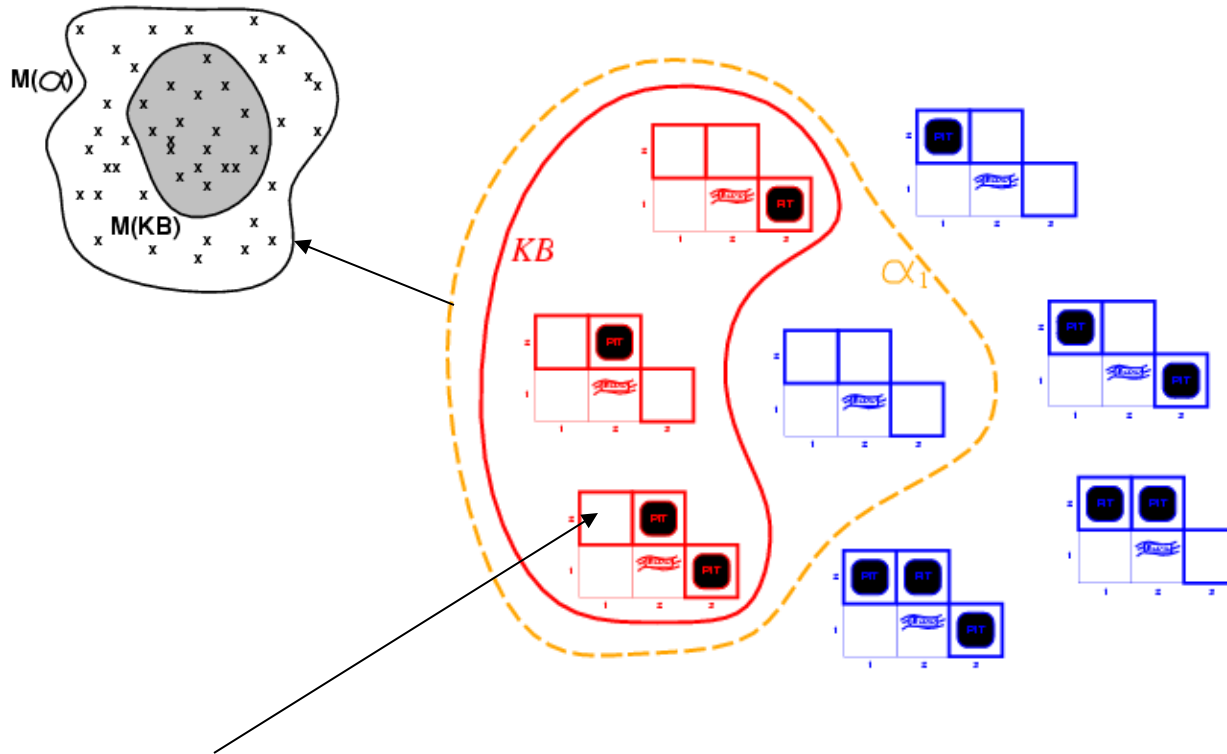
Review: Wumpus models



$\alpha_1 = "[1,2] \text{ is safe}"$, $KB \models \alpha_1$, proved by **model checking**.

Every model that makes KB true also makes α_1 true.

Wumpus models



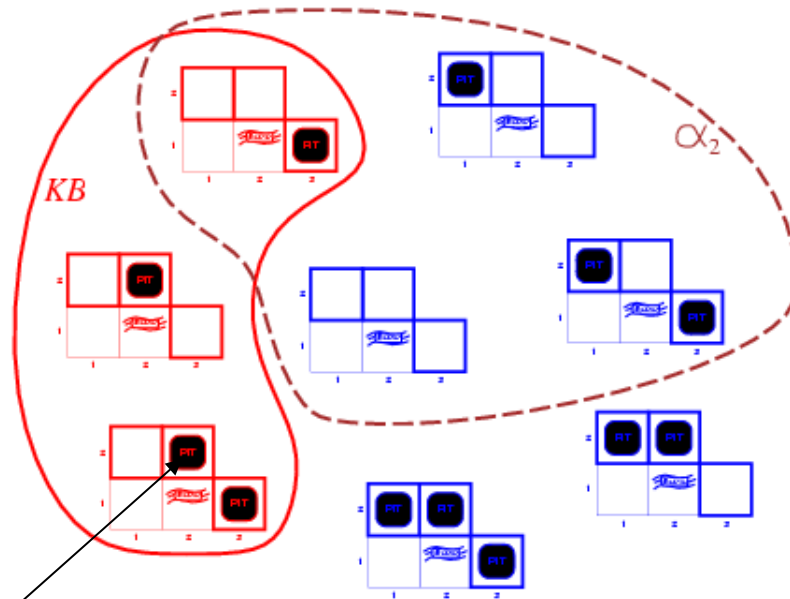
Now we have a query sentence, $\alpha_1 = "[1,2] \text{ is safe}"$

$KB \models \alpha_1$, proved by **model checking**

$M(KB)$ (red outline) is a subset of $M(\alpha_1)$ (orange dashed outline)

$\Rightarrow \alpha_1$ is true in any world in which KB is true

Wumpus models



Now we have another query sentence, $\alpha_2 = "[2,2] \text{ is safe}"$

$KB \not\models \alpha_2$, proved by **model checking**

$M(KB)$ (red outline) is a not a subset of $M(\alpha_2)$ (dashed outline)

$\Rightarrow \alpha_2$ is false in some world(s) in which KB is true

Recap propositional logic: **Validity and satisfiability**

A sentence is **valid** if it is true in **all** models,

e.g., True , $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$

Validity is connected to inference via the **Deduction Theorem**:

$KB \models \alpha$ if and only if $(KB \Rightarrow \alpha)$ is valid

A sentence is **satisfiable** if it is true in **some** model

e.g., $A \vee B$, C

A sentence is **unsatisfiable** if it is **false** in **all** models

e.g., $A \wedge \neg A$

Satisfiability is connected to inference via the following:

$KB \models A$ if and only if $(KB \wedge \neg A)$ is unsatisfiable

(there is no model for which KB is true and A is false)

Logical inference

- The notion of entailment can be used for logic inference.
 - Model checking (see wumpus example):
enumerate all possible models and check whether α is true.
- $KB \vdash_i \alpha$ means KB derives a sentence α using inference procedure i
- Sound (or *truth preserving*):
The algorithm only derives entailed sentences.
 - Otherwise it just makes things up.
 i is sound iff whenever $KB \vdash_i \alpha$ it is also true that $KB \models \alpha$
 - E.g., model-checking is sound
Refusing to infer any sentence is Sound; so, Sound is weak alone.
- Complete:
The algorithm can derive every entailed sentence.
 i is complete iff whenever $KB \models \alpha$ it is also true that $KB \vdash_i \alpha$
Deriving every sentence is Complete; so, Complete is weak alone.

Inference by Resolution

- KB is represented in CNF
 - KB = AND of all the sentences in KB
 - KB sentence = clause = OR of literals
 - Literal = propositional symbol or its negation
- Find two clauses in KB, one of which contains a literal and the other its negation
 - Cancel the literal and its negation
 - Bundle everything else into a new clause
 - Add the new clause to KB
 - Repeat

Example: Conversion to CNF

Example: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

1. Eliminate \Leftrightarrow by replacing $\alpha \Leftrightarrow \beta$ with $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.
 $= (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$
2. Eliminate \Rightarrow by replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$ and simplify.
 $= (\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$
3. Move \neg inwards using de Morgan's rules and simplify.
 $\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta), \neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$
 $= (\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$
4. Apply distributive law (\wedge over \vee) and simplify.
 $= (\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Example: Conversion to CNF

Example: $B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

From the previous slide we had:

$$= (\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$$

5. KB is the conjunction of all of its sentences (all are true), so write each clause (disjunct) as a sentence in KB:

KB =

...

$$(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1})$$

$$(\neg P_{1,2} \vee B_{1,1})$$

$$(\neg P_{2,1} \vee B_{1,1})$$

...



