CMSC 671
Fall 2010

Thu 09/30/10

FOL (First Order Logic)
Chapter 8

Prof. Laura Zavala, laura.zavala@umbc.edu, ITE 373, 410-455-8775

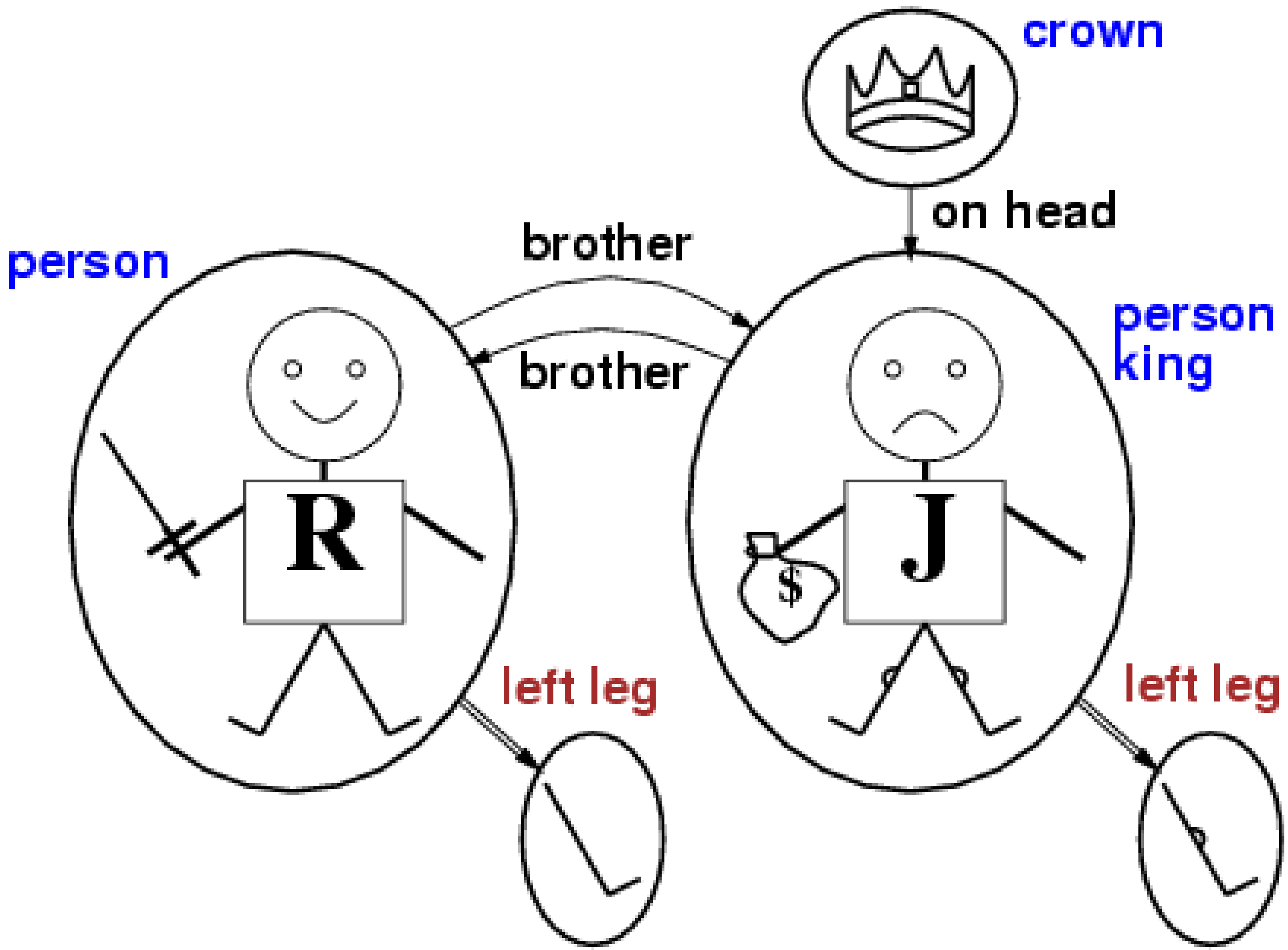
Propositional logic is a weak language

- Propositional logic lacks expressive power
- Hard to identify "individuals" (e.g., Mary, 3)
- Can't directly talk about properties of individuals or relations between individuals (e.g., "Bill is tall")
- Generalizations, patterns, regularities can't easily be represented (e.g., "all triangles have 3 sides")

