Common Nouns and Adjectives in Predicate Position[1]

(1) **The Lexicon of Our System at Present**

a. *Proper Names:* [[Barack]] = Barack

b. *Intransitive Verbs:*

[[smokes]] = [λx : x $\in$ D_e . **IF** x smokes **THEN** T **ELSE** F]

c. *Transitive Verbs:*

[[likes]] = [λx : x $\in$ D_e . [λy : y $\in$ D_e . **IF** x likes y **THEN** T **ELSE** F]]

d. *Logical Connectives*

[[or_S]] = [λx : x $\in$ D_t . [λy : y $\in$ D_t . **IF** x = T or y = T **THEN** T **ELSE** F]]

[[and_{VP}]] =

[λg $\in$ $D_{<et>}$: [λf $\in$ $D_{<et>}$: [λx $\in$ D_e : **IF** f(x) = T and g(x) = T **THEN** T, **ELSE** F]]]

(2) **Question: Adjectives and 'Common Nouns'?**

But what about adjectives like *male* and so-called 'common nouns' like *politician*? What sort of entries should they receive?

(3) **Our Methodology for Answering Question (2)**

a. Find some (restricted) class of sentences where these lexical items appear.

b. Develop lexical entries for those lexical items that will capture the truth-conditions of those sentences...(*and work out from there to other constructions…*)

(4) **Predicative Uses of Adjectives and Common Nouns**

Following the methodology in (3), the restricted class of sentences we'll work with are those where adjectives and common nouns seem to function as a 'main predicate'.

a. Barack is **male**.
b. Barack is a **politican**.

[1] These notes are based on the material in Heim & Kratzer (1998: 61-63).

Why are we starting here? As we'll see, the semantics we develop for sentences like (4) will be applicable to other places where we find common Ns and As!

(5) **The Questions We Have to Answer**

a. What is the extension of *male*?
b. What is the extension of *politician?*
c. What is the extension of *is*?
d. What is the extension of *a*?

(6) **A Space-Saving Measure for our Lambda Formulas**

As compact as it is, the formulae we're going to write in our lambda notation are only going to get bigger. For that reason, let's use the following abbreviating convention.

Old Notation: [$\lambda x : x \in D_e$. **IF** $\varphi(x)$ **THEN** T **ELSE** F]

New Notation: [$\lambda x : x \in D_e$. $\underline{\varphi(x)}$]

Examples

[$\lambda x : x \in D_e$. **IF** x smokes **THEN** T **ELSE** F] = [$\lambda x : x \in D_e$. $\underline{\text{x smokes}}$]

[$\lambda x : x \in D_e$. [$\lambda y : y \in D_e$. **IF** x likes y **THEN** T **ELSE** F]] = [$\lambda x : x \in D_e$. [$\lambda y : y \in D_e$. $\underline{\text{x likes y}}$]]

[λx_t : [λy_t : **IF** x = T or y = T **THEN** T **ELSE** F]] = [λx_t : [λy_t : $\underline{\text{x = T or y = T}}$]]

1. The Semantics of the English Copula

Let's start off with the sentence in (4a), *Barack is male*.

In order for our system to be able to interpret this sentence, we need a lexical entry for the copula *is* and a lexical entry for the adjective *male*.

(8) **The Semantics of the Copula: The Leading Idea**

For the purposes of our class, let's assume that the copula *is* in English essentially has no real meaning, that it is *semantically vacuous*.

- The idea here is that the copula appears in a sentence for *purely syntactic* reasons (*e.g.* in order to express the tense suffix, which can't go on the noun)

- Since it's only there for syntactic reasons, the copula itself doesn't really contribute to the sentence's meaning…

(9) **Some Motivation for 'The Leading Idea'**

a. It's Tradition!
Grammarians have long considered 'copulas' to be semantically empty.