(same)

Often, Won't Write “ \vee ” or “ \wedge ”
(we know they are there)

$$\begin{array}{l} (\neg B_{1,1} \quad P_{1,2} \quad P_{2,1}) \\ (\neg P_{1,2} \quad B_{1,1}) \\ (\neg P_{2,1} \quad B_{1,1}) \end{array}$$

Resolution = Efficient Implication

Recall that $(A \Rightarrow B) = ((\text{NOT } A) \text{ OR } B)$

and so:

$$\begin{aligned} (Y \text{ OR } X) &= ((\text{NOT } X) \Rightarrow Y) \\ ((\text{NOT } Y) \text{ OR } Z) &= (Y \Rightarrow Z) \end{aligned}$$

which yields:

$$((Y \text{ OR } X) \text{ AND } ((\text{NOT } Y) \text{ OR } Z)) = ((\text{NOT } X) \Rightarrow Z) = (X \text{ OR } Z)$$

(OR A B C D)

(OR \neg A E F G)

->Same ->

->Same ->

(NOT (OR B C D)) \Rightarrow A

A \Rightarrow (OR E F G)

(OR B C D E F G)

(NOT (OR B C D)) \Rightarrow (OR E F G)

(OR B C D E F G)

Recall: All clauses in KB are conjoined by an implicit AND (= CNF representation).

Resolution Examples

- **Resolution:** inference rule for CNF: **sound and complete!** *

$(A \vee B \vee C)$

$(\neg A)$

“If A or B or C is true, but not A, then B or C must be true.”

 $\therefore (B \vee C)$

$(A \vee B \vee C)$

$(\neg A \vee D \vee E)$

“If A is false then B or C must be true, or if A is true then D or E must be true, hence since A is either true or false, B or C or D or E must be true.”

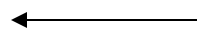
 $\therefore (B \vee C \vee D \vee E)$

$(A \vee B)$

$(\neg A \vee B)$

“If A or B is true, and not A or B is true, then B must be true.”

 $\therefore (B \vee B) \equiv B$



Simplification
is done always.

* Resolution is “refutation complete” in that it can prove the truth of any entailed sentence by refutation.

More Resolution Examples

1. $(P \ Q \ \neg R \ S)$ with $(P \ \neg Q \ W \ X)$ yields $(P \ \neg R \ S \ W \ X)$
Order of literals within clauses does not matter.
2. $(P \ Q \ \neg R \ S)$ with $(\neg P)$ yields $(Q \ \neg R \ S)$
3. $(\neg R)$ with (R) yields $(\)$ or FALSE
4. $(P \ Q \ \neg R \ S)$ with $(P \ R \ \neg S \ W \ X)$ yields $(P \ Q \ \neg R \ R \ W \ X)$ or $(P \ Q \ S \ \neg S \ W \ X)$ or TRUE
5. $(P \ \neg Q \ R \ \neg S)$ with $(P \ \neg Q \ R \ \neg S)$ yields None possible (no complementary literals)
6. $(P \ \neg Q \ \neg S \ W)$ with $(P \ R \ \neg S \ X)$ yields None possible (no complementary literals)
7. $((\neg A) (\neg B) (\neg C) (\neg D))$ with $((\neg C) D)$ yields $((\neg A) (\neg B) (\neg C))$
8. $((\neg A) (\neg B) (\neg C))$ with $((\neg A) C)$ yields $((\neg A) (\neg B))$
9. $((\neg A) (\neg B))$ with (B) yields $(\neg A)$
10. $(A \ C)$ with $(A \ \neg C)$ yields (A)
11. $(\neg A)$ with (A) yields $(\)$ or FALSE

Only Resolve ONE Literal Pair!

If more than one pair, result always = TRUE.

Useless!! Always simplifies to TRUE!!

No!

(OR (A B) C D)
(OR \neg A \neg B F G)

(OR C D F G)

No! This is wrong!

No!

(OR (A B C) D)
(OR \neg A \neg B \neg C)

(OR D)

No! This is wrong!

Yes! (but = TRUE)

(OR (A) B C D)
(OR \neg A \neg B F G)

(OR B \neg B C D F G)

Yes! (but = TRUE)

Yes! (but = TRUE)

(OR A B (C) D)
(OR \neg A \neg B \neg C)

(OR A \neg A B \neg B D)

Yes! (but = TRUE)

Resolution Algorithm

- The resolution algorithm tries to prove: $KB \models \alpha$ equivalent to $KB \wedge \neg\alpha$ unsatisfiable
- Generate all new sentences from KB and the (negated) query.
- One of two things can happen:
 1. We find $P \wedge \neg P$ which is unsatisfiable. I.e. we can entail the query.
 2. We find no contradiction: there is a model that satisfies the sentence $KB \wedge \neg\alpha$ (non-trivial) and hence we cannot entail the query.

Resolution example

Resulting Knowledge Base stated in CNF

- “Laws of Physics” in the Wumpus World:

$$\begin{aligned} & (\neg B_{1,1} \quad P_{1,2} \quad P_{2,1}) \\ & (\neg P_{1,2} \quad B_{1,1}) \\ & (\neg P_{2,1} \quad B_{1,1}) \end{aligned}$$

- Particular facts about a specific instance:

$$(\neg B_{1,1})$$

- Negated goal or query sentence:

$$(P_{1,2})$$

Resolution example

A Resolution proof ending in ()

- Knowledge Base at start of proof:

$(\neg B_{1,1} \quad P_{1,2} \quad P_{2,1})$

$(\neg P_{1,2} \quad B_{1,1})$

$(\neg P_{2,1} \quad B_{1,1})$

$(\neg B_{1,1})$

$(P_{1,2})$

A resolution proof ending in ():

- Resolve $(\neg P_{1,2} \quad B_{1,1})$ and $(\neg B_{1,1})$ to give $(\neg P_{1,2})$
- Resolve $(\neg P_{1,2})$ and $(P_{1,2})$ to give ()
- Consequently, the goal or query sentence is entailed by KB.
- Of course, there are many other proofs, which are OK iff correct.

Detailed Resolution Proof Example

- **In words:** *If the unicorn is mythical, then it is immortal, but if it is not mythical, then it is a mortal mammal. If the unicorn is either immortal or a mammal, then it is horned. The unicorn is magical if it is horned.*

Prove that the unicorn is both magical and horned.

$(\neg Y) (\neg R) \quad (M Y) \quad (R Y) \quad (H (\neg M))$
 $(H R) \quad (\neg H) G \quad (\neg G) (\neg H)$

- **Fourth, produce a resolution proof ending in ():**
- Resolve $(\neg H \neg G)$ and $(\neg H G)$ to give $(\neg H)$
- Resolve $(\neg Y \neg R)$ and $(Y M)$ to give $(\neg R M)$
- Resolve $(\neg R M)$ and $(R H)$ to give $(M H)$
- Resolve $(M H)$ and $(\neg M H)$ to give (H)
- Resolve $(\neg H)$ and (H) to give $()$
- Of course, there are many other proofs, which are OK iff correct.