First-Order Logic: a more expressive logic

- First-Order Logic (abbreviated FOL or FOPC) is expressive enough to concisely represent this kind of information
- FOL adds relations, variables, and quantifiers, e.g.,
  - *"Every elephant is gray":* $\forall$ x (elephant(x) $\rightarrow$ gray(x))
  - *"There is a white alligator":* $\exists$ x (alligator(X) ^ white(X))

FOL: Example



crown
on head
brother
brother
person
person
king
R
J
$
left leg
left leg

Why a more expressive logic?

- Example:
  - John loves all girls
  - Janet is a girl
  - *Therefore*, John loves Janet
- Propositional Logic:
  - {j_loves_all_girls, janet_is_girl} $\not\Rightarrow$ {j_loves_janet}
  - But: argument above still valid
- → We have to be able to talk about **objects/individuals**

First-order logic

- First-order logic (FOL) models the world in terms of
  - **Objects,** which are things with individual identities
  - **Properties** of objects that distinguish them from other objects
  - **Relations** that hold among sets of objects
  - **Functions,** which are a subset of relations where there is only one "value" for any given "input"
- Examples:
  - Objects: Students, lectures, companies, cars ...
  - Relations: Brother-of, bigger-than, outside, part-of, has-color, occurs-after, owns, visits, precedes, ...
  - Properties: blue, oval, even, large, ...
  - Functions: father-of, best-friend, second-half, one-more-than ...

First-order logic (2)

- Uses **Terms** for referring to **Objects**
  - a constant symbol, a variable symbol, or an n-place function of n terms
  - *e.g. John, x, LeftLeg(John)*
- Uses **Predicate Symbols** for referring to **Relations**
  - e.g. *Brother*
- **Atomic Sentences** state facts. e.g. *Brother(John, Sam)*
- **Complex Sentences** are formed from atomic sentences connected by the logical connectives: $\neg P$, $P \vee Q$, $P \wedge Q$, $P \rightarrow Q$, $P \leftrightarrow Q$
- **Quantified sentences** add quantifiers $\forall$ and $\exists$

First-order logic (3)

- A **well-formed formula (wff)** is a sentence containing no “free” variables. That is, all variables are “bound” by universal or existential quantifiers.
  - $(\forall x)P(x,y)$ has x bound as a universally quantified variable, but y is free.

A BNF for FOL

- S := <Sentence> ;
- <Sentence> := <AtomicSentence> |
- <Sentence> <Connective> <Sentence> |
- <Quantifier> <Variable>,... <Sentence> |
- "NOT" <Sentence> |
- "(" <Sentence> ")";
- <AtomicSentence> := <Predicate> "(" <Term>, ... ")" |
- <Term> "=" <Term>;
- <Term> := <Function> "(" <Term>, ... ")" |
- <Constant> |
- <Variable>;
- <Connective> := "AND" | "OR" | "IMPLIES" | "EQUIVALENT";
- <Quantifier> := "EXISTS" | "FORALL" ;
- <Constant> := "A" | "X1" | "John" | ... ;
- <Variable> := "a" | "x" | "s" | ... ;
- <Predicate> := "Before" | "HasColor" | "Raining" | ... ;
- <Function> := "Mother" | "LeftLegOf" | ... ;

Ontology and epistemology

- **Ontological commitment** – what the language assumes about the nature of reality
- **Epistemological commitment** – the possible states of knowledge

Language	Ontological Commitment (What exists in the world)	Epistemological Commitment (What an agent believes about facts)
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 0…1
Fuzzy logic	degree of truth	degree of belief 0…1

Quantifiers

- Universal **quantification**
  - $(\forall x)P(x)$ means that P holds for **all** values of x in the domain associated with that variable
  - E.g., $(\forall x)$ dolphin(x) $\rightarrow$ mammal(x)
- Existential **quantification**
  - $(\exists x)P(x)$ means that P holds for **some** value of x in the domain associated with that variable
  - E.g., $(\exists x)$ mammal(x) $\wedge$ lays-eggs(x)
  - Permits one to make a statement about some object without naming it

