COMPUTATIONAL SEMANTICS: DAY 2

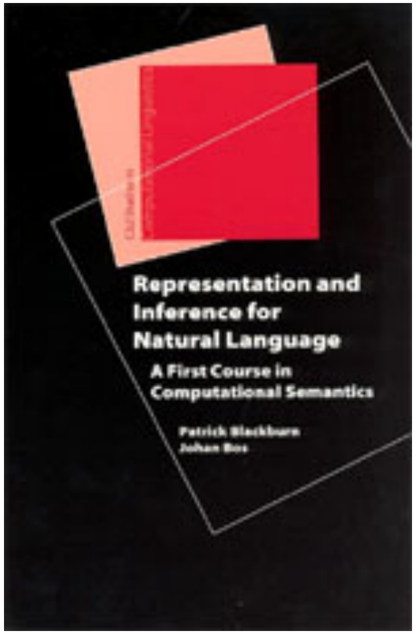
Johan Bos

University of Groningen

www.rug.nl/staff/johan.bos

Computational Semantics

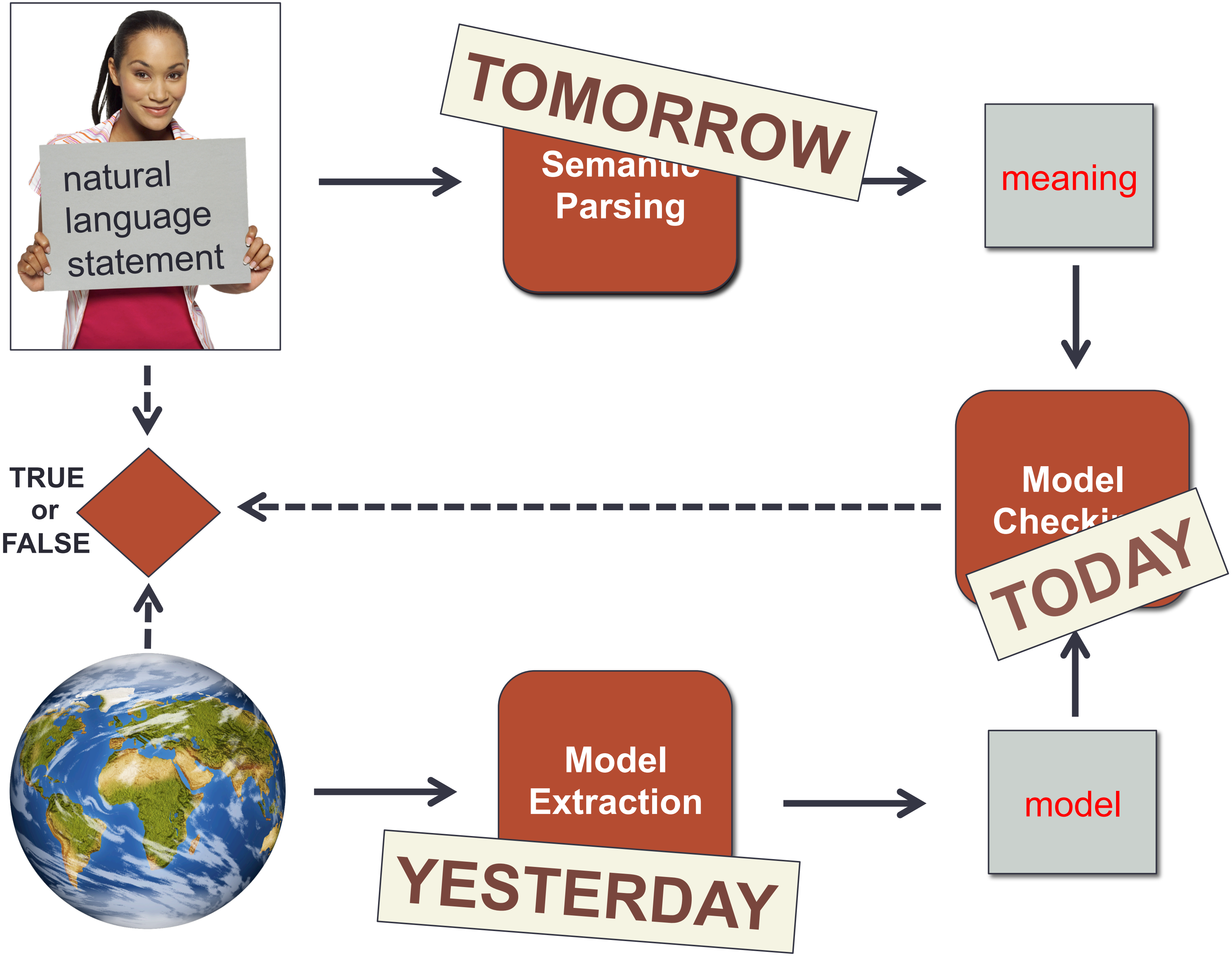
- Day 1: Exploring Models
- **Day 2: Meaning Representations**
- Day 3: Computing Meanings
- Day 4: Drawing Inferences
- Day 5: Meaning Banking



Questions after yesterday’s lecture

- Inferring from observations (“flying bird”)
- (Too?) detailed lexical semantics for verbs
- Small (?) dataset of image models in GRIM
- Adding probabilities
- What are the “zero-place” symbols?