Horn Clauses

- Resolution can be exponential in space and time.
- If we can reduce all clauses to “Horn clauses” inference is linear in space and time

A clause with at most 1 positive literal.

e.g. $A \vee \neg B \vee \neg C$

- Every Horn clause can be rewritten as an implication with a conjunction of positive literals in the premises and at most a single positive literal as a conclusion.

e.g. $A \vee \neg B \vee \neg C \equiv B \wedge C \Rightarrow A$

- 1 positive literal and ≥ 1 negative literal: definite clause (e.g., above)
- 0 positive literals: integrity constraint or goal clause
e.g. $(\neg A \vee \neg B) \equiv (A \wedge B \Rightarrow \text{False})$ states that $(A \wedge B)$ must be false
- 0 negative literals: fact
e.g., $(A) \equiv (\text{True} \Rightarrow A)$ states that A must be true.
- Forward Chaining and Backward chaining are sound and complete with Horn clauses and run linear in space and time.

Propositional Logic --- Summary

- Logical agents apply inference to a knowledge base to derive new information and make decisions
- Basic concepts of logic:
 - syntax: formal structure of sentences
 - semantics: truth of sentences wrt models
 - entailment: necessary truth of one sentence given another
 - inference: deriving sentences from other sentences
 - soundness: derivations produce only entailed sentences
 - completeness: derivations can produce all entailed sentences
 - valid: sentence is true in every model (a tautology)
- Logical equivalences allow syntactic manipulations
- Propositional logic lacks expressive power
 - Can only state specific facts about the world.
 - Cannot express general rules about the world
(use First Order Predicate Logic instead)

CS-171 Midterm Review

- **Agents**
 - (R&N Ch. 1-2, 26.preamble, 26.3-4, 27.4)
- **Propositional Logic**
 - (R&N Ch. 7.1-7.5)
- **First-Order Logic**
 - (R&N Ch. 8.1-8.5, 9.1-9.2)
- **Probability & Bayesian Networks**
 - (R&N Ch. 13, 14.1-14.5)
- **Hidden Markov Models**
 - (R&N Ch. 5.1-15.3)
- Questions on any topic
- Please review your quizzes & old test

Review First-Order Logic

Chapter 8.1-8.5, 9.1-9.2, 9.5.1-9.5.5

- Syntax & Semantics
 - Predicate symbols, function symbols, constant symbols, variables, quantifiers.
 - Models, symbols, and interpretations
- De Morgan's rules for quantifiers
- Nested quantifiers
 - Difference between " $\forall x \exists y P(x, y)$ " and " $\exists x \forall y P(x, y)$ "
- Translate simple English sentences to FOL and back
 - $\forall x \exists y \text{ Likes}(x, y) \Leftrightarrow$ "Everyone has someone that they like."
 - $\exists x \forall y \text{ Likes}(x, y) \Leftrightarrow$ "There is someone who likes every person."
- Unification and the Most General Unifier
- Inference in FOL
 - By Resolution (CNF)

Syntax of FOL: Basic syntax elements are symbols

- **Constant** Symbols (correspond to English nouns)
 - Stand for objects in the world.
 - E.g., KingJohn, 2, UCI, ...
- **Predicate** Symbols (correspond to English verbs)
 - Stand for relations (**maps a tuple of objects to a truth-value**)
 - E.g., Brother(Richard, John), greater_than(3,2), ...
 - $P(x, y)$ is usually read as “ x is P of y .”
 - E.g., Mother(Ann, Sue) is usually “Ann is Mother of Sue.”
- **Function** Symbols (correspond to English nouns)
 - Stand for functions (**maps a tuple of objects to an object**)
 - E.g., Sqrt(3), LeftLegOf(John), ...
- **Model** (world) = set of domain objects, relations, functions
- **Interpretation** maps symbols onto the model (world)
 - Very many interpretations are possible for each KB and world!
 - The KB is to rule out those inconsistent with our knowledge.

Syntax of FOL: Terms

- **Term** = logical expression that **refers to an object**
- **There are two kinds of terms:**
 - **Constant Symbols** stand for (or name) objects:
 - E.g., KingJohn, 2, UCI, Wumpus, ...
 - **Function Symbols** map tuples of objects to an object:
 - E.g., LeftLeg(KingJohn), Mother(Mary), Sqrt(x)
 - This is nothing but a complicated kind of name
 - No “subroutine” call, no “return value”

Syntax of FOL: Atomic Sentences

- **Atomic Sentences** state facts (logical truth values).
 - An **atomic sentence** is a Predicate symbol, optionally followed by a parenthesized list of any argument terms
 - E.g., *Married(Father(Richard), Mother(John))*
 - An **atomic sentence** asserts that some relationship (some predicate) holds among the objects that are its arguments.
- An **Atomic Sentence is true** in a given model if the relation referred to by the predicate symbol holds among the objects (terms) referred to by the arguments.

Syntax of FOL:

Connectives & Complex Sentences

- **Complex Sentences** are formed in the same way, using the same logical connectives, as in propositional logic
- **The Logical Connectives:**
 - \Leftrightarrow biconditional
 - \Rightarrow implication
 - \wedge and
 - \vee or
 - \neg negation
- **Semantics** for these logical connectives are the same as we already know from propositional logic.

Syntax of FOL: Variables

- **Variables** range over objects in the world.
- A **variable** is like a **term** because it represents an object.
- A **variable** may be used wherever a **term** may be used.
 - **Variables** may be arguments to functions and predicates.
- (A **term with NO variables** is called a **ground term**.)
- (A **variable not bound by a quantifier** is called **free**.)
 - All variables we will use are bound by a quantifier.

Syntax of FOL: Logical Quantifiers

- There are two **Logical Quantifiers**:
 - **Universal:** $\forall x P(x)$ means “For all x , $P(x)$.”
 - The “upside-down A” reminds you of “ALL.”
 - Some texts put a comma after the variable: $\forall x, P(x)$
 - **Existential:** $\exists x P(x)$ means “There exists x such that, $P(x)$.”
 - The “backward E” reminds you of “EXISTS.”
 - Some texts put a comma after the variable: $\exists x, P(x)$
- You can **ALWAYS** convert one quantifier to the other.
 - $\forall x P(x) \equiv \neg \exists x \neg P(x)$
 - $\exists x P(x) \equiv \neg \forall x \neg P(x)$
 - **RULES:** $\forall \equiv \neg \exists \neg$ and $\exists \equiv \neg \forall \neg$
- **RULES:** To move negation “in” across a quantifier,
Change the quantifier to “the other quantifier”
and negate the predicate on “the other side.”
 - $\neg \forall x P(x) \equiv \neg \neg \exists x \neg P(x) \equiv \exists x \neg P(x)$
 - $\neg \exists x P(x) \equiv \neg \neg \forall x \neg P(x) \equiv \forall x \neg P(x)$

Universal Quantification \forall

- $\forall x$ means “for all x it is true that...”
- Allows us to make statements about all objects that have certain properties
- Can now state general rules:

$\forall x \text{ King}(x) \Rightarrow \text{Person}(x)$ “All kings are persons.”

$\forall x \text{ Person}(x) \Rightarrow \text{HasHead}(x)$ “Every person has a head.”

$\forall i \text{ Integer}(i) \Rightarrow \text{Integer}(\text{plus}(i,1))$ “If i is an integer then $i+1$ is an integer.”

- **Note: $\forall x \text{ King}(x) \wedge \text{Person}(x)$ is not correct!**

This would imply that all objects x are Kings and are People (!)