Quantifiers

- Universal quantifiers are often used with "implies" to form "rules":
  - $(\forall x)$ student(x) $\rightarrow$ smart(x) means "All students are smart"
- Universal quantification is *rarely* used to make blanket statements about every individual in the world:
  - $(\forall x)$student(x)$\wedge$smart(x) means "Everyone in the world is a student and is smart"
- Existential quantifiers are usually used with "and" to specify a list of properties about an individual:
  - $(\exists x)$ student(x) $\wedge$ smart(x) means "There is a student who is smart"
- A common mistake is to represent this English sentence as the FOL sentence:
  - $(\exists x)$ student(x) $\rightarrow$ smart(x)
  - But what happens when there is a person who is *not* a student?

likes(x,y) - x likes y
student(x) - x is a student
grade(x,y) - x receives grade y
person(x) - x is a person
fool(x,y,t) - x fools y at time t

Pretest

- **Everybody likes Raymond**
  - $\forall x$ likes(x,Raymond)
- **At least one student will get an A**
  - $\exists x$ student(x) $\wedge$ grade (x, A)
- **At least two students will get a B**
  - $\exists x\ \exists y$ student(x) $\wedge$ student(y) $\wedge$ grade (x, A) $\wedge$ grade (y, A) ^ $\neg$(x=y)
- **You can fool some of the people all of the time**
  - $\exists x$ person(x) $\wedge$ ($\forall t\ \forall y$ person(y) $\rightarrow$ fool(y,x,t))
  - $\exists x$ person(x) $\wedge$ $\forall t$ fool(You,x,t))
- **You can fool all of the people some of the time**
  - $\forall x\ \exists t$ person(x) $\rightarrow$ fool(You,x,t)
  - $\forall x$ person(x) $\rightarrow$ $\exists t$ fool(You,x,t)

Translating English to FOL

- **Every gardener likes the sun.**
  - $\forall x\ gardener(x) \rightarrow likes(x,Sun)$
- **You can fool some of the people all of the time.**
  - $\exists x\ \forall t\ \ person(x) \wedge time(t) \rightarrow can\text{-}fool(x,t)$
- **You can fool all of the people some of the time.**
  - $\forall x\ \exists t\ (person(x) \rightarrow time(t) \wedge can\text{-}fool(x,t))$
  - $\forall x\ (person(x) \rightarrow \exists t\ (time(t) \wedge can\text{-}fool(x,t))$

  Equivalent
- **All purple mushrooms are poisonous.**
  - $\forall x\ (mushroom(x) \wedge purple(x)) \rightarrow poisonous(x)$
- **No purple mushroom is poisonous.**
  - $\neg\exists x\ purple(x) \wedge mushroom(x) \wedge poisonous(x)$
  - $\forall x\ \ (mushroom(x) \wedge purple(x)) \rightarrow \neg poisonous(x)$

  Equivalent
- **There are exactly two purple mushrooms.**
  - $\exists x\ \exists y\ mushroom(x) \wedge purple(x) \wedge mushroom(y) \wedge purple(y)$ ^ $\neg(x=y) \wedge \forall z\ (mushroom(z) \wedge purple(z)) \rightarrow ((x=z) \vee (y=z))$
- **Clinton is not tall.**
  - $\neg tall(Clinton)$
- **X is above Y iff X is on directly on top of Y or there is a pile of one or more other objects directly on top of one another starting with X and ending with Y.**
  - $\forall x\ \forall y\ above(x,y) \leftrightarrow (on(x,y) \vee\ \exists z\ (on(x,z) \wedge above(z,y)))$

Properties of quantifiers

- $\forall x \, \forall y$ is the same as $\forall y \, \forall x$
- $\exists x \, \exists y$ is the same as $\exists y \, \exists x$

- $\exists x \, \forall y$ 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
- $\forall x \, Likes(x,IceCream) \quad \neg\exists x \, \neg Likes(x,IceCream)$
- $\exists x \, Likes(x,Broccoli) \qquad \neg\forall x \, \neg Likes(x,Broccoli)$

