CS W4701
Artificial Intelligence

Fall 2013

Chapter 8:

First Order Logic

Jonathan Voris

(based on slides by Sal Stolfo)

Gomoku Tournament

- Thanks to everyone for playing!
- 84 students participated
- 6 rounds played
- Over 700 bonus points awarded

Gomoku Tournament

- Congratulations to our winners!
  - 1st place: Kevin Roark
  - 2nd place: Shiyu Song
  - 3rd place: Kaili Zhang
- Honorable mentions:
  - Guillaume Le Chenadec
  - Jiuyang Zhao
  - Dewei Zhu

Assignment 4

- Implement inference algorithms for:
  - Forward chaining
  - Backward chaining
  - Resolution (optional)
- Inputs:
  - Mode
  - KB file
  - Q to be entailed

Assignment 3

- Due in 2 weeks
  - Tuesday December 10th @ 11:59:59 PM EST
- Please follow submission instructions
  - Pretty please?
- Submit:
  - Install script
  - Execution script
  - Test files
  - Documentation

Outline

- Why FOL?
- Syntax and semantics of FOL
- Using FOL
- Wumpus world in FOL
- Knowledge engineering in FOL

Pros and cons of propositional logic

- ☺ Propositional logic is declarative
- ☺ Propositional logic allows partial/disjunctive/negated information
  - (Unlike most data structures and databases)
- ☺ Propositional logic is compositional:
  - Meaning of $B_{1,1} \wedge P_{1,2}$ is derived from meaning of $B_{1,1}$ and of $P_{1,2}$
- ☺ Meaning in propositional logic is context-independent
  - (Unlike natural language, where meaning depends on context)
- ☹ Propositional logic has very limited expressive power
  - (Unlike natural language)
  - E.g., cannot say "pits cause breezes in adjacent squares“
    - Except by writing one sentence for each square

Problems with Propositional Logic

Propositional Logic is Weak

- 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 (FOL) 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))

Example

- Consider the problem of representing the following information:
  - Every person is mortal.
  - Confucius is a person.
  - Confucius is mortal.
- How can these sentences be represented so that we can infer the third sentence from the first two?

Another Example

- In PL we have to create propositional symbols to stand for all or part of each sentence. For example, we might have:

  P = “person”; Q = “mortal”; R = “Confucius”

- so the above 3 sentences are represented as:

  $P \rightarrow Q$; $R \rightarrow P$; $R \rightarrow Q$

- Although the third sentence is entailed by the first two, we needed an explicit symbol, R, to represent an individual, Confucius, who is a member of the classes “person” and “mortal”
- To represent other individuals we must introduce separate symbols for each one, with some way to represent the fact that all individuals who are “people” are also “mortal”

Wumpus World Agent

- Some atomic propositions:

  S12 = There is a stench in cell (1,2)

  B34 = There is a breeze in cell (3,4)

  W22 = The Wumpus is in cell (2,2)

  V11 = We have visited cell (1,1)

  OK11 = Cell (1,1) is safe.

  etc
- Some rules:

  (R1) $\neg S11 \rightarrow \neg W11 \wedge \neg W12 \wedge \neg W21$

  (R2) $\neg S21 \rightarrow \neg W11 \wedge \neg W21 \wedge \neg W22 \wedge \neg W31$

  (R3) $\neg S12 \rightarrow \neg W11 \wedge \neg W12 \wedge \neg W22 \wedge \neg W13$

  (R4) $S12 \rightarrow W13 \vee W12 \vee W22 \vee W11$

  etc
- Note that the lack of variables requires us to give similar rules for each cell

Propositional Wumpus Agent Shortcomings

- Lack of variables prevents stating more general rules
  - We need a set of similar rules for each cell
- Change of the KB over time is difficult to represent
  - Standard technique is to index facts with the time when they're true
  - This means we have a separate KB for every time point
- Propositional logic quickly becomes impractical, even for very small worlds

First-order Logic

- Whereas propositional logic assumes the world contains facts,
- First-order logic (like natural language) assumes the world contains
  - Objects: people, houses, numbers, colors, baseball games, wars, …
  - Relations: red, round, prime, brother of, bigger than, part of, comes between, …
  - Functions: father of, best friend, one more than, plus, …

