COMP3411/COMP9414/9814: Artificial Intelligence

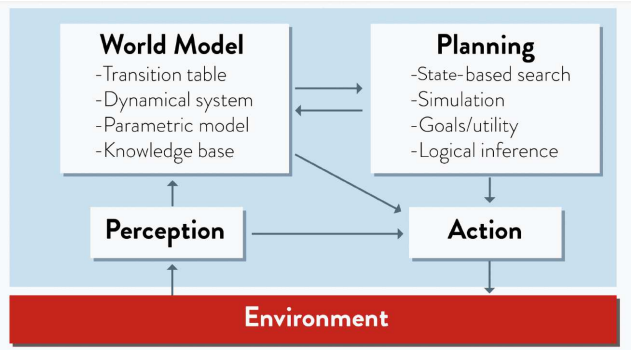
Week 8: Logical Agents

Russell & Norvig, Chapters 7 & 8.

Outline

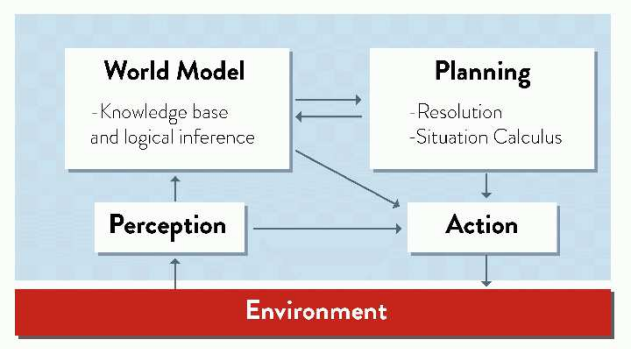
- Logic in general – models and entailment
- Propositional Logic
- Equivalence, Validity, Satisfiability
- Inference Rules and Theorem Proving
- Resolution and Conjunctive Normal Form
- Forward and Backward Chaining
- First Order Logic
- Universal and Existential Quantifiers
- Situation Calculus

Models and Planning



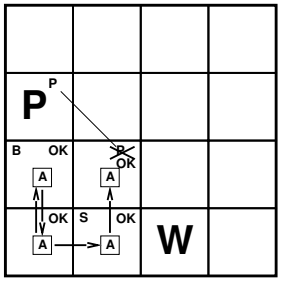
We previously used a transition table for our World Model, with Planning done by State-Based Search (BFS, DFS, UCS, IDS, Greedy, A*, etc.)

Models and Planning



Some environments instead require a Knowledge Base of facts and a set of Logical Inference Rules to reason about those facts.

Logical Reasoning for Wumpus World

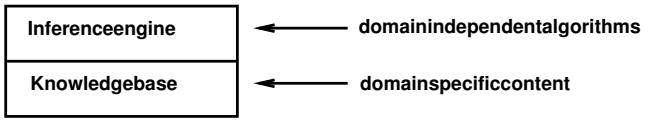


Need to represent:

- Facts: “Breeze in Square (1,2)”, “Stench in Square (2,1)”
- Inference Rules: “If there is a Breeze in Square (1,2) then there is a Wumpus in Square (1,1), (2,2) or (1,3)”.

Then try to deduce, for example, whether it is safe to move into Square (2,2).

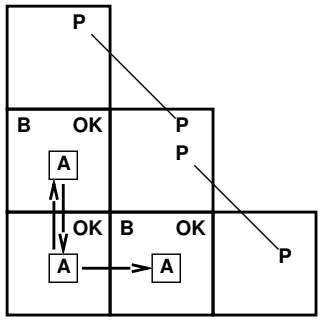
Knowledge bases



A Knowledge Base is a set of sentences in a formal language. It takes a Declarative approach to building an agent (or other system):

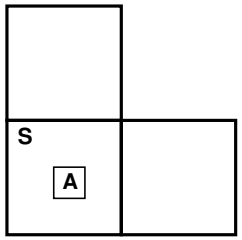
- Tell the system what it needs to know, then it can Ask itself what it needs to do
- Answers should follow from the KB.

