Artificial Intelligence

09/18/2002

Limitations of Propositioinal Logic

- Cumbersome for large domains:
  - Man-Abraham, Man-Isaac, Man-Jacob
  - Woman-Sara, Woman-Rachel, Woman-Leah
  - Man-Abraham $\Rightarrow$ Human-Abraham
  - Woman-Sara $\Rightarrow$ Human-Sara
- Cannot deal with infinite domains.
- We'd like to say:
  - Abraham, Sara etc. are objects.
  - for all x, Man(x) $\Rightarrow$ Human(x)
  - for all n, Integer(n) $\Rightarrow$ Integer(n+1).

First Order Logic (FOL)

- Propositional logic can talk about: atomic facts
- FOL can talk about: objects, properties of objects, and relations between objects (including functions)
  - Examples of
    - Objects: people, numbers, states, days of week, Woojin Paik, the moon, the biggest bar in Boston, …
    - Relations: comes before, stupider than, brother of, bigger than, beats, …
    - Properties: stupid, tall, blue
    - Functions (relations in which there is only one ‘value’ for a given input): father of, inverse of, favorite wild flower of

FOL (a.k.a. Predicate Calculus)

- We identify the objects in our domain.
  - Abraham, Sara, Isaac, Rachel,
  - Father-of(Isaac), Mother-of(Isaac).
- Predicates specify properties of objects, and tuples of objects:
  - Man(Abraham), Woman(Sara),
  - Married(Abraham, Sara).
- Quantified formulas:
  - $\forall$ x Man(x) $\Rightarrow$ Human(x)
  - $\forall$ x $\exists$ y Loves(y,x).

FOL Syntax

Sentence ::= AtomicSentence | ComplexSentence

AtomicSentence ::= Predicate (Term, ….) | Term = Term

Term ::= Function (Term, …) | Constant | Variable

ComplexSetence ::= (Sentence) | Sentence Connective Sentence | ¬Sentence |
Quantifier Variable Sentence

Connective ::= $\vee$ | $\wedge$ | $\Leftrightarrow$ | $\Rightarrow$

Quantifier ::= $\forall$ | $\exists$

Constant ::= A | B | … | X | X1 | X2 | Woojin | Alex …

Variable ::= a | b | c | …| x | y | z

Predicate ::= Before | HasColor | Raining | AreBrothers

…

Function ::= Mother | LeftLegOf | log | Height …

FOL Definitions

- Connectives: and/or/implies/iff
- Predicates: specifies a relation between objects
- Constants: A,B, Woojin, Tweety
  - Name a specific object.
- Variables: x, y.
  - Refer to an object without naming it.
- Functions: MotherOf
  - Mapping from objects to objects.
- Terms: ways of referring objects in the world
  - Refer to objects (it can be constant, a variable, or a function)
- Atomic Sentences: IsBird(Tweety), LikesBirds(Sylvester)
  - Can be true or false
  - Correspond to propositional symbols P, Q

FOL: examples of legal sentences

$\vee_m \vee_c$ Mother(c) = m $\Leftrightarrow$ Female(m) ^ Parent(m,c)

$\vee_w \vee_h$ Husband(h,w) $\Leftrightarrow$ Male(h) ^ Spouse(h,w)

Father(Woojin) = JK

$\neg$SitsInChair(Lecturer)

$\vee_x$ Female(x) $\Leftrightarrow$ $\neg$Male(x)

Female(HJ)

$\vee_x \vee_y$ Sibling(x,y) $\Leftrightarrow$ $\neg$(x=y) ^ $\exists_p$ Parent (p,x) ^ Parent(p,y)

Proposition Logic/FOL Semantics

Steps to

- First we define what is an interpretation
- Then we decide how to evaluate a sentence under a given interpretation
- Then, we say the meaning (i.e. semantic) of a sentence is the set of interpretations in which the sentence evaluates to TRUE

Constant Symbols in Interpretation

- Example: Woojin Paik, Lecturer, HJ, …
- An interpretation does the following
  - With each constant symbol, it associates an object
  - Some constant symbols may be given the same object
  - One interpretation of the constant symbols: Woojin Paik and Lecturer would tie Woojin and Lecturer to the same object, HJ is a different object
- In the following, we will implicitly assume that each constant symbol is associated with its own distinct object
  - i.e. we will not consider, say, interpretations where Woojin and Lecturer are bound to the same object
  - Good thing: it means we can look at examples of interpretations without getting utterly swallowed up in complexity
  - Bad thing: it will not emphasize just how many different interpretations there can be