$\forall x \text{ King}(x) \Rightarrow \text{Person}(x)$ is the correct way to say this

- **Note that \Rightarrow (or \Leftrightarrow) is the natural connective to use with \forall .**

Existential Quantification \exists

- $\exists x$ means “there exists an x such that....”
 - There is in the world at least one such object x
- Allows us to make statements about some object without naming it, or even knowing what that object is:
 - $\exists x \text{ King}(x)$ “Some object is a king.”
 - $\exists x \text{ Lives_in}(\text{John}, \text{Castle}(x))$ “John lives in somebody’s castle.”
 - $\exists i \text{ Integer}(i) \wedge \text{Greater}(i,0)$ “Some integer is greater than zero.”
- **Note: $\exists i \text{ Integer}(i) \Rightarrow \text{Greater}(i,0)$ is not correct!**

It is vacuously true if anything in the world were not an integer (!)

$\exists i \text{ Integer}(i) \wedge \text{Greater}(i,0)$ is the correct way to say this
- **Note that \wedge is the natural connective to use with \exists .**

Combining Quantifiers --- Order (Scope)

The order of “unlike” quantifiers is important.

Like nested variable scopes in a programming language.

Like nested ANDs and ORs in a logical sentence.

$\forall x \exists y \text{ Loves}(x,y)$

- For everyone (“all x”) there is someone (“exists y”) whom they love.
- There might be a different y for each x (y is inside the scope of x)

$\exists y \forall x \text{ Loves}(x,y)$

- There is someone (“exists y”) whom everyone loves (“all x”).
- Every x loves the same y (x is inside the scope of y)

Clearer with parentheses: $\exists y (\forall x \text{ Loves}(x,y))$

The order of “like” quantifiers does not matter.

Like nested ANDs and ANDs in a logical sentence

$$\forall x \forall y P(x, y) \equiv \forall y \forall x P(x, y)$$

$$\exists x \exists y P(x, y) \equiv \exists y \exists x P(x, y)$$

De Morgan's Law for Quantifiers

De Morgan's Rule

$$P \wedge Q \equiv \neg (\neg P \vee \neg Q)$$

$$P \vee Q \equiv \neg (\neg P \wedge \neg Q)$$

$$\neg (P \wedge Q) \equiv (\neg P \vee \neg Q)$$

$$\neg (P \vee Q) \equiv (\neg P \wedge \neg Q)$$

Generalized De Morgan's Rule

$$\forall x P(x) \equiv \neg \exists x \neg P(x)$$

$$\exists x P(x) \equiv \neg \forall x \neg P(x)$$

$$\neg \forall x P(x) \equiv \exists x \neg P(x)$$

$$\neg \exists x P(x) \equiv \forall x \neg P(x)$$

AND/OR Rule is simple: if you bring a negation inside a disjunction or a conjunction, always switch between them (\neg OR \rightarrow AND \neg ; \neg AND \rightarrow OR \neg).

QUANTIFIER Rule is similar: if you bring a negation inside a universal or existential, always switch between them ($\neg \exists \rightarrow \forall \neg$; $\neg \forall \rightarrow \exists \neg$).

Fun with sentences

Brothers are siblings

$$\forall x, y \text{ Brother}(x, y) \Rightarrow \text{Sibling}(x, y).$$

“Sibling” is symmetric

$$\forall x, y \text{ Sibling}(x, y) \Leftrightarrow \text{Sibling}(y, x).$$

One's mother is one's female parent

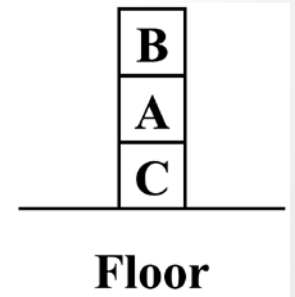
$$\forall x, y \text{ Mother}(x, y) \Leftrightarrow (\text{Female}(x) \wedge \text{Parent}(x, y)).$$

A first cousin is a child of a parent's sibling

$$\forall x, y \text{ FirstCousin}(x, y) \Leftrightarrow \exists p, ps \text{ Parent}(p, x) \wedge \text{Sibling}(ps, p) \wedge \text{Parent}(ps, y)$$

Semantics: Interpretation

- An interpretation of a sentence is an assignment that maps
 - Object constants to objects in the worlds,
 - n-ary function symbols to n-ary functions in the world,
 - n-ary relation symbols to n-ary relations in the world
- Given an interpretation, an atomic sentence has the value “true” if it denotes a relation that holds for those individuals denoted in the terms. Otherwise it has the value “false.”
 - Example: Block world:
 - A, B, C, Floor, On, Clear
 - World:
 - On(A,B) is false, Clear(B) is true, On(C,Floor) is true...
 - Under an interpretation that maps symbol A to block A, symbol B to block B, symbol C to block C, symbol Floor to the Floor
 - Some other interpretation might result in different truth values.



Semantics: Models and Definitions

- An interpretation and possible world satisfies a wff (sentence) if the wff has the value “true” under that interpretation in that possible world.
- **Model**: A domain and an interpretation that satisfies a wff is a model of that wff
- **Validity**: Any wff that has the value “true” in all possible worlds and under all interpretations is valid.
- Any wff that does not have a model under any interpretation is inconsistent or unsatisfiable.
- Any wff that is true in at least one possible world under at least one interpretation is satisfiable.
- If a wff w has a value true under all the models of a set of sentences KB then KB logically entails w .

Conversion to CNF

- Everyone who loves all animals is loved by someone:

$$\forall x [\forall y \text{ Animal}(y) \Rightarrow \text{Loves}(x,y)] \Rightarrow [\exists y \text{ Loves}(y,x)]$$

1. Eliminate biconditionals and implications

$$\forall x [\neg \forall y \neg \text{Animal}(y) \vee \text{Loves}(x,y)] \vee [\exists y \text{ Loves}(y,x)]$$

2. Move \neg inwards:

$$\neg \forall x p \equiv \exists x \neg p, \quad \neg \exists x p \equiv \forall x \neg p$$

$$\forall x [\exists y \neg(\neg \text{Animal}(y) \vee \text{Loves}(x,y))] \vee [\exists y \text{ Loves}(y,x)]$$

$$\forall x [\exists y \neg \neg \text{Animal}(y) \wedge \neg \text{Loves}(x,y)] \vee [\exists y \text{ Loves}(y,x)]$$

$$\forall x [\exists y \text{ Animal}(y) \wedge \neg \text{Loves}(x,y)] \vee [\exists y \text{ Loves}(y,x)]$$

Conversion to CNF contd.