Logical Reasoning for Wumpus World



- If there is a Breeze in (1,2) and (2,1), then there are no safe actions.
- Assuming that pits are uniformly distributed, a pit is more likely in (2,2) than in (3,1). How much more likely?

Reasoning about Future States



If there is a Smell in (1,1), there is no safe square to move into.
However, we can use logic to reason about future states.

- Shoot straight ahead
- Wumpus was there $\Rightarrow$ dead $\Rightarrow$ safe
- Wumpus wasn’t there $\Rightarrow$ safe

Knowledge Based Agent

The agent must be able to:

- represent states, actions, etc.
- incorporate new percepts
- update internal representations of the world
- deduce hidden properties of the world
- determine appropriate actions

Logic in general

Logics are formal languages for representing information such that conclusions can be drawn.

Syntax defines the sentences in the language.

Semantics define the "meaning" of sentences; i.e. define truth of a sentence in a world.

For example, the language of arithmetic:

$x+2 \geq y$ is a sentence; $x2+y >$ is not a sentence

$x+2 \geq y$ is true iff the number $x+2$ is no less than the number y

$x+2 \geq y$ is true in a world where $x=7,\ y=1$
$x+2 \geq y$ is false in a world where $x=0,\ y=6$

Entailment

Entailment means that one thing follows from another:

$$KB \models \alpha$$

Knowledge base KB entails sentence α if and only if
α is true in all worlds where KB is true.

e.g. the KB containing "the Moon is full" and "the tide is high"
entails "Either the Moon is full or the tide is high".

e.g. $x+y=4$ entails $4=x+y$

Entailment is a relationship between sentences (i.e. syntax)
that is based on semantics.

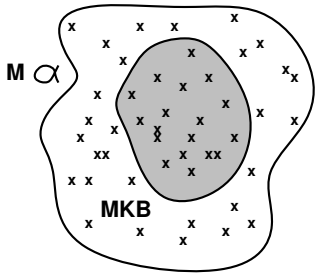
Models

Logicians typically think in terms of models, which are formally structured worlds with respect to 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

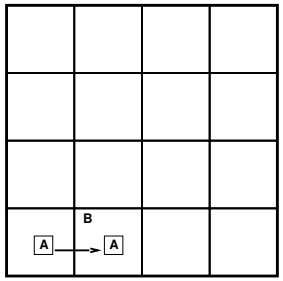
Then $KB \models \alpha$ if and only if $M(KB) \subseteq M(\alpha)$



Entailment in the wumpus world

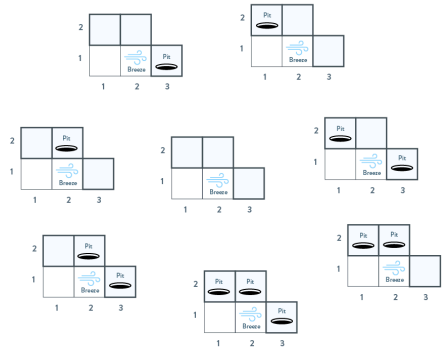
Situation after detecting nothing in [1,1], moving right, Breeze in [2,1]

Consider possible combinations for ?s assuming only pits.