Syntax of FOL: Basic Elements

- Constants KingJohn, 2, CU,...
- Predicates Brother, >,...
- Functions Sqrt, LeftLegOf,...
- Variables x, y, a, b,...
- Connectives $\neg$, $\Rightarrow$, $\wedge$, $\vee$, $\Leftrightarrow$
- Equality =
- Quantifiers $\forall$, $\exists$

FOL Examples

- *Brother(KingJohn) -> RichardTheLionheart*
- *Brother(RichardTheLionheart) -> KingJohn*
- *Length(LeftLegOf(Richard))*
- *Length(LeftLegOf(KingJohn))*

Predicates vs Functions

- Predicates are relationships between things
  - Objects in
  - Truth out
- Functions are mappings between things
  - Objects in
  - Objects out

Predicates vs Functions

- Predicates are essentially functions with Boolean output, but
- Predicates combine symbols to form **atomic sentences**
  - True or false (under a given model)
- Functions combine symbols to form **terms**
  - Expressions which refer to objects

Atomic Sentences

Atomic sentence = *predicate* ($term_1$,...,$term_n$)
or $term_1$ = $term_2$

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

Predicates vs Functions

- Predicates:
  - Holiday(November 28 2013) – true
  - Short(Bill de Blasio) – false
  - Brother(Peyton Manning, Eli Manning) - true
- Functions:
  - SittingNextTo(student) – student
  - InCity(person) – New York
  - Brother(Peyton Manning) - Eli Manning

A Note About Functions

- Not a function in the programming sense
- Really just a weird naming convention
- Not providing a definition
  - No need to explain what an eye is to reason about how many people have
- Parallels to lambda in Lisp

