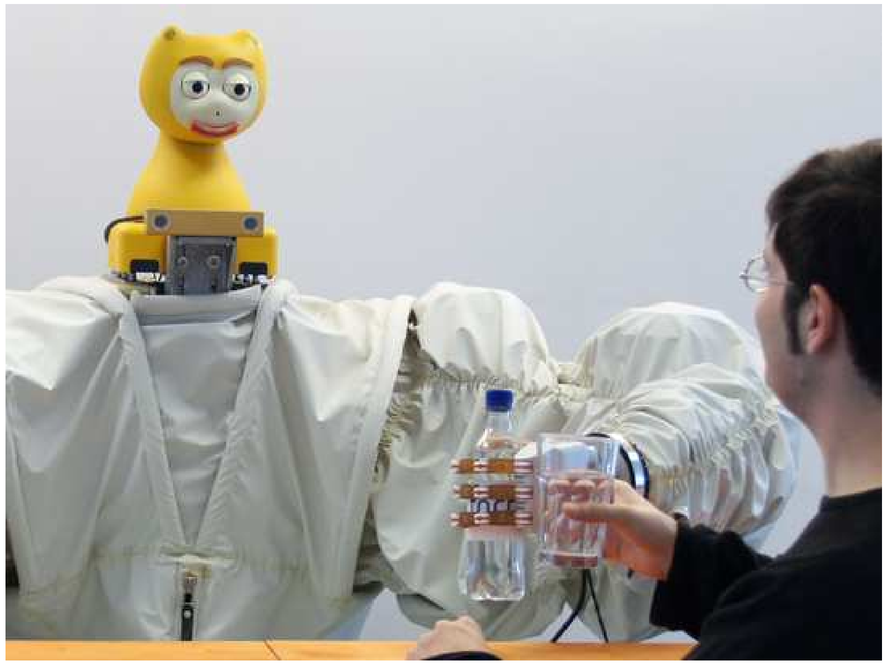
Vorlesung

Grundlagen der Künstlichen Intelligenz

Reinhard Lafrenz / Prof. A. Knoll

Robotics and Embedded Systems
Department of Informatics – I6
Technische Universität München

www6.in.tum.de
lafrenz@in.tum.de
089-289-18136
Room 03.07.055



Wintersemester 2012/13 26.11.2012

Chapter 7,8 (3rd ed.)

Propositional and Frist-Order Logic

From the last lecture we know

- Propositional Logic
  - Restrictions to e.g. Horn Clauses

- Proof methods:
  - Resolution
  - Forward/Backward Chaining
  - DPLL algorithm
  - WalkSAT algorithm