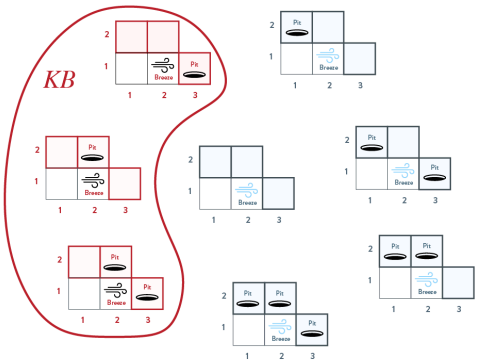


3 Boolean choices $\Rightarrow$ 8 possible combinations.

Wumpus models

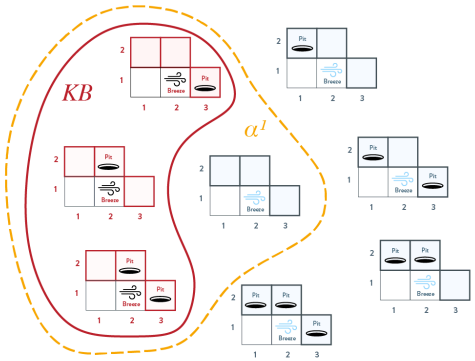


Wumpus models



KB = wumpus-world rules + observations

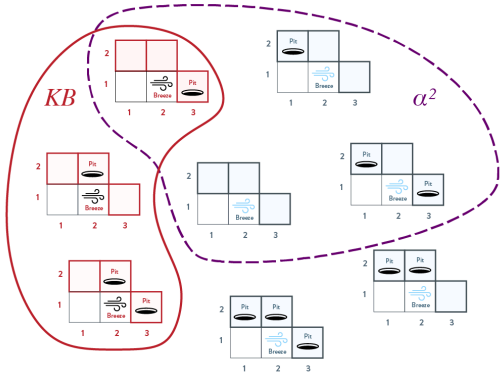
Wumpus models



KB = wumpus-world rules + observations

α_1 = "[1,2] is safe", $KB \models \alpha_1$, proved by model checking

Wumpus models



KB = wumpus-world rules + observations

α_2 = "[2,2] is safe", $KB \not\models \alpha_2$

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)

Propositional logic: Semantics

Each model specifies TRUE / FALSE for each proposition symbol. For example, if there are Pits in (1,2) and (2,2) but not (3,1) we would have the following assignments:

E.g.	$P_{1,2}$	$P_{2,2}$	$P_{3,1}$
	TRUE	TRUE	FALSE

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

Propositional logic: Semantics

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.

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

Truth Tables

P	Q	¬ P	P ∧ Q	P ∨ Q	P ⇒ Q
F	F	T	F	F	T
F	T	T	F	T	T
T	F	F	F	T	F
T	T	F	T	T	T

P "Fred is served alcohol"

Q "Fred is over 18 years old"

P ⇒ Q "If Fred is served alcohol, then he must be over 18"

Implication is not a causal relationship, but a rule that needs to be checked.

Wumpus World Sentences

Let $P_{i,j}$ be true if there is a pit in $[i, j]$.
Let $B_{i,j}$ be true if there is a breeze in $[i, j]$.

$$\neg P_{1,1}$$
$$\neg B_{1,1}$$
$$B_{2,1}$$

"Pits cause breezes in adjacent squares"

Wumpus World Sentences

Let $P_{i,j}$ be true if there is a pit in $[i, j]$.
Let $B_{i,j}$ be true if there is a breeze in $[i, j]$.

$$\neg P_{1,1}$$
$$\neg B_{1,1}$$
$$B_{2,1}$$

"Pits cause breezes in adjacent squares"

$$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$
$$B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

"A square is breezy if and only if there is an adjacent pit"

Logical Equivalence and Inference Rules

Inference Rules:

generalization:	$p \Rightarrow p \vee q$
specialization:	$p \wedge q \Rightarrow p$

Two sentences are logically equivalent iff true in same models:
$\alpha \equiv \beta$ if and only if $\alpha \models \beta$ and $\beta \models \alpha$

Logical Equivalence Rules

commutativity:	$p \wedge q \Leftrightarrow q \wedge p$	$p \vee q \Leftrightarrow q \vee p$
associativity:	$p \wedge (q \wedge r) \Leftrightarrow (p \wedge q) \wedge r$	$p \vee (q \vee r) \Leftrightarrow (p \vee q) \vee r$
distributivity:	$p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$	$p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$
implication:	$(p \Rightarrow q) \Leftrightarrow (\neg p \vee q)$	
idempotent:	$p \wedge p \Leftrightarrow p$	$p \vee p \Leftrightarrow p$
double negation:	$\neg\neg p \Leftrightarrow p$	
contradiction:	$p \wedge \neg p \Leftrightarrow \text{FALSE}$	
excluded middle:		$p \vee \neg p \Leftrightarrow \text{TRUE}$
de Morgan:	$\neg(p \wedge q) \Leftrightarrow (\neg p \vee \neg q)$	$\neg(p \vee q) \Leftrightarrow (\neg p \wedge \neg q)$

