

Computational Semantics

Deep Processing for NLP
Ling 571
February 10, 2014

Roadmap

- Motivation: Dialog Systems
- Key challenges
- Meaning representation
 - Representational requirements
 - First-order logic
 - Syntax & Semantics
 - Representing compositional meaning

Dialogue Systems

- User: What do I have on Thursday?
- Parse:
 - (S
 - (Q-WH-Obj
 - (Whwd What)
 - (Aux do)
 - (NP (Pron I))
 - (VP/NP (V have)
 - (NP/NP *t*)
 - (PP (Prep on)
 - (NP (N Thursday))))))

Dialogue Systems

- Parser:
 - Yes, it's grammatical!
 - Here's the structure!
- System: Great, but what am I supposed to DO?!
- Need to associate meaning with structure

Dialogue Systems

- (S
- (Q-WH-Obj Action: check; cal: USER; Date:Thursday
- (Whwd What)
- (Aux do)
- (NP (Pron I)) Cal: USER
- (VP/NP (V have)
- (NP/NP *t*)
- (PP (Prep on)
- (NP (N Thursday)))))) Date: Thursday

Natural Language

- Syntax: Determine the structure of natural language input
- Semantics: Determine the meaning of natural language input

Tasks for Semantics

- Semantic interpretation required for many tasks
 - Answering questions
 - Following instructions in a software manual
 - Following a recipe
- Requires more than phonology, morphology, syntax
- Must link linguistic elements to world knowledge

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters.*

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters.*
 - The protests became bloody.

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters.*
 - The protests became bloody.
 - The protests had been peaceful.

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters.*
 - The protests became bloody.
 - The protests had been peaceful.
 - Crowds oppose the government.

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters.*
 - The protests became bloody.
 - The protests had been peaceful.
 - Crowds oppose the government.
 - Some support Mubarak.

Semantics is Complex

- Sentences have many entailments, presuppositions
- *Instead, the protests turned bloody, as anti-government crowds were confronted by what appeared to be a coordinated group of Mubarak supporters.*
 - The protests became bloody.
 - The protests had been peaceful.
 - Crowds oppose the government.
 - Some support Mubarak.
 - There was a confrontation between two groups.
 - Anti-government crowds are not Mubarak supporters.
 - Etc..

Perspectives on Meaning

- Meaning and the mind:
 - Meanings are mental constructs, mediating between language and the world

Perspectives on Meaning

- Meaning and the mind:
 - Meanings are mental constructs, mediating between language and the world
- Meaning and action:
 - Meaning maps language into actions (robotics)
 - Utterances are actions, and activate procedures in hearer

Perspectives on Meaning

- Meaning and the mind:
 - Meanings are mental constructs, mediating between language and the world
- Meaning and action:
 - Meaning maps language into actions (robotics)
 - Utterances are actions, and activate procedures in hearer
- Semantics and models:
 - Meaning maps onto states in model theoretic 'worlds', e.g. Montague
 - Focuses on truth conditions of sentences, and their representation

Challenges in Semantics

- Semantic representation:
 - What is the appropriate formal language to express propositions in linguistic input?

Challenges in Semantics

- Semantic representation:
 - What is the appropriate formal language to express propositions in linguistic input?
 - E.g. predicate calculus
 - $\exists x (\text{dog}(x) \wedge \text{disappear}(x))$

Challenges in Semantics

- Semantic representation:
 - What is the appropriate formal language to express propositions in linguistic input?
 - E.g. predicate calculus
 - $\exists x.(\text{dog}(x) \wedge \text{disappear}(x))$
- Entailment:
 - What are all the valid conclusions that can be drawn from an utterance?

Challenges in Semantics

- Semantic representation:
 - What is the appropriate formal language to express propositions in linguistic input?
 - E.g. predicate calculus
 - $\exists x.(\text{dog}(x) \wedge \text{disappear}(x))$
- Entailment:
 - What are all the valid conclusions that can be drawn from an utterance?
 - 'Lincoln was assassinated' entails

Challenges in Semantics

- Semantic representation:
 - What is the appropriate formal language to express propositions in linguistic input?
 - E.g. predicate calculus
 - $\exists x.(\text{dog}(x) \wedge \text{disappear}(x))$
- Entailment:
 - What are all the valid conclusions that can be drawn from an utterance?
 - 'Lincoln was assassinated' entails 'Lincoln is dead.'