Connections between All and Exists

- We can relate sentences involving $\forall$ and $\exists$ using De Morgan's laws:
  - $(\forall x)\ \neg P(x) \leftrightarrow \neg(\exists x)\ P(x)$
  - $\neg(\forall x)\ P \leftrightarrow (\exists x)\ \neg P(x)$
  - $(\forall x)\ P(x) \leftrightarrow \neg\ (\exists x)\ \neg P(x)$
  - $(\exists x)\ P(x) \leftrightarrow \neg(\forall x)\ \neg P(x)$

Example: A simple genealogy KB in FOL

- **Build a small genealogy knowledge base using FOL that**
  - contains facts of immediate family relations (spouses, parents, etc.)
  - contains definitions of more complex relations (ancestors, relatives)
  - is able to answer queries about relationships between people
- **Predicates:**
  - parent(x, y), child(x, y), father(x, y), daughter(x, y), etc.
  - spouse(x, y), husband(x, y), wife(x,y)
  - ancestor(x, y), descendant(x, y)
  - male(x), female(y)
  - relative(x, y)
- **Facts:**
  - husband(Joe, Mary), son(Fred, Joe)
  - spouse(John, Nancy), male(John), son(Mark, Nancy)
  - father(Jack, Nancy), daughter(Linda, Jack)
  - daughter(Liz, Linda)
  - etc.

- **Rules for genealogical relations**
  - $(\forall x,y)$ parent(x, y) $\leftrightarrow$ child (y, x)
  - $(\forall x,y)$ father(x, y) $\leftrightarrow$ parent(x, y) $\wedge$ male(x) (similarly for mother(x, y))
  - $(\forall x,y)$ daughter(x, y) $\leftrightarrow$ child(x, y) $\wedge$ female(x) (similarly for son(x, y))
  - $(\forall x,y)$ husband(x, y) $\leftrightarrow$ spouse(x, y) $\wedge$ male(x) (similarly for wife(x, y))
  - $(\forall x,y)$ spouse(x, y) $\leftrightarrow$ spouse(y, x) (**spouse relation is symmetric**)
  - $(\forall x,y)$ parent(x, y) $\rightarrow$ ancestor(x, y)
  - $(\forall x,y)(\exists z)$ parent(x, z) $\wedge$ ancestor(z, y) $\rightarrow$ ancestor(x, y)
  - $(\forall x,y)$ descendant(x, y) $\leftrightarrow$ ancestor(y, x)
  - $(\forall x,y)(\exists z)$ ancestor(z, x) $\wedge$ ancestor(z, y) $\rightarrow$ relative(x, y)
  - (related by common ancestry)
  - $(\forall x,y)$ spouse(x, y) $\rightarrow$ relative(x, y) (related by marriage)
  - $(\forall x,y)(\exists z)$ relative(z, x) $\wedge$ relative(z, y) $\rightarrow$ relative(x, y) (**transitive**)
  - $(\forall x,y)$ relative(x, y) $\leftrightarrow$ relative(y, x) (**symmetric**)
- **Queries**
  - ancestor(Jack, Fred) /* the answer is yes */
  - relative(Liz, Joe) /* the answer is yes */
  - relative(Nancy, Matthew)
  - /* no answer in general, no if under closed world assumption */
  - $(\exists z)$ ancestor(z, Fred) $\wedge$ ancestor(z, Liz)

Axioms for Set Theory in FOL

- 1. The only sets are the empty set and those made by adjoining something to a set:
  - $\forall s\ set(s) \Leftrightarrow (s=EmptySet) \vee (\exists x,r\ Set(r) \wedge s=Adjoin(s,r))$
- 2. The empty set has no elements adjoined to it:
  - $\sim \exists x,s\ Adjoin(x,s)=EmptySet$
- 3. Adjoining an element already in the set has no effect:
  - $\forall x,s\ Member(x,s) \Leftrightarrow s=Adjoin(x,s)$
- 4. The only members of a set are the elements that were adjoined into it:
  - $\forall x,s\ Member(x,s) \Leftrightarrow \exists y,r\ (s=Adjoin(y,r) \wedge (x=y \vee Member(x,r)))$
- 5. A set is a subset of another iff all of the 1st set's members are members of the 2^{nd}:
  - $\forall s,r\ Subset(s,r) \Leftrightarrow (\forall x\ Member(x,s) \Rightarrow Member(x,r))$
- 6. Two sets are equal iff each is a subset of the other:
  - $\forall s,r\ (s=r) \Leftrightarrow (subset(s,r) \wedge subset(r,s))$
- 7. Intersection
  - $\forall x,s1,s2\ member(X,intersection(S1,S2)) \Leftrightarrow member(X,s1) \wedge member(X,s2)$
- 8. Union
  - $\exists x,s1,s2\ member(X,union(s1,s2)) \Leftrightarrow member(X,s1) \vee member(X,s2)$