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. $(A \vee B) \wedge C$

A sentence is unsatisfiable if it is true in no models
e.g. $A \wedge \neg A$

Satisfiability is connected to inference via the following:
$KB \models \alpha$ if and only if $(KB \wedge \neg\alpha)$ is unsatisfiable
i.e. prove α by *reductio ad absurdum*

Example

Prove that:

$(A \wedge (B \Rightarrow C)) \Leftrightarrow (\neg(A \Rightarrow B) \vee (A \wedge C))$

$$
\begin{aligned}
(A \wedge (B \Rightarrow C)) &\Leftrightarrow A \wedge (\neg B \vee C) && \text{[implication]} \\
&\Leftrightarrow (A \wedge \neg B) \vee (A \wedge C) && \text{[distributivity]} \\
&\Leftrightarrow \neg(\neg A \vee B) \vee (A \wedge C) && \text{[de Morgan]} \\
&\Leftrightarrow \neg(A \Rightarrow B) \vee (A \wedge C) && \text{[implication]}
\end{aligned}
$$

Inference

$KB \vdash_i \alpha$ = sentence α can be derived from KB by procedure i

Soundness: i is sound if
whenever $KB \vdash_i \alpha$, it is also true that $KB \models \alpha$

Completeness: i is complete if
whenever $KB \models \alpha$, it is also true that $KB \vdash_i \alpha$

Consequences of KB are a haystack; α is a needle.
Entailment = needle in haystack; inference = finding it.

Conjunctive Normal Form

In order to apply Resolution, we must first convert the KB into Conjunctive Normal Form (CNF).

This means that the KB is a conjunction of clauses, and each clause is a disjunction of (possibly negated) literals.

e.g. $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$

Conversion to CNF

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

1. Eliminate $\Leftrightarrow$, 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 $\Rightarrow$, replacing $\alpha \Rightarrow \beta$ with $\neg\alpha \vee \beta$.

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

3. Move $\neg$ inwards using de Morgan's rules and double-negation:

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

4. Apply distributivity law ($\vee$ over $\wedge$) and flatten:

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

Resolution

Suppose we have two disjunctive clauses $\ell_1 \wedge \ldots \wedge \ell_i \wedge \ldots \wedge \ell_k$ and $m_1 \wedge \ldots \wedge m_j \wedge \ldots \wedge m_n$

We can then derive a new clause by eliminating l_i, m_j and combining all the other literals, i.e.

$$\frac{\ell_1 \vee \ldots \vee \ell_i \vee \ldots \vee \ell_k, \qquad m_1 \vee \ldots \vee m_j \vee \ldots \vee m_n}{\ell_1 \vee \ldots \vee \ell_{i-1} \vee \ell_{i+1} \vee \ldots \vee \ell_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n}$$

where ℓ_i and m_j are complementary literals. e.g.

$$\frac{P_{1,3} \vee P_{2,2}, \qquad \neg P_{2,2}}{P_{1,3}}$$

Resolution is sound and complete for propositional logic.

Wumpus World Example

Use resolution to prove that if there is a Breeze in (1,1), there must also be a Breeze in (2,2), i.e. prove $(\neg B_{1,1} \vee B_{2,2})$, from this KB:

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

Answer:

$$\frac{\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}, \quad \neg P_{1,2} \vee B_{2,2}}{\neg B_{1,1} \vee P_{2,1} \vee B_{2,2}}$$

$$\frac{\neg B_{1,1} \vee P_{2,1} \vee B_{2,2}, \quad \neg P_{2,1} \vee B_{2,2}}{\neg B_{1,1} \vee B_{2,2} \vee B_{2,2}}$$

The last clause is equivalent to $\neg B_{1,1} \vee B_{2,2}$

Proof Methods

Resolution provides us an alternative proof method which is generally somewhat faster than Truth Table Enumeration:

1. convert the into Conjunctive Normal Form,
2. add the negative of the clause you are trying to prove,
3. continually apply a series of resolutions until either
   (a) you derive the empty clause, or
   (b) no more pairs of clauses to which resolution can be applied

Horn Clauses

Model Checking can be done more efficiently if the clauses in the all happen to be in a special form for example they may all be Horn Clauses. Each Horn Clause is an implication involving only positive literals, in the form:

(conjunction of symbols) $\Rightarrow$ symbol

e.g. $C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$

Deduction with Horn Clauses can be done by Modus Ponens:

$$\frac{p_1, \ldots, p_n \,, \qquad p_1 \wedge \ldots \wedge p_n \Rightarrow q}{q}$$

Efficient Proof Methods, using Horn Clauses, can generally be divided into Forward Chaining and Backward Chaining.

