Discourse

BSc Artificial Intelligence, Spring 2011

Raquel Fernández

Institute for Logic, Language & Computation
University of Amsterdam

Course Practicalities

Practical Matters

- Lecturer: Raquel Fernández, ILLC, room C3.132
- Timetable: 2 hours per week
  - Wednesdays 13-15h in G2.13 (period 4)
  - Tuesdays 15-17h in G0.05 (period 5)
- Some weeks we may meet in a Computerpracticum room
- No class in weeks 11 and 12. Last lecture on week 19.
- Website: http://www.illc.uva.nl/~raquel/teaching/discourse
- UvA Blackboard: `http://blackboard.uva.nl`

Objectives and Contents of the Course

This is a course about logic and NLP, focusing on *meaning* or *semantics*

- what techniques can we use to represent natural language meaning and what can we do with the resulting representations?

Main Objectives:

- to understand a number of fundamental techniques behind logic-based computational semantics for processing natural language.
- to develop hands-on experience in automating such processing in Prolog.

Structure:

- Part I: how to represent the meaning of sentences in First Order Logic
- Part II: core of the course, how to deal with discourses (coherent sequences of sentence)
  - we'll introduce Discourse Representation Theory
  - and deal with pronoun resolution and presuppositions
- Part III: time permitting, we'll look into dialogue management

Background & Related Courses

Related courses in the Bachelor AI Programme:

- Taaltheorie en Taalverwerking
- Natuurlijke Taal Interfaces
- Logisch Programmeren en Zoektechnieken
- Computational Logic

Only prerequisite that matters: to have some knowledge of **Prolog**, which you should have if you have taken the course:

- Logisch Programmeren en Zoektechnieken
  Ulle Endriss, Lecture Notes: An Introduction to Prolog Programming

Course Materials

Books on Computational Semantics by Patrick Blackburn & Johan Bos
www.blackburnbos.org

- Volume 1: Representation and Inference for Natural Language,
  A First Course in Computational Semantics
  - not freely available; the background needed from this volume will be covered in class.
- Volume 2: Working with Discourse Representation Theory
  An Advanced Course in Computational Semantics
  - main textbook for the course, freely available online:
    http://homepages.inf.ed.ac.uk/jbos/comsem/book2.html
- The Prolog software for the two volumes can be downloaded from www.blackburnbos.org. You can start by downloading the directory BB1, which we'll use in part I of the course.

Other topics, mostly related to Dialogue, will be covered in class.

Evaluation

- Regular homework exercises (50%)
  distributed and handled in via Blackboard
- Final exam (50%)
  Possibility of doing an individual project instead of exam
- To pass the course, you need at least a 6 in each component

Introduction

AI and Natural Language Semantics

Possible definition of AI: use of computer programs to cast light on intelligent tasks done by humans. AI involves:

- representing knowledge about the world computationally, and
- carrying out tasks with the resulting representations.

A lot of human knowledge is represented through language.
Computational semantics involves:

- using a computer to build meaning representations for NL, and
- reasoning with the result.

Two main approaches in AI and NLP:

- statistical: represent knowledge as probabilities based on observations
- symbolic: represent knowledge using logic and define inference procedures

⇒ ***We'll explore the latter approach for natural language semantics***

Why Logic?

How can we express the meaning of natural language utterances? How can we represent what utterances mean?

'This chair seems comfortable' *'Obama is married'* *'Sara likes pancakes'*

Natural language can be used to talk about the world around us:

- knowing the meaning of an utterance involves knowing how it connects to the world (knowing its *truth-conditions*)

Can we link NL directly to the world, represented e.g. as a database?

- one problem with this direct approach is that language is often ambiguous:

'I saw the Black Sea flying to Tbilisi'
'I saw a Boeing 737 flying to Tbilisi'

We need a formal way of representing the knowledge encapsulated in language $\Rightarrow$ *Logic!*

- it gives us a precise way to represent meaning and allows us to exploit well understood inferential tools.

Model-theoretic Semantics

Formal semanticists (since Montague) have been using logic as an intermediate level of representation to link language to the world. This is the essence of model-theoretic semantics.