The Big Picture



Constructing basic formulas

- Suppose we're given a model **M** and want to check whether this model satisfies certain descriptions
- For instance, perhaps we want to check whether there is a *cat* present

Constructing basic formulas

- Suppose we're given a model **M** and want to check whether this model satisfies certain descriptions
- For instance, perhaps we want to check whether there is a *cat* present

- We could construct a formula by applying the one-place non-logical symbol CAT to a variable, say x:

CAT(x)

Constructing basic formulas

- Suppose we're given a model **M** and want to check whether this model satisfies certain descriptions
- For instance, perhaps we want to check whether there is a *cat* present

- We could construct a formula by applying the one-place non-logical symbol CAT to a variable, say x:

CAT(x)

- We can now check whether **M** satisfies this formula, but we need the help of an *assignment function*

Variable assignment function

- An assignment function maps all variables in a formula to an entity in the model's domain
- Usually a lowercase letter g is used to denote an assignment function

Variable assignment function

- An assignment function maps all variables in a formula to an entity in the model's domain
- Usually a lowercase letter *g* is used to denote an assignment function
- For instance, for two variables x and y and a domain with three entities d1, d2, and d3, the following assignment functions are possible:
  - $g_1(x)=d1$, $g_1(y)=d1$
  - $g_2(x)=d1$, $g_2(y)=d2$
  - $g_3(x)=d1$, $g_3(y)=d3$
  - $g_4(x)=d2$, $g_4(y)=d1$, etc.

Satisfaction

- Suppose we have this model:

  and assignment: g(x)=d1

  and this formula: CAT(x)

M=<D,F>
D={d1,d2,d3}
F(CAT)={d2}
F(DOG)={d1,d3}
F(TOUCHES)={(d3,d2)}

Does this model satisfy this formula wrt g?

M,g |= CAT(x) ?

Satisfaction

- Suppose we have this model:

M=<D,F>
D={d1,d2,d3}
F(CAT)={d2}
F(DOG)={d1,d3}
F(TOUCHES)={(d3,d2)}

and assignment: g(x)=d1

and this formula: CAT(x)

Does this model satisfy this formula wrt g?

M,g |= CAT(x) ?

Only if g(x) is in F(CAT)

Logic should not be a lottery

- There is clearly a cat in our model, but the answer we get is *false*...
- We only get *true* if we pick out the right value for x
- But we don't want to rely on luck!

Existential quantification

- Logic has a well-known device to avoid being dependent on luck:

$$\exists$$

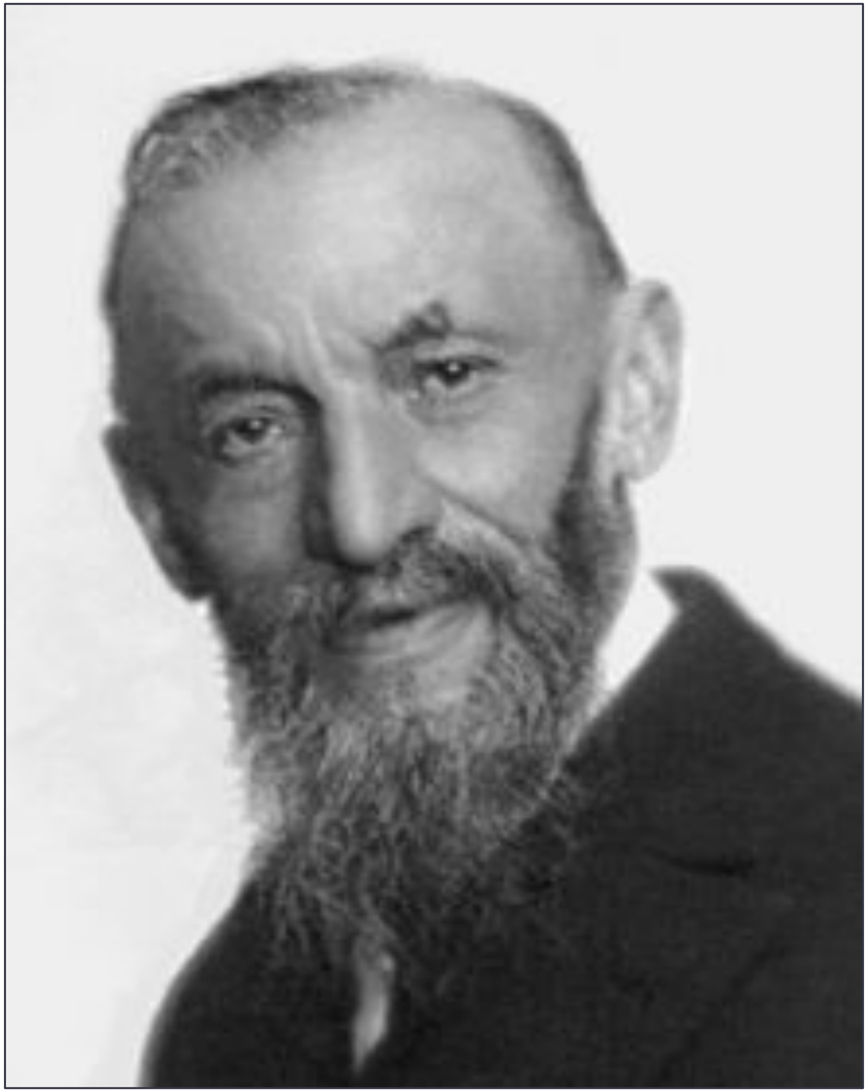
The existential quantifier is always connected to a variable:

If F is a formula, and x a variable,
then $\exists x F$ is a formula.