Forward chaining

Look for a rule $p_1 \wedge \ldots \wedge p_n \Rightarrow q$ such that all the clauses on the left hand side are already in the KB.

Apply this rule, and add q to the KB.

Repeat this process until the goal clause α has been derived
(or we run out of rules to apply).

Backward chaining

Backward Chaining instead maintains a list of subgoals that it is trying to prove. Initially, this list consists of the ultimate goal α.

Choose a clause q from the list of subgoals.

- check if q is known already
- otherwise, find a rule with q on the right side and add clauses from the left side of this rule as new subgoals
- check to make sure each new subgoal is not on the list already, and has not already been proved, or failed

Forward vs. Backward Chaining

Forward Chaining is data-driven automatic, unconscious processing e.g. object recognition, routine decisions

- May do lots of work that is irrelevant to the goal

Backward Chaining is goal-driven, appropriate for problem-solving

- e.g. Where are my keys? How do I get into a PhD program?

Satisfiability as Constraint Satisfaction

Suppose you are given a KB written in 3-CNF. (This means Conjunctive Normal Form, with at most three literals in each clause.)

Does there exist any assignment of truth values to the symbols which will make all of the clauses in the KB TRUE?

For example, is there an assignment of truth values to A, B, C, D, E which will make the following TRUE?

$$(\neg D \vee \neg B \vee C) \wedge (B \vee \neg A \vee \neg C) \wedge (\neg C \vee \neg B \vee E) \wedge (E \vee \neg D \vee B) \wedge (B \vee E \vee \neg C)$$

This provides a classic example of a Constraint Satisfaction Problem, to which methods such as Hill Climbing or Simulated Annealing can be applied.

Satisfiability as Constraint Satisfaction

Difficulty of finding a solution depends on the ratio (m/n) where m is the number of clauses and n is the number of distinct symbols.

Suppose $n = 50$ and the KB is in 3-CNF.

- $m/n < 4.3 \Rightarrow$ under-constrained
- $m/n \simeq 4.3 \Rightarrow$ critically difficult
- $m/n > 4.3 \Rightarrow$ over-constrained

Other CSPs like n-Queens are sometimes converted to 3-CNF, as a way of measuring whether they are under- or over-constrained.

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

Limitations of Propositional Logic

"A square is breezy if and only if there is an adjacent pit."

This statement must be converted into a separate sentence for each square:

$$\begin{aligned} B_{1,1} &\Leftrightarrow (P_{1,2} \vee P_{2,1}) \\ B_{2,1} &\Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1}) \\ &\vdots \end{aligned}$$

What we really want is a way to express such a statement in one sentence for all squares, e.g.

$$\text{Breezy}(i,j) \Leftrightarrow (\text{Pit}(i-1,j) \vee \text{Pit}(i+1,j) \vee \text{Pit}(i,j-1) \vee \text{Pit}(i,j+1))$$

First-Order Logic will allow us to do this.

Syntax of First Order Logic

- Objects: people, houses, numbers, theories, colors, football games, wars, centuries ...
- Predicates: red, round, bogus, prime, multistoried, ...
  brother of, bigger than, inside, part of, has color, occurred after, owns, comes between, ...
- Functions: father of, best friend, third inning of, one more than, ...