- logical formulas are used as representations that encapsulate meaning in a clean and compact way
- logical models ($\approx$ databases) are used to represent world situations
- we can then use the satisfaction relation to link meaning representations to situation representations

Natural Language sentence:	*'A dealer sells a bike'*
FOL meaning representation:	$\exists x \exists y (\textsc{dealer}(x) \wedge \textsc{bike}(y) \wedge \textsc{sell}(x, y))$
Model:	$D = \{d_1, d_2\}$ $F(\textsc{dealer}) = \{d_1\}$ $F(\textsc{bike}) = \{d_2\}$ $F(\textsc{sell}) = \{(d_1, d_2)\}$

Computational Semantics

From the perspective of AI and computational semantics, the questions are:

- How can we automate the process of associating semantic representations with expressions of natural language?
- How can we use logical representations of natural language expressions to automate the process of drawing inferences?

We'll follow the same strategy as formal semanticists, but do it *computationally*. We'll start by using First Order Logic to represent the meaning of sentences

- we'll use Prolog to implement first order formulas, first order models, and the first order satisfaction definition
- for the latter, we'll implement a first order *model checker* in Prolog, which takes as input a formula and a model, and checks whether the formula is satisfied in the model
- this will take care of the so-called Querying Task – related to conventional database querying and important for natural language question-answering systems.

Dealing with Discourse (1)

Recall that we aim at going beyond single sentences and dealing with *discourse* (sequences of natural language sentences)

This requires other inferential tasks besides querying. We are interested in checking whether a discourse is consistent and informative.

Is this discourse consistent?
'Sara likes pancakes. Sara doesn't like pancakes'

Is this discourse informative?
'Obama is married. Obama is married. Obama is married.'

- Consistency Checking Task: given the logical representation of a discourse, is it consistent or inconsistent? we'll use the logical concept of *satisfiability*
- Informativity Checking Task: given the logical representation of a discourse, is it informative or uninformative? we'll use the logical concept of *validity*

Dealing with Discourse (2)

How should we actually represent the meaning of discourses?

It turns out that the meaning of a discourse is often not just the conjunction of the FO representations of the individual sentences:

'Mia is a woman. She loves Vincent.'
$\text{WOMAN}(\text{MIA}) \wedge \text{LOVE}(x, \text{VINCENT})$

But what we want is something like this:
$\text{WOMAN}(\text{MIA}) \wedge \text{LOVE}(\text{MIA}, \text{VINCENT})$

The phenomenon of pronoun resolution will show us that there is more to discourse than truth-conditions $\Rightarrow$ *context change potential*

N.B.: as you will notice, Blackburn & Bos are obsessed with Pulp Fiction...

Dealing with Discourse (3)

Consider the following discourse. Is it consistent? Why?

Jody is not married. ??She loves her husband.

As we shall see, this has to do with the phenomenon of presupposition: the second sentence *presupposes* information that contradicts the first one.

Note that presuppositions are different from entailments:

'Vincent is a bachelor' entails *'Vincent is not married'*, but *'Vincent is not a bachelor'* does not.

'Jody loves her husband' presupposes that *'Jody is married'*, and so does *'Jody does not love her husband'*.

To deal with discourse phenomena such as pronouns and presuppositions in a principled way, we'll introduce Discourse Representation Theory.

What will we get by the end of Part II?

A pretty sophisticated system (Curt = Clever Use of Reasoning Tools)

- that can automatically build semantic representations for sentences and discourses, handling pronouns and presuppositions;
- can handle consistency checking and informativity checking;
- can incorporate background knowledge and answer simple questions;

```
> Mia or Jody dances.
Curt:  OK.
> Who dances?
Curt:  This question makes sense!
Curt:  Maybe Jody.

> Vincent loves every woman?
Curt:  OK.
> Who loves Mia?
Curt:  This question makes sense!
Curt:  Vincent.
> Who is a plant?
Curt:  I have no idea.
```

Quick Review of First Order Logic

First Order Logic: Key Notions

- FOL makes a clear distinction between languages (*syntax*) and models (*semantics*).
- Languages tell us how basic elements may be put together to form formulas, and models tell us how to interpret the formulas.
- We can think of models as mathematical idealisations of situations, and of formulas as descriptions of situations.
- The main question then is: is a description true or false in a given situation?