Using FOL

- **Domain M**: the set of all objects in the world (of interest)
- **Assertions**: sentences added to KB by using TELL (as in propositional logic)
  - TELL (KB, Person(Richard))
- **Queries/Goals**: ask questions of the KB. Any query that is logically entailed by the knowledge base should be answered affirmatively.
  - ASK (KB, Person(Richard))

Semantics of FOL

- **Model:** an interpretation of a set of sentences such that every sentence is *True*
- **A sentence is**
  - **satisfiable** if it is true under some interpretation
  - **valid** if it is true under all possible interpretations
  - **inconsistent** if there does not exist any interpretation under which the sentence is true
- **Logical consequence:** S |= X if all models of S are also models of X

Axioms, definitions and theorems

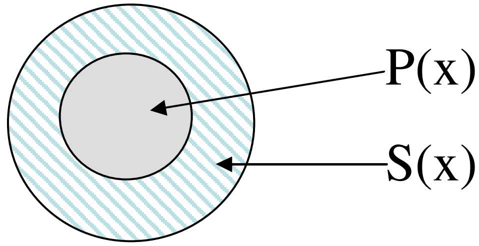
- **Axioms** are facts and rules that attempt to capture all of the (important) facts and concepts about a domain; axioms can be used to prove **theorems**
  - Mathematicians don't want any unnecessary (dependent) axioms –ones that can be derived from other axioms
  - Dependent axioms can make reasoning faster, however
  - Choosing a good set of axioms for a domain is a kind of design problem
- A **definition** of a predicate is of the form "p(X) ↔ …" and can be decomposed into two parts
  - **Necessary** description: "p(x) → …"
  - **Sufficient** description "p(x) ← …"
  - Some concepts don't have complete definitions (e.g., person(x))

More on definitions

- Examples: define father(x, y) by parent(x, y) and male(x)
  - parent(x, y) is a necessary (**but not sufficient**) description of
    father(x, y)
    - father(x, y) → parent(x, y)
  - parent(x, y) ^ male(x) ^ age(x, 35) is a **sufficient** (**but not necessary**) description of father(x, y):
  - father(x, y) ← parent(x, y) ^ male(x) ^ age(x, 35)
  - parent(x, y) ^ male(x) is a **necessary and sufficient** description of father(x, y)
  - parent(x, y) ^ male(x) ↔ father(x, y)

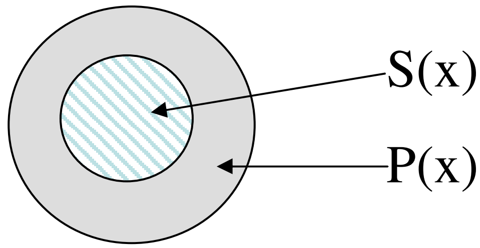
More on definitions

S(x) is a necessary condition of P(x)



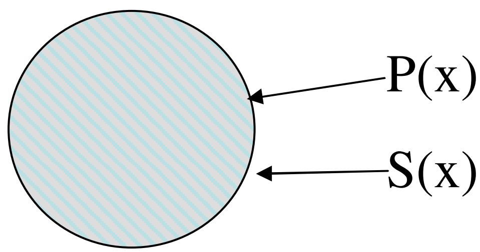
$(\forall x)\ P(x) \Rightarrow S(x)$

S(x) is a sufficient condition of P(x)



$(\forall x)\ P(x) \Leftarrow S(x)$

S(x) is a necessary and sufficient condition of P(x)



