Nominal Comparatives in Type-Logical Semantics

Volker Nannen

May 2000

Contents

1 Introduction

This paper is about nominal comparatives in type-logical semantics. As an underlying intention this paper also wants to show the need for more realistic methods of sentence processing: recursion and constraints on ambiguity.
In type-logical semantics nominal comparatives are a tricky subject due to the way the semantics of two compared sets are interwoven. This analysis of nominal comparatives hopes to show that recursion is a computationally more realistic method of interpretation. It is essential in solving complex problems that are out of reach of non-recursive interpretation.

The present system of type-logical semantics is not prepared for recursion. The way the interpretation is coded in a mixture of lambda calculus and first order logic is unfit for further processing. Therefor I can't give a working example of recursion without significant changes to the semantical encoding.

1.1 Recursion

Ordinary semantics take a sentence as input and search for dependencies between the constituents. Then they combine the constituents in the best possible way to produce an interpretation as output. This is a non-recursive method where the output depends on the input only. It allows for a clean separation between textual input and semantical output. Also it produces nice parsing trees on paper.
But this method assumes that everything in a sentence is present at the same time. As long as it yields an interpretation you are allowed to combine any constituent with any other in any order. This is OK for writing where you eyes can scan back and forth over the text. With speech it is unlikely because in speech only a word at a time is really present. Previous words are buffered in a so-called acoustic loop but that loop has limited capacity. Future input is not available at all. Therefor interpretation of the present word can rely only on the last words present in the acoustic loop and the semantics derived from past input. This calls for a recursive interpretation that takes single words as input and combines them with the output of previous interpretation, tearing down the separation between input and output.

In his book "Type-Logical Semantics"[1]. Bob Carpenter associates a logical category with every word and every constituent. This category is responsible for combining a constituent with constituents of other categories in a predefined order. Also every word and constituent has a lambda expression that represents the actual meaning of the constituent. A constituent Con of category $x/z/y$ needs to be followed by some y and some z (in that order). When they are found their meanings become the arguments to the lambda expression of Con. The final output has category x. One of the strong features of this system is that even incomplete sentences can be processed into semantical structures.

But it has two major disadvantages:

- In order to correctly interpret a constituent Con of a backward-looking category $y\backslash z\backslash x$ you must scan **backward** through the past input to find z. And Con may be separated from z by an arbitrary long sequence y.
- A word at the end of a sentence can force you to reorder everything, taking the previous constituents as arguments to its own lambda expression. In this case all the previous interpretation turns obsolete.

Unless some evidence like immediate neighbourhood guarantees its availability in the acoustic loop, interpretation should not depend on previous input. The number of backward-looking slashes in a semantic type therefore should not exceed a constant number, which stands for the most recent constituents. All other dependencies on previous input should be replaced by recursion so that instead of the textual input its semantic interpretation is used.

[1] Bob Carpenter: Type-Logical Semantics, MIT-Press 1997

1.2 Ambiguity

Another issue is the multitude of categories and types that can be assigned to the same word. Imagine a short sentence with four ambiguous words that each has three interpretations. Theoretically this produces $3^4 = 81$ possible combinations. For longer sentences the number of possible combinations would grow exponentially. In ordinary speech even a single ambiguity can throw a listener out of track.
The best thing would be to avoid ambiguities whenever possible. And if they really can't be avoided they have to be eliminated early in the processing. If a word has multiple interpretations, most of them should be canceled out immediately as incompatible with the previous input. The rest should be eliminated within the current sub-clause so that only a single interpretation emerges for further combination. This ensures a realistic computational workload.

This means that categories and types may only differ

- with respect to backward-looking functors
- with respect to the previous interpretation they can combine with
- and with respect to the next two or three arguments they take according to their forward-looking functors, i.e. the constituents that immediately follow them in the sentence.

Any two interpretations that differ only in arguments they would take in a later sub-clause should be avoided. They would produce different threads of interpretation over a long period of time. And they produce an exponential increase in workload.