Vocabularies

A key ingredient of a FO language is its *vocabulary* (or *signature*): the set on non-logical symbols used in the language.

For instance, here is a possible vocabulary:

$\{$ $(\textsc{love}, 2)$,
$(\textsc{customer}, 1)$,
$(\textsc{robber}, 1)$,
$(\textsc{mia}, 0)$,
$(\textsc{vincent}, 0)$,
$(\textsc{honey-bunny}, 0)$,
$(\textsc{yolanda}, 0)$ $\}$

Each symbol is associated with an *arity* $n \geq 0$

- arity 2: binary relations that hold between 2 individuals
- arity 1: unary relations that can hold of single individuals
- arity 0: *constants* representing individuals, not properties or relations.

Vocabularies tell us:

- what the topic we want to talk about is (what kinds of situations we want to model), and
- what kind of expressions we want to use to describe those situations.

First Order Models

Given a vocabulary, what should a model for this vocabulary be?

Recall that intuitively a model is a situation. A model gives us two pieces of information:

- the collection of entities we want to talk about – the *domain* D
- how to interpret the symbols in the vocabulary with respect to the entities in the domain – the *interpretation function* F

A model M is thus an ordered pair (D, F) consisting of a non-empty domain D, and an interpretation function F specifying a semantic value in D for each symbol in the vocabulary.

$D = \{d_1, d_2, d_3, d_4\}$	$D = \{d_1, d_2, d_3, d_4, d_5\}$
$F(\text{MIA}) = d_1$	$F(\text{MIA}) = d_2$
$F(\text{HONEY-BUNNY}) = d_2$	$F(\text{HONEY-BUNNY}) = d_1$
$F(\text{VINCENT}) = d_3$	$F(\text{VINCENT}) = d_4$
$F(\text{YOLANDA}) = d_4$	$F(\text{YOLANDA}) = d_1$
$F(\text{CUSTOMER}) = \{d_1, d_3\}$	$F(\text{CUSTOMER}) = \{d_1, d_2, d_4\}$
$F(\text{ROBBER}) = \{d_2, d_4\}$	$F(\text{ROBBER}) = \{d_3, d_5\}$
$F(\text{LOVE}) = \{(d_3, d_1), (d_4, d_2)\}$	$F(\text{LOVE}) = \{(d_3, d_4)\}$

First Order Languages

Given some vocabulary, we build the FO language over that vocabulary out of the following ingredients:

- all symbols in the vocabulary will be our non-logical symbols
- a set of variables $x, y, z, w, \ldots$
- the boolean connectives: $\neg, \wedge, \vee, \rightarrow$
- the quantifiers $\forall$ (universal) and $\exists$ (existential)

How do we mix these ingredients together?

- Terms: a FO terms τ is any constant or any variable
- Atomic formulas: if R is a relation symbol of arity n and $\tau_1, \ldots, \tau_n$ are terms, then $R(\tau_1, \ldots, \tau_n)$ is an atomic formula.

Intuitively, constants are the analogs of proper names, variables the analogs of pronouns, and atomic formulas the counterpart of simple sentences in natural language.

Well-formed Formulas

Now we can define more complex formulas. The following inductive definition tells us exactly which *well-formed formulas* (wffs) we can form:

- all atomic formulas as wffs;
- if φ and ψ are wffs, so are $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, and $\varphi \rightarrow \psi$;
- if x is a variable and φ is a wff, then $(\forall x)\varphi$ and $(\exists x)\varphi$ are wffs;
- nothing else is a wff.

Free and Bound Variables

Formulas may contain very different types of variables, free and bound:

- any variable in an atomic formula is free;
- if a variable is free in φ or ψ, then it is also free in $\neg\varphi$, $\varphi \wedge \psi$, $\varphi \vee \psi$, and $\varphi \rightarrow \psi$;
- if x is free in φ, then so is it in $(\forall y)\varphi$ and $(\exists y)\varphi$ (for any $y \neq x$);
- no occurrence of x in φ is free in $(\forall x)\varphi$ and $(\exists x)\varphi$;
- any variable that is not free is said to be *bound*.

If a formula contains no occurrences of free variables, then it is called a *sentence* of first order logic (or a *closed* formula).