Complex sentences

- Complex sentences are made from atomic sentences using connectives

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

E.g. *Sibling(KingJohn,Richard)* $\Rightarrow$ *Sibling(Richard,KingJohn)*

$>(1,2) \vee \leq (1,2)$

$>(1,2) \wedge \neg >(1,2)$

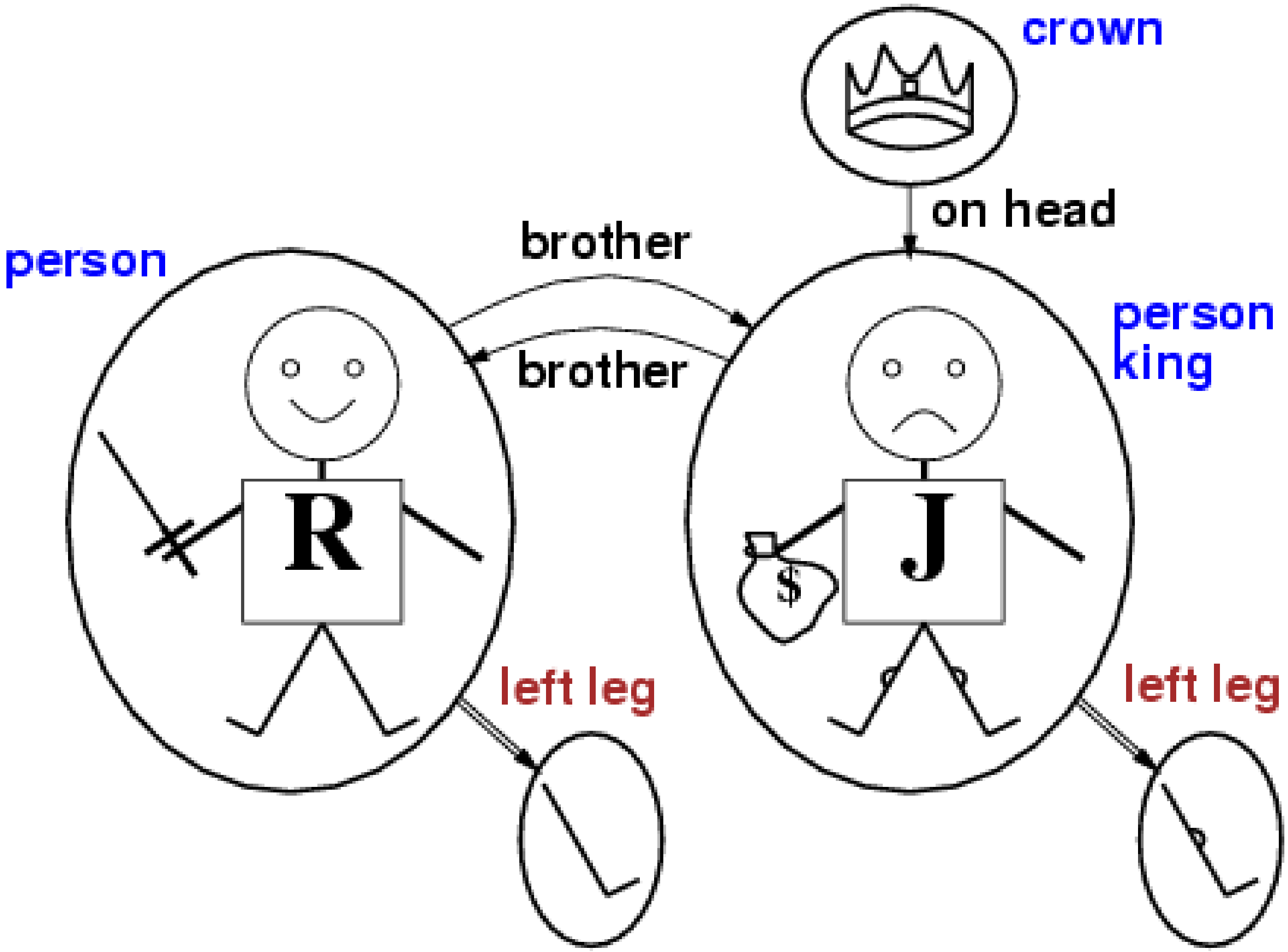
Truth in first-order logic

- Sentences are true with respect to a model which includes an interpretation
- Model contains info needed to evaluate sentences
  - Objects or domain elements
  - Interpretation pairing symbols with objects
  - Relationships between objects (predicates)
  - Functions with objects
- Interpretation specifies referents for

constant symbols	→	objects
predicate symbols	→	relations
function symbols	→	functional relations

- An atomic sentence *predicate*$(term_1,\ldots,term_n)$ is true iff the objects referred to by $term_1,\ldots,term_n$ are in the relation referred to by *predicate*

Models for FOL: Example



FOL Examples

- Intended interpretation:
  - Richard refers to Richard the Lionheart
  - John refers to King John
  - Brother: brotherhood relation
  - OnHead: Relation between crown and King John
  - Person refers to Richard and John
  - King refers to John
  - LeftLeg reflects mapping in figure

FOL Examples

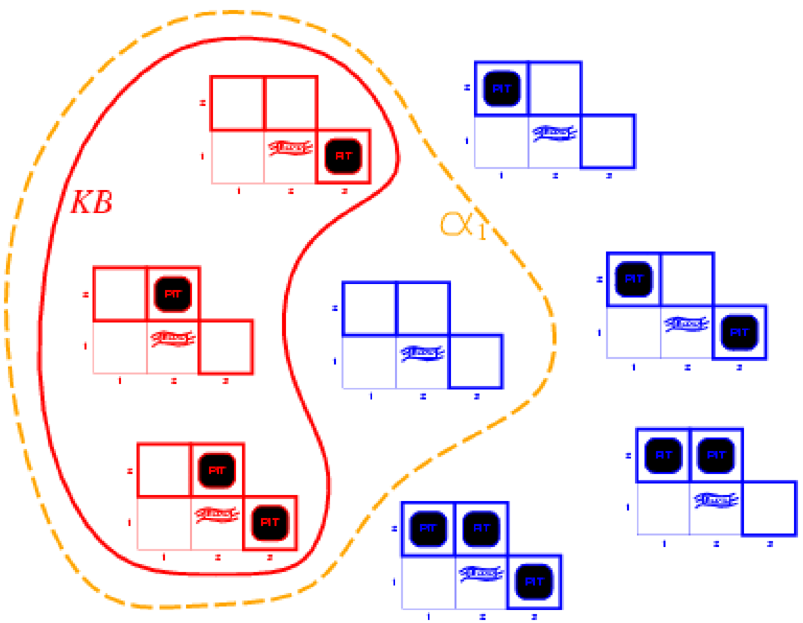
- Another possible interpretation:
  - Richard refers to the crown
  - John refers to King John's left leg
  - Brother: false if an input is wearing crown
  - OnHead: Relation between Richard and King John
  - Person refers to legs
  - King refers to Richard
  - LeftLeg maps to right arms
- 5 objects in model
  - $2^5 = 25$ possible interpretations just for John and Richard

