CS 181: Natural Language Processing

Lecture 14: Semantic Representations

Kim Bruce
Pomona College
Spring 2008

Disclaimer: Slide contents borrowed from many sources on web!

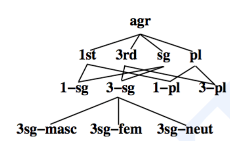
Types & Inheritance

Constraining Features

- Each feature structure is labeled by a type.
- Types have **appropriateness conditions** expressing which features are appropriate for it.
- Types organized into **hierarchy** -- more specific types inherit properties of more abstract ones.
- The unification operation is modified to unify the types of feature structures in addition to unifying the attributes and values.

Types

- Simple types:
  - Atomic features -- sg, pl, 3rd, ...
  - Arranged into multiple hierarchies:



  - Note unification moves down hierarchy!
  - Is this the best approach?

Complex Types

- Specify set of appropriate features
- Type restrictions on values of features
- Equality constraints between values.

$$\begin{bmatrix} \textbf{verb} & \\ \text{AGREE} & \textbf{agr} \\ \text{VFORM} & \textbf{vform} \end{bmatrix} \qquad \begin{bmatrix} \textbf{noun} & \\ \text{AGREE} & \textbf{agr} \end{bmatrix}$$

$$\begin{bmatrix} \textbf{verb} & \\ \text{AGREE} & \textbf{1st} \\ \text{VFORM} & \textbf{gerund} \end{bmatrix} \sqcup \begin{bmatrix} \textbf{verb} & \\ \text{AGREE} & \textbf{sg} \\ \text{VFORM} & \textbf{gerund} \end{bmatrix} = \begin{bmatrix} \textbf{verb} & \\ \text{AGREE} & \textbf{1-sg} \\ \text{VFORM} & \textbf{gerund} \end{bmatrix}$$

Other Features

- Instead of writing:
  - S → NP VP
    <NP Head Agreement> = <VP Head Agreement>
    <S Head> = <VP Head>
- Could push categories of non-terminals into features (next slide)

Unification Parsing

- Write:
  - X → Y Z
    <X Cat> = S
    <Y Cat> = NP
    <Z Cat> = VP
    <Y Head Agreement> = <Z Head Agreement>
    <X Head> = <Z Head>
- Why bother?

Polymorphism

- 'and' and 'or' polymorphism
  - X → Y 'and' Z
    <Y Cat> = <Z Cat>
    <X Cat> = <Y Cat>
- Requires revision of parser to look at categories rather than non-terminals.

Summary

- Feature structures allow us to capture fine-grained distinctions without multiplying our non-terminals.
- Constraints express restrictions on sentences generated.
  - Provide fixed values for lexical entries
  - Other rules have constraints requiring unification.
- Unification requires sharing.

Semantic Analysis

Why semantics

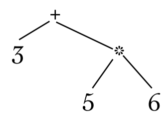
- Information retrieval
  - Some queries simple:
    - What is the current Democratic delegate count?
  - Others are not:
    - What is the city east of Claremont?
- Dialog systems, e.g., as travel agents

Semantic Analysis

- How do we represent meanings of sentences?
  - First-order logic (or extensions)
  - Semantic network (graph)
  - Conceptual Dependency Graph
  - Frame-based representation (fill slots)
- Worry later about how we compute them.

Programming languages

- What is meaning of 3+5*6?
- First parse into tree representing 3 + (5 * 6)
- Then interpret recursively from meanings of nodes



- What about 3 + x * 6?
  - Need to find meaning of x from environment

Representations

- Symbols correspond to objects, properties, and relations among objects
- True or false in some "state of affairs" or model.
- Two points of view:
  - Meanings of statements that may be true or false in model
  - Set of models in which statement is true.
- Ignore context of other statements for now

What do We Need?

- Verifiability
  - Can we answer questions using representation?
- Unambiguous representation
  - Statement may be ambiguous, but better to provide several distinct semantic reps than one ambiguous one.
    - A woman in the U.S. has a baby every 15 minutes.
  - By contrast, vagueness is OK
- Canonical form
  - Identical meanings should have same representation, if possible.

What do We Need?

- Inference
  - Can we deduce conclusions from semantic representation?
- Expressiveness
  - Can we represent everything necessary?
- *Prefer syntax-driven construction of meaning representations – Principal of Compositionality:*
  - *The meaning of a whole is a function of the meanings of the parts and of the way they are syntactically combined.*

Thematic Roles

- Verb subcategorization frames enforce linking of arguments with their semantic roles.
  - Ex: NP$_1$ gave NP$_2$ NP$_3$
  - What are restrictions on each NP?
  - *Giver* vs *Given* vs *Receiver*
  - Consider meaning of gave as three-place relation *gave(x,y,z)* w/ restrictions on x, y, and z.
  - Ex: NP$_1$ was given NP$_2$ by NP$_3$
- Prepositions similar: NP$_1$ on NP$_2$.