Predicate Symbols

$\vee_m \vee_c$ Mother(c) = m $\Leftrightarrow$ Female(m) ^ Parent(m,c)

$\vee_w \vee_h$ Husband(h,w) $\Leftrightarrow$ Male(h) ^ Spouse(h,w)

Father(Woojin) = JK

¬SitsInChair(Lecturer)

$\vee_x$ Female(x) $\Leftrightarrow$ ¬Male(x)

Female(HJ)

OlderThan(HJ,Woojin)

$\vee_x \vee_y$ Sibling(x,y) $\Leftrightarrow$ ¬(x=y) ^ $\exists_p$ Parent (p,x) ^ Parent(p,y)

- In an interpretation, each Predicate symbol is specified by a set of Tuples (a Predicate with N arguments is a set of N-tuples)
  - e.g., One interpretation could have
    Female = { (HJ) }
    OlderThan = {(HJ,Woojin), (JK,Woojin), (Woojin,Lecturer), (Lecturer, HJ)}
    Sibling = {}

Function Symbols

Father(Woojin) = JK

- In an interpretation, each function symbol is specified by a set of tuples.
  - A function with N arguments is a set of N+1-tuples
  - e.g., one interpretation could have:

    Father = { (Woojin, JK), (JK, HJ), (HJ, JK), (Lecturer, JK) }
  - With four constant symbols, a function symbol that took two arguments would need 16 3-tuples in the interpretation
    - e.g., AgeDifference(Woojin,HJ) = 8

      AgeDifference = {(Woojin,Woojin,0), (Woojin,HJ,8), …}

Terms

Term ::= Function (Term, …) | Constant | Variable

- In an interpretation, a Term is evaluated as follows
  - If it is a constant, merely return the object associated with that constant in the interpretation
  - If it's a function applied to arguments
    - Evaluate all the arguments in the current interpretation
    - Search (among the tuples associated this function symbol) for the tuple which has the same first N elements as the N argument values
    - Once you've found the one that matches (there must be precisely one, according to the definition of the interpretation of a function symbol) return the N+1'th element of that tuple

Example

- Assume in interpretation that

  Father = {(Woojin,JK), (JK, HJ), (HJ, Woojin), (Lecturer,Woojin)}

  AgeDifference = AgeDifference = {(Woojin,Woojin,0), (Woojin,HJ,8), …}

  Suppose you are asked to evaluate

  AgeDifference(Woojin,Father(Woojin))

  What would you reply?

Conceptual Graph (CG) Examples

- Conceptual graphs are formally defined in an abstract syntax that is independent of any notation
  - CGs can be represented in
    - the graphical *display form* (DF),
    - the formally defined *conceptual graph interchange form* (CGIF), and
    - the compact, but readable *linear form* (LF)
  - Every CG is represented in each of these three forms and is translated to a logically equivalent representation in predicate calculus and in the Knowledge Interchange Format (KIF).

CG Examples (cont'd)

- A cat is on a mat.
  - Display Form (DF)
  - Linear Form (LF)

    [Cat]→(On)→[Mat].
  - Conceptual Graph Interchange Form (CGIF)

    [Cat: *x] [Mat: *y] (On ?x ?y)

    (The symbols *x and *y are called *defining labels*. The matching symbols ?x and ?y are the *bound labels* that indicate references to the same instance of a cat *x* or a mat *y*.)
  - Knowledge Interchange Format (KIF)

    (exists ((?x Cat) (?y Mat)) (On ?x ?y))
  - Predicate Calculus

    $(\exists x\text{:Cat})(\exists y\text{:Mat})\text{on}(x,y)$.