Giuseppe Peano

Italian mathematician, founder of mathematical logic (1858-1932)

source: en.wikipedia.org



Satisfaction again

- Suppose we have this model:
- And assignment: g(x)=d1
- And this formula: ∃xCAT(x)

M=<D,F>
D={d1,d2,d3}
F(CAT)={d2}
F(DOG)={d1,d3}
F(TOUCHES)={(d3,d2)}

Does this model (let's call it M) satisfy this formula wrt g?

M,g |= ∃xCAT(x) ?

Only if M,g' |= CAT(x), where g' is a copy of g but changes are allowed only with respect to x. E.g.: g'(x)=d2

Constructing complex formulas

- Suppose we're given a model **M** and want to check whether this model satisfies multiple descriptions
- For instance, perhaps we want to check whether there is a *cat* present **and** that it is *white*
- We could construct two basic formulas and form a conjunction (using a new symbol ∧ and brackets):

$$[\mathrm{CAT}(x) \wedge \mathrm{WHITE}(x)]$$

- We can now check whether **M** satisfies this formula.

Satisfaction again

M=<D,F>
D={d1,d2,d3}
F(CAT)={d2}
F(DOG)={d1,d3}
F(WHITE)={d2,d3}

- Suppose we have this model M:
- And assignment: g(x)=d1
- And this formula: [CAT(x)∧WHITE(x)]

Does M satisfy this formula wrt g?

M,g |= [CAT(x)∧WHITE(x)] ?

Only if M,g |= CAT(x) *and* M,g |= WHITE(x)

Constructing negated formulas

- Suppose we're given a model **M** and want to check whether this model does not satisfy a certain description
- For instance, perhaps we want to check that there is *no dog* present, or that there are *no sleeping cats* around
- We can do this by introducing a negation: ¬
  A negated formula is simply formed by putting ¬ in front