3. Standardize variables: each quantifier should use a different one

$$\forall x [\exists y \textit{Animal}(y) \wedge \neg \textit{Loves}(x,y)] \vee [\exists z \textit{Loves}(z,x)]$$

4. Skolemize: a more general form of existential instantiation.
Each existential variable is replaced by a **Skolem function** of the enclosing universally quantified variables:

$$\forall x [\textit{Animal}(F(x)) \wedge \neg \textit{Loves}(x,F(x))] \vee \textit{Loves}(G(x),x)$$

5. Drop universal quantifiers:

$$[\textit{Animal}(F(x)) \wedge \neg \textit{Loves}(x,F(x))] \vee \textit{Loves}(G(x),x)$$

6. Distribute \vee over \wedge :

$$[\textit{Animal}(F(x)) \vee \textit{Loves}(G(x),x)] \wedge [\neg \textit{Loves}(x,F(x)) \vee \textit{Loves}(G(x),x)]$$

Unification

- Recall: $\text{Subst}(\theta, p)$ = result of substituting θ into sentence p
- Unify algorithm: takes 2 sentences p and q and returns a unifier if one exists

$$\text{Unify}(p,q) = \theta \quad \text{where } \text{Subst}(\theta, p) = \text{Subst}(\theta, q)$$

where θ is a list of variable/substitution pairs that will make p and q syntactically identical

- Example:

$p = \text{Knows}(\text{John}, x)$

$q = \text{Knows}(\text{John}, \text{Jane})$

$$\text{Unify}(p,q) = \{x/\text{Jane}\}$$

Unification examples

- simple example: query = $\text{Knows}(\text{John},x)$, i.e., who does John know?

p	q	θ
$\text{Knows}(\text{John},x)$	$\text{Knows}(\text{John},\text{Jane})$	$\{x/\text{Jane}\}$
$\text{Knows}(\text{John},x)$	$\text{Knows}(y,\text{OJ})$	$\{x/\text{OJ},y/\text{John}\}$
$\text{Knows}(\text{John},x)$	$\text{Knows}(y,\text{Mother}(y))$	$\{y/\text{John},x/\text{Mother}(\text{John})\}$
$\text{Knows}(\text{John},x)$	$\text{Knows}(x,\text{OJ})$	$\{\text{fail}\}$

- Last unification fails: only because x can't take values John and OJ at the same time
 - But we know that if John knows x , and everyone (x) knows OJ, we should be able to infer that John knows OJ
- Problem is due to use of same variable x in both sentences
- Simple solution: Standardizing apart eliminates overlap of variables, e.g., $\text{Knows}(z,\text{OJ})$

Unification examples

- 1) UNIFY(Knows(John, x), Knows(John, Jane)) { x / Jane }
- 2) UNIFY(Knows(John, x), Knows(y, Jane)) { x / Jane, y / John }
- 3) UNIFY(Knows(y, x), Knows(John, Jane)) { x / Jane, y / John }
- 4) UNIFY(Knows(John, x), Knows(y, Father (y))) { y / John, x / Father (John) }
- 5) UNIFY(Knows(John, F(x)), Knows(y, F(F(z)))) { y / John, x / F (z) }
- 6) UNIFY(Knows(John, F(x)), Knows(y, G(z))) None
- 7) UNIFY(Knows(John, F(x)), Knows(y, F(G(y)))) { y / John, x / G (John) }

Example knowledge base

- The law says that it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles, and all of its missiles were sold to it by Colonel West, who is American.
- Prove that Col. West is a criminal

Example knowledge base (Horn clauses)

... it is a crime for an American to sell weapons to hostile nations:

$American(x) \wedge Weapon(y) \wedge Sells(x,y,z) \wedge Hostile(z) \Rightarrow Criminal(x)$

Nono ... has some missiles, i.e., $\exists x Owns(Nono,x) \wedge Missile(x)$:

$Owns(Nono,M_1) \wedge Missile(M_1)$

... all of its missiles were sold to it by Colonel West

$Missile(x) \wedge Owns(Nono,x) \Rightarrow Sells(West,x,Nono)$

Missiles are weapons:

$Missile(x) \Rightarrow Weapon(x)$

An enemy of America counts as "hostile":

$Enemy(x,America) \Rightarrow Hostile(x)$

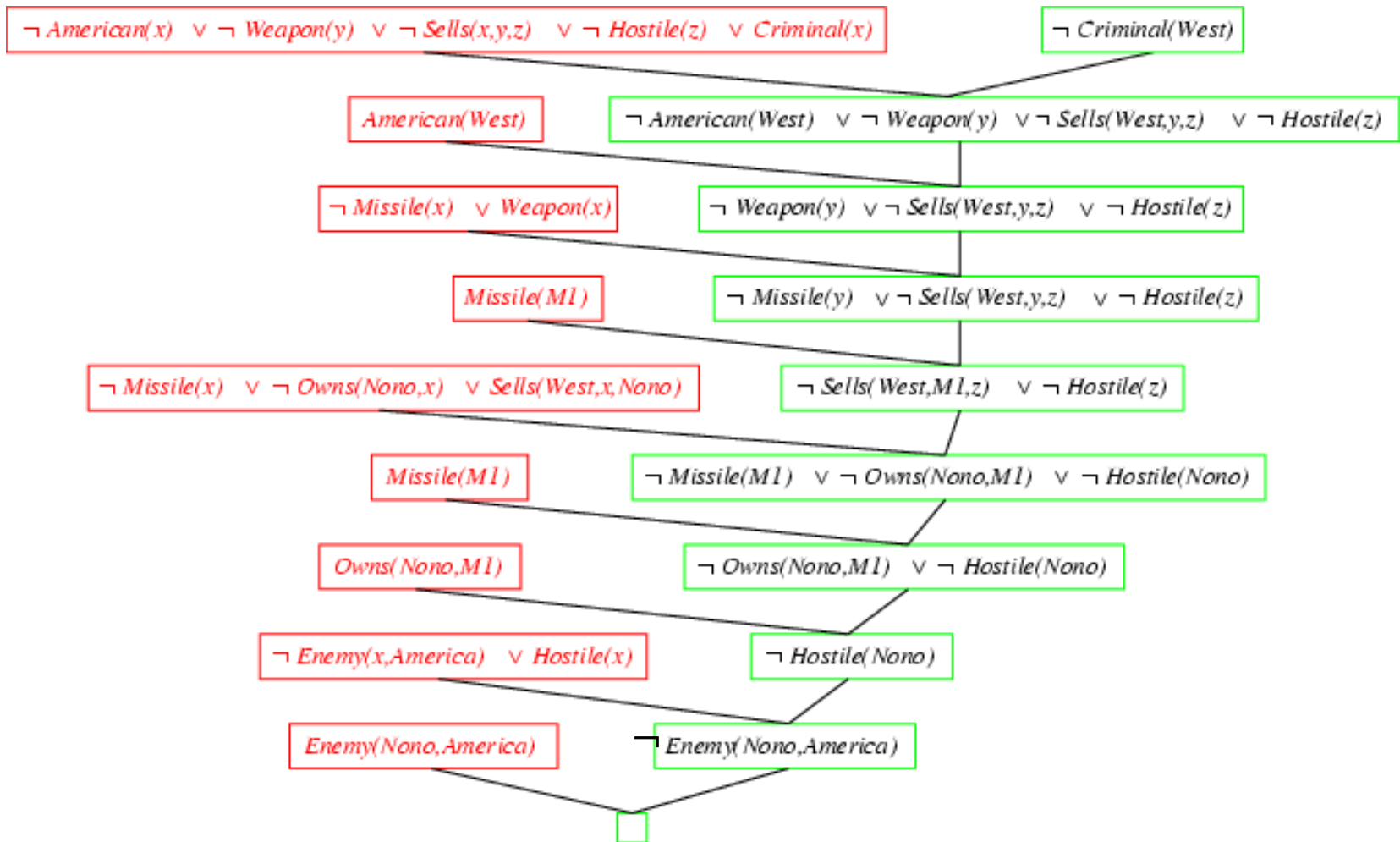
West, who is American ...

$American(West)$

The country Nono, an enemy of America ...