Logics in General

Language	Ontology	Epistemology
Propositional logic	facts	true / false / unknown
First-order logic	facts, objects, relations	true / false / unknown
Temporal logic	facts, objects, relations, times	true / false / unknown
Probability theory	facts	degree of belief
Fuzzy logic	facts + degree of truth	known interval value

Syntax of First Order Logic

Constants	$Gold, Wumpus, [1,2], [3,1]$, etc.
Predicates	$Adjacent(), Smell(), Breeze(), At()$
Functions	$Result()$
Variables	$x,\ y,\ a,\ t, \ldots$
Connectives	$\wedge\ \vee\ \neg\ \Rightarrow\ \Leftrightarrow$
Equality	$=$
Quantifiers	$\forall\ \exists$

Sentences

Atomic sentence = $predicate(term_1, \ldots, term_n)$
or $term_1 = term_2$

Term = $function(term_1, \ldots, term_n)$
or *constant* or *variable*

e.g. $At(Agent, [1,1], S_0)$
$Holding(Gold, S_5)$

Complex sentences are made from atomic sentences using connectives

$$\neg S, \quad S_1 \wedge S_2, \quad S_1 \vee S_2, \quad S_1 \Rightarrow S_2, \quad S_1 \Leftrightarrow S_2$$

e.g. $Pit(x) \wedge Adjacent(x, y)$

Universal Quantification

$\forall \langle \text{variables} \rangle \langle \text{sentence} \rangle$

Where there's glitter, there's gold:
$\forall x \; Glitter(x) \Rightarrow At(Gold, x)$

$\forall x \; P$ is equivalent to the conjunction of instantiations of P

$$\begin{aligned} & Glitter([1,1]) \Rightarrow At(Gold,[1,1]) \\ \wedge \quad & Glitter([1,2]) \Rightarrow At(Gold,[1,2]) \\ \wedge \quad & Glitter([1,3]) \Rightarrow At(Gold,[1,3]) \\ \wedge \quad & \ldots \end{aligned}$$

Universal Quantification

Typically, $\Rightarrow$ is the main connective with $\forall$

Common mistake: using $\wedge$ as the main connective with $\forall$

$$\forall x \; Glitter(x) \wedge At(Gold, x)$$

means "There is Glitter everywhere and Gold everywhere."

Existential Quantification

$\exists \langle \text{variables} \rangle \langle \text{sentence} \rangle$

Some sheep are black
$\exists x \; Sheep(x) \wedge Black(x)$

$\exists x \; P$ is equivalent to the disjunction of instantiations of P

$$\begin{aligned} & Sheep(Dolly) \wedge Black(Dolly) \\ \vee \quad & Sheep(Lassie) \wedge Black(Lassie) \\ \vee \quad & Sheep(Skippy) \wedge Black(Skippy) \\ \vee \quad & \ldots \end{aligned}$$

Existential Quantification

Typically, $\wedge$ is the main connective with $\exists$

Common mistake: using $\Rightarrow$ as the main connective with $\exists$

$$\exists x\ Sheep(x) \Rightarrow Black(x)$$

is true if there is anyone who is not at sheep!

Properties of Quantifiers

$\forall x \forall y$ is the same as $\forall y \forall x$ (Why?)

$\exists x \exists y$ is the same as $\exists y \exists x$ (Why?)

$\exists x \forall y$ is not the same as $\forall y \exists x$

$\exists x \forall y\ Loves(x,y)$
"There is a person who loves everyone in the world"

$\forall y \exists x\ Loves(x,y)$
"Everyone in the world is loved by at least one person"

Quantifier duality: each can be expressed using the other

$\forall x\ Likes(x, IceCream)$ $\quad\quad$ $\neg\exists x\ \neg Likes(x, IceCream)$

$\exists x\ Likes(x, Broccoli)$ $\quad\quad$ $\neg\forall x\ \neg Likes(x, Broccoli)$

Fun with Sentences

Brothers are siblings

"Sibling" is symmetric

One's mother is one's female parent

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

Fun with Sentences

Brothers are siblings

$\forall x, y\ Brother(x,y) \Rightarrow Sibling(x,y)$

"Sibling" is symmetric

$\forall x, y\ Sibling(x,y) \Leftrightarrow Sibling(y,x)$

One's mother is one's female parent

$\forall x, y\ Mother(x,y) \Leftrightarrow (Female(x) \wedge Parent(x,y))$

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

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

Deducing Hidden Properties

Properties of locations:
$\forall x,t\ At(Agent,x,t) \wedge Smell(t) \Rightarrow Smelly(x)$
$\forall x,t\ At(Agent,x,t) \wedge Breeze(t) \Rightarrow Breezy(x)$
$\forall x,t\ At(Agent,x,t) \wedge Glitter(t) \Rightarrow AtGold(x)$