Challenges in Semantics

- Reference: How do linguistic expressions link to objects/concepts in the real world?
 - ‘the dog’ , ‘the President’, ‘the Superbowl’

Challenges in Semantics

- Reference: How do linguistic expressions link to objects/concepts in the real world?
 - ‘the dog’ , ‘the President’, ‘the Superbowl’
- Compositionality: How can we derive the meaning of a unit from its parts?
 - How do syntactic structure and semantic composition relate?
 - ‘rubber duck’ vs ‘rubber chicken’

Challenges in Semantics

- Reference: How do linguistic expressions link to objects/concepts in the real world?
 - ‘the dog’ , ‘the President’, ‘the Superbowl’
- Compositionality: How can we derive the meaning of a unit from its parts?
 - How do syntactic structure and semantic composition relate?
 - ‘rubber duck’ vs ‘rubber chicken’
 - ‘kick the bucket’

More Challenges

- Semantic analysis:
 - How do we derive a representation of the meaning of an utterance?
 - AyCaramba serves meat. ->
 $\exists e \text{ Isa}(e, \text{Serving}) \wedge \text{Server}(e, \text{AyCaramba}) \wedge \text{Served}(e, \text{Meat})$

Tasks in Computational Semantics

- Computational semantics aims to extract, interpret, and reason about the meaning of NL utterances, and includes:
 - Defining a **meaning representation**

Tasks in Computational Semantics

- Computational semantics aims to extract, interpret, and reason about the meaning of NL utterances, and includes:
 - Defining a **meaning representation**
 - Developing techniques for **semantic analysis**, to convert NL strings to meaning representations

Tasks in Computational Semantics

- Computational semantics aims to extract, interpret, and reason about the meaning of NL utterances, and includes:
 - Defining a **meaning representation**
 - Developing techniques for **semantic analysis**, to convert NL strings to meaning representations
 - Developing methods for reasoning about these representations and performing inference from them

Complexity of Computational Semantics

- Requires:

Complexity of Computational Semantics

- Requires:
 - Knowledge of language: words, syntax, relationships b/t structure and meaning, composition procedures

Complexity of Computational Semantics

- Requires:
 - Knowledge of language: words, syntax, relationships b/t structure and meaning, composition procedures
 - Knowledge of the world: what are the objects that we refer to, how do they relate, what are their properties?

Complexity of Computational Semantics

- Requires:
 - Knowledge of language: words, syntax, relationships b/t structure and meaning, composition procedures
 - Knowledge of the world: what are the objects that we refer to, how do they relate, what are their properties?
 - Reasoning: Given a representation and a world, what new conclusions – bits of meaning – can we infer?

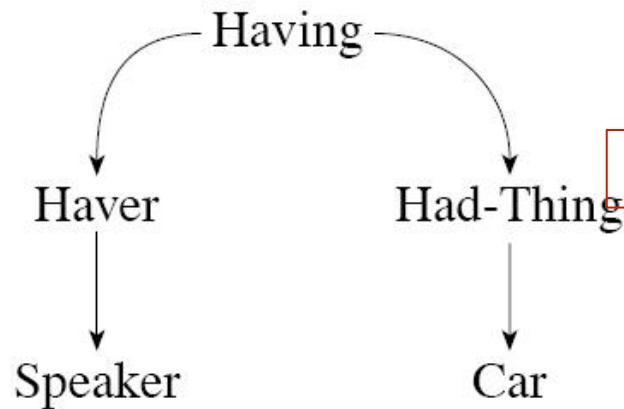
Complexity of Computational Semantics

- Requires:
 - Knowledge of language: words, syntax, relationships b/t structure and meaning, composition procedures
 - Knowledge of the world: what are the objects that we refer to, how do they relate, what are their properties?
 - Reasoning: Given a representation and a world, what new conclusions – bits of meaning – can we infer?
- Effectively AI-complete
 - Need representation, reasoning, world model, etc

