CS 4705: Semantic Analysis: Syntax-Driven Semantics

Slides adapted from Julia Hirschberg

Announcements

Homework:

Note POS tag corrections. Use POS tags as guide.
You may change them if they hold you back.

Today

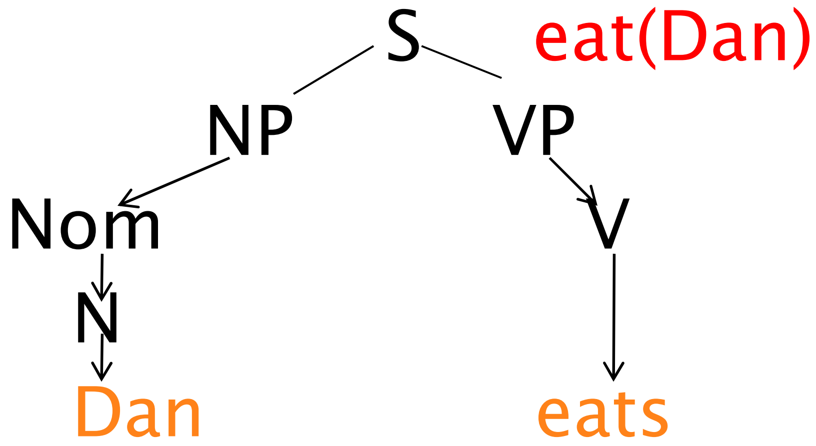
- Reading: Ch 17.2–17.4, 18.1–18.7 (cover material through today); Ch 19.1–19.5 (next time)
- Semantic Analysis: translation from syntax to FOPC
- Hard problems in semantics

Meaning derives from

- The entities and actions/states represented (predicates and arguments, or, nouns and verbs)

- The way they are ordered and related:
  - The syntax of the representation may correspond to the syntax of the sentence
  - Can we develop a mapping between syntactic representations and formal representations of meaning?

Syntax-Driven Semantics



- Goal: Link syntactic structures to corresponding semantic representation to produce representation of the 'meaning' of a sentence while parsing it

Specific vs. General-Purpose Rules

- *Don't* want to have to specify for every possible parse tree what semantic representation it maps to
- *Do* want to identify general mappings from parse trees to semantic representations
- One way:
  - Augment lexicon and grammar
  - Devise mapping between rules of grammar and rules of semantic representation
  - Rule-to-Rule Hypothesis: such a mapping exists

Semantic Attachment

- Extend every grammar rule with \`instructions' on how to map components of rule to a semantic representation, e.g.
  S → NP VP {VP.sem(NP.sem)}
- Each semantic function defined in terms of semantic representation of choice
- Problem: how to define semantic functions and how to specify their *composition* so we always get the \`right' meaning representation from the grammar

Example: McDonalds serves burgers.

- Associating constants with constituents
  - ProperNoun → McDonalds {McDonalds}
  - PluralNoun → burgers {burgers}
- Defining functions to produce these from input
  - NP → ProperNoun {ProperNoun.sem}
  - NP → PluralNoun {PluralNoun.sem}
  - Assumption: meaning representations of children are passed up to parents when non-branching (e.g. ProperNoun.sem(X) = X)
- *But…verbs are where the action is*

- V → serves {Э(e,x,y) (Isa(e,Serving) ^ Agent(e,x) ^ Patient (e,y))} where e = event, x = agent, y = patient
- Will every verb needs its own distinct representation?

  McDonalds hires students.

  - Predicate(Agent, Patient)

  McDonalds gave customers a bonus.

  - Predicate(Agent, Patient, Beneficiary)

Composing Semantic Constituents

- Once we have the semantics for each constituent, how do we combine them?
  - E.g. VP → V NP {V.sem(NP.sem)}
  - If goal for VP semantics of 'serve' is the representation (∃ e,x) (Isa(e,Serving) ^ Agent(e,x) ^ Patient(e,burger)) then
  - VP.sem must tell us
    - Which variables to be replaced by which arguments?
    - How is replacement accomplished?