Hard satisfiability problems

- Consider random 3-CNF sentences (with at most 3 variables per clause) e.g.,

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

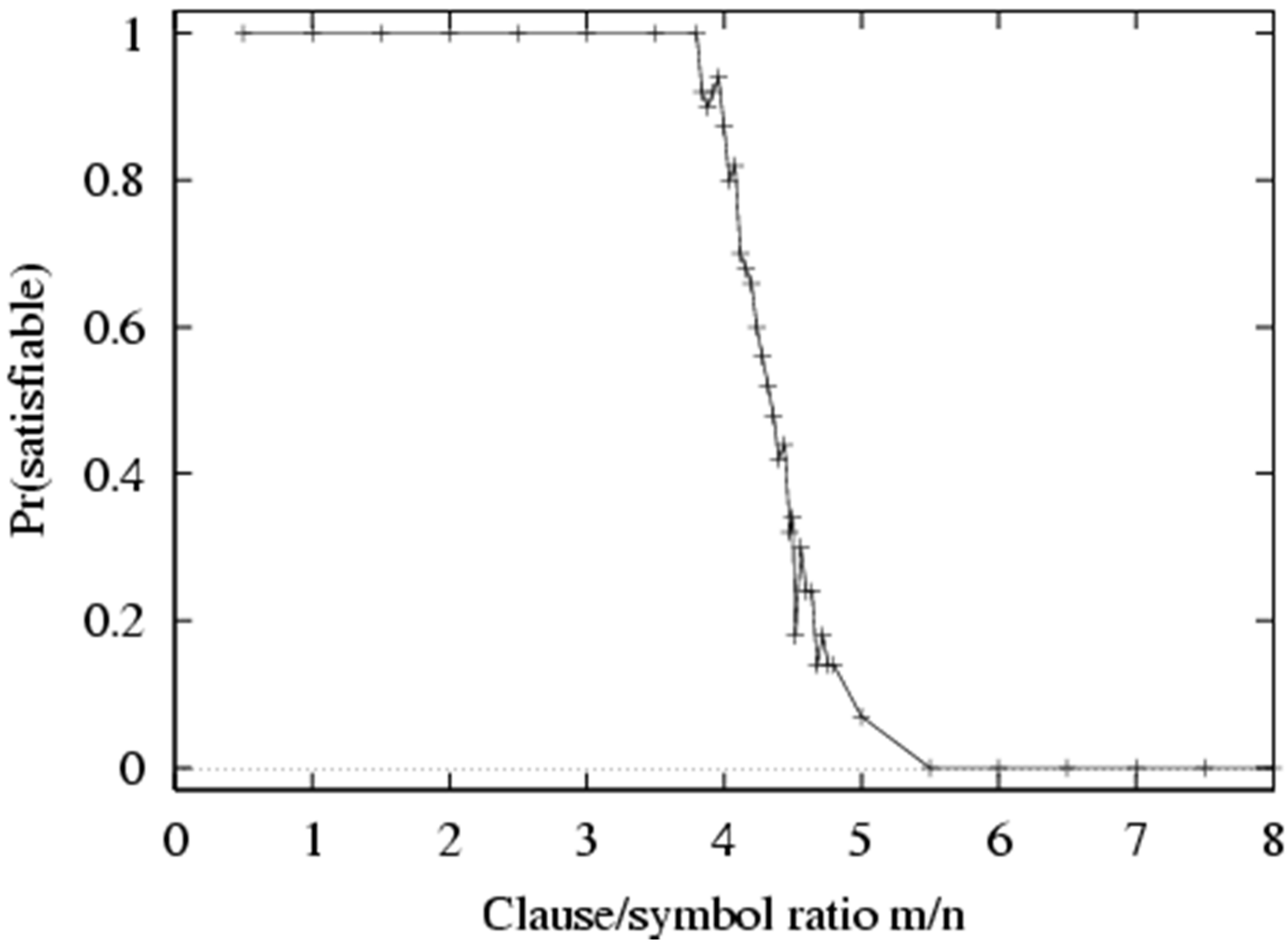
Analyse "hardness" of satisfiability problem using