$$\neg DOG(x)$$

A model satisfies a negated formula F iff it doesn't satisfy F

$M,g \models \neg DOG(x)$ iff it is not the case that $M,g \models DOG(x)$

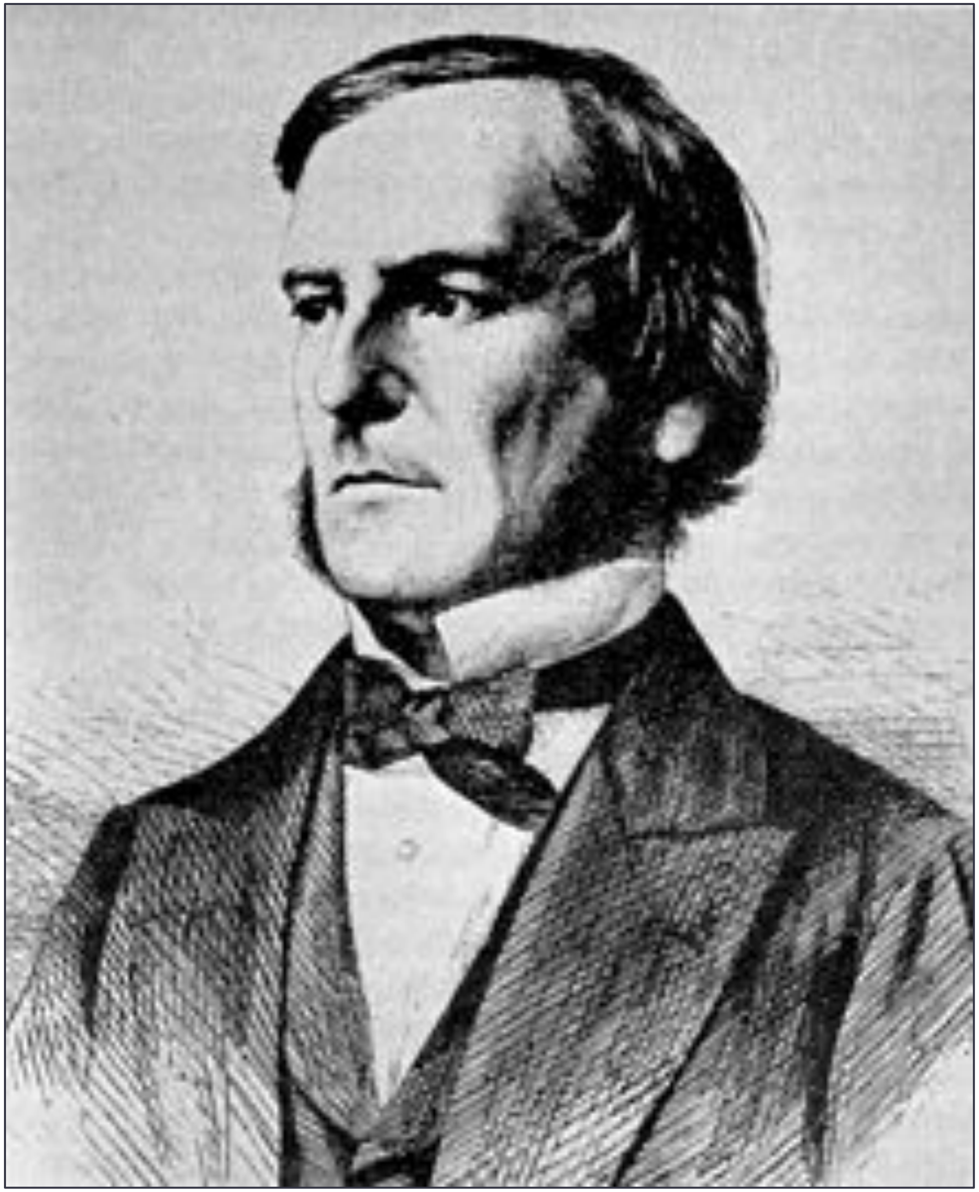
Ingredients of a first-order language

1. All **symbols** in the vocabulary – the non-logical symbols of the first-order language
2. Enough **variables** (a countably infinite collection): x, y, z, etc.
3. The Boolean **connectives** ¬ (negation), ∧ (conjunction), ∨ (disjunction), and → (implication)
4. The **quantifiers** ∀ (the universal quantifier) and ∃ (the existential quantifier)
5. Some **punctuation** symbols: brackets and the comma.

George Boole

English mathematician, pioneer of modern mathematical logic

(1815-1864)



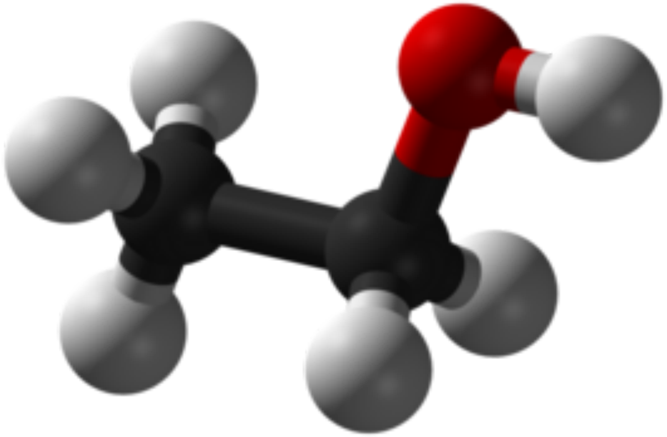
source: en.wikipedia.org

First-order terms

- Any constant or any variable is a first-order term
- Constants are sometimes called 0-place predicates
- Terms are the “noun phrases” of first-order languages
  - constants are first-order analogs of proper names
  - variables are first-order analogs of pronouns

Atomic formulas

- If R is a relation symbol of arity n, and $t_1,\ldots,t_n$ are terms, then $R(t_1,\ldots.,t_n)$ is an atomic formula
- If t_1 and t_2 are terms, then $t_1 = t_2$ is an atomic formula



Well formed formulas (wffs)

1. All atomic formulas are wffs
2. If φ and ψ are wffs,
   then so are $\neg\varphi$, $(\varphi\wedge\psi)$, $(\varphi\vee\psi)$, $(\varphi\rightarrow\psi)$
3. If φ is a wff, and x is a variable,
   then both $\exists x\varphi$ and $\forall x\varphi$ are wffs

4. Nothing else is a wff

Well formed formulas (wffs)

1. All atomic formulas are wffs
2. If φ and ψ are wffs,
   then so are $\neg\varphi$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \rightarrow \psi)$
3. If φ is a wff, and x is a variable,
   then both $\exists x\varphi$ and $\forall x\varphi$ are wffs
   [φ is the *scope* of the quantifier $\exists/\forall$]
4. Nothing else is a wff

Free and Bound Variables

- An occurrence of a variable is free in a formula ψ if it is not bound in ψ
- An occurrence of a variable x is bound in a formula ψ if it appears in the scope of a quantifier $\exists x$ or $\forall x$ in ψ

Closed formulas

- Formulas that have no free variables are called *closed*
- Usually we're only interested in closed formulas
- Translating a natural language sentence to first-order logic should produce a closed formula
- Free variables can be thought of as “pronouns”

Logicians are only human

- Logicians (and mathematicians) are usually very precise in their formulations
- However, they sometimes drop punctuation symbols if no confusion arises
- Often outermost brackets are dropped; also other brackets as long as no confusion arises
- Examples:

$p \wedge q$	instead of	$(p \wedge q)$
$p \vee (q \wedge r)$	instead of	$(p \vee (q \wedge r))$
$(p \vee q \vee r)$	instead of	$(p \vee (q \vee r))$

The satisfaction definition for FOL

$$
\begin{array}{lll}
M, g \models R(\tau_1, \cdots, \tau_n) & \textit{iff} & (I_F^g(\tau_1), \cdots, I_F^g(\tau_n)) \in F(R), \\
M, g \models \tau_1 = \tau_2 & \textit{iff} & I_F^g(\tau_1) = I_F^g(\tau_2), \\
M, g \models \neg\phi & \textit{iff} & \text{not } M, g \models \phi, \\
M, g \models (\phi \wedge \psi) & \textit{iff} & M, g \models \phi \text{ and } M, g \models \psi, \\
M, g \models (\phi \vee \psi) & \textit{iff} & M, g \models \phi \text{ or } M, g \models \psi, \\
M, g \models (\phi \rightarrow \psi) & \textit{iff} & \text{not } M, g \models \phi \text{ or } M, g \models \psi, \\
M, g \models \exists \mathrm{x}\phi & \textit{iff} & M, g' \models \phi, \text{ for some x-variant } g' \text{ of } g, \\
M, g \models \forall \mathrm{x}\phi & \textit{iff} & M, g' \models \phi, \text{ for all x-variants } g' \text{ of } g.
\end{array}
$$

$I_F^g(\tau)$ is F(c) if the term τ is a constant c, and g(x) if τ is a variable x.

Do we really need all this stuff?

- implication
- disjunction
- quantifiers

A note on notation…

- Negation: ¬ or ~
- Conjunction: ∧ or &
- Implication: → or ⊃
- Equivalence: ↔ or ≡
- Brackets: (…) or […]

A note on naming…

First-order logic = predicate logic = classical/standard logic

What's wrong with these translations?

English	First-order logic
A dog barks.	$\exists x(dog(x) \rightarrow bark(x))$
Vincent likes every dog.	$\forall x(dog(x) \wedge like(vincent,x))$
No dog barks.	$\exists x(dog(x) \wedge \neg bark(x))$
Every dog chases a cat.	$\forall x(dog(x) \rightarrow \exists y(cat(y) \wedge chase(y,x))$

What’s wrong with these translations?

English	First-order logic
A dog barks.	$\exists x(dog(x) \wedge bark(x))$
Vincent likes every dog.	$\forall x(dog(x) \rightarrow like(vincent,x))$
No dog barks.	$\neg\exists x(dog(x) \wedge bark(x))$
Every dog chases a cat.	$\forall x(dog(x) \rightarrow \exists y(cat(y) \wedge chase(x,y))$

Model Checking

- The task of the determining whether a given model satisfies a formula (or a set of formulas)

Input: **model + formula**
Output: true or false

Model Checking

$M=\langle D,F\rangle$
$D=\{d1,d2,d3,d4\}$
$F(mia)=d1$
$F(honey\text{-}bunny)=d2$
$F(vincent)=d3$
$F(yolanda)=d4$
$F(customer)=\{d1,d3\}$
$F(robber)=\{d2,d4\}$
$F(love)=\{(d4,d2),(d3,d1)\}$

Q1: Does M satisfy: $\exists x(customer(x) \wedge \exists y(customer(y) \wedge love(x,y)))$

Q2: Does M satisfy: $\exists x(robber(x) \wedge love(x,x))$

Model Checking (“amazing” demo)

1. ~/grim % cat scripts/model_checker.pl | more
2. scripts/_checkmodels "some(X,n_cat_1(X))”
3. scripts/_checkmodels "some(X,n_cat_1(X))” > out.tex
4. pdflatex out

Model Checking (“amazing” demo)

Nice examples (1):

a cat and dog

a cat and a dog

a white cat and a dog

Nice examples (2):

a bicycle

a woman and a bicycle

a woman on a bicycle

Combining Model Extraction & Model Checking

A man threw a bottle in the ocean.

A woman wrote a letter.

Someone dropped two bottles in the sea.

A man with a beard wrote something on a piece of paper.