CG Examples (cont'd)

- John is going to Boston by bus.
  - Display Form (DF)
  - Linear Form (LF)

    [Go]-

    (Agnt)→[Person: John]

    (Dest)→[City: Boston]

    (Inst)→[Bus].
  - Conceptual Graph Interchange Form (CGIF)

    [Go: *x] [Person: John *y] [City: Boston *z] [Bus: *w]

    (Agnt ?x ?y) (Dest ?x ?z) (Inst ?x ?z)

CG Examples (cont'd)

- John is going to Boston by bus.
  - Display Form (DF)

  - KIF

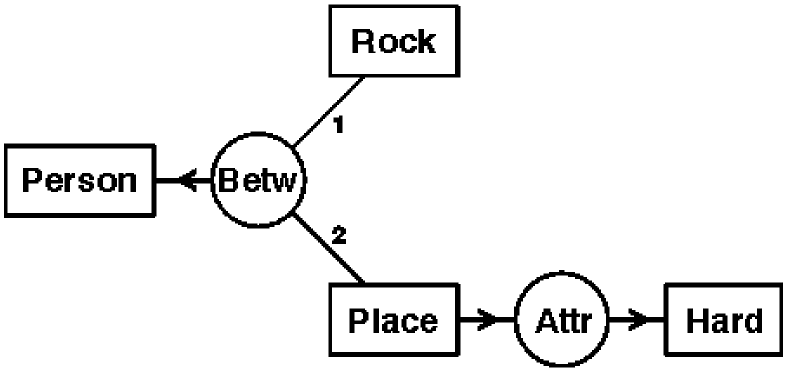
    (exists ((?x Go) (?y Person) (?z City) (?w Bus))
    (and (Name ?y John) (Name ?z Boston)
    (Agnt ?x ?y) (Dest ?x ?z) (Inst ?x ?w)))

  - Predicate Calculus

    $(\exists x\text{:Go})(\exists y\text{:Person})(\exists z\text{:City})(\exists w\text{:Bus})$
    $(\text{name}(y,\text{'John'}) \wedge \text{name}(z,\text{'Boston'}) \wedge$
    $\text{agnt}(x,y) \wedge \text{dest}(x,z) \wedge \text{inst}(x,w))$

CG Examples (cont'd)

- A person is between a rock and a hard place.
  - Display Form (DF)



  - LF

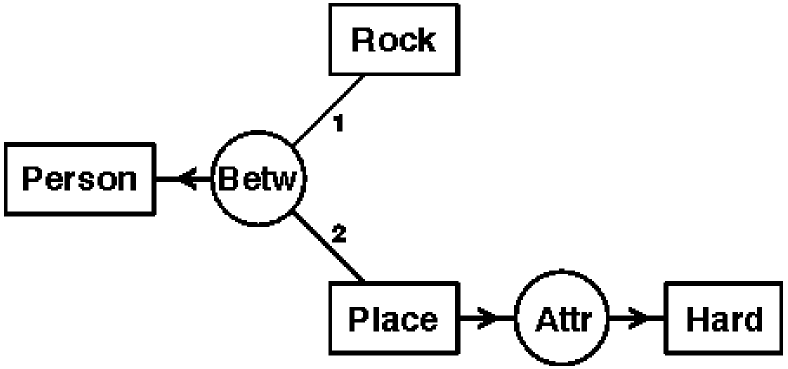
[Person]←(Betw)-
←1-[Rock]
←2-[Place]→(Attr)→[Hard].

  - CGIF

(Betw [Rock] [Place *x] [Person]) (Attr ?x [Hard])

CG Examples (cont'd)

- A person is between a rock and a hard place.
  - Display Form (DF)



  - KIF

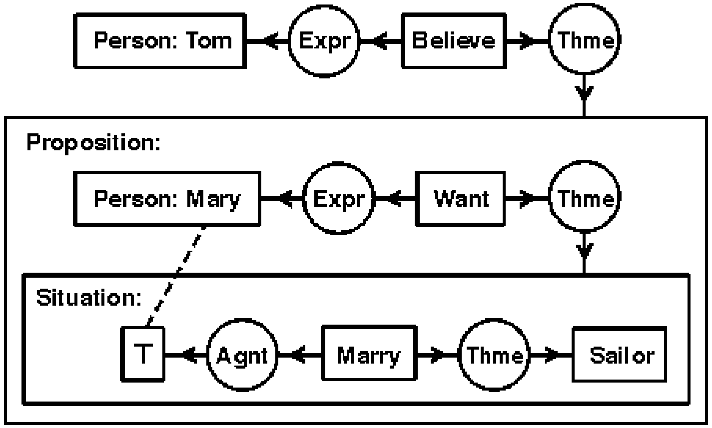
(exists ((?x person) (?y rock) (?z place) (?w hard))
(and (betw ?y ?z ?x) (attr ?z ?w)))