2 Numerical Quantification

2.1 Determination

Nominal comparatives deal with quantification. The quantifiers as developed by Bob Carpenter are not very useful to the analysis of nominal comparatives. I first have to develope a stricter system of quantification:

For existential quantification like *"a man"* Carpenter uses the construction

$$some(\lambda x \,.\, man(x)). \tag{1}$$

This is logically equivalent to

$$\exists\, x \; man(x). \tag{2}$$

For determination like in *"the man"* he uses the ι-operator:

$$\iota(man). \tag{3}$$

But the behavior of ι is sometimes unregulated as Carpenter admits himself [2]. Determination makes an object unique in a given context. So instead of using a poorly defined quantifier like ι it is better to stick to the well defined

[2] Type-Logical Semantics, page 98

logical quantifier $\exists!$[3] $\exists$ is context dependent and Carpenter argues against this quantifier that there is no theory that can deal with referential context[4] . But Discourse Representation Theory[5] is a known example of a theory that can model the context for every single expression of a discourse. In such a model the use of logical $\exists!$ is perfectly legitimate: *"the man"* should be translated as

$$\exists!\, x\ man(x) \tag{4}$$

saying that there is exactly one man in the given context. In logical formulas the exclamation mark is a shortcut for writing

$$\exists\, x\ (man(x)\ \wedge\ \forall\, y\ (man(y) \rightarrow y = x)). \tag{5}$$

To be conform with the style of Carpenter who writes $some(\lambda x.man(x))$ instead of $\exists x\ man(x)$ I will rewrite (4) as

$$some!(\lambda x\ .\ man(x)). \tag{6}$$

The only difference with (1) is the exclamation mark attached to *some*. The exclamation mark is a useful translation for determination in other cases too.

2.2 Plurals & Numerical Quantification

Carpenter treats plurals as sets, not as numbers of individuals. He makes the cardinality of a set available via a special operator: $\|Q\|$ denotes the cardinality of the set Q.
But this is not the only way to deal with plurals. When combined with a number or with words like *more* or *less* they should better be described as a number of individuals. This can be described as numerical quantification.

Numerical quantification is the numerical extension of existential quantification. Any cardinal number is a numerical determiner. It defines the cardinality of a set of objects. Just as

1 *One apple ripens.*

translates to

$$some\,(\lambda x\ .\ P(x))\,[6], \tag{7}$$

so

2 *Five apples ripen.*

[3] Some logicians call $\exists$ and $\forall$ quantifiers and others call them so only in combination with the variable they bind. From a linguistic perspective it would be appropriate to call them determiners and to call only the combination of determiner and bounded variable quantifier. But since this is not established, I will call them quantifiers both with and without the bounded variable.

[4] page 99: "We have no theory of how such referential restrictions might be constrained grammatically."

[5] Patrick Blackburn & Johan Bos: *Discourse Representation Theory*

[6] $P(x) \equiv apple(x) \wedge ripens(x)$

should translate to

$$some\left(\lambda x_1\,.\,some\left(\lambda x_2\,.\,...\;some\left(\lambda x_5\,.\left(\begin{array}{ccc} & P(x_1)\;\wedge\;P(x_2) & \\ \wedge & ... \quad\wedge\;P(x_5) & \\ \wedge & x_1\neq x_2\neq ...\neq x_5 & \end{array}\right)\right)\right)\right) \tag{8}$$

where there are five distinct existentially quantified objects.

Analogous to (7) this expression can be written as

$$5(\lambda x.P(x)) \tag{9}$$

which means that there exist at least five distinct objects that satisfy P.

The general form for n objects is

$$n(\lambda x.P(x)). \tag{10}$$

2.3 Determination

Determination defines an object as unique in a given context. *"the apple"* allows for exactly one apple in the context of discourse. The same is true for plurals. When you say *"the five apples"*, there can't be more than five apples in your context of discourse. Since the exclamation mark was useful with ordinary determination we will use it here too.