m = number of clauses

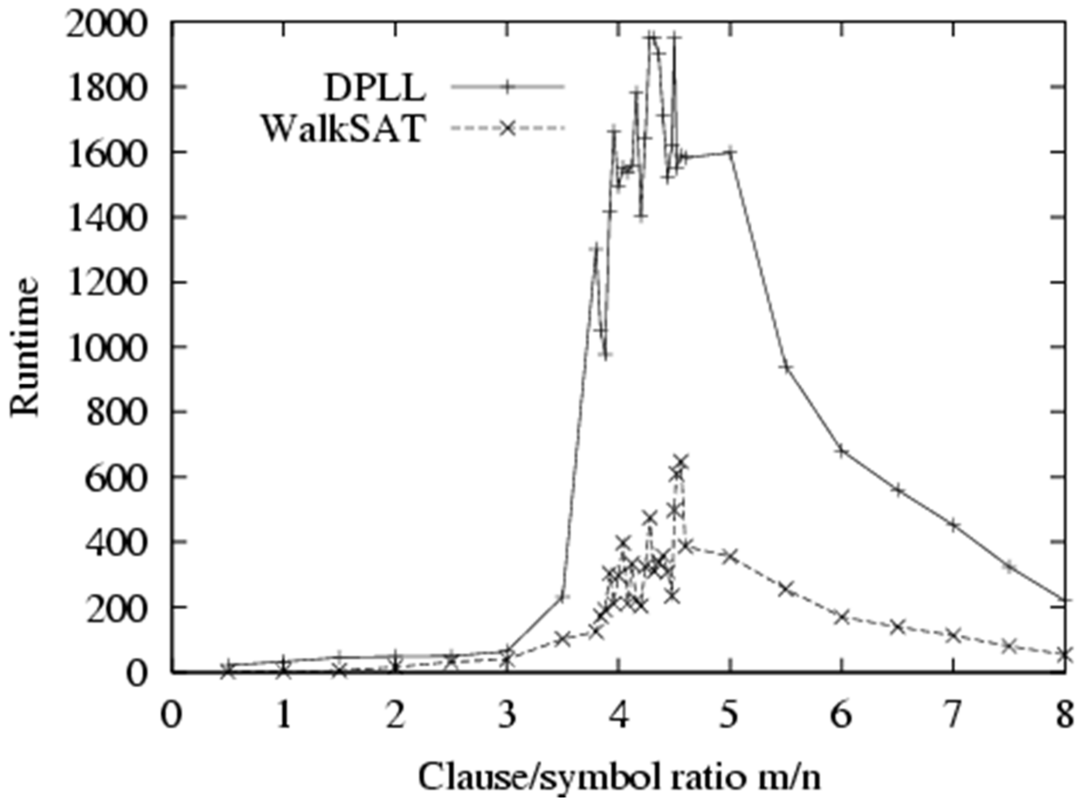
n = number of symbols

- Hard problems seem to cluster near $m/n = 4.3$ (critical point)

Hard satisfiability problems



Hard satisfiability problems



- Median runtime for 100 satisfiable random 3-CNF sentences, $n = 50$

Inference-based agents in the wumpus world

A wumpus-world agent using propositional logic:

$\neg P_{1,1}$	1
$\neg W_{1,1}$	1
$B_{x,y} \Leftrightarrow (P_{x,y+1} \vee P_{x,y-1} \vee P_{x+1,y} \vee P_{x-1,y})$	16
$S_{x,y} \Leftrightarrow (W_{x,y+1} \vee W_{x,y-1} \vee W_{x+1,y} \vee W_{x-1,y})$	16
$W_{1,1} \vee W_{1,2} \vee \ldots \vee W_{4,4}$	1
$\neg W_{1,1} \vee \neg W_{1,2}$	
$\neg W_{1,1} \vee \neg W_{1,3}$	$120 = (16^2 - 16)/2$
…	

- 64 distinct proposition symbols (16 x P, W, B, S)
- 155 sentences

```
function PL-WUMPUS-AGENT(percept) returns an action
    inputs: percept, a list, [stench,breeze,glitter]
    static: KB, initially containing the "physics" of the wumpus world
            x, y, orientation, the agent's position (init. [1,1]) and orient. (init. right)
            visited, an array indicating which squares have been visited, initially false
            action, the agent's most recent action, initially null
            plan, an action sequence, initially empty

    update x,y,orientation, visited based on action
    if stench then TELL(KB, S_x,y) else TELL(KB, ¬ S_x,y)
    if breeze then TELL(KB, B_x,y) else TELL(KB, ¬ B_x,y)
    if glitter then action ← grab
    else if plan is nonempty then action ← POP(plan)
    else if for some fringe square [i,j], ASK(KB, (¬ P_i,j ∧ ¬ W_i,j)) is true or
            for some fringe square [i,j], ASK(KB, (P_i,j ∨ W_i,j)) is false then do
        plan ← A*-GRAPH-SEARCH(ROUTE-PB([x,y], orientation, [i,j],visited))
        action ← POP(plan)
    else action ← a randomly chosen move
    return action
```

Expressiveness limitation of propositional logic

- KB contains "physics" sentences for every single square

- For every time *t* and every location [*x,y*]:

$$L^{t}_{x,y} \land \mathit{FacingRight}^{t} \land \mathit{Forward}^{t} \Rightarrow L^{t+1}_{x+1,y} \land \neg L^{t+1}_{x,y}$$

- Rapid proliferation of clauses

- Check for danger in a field:

$$OK^{t}_{x,y} \Leftrightarrow \neg P_{x,y} \land \neg (W_{x,y} \land \mathrm{WumpusAlive}^{t})$$

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)

BUT:

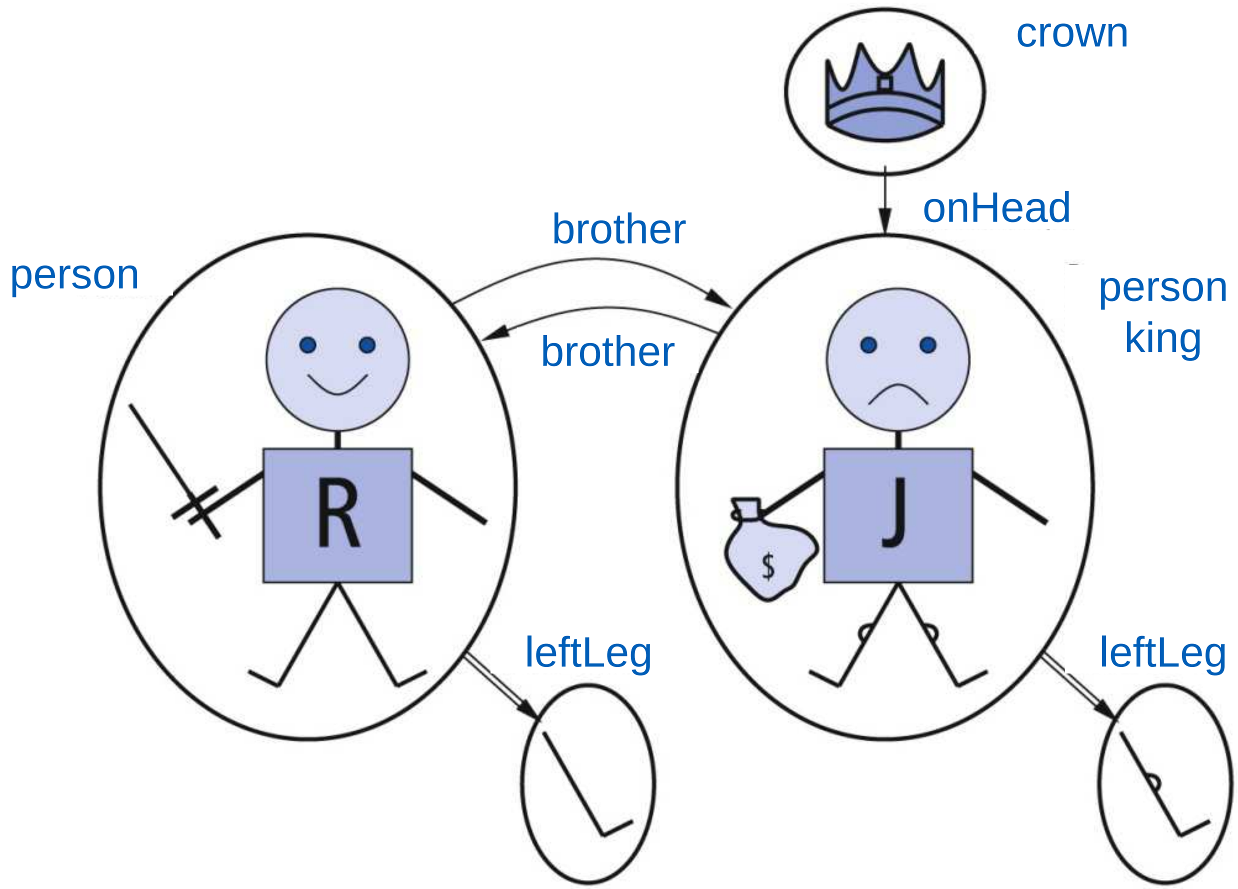
- ☹ Propositional logic has very limited expressive power
  - (unlike natural language)
  - E.g., cannot say "pits cause breezes in adjacent squares“
    - except by writing one sentence for each square

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, …
  - Relations: red, round, prime, brother of, bigger than, part of, comes between, …
  - Functions: father of, best friend, one more than, plus, …

Models for FOL: Example



Syntax of FOL: Basic elements

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

Atomic sentences

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

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

Examples:

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

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)*

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

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

First-Order-Logic: Syntax in BNF

$$
\begin{aligned}
Sentence &\rightarrow AtomicSentence \mid ComplexSentence \\
AtomicSentence &\rightarrow Predicate \mid Predicate(Term, \ldots) \mid Term = Term \\
ComplexSentence &\rightarrow (\ Sentence\ ) \mid [\ Sentence\ ] \\
&\mid \neg Sentence \\
&\mid Sentence \land Sentence \\
&\mid Sentence \lor Sentence \\
&\mid Sentence \Rightarrow Sentence \\
&\mid Sentence \Leftrightarrow Sentence \\
&\mid Quantifier\ Variable, \ldots\ Sentence \\
Term &\rightarrow Function(Term, \ldots) \\
&\mid Constant \\
&\mid Variable \\
Quantifier &\rightarrow \forall \mid \exists \\
Constant &\rightarrow A \mid X_1 \mid John \mid \ldots \\
Variable &\rightarrow a \mid x \mid s \mid \ldots \\
Predicate &\rightarrow True \mid False \mid After \mid Loves \mid Raining \mid \ldots \\
Function &\rightarrow Mother \mid LeftLeg \mid \ldots
\end{aligned}
$$