FOL Examples

- Not all objects need names
  - What do you call the crown?
- Objects can have multiple names
  - Richard and John can refer to the crown
- Were we better off with first order logic in this regard?
- Duty of the knowledge base to avoid these things

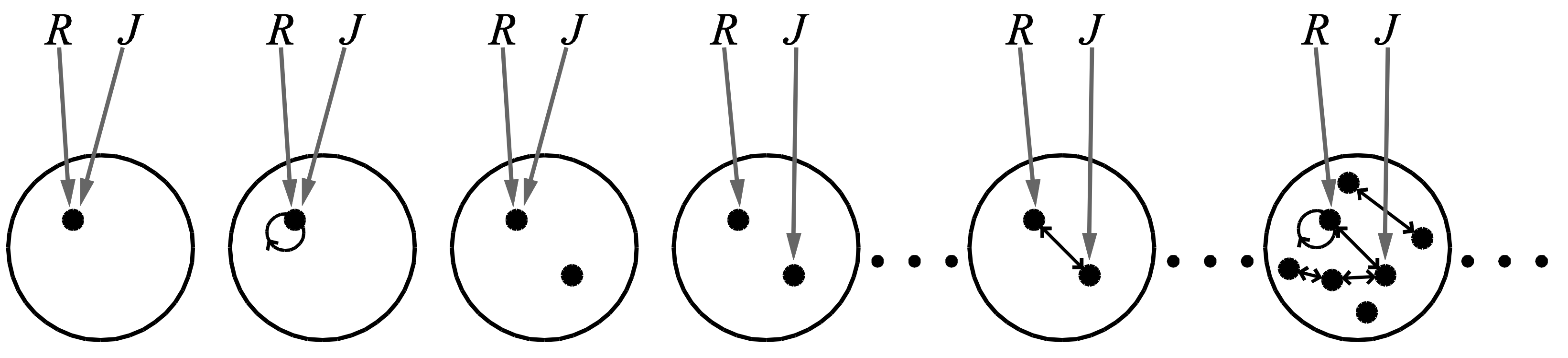
A Note About Models

- Recall in propositional logic, reasoning was performed with respect to all possible models



A Note About Models

- Same is true for FOL
- But instead of T/F, now need to consider
  - Objects
  - Interpretations
  - Relationships
- Example: 1 or 2 objects, 2 names, 1 relationship



Quantifiers

- FOL lets us refer to objects

Quantifiers

- FOL lets us refer to objects
- Has this solved the propositional logic naming problem?

Quantifiers

- FOL lets us refer to objects
- Has this solved the propositional logic naming problem?
- Still need to enumerate!

Quantifiers

- FOL lets us refer to objects
- Has this solved the propositional logic naming problem?
- Still need to enumerate!
- We want to be able to refer to collections of objects

Quantifiers

- FOL lets us refer to objects
- Has this solved the propositional logic naming problem?
- Still need to enumerate!
- We want to be able to refer to collections of objects
  - All pits have breezes next to them
  - All sons have fathers
- We want to refer to some example of an object
  - There is a bird that can't fly
  - There is a bridge that connects Brooklyn and Manhattan

Universal Quantification

- $\forall$<*variables*> <*sentence*>

All Columbia students are smart:
$\forall$x Student(x, Columbia) $\Rightarrow$ Smart(x)

- $\forall$x *P* is true in a model *m* iff *P* is true with *x* being each possible object in the model

- Roughly speaking, equivalent to the conjunction of instantiations of *P*

Student(KingJohn,Columbia) $\Rightarrow$ Smart(KingJohn)
$\wedge$ Student(Richard, Columbia) $\Rightarrow$ Smart(Richard)
$\wedge$ Student(Columbia, Columbia) $\Rightarrow$ Smart(Columbia)
$\wedge$ ...