3 *The apple ripens.*

now translates to

$$\exists!\,xP(x)^{7} \tag{11}$$

which means that exactly one object satisfies P. It is a shortcut for writing

$$\exists\,x\;(P(x)\;\wedge\;\forall\,y\;(P(y)\rightarrow y=x)). \tag{12}$$

When *the* combines with a numerical determiner as in

4 *The five apples ripen.*

I want to use a numerical variant of $\exists!$ to express it:

$$5!\;xP(x) \tag{13}$$

which is short for

$$\exists\;x_1\;x_2\;...\;x_5\left(\begin{array}{ccc} & P(x_1)\wedge P(x_2)\wedge ...\wedge P(x_5) & \\ \wedge & x_1\neq x_2\neq ...\neq x_5 & \\ \wedge & \forall\;y\;(P(y)\rightarrow(y=x_1\vee y=x_2\vee ...\vee y=x_5)) & \end{array}\right) \tag{14}$$

and means that there exist *exactly* 5 objects that satisfy P in the given context. The exclamation mark carries the meaning of the determination. Don't confuse it with mathematical faculty. Mathematical 5! evaluates to 120. But this is a

[7] $P(x)\equiv apple(x)\wedge ripens(x)$

logical expression where 5! evaluates to *exactly five*. A case of operator overloading.

Because analogous to Carpenter I wrote formula (11) as

$$some!(\lambda x\ .\ P(x)) \tag{15}$$

I will write the general form of formula (13) as

$$n!(\lambda x\ .\ P(x)) \tag{16}$$

meaning that there are exactly n objects that satisfy P in the given context.

2.4 The Category and Semantical Type of a Numerical Determiner

To integrate numerical determiners (i.e. numbers) into type-logical semantics they need a category and a semantic type. Category and semantic type of an *nd* without the determiner *the* are respectively:

$$\mathbf{nd} : (np \Uparrow s)/n \tag{17}$$

$$nd \equiv \lambda PQ\ .\ nd(\lambda x.\ P(x)\ \wedge\ Q(x)) \tag{18}$$

In combination with the determiner *the* (which is of category np/n) this becomes:

$$\mathbf{nd^2} : (np/n)\backslash((np \Uparrow s)/n) \tag{19}$$

$$nd^2 \equiv \lambda\iota PQ\ .\ nd!(\lambda x.\ P(x)\ \wedge\ Q(x)) \tag{20}$$

For convenience I use the letter ι for the word *the* but it is not Carpenters ι-operator.

3 Nominal Comparatives

5 *There are more bagels than I can count.*

6 *There are more bagels than I can count stars in the sky.*

7 *There are as many bagels as donuts.*

8 *The students ate fewer bagels than the professors.*

Comparatives compare two sets of objects that are described by separate statements. For clarity I will call the set combined with *more*, *fewer* or similar keywords the referent and the set usually combined with *than* I will call the standard. Most of the time I will pick out *more* and *than* and mention the other keywords only when necessary.
When I speak of the complement of *more* I mean the referent that follows it. But when I speak of the complement of *than* I mean the referent and the entire statement that describes the referent.

If you would replace *more* by a number then the statement that describes the referent would be a complete sentence. The complement of *than* can be a complete sentence as well, as in [5]. But usually it is incomplete and its full content has to be derived from the first statement. The standard itself doesn't have to be named explicitly [5, 6 and 8].

Any analysis of nominal comparatives has to address the following points:

- what is the relation between referent and standard?
- what are their monotonic properties?
- how are they quantified?
- what are the categories and semantic types of *more* and *than*?
- how to deal with incomplete complements of *than*?

Especially the last question is where we can put recursion to the test.

3.1 The Relation between the Compared Sets

9 *Butter is cheaper than cheese.*

Comparatives compare two sets of objects. But while adjectival comparatives like in [9] compare the quality of two objects, nominal comparatives usually compare their quantity, e.g. the amount of bagels eaten by the students is compared to the amount eaten by the professors [8].

Both sorts of comparatives, nominal as well as adjectival, have two readings:

relational: as a statement on the relation between referent and standard. In this case the statement tells you for example that one set has more members or is of better quality than the other set. You don't have to know the exact properties of the sets and no calculation is necessary in order to derive the correct meaning.

complex determiner: as a statement on the quality or quantity of the referent. In this case the quality or quantity of the standard is the measure for the referent. The quality or quantity of the standard can be mentioned explicitly or must be known from context.

