Semantiek – end exam

Dr. Yoad Winter and Chris Blom

15 April 2011

Instructions

1. Please fill in your answers on the exam sheets (5 pages).
2. Exam duration: 2.5 hours
3. You may use any pre-prepared material.
4. Please write your student number here: ______________.

Good luck!

Question 1 (5+5+3+7+5=25 points)

Consider the following sentences.

(1.1) John is mayor (of Utrecht).
(1.2) John was mayor (of Utrecht).

Remark: the addition "in Utrecht" does not matter for our analysis, and is only for the sake of clarification.

Obviously, there is no entailment between (1.1) and (1.2). In order to capture this, we treat grammatical tense (*is*/*was*) as indicating *time* in possible world semantics.
To do that, we assume a function time that maps every index $i \in D_s$ to a real number. The times of the indices in D_s introduce an order between them. Thus, if $\mathsf{time}(i_1) < \mathsf{time}(i_2)$ then the time of the index i_1 is earlier than that of the index i_2.
For the words *John*, *mayor is* and *was* in (1.1) and (1.2) we assume the following types and denotations:

John	e	**john**
mayor	$e(st)$	**mayor**
is	$(e(st))(e(st))$	$\textsc{is} = \lambda P_{e(st)}.P$
was	$(e(st))(e(st))$	$\textsc{was} = \lambda P_{e(st)}.\lambda x_e.\lambda i_s.\exists j_s[\mathsf{time}(j) < \mathsf{time}(i) \wedge P(x)(j)]$

a. Simplify as much as possible the following formulas for (1.1) and (1.2), respectively:

$\textsc{is}(\mathbf{mayor})(\mathbf{john})$

____

$\textsc{was}(\mathbf{mayor})(\mathbf{john})$

____

b. Assume a model M where $D_s = \{i_1, i_2\}$, $\mathsf{time}(i_1) = 1$, $\mathsf{time}(i_2) = 2$ and $\mathbf{mayor}(\mathbf{john}) = \{i_1\}$. Write down the denotations of sentences (1.1) and (1.2) in M:

$[\![(1.1)]\!]^M$ = ____

$[\![(1.2)]\!]^M$ = ____

c. Explain briefly how the denotations that you showed in your answer to *b* account for the lack of entailment (in both directions) between sentences (1.1) and (1.2):

____

____

d. Consider now the following sentences.

(1.3) John is former mayor (of Utrecht).
(1.4) John was mayor (of Utrecht) and John is not mayor (of Utrecht).

Assuming that sentence (1.3) is equivalent to (1.4), suggest a type and a meaning for the adjective *former* in (1.3).

type *former*: ____

denotation *former*: FORMER = ____

e. Using your answer to *d*, simplify as much as possible the following formula for (1.3):

$\text{IS}(\text{FORMER}(\mathbf{mayor}))(\mathbf{john})$

____

Make sure that the result is equivalent to our treatment of (1.4).

Question 2 (4+6+5+5+6+6=32 points)

Consider the following sentences, where V_1 and V_2 stand for verbs.

(2.1) All students who V_1 V_2.
(2.2) All students who V_2 V_1.

For instance, when V_1=*dance* and V_2=*smile* we get:
sentence (2.1) = *all students who dance smile*;
sentence (2.2) = *all students who smile dance*.

a. Write down two verbs for V_1 and V_2, for which sentence (2.1) is a tautology, but (2.2) is not.

V_1 = ____________________ V_2 = ____________________

b. Write down two other examples for pairs like V_1 and V_2.

pair 1: ____________________ ____________________

pair 2: ____________________ ____________________

c. Using only the words *students, who, entities, only, are* and the two verbs V_1 and V_2 from your answer to *a*, form a sentence (2.3) that is equivalent to (2.1). Assume that the noun *entities* denotes the function characterizing the whole domain D_e of entities.

(2.3) ____

d. Reconsider the two verbs V_1 and V_2 from your answer to *a*. Using the set denotations $[\![V_1]\!]$ (for the verb V_1), $[\![V_2]\!]$ (for the verb V_2) and S (for the noun *students*), and the set theoretical operations $\cap$ (intersection) and $\subseteq$ (set inclusion), write down the (identical) truth-value of sentences (2.1) and (2.3).

____

e. Reconsider the two verbs V_1 and V_2 from your answer to *a*, as appearing in the following (non-)entailments, where D_1, D_2 and D_3 are determiner expressions.

(2.4) D_1 student(s) who V_1 smiled $\Rightarrow$ D_1 student(s) who V_2 smiled

(2.5) D_2 student(s) who V_2 smiled $\Rightarrow$ D_2 student(s) who V_1 smiled

(2.6) D_3 student(s) who V_1 smiled $\not\Rightarrow$ D_3 student(s) who V_2 smiled;
D_3 student(s) who V_2 smiled $\not\Rightarrow$ D_3 student(s) who V_1 smiled

Write down examples for the determiner expressions in (2.4)-(2.6) that satisfy these non-entailments:

D_1 = ________________ D_2 = ________________ D_3 = ________________

f. Answer the following questions:

Which property of D_1 does entailment (2.4) illustrate?

____

Which property of D_2 does entailment (2.5) illustrate?

____

Which property of D_3 does entailment (2.6) illustrate?