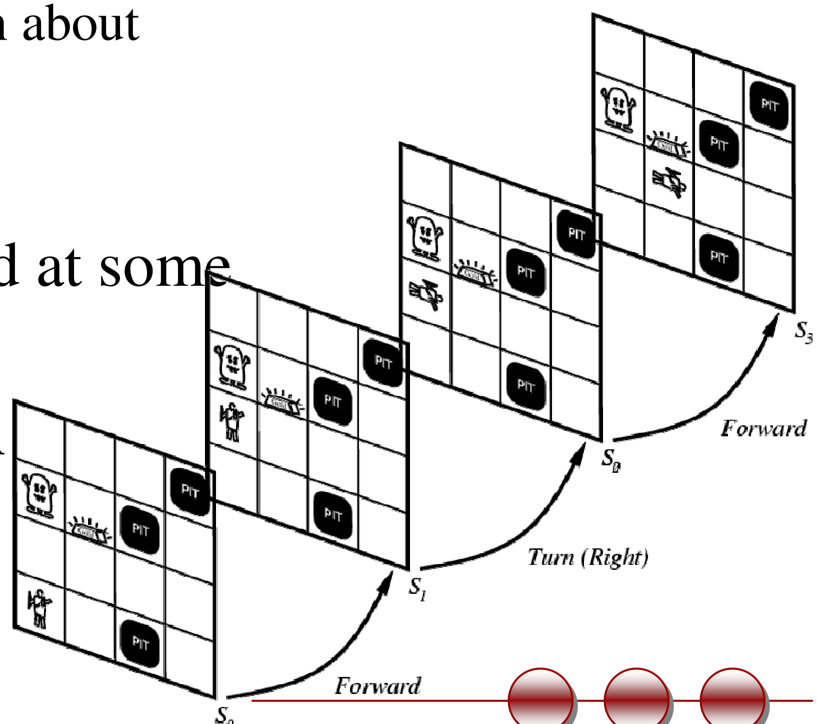
$(\forall x)\ P(x) \Leftrightarrow S(x)$

Higher-order logic

- FOL only allows to quantify over variables, and variables can only range over objects.
- HOL allows us to quantify over relations
- Example: (quantify over functions)
  - "two functions are equal iff they produce the same value for all arguments"
  - $\forall f \, \forall g \, (f = g) \leftrightarrow (\forall x \; f(x) = g(x))$
- Example: (quantify over predicates)
  - $\forall r \; transitive(r) \rightarrow (\forall xyz) \; r(x,y) \wedge r(y,z) \rightarrow r(x,z))$
- More expressive, but undecidable.

Representing change

- Representing change in the world in logic can be tricky.
- One way is just to change the KB
  - Add and delete sentences from the KB to reflect changes
  - How do we remember the past, or reason about changes?
- **Situation calculus** is another way
- A **situation** is a snapshot of the world at some instant in time
- When the agent performs an action A in situation S1, the result is a new situation S2.



Situation calculus

- A **situation** is a snapshot of the world at an interval of time during which nothing changes
- Every true or false statement is made with respect to a particular situation.
  - Add **situation variables** to every predicate.
  - at(Agent,1,1) becomes at(Agent,1,1,s0): at(Agent,1,1) is true in situation (i.e., state) s0.
  - Alernatively, add a special 2^{nd}-order predicate, **holds(f,s),** that means "f is true in situation s." E.g., holds(at(Agent,1,1),s0)
- Add a new function, **result(a,s),** that maps a situation s into a new situation as a result of performing action a. For example, result(forward, s) is a function that returns the successor state (situation) to s
- Example: The action agent-walks-to-location-y could be represented by
  - $(\forall x)(\forall y)(\forall s)$ (at(Agent,x,s) $\wedge \neg$onbox(s)) $\rightarrow$ at(Agent,y,result(walk(y),s))

Deducing hidden properties

- From the perceptual information we obtain in situations, we can **infer properties of locations**
  - $\forall l,s\ at(Agent,l,s) \wedge Breeze(s) \rightarrow Breezy(l)$
  - $\forall l,s\ at(Agent,l,s) \wedge Stench(s) \rightarrow Smelly(l)$
- Neither Breezy nor Smelly need situation arguments because pits and Wumpuses do not move around