First… Lambda Notation

- Extension to First Order Predicate Calculus

  λ x P(x): λ + variable(s) + FOPC expression in those variables

- Lambda reduction
  - Apply lambda-expression to logical terms to *bind* lambda-expression's parameters to terms

    λxP(x)

    λxP(x)(car)

    P(car)

For NLP Semantics

- Parameter list (e.g. x in λx) in lambda expression makes variables (x) in logical expression (P(x)) available for binding to external arguments (car) provided by semantics of other constituents
  - P(x): loves(Mary,x)
  - λxP(x)car: loves(Mary,car)

Defining VP Semantics

- Recall we have VP → V NP {V.sem(NP.sem)}
- Target semantic representation is:

  {∃(e,x,y) (Isa(e,Serving) ^ Agent(e,x) ^ Patient(e,y))}
- Define V.sem as:

  {λy ∃(e,x) (Isa(e,Serving) ^ Agent(e,x) ^ Patient(e,y))}
  - Now 'y' will be available for binding when V.sem applied to NP.sem of direct object

V.sem Applied to McDonalds serves burgers

- λ application binds x to value of NP.sem (burgers)

λy Ǝ(e,x) (Isa(e,Serving) ^ Agent(e,x) ^ Patient(e,y)) (burgers)

- λ–reduction replaces y within λ–expression with burgers
- Value of V.sem(NP.sem) is now Ǝ(e,x) (Isa(e,Serving) ^ Agent(e,x) ^ Patient(e,burgers))

But we're not done yet….

- Need to define semantics for
  - S→ NP VP {VP.sem(NP.sem)}
  - Where is the subject?
  - Э(e,x) (Isa(e,Serving) ^ Agent(e,x) ^ Patient(e,burgers))
  - Need another λ-expression in V.sem so the subject NP can be bound later in VP.sem
  - V.sem, version 2
    - λy λx Э(e) (Isa(e,Serving) ^ Agent(e,x) ^ Patient(e,y))

- VP → V NP {V.sem(NP.sem)}

  λy λx ∃(e) (Isa(e,Serving) ^ Agent(e,x) ^ Patient(e,y))(burgers)

  λx ∃(e) (Isa(e,Serving) ^ Agent(e,x) ^ Patient(e,burgers))

- S → NP VP {VP.sem(NP.sem)}

  λx ∃(e) Isa(e,Serving) ^ Agent(e,x) ^ Patient(e,burgers)}(McDonald's)

  ∃(e) Isa(e,Serving) ^ Agent(e,McDonald's) ^ Patient(e,burgers)

What is our grammar now?

S → NP VP {VP.sem(NP.sem)}

VP → V NP {V.sem(NP.sem)}

V → serves {λx λy E(e) (Isa(e,Serving) ^ Agent(e,y) ^ Patient(e,x))}

NP → Propernoun {Propernoun.sem}

NP → Pluralnoun {Pluralnoun.sem}

Propernoun → McDonalds

Pluralnoun → burgers

Parsing with Semantic Attachments

- Modify parser to include operations on semantic attachments as well as syntactic constituents
  - E.g., change an Early-style parser so when constituents are completed, their attached semantic function is applied and a meaning representation created and stored with state
- Or… let parser run to completion and then walk through resulting tree, applying semantic attachments from bottom-up

Option 1 (Integrated Semantic Analysis)

S → NP VP {VP.sem(NP.sem)}

- VP.sem has been stored in state representing VP
- NP.sem stored with the state for NP
- When rule completed, retrieve value of VP.sem and of NP.sem, and apply VP.sem to NP.sem
- Store result in S.sem.

- As fragments of input parsed, semantic fragments created
- Can be used to block ambiguous representations

Example carried through

What about

- John slept.
- John gave Mary the book.
- The door opened
- Any others?

