24.903
Language & Structure III: Semantics and Pragmatics
Spring 2003, 2-151, MW 1-2.30
February 26, 2003
Assignment 3, due in class on March 5

---

1. Let A be an expression of type e, B of type et, C of type (et)et, and D of type (et)t. Determine whether the following expressions are well-formed, and if yes, give the type of those expressions:

a. A(B)
No.
b. C(B(A))
No.
c. C(B)
et
d. C(B)(A)
t
e. D(B)
t
f. D(B)(A)
No.
g. D(C(B))
t
h. C(C(B))(A)
t

2. Give the types of the subexpressions marked by "?", assuming that the examples are well-formed.
a. A(B), A: (e(et))t, B: ? A(B): ?
B: eet (= $e(et)$), A(B): t
b. A(B), A(B): (et)et, B: et, A: ?
A: $(et)(et)et$
c. A(B)(C), A: ((et)t)(et)et, B: ?, C: ?, A(B)(C): ?
B: $(et)t$, C: et, A(B)(C): et
d. A(B)(C), A(B)(C): et, A(B): eet, B: (et)t, A:? C:?
A: $((et)t)eet$, C: e
e. A(B(C)) A(B(C)): e B: (et)t A: ? C: ?
A: te, C: et
f. Is it possible to assign types to A, B and C in such a way that both A(B(C)) and A(B)(C) are well-formed expressions?
No. If A(B(C)) is well-formed, the type of A's first argument is the same as the type of B(C). If A(B)(C) is well-formed, then the type of A's first argument must be the same as the type of B. We know that the type of B must be bigger than the type of B(C). The type of the first argument of A can be either the type of B(C) or the type of B, but not both. Hence no types can be assigned.

(3a) What is the type of **and** as a conjunction of intransitive verbs, as in
John [sleeps and snores]?
Assuming binary branching and schönfinkelization, $(et)(et)et$.
If you assumed VP-internal subjects, a type of *ttt* is also tenable.
b) What is the type of **and** as a conjunction of transitive verbs, as in
John [read and enjoyed] "Ulysses"?
Assuming binary branching and schönfinkelization, $(eet)(eet)eet$
If you assumed VP-internal subjects, a type of *ttt* is tenable here too. However, note that the derivation of the surface word order assuming a type of *ttt* involves not just getting the subject out of the way, but also the object. Given current syntactic assumptions, such a structure ([John read *Ullyses*] and [*John* enjoyed

Ullyses]) is not plausible.
c) What is the type of the adverb **quickly** in a sentence like
John [walked quickly]?
$(et)et$
d) What is the type of the preposition **with** in the sentence
John [walked [with [Mary]]]?
Since *walked* is of type et, and so is *walked with Mary*, the type of *with Mary* has to be $(et)et$. The type of *Mary* is e, so the type of *with* must be $e(et)et$.
e) What are the types of **faster** and **than** in the sentence
John [walks [faster [than Mary]]]?
Several type-assignments are possible here:
1. Assume a very simple type for *than*(= ee) and assign *faster* a more complex type (= $e(et)et$).
2. Assume a complex type for *than* (= $e(et)(et)et$) and assign *faster* a simpler type (= et).
I prefer option 1 for reasons pertaining to a close parallel between adjectives and adverbs. A word that in adjectival guise has type et, typically has type $(et)et$ in its adverbial guise. If we assume *faster*, the comparative form of *fast*, to be of type eet, a type of $e(et)et$ when it behaves as an adverbial does not seem unusual. In addition, this type assignment allows us to assign *than* a uniform type in *John walks faster than Mary* and *John is faster than Mary*.
A third option, very closely related to the second option, is also possible. Assume a complex type for *than* (= $e(eet)(et)et$) and assign *faster* a simpler type (= eet). This way we can keep the type of *faster* uniform across *John walks faster than Mary* and *John is faster than Mary*. My argument against option 2 applies here also, perhaps even more trenchantly. Since it seems clear that *faster* is playing different roles in *John walks faster than Mary* and *John is faster than Mary*, isn't keeping its type constant exactly the **wrong** thing to do?

4. There are different conventions for naming types.
a. Translate the following three abbreviated type names to type names using the official convention of angled brackets:
eeet, $< e, < e, < e, t >>>$
(et)et, $<< e, t >, < e, t >>$
((et)et)(et)et, $<<< e, t >, < e, t >>, << e, t >, < e, t >>>$
b. Translate the following three official type names to type names using the abbreviatory convention:
$<< e, t >, < e, < e, t >>>$, $(et)eet$
$<<< e, t >, < e, t >>, < e, < e, < e, t >>>>$, $((et)et)eeet$
$<<<< e, t >, t >, << e, t >, t >>, < e, t >>$, $(((et)t)(et)t)et$