OPERATOR PRECEDENCE : $\neg, =, \land, \lor, \Rightarrow, \Leftrightarrow$

Truth in first-order logic

- Sentences are true (a model) or false with respect to an an interpretation
- 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*

Universal quantification

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

Everyone at TUM is smart:
$\forall x\ At(x,TUM) \Rightarrow Smart(x)$

- $\forall x$ *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*

$$\begin{aligned} & At(KingJohn,TUM) \Rightarrow Smart(KingJohn) \\ \wedge\ & At(Richard,TUM) \Rightarrow Smart(Richard) \\ \wedge\ & At(TUM,TUM) \Rightarrow Smart(TUM) \\ \wedge\ & \ldots \end{aligned}$$

A common mistake to avoid

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

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

  $\forall$x At(x,TUM) $\land$ Smart(x)

  means "Everyone is at TUM and everyone is smart"

Existential quantification

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

- Someone at TUM is smart:
- $\exists x\ \mathrm{At}(x,\mathrm{TUM}) \land \mathrm{Smart}(x)\$$

- $\exists x\ P$ 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*

$$
\begin{aligned}
& \mathrm{At}(\mathrm{KingJohn},\mathrm{TUM}) \land \mathrm{Smart}(\mathrm{KingJohn}) \\
\lor\ & \mathrm{At}(\mathrm{Richard},\mathrm{TUM}) \land \mathrm{Smart}(\mathrm{Richard}) \\
\lor\ & \mathrm{At}(\mathrm{TUM},\mathrm{TUM}) \land \mathrm{Smart}(\mathrm{TUM}) \\
\lor\ & \ldots
\end{aligned}
$$

Another common mistake to avoid

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

$$\exists x \; At(x,TUM) \Rightarrow Smart(x)$$

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

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

De Morgan Rules

Quantified

- $\forall x \ \neg P \equiv \neg \exists x \ P$
- $\neg \forall x \ P \equiv \exists x \ \neg P$
- $\forall x \ P \equiv \neg \exists x \ \neg P$
- $\exists x \ P \equiv \neg \forall x \ \neg P$

Not quantified

$\neg (P \lor \neg Q) \equiv \neg P \land Q$

$\neg (P \land Q) \equiv \neg P \lor \neg Q$

$P \land Q \equiv \neg(\neg P \lor \neg Q)$

$P \lor Q \equiv \neg(\neg P \land \neg Q)$

Equality

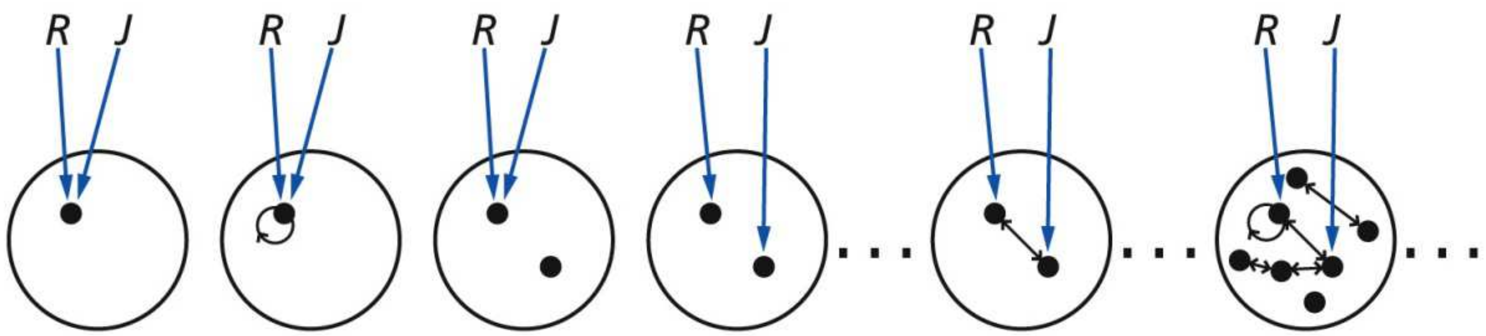
- $term_1 = term_2$ is true under a given interpretation if and only if $term_1$ and $term_2$ refer to the same object

- E.g., definition of *Sibling* in terms of *Parent*:

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

Possible models

- Language with 2 constant symbols and 1 binary relation



- Up to 6 objects: 137.506.194.466 possibilities

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 – defining exact semantics

Write the sentence

“Richard has 2 brothers, John and Geoffrey” in FOL

Brother(John, Richard) $\wedge$ *Brother*(Geoffrey, Richard)

- Is this enough?
- What if Geoffrey = John?