Representing Meaning

$\exists e, y \text{ Having}(e) \wedge \text{Haver}(e, \text{Speaker}) \wedge \text{HadThing}(e, y) \wedge \text{Car}(y)$

First-order Logic



Semantic Network

Conceptual
Dependency

Car
↑ POSS-BY
Speaker

Having
Haver: Speaker
HadThing: Car

Frame-Based

Meaning Representations

- All consist of structures from set of symbols
 - Representational vocabulary

Meaning Representations

- All consist of structures from set of symbols
 - Representational vocabulary
- Symbol structures correspond to:
 - Objects
 - Properties of objects
 - Relations among objects

Meaning Representations

- All consist of structures from set of symbols
 - Representational vocabulary
- Symbol structures correspond to:
 - Objects
 - Properties of objects
 - Relations among objects
- Can be viewed as:

Meaning Representations

- All consist of structures from set of symbols
 - Representational vocabulary
- Symbol structures correspond to:
 - Objects
 - Properties of objects
 - Relations among objects
- Can be viewed as:
 - Representation of meaning of linguistic input

Meaning Representations

- All consist of structures from set of symbols
 - Representational vocabulary
- Symbol structures correspond to:
 - Objects
 - Properties of objects
 - Relations among objects
- Can be viewed as:
 - Representation of meaning of linguistic input
 - Representation of state of world
- Here we focus on **literal** meaning

Representational Requirements

- Verifiability
- Unambiguous representations
- Canonical Form
- Inference and Variables
- Expressiveness
 - Should be able to express meaning of any NL sent

Verifiability

- Can a system compare
 - Description of state given by representation to
 - State of some world modeled by a knowledge base (kb)?

Verifiability

- Can a system compare
 - Description of state given by representation to
 - State of some world modeled by a knowledge base (kb)?
- Is the proposition encoded by the representation true?

Verifiability

- Can a system compare
 - Description of state given by representation to
 - State of some world modeled by a knowledge base (kb)?
- Is the proposition encoded by the representation true?
- E.g.
 - Input: Does Maharani serve vegetarian food?
 - Representation: Serves(Maharani,VegetarianFood)
 - KB: Set of assertions about restaurants

Verifiability

- Can a system compare
 - Description of state given by representation to
 - State of some world modeled by a knowledge base (kb)?
- Is the proposition encoded by the representation true?
- E.g.
 - Input: Does Maharani serve vegetarian food?
 - Representation: Serves(Maharani,VegetarianFood)
 - KB: Set of assertions about restaurants
 - If representation matches in KB -> True

Verifiability

- Can a system compare
 - Description of state given by representation to
 - State of some world modeled by a knowledge base (kb)?
- Is the proposition encoded by the representation true?
- E.g.
 - Input: Does Maharani server vegetarian food?
 - Representation: Serves(Maharani,VegetarianFood)
 - KB: Set of assertions about restaurants
 - If representation matches in KB -> True
 - If not, False

Verifiability

- Can a system compare
 - Description of state given by representation to
 - State of some world modeled by a knowledge base (kb)?
- Is the proposition encoded by the representation true?
- E.g.
 - Input: Does Maharani server vegetarian food?
 - Representation: Serves(Maharani,VegetarianFood)
 - KB: Set of assertions about restaurants
 - If representation matches in KB -> True
 - If not, False or Don't Know
 - Is KB assumed complete or incomplete?

Unambiguous Representations

- Semantics is ambiguous:
 - *I wanna eat someplace close to UW*

Unambiguous Representations

- Semantics is ambiguous:
 - *I wanna eat someplace close to UW*
 - Eat *at* someplace OR eat the restaurant
- (Final) Representation must be unambiguous, e.g.,
 - $E_1 = \text{want}(I, E_2)$
 - $E_2 = \text{eat}(I, O_1, \text{Loc}_1)$

Unambiguous Representations

- Semantics is ambiguous:
 - *I wanna eat someplace close to UW*
 - Eat *at* someplace OR eat the restaurant
- (Final) Representation must be unambiguous, e.g.,
 - $E_1 = \text{want}(I, E_2)$
 - $E_2 = \text{eat}(I, O_1, \text{Loc}_1)$
- Resolving the ambiguity?
 - Later

Canonical Form

- Input can have many meanings, and
- Many inputs can have same meaning
 - Flights from Seattle to Chicago