In this sort of statement *more* fulfills the same function as a numerical determiner. That's why its called a complex determiner.

Take *than* in sentence [7]: **relationally** it tells you that there are bagels and donuts and that the bagels are in the majority. As a **complex determiner** it supposes you already now the amount of donuts (e.g. five in a given context) and tells you that the amount of bagels is bigger (e.g. there are at least six bagels).
When translated into logical formulas the two interpretations look essentially the same. Carpenter translates adjective comparatives with the help of existential quantifiers. This works for nominal comparatives too[8]:

[8] here in plain words, for the extensive formulas and definitions of **add** and **addsome** look at page 274 note 103 of his book

There exists some degree d, some amount $e > 0$ and some quality/quantity Q such that the standard does fulfill Q up to degree d and the referent fulfills Q even up to degree $d + e$.

For nominal comparatives this can be written as

$$some\left(\lambda d\,.\,some\left(\lambda e\,.\,some\left(\lambda Q\,.\,\left(\begin{array}{ccc} & & e>0 \\ \wedge & d & (\lambda x\,.\,stan(x)) \\ \wedge & d+e & (\lambda y\,.\,ref(y)) \end{array}\right)\right)\right)\right) \quad (21)$$

This is good for the relational reading where d only needs to be existentially quantified. But for the complex determiner reading the true properties of the standard might have to be taken from context. In this case you want an abstraction: $\lambda d.(...)$ where the context provides d as an argument. A high level interpretation system could check whether the context provides that argument, choose the proper reading and supply the argument. But apart from the binding of d everything else is equal and I think it sufficient to deal only with the relational reading in this paper.

3.2 Monotonic Properties

Recall the sentences from page 6. There can be *more* bagels than donuts, there can be *fewer* bagels than donuts and there can be *as many* bagels as donuts. These statements are equivalent to the mathematical relations $>$, $<$ and $=$. They have the same monotonic properties:

- *more than* is upward monotonic in its first and downward monotonic in the second set, just as mathematical $>$.
- *fewer than* is downward monotonic in its first and upward monotonic in the second set, just as mathematical $<$.
- *as many as* looses its truth value if any set is replaced by a subset or a superset, just as mathematical $=$.

more and *fewer* have inverse monotonicity properties and *as many* isn't monotonic at all, just as one expects from their mathematical counterparts. Nominal comparatives have the same monotonicity properties as numbers in mathematical equations, an indication that they should be treated in a similar way.

4 Analysis of *more* and *than*

A nominal comparative can appear in three different general forms:

- only with *more*, the standard being supplied by the context:

  10 *Adam eats more apples.*

- *more than* as a single expression:

  11 *Adam eats more than the five apples for breakfast.*

- *than* introducing a new statement that defines the standard. This can be further divided into
  - statements that form a complete sentence:

    12 *Adam eats more apples than Eva eats peaches.*
  - statements that have a verb but miss a noun:

    13 *Adam eats more apples than Eva gave him.*
  - statements that miss a verb and other constituents:

    14 *Adam eats more apples than Eva.*

Each of these appearances will be investigated.

4.1 Analysis of *more*

4.1.1 *more* as a Complex Determiner

15 *Five apples cost two dollars.*

16 *More apples cost three dollars.*

17 *More butter costs three dollars.*

18 **Five butter cost two dollars.*

more is a complex determiner that says that the number of determined objects is greater than some other number. Usually a noun combined with *more* can also be combined with a numerical determiner and vice versa. But words like *butter* form an exception to this rule. They are mass-terms describing some uncountable quantity. In order to combine with a number they first have to become the modifier of a countable unit like pound:

19 *Five pounds of butter cost two dollars.*

This leads to two different interpretations for *more*:

- as abstraction of a numerical quantifier, as in sentence [16]:

$$\lambda\, x\, .\, x\ apples\ cost\ three\ dollar.$$

- as abstraction of a numerical quantifier **and** the unit for the numerical quantifier, as in [19]:

$$\lambda\, x\, \lambda\, unit\, .\, x\ unit\ of\ butter\ cost\ three\ dollar.$$

Because the difference completely depends on *butter* this ambiguity can best be solved by agreement in number. English is rather an exception in the little use it makes of agreement because most associations between constituents depend on their position in the sentence. But other languages use agreement as a key for associations. When type-logical semantics will be applied to such languages agreement has to be incorporated into type-logical semantics anyway. *butter* appears in singular number whereas words that combine with a number must take a plural form[9]. So *more* of the first type should be made to agree with plural forms and *more* of the second type should be made to agree with singular, i.e. mass forms.
For the aim of this paper it is sufficient to deal only with the first type of *more*. By adding the notion of a unit any analysis of *more* can easily be extended to mass-terms.

