600.465 - Intro to NLP
Assignment 5: Semantics

Submitted By: Sindhuula Selvaraju
JHED ID:sselvar4

April 5, 2016

3. (a) $f(John) = loves(Mary, John)$
   i. Written in the form of $\lambda x...$: f = $\lambda x\, loves(Mary, x)$ and in the above case x = John
   ii. Written without λ : f = $loves(Mary)$

   (b) The VP will have a compact semantics because the sem attributes for it will be constants instead of variables.

   (c) i. f = $\lambda y\, woman(x) \Rightarrow loves(x, y)$
   ii. f is a function that gives who loves or in English it can be said f is lover.
   So the entire expression for f will be translated as some person y who loves all women.
   f(John) gives that John is that lover/person who loves. So in this context the English translation will be John loves or lover John.
   So the entire expression for f(John) translates to John loves all women.

   (d) f here is the function that shows the obvious. It can be used in constructing the semantics of "Sue obviously loves Mary" (here x = Sue for the lambda expression):
   For Sue to obviously love Mary, Sue will first have to love mary.
   So first to show Sue loves Mary the $\lambda x loves(Mary, x)\, Sue$ expression will be pushed and then only expression for f will be pushed to show the "Sue obviousy loves Mary".

   (e) The function f takes subject and object of the verb as input. It shows the event for the act of loving and tells who's the lover and who is the lovee.

   (f) g is the function for the manner (in this case passionate) of the love function.

   (g) i. f is the function for all y satisfying the condition(here the condition is that y must be a woman for the condition to be true)

ii. f(λxloves(Mary,y)) = All women love Mary.
(${\lambda}x$loves(Mary,y)) = y loves Mary(the value of y is unknown)
f = All y that satisfies the condition loves Mary

(h) i. g = $\lambda y\, woman(y)$???

ii. woman ???

(i) i. Function f has no use as such so maybe will simply be f = $\lambda x\, g(x)$ where g(x) = loves(Mary,x)

ii. To allow multiple forms of a noun to use the same grammar rule instead of having to specify different rules for proper(John), common(Papa) and mass(Woman) nouns.

4. In most cases for a grammatical sentence the system found the most plausible semantics.

   Yes the system printed the message for ungrammatical sentences. Following is a sentence-wise analysis of correctness of the system:

   1. George love -s Laura .
      This sentence is correct
   2. he love -s her .
      This sentence is correct
   3. him love -s she .
      Though this sentence follows the grammar rules but the sentence is not actually possible.Hence, the attribute values are empty.
   4. Papa sleep -s with a spoon .
      This sentence is correct.
   5. Papa eat -ed with a spoon .
      This sentence is correct.
   6. Papa sleep -s every bonbon with a spoon .
      For this sentence it says that there is no consistent way to assign attributes. Since the sentence does not make any sense this seems right. Maybe parsing this as NP -¿ NP PP instead of VP -¿ VP PP might have given a better result.
   7. Papa eat -ed every bonbon with a spoon .
      This sentence is correct.
   8. have a bonbon !
      The sense of this sentence is actually that the speaker is asking the hearer if they want a bonbon but due to the ambiguity(and the ! in place of ? at the end of the sentence) the sentence is parsed and the system assigns attributes in the sense that the speaker is telling the hearer that they have a bonbon. Adding rules in the grammar to deal with ? should deal with this. So that instead of it showing possession it shows that it's a question.(Though the speaker would have to possess a bonbon in order to offer it.)

9. a bonbon on the spoon entice -s .
   This sentence is correct.

10. a bonbon on the spoon entice -0 .
    For this sentence it says that there is no consistent way to assign attributes. As the sense of the sentence is not right the system's output is correct.

11. the bonbon -s on the spoon entice -0 .
    This sentence is correct.

12. George kiss -ed every chief of staff .
    This sentence is correct.

13. Laura say -s that George might sleep on the floor !
    This sentence means that Laura says that "George might sleep on the floor". However, the parse makes it look like Laura who is on the floor said that George might sleep. Hence, an alternate parse where the sub-sentence "George might sleep on the floor" is parsed and then joined with "Laura says that" would have been a better choice.

14. the perplexed president eat -ed a pickle .
    This sentence is correct.

15. Papa is perplexed .
    This sentence is correct.

16. Papa is chief of staff .
    This sentence is correct.

17. Papa want -ed a sandwich .
    This sentence is correct.

18. Papa want -ed to eat a sandwich .
    This sentence is correct.

19. Papa want -ed George to eat a pickle .
    This sentence is correct.

20. Papa would have eat -ed his sandwich -s .
    This sentence is correct.

21. every sandwich was go -ing to have been delicious .
    This sentence is correct

22. the fine and blue woman and every man must have eat -ed two sandwich -s and sleep -ed on the floor .
    For this sentence the system says that there is no consistent way to assign attributes. This is mainly because the system is unable to decide which verb eat or sleep will come on the head of the sentence. We would need grammar rules to provide higher priority to later rules so that they make it to the head.(Example in this case the man and woman would've eaten and then slept so head would be sleep.)

The following semantics found by the syste seem wrong to me:

(a) have a bonbon ! : This sentence is meant more as a question like do you want to have a bonbon but the system identifies it as a sentence showing possession i.e. to show something like I have a bonbon. The only way for grammar to possibly identify it as a question is by giving a "?" at the sentence end instead of "!". But since the grammar does not have "?" let us ignore this

(b) a bonbon on the spoon entice -s : For this sentence the parse fails to see that it is the bonbon which is on the spoon. It expects another argument to tell type of bonbon. For example in terms of attributes in the top most level: Attributes: head=entice sem=Pres(entice(SOMETHING,some(%x bonbon(x) ∧ on(the(spoon),x)))) This clearly shows that arguments for bonbon i.e. x are still required.

5. For words like caviar:
The following determiners should work : the, some, all, (his, her, their in terms of possessing the caviar can work like "Papa ate his caviar" should accept.)
The following determiners should not work: a, every, each, two
Similarly verbs is and was need to be handled
We do not need separate singular and plural forms. Instead we can just use a generalized form mass.

6. (a) The grammar gives pretty complicates semantic attributes to *two* and plural *the* this is required to understand that more than one noun is associated with the determiner. The lambda terms in the rules corresponding to the Dets *two* and *the* mean :

   i. **1 Det[=1 num=pl sem=%̈dom %pred E%first E%second [first!=second ∧ dom(first)∧ dom(second)] ∧ pred(first) ∧ pred(second)]̈**
   The lambda terms E%first E%second tell that for all the cases where the determiner is *two* there are 2 objects(in this case sandwiches) that are different from each other. The other lambda terms are similar to all other rules and give values for sem and the noun through %dom and %pred.

   ii. **1 Det[=1 num=pl sem=%̈dom %pred E%T [exhaustive(T,dom)] ∧ pred(T)]̈ the**
   The lambda term E%T shows that for the determiner *the* there is a plural exhaustive noun in the %pred. The other lambda terms are similar to all other rules and give values for sem and the noun through %dom and %pred.

   (b) The final result for the sentence is:
   ROOT: Papa want -ed George to eat a pickle .
   Attributes: sem=Assert(Speaker, Past(???(Papa))) head=want

Here the ??? is part of the Past() which gives the object for the head verb. So in this case the ??? refers to the sub-sentence(actually the NP VP combination) ”George to eat a pickle” So the ??? will be replaced by % dom %pred [E%b [pickle(b)] $\wedge$ [eat(b,%pred)]] George