Canonical Form

- Input can have many meanings, and
- Many inputs can have same meaning
 - Flights from Seattle to Chicago
 - Are there any flights from Seattle to Chicago?
 - Do flights go from Seattle to Chicago?
 - Which flights are flown from Seattle to Chicago?
- Could all have different forms

Canonical Form

- Input can have many meanings, and
- Many inputs can have same meaning
 - Flights from Seattle to Chicago
 - Are there any flights from Seattle to Chicago?
 - Do flights go from Seattle to Chicago?
 - Which flights are flown from Seattle to Chicago?
- Could all have different forms
 - Difficult to test in KB

Canonical Form

- Input can have many meanings, and
- Many inputs can have same meaning
 - Flights from Seattle to Chicago
 - Are there any flights from Seattle to Chicago?
 - Do flights go from Seattle to Chicago?
 - Which flights are flown from Seattle to Chicago?
- Could all have different forms
 - Difficult to test in KB
- Single canonical form allows consistent verification

Canonical Form

- Issue:

Canonical Form

- Issue:
 - Pushes ambiguity resolution into semantic analysis
- Different surface forms, but same underlying meaning

Canonical Form

- Issue:
 - Pushes ambiguity resolution into semantic analysis
- Different surface forms, but same underlying meaning
 - Words: E.g, food, fare, dishes
 - Word senses, synonymy
 - Word sense disambiguation

Canonical Form

- Issue:
 - Pushes ambiguity resolution into semantic analysis
- Different surface forms, but same underlying meaning
 - Words: E.g, food, fare, dishes
 - Word senses, synonymy
 - Word sense disambiguation
 - Syntactic alternations:
 - E.g. active vs passive
 - Interrogative vs declarative forms, topicalization, etc

Inference

- Can vegetarians eat at Maharani?
- Does Maharani serve vegetarian food?

Inference

- Can vegetarians eat at Maharani?
- Does Maharani serve vegetarian food?
- Meanings are not identical, but

Inference

- Can vegetarians eat at Maharani?
- Does Maharani serve vegetarian food?
- Meanings are not identical, but
- Linked by facts in the world

Inference

- Can vegetarians eat at Maharani?
- Does Maharani serve vegetarian food?
- Meanings are not identical, but
- Linked by facts in the world
- *Inference* allows system to draw valid conclusions from meaning rep. and KB
 - Serves(Maharani, VegetarianFood) =>
 - CanEat(Vegetarians, AtMaharani)

Variables

- *I want a restaurant that serves vegetarian food.*
- Can we match this in KB?

Variables

- *I want a restaurant that serves vegetarian food.*
- Can we match this in KB?
 - No restaurant specified, so no simple assertion match
- Solution:
 - Variables
 - Serves(x, VegetarianFood)

Variables

- *I want a restaurant that serves vegetarian food.*
- Can we match this in KB?
 - No restaurant specified, so no simple assertion match
- Solution:
 - Variables
 - Serves(x, VegetarianFood)
 - True if variable can be replaced by some object s.t. resulting proposition can match some assertion in KB

Meaning Structure of Language

- Human languages
 - Display basic predicate-argument structure
 - Employ variables
 - Employ quantifiers
 - Exhibit a (partially) compositional semantics

Predicate-Argument Structure

- Represent concepts and relationships
- Words behave like predicates:

Predicate-Argument Structure

- Represent concepts and relationships
- Words behave like predicates:
 - Verbs, Adj, Adv:
 - **Eat**(John,VegetarianFood); **Red**(Ball)
- Some words behave like arguments:

Predicate-Argument Structure

- Represent concepts and relationships
- Words behave like predicates:
 - Verbs, Adj, Adv:
 - **Eat**(John,VegetarianFood); **Red**(Ball)
- Some words behave like arguments:
 - Nouns: Eat(**John,VegetarianFood**); Red(**Ball**)
- Subcategorization frames indicate:

Predicate-Argument Structure

- Represent concepts and relationships
- Words behave like predicates:
 - Verbs, Adj, Adv:
 - **Eat**(John, VegetarianFood); **Red**(Ball)
- Some words behave like arguments:
 - Nouns: Eat(**John**, **VegetarianFood**); Red(**Ball**)
- Subcategorization frames indicate:
 - Number, Syntactic category, order of args