Thematic Roles

- Must support relations with
  - Semantic labeling of arguments
  - Semantic constraints on allowable arguments

Meaning

- Meaning will be understood as a relation between
  - Semantic representations (e.g., FOL)
  - State-of-Affairs (SOA) or model providing meaning of representations.

Model

- Model should represent all relevant info about current context
  - Set(s) of objects -- individual elements
  - Names of objects
  - Properties of objects -- unary relations
  - Relations between objects -- n-ary relations

Vocabulary of Representation

- Non-logical vocabulary: Open-ended set of names for objects, properties, and relations of model.
- Logical vocabulary: Closed set of symbols, operators, quantifiers, etc.
  - Allow us to compose more complex expressions from simpler ones.

Model

- Model consists of a mapping of all non-logical symbols to objects, sets and sets of tuples of objects of the model.
- Logical symbols are used to build up meaning of more complex representations in the model.

First-Order Logic

- Non-logical symbols, L, consist of:
  - a finite set of constant names: $c_1, \ldots, c_n$
  - a finite set of function symbols: $f_1^{(n1)}, \ldots, f_m^{(nm)}$ where $f^{(j)}$ is a function symbol of arity j.
  - a finite set of relation symbols: $R_1^{(k1)}, \ldots, R_m^{(km)}$ where $R^{(j)}$ is a j-ary relation symbol.
- Logical symbols:
  - potentially infinite set of variables: $v_1, \ldots, v_j, \ldots$
  - logical connectives: $\neg, \wedge, \vee, \Rightarrow$
  - Quantifiers: $\forall, \exists$

Terms

- A term t is one of:
  - a constant, $t = c_i$.
  - a variable, $t = v_j$.
  - a function application, $t = f_j^{(k)}(t_1, \ldots, t_k)$ where $f_j^{(k)}$ is a k-ary function symbol and $t_1, \ldots, t_k$ are terms.
- Example: $f^{(3)}(g^{(2)}(c, x), y, z)$

Formulas

- A formula ϕ is one of:
  - An atomic formula, $\phi = R_i^{(j)}(t_1, ..., t_j)$, where $R_i^{(j)}$ is a j-ary relation symbol and $t_1, ..., t_j$ are terms.
  - $\phi = \neg\psi$, $\phi = \psi \vee \theta$, $\phi = \psi \wedge \theta$, or $\phi = \psi \Rightarrow \theta$, where ψ and θ are formulas
  - $\phi = \forall x.\theta$ or $\phi = \exists x.\theta$, where x is a variable and θ is a formula.
  - Quantifiers have tighter binding than connectives
  - Can define scope of variable, bound and free variables

FOL Model

- A *model* M for language L is an ordered pair <D,F> consisting of a non-empty domain, D, and an interpretation function F mapping the non-logical symbols of the language L to elements of the model, n-ary functions, or m-ary relations to match the arity of the symbols.
- There are also typed models with a collection of domains D_i.

Assignments

- Cannot determine meaning of term or assignment unless know meaning of its free variables.
- g is an *assignment* in M iff g is a partial function from the set of variables to D.
- If g is an assignment in M, then if d in D, define g[d/x] to be function g' s.t. for all y $\neq$ x, g'(y) = g(y), and g(x) = d.

Meaning of Terms

- Let τ be a term, M = <D,F> be a model, and g be an assignment in M whose domain includes all variables in τ. Define the meaning of τ by induction as follows:
  - $M^g(c) = F(c)$.
  - $M^g(x) = g(x)$.
  - $M^g(f(t_1, ..., t_n)) = F(f)(M^g(t_1), ..., M^g(t_n))$.

Meaning of Formulas

- Let ϕ be a formula, M = <D,F> be a model, and g be an assignment in M whose domain includes all variables in ϕ. Define the satisfaction of ϕ by (M,g) by induction:
  - $(M,g) \models (R(t_1, ..., t_n))$ iff $(M^g(t_1), ..., M^g(t_n)) \in F(R)$.
  - $(M,g) \models \psi \wedge \theta$ iff $(M,g) \models \psi$ and $(M,g) \models \theta$
  - $(M,g) \models \psi \vee \theta$ iff $(M,g) \models \psi$ or $(M,g) \models \theta$
  - $(M,g) \models \neg\psi$ iff not $(M,g) \models \psi$

Meaning of Formulas

- Let ϕ be a term, M = <D,F> be a model, and g be an assignment in M whose domain includes all variables in ϕ. Define the satisfaction of ϕ by (M,g) by induction:
  - $(M,g) \models \exists x.\psi$ iff there is an $a \in D$, s.t. $(M,g[a/x]) \models \psi$.
  - $(M,g) \models \forall x.\psi$ iff for all $a \in D$, $(M,g[a/x]) \models \psi$.

Example

- Model w/ D = set of students in room and the chairs, w/interpretations for
  - names,
  - awake: D -> boolean
  - ifo $\subseteq D \times D$
  - isi $\subseteq D \times D$
- Write sentences and determine which true.

Any Questions?