Non-closed formula with a free variable: $\textsc{robber}(x)$ Sentence: $\forall x \textsc{customer}(x)$

Free variables are similar to pronouns in natural language: pronouns require contextual information to supply a suitable referent. Sentences, on the other hand, are relatively self-contained.

$\Rightarrow$ This distinction is critical when interpreting formulas in models.

Semantics of FOL: Assignments

How do we interpret formulas in models? Given a model of appropriate vocabulary, any sentence over this vocabulary is ether true or false in that model. But how do we handle free variables?

- Assignments: given a model $M = (D, F)$, an assignment in M is a function g from the set of variables to D.

Assignments can be thought of as mathematical models of contexts in natural language.

With assignments in place, we can interpret terms τ as follows (where $\mathcal{I}_F^g(\tau)$ denotes the interpretation of τ with respect to M and g)

- if τ is a constant, $\mathcal{I}_F^g(\tau) = F(\tau)$
- if τ is a variable, $\mathcal{I}_F^g(\tau) = g(\tau)$

Assignment Variants

We need one more idea before we are ready to state the satisfaction definition.

- Assignment variants: Let g and g' be assignments over $\mathcal{D}$ and let x be a variable, Then g' is called an x-variant of g iff $g(y) = g'(y)$ for all variables $y \neq x$.

Assignment variants allow us to try different values for a variable while keeping the values assigned to all other variables the same.

The Satisfaction Definition

The satisfaction relation holds between a formula, a model $D = (D, F)$, and an assignment g of values in D to variables.

[Here 'iff' is shorthand for 'if and only if.' That is, the relationship on the left-hand side holds precisely when the relationship on the right-hand side does too.]

1. $M, g \models R(\tau_1, \ldots, \tau_n)$ iff $(\mathcal{I}_F^g(\tau_1), \ldots, \mathcal{I}_F^g(\tau_n)) \in F(R)$;
2. $M, g \models \neg\varphi$ iff not $M, g \models \varphi$;
3. $M, g \models \varphi \wedge \psi$ iff $M, g \models \varphi$ and $M, g \models \psi$;
4. $M, g \models \varphi \vee \psi$ iff $M, g \models \varphi$ or $M, g \models \psi$;
5. $M, g \models \varphi \rightarrow \psi$ iff not $M, g \models \varphi$ or $M, g \models \psi$;
6. $M, g \models \forall x\varphi$ iff $M, g' \models \varphi$ for all x-variants g' of g; and
7. $M, g \models \exists x\varphi$ iff $M, g' \models \varphi$ for some x-variant g' of g.

Truth: A sentence φ is true in a model M iff for any assignment g, $M, g \models \varphi$. If φ is true in M, we write $M \models \varphi$.

The Querying Task

Satisfaction and truth are critical concepts. In natural language, they refers to the process of evaluating descriptions in situations while making use of contextual information.

The querying task: Given a model M and a first-order formula ϕ, is ϕ satisfied in M or not?

$D = \{d_1, d_2, d_3, d_4, d_5\}$
$F(\text{MIA}) = d_2$
$F(\text{HONEY-BUNNY}) = d_1$
$F(\text{VINCENT}) = d_4$
$F(\text{YOLANDA}) = d_1$
$F(\text{CUSTOMER}) = \{d_1, d_2, d_4\}$
$F(\text{ROBBER}) = \{d_3, d_5\}$
$F(\text{LOVE}) = \{(d_3, d_4)\}$

Is the following sentence true or false in this model?

$$\exists x \text{LOVE}(x, \text{VINCENT})$$

What is Next?

Prolog and FOL

We would like to use FOL as a semantic representation formalism, and we want to deal with it in Prolog.

- The first thing we need is a way of representing in Prolog the basic entities that play a role in FOL: models and formulas.
- We then need to deal computationally with the notion of truth in a model or satisfiability – the querying task

We shall approach all these issues in Prolog by implementing a simple **first-order model checker**.

- it takes a first-order formula and a first-order model as input and checks whether the formula is true in the model.

Relevant code in BB1, downloadable from `www.blackburnbos.org`:

```
exampleModels.pl
modelChecker1.pl
modelChecker2.pl
```