4.1.2 Category and Semantical Type of *more*

20 *Adam needs more apples for the new distillery project.*

21 *Adam needs more apples for the new distillery project than Eva.*

more can appear in a sentence with and without *than*. Without it the standard has to be taken from context. As explained on page 3 many words may pass before it becomes clear whether *than* will supply a standard. Therefor *more* should get an interpretation that can satisfy both cases. Otherwise the ambiguity could stay unsolved for quite a while. The difference should be worked out by *than* alone.

Like every determiner, *more* takes a noun at its right as an argument and produces a quantifier with scope over the complete sentence. But here the sentence in question isn't complete. It still lacks the standard that has to be supplied either by *than* or by the context. A determiner-like category like

$$^{*}\mathbf{more} : (np \Uparrow s)/n \tag{22}$$

wouldn't be correct because it implies that after taking scope over the sentence we already reached category s when in fact we need to reach category $s{\uparrow}nd$ [10] . The Moortgat-connective q serves us better. Something of category $q(A, B, C)$ acts as a C in the derivation of B, at which point it produces category A[11]. This involves an operation which in computer programming is called dynamic type casting. With this connective we can define the category and semantical type of *more* as:

$$\mathbf{more} : q(s \uparrow nd, s, np)/n \tag{23}$$

$$more \equiv \lambda PQc \,.\, some\ (\lambda n \,.\, (n > c \ \wedge\ n(\lambda x \,.\ P(x) \ \wedge\ Q(x)))) \tag{24}$$

[9] Actually *butter* is a mass-term that is neither singular nor plural. In English a mass-term is not recognized as a separate number, but other languages do recognize it. E.g. in Arabic for the word fish you have a singular form, a plural (and dual) form and you have a special form to speak of fish in general, like butter.
[10] When *than* follows, category s/nd is also fine. But since that is not guaranteed the more general $s{\uparrow}nd$ is better.
[11] Type-Logical Semantics, page 355

The lambda term takes a noun P, a sentence Q and a number c as arguments. It says that there exists some number n such that the number of objects that satisfy $P \wedge Q$ is greater than c.

To give an example:

22 *Adam sees more apples.*

translates to

$$\lambda c \,.\, some\ (\lambda n \,.\, (n > c \ \wedge\ n(\lambda x \,.\, apple(x) \ \wedge\ see(x)(adam)))) \ :\ \mathbf{s} \uparrow \mathbf{nd} \quad (25)$$

4.1.3 *more than* as a single expression

23 *More than two professors ate apples.*

24 *More than* the *two professors ate apples.*

In [23] and [24] *than* directly supplies the numerical determiner for the interpretation of *more*.
[23] isn't actually a comparative because nothing is compared here. The sentence only states that the number of persons that ate apples is bigger than two.
[24] is a true comparative. There are two sets of persons. One set consists of the only two persons in the current context of discourse who are known to have eaten apples. All we know about the other set is that these persons ate apples too and that there are more than two of them. Who they are and whether they are part of the current context is undecided.
With more focus sensitive semantics the meaning of [24] would imply that the two persons that have focus actually ate apples. But with our limited logical tools this is not possible and we can't express this implication. The meaning of the second sentence becomes the same as of the first sentence: the number of persons that ate apples is bigger than two.