$Enemy(Nono,America)$

Resolution proof:



CS-171 Midterm Review

- **Agents**
 - (R&N Ch. 1-2, 26.preamble, 26.3-4, 27.4)
- **Propositional Logic**
 - (R&N Ch. 7.1-7.5)
- **First-Order Logic**
 - (R&N Ch. 8.1-8.5, 9.1-9.2)
- **Probability & Bayesian Networks**
 - (R&N Ch. 13, 14.1-14.5)
- **Hidden Markov Models**
 - (R&N Ch. 5.1-15.3)
- Questions on any topic
- Please review your quizzes & old test

Review Probability

Chapter 13

- Basic probability notation/definitions:
 - Probability model, unconditional/prior and conditional/posterior probabilities, factored representation (= variable/value pairs), random variable, (joint) probability distribution, probability density function (pdf), marginal probability, (conditional) independence, normalization, etc.
- Basic probability formulas:
 - Probability axioms, sum rule, product rule, Bayes' rule.
- How to use Bayes' rule:
 - Naïve Bayes model (naïve Bayes classifier)

Syntax

- Basic element: **random variable**
- Similar to propositional logic: possible worlds defined by assignment of values to random variables.
- **Boolean** random variables
e.g., *Cavity (= do I have a cavity?)*
- **Discrete** random variables
e.g., *Weather is one of <sunny,rainy,cloudy,snow>*
- Domain values must be exhaustive and mutually exclusive
- Elementary proposition is an assignment of a value to a random variable:
e.g., *Weather = sunny; Cavity = false (abbreviated as \neg cavity)*
- Complex propositions formed from elementary propositions and standard logical connectives :
e.g., *Weather = sunny \vee Cavity = false*

Probability

- $P(a)$ is the probability of proposition “a”
 - e.g., $P(\text{it will rain in London tomorrow})$
 - The proposition a is actually true or false in the real-world
- **Probability Axioms:**
 - $0 \leq P(a) \leq 1$
 - $P(\text{NOT}(a)) = 1 - P(a) \quad \Rightarrow \quad \sum_A P(A) = 1$
 - $P(\text{true}) = 1$
 - $P(\text{false}) = 0$
 - $P(A \text{ OR } B) = P(A) + P(B) - P(A \text{ AND } B)$
- Any agent that holds degrees of beliefs that contradict these axioms will act irrationally in some cases
- **Rational agents cannot violate probability theory.**
 - Acting otherwise results in irrational behavior.

Conditional Probability

- $P(a|b)$ is the conditional probability of proposition a , conditioned on knowing that b is true,
 - E.g., $P(\text{rain in London tomorrow} | \text{raining in London today})$
 - $P(a|b)$ is a “posterior” or conditional probability
 - The updated probability that a is true, now that we know b
 - $P(a|b) = P(a \wedge b) / P(b)$
 - Syntax: $P(a | b)$ is the probability of a given that b is true
 - a and b can be any propositional sentences
 - e.g., $p(\text{John wins OR Mary wins} | \text{Bob wins AND Jack loses})$
- $P(a|b)$ obeys the same rules as probabilities,
 - E.g., $P(a | b) + P(\text{NOT}(a) | b) = 1$
 - All probabilities in effect are conditional probabilities
 - E.g., $P(a) = P(a | \text{our background knowledge})$

Concepts of Probability

- **Unconditional Probability**

- **$P(a)$** , the probability of “a” being true, or **$P(a=True)$**
- Does not depend on anything else to be true (**unconditional**)
- Represents the probability prior to further information that may adjust it (**prior**)

- **Conditional Probability**


- **$P(a|b)$** , the probability of “a” being true, given that “b” is true
- Relies on “b” = true (**conditional**)
- Represents the prior probability adjusted based upon new information “b” (**posterior**)
- Can be generalized to more than 2 random variables:
 - e.g. $P(a|b, c, d)$

- **Joint Probability**

- **$P(a, b) = P(a \wedge b)$** , the probability of “a” and “b” both being true
- Can be generalized to more than 2 random variables:
 - e.g. $P(a, b, c, d)$

Basic Probability Relationships

- **$P(A) + P(\neg A) = 1$**
 - Implies that $P(\neg A) = 1 - P(A)$
- **$P(A, B) = P(A \wedge B) = P(A) + P(B) - P(A \vee B)$**
 - Implies that $P(A \vee B) = P(A) + P(B) - P(A \wedge B)$
- **$P(A | B) = P(A, B) / P(B)$**
 - Conditional probability; “Probability of A given B”
- **$P(A, B) = P(A | B) P(B)$**
 - Product Rule (Factoring); applies to any number of variables
 - $P(a, b, c, \dots, z) = P(a | b, c, \dots, z) P(b | c, \dots, z) P(c | \dots, z) \dots P(z)$
- **$P(A) = \sum_{B, C} P(A, B, C) = \sum_{b \in B, c \in C} P(A, b, c)$**
 - Sum Rule (Marginal Probabilities); for any number of variables
 - $P(A, D) = \sum_B \sum_C P(A, B, C, D) = \sum_{b \in B} \sum_{c \in C} P(A, b, c, D)$
- **$P(B | A) = P(A | B) P(B) / P(A)$**
 - Bayes’ Rule; for any number of variables



You need to know these !

Full Joint Distribution

- We can fully specify a probability space by constructing a **full joint distribution**:
 - A full joint distribution contains a probability for every possible combination of variable values.
 - E.g., $P(J=f, M=t, A=t, B=t, E=f)$
- From a full joint distribution, the product rule, sum rule, and Bayes' rule can create any desired joint and conditional probabilities.

Computing with Probabilities: Law of Total Probability

Law of Total Probability (aka “summing out” or marginalization)

$$\begin{aligned} P(a) &= \sum_b P(a, b) \\ &= \sum_b P(a | b) P(b) \quad \text{where } B \text{ is any random variable} \end{aligned}$$

Why is this useful?

Given a joint distribution (e.g., $P(a,b,c,d)$) we can obtain any “marginal” probability (e.g., $P(b)$) by summing out the other variables, e.g.,

$$P(b) = \sum_a \sum_c \sum_d P(a, b, c, d)$$

We can compute any conditional probability given a joint distribution, e.g.,

$$\begin{aligned} P(c | b) &= \sum_a \sum_d P(a, c, d | b) \\ &= \sum_a \sum_d P(a, c, d, b) / P(b) \\ &\quad \text{where } P(b) \text{ can be computed as above} \end{aligned}$$

Computing with Probabilities: The Chain Rule or Factoring

We can always write

$$P(a, b, c, \dots z) = P(a \mid b, c, \dots z) P(b, c, \dots z)$$

(by definition of joint probability)

Repeatedly applying this idea, we can write

$$P(a, b, c, \dots z) = P(a \mid b, c, \dots z) P(b \mid c, \dots z) P(c \mid \dots z) \dots P(z)$$

This factorization holds for any ordering of the variables

This is the chain rule for probabilities

Independence

- Formal Definition:

- 2 random variables A and B are **independent** iff:

$$P(\mathbf{a}, \mathbf{b}) = P(\mathbf{a}) P(\mathbf{b}), \quad \text{for all values } \mathbf{a}, \mathbf{b}$$

- Informal Definition:

- 2 random variables A and B are **independent** iff:

$$P(\mathbf{a} \mid \mathbf{b}) = P(\mathbf{a}) \quad \text{OR} \quad P(\mathbf{b} \mid \mathbf{a}) = P(\mathbf{b}), \quad \text{for all values } \mathbf{a}, \mathbf{b}$$

- $P(\mathbf{a} \mid \mathbf{b}) = P(\mathbf{a})$ tells us that knowing \mathbf{b} provides no change in our probability for \mathbf{a} , and thus \mathbf{b} contains no information about \mathbf{a} .

- Also known as **marginal independence**, as all other variables have been marginalized out.