- In Elementary School, we learn to call *is* a 'helping verb'. (It doesn't really mean anything, it just 'helps' adjectives and nouns to be predicates.)
- The technical term 'copula' comes from a Latin word meaning 'joiner'.

b. Typological
There are many languages that lack copulas, and get along fine without them.
(This suggests that the copula in an English sentence doesn't really add anything to the meaning of the sentence…)

Lillooet (Salish; British Columbia)
emh-ál'qwem' [ti=pelalhtsítcw=a]
good.looking DET=stranger=DET
The stranger ***is good looking***.

(10) **Question:**

How do we represent in our system the idea that the coupla (*is*) is 'semantically vacuous', …that it doesn't contribute anything to the meaning of the sentence?

(11) **Naïve Answer:**

What if we just don't give *is* a lexical entry? Then *is* won't mean anything in our system!

PROBLEM:
If *is* doesn't have *any* lexical entry, then our system won't be able to compute the meaning of sentences that contain *is*.

(12) **Better Answer:**

How about we give *is* a lexical entry, but one that effectively adds *nothing* to the meaning of the larger phrase?

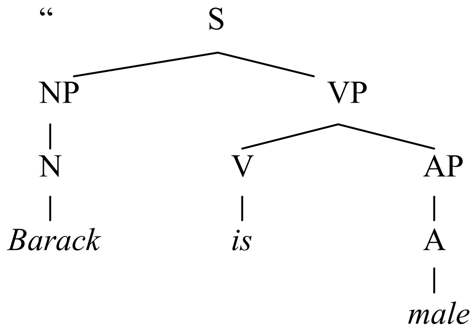
…that is, we treat the meaning of the copula as an identity function!

a. [[is]] = $[\ \lambda f : f \in D_{<e,t>} .\ f\ ]$

The function from <et> functions to <et> functions, which takes an <et> function *f* and simply returns *f*.

2. The Semantics of Adjectives

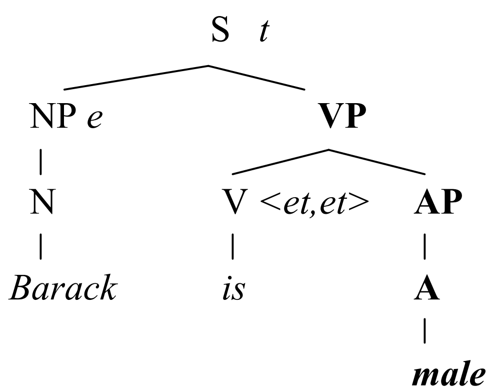
(13) **Our Goal: Derive the Following T-Conditional Statement**

“ 

” is T *iff* Barack is male.

(14) **Let’s Work Out The Types!**

a. Types We Already Know
(i) S *type t*
(ii) NP, N, *Barack* *type e*
(iii) V *type <et,et>* (see (12))



b. Deducing the Types of VP, AP

(i) *The VP must be of type <e,t>*

- The extension of the VP must take the extension of the subject as argument and return the extension of the sentence.

(ii) *The AP must be of type <e,t>*

- Since VP is a branching node, its meaning must be derived via *FA*
- Thus, [[VP]] = [[V]]([[AP]])
- However, [[V]] is of type <et,et>, and so it only takes <et> functions as argument.
- Thus, [[AP]] is of type <et>.

(iii) *The A and 'male' must be of type <et>*

- Since AP is a non-branching node, [[AP]] = [[A]]
- Since A is a non-branching node, [[A]] = [[*male*]]
- Thus, [[AP]], [[A]] and [[*male*]] are all of type <e,t>

But what kind of <e,t> function is the extension of "male"?