Add $\wedge$ (John $\neq$ Geoffrey)

- What if there are more brothers?

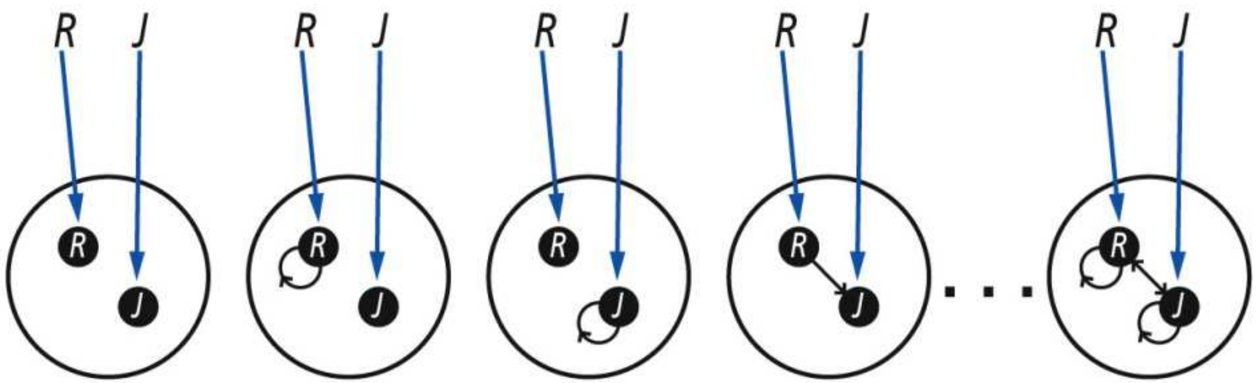
Brother(John, Richard) $\wedge$ *Brother*(Geoffrey, Richard)

$\wedge$ (John $\neq$ Geoffrey)

$\wedge$ ($\forall$x *Brother*(x, Richard) $\Rightarrow$ (x=John $\vee$ x=Geoffrey)

Using FOL – database semantics

Reconsider set of possible models



- Unique identities (John ≠ Geoffrey is implicit)
- Closed-world assumption (no constants not in the KB)

The number of possible models is reduced to $2^4 = 16$

Database semantics are used in logic programming languages

Using FOL

The set domain:

- $\forall s\ \mathit{Set}(s) \Leftrightarrow (s = \{\}\ ) \vee (\exists x,s_2\ \mathit{Set}(s_2) \wedge s = \{x|s_2\})$

- $\neg\exists x,s\ \{x|s\} = \{\}$

- $\forall x,s\ \ x \in s \Leftrightarrow s = \{x|s\}$

- $\forall x,s\ \ x \in s \Leftrightarrow [\ \exists y,s_2\}\ (s = \{y|s_2\} \wedge (x = y \vee x \in s_2))]$

- $\forall s_1,s_2\ \ \ s_1 \subseteq s_2 \Leftrightarrow (\forall x\ x \in s_1 \Rightarrow x \in s_2)$
- $\forall s_1,s_2\ \ \ (s_1 = s_2) \Leftrightarrow (s_1 \subseteq s_2 \wedge s_2 \subseteq s_1)$

- $\forall x,s_1,s_2\ \ \ x \in (s_1 \cap s_2) \Leftrightarrow (x \in s_1 \wedge x \in s_2)$

- $\forall x,s_1,s_2\ \ \ x \in (s_1 \cup s_2) \Leftrightarrow (x \in s_1 \vee x \in s_2)$

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)
  - Diagnostic rule---infer cause from effect

    $\forall$s Breezy(s) $\Rightarrow$ $\exists$r Adjacent(r,s) $\wedge$ Pit(r)
  - Causal rule---infer effect from cause

    $\forall$r Pit(r) $\Rightarrow$ [$\forall$s Adjacent(r,s) $\Rightarrow$ Breezy(s)]

Consideration of time

$\forall$t *HaveArrow*(t+1) $\Leftrightarrow$ *HaveArrow*(t) $\wedge$ $\neg$*Action*(*Shoot*, t))

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

Summary

- First-order logic:
  - objects and relations are semantic primitives
  - syntax: constants, functions, predicates, equality, quantifiers
- Increased expressive power: sufficient to define wumpus world including “hidden properties” such as “hasArrow”
-