- In practice true independence is very rare:

- “butterfly in China” effect
- Conditional independence is much more common and useful

Conditional Independence

- Formal Definition:

- 2 random variables A and B are **conditionally independent** given C iff:
 $P(a, b | c) = P(a | c) P(b | c)$, for all values a, b, c

- Informal Definition:

- 2 random variables A and B are **conditionally independent** given C iff:
 $P(a | b, c) = P(a | c)$ OR $P(b | a, c) = P(b | c)$, for all values a, b, c
- $P(a | b, c) = P(a | c)$ tells us that learning about b, given that we already know c, provides no change in our probability for a, and thus b contains no information about a beyond what c provides.

- Naïve Bayes Model:

- Often a single variable can directly influence a number of other variables, all of which are conditionally independent, given the single variable.
- E.g., k different symptom variables X_1, X_2, \dots, X_k , and $C = \text{disease}$, reducing to:
 $P(X_1, X_2, \dots, X_k | C) = P(C) \prod P(X_i | C)$

Examples of Conditional Independence

- **H=Heat, S=Smoke, F=Fire**

- $P(H, S | F) = P(H | F) P(S | F)$

- $P(S | F, H) = P(S | F)$

- If we know there is/is not a fire, observing heat tells us no more information about smoke

- **F=Fever, R=RedSpots, M=Measles**

- $P(F, R | M) = P(F | M) P(R | M)$

- $P(R | M, F) = P(R | M)$

- If we know we do/don't have measles, observing fever tells us no more information about red spots

- **C=SharpClaws, F=SharpFangs, S=Species**

- $P(C, F | S) = P(C | S) P(F | S)$

- $P(F | S, C) = P(F | S)$

- If we know the species, observing sharp claws tells us no more information about sharp fangs

Review Bayesian Networks

Chapter 14.1-5

- **Basic concepts and vocabulary of Bayesian networks.**
 - Nodes represent random variables.
 - Directed arcs represent (informally) direct influences.
 - Conditional probability tables, $P(X_i \mid \text{Parents}(X_i))$.
- **Given a Bayesian network:**
 - Write down the full joint distribution it represents.
- **Given a full joint distribution in factored form:**
 - Draw the Bayesian network that represents it.
- **Given a variable ordering and background assertions of conditional independence among the variables:**
 - Write down the factored form of the full joint distribution, as simplified by the conditional independence assertions.
- **Use the network to find answers to probability questions about it.**

Bayesian Networks

- Represent dependence/independence via a directed graph
 - Nodes = random variables
 - Edges = direct dependence
- Structure of the graph \Leftrightarrow Conditional independence

- Recall the chain rule of repeated conditioning:

$$P(X_1, X_2, X_3, \dots, X_N) = P(X_1 | X_2, X_3, \dots, X_N) P(X_2 | X_3, \dots, X_N) \cdots P(X_N)$$

$$P(X_1, X_2, X_3, \dots, X_N) = \prod_{i=1}^n P(X_i | \text{parents}(X_i))$$

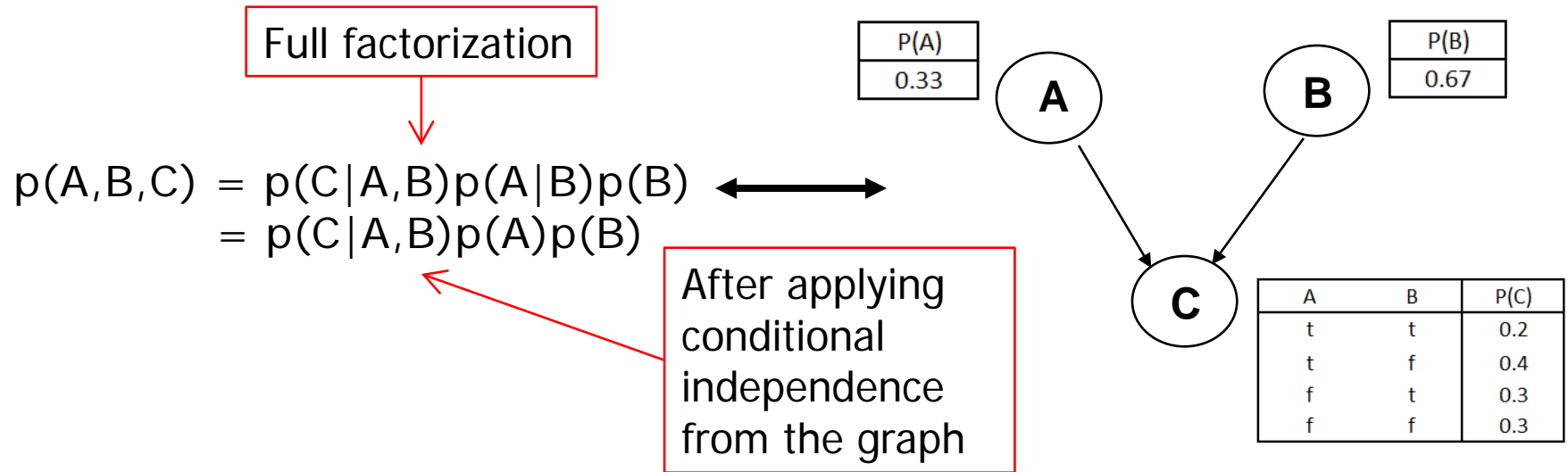
The full joint distribution

The graph-structured approximation

- Requires that graph is acyclic (no directed cycles)
- 2 components to a Bayesian network
 - The graph structure (conditional independence assumptions)
 - The numerical probabilities (of each variable given its parents)

Bayesian Network

- A Bayesian network specifies a joint distribution in a structured form:

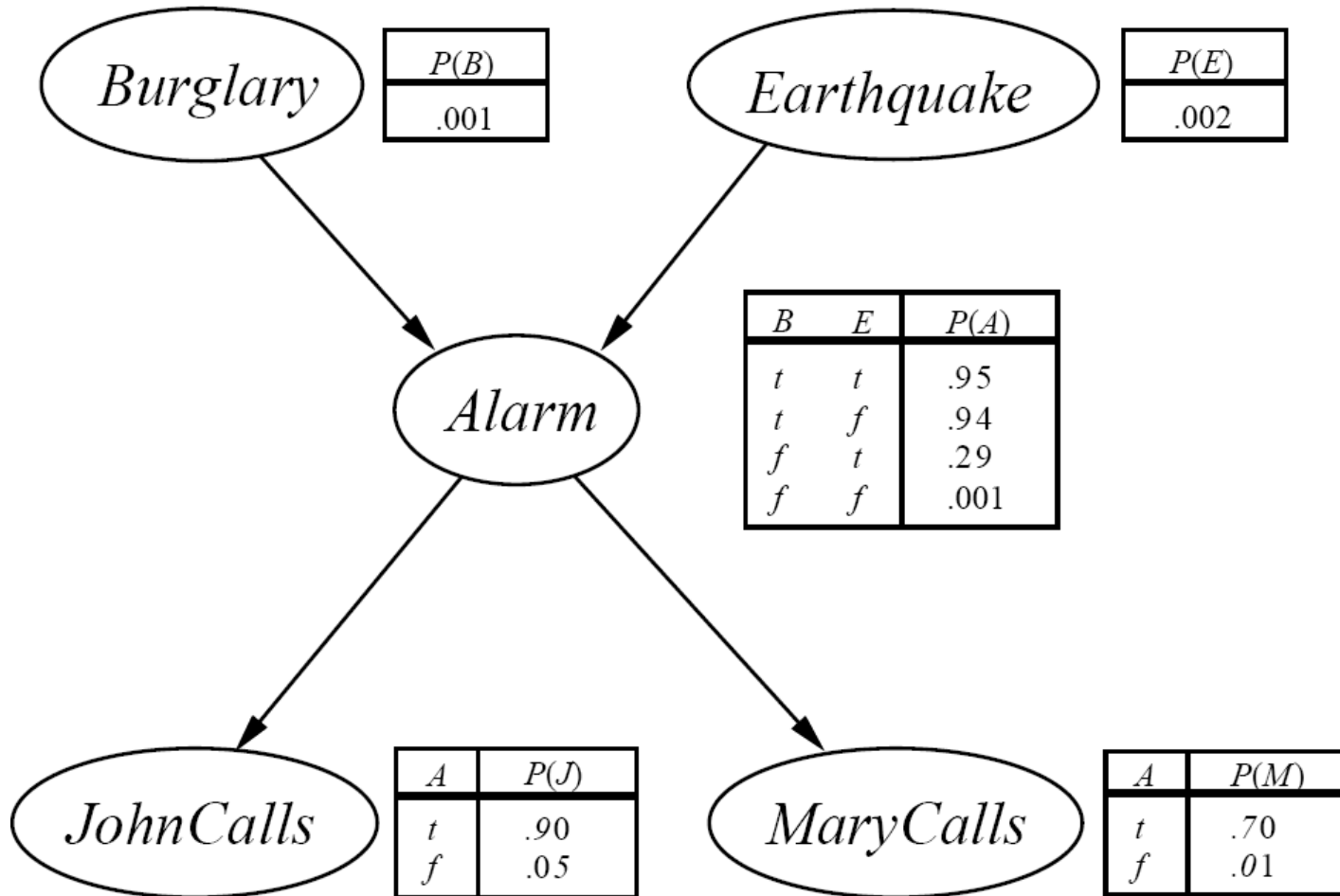


- Dependence/independence represented via a directed graph:
 - Node = random variable
 - Directed Edge = conditional dependence
 - Absence of Edge = conditional independence
- Allows concise view of joint distribution relationships:
 - Graph nodes and edges show conditional relationships between variables.
 - Tables provide probability data.

Burglar Alarm Example

- Consider the following 5 binary variables:
 - B = a burglary occurs at your house
 - E = an earthquake occurs at your house
 - A = the alarm goes off
 - J = John calls to report the alarm
 - M = Mary calls to report the alarm
- Sample Query: What is $P(B | M, J)$?
- Using full joint distribution to answer this question requires
 - $2^5 - 1 = 31$ parameters
- Can we use prior domain knowledge to come up with a Bayesian network that requires fewer probabilities?

The Resulting Bayesian Network



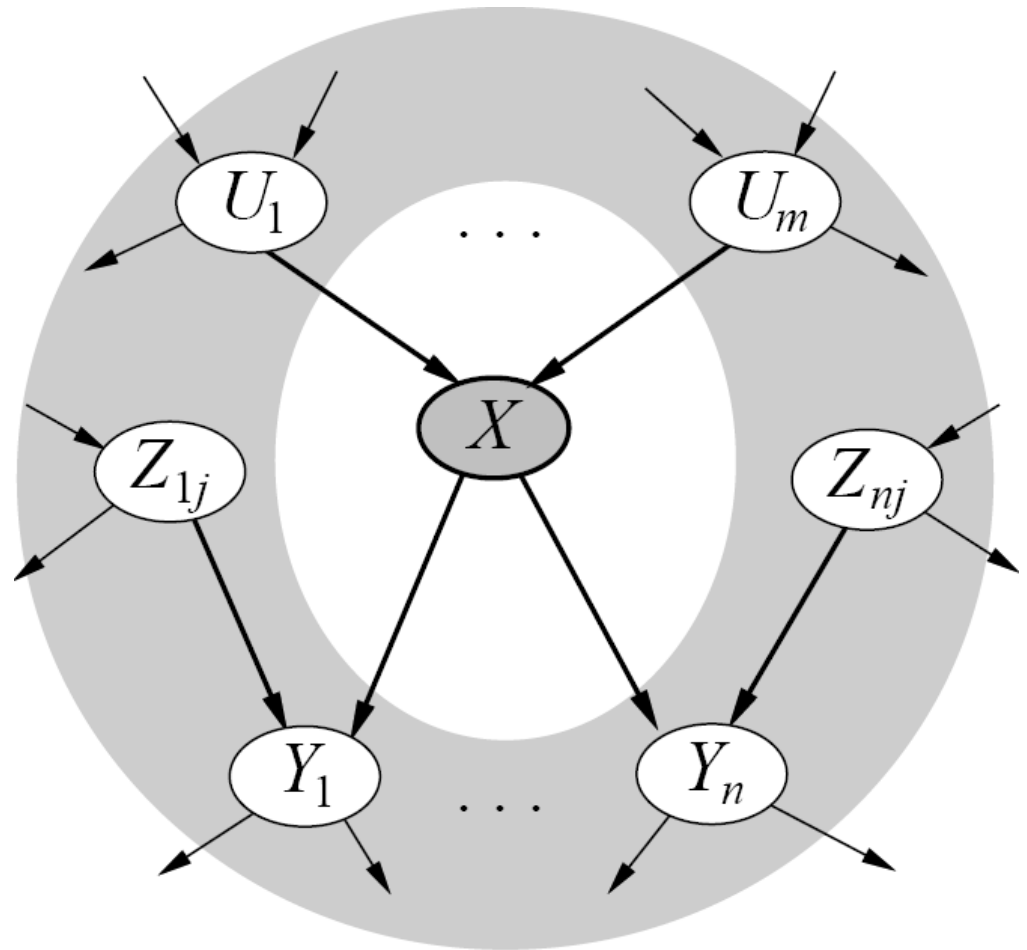
Given a graph, can we “read off” conditional independencies?

The “Markov Blanket” of X (the gray area in the figure)

X is conditionally independent of everything else, GIVEN the values of:

- * X 's parents
- * X 's children
- * X 's children's parents

X is conditionally independent of its non-descendants, GIVEN the values of its parents.



Summary

- Bayesian networks represent a joint distribution using a graph
- The graph encodes a set of conditional independence assumptions
- Answering queries (or inference or reasoning) in a Bayesian network amounts to computation of appropriate conditional probabilities
- Probabilistic inference is intractable in the general case
 - Can be done in linear time for certain classes of Bayesian networks (polytrees: at most one directed path between any two nodes)
 - Usually faster and easier than manipulating the full joint distribution

CS-171 Midterm Review

- **Agents**
 - (R&N Ch. 1-2, 26.preamble, 26.3-4, 27.4)
- **Propositional Logic**
 - (R&N Ch. 7.1-7.5)
- **First-Order Logic**
 - (R&N Ch. 8.1-8.5, 9.1-9.2)
- **Probability & Bayesian Networks**
 - (R&N Ch. 13, 14.1-14.5)
- **Hidden Markov Models**
 - (R&N Ch. 5.1-15.3)
- Questions on any topic
- Please review your quizzes & old test

CS-171 Midterm Review

- **Agents**
 - (R&N Ch. 1-2, 26.preamble, 26.3-4, 27.4)
- **Propositional Logic**
 - (R&N Ch. 7.1-7.5)
- **First-Order Logic**
 - (R&N Ch. 8.1-8.5, 9.1-9.2)
- **Probability & Bayesian Networks**
 - (R&N Ch. 13, 14.1-14.5)
- **Hidden Markov Models**
 - (R&N Ch. 5.1-15.3)
- Questions on any topic
- Please review your quizzes & old test