(15) **Some Reasoning, Part 1**

The extension of the VP "is male" is identical to the extension of the A "male"

a. Given our rules of FA and NN, and the types deduced above:
[[is male]] = [[is]] ([[male]])

b. Given the lexical entry in (12):
[[is]] ([[male]]) = [λf : f $\in D_{<e,t>}$. f] ([[male]]) = (by LC) [[male]]

(16) **Some Reasoning, Part 2**

The extension of "male" is a function which takes x and returns T iff x is male

a. Given our rules of FA, NN and TN, and the types deduced above:
[[Barack is male]] = [[is male]](Barack)

b. Given our reasoning in (15), it follows that:
[[is male]](Barack) = [[male]](Barack)

c. Thus, we know that the T-conditional statement in (i) below is equivalent to that in (ii) below:

(i) [[Barack is male]] = T *iff* Barack is male.
(ii) [[male]](Barack) = T *iff* Barack is male.

d. **CONCLUSION:** The extension of "male" is a function which takes an entity x as argument, and returns T *iff* x is male.

(17) **The Deduced Lexical Entry for "Male"**

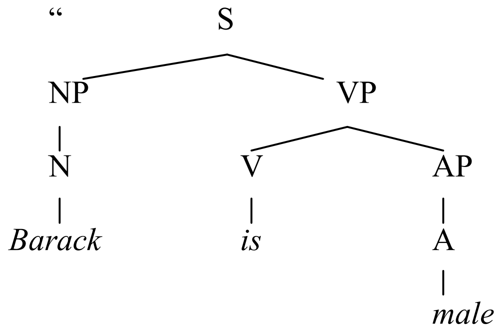
[[male]] =

Old Lambda Notation: [λx : x $\in D_e$. **IF** x is male **THEN** T **ELSE** F]

New, Abbreviated Lambda Notation: [λx : x $\in D_e$. <u>x is male</u>]

Let's now check that our system of lexical entries indeed derives the statement in (13)!

(18) **Truth-Conditional Derivation**

a. " 

" is T *iff* (by notation)

b. [[S]] = T

c. <u>Subproof:</u>
(i) [[NP]] = (by NN x2)
(ii) [[*Barack*]] = (by TN)
(iii) Barack

d. <u>Subproof:</u>
(i) [[AP]] = (by NN x 2)
(ii) [[*male*]] = (by TN)
(iii) [$\lambda x : x \in D_e$. <u>x is male</u>]

e. <u>Subproof:</u>
(i) [[V]] = (by NN)
(ii) [[*is*]] = (by TN)
(iii) [$\lambda f : f \in D_{<e,t>}$. f]

f. <u>Subproof:</u>
(i) [[VP]] = (by FA, d, e)
(ii) [[V]] ([[AP]]) = (by d, e)
(iii) [$\lambda f : f \in D_{<e,t>}$. f] ([$\lambda x : x \in D_e$. <u>x is male</u>]) = (by LC)
(iv) [$\lambda x : x \in D_e$. <u>x is male</u>]

g. [[S]] = T *iff* (by FA, c, f)

h. [[VP]] ([[NP]]) = T *iff* (by c, f)

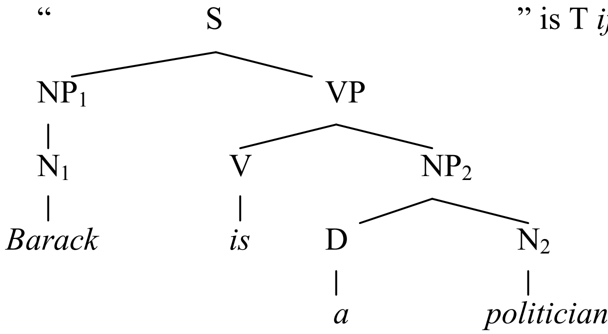
i. [$\lambda x : x \in D_e$. <u>x is male</u>] (Barack) = T *iff* (by LC)

j. Barack is male.

3. The Semantics of Nouns

So, we now have expanded our semantic system so that can interpret sentences like (4a)…
… Now let's turn our attention to sentences like (4b), "Barack is a politician"…

(19) **Our Goal: Derive the Following T-Conditional Statement**

" 

" is T *iff* Barack is a politician.