Combining Model Extraction & Model Checking

A man threw a bottle in the ocean. ✓

A woman wrote a letter. ✗

Someone dropped two bottles in the sea. ✗

A man with a beard wrote something on a piece of paper. ✓

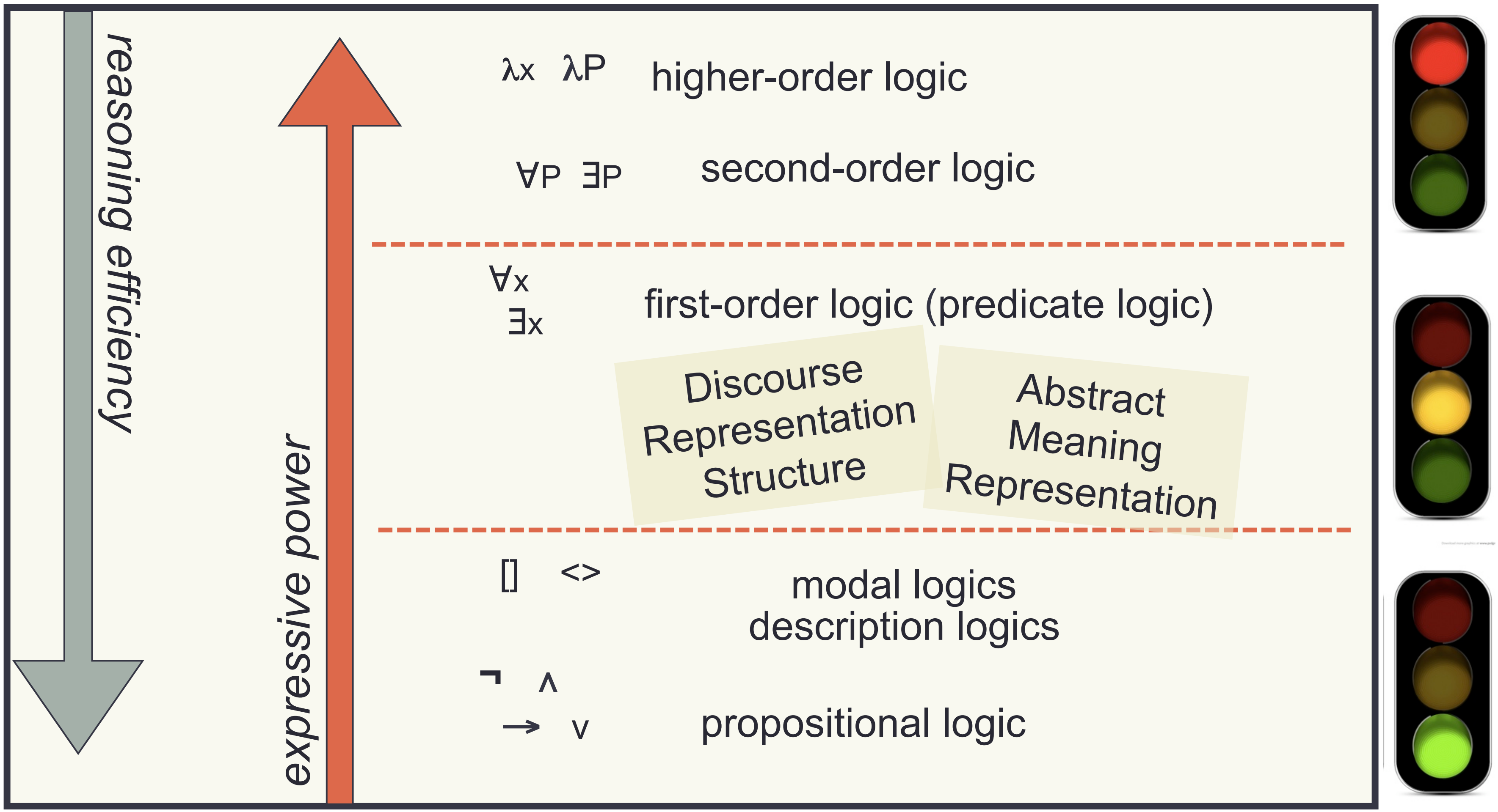
Different kinds of meaning representations

- Expressive power
- FOL, DRS, AMR
- Syntax
- Semantics

What is a good meaning representation?

- All-purpose? Or tailored to specific application?
- Worry about inference efficiency?
- Readability for humans?

Controlling Inference



Syntax of FOL (in Backus-Nauer Form, BNF)

$$
\begin{aligned}
F ::= {} & P_n(t_1 \ldots t_n) \mid \\
& \neg F \mid \\
& (F \wedge F) \mid \\
& \exists x\ F
\end{aligned}
$$

Short version

F is a formula of first-order logic

P_n is a n-place non-logical symbol

t is a term (constant or variable)

Syntax of FOL (in Backus-Nauer Form, BNF)

$F ::= P_n(t_1 \ldots t_n) \mid$
$\quad \neg F \mid$
$\quad (F \wedge F) \mid (F \vee F) \mid (F \rightarrow F) \mid$
$\quad \exists x\, F \mid \forall x\, F$

Long version (with disjunction and universal quantifier)

F is a formula of first-order logic

P_n is a n-place non-logical symbol

t is a term (constant or variable)

Syntax of FOL (in Backus-Nauer Form, BNF)

$$
\begin{aligned}
F ::= {} & P_n(t_1 \ldots t_n) \mid \\
& \neg F \mid \; t{=}t \mid \\
& (F \wedge F) \mid (F \vee F) \mid (F \rightarrow F) \mid \\
& \exists x\, F \mid \forall x\, F
\end{aligned}
$$

Long version with equality

F is a formula of first-order logic

P_n is a n-place non-logical symbol

t is a term (constant or variable)

Back to yesterday's events…

$$
\begin{aligned}
\mathrm{kick}(x,y) \Leftrightarrow\ & \mathrm{abut}(t1,t2)\ \& \\
& \mathrm{abut}(t2,t3)\ \& \\
& \mathrm{near}(t1,x,y)\ \& \\
& \mathrm{touch}(t2,x,y)\ \& \\
& \neg\mathrm{near}(t3,x,y)
\end{aligned}
$$

Pre-Davidsonian

Analysis of an achievement event without explicit entities for events.