Squares are breezy near a pit:

Causal rule – infer effect from cause
$\forall x,y\ Pit(x) \wedge Adjacent(x,y) \Rightarrow Breezy(y)$

Diagnostic rule – infer cause from effect
$\forall y\ Breezy(y) \Rightarrow \exists x\ Pit(x) \wedge Adjacent(x,y)$

Definition for the *Breezy* predicate (combines Causal and Diagnostic):
$\forall y\ Breezy(y) \Leftrightarrow [\exists x\ Pit(x) \wedge Adjacent(x,y)]$

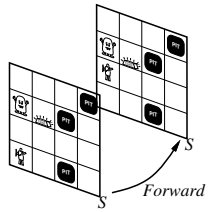
Keeping Track of Change

Facts hold only in certain situations, not universally.
e.g. $Holding(Gold,Now)$ rather than just $Holding(Gold)$

Situation calculus is one way to represent change:

- Adds a situation argument to each non-eternal predicate
- e.g. *Now* denotes a situation in $Holding(Gold,Now)$

Situations are connected by the *Result* function
$Result(a,s)$ is the situation that results from doing action a is state s



Describing Actions

We can plan a series of actions in a logical domain in a manner analogous to the Path Search algorithms discussed in Weeks 3 & 4. But, instead of the successor state being explicitly specified, we instead need to deduce what will be true and false in the state resulting from the previous state and action:

Effect axiom describe changes due to action

$\forall s\ AtGold(s) \Rightarrow Holding(Gold,Result(Grab,s))$

Describing Actions

Frame problem: Some facts will change as a result of an action, but many more will stay as they were.

$\forall s\ HaveArrow(s) \Rightarrow HaveArrow(Result(Grab,s))$

However, adding too many of these frame axioms can make the process unmanageable.

For example, if a cup is red, and you turn it upside down, it is still red. But, if a cup is full of water, and you turn it upside down, it is no longer full of water.

Large-scale expert systems of the 1980's often failed because of their inability to encode this kind of "commonsense" reasoning in explicit rules.

Describing Actions

Qualification problem: Normally, we expect actions to have a certain effect. But, in the real world there could be endless caveats. What happens if the gold is slippery, or nailed down, or too heavy, or you can't reach it, etc.

Ramification problem: Real actions have many secondary consequences – what about the dust on the gold, wear and tear on gloves, shoes, etc..

In general, we assume that a fact is true if a rule tells us that an action made it true, or if it was true before and no action made it false.

Searching for a Situation

Initial condition in KB (knowledge base):
$At(Agent,[1,1],S_0)$
$At(Gold,[1,2],S_0)$

Query: $Ask(KB,\exists s\ Holding(Gold,s))$
i.e., in what situation will I be holding the gold?

Answer: $s = Result(Grab,Result(Forward,S_0))$
i.e., go forward and then grab the gold

This assumes that the agent is interested in plans starting at S_0 and that S_0 is the only situation described in the KB.

Searching for a Plan of Actions

Represent plans as action sequences $[a_1,a_2,\ldots,a_n]$

$PlanResult(p,s)$ is the result of executing p in s

Then the query $Ask(KB,\exists p\ Holding(Gold,PlanResult(p,S_0)))$
has the solution $p = [Forward,Grab]$

Definition of $PlanResult$ in terms of $Result$:
$\forall s\ PlanResult([],s) = s$
$\forall a,p,s\ PlanResult([a|p],s) = PlanResult(p,Result(a,s))$

Planning systems are special-purpose reasoners designed to do this type of inference more efficiently than a general-purpose reasoner.

Summary

- First Order Logic:
  - objects and relations are semantic primitives
  - syntax: constants, functions, predicates, equality, quantifiers
- Increased expressive power: sufficient to define Wumpus World
- Situation calculus:
  - conventions for describing actions and change
  - can formulate planning as inference on a knowledge base