(20) **Question:**

What kind of meaning should we give to the determiner '*a*' in sentences like (4b)/(19)?

(21) **The Semantics of the Article '*a*': The Leading Idea**

For the purposes of our class, let's assume that the determiner '*a*' in sentences like (4b) and (19) is *semantically vacuous*

(22) **Some Motivation for 'The Leading Idea'**

There seem to be languages where nominal predication doesn't require any determiner.

(This suggests that the determiner in an English sentence like (4a) or (19) doesn't really add anything to the meaning of the sentence…)

<u>Lillooet (Salish; British Columbia)</u>
nk'yap [ti=t'ák=a]
coyote DET=go.along=DET
The one going along is ***a coyote****.*

(23) **Implementation of 'The Leading Idea'**

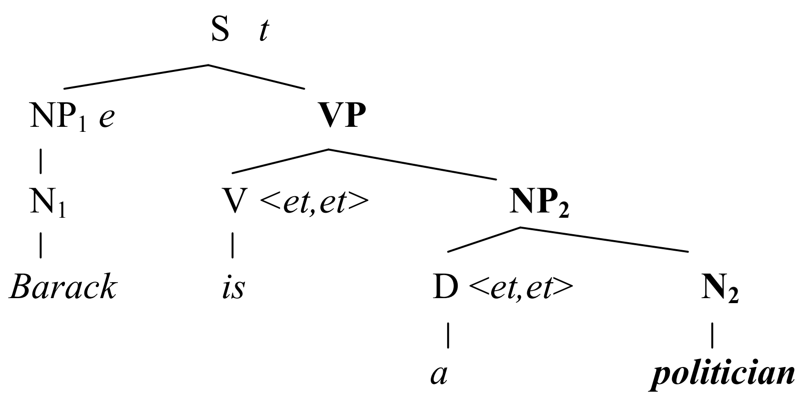
As we did for the copula, we'll assume that the article "a" in sentences like (4b) and (19) simply has the identity function as its extension.

$[[\text{ a }]] = \quad [\ \lambda f : x \in D_{<e,t>} .\ f\ ]$

(24) **Let's Work Out The Types!**

a. <u>Types We Already Know</u>

(i)	S	*type t*	
(ii)	NP_1, N_1, *Barack*	*type e*	
(iii)	V	*type <et,et>*	(see (12))
(iv)	D	*type <et,et>*	(see (23))



b. <u>Deducing the Types of VP, AP</u>

(i) *The VP must be of type <e,t>*

- (See the reasoning in (14b))

(ii) *NP_2 must be of type <e,t>*

- Since VP is a branching node, its meaning must be derived via *FA*
- Thus, [[VP]] = [[V]]([[NP_2]])
- Since [[V]] is of type <et,et>, [[NP_2]] must be of type <et>.

(iii) *N_2 and 'politician' must be of type <e,t>*

- Since NP_2 is a branching node, its meaning must be derived via *FA*
- Thus, [[NP_2]] = [[D]]([[N_2]])
- Since [[D]] is of type <et,et>, [[N_2]] must be of type <et>
- Since [[N_2]] is a non-branching node, [[N_2]] = [[*politician*]].
- Thus, both [[N_2]] and [[politician]] are of type <et>

(25) **Some Reasoning, Part 1**

The extension of VP "is a politician" is identical to the extension of the N "politician"

a. Given our rules of FA and NN, and the types deduced above:

[[is a politician]] = [[is]] ([[a politician]])

b. Given the lexical entry in (12):

[[is]] ([[a politician]]) = [λf : f $\in D_{<e,t>}$. f] ([[a politician]])

c. Given our rule of 'Lambda Conversion':

[λf : f $\in D_{<e,t>}$. f] ([[a politician]]) = [[a politician]]

d. Moreover, given our rules of FA and NN, and the types deduced above:

[[a politician]] = [[a]] ([[politician]])

e. Given the lexical entry in (23):