Problem: integration of event modifiers.

$$
\begin{aligned}
\text{kick}(e,x,y) \land \text{time}(e,t2) \Leftrightarrow\ & \text{abut}(t1,t2)\ \& \\
& \text{abut}(t2,t3)\ \& \\
& \text{near}(t1,x,y)\ \& \\
& \text{touch}(t2,x,y)\ \& \\
& \neg\text{near}(t3,x,y)
\end{aligned}
$$

Davidsonian

Analysis of an achievement event with explicit entities for events.

Advantage: dealing with event modifiers (manner, temporal)

Disadvantage: number of arguments not always consistent

$$
\begin{aligned}
\text{kick}(e) \wedge \text{agent}(e,x) \wedge \text{patient}(e,y) \wedge \text{time}(e,t2) \Leftarrow\!\!\!\Rightarrow\ & \text{abut}(t1,t2)\ \& \\
& \text{abut}(t2,t3)\ \& \\
& \text{near}(t1,x,y)\ \& \\
& \text{touch}(t2,x,y)\ \& \\
& \neg\text{near}(t3,x,y)
\end{aligned}
$$

neo-Davidsonian

Analysis with explicit entities for events and explicit thematic roles.

Advantage: consistent number of argument for event symbols

Disadvantage: need an inventory of thematic roles

kick(e,x,y,z) ∧ time(e,t2) <=> abut(t1,t2) &
abut(t2,t3) &
near(t1,x,y) &
touch(t2,x,y) &
¬near(t3,x,y)

Hobbsian

Analysis of all event with fixed number (4) of arguments.

Advantage: consistent number of arguments

Disadvantage: need dummy variables

Other meaning representations

- First-order formula syntax not always handy
- Readability (brackets…)
- Dealing with pronouns in texts (rather than sentences)
- Donkey sentences (where the article “a” seems to introduce a universal rather than an existential quantifier

- This lead in the early 1980s do the development of “dynamic” semantic theories such as DRT

Syntax of DRS (in Backus-Nauer Form, BNF)

$B ::= [x_1 \ldots x_n \mid C_1 \ldots C_m]$
$C ::= \neg B \mid$
$\quad B \lor B \mid$
$\quad B \Rightarrow B \mid$
$\quad P_n(x_1, \ldots x_n)$

B is a DRS (Discourse Representation Structure)

C is a DRS-condition

P_n is a n-place predicate symbol

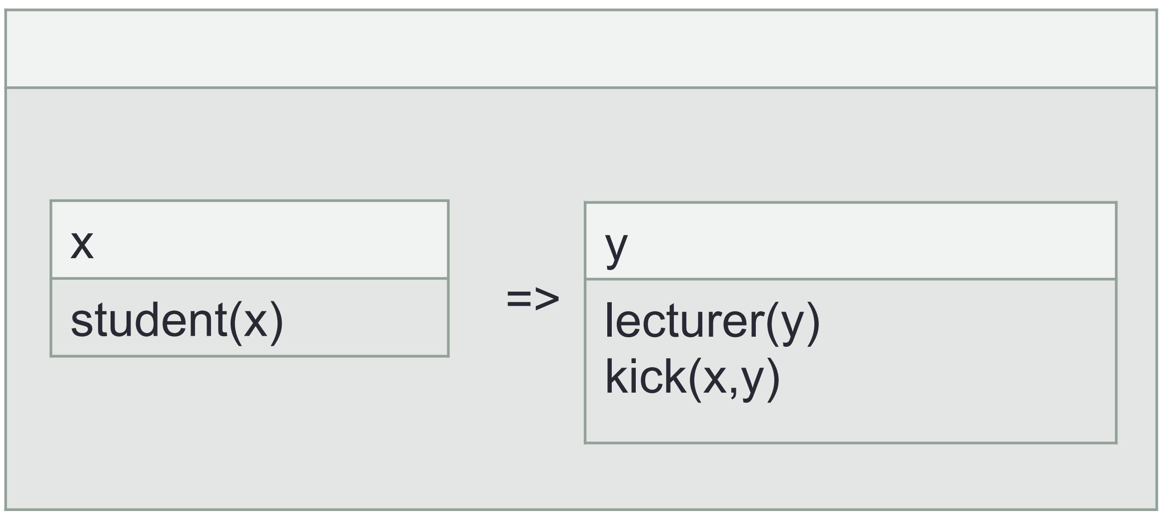
x is discourse referent (variable)

Comparing FOL with DRS syntax

Every student kicked a lecturer.

FOL: $\forall x(student(x) \rightarrow \exists y(lecturer(y) \wedge kick(x,y)))$

DRS: [| [x | student(x)]=>[y | lecturer(y),kick(x,y)]]



Interpretation of DRS

- But wait a minute?
  We don’t have a satisfaction definition for DRSs!