  - Predicate Calculus

$$(\exists x{:}Person)(\exists y{:}Rock)(\exists z{:}Place)(\exists w{:}Hard)$$
$$(betw(y,z,x) \wedge attr(z,w))$$

CG Examples (cont'd)

– **Tom believes that Mary wants to marry a sailor.**
– Display Form (DF)



– LF

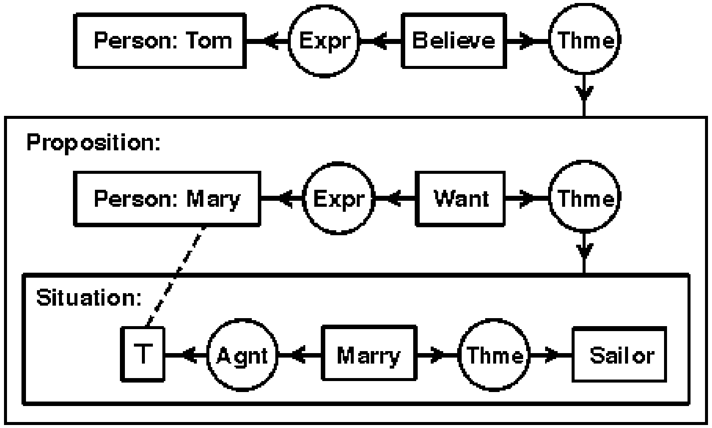
[Person: Tom]←(Expr)←[Believe]→(Thme)-
    [Proposition: [Person: Mary *x]←(Expr)←[Want]→(Thme)-
        [Situation: [?x]←(Agnt)←[Marry]→(Thme)→[Sailor]]].

– CGIF

[Person: *x1 'Tom'] [Believe *x2] (Expr ?x2 ?x1)
    (Thme ?x2 [Proposition:
        [Person: *x3 'Mary'] [Want *x4] (Expr ?x4 ?x3)
            (Thme ?x4 [Situation:
                [Marry *x5] (Agnt ?x5 ?x3) (Thme ?x5 [Sailor])])])

CG Examples (cont'd)

– **Tom believes that Mary wants to marry a sailor.**
– Display Form (DF)

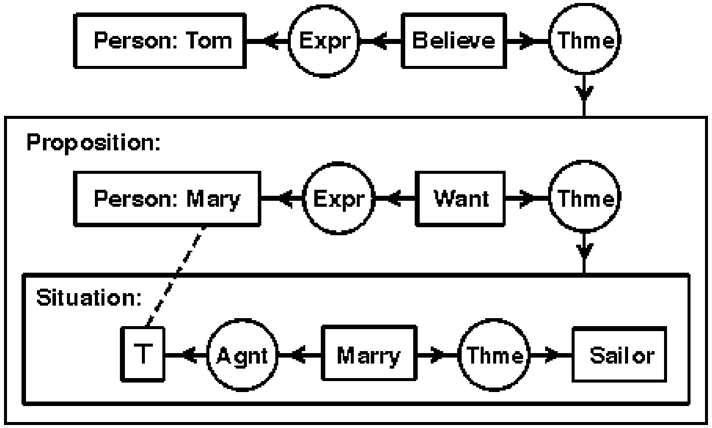


– KIF

(exists ((?x1 person) (?x2 believe))
(and (expr ?x2 ?x1)
(thme ?x2 (exists ((?x3 person) (?x4 want) (?x8 situation))
(and (name ?x3 'Mary) (expr ?x4 ?x3) (thme ?x4 ?x8)
(dscr ?x8 (exists ((?x5 marry) (?x6 sailor))
(and (Agnt ?x5 ?x3) (Thme ?x5 ?x6))))))))