A Common Mistake to Avoid

- Typically, $\Rightarrow$ is the main connective with $\forall$
- Common mistake: using $\wedge$ as the main connective with $\forall$:

  $\forall x$ Student(x, Columbia) $\wedge$ Smart(x)

  means “Everyone is a Columbia student and everyone is smart”

Existential Quantification

- $\exists$<*variables*> <*sentence*>

- Someone student at Columbia is smart:
- $\exists x$ Student(x,Columbia) $\wedge$ Smart(x)

- $\exists x$ *P* is true in a model *m* iff *P* is true with *x* being some possible object in the model

- Roughly speaking, equivalent to the disjunction of instantiations of *P*

  Student(KingJohn, Columbia) $\wedge$ Smart(KingJohn)
  $\vee$ Student (Richard, Columbia) $\wedge$ Smart(Richard)
  $\vee$ Student (Columbia, Columbia) $\wedge$ Smart(Columbia)
  $\vee$ ...

Another Common Mistake to Avoid

- Typically, $\wedge$ is the main connective with $\exists$
- Common mistake: using $\Rightarrow$ as the main connective with $\exists$:

  $$\exists x \text{ Student}(x, \text{Columbia}) \Rightarrow \text{Smart}(x)$$

  is true if there is anyone who is not a Columbia student!

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

Equality

- $term_1 = term_2$ is true under a given interpretation if and only if $term_1$ and $term_2$ refer to the same object
  - Father(John) = Henry
- Can also be used to state facts about functions
- E.g., definition of *Sibling* in terms of *Parent*:

  $\forall x,y\ Sibling(x,y) \Leftrightarrow [\neg(x = y) \land \exists m,f\ \neg(m = f) \land Parent(m,x) \land Parent(f,x) \land Parent(m,y) \land Parent(f,y)]$

Different Semantics

- How would you say Huey has 2 brothers: Dewey & Louie

Different Semantics

- How would you say Huey has 2 brothers: Dewey & Louie
- Brother(Huey, Dewey) ^ Brother(Huey, Louie)

Different Semantics

- How would you say Huey has 2 brothers: Dewey & Louie
- Brother(Huey, Dewey) ^ Brother(Huey, Louie)
- What if Dewey *is* Louie?

Different Semantics

- How would you say Huey has 2 brothers: Dewey & Louie
- Brother(Huey, Dewey) ^ Brother(Huey, Louie)
- What if Dewey *is* Louie?
  - Brother(Huey, Dewey) ^ Brother(Huey, Louie) ^ (Dewey != Louie)

Different Semantics

- How would you say Huey has 2 brothers: Dewey & Louie
- Brother(Huey, Dewey) ^ Brother(Huey, Louie)
- What if Dewey *is* Louie?
  - Brother(Huey, Dewey) ^ Brother(Huey, Louie) ^ (Dewey != Louie)
- There could be a fourth brother!

Different Semantics

- How would you say Huey has 2 brothers: Dewey & Louie
- Brother(Huey, Dewey) ^ Brother(Huey, Louie)
- What if Dewey *is* Louie?
  - Brother(Huey, Dewey) ^ Brother(Huey, Louie) ^ (Dewey != Louie)
- There could be a fourth brother!
  - Brother(Huey, Dewey) ^ Brother(Huey, Louie) ^ (Dewey != Louie) ^ $\forall$x Brother(x, Huey) $\Rightarrow$ (x=Dewey v x=Louie)

Different Semantics

- FOL is easier to work with if you assume **database semantics**
- Symbols refer to distinct objects
- Closed world
  - Everything not explicitly true is false
- Domain closure
  - No unnamed elements
- Now Brother(Huey, Dewey) ^ Brother(Huey, Louie) works as intended

Best Semantics?