Semantic Roles

- Roles of entities in an event
 - E.g. John_{AGENT} hit Bill_{PATIENT}
- Semantic restrictions constrain entity types
 - The dog slept.
 - ?The rocks slept.
- Verb subcategorization links surface syntactic elements with semantic roles

First-Order Logic

- Meaning representation:
 - Provides sound computational basis for verifiability, inference, expressiveness
- Supports determination of propositional truth
- Supports compositionality of meaning
- Supports inference
- Supports generalization through variables

First-Order Logic

- FOL **terms**:
 - **Constants**: specific objects in world;
 - *A, B, Maharani*
 - Refer to exactly one object; objects referred to by many

First-Order Logic

- FOL **terms**:
 - **Constants**: specific objects in world;
 - *A, B, Maharani*
 - Refer to exactly one object; objects referred to by many
 - **Functions**: concepts refer to objects, e.g. Frasca's loc
 - *LocationOf(Frasca)*
 - Refer to objects, avoid using constants

First-Order Logic

- **FOL terms:**
 - **Constants:** specific objects in world;
 - *A, B, Maharani*
 - Refer to exactly one object; objects referred to by many
 - **Functions:** concepts refer to objects, e.g. Frasca's loc
 - *LocationOf(Frasca)*
 - Refer to objects, avoid using constants
 - **Variables:**
 - *x, e*

FOL Representation

- **Predicates:**
 - Relations among objects
 - *Maharani serves vegetarian food.* →
 - *Serves(Maharani, VegetarianFood)*
 - *Maharani is a restaurant.* →
 - *Restaurant(Maharani)*

FOL Representation

- **Predicates:**

- Relations among objects
 - *Maharani serves vegetarian food.* →
 - *Serves(Maharani, VegetarianFood)*
 - *Maharani is a restaurant.* →
 - *Restaurant(Maharani)*

- **Logical connectives:**

- Allow compositionality of meaning
 - *Maharani serves vegetarian food and is cheap.*

FOL Representation

- **Predicates:**

- Relations among objects
 - *Maharani serves vegetarian food.* →
 - *Serves(Maharani, VegetarianFood)*
 - *Maharani is a restaurant.* →
 - *Restaurant(Maharani)*

- **Logical connectives:**

- Allow compositionality of meaning
 - *Maharani serves vegetarian food and is cheap.*
 - *Serves(Maharani, VegetarianFood) \wedge Cheap(Maharani)*

Variables & Quantifiers

- Variables refer to:

Variables & Quantifiers

- Variables refer to:
 - Anonymous objects

Variables & Quantifiers

- Variables refer to:
 - Anonymous objects
 - All objects in some collection
- Quantifiers:

Variables & Quantifiers

- Variables refer to:
 - Anonymous objects
 - All objects in some collection
- Quantifiers:
 - \exists : existential quantifier: “there exists”
 - Indefinite NP, one such object for truth
 - A cheap restaurant that serves vegetarian food

$$\exists x \text{Restaurant}(x) \wedge \text{Serves}(x, \text{VegetarianFood}) \wedge \text{Cheap}(x)$$

Variables & Quantifiers

- Variables refer to:
 - Anonymous objects
 - All objects in some collection
- Quantifiers:
 - \exists : existential quantifier: “there exists”
 - Indefinite NP, one such object for truth
 - A cheap restaurant that serves vegetarian food
$$\exists x \text{Restaurant}(x) \wedge \text{Serves}(x, \text{VegetarianFood}) \wedge \text{Cheap}(x)$$
 - \forall : universal quantifier: “for all”
 - All vegetarian restaurants serve vegetarian food.
$$\forall x \text{Vegetarian Restaurant}(x) \Rightarrow \text{Serves}(x, \text{VegetarianFood})$$