- Two possibilities:
  - Translate DRS into FOL
  - Give a satisfaction definition for DRS

- Both are possible
  (see Kamp & Reyle, Muskens, and many others)

Butch stole a chopper.
It was parked in a garage.

no events

x y u v
Butch(x) chopper(y) stole(x,y) u=y garage(v) parked-in(u,v)

Davidsonian

x y e u v e’
Butch(x) chopper(y) stole(e,x,y) u=y garage(v) parked(e’,u) in(e’,v)

neo-Davidsonian

x y e u v e’
Butch(x) chopper(y) stole(e) agent(e,x) theme(e,y) u=y garage(v) parked(e’) theme(e’,u) location(e’,v)

Hobbsian

x y e u v e’ a b c d
Butch(x) chopper(y) stole(e,x,a,b) agent(e,x) u=y garage(v) parked(e’,u,c,d) in(e’,v)

Abstract Meaning Representations

- Simple meaning representations without explicit scope and quantifiers
- Relatively easy to edit by human beings

Syntax of AMR (in Backus-Nauer Form, BNF)

$$
\begin{aligned}
A ::= \;& c \mid \\
& x \mid \\
& (x \,/\, P_1) \mid \\
& (x/P_1 \; :P_2 A \; \ldots \; :P_2 A)
\end{aligned}
$$

A is an AMR (Abstract Meaning Representation)

P_n is a n-place predicate symbol

x is a variable

c is a constant

"Johan wants money." AMR

```
( e / want
     :ARG0 ( x / johan )
     :ARG1 ( y / money ))
```

Comparing AMR to DRS

“Johan wants money.” DRS

x y e
johan(x) money(y) want(e) Agent(e,x) Theme(e,y)

"Johan wants money." DRS

x y e
want(e) johan(x) money(y) Agent(e,x) Theme(e,y)

“Johan wants money.” DRS

x y e
want(e) money(y) Agent(e,x) johan(x) Theme(e,y)

"Johan wants money." DRS

x y e
want(e) Agent(e,x) johan(x) Theme(e,y) money(y)

"Johan wants money." DRS

x y e
want(e) Agent(e,x) johan(x) Theme(e,y) money(y)

“Johan wants money.” DRS

```
x y e
---------------
want(e)
   :ARG0(e,x)
         johan(x)
   :ARG1(e,y)
         money(y)
```

"Johan wants money." DRS

x y e
(e / want :ARG0(e,x) (x / johan) :ARG1(e,y) (y / money))

“Johan wants money.”

```
(e / want
   :ARG0(e,x)
         (x / johan)
   :ARG1(e,y)
       (y / money))
```

“Johan wants money.” AMR

```
(e / want
   :ARG0 (x / johan)
   :ARG1 (y / money))
```

“Johan wants money.” AMR

```
( e / want
     :ARG0 ( x / johan )
     :ARG1 ( y / money ))
```

JOHAN wants money.

```
( x / johan
    :ARG0-of ( e / want
                   :ARG1 (y / money)))
```

Johan wants MONEY.

```
( y / money
     :ARG1-of ( e / want
                    :ARG0 (x / johan)))
```

Every student kicked a lecturer

```
(x / student :polarity –
   :ARG0-of (e / kick :polarity -
                  :ARG1 (y / lecturer)))
```

An observation about AMR

AMRs without recurring variables are part of the 2-variable fragment of First-Order Logic

It was a picture of a boa constrictor in the act of swallowing an animal.

```
(p / picture
    :domain (i / it)
    :topic (b2 / boa
               :mod (c2 / constrictor)
               :ARG0-of (s / swallow-01
                              :ARG1 (a /
animal))))
```

∃x(picture(x) &
    ∃y(it(y) & domain(x,y)) &
    ∃y(boa(y) & topic(x,y) &
        ∃x(constrictor(x) & mod(y,x)) &
        ∃x(swallow-01(x) & ARG0(x,y) &
            ∃y(animal(y) & ARG1(x,y)))))

Proper names

"Mia is happy."

happy(mia)

Proper names

“Mia is happy.”

happy(mia)

∃x(x=mia ∧ happy(x))

Proper names

"Mia is happy."

$$happy(mia)$$

$$\exists x(x{=}mia \land happy(x))$$

$$\exists x(person(x) \land named(x,mia) \land happy(x))$$

Proper names

"Mia is happy."

$happy(mia)$

$\exists x(x{=}mia \wedge happy(x))$

$\exists x(person(x) \wedge named(x,mia) \wedge happy(x))$

$\exists x \exists y(person(x) \wedge has(x,y) \wedge name(y) \wedge y{=}mia \wedge happy(x))$

Proper names

"Mia is happy."

$happy(mia)$

$\exists x(x{=}mia \wedge happy(x))$

$\exists x(person(x) \wedge named(x,mia) \wedge happy(x))$

$\exists x \exists y(person(x) \wedge has(x,y) \wedge name(y) \wedge y{=}mia \wedge happy(x))$

$\exists x(mia(x) \wedge happy(x))$

FOL, DRS, AMR

- All first-order representations of meaning
- But with different properties
  - Logical aspects (negation, quantification)
  - Human readability
  - Information structure

The Big Picture