- Are you always certain two objects aren’t the same?
- No right or wrong semantics
- Pick what is useful
  - Concise
  - Natural

Using FOL

The kinship domain:

- Brothers are siblings

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

- One's mother is one's female parent

  $\forall$m,c *Mother(c)* = m $\Leftrightarrow$ *(Female(m)* $\wedge$ *Parent(m,c))*

- “Sibling” is symmetric

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

Using FOL

The set domain:

- $\forall s\ Set(s) \Leftrightarrow (s = \{\}) \lor (\exists x,s_2\ Set(s_2) \land s = \{x|s_2\})$
- $\neg\exists x,s\ \{x|s\} = \{\}$
- $\forall x,s\ x \in s \Leftrightarrow s = \{x|s\}$
- $\forall x,s\ x \in s \Leftrightarrow [\ \exists y,s_2\}\ (s = \{y|s_2\} \land (x = y \lor x \in s_2))]$
- $\forall s_1,s_2\ s_1 \subseteq s_2 \Leftrightarrow (\forall x\ x \in s_1 \Rightarrow x \in s_2)$
- $\forall s_1,s_2\ (s_1 = s_2) \Leftrightarrow (s_1 \subseteq s_2 \land s_2 \subseteq s_1)$
- $\forall x,s_1,s_2\ x \in (s_1 \cap s_2) \Leftrightarrow (x \in s_1 \land x \in s_2)$
- $\forall x,s_1,s_2\ x \in (s_1 \cup s_2) \Leftrightarrow (x \in s_1 \lor x \in s_2)$

Interacting with FOL KBs

- Suppose a wumpus-world agent is using an FOL KB and perceives a smell and a breeze (but no glitter) at *t=5*:

  `Tell`(KB,Percept([Smell,Breeze,None],5))
  `Ask`(KB,∃a BestAction(a,5))

- I.e., does the KB entail some best action at *t=5*?

- Answer: *Yes*, {*a/Shoot*} ← substitution (binding list)

- Given a sentence *S* and a substitution σ,
- *S*σ denotes the result of plugging σ into *S*; e.g.,
  S = Smarter(x,y)
  σ = {x/Hillary,y/Bill}
  *S*σ = Smarter(Hillary,Bill)

- `Ask`(KB,S) returns some/all σ such that KB ⊨ σ

Knowledge Base for the Wumpus World

- Perception
  - $\forall$t,s,b Percept([s,b,Glitter],t) $\Rightarrow$ Glitter(t)

- Reflex
  - $\forall$t Glitter(t) $\Rightarrow$ BestAction(Grab,t)

Deducing Hidden Properties

- $\forall$x,y,a,b *Adjacent*([x,y],[a,b]) $\Leftrightarrow$ [a,b] $\in$ {[x+1,y], [x-1,y],[x,y+1],[x,y-1]}
- Properties of squares:
  - $\forall$s,t *At*(Agent,s,t) $\wedge$ Breeze(t) $\Rightarrow$ Breezy(s)
- Squares are breezy near a pit:
  - $\forall$s Breezy(s) $\Leftrightarrow$ $\exists$ r Adjacent(r,s) $\wedge$ Pit(r)

Knowledge Engineering in FOL

1. Identify the task
   - Just like PEAS!
2. Assemble the relevant knowledge
   - Work with experts
3. Decide on a vocabulary of predicates, functions, and constants
   - Formalize concepts into logic names
4. Encode general knowledge about the domain
   - Record specific logical axioms

Knowledge Engineering in FOL

5. Encode a description of the specific problem instance
   – Agent is fed precepts as logical statements
6. Pose queries to the inference procedure and get answers
   – Axioms + facts = interesting facts (hopefully)
7. Debug the knowledge base