____

Question 3 (3+3+13=19 points)

Consider the following sentences, with the assumed binary structures:

(3.0) John [[showed Mary] Fido].
(3.1) John [[showed [every student]] his dog].

We analyze the verb *show* as being of type $e(e(et))$, denoting a (Curried char. function) of a trinary relation between entities.

a. Write down the (most simplified) truth-value denotation of sentence (3.0), using the denotations $\mathbf{show}$ of type $e(e(et))$ and $\mathbf{john}$, $\mathbf{mary}$ and $\mathbf{fido}$ (all three of type e). You must use the assumed structure in (3.0).

____

For the analysis of (3.1), we define the following Z operator:

$$Z = \lambda R_{e(e(et))}.\lambda Q_{(et)t}.\lambda f_{e^e}.\lambda x_e.Q(\lambda y_e.R(y)(f(y))(x))$$

Using this operator we analyze sentence (3.1) as follows:

(3.2) $Z(\mathbf{showed}_{e(e(et))})(\lambda A_{et}.\mathbf{student}_{et} \subseteq A)(\mathbf{his\_dog}_{e^e})(\mathbf{john}_e)$

b. Consider the following four statements in (i)-(iv).

Formula (3.2) represents the following paraphrase of sentence (3.1) –

(i) There is some masculine entity x, and John showed every student the dog that belongs to x.
(ii) John showed every student the dog that belongs to John.
(iii) For every student x, John showed x the dog that belongs to x.
(iv) No one of the statements above.

Mark the most appropriate statement among (i)-(iv).

c. To support your answer to a, write down the most simplified form of formula (3.2).

____

Remark: You are requested not to write your simplification steps on the exam sheet.

Question 4 (6+6+6+6=24 points)

For this question, refer to the lexicon on page 5. Suppose we have a predefined function `height :: E -> Int` that takes an entity and returns an integer which represents the entities length in centimeters.

1. Use `height` to define a function `taller :: E -> E -> T` that takes two entities and returns `True` if the second has a larger or equal height than the first and `False` otherwise.

```
taller ::  E -> E -> T

taller ________________________________________
```

2. Add a lexicon entry for `taller` such that the following sentences can be parsed with the given lexicon.

   1) `"Everyone is taller than Yoda"`
   2) `"Chewbacca is @ taller than everyone"`
   3) `"No_one is taller than Chewbacca"`
   (`@` is short for the SAT combinator)

```
, entry "taller"   ( ____________________ )      taller
```

3. Give a denotation for the adjective *tall* in terms of taller, such that the following entailments hold for all x of type e and all F of type et:

   1) x is a tall $F \Rightarrow x$ is a F
   2) x is a tall $F \Rightarrow x$ is taller than most F

   You may use any of the functions that are present in the lexicon in your definition.

```
tall_adj :: ____________________________________

tall_adj ________________________________________
```

4. Add a lexicon entry for `taller` such that the following sentences can be parsed with the given lexicon.

   4) `"Vader is_a tall boy"`
   5) `"Leia is_a tall girl"`

```
, entry "tall"   ( ____________________ )      tall_adj
```

```
-- charf takes a set of entities and returns its characteristic function
charf :: (Eq a) => [a] -> a -> T
charf = \set -> \x -> x `elem` set

-- toList take a char. funtion and return the set it characterizes
toList :: (E->T) -> [E]
toList f = filter f (domain f)

{- card : takes a function f and returns the
   cardinality of the set that f characterizes -}
card :: (E -> Bool) -> Int
card f = length (toList f)

sat :: (E->E->T) -> ((E->T)->T) -> (E -> T)
sat r q y = q (\x -> r x y)

{---- Denotations of GQ and DET's--------}
every f g  = f .<. g
some f g   = exists (f /\ g)
most f g   = card (f /\ g) > card (f /\ (compl g))
everyone f = forall f

-- some abbreviations for common syntactic categories
n   = N                -- nouns
s   = S                -- sentences
np  = NP               -- noun phrases
iv  = np :\ s          -- intransitive verbs
tv  = iv :/ np         -- transitive verbs
det = s :/ iv :/ n     -- determiners
gq  = s:/(np:\s)       -- generalized quantifiers

lexicon = Lexicon
  {-=== Entities ===-}
  [ entry "Luke"      np             Luke
  , entry "Leia"      np             Leia
  , entry "Chewbacca" np             Chewbacca
  , entry "Yoda"      np             Yoda
  , entry "Vader"     np             Vader
  , entry "@"         (iv:/gq:/tv)   sat
  , entry "is_a"      ((np:\s):/n)   ( (\x -> x) :: (E->T) -> (E->T))
  , entry "is"        ((np:\s):/n)   ( (\x -> x) :: (E->T) -> (E->T))
  , entry "is"        (iv:/iv)       ( (\x -> x) :: (E->T) -> (E->T))
  , entry "than"      (tv:\tv)       ( (\x -> x) :: (E->E->T) -> (E->E->T) )
  , entry "boy"       n              (charf [Luke,Yoda,Chewbacca,Vader])
  , entry "girl"      n              (charf [Leia])
  , entry "alien"     n              (charf [Chewbacca,Yoda])
  , entry "every"     det            every
  , entry "most"      det            most
  , entry "everyone"  gq             everyone
  , entry "no_one"    gq             (\f -> card f == 0)
  ]
```