FOL Syntax Summary

<i>Formula</i>	→	<i>AtomicFormula</i> <i>Formula</i> <i>Connective</i> <i>Formula</i> <i>Quantifier</i> <i>Variable</i> , ... <i>Formula</i> \neg <i>Formula</i> (<i>Formula</i>)
<i>AtomicFormula</i>	→	<i>Predicate</i> (<i>Term</i> , ...)
<i>Term</i>	→	<i>Function</i> (<i>Term</i> , ...) <i>Constant</i> <i>Variable</i>
<i>Connective</i>	→	\wedge \vee \Rightarrow
<i>Quantifier</i>	→	\forall \exists
<i>Constant</i>	→	<i>A</i> <i>VegetarianFood</i> <i>Maharani</i> ...
<i>Variable</i>	→	<i>x</i> <i>y</i> ...
<i>Predicate</i>	→	<i>Serves</i> <i>Near</i> ...
<i>Function</i>	→	<i>LocationOf</i> <i>CuisineOf</i> ...

Compositionality

- **Compositionality:** The meaning of a complex expression is a function of the meaning of its parts and the rules for their combination.
- Formal languages are compositional.
- Natural language meaning is largely, though not fully, compositional, but much more complex.
 - How can we derive things like `loves(John, Mary)` from `John`, `loves(x,y)`, and `Mary`?

Lambda Expressions

- Lambda (λ) notation: (Church, 1940)
 - Just like lambda in Python, Scheme, etc
 - Allows abstraction over FOL formulas
 - Supports compositionality

Lambda Expressions

- Lambda (λ) notation: (Church, 1940)
 - Just like lambda in Python, Scheme, etc
 - Allows abstraction over FOL formulas
 - Supports compositionality
- Form: λ + variable + FOL expression
 - E.g. $\lambda x.P(x)$ “Function taking x to $P(x)$ ”

Lambda Expressions

- Lambda (λ) notation: (Church, 1940)
 - Just like lambda in Python, Scheme, etc
 - Allows abstraction over FOL formulas
 - Supports compositionality
- Form: λ + variable + FOL expression
 - E.g. $\lambda x.P(x)$ “Function taking x to P(x)”
- $\lambda x.P(x) (A) \rightarrow P(A)$

λ -Reduction

- λ -reduction: Apply λ -expression to logical term
 - Binds formal parameter to term

$$\lambda x.P(x)$$

λ -Reduction

- λ -reduction: Apply λ -expression to logical term
 - Binds formal parameter to term

$$\lambda x.P(x)$$

$$\lambda x.P(x)(A)$$

λ -Reduction

- λ -reduction: Apply λ -expression to logical term
 - Binds formal parameter to term

$$\lambda x.P(x)$$

$$\lambda x.P(x)(A)$$

$$P(A)$$

- Equivalent to function application

Nested λ -Reduction

- Lambda expression as body of another

$\lambda x.\lambda y.Near(x,y)$

Nested λ -Reduction

- Lambda expression as body of another

$\lambda x.\lambda y.Near(x,y)$

$\lambda x.\lambda y.Near(x,y)(Bacaro)$

Nested λ -Reduction

- Lambda expression as body of another

$\lambda x. \lambda y. \text{Near}(x, y)$

$\lambda x. \lambda y. \text{Near}(x, y)(\text{Bacaro})$

$\lambda y. \text{Near}(\text{Bacaro}, y)$

Nested λ -Reduction

- Lambda expression as body of another

$\lambda x. \lambda y. \text{Near}(x, y)$

$\lambda x. \lambda y. \text{Near}(x, y)(\text{Bacaro})$

$\lambda y. \text{Near}(\text{Bacaro}, y)$

$\lambda y. \text{Near}(\text{Bacaro}, y)(\text{Centro})$

Nested λ -Reduction

- Lambda expression as body of another

$\lambda x. \lambda y. \text{Near}(x, y)$

$\lambda x. \lambda y. \text{Near}(x, y)(\text{Bacaro})$

$\lambda y. \text{Near}(\text{Bacaro}, y)$

$\lambda y. \text{Near}(\text{Bacaro}, y)(\text{Centro})$

$\text{Near}(\text{Bacaro}, \text{Centro})$

Lambda Expressions

- Currying;
 - Converting multi-argument predicates to sequence of single argument predicates
- Why?

Lambda Expressions

- Currying;
 - Converting multi-argument predicates to sequence of single argument predicates
- Why?
 - Incrementally accumulates multiple arguments spread over different parts of parse tree

Semantics of Meaning Rep.