Wumpus Domain

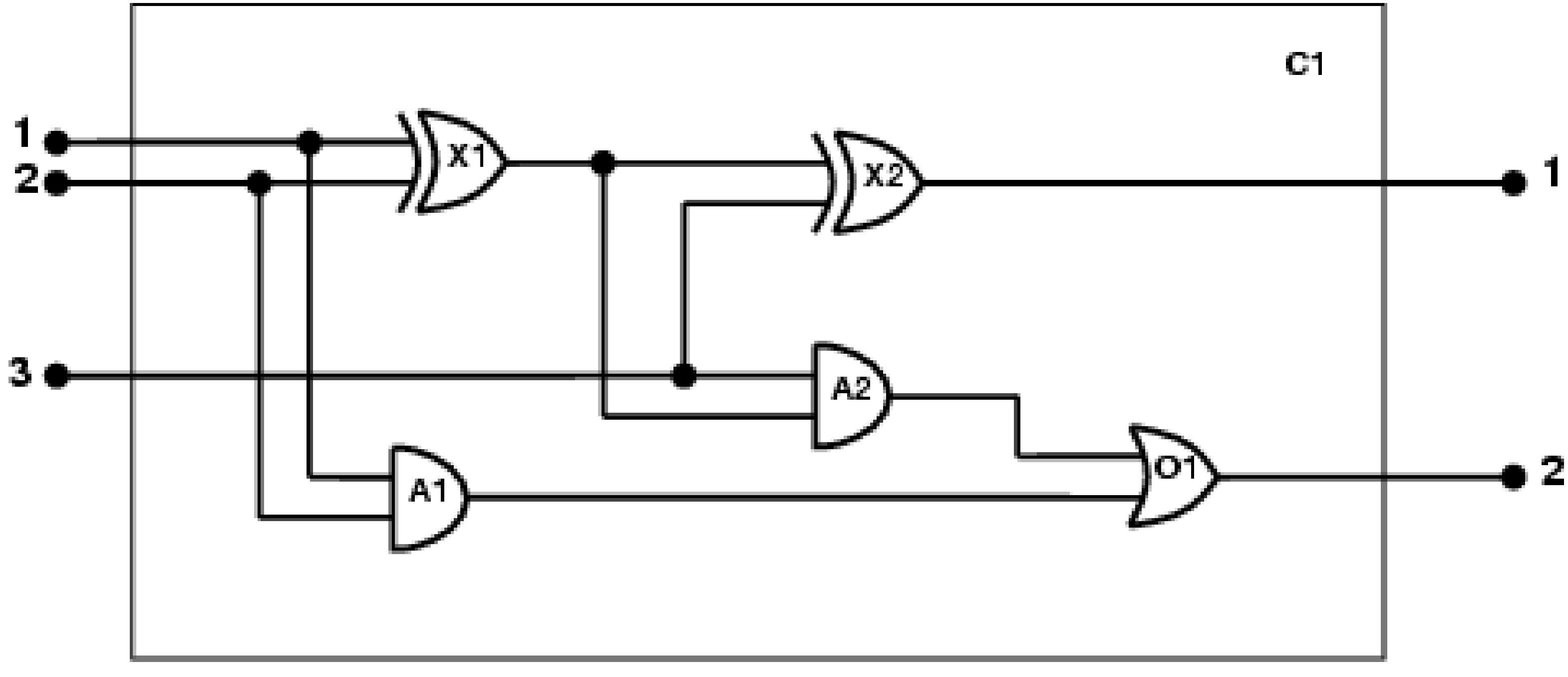
1. Identify the task
   - Answering questions? Choosing actions?
   - Track location or reported in percept?
2. Assemble the relevant knowledge
   - Pits cause breezes, etc
3. Decide on a vocabulary of predicates, functions, and constants
   - Pits objects or predicates?
   - Orientation a function or predicate?
   - Do objects' locations depend on time?