But this is just the tip of the iceberg….

- Terms can be complex

  A restaurant serves burgers.
  - ‘a restaurant’: Э x Isa(x,restaurant)
  - Э e Isa(e,Serving) ^ Agent(e,< Э x Isa(x,restaurant)>) ^ Patient(e,burgers)
  - Allows quantified expressions to appear where terms can by providing rules to turn them into well-formed FOPC expressions
- Issues of quantifier scope

  Every restaurant serves a burger.

How to represent other constituents?

- Adjective phrases:
  - Happy people, cheap food, purple socks
  - Intersective semantics works for some…

Nom → Adj Nom {λx (Nom.sem(x) ^ Isa(x,Adj.sem))}

Adj → cheap {Cheap}

λx Isa(x, Food) ^ Isa(x,Cheap)

But….fake gun? Local restaurant? Former friend? Would-be singer?

Ex Isa(x, Gun) ^ Isa(x,Fake)

Doing Compositional Semantics

- To incorporate semantics into grammar we must
  - Determine `right' representation for each basic constituent
  - Determine `right' representation constituents that take these basic constituents as arguments
  - Incorporate semantic attachments into each rule of our CFG

Drawback

- You also perform semantic analysis on orphaned constituents that play no role in final parse
- Case for pipelined approach: Do semantics *after* syntactic parse

Non-Compositional Language

- Some meaning *isn't* compositional
  - Non-compositional modifiers: fake, former, local, so-called, putative, apparent,…
  - Metaphor:
    - You're the cream in my coffee. She's the cream in George's coffee.
    - The break-in was just the tip of the iceberg. This was only the tip of Shirley's iceberg.
  - Idiom:
    - The old man finally kicked the bucket. The old man finally kicked the proverbial bucket.
  - Deferred reference: The ham sandwich wants his check.
- Solution: special rules? Treat idiom as a unit?

Temporal Representations

- How do we represent time and temporal relationships between events?

  It seems only yesterday that Martha Stewart was in prison but now she has a popular TV show. There is no justice.
- Where do we get temporal information?
  - Verb tense
  - Temporal expressions
  - Sequence of presentation
- Linear representations: Reichenbach '47

- Utterance time (U): when the utterance occurs
- Reference time (R): the temporal point-of-view of the utterance
- Event time (E): when events described in the utterance occur

George is eating a sandwich.

-- E,R,U →

George had eaten a sandwich (when he realized…)

E – R – U →

George will eat a sandwich.

--U,R – E →

While George was eating a sandwich, his mother arrived.

Verbs and Event Types: Aspect

- Statives: states or properties of objects at a particular point in time

  I am hungry.
- Activities: events with no clear endpoint

  I am eating.
- Accomplishments: events with durations and endpoints that result in some change of state

  I ate dinner.
- Achievements: events that change state but have no particular duration – they occur in an instant

  I got the bill.

Beliefs, Desires and Intentions

- Very hard to represent internal speaker states like believing, knowing, wanting, assuming, imagining
  - Not well modeled by a simple DB lookup approach so..
  - Truth in the world vs. truth in some possible world

  George imagined that he could dance.

  George believed that he could dance.
- Augment FOPC with special modal operators that take logical formulae as arguments, e.g. believe, know

Believes(George, dance(George))

Knows(Bill,Believes(George,dance(George)))

- Mutual belief: I believe you believe I believe….
  - Practical importance: modeling belief in dialogue
  - Clark's grounding

Summing Up

- Hypothesis: Principle of Compositionality
  - Semantics of NL sentences and phrases can be composed from the semantics of their subparts
- Rules can be derived which map syntactic analysis to semantic representation (Rule-to-Rule Hypothesis)
  - Lambda notation provides a way to extend FOPC to this end
  - But coming up with rule to rule mappings is hard
- Idioms, metaphors and other non-compositional aspects of language makes things tricky (e.g. fake gun)

Next

- Read Ch 19: 1–5