more and *than* form a unity with no word standing between them and separating them. To combine them into a logical unity one of them should get a category that consumes the category of the other one and produces a combined category. Which one consumes which one and what the original but consumed category was is irrelevant.
Given the fact that we already assigned a category and semantic type to *more*, for the interpretation of [23] *than* can take the following category and semantic type:

$$\mathbf{than^1} : (q(s \uparrow nd, s, np)/n)\backslash((np \Uparrow s)/n/nd) \quad (26)$$

$$than^1 \equiv \lambda mcPQ \,.\, some\ (\lambda n \,.\, (n > c \ \wedge\ n(\lambda x \,.\ P(x) \ \wedge\ Q(x)))) \quad (27)$$

the first argument m of the semantic type stands for the word *more* that is simply consumed[12] to produce the combined category and semantic type:

[12] *consumed* doesn't mean of no effect. *more* standing to the left of *than* served to select the right category and semantic type for *than*, thus leading to the combined category and semantic type.

$$\textbf{more than} : (np \Uparrow s)/n/nd \tag{28}$$

$$more\ than \equiv \lambda cPQ\ .\ some\ (\lambda n\ .\ (n > c\ \wedge\ n(\lambda x\ .\ \ P(x)\ \wedge\ Q(x)))) \tag{29}$$

The semantic type takes a number c, a noun P and a sentence Q as arguments. The final meaning is that some number n exists that is bigger than c and that there are n objects that satisfy both P and Q. With this analysis sentence [23] translates to

$$some\ (\lambda n\ .\ (n > 2\ \wedge\ n(\lambda x\ .\ \ person(x)\ \wedge\ eat(apple)(x)) : \mathbf{s} \tag{30}$$

For sentence [24] the category and semantic type have to take care of *the*. But since with our limited tools we can't make use of the contextual implications of *the*, this argument is consumed without effect and the result is identical to a sentence without this determination:

$$\textbf{more than} : (np \Uparrow s)/n/nd/\iota \tag{31}$$

$$more\ than \equiv \lambda \iota cPQ\ .\ some\ (\lambda n\ .\ (n > c\ \wedge\ n(\lambda x\ .\ \ P(x)\ \wedge\ Q(x)))) \tag{32}$$

4.2 Analysis of *than*

25 *Adam sees more apples than Eva sees peaches.*

26 *Adam sees more apples than Eva.*

A numerical determiner is missing in the quantification of the complement of *more*. The task of *than* is to provide this numerical determiner. In [25] and [26] *than* introduces a new statement after the statement containing *more* is completed. This statement defines the standard, the set of objects the referent is compared to. In [25] the peaches Eva sees are compared to the apples Adam sees and in [26] the apples Eva sees are compared to the apples Adam sees.
Before we get lost in detail it is helpful to have a general idea of what we want. In order to provide *more* with the numerical determiner, we expect *than* to be of general category and semantic type

$$\textbf{than}^{\mathbf{2}} : (s \uparrow nd)\backslash s/(s \uparrow nd) \tag{33}$$

$$than^2 \equiv \lambda PQ\ .\ some(\lambda n\ .\ P(n)\ \wedge\ Q(n)) \tag{34}$$

meaning that there exists some number n that satisfies the two arguments of *than*, i.e. a number that is equal to the standard and smaller than the referent.

4.2.1 *than* with a Complete Statement

To begin with I will first consider cases where the complement of *than* is complete. If the complement of *than* in [25] was a full sentence, *peaches* would be existentially quantified and take scope over the whole sub-clause:

$$some(\lambda x\,.\,peach(x)\ \wedge\ see(x)(Eva))\ :\ \mathbf{s} \tag{35}$$

To convert this into category $s{\uparrow}nd$ we have to reinterpret the sub-clause. In English any existentially quantified plural noun can be considered as missing a numerical quantifier and hence we can reinterpret "*Eva sees peaches*" as "*Eva sees the n peaches*" with "*the n*" missing. "*the n*" is of category *nd* and when it is missing we get

$$\lambda n\,.\,n!(\lambda x\,.\,peach(x)\ \wedge\ see(x)(Eva)) : \mathbf{s}\uparrow\mathbf{nd} \tag{36}$$

This is true iff the number n supplied in (34) is equal to the number of peaches seen by Eva. The full interpretation of [25] is now

$$some\left(\lambda c\,.\,\left(\begin{array}{c} some(\lambda n\,.\,n > c \wedge n(\lambda x\,.\,apple(x) \wedge see(x)(Adam))) \\ \wedge\ c!(\lambda y\,.\,peach(y)\ \wedge\ see(y)(Eva)) \end{array}\right)\right) \tag{37}$$

In fact any sentence with an existentially quantified plural noun can take the category $s{\uparrow}nd$. This can lead to ambiguity when a numerical quantifier could be considered absent at different places in a sentence. Is the number of apples in the following sentence more than the number of children or more than the number of professors?