- Model-theoretic approach:
 - FOL terms (objects): denote elements in a domain
 - Atomic formulas are:
 - If properties, sets of domain elements
 - If relations, sets of tuples of elements
- Formulas based on logical operators:

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$
<i>False</i>	<i>False</i>	<i>True</i>	<i>False</i>	<i>False</i>	<i>True</i>
<i>False</i>	<i>True</i>	<i>True</i>	<i>False</i>	<i>True</i>	<i>True</i>
<i>True</i>	<i>False</i>	<i>False</i>	<i>False</i>	<i>True</i>	<i>False</i>
<i>True</i>	<i>True</i>	<i>False</i>	<i>True</i>	<i>True</i>	<i>True</i>

- Compositionality provided by lambda expressions

Inference

- Standard AI-type logical inference procedures
 - Modus Ponens
 - Forward-chaining, Backward Chaining
 - Abduction
 - Resolution
 - Etc,...
- We'll assume we have a prover

Representing Events

- Initially, single predicate with some arguments
 - Serves(Maharani,IndianFood)

Representing Events

- Initially, single predicate with some arguments
 - Serves(Maharani,IndianFood)
 - Assume # ags = # elements in subcategorization frame

Representing Events

- Initially, single predicate with some arguments
 - Serves(Maharani,IndianFood)
 - Assume # ags = # elements in subcategorization frame
- Example:
 - I ate.
 - I ate a turkey sandwich.
 - I ate a turkey sandwich at my desk.
 - I ate at my desk.
 - I ate lunch.
 - I ate a turkey sandwich for lunch.
 - I ate a turkey sandwich for lunch at my desk.

Events

- Issues?

Events

- Issues?
 - Arity – how can we deal with different #s of arguments?

Events

- Issues?
 - Arity – how can we deal with different #s of arguments?
- One predicate per frame
 - Eating₁(Speaker)
 - Eating₂(Speaker,TS)
 - Eating₃(Speaker,TS,Desk)
 - Eating₄(Speaker,Desk)
 - Eating₅(Speaker,TS,Lunch)
 - Eating₆(Speaker,TS,Lunch,Desk)

Events (Cont'd)

- Good idea?

Events (Cont'd)

- Good idea?
 - Despite the names, actually unrelated predicates

Events (Cont'd)

- Good idea?
 - Despite the names, actually unrelated predicates
 - Can't derive obvious info
 - E.g. I ate a turkey sandwich for lunch at my desk
 - Entails all other sentences

Events (Cont'd)

- Good idea?
 - Despite the names, actually unrelated predicates
 - Can't derive obvious info
 - E.g. I ate a turkey sandwich for lunch at my desk
 - Entails all other sentences
 - Can't directly associate with other predicates

Events (Cont'd)

- Good idea?
 - Despite the names, actually unrelated predicates
 - Can't derive obvious info
 - E.g. I ate a turkey sandwich for lunch at my desk
 - Entails all other sentences
 - Can't directly associate with other predicates
- Could write rules to implement implications

Events (Cont'd)

- Good idea?
 - Despite the names, actually unrelated predicates
 - Can't derive obvious info
 - E.g. I ate a turkey sandwich for lunch at my desk
 - Entails all other sentences
 - Can't directly associate with other predicates
- Could write rules to implement implications
 - But?
 - Intractable in the large
 - Like the subcat problem generally.

Variabilizing

- Create predicate with maximum possible arguments
 - Include appropriate args
 - Maintains connections

$\exists w, x, y \text{Eating}(\text{Speaker}, w, x, y)$

$\exists w, x \text{Eating}(\text{Speaker}, TS, w, x)$

$\exists w \text{Eating}(\text{Speaker}, TS, w, \text{Desk})$

$\text{Eating}(\text{Speaker}, TS, \text{Lunch}, \text{Desk})$

Variabilizing

- Create predicate with maximum possible arguments
 - Include appropriate args
 - Maintains connections

$\exists w, x, y \text{Eating}(\text{Speaker}, w, x, y)$

$\exists w, x \text{Eating}(\text{Speaker}, TS, w, x)$

$\exists w \text{Eating}(\text{Speaker}, TS, w, \text{Desk})$

$\text{Eating}(\text{Speaker}, TS, \text{Lunch}, \text{Desk})$

- Better?

