Word order and thematic roles

Edgar Onea
University of Gttingen

30.October.2013

Consider three types of languages. In Type A languages (English), word order is responsible for the assignment of θ-roles and plays a reduced role in the assignment of case. For type B languages (Hungarian), case is driving the assignment of θ-roles, and word order is responsible for quantifier scope and discourse configuration. In type C languages (Italian) word order plays some role in in the assignment of θ-role, but is even more influences scope or information structure. It seems that λ-calculus is particularly useful for type A languages, but we need lots of type-shifting or movement to deal with B and C. I will propose a version of λ-calculus, flexible enough for type B and C languages as well.

It is known that one can modify λ-calculus to allow simultaneous abstraction. Best worked out is Ruhrberg (1996, PhD Edinburgh). However, he doesn't have α-equivalence, hence loosing an essential feature of the λ-calculus. I will propose, instead, a conservative extension of the λ-calculus. This leads to new technical possibilities without the need to give up anything practiced before.

As a brief example, assume a system in which individuals are flagged with indices. Each λ binds an unordered set of flagged variables. (hence λxy is no longer a notational shorthand for $\lambda x.\lambda y.$) Beta reduction is driven by flags. Order of application can be specified, by using several λ binders. (1) illustrates.

(1) a. $\lambda x^1 y^2.loves(x,y)(a^1) = \lambda y^2.loves(a,y)$ $\lambda x^1 y^2.loves(x,y)(a^2) = \lambda x^1.loves(x,a)$
b. $\lambda x^1.\lambda y^2.loves(x,y)(a^1) = \lambda y^2.loves(a,y)$ $\lambda x^1.\lambda y^2.loves(x,y)(a^2) = \#$

Now to the linguistic application: flags are assigned e.g. by case markers, like in (2). When applying an unflagged argument, it is per default treated as flagged with ^{0} (as part of the system, not as convention). Note the difference between Hungarian and English transitive verbs. Hungarian does not specify order, English does.

(2) Denotations:
a. $[\![Peter]\!] = p$ $[\![Mary]\!] = m$
b. $[\![acc]\!] = \lambda x^0.x^a$ $[\![nom]\!] = \lambda x^0.x^n$
c. $[\![loves_H]\!] = \lambda x^a y^n.\lambda e^0.love(e) \wedge Agent(e,y) \wedge Theme(e,x)$
d. $[\![loves_E]\!] = \lambda x^a.\lambda y^n.\lambda e^0.love(e) \wedge Agent(e,y) \wedge Theme(e,x)$
e. $[\![Peter + acc]\!] = \lambda x^0.x^a(p) = p^a$
f. $[\![Peter + acc + love_H]\!] = \lambda x^a y^n.\lambda e^0.love(e) \wedge Agent(e,y) \wedge Theme(e,x)(p^a)$
$= \lambda y^n.\lambda e^0.love(e) \wedge Agent(e,y) \wedge Theme(e,p)$

Obviously, such a system is fully free for both flexible and free word order specification both on the main projection line of a syntactic tree and on any level (PP, relative clauses, genitives, etc.). Moreover, since the ^{0} flag is optional it is a conservative extension of the classical system. In the talk, I will concentrate on both the mathematical spell out of the system and on the linguistic application.