Deducing hidden properties II

- We need to write some rules that relate various aspects of a single world state (as opposed to across states)
- There are two main kinds of such rules:
  - **Causal rules** reflect the assumed direction of causality in the world:
    - $(\forall l1,l2,s)$ At(Wumpus,l1,s) $\wedge$ Adjacent(l1,l2) $\rightarrow$ Smelly(l2)
    - $(\forall$ l1,l2,s) At(Pit,l1,s) $\wedge$ Adjacent(l1,l2) $\rightarrow$ Breezy(l2)
  - Systems that reason with causal rules are called **model-based reasoning systems**
  - **Diagnostic rules** infer the presence of **hidden properties** directly from the percept-derived information. We have already seen two diagnostic rules:
    - $(\forall$ l,s) At(Agent,l,s) $\wedge$ Breeze(s) $\rightarrow$ Breezy(l)
    - $(\forall$ l,s) At(Agent,l,s) $\wedge$ Stench(s) $\rightarrow$ Smelly(l)

Representing change: The frame problem

- **Frame axioms**: If property x doesn't change as a result of applying action a in state s, then it stays the same.
  - $On(x, z, s) \wedge Clear(x, s) \rightarrow$
    $On(x, table, Result(Move(x, table), s)) \wedge$
    $\neg On(x, z, Result(Move(x, table), s))$
  - $On(y, z, s) \wedge y \neq x \rightarrow On(y, z, Result(Move(x, table), s))$
  - The proliferation of frame axioms becomes very cumbersome in complex domains

The frame problem II

- **Successor-state axiom**: General statement that characterizes every way in which a particular predicate can become true:
  - Either it can be **made true**, or it can **already be true and not be changed**:
  - On (x, table, Result(a,s)) $\leftrightarrow$
    [On (x, z, s) $\wedge$ Clear (x, s) $\wedge$ a = Move(x, table)] v
    [On (x, table, s) $\wedge$ a $\neq$ Move (x, z)]
- In complex worlds, where you want to reason about longer chains of action, even these types of axioms are too cumbersome
  - Planning systems use special-purpose inference methods to reason about the expected state of the world at any point in time during a multi-step plan

Qualification problem

- Qualification problem:
  - How can you possibly characterize every single effect of an action, or every single exception that might occur?
  - When I put my bread into the toaster, and push the button, it will become toasted after two minutes, unless…
    - The toaster is broken, or…
    - The power is out, or…
    - I blow a fuse, or…
    - A neutron bomb explodes nearby and fries all electrical components, or…
    - A meteor strikes the earth, and the world we know it ceases to exist, or…

Ramification problem

- Similarly, it's just about impossible to characterize every side effect of every action, at every possible level of detail:
  - When I put my bread into the toaster, and push the button, the bread will become toasted after two minutes, and…
    - The crumbs that fall off the bread onto the bottom of the toaster over tray will also become toasted, and…
    - Some of the aforementioned crumbs will become burnt, and…
    - The outside molecules of the bread will become "toasted," and…
    - The inside molecules of the bread will remain more "breadlike," and…
    - The toasting process will release a small amount of humidity into the air because of evaporation, and…
    - The heating elements will become a tiny fraction more likely to burn out the next time I use the toaster, and…
    - The electricity meter in the house will move up slightly, and…

Knowledge engineering!

- Modeling the "right" conditions and the "right" effects at the "right" level of abstraction is very difficult
- Knowledge engineering (creating and maintaining knowledge bases for intelligent reasoning) is an entire field of investigation
- Many researchers hope that automated knowledge acquisition and machine learning tools can fill the gap:
  - Our intelligent systems should be able to **learn** about the conditions and effects, just like we do!
  - Our intelligent systems should be able to learn when to pay attention to, or reason about, certain aspects of processes, depending on the context!

Knowledge engineering in FOL

1. Identify the task
2. Assemble the relevant knowledge
3. Decide on a vocabulary of predicates, functions, and constants
4. Encode general knowledge about the domain
5. Encode a description of the specific problem instance
6. Pose queries to the inference procedure and get answers
7. Debug the knowledge base

Goal-based agents

- Once the gold is found, it is necessary to change strategies. So now we need a new set of action values.
- We could encode this as a rule:
  - $(\forall s)$ Holding(Gold,s) $\rightarrow$ GoalLocation([1,1]),s)
- We must now decide how the agent will work out a sequence of actions to accomplish the goal.
- Three possible approaches are:
  - **Inference**: good versus wasteful solutions **(Next topic!)**
  - **Search**: make a problem with operators and set of states
  - **Planning**: to be discussed later