27 *Adam sees more apples than professors see children.*

$$some\left(\lambda c.\left(\begin{array}{l} c!(\lambda \mathbf{x}.(\mathbf{child}(\mathbf{x}) \wedge some(\lambda \mathbf{y}\,.\,\mathbf{prof}(\mathbf{y}) \wedge see(\mathbf{x})(\mathbf{y})))) \ \wedge \\ some(\lambda n.(n > c\ \wedge\ n(\lambda z.\,apple(z) \wedge see(z)(adam)))) \end{array}\right)\right) \tag{38}$$

$$some\left(\lambda c.\left(\begin{array}{l} c!(\lambda \mathbf{x}.(\mathbf{prof}(\mathbf{x}) \wedge some(\lambda \mathbf{y}\,.\,\mathbf{child}(\mathbf{y}) \wedge see(\mathbf{y})(\mathbf{x})))) \ \wedge \\ some(\lambda n.(n > c\ \wedge\ n(\lambda z.\,apple(z) \wedge see(z)(adam)))) \end{array}\right)\right) \tag{39}$$

4.2.2 Incomplete *than* with a Verb

In the optimal case the complement of *than* is a full grammatical sentence that can be reinterpreted as of category $s{\uparrow}nd$. But most of the time the complement of*than* is not a full sentence. It may lack the verb with and without its modifiers, the subject and any object. All these have to be derived from the first statement before the complement of *than* can be fully interpreted.
Generally there is a division between complements of *than* where the verb is missing and complements of *than* where the verb is present.

28 *Eva found more apples than usually grow on trees.*

29 *Eva found more apples than Adam will eat.*

When a verb is present, only one noun may may be missing in the statement complementing *than*. This noun must always be identified with the complement of *more*, independently of its thematic role. When complete, the second statement was of category $s\uparrow nd$. Now with the noun missing it must be of category $(s\uparrow nd)\uparrow n$.
Since *more* and its complement can appear at virtually any position in the first statement, there is no way for *than* to extract the missing noun from the first statement by some back-looking category. From the previous section we know that when *than* was encountered, the interpretation of the first statement must have yielded something of type

$$\lambda c\,.\,some\;(\lambda n\,.\,(n > c\;\wedge\;n(\lambda x\,.\;\;P(x)\;\wedge\;Q(x)))):\;\;\mathbf{s}\uparrow\mathbf{nd} \tag{40}$$

where P stands for the noun complement of *more* and Q stands for the first statement. A simple recursive solution would be to let *than* unify the missing noun with P.
But there is a conventional solution, although it is not very realistic because it introduces some ambiguity for *more* that can't be solved before the first statement is completed: *more* can be given a category and semantic type such that it applies P to *than* and its complement. *than* in this case does not play an active role. It only serves to separate the two statements and to trigger the selection of the right category for *more*[13]. I give it the dummy-category τ:

$$\mathbf{more^2} : q(s/((s\uparrow nd)\uparrow n)/\tau\,,\,s\,,\,np)/n \tag{41}$$

$$more^2 \equiv \lambda PQ\tau R\,.\,some\left(\lambda c\,.\,some\left(\lambda n\,.\left(\begin{array}{cc} R(P)(c) & \wedge \\ n > c & \wedge \\ n(\lambda y\,.P(y)\wedge Q(y)) & \end{array}\right)\right)\right) \tag{42}$$

Where R is the complement of *than* and of category $(s\uparrow nd)\uparrow n$. Compare (36) to see that c has to exactly match the quantity of the standard in order to satisfy R. (29) now translates to

$$some\left(\lambda c\,.\,some\left(\lambda