Wumpus Domain

4. Encode general knowledge about the domain
   – $\forall$s Breezy(s) $\Leftrightarrow$ $\exists$r Adjacent(r,s) ^ Pit(r)
5. Encode a description of the specific problem instance
   – Explore map instance and observe squares
6. Pose queries to the inference procedure and get answers
   – Is it safe to move up?
   – What is my best action?
7. Debug the knowledge base
   – Problem with $\forall$s Breezy(s) $\Rightarrow$ $\exists$r Adjacent(r,s) ^ Pit(r)

The Electronic Circuits Domain

One-bit full adder



The Electronic Circuits Domain

1. Identify the task
   - Does the circuit actually add properly? (circuit verification)
   - Input/output of certain gates?
   - Loops?
2. Assemble the relevant knowledge
   - What is known about circuits?
   - Composed of wires and gates
   - Types of gates (AND, OR, XOR, NOT)
   - Irrelevant: physical wire paths, size, shape, color, cost of gates, cost

The Electronic Circuits Domain

3. Decide on a vocabulary
   - Symbols, predicates, functions, etc.
   - Gates are objects
   - Behavior determined by type constants
   - Alternatives:

     $Type(X_1) = XOR$

     $Type(X_1, XOR)$

     $XOR(X_1)$
   - Terminals, signals, etc.

The Electronic Circuits Domain

4. Encode general knowledge of the domain
   - Two connected terminals have the same signal:
     $\forall t_1, t_2\ Connected(t_1, t_2) \Rightarrow Signal(t_1) = Signal(t_2)$
   - Signals are 1 or 0
     $\forall t\ Signal(t) = 1 \vee Signal(t) = 0$
   - The two signals are distinct
     $1 \neq 0$
   - Connected is commutative
     $\forall t_1, t_2\ Connected(t_1, t_2) \Leftrightarrow Connected(t_2, t_1)$

The Electronic Circuits Domain

4. Encode general knowledge of the domain
   - Gate definitions:
     - $\forall g\ Type(g) = OR \Rightarrow Signal(Out(1,g)) = 1 \Leftrightarrow \exists n\ Signal(In(n,g)) = 1$
     - $\forall g\ Type(g) = AND \Rightarrow Signal(Out(1,g)) = 0 \Leftrightarrow \exists n\ Signal(In(n,g)) = 0$
     - $\forall g\ Type(g) = XOR \Rightarrow Signal(Out(1,g)) = 1 \Leftrightarrow Signal(In(1,g)) \neq Signal(In(2,g))$
     - $\forall g\ Type(g) = NOT \Rightarrow Signal(Out(1,g)) \neq Signal(In(1,g))$

The Electronic Circuits Domain

5. Encode the specific problem instance

$Type(X_1) = XOR$ $Type(X_2) = XOR$

$Type(A_1) = AND$ $Type(A_2) = AND$

$Type(O_1) = OR$

$Connected(Out(1,X_1),In(1,X_2))$ $Connected(In(1,C_1),In(1,X_1))$

$Connected(Out(1,X_1),In(2,A_2))$ $Connected(In(1,C_1),In(1,A_1))$

$Connected(Out(1,A_2),In(1,O_1))$ $Connected(In(2,C_1),In(2,X_1))$

$Connected(Out(1,A_1),In(2,O_1))$ $Connected(In(2,C_1),In(2,A_1))$

$Connected(Out(1,X_2),Out(1,C_1))$ $Connected(In(3,C_1),In(2,X_2))$

$Connected(Out(1,O_1),Out(2,C_1))$ $Connected(In(3,C_1),In(1,A_2))$

The Electronic Circuits Domain

6. Pose queries to the inference procedure
   - What are the possible sets of values of all the terminals for the adder circuit?
   - Returns input/output table:

   $\exists i_1,i_2,i_3,o_1,o_2$ Signal(In(1,C_1)) = $i_1 \wedge$ Signal(In(2,C_1)) = $i_2 \wedge$ Signal(In(3,C_1)) = $i_3 \wedge$ Signal(Out(1,C_1)) = $o_1 \wedge$ Signal(Out(2,C_1)) = o_2

7. Debug the knowledge base

   May have omitted assertions like $1 \neq 0$

Summary

- First-order logic:
  - Objects and relations are semantic primitives
  - Syntax: constants, functions, predicates, equality, quantifiers
- Increased expressive power
  - Sufficient to efficiently define Wumpus World