Variabilizing

- Create predicate with maximum possible arguments
 - Include appropriate args
 - Maintains connections
$$\exists w, x, y \text{Eating}(\text{Speaker}, w, x, y)$$
$$\exists w, x \text{Eating}(\text{Speaker}, TS, w, x)$$
$$\exists w \text{Eating}(\text{Speaker}, TS, w, \text{Desk})$$
$$\text{Eating}(\text{Speaker}, TS, \text{Lunch}, \text{Desk})$$
- Better?
 - Yes, but
 - Too many commitments – assume all details show up

Variabilizing

- Create predicate with maximum possible arguments
 - Include appropriate args
 - Maintains connections

$\exists w, x, y \text{Eating}(\text{Speaker}, w, x, y)$

$\exists w, x \text{Eating}(\text{Speaker}, TS, w, x)$

$\exists w \text{Eating}(\text{Speaker}, TS, w, \text{Desk})$

$\text{Eating}(\text{Speaker}, TS, \text{Lunch}, \text{Desk})$

- Better?
 - Yes, but
 - Too many commitments – assume all details show up
 - Can't individuate – don't know if same event

Events - Finalized

- Neo-Davidsonian representation:
 - Distill event to single argument for event itself
 - Everything else is additional predication

$\exists e \text{Eating}(e) \wedge \text{Eater}(e, \text{Speaker}) \wedge \text{Eaten}(e, \text{TS}) \wedge \text{Meal}(e, \text{Lunch}) \wedge \text{Location}(e, \text{Desk})$

Events - Finalized

- Neo-Davidsonian representation:
 - Distill event to single argument for event itself
 - Everything else is additional predication

$\exists e \text{Eating}(e) \wedge \text{Eater}(e, \text{Speaker}) \wedge \text{Eaten}(e, \text{TS}) \wedge \text{Meal}(e, \text{Lunch}) \wedge \text{Location}(e, \text{Desk})$

- Pros:

Events - Finalized

- Neo-Davidsonian representation:
 - Distill event to single argument for event itself
 - Everything else is additional predication

$\exists e \text{Eating}(e) \wedge \text{Eater}(e, \text{Speaker}) \wedge \text{Eaten}(e, \text{TS}) \wedge \text{Meal}(e, \text{Lunch}) \wedge \text{Location}(e, \text{Desk})$

- Pros:
 - No fixed argument structure
 - Dynamically add predicates as necessary

Events - Finalized

- Neo-Davidsonian representation:
 - Distill event to single argument for event itself
 - Everything else is additional predication

$\exists e \text{Eating}(e) \wedge \text{Eater}(e, \text{Speaker}) \wedge \text{Eaten}(e, \text{TS}) \wedge \text{Meal}(e, \text{Lunch}) \wedge \text{Location}(e, \text{Desk})$

- Pros:
 - No fixed argument structure
 - Dynamically add predicates as necessary
 - No extra roles

Events - Finalized

- Neo-Davidsonian representation:
 - Distill event to single argument for event itself
 - Everything else is additional predication

$\exists e \text{Eating}(e) \wedge \text{Eater}(e, \text{Speaker}) \wedge \text{Eaten}(e, \text{TS}) \wedge \text{Meal}(e, \text{Lunch}) \wedge \text{Location}(e, \text{Desk})$

- Pros:
 - No fixed argument structure
 - Dynamically add predicates as necessary
 - No extra roles
 - Logical connections can be derived

Meaning Representation for Computational Semantics

- Requirements:
 - Verifiability, Unambiguous representation, Canonical Form, Inference, Variables, Expressiveness
- Solution:
 - First-Order Logic
 - Structure
 - Semantics
 - Event Representation
- Next: Semantic Analysis
 - Deriving a meaning representation for an input

Summary

- First-order logic can be used as a meaning representation language for natural language
- Principle of compositionality: the meaning of an complex expression is a function of the meaning of its parts
- λ -expressions can be used to compute meaning representations from syntactic trees based on the principle of compositionality
- In the next lecture, we will look at a syntax-driven approach to semantic analysis in more detail