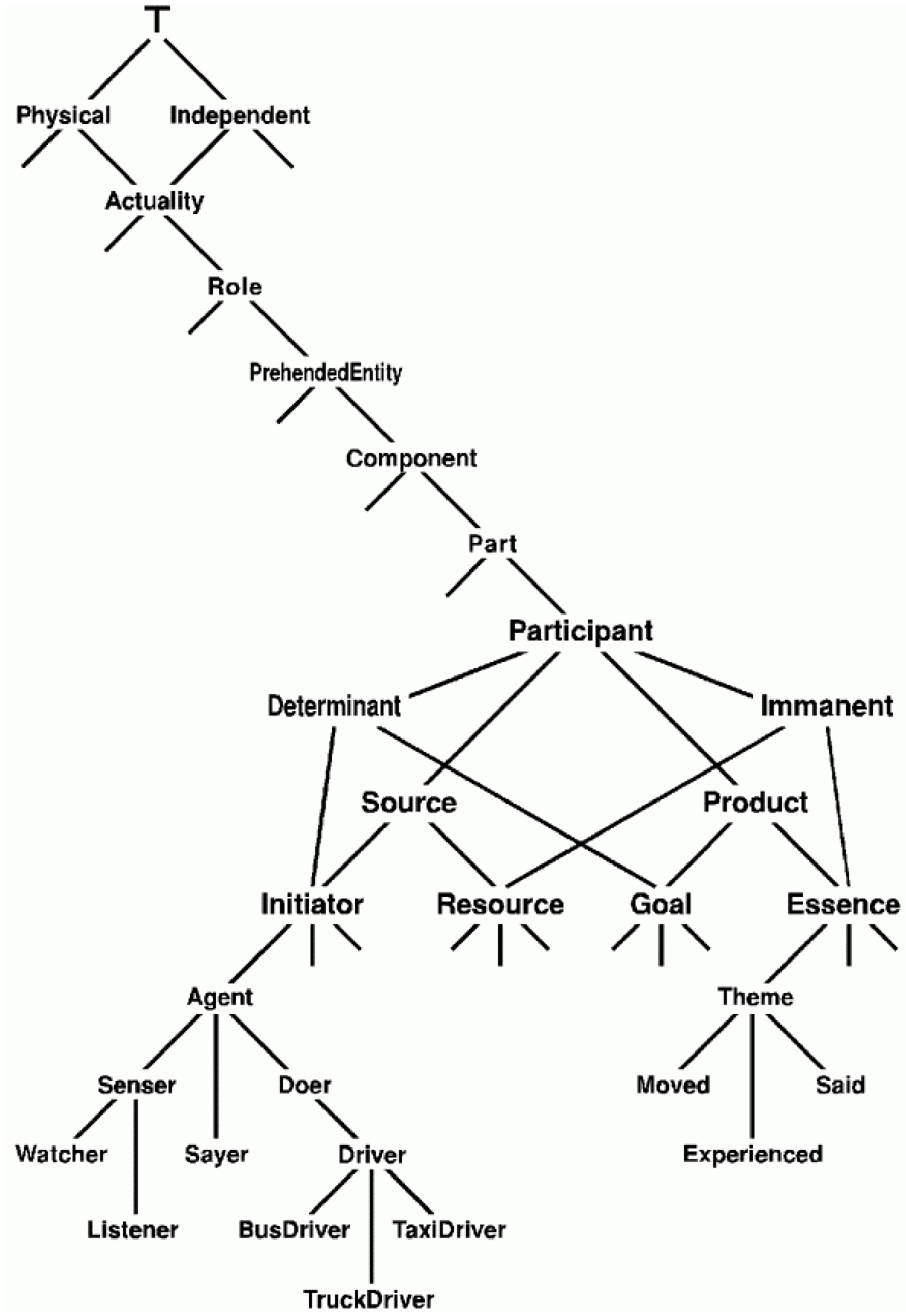
CG Examples (cont'd)

- **Tom believes that Mary wants to marry a sailor.**
- Display Form (DF)



- Predicate Calculus

(∃x1:Person)(∃x2:Believe)(expr(x1,x2) ∧
thme(x2, (∃x3:Person)(∃x4:Want)(∃x8:Situation)
(name(x3,'Mary') ∧ expr(x4,x3) ∧ thme(x4,x8) ∧
dscr(x8, (∃x5:Marry)(∃x6:Sailor)
(agnt(x5,x3) ∧ thme(x5,x6))))))


T
Physical
Independent
Actuality
Role
PrehendedEntity
Component
Part
Participant
Determinant
Immanent
Source
Product
Initiator
Resource
Goal
Essence
Agent
Theme
Senser
Doer
Moved
Said
Watcher
Sayer
Driver
Experienced
Listener
BusDriver
TaxiDriver
TruckDriver