[[a]] ([[politician]]) = [λf : f $\in D_{<e,t>}$. f] ([[politician]])

g. Given our rule of 'Lambda Conversion':

[λf : f $\in D_{<e,t>}$. f] ([[politician]]) = [[politician]]

d. *Thus, taking putting all the steps above together:*

[[is a politician]] = [[politician]]

(26) **Some Reasoning, Part 2**

a. Given our rule sof FA, NN and TN, and the types deduced above:

[[Barack is a politician]] = [[is a politician]](Barack)

b. Given the equivalence above, along with our reasoning in (25), it follows that the following two T-conditional statements are equivalent:

(i) [[Barack is a politician]] = T *iff* Barack is a politician
(ii) [[politician]](Barack) = T *iff* Barack is a politician

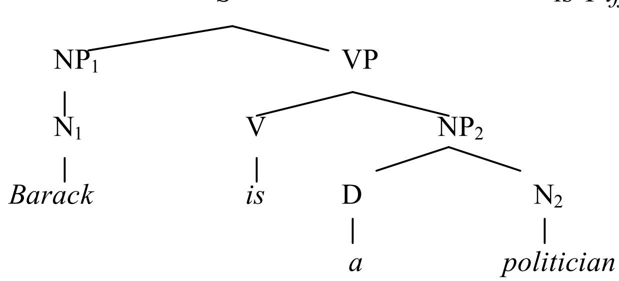
c. **CONCLUSION:** The extension of "politician" is a function which takes an entity x as argument, and returns T *iff* x is a politician.

(27) The Deduced Lexical Entry for "politician"

[[politician]] = [$\lambda x : x \in D_e$. x is a politician]

(28) Truth-Conditional Derivation

a. " S " is T *iff* (by notation)



b. [[S]] = T

c. Subproof:
- (i) [[NP_1]] = (by NN x2)
- (ii) [[*Barack*]] = (by TN)
- (iii) Barack

d. Subproof:
- (i) [[N_2]] = (by NN)
- (ii) [[*politician*]] = (by TN)
- (iii) [$\lambda x : x \in D_e$. x is a politician]

e. Subproof:
- (i) [[D]] = (by NN)
- (ii) [[*a*]] = (by TN)
- (iii) [$\lambda f : f \in D_{<e,t>}$. f]

f. Subproof:
- (i) [[NP_2]] = (by FA, d, e)
- (ii) [[D]] ([[N_2]]) = (by d, e)
- (iii) [$\lambda f : f \in D_{<e,t>}$. f] ([$\lambda x : x \in D_e$. x is a politician]) = (by LC)
- (iv) [$\lambda x : x \in D_e$. x is a politician]

g. Subproof:
- (i) [[V]] = (by NN)
- (ii) [[*is*]] = (by TN)
- (iii) [$\lambda f : f \in D_{<e,t>}$. f]

h. Subproof:
- (i) [[VP]] = (by FA, f, g)
- (ii) [[V]] ([[NP_2]]) = (by f, g)
- (iii) [$\lambda f : f \in D_{<e,t>}$. f] ([$\lambda x : x \in D_e$. x is a politician]) = (by LC)
- (iv) [$\lambda x : x \in D_e$. x is a politician]

… continued below…

i. [[S]] = T *iff* (by FA, c, h)

j. [[VP]] ([[NP_1]]) = T *iff* (by c, h)

k. [λx : x ∈ D_e . <u>x is a politician</u>] (Barack) = T *iff* (by LC)

l. Barack is a politician.

(29) **Summary**

Our system is now able to interpret sentences where:

a. An adjective occupies predicate position. *Barack is* ***male.***
b. A 'common noun' occupies predicate position. *Barack is a* ***politician.***

But another common use of adjectives is as <u>modifiers</u> of nouns, in structures like the following:

c. Barack is a **male politician.**

How do we expand our system so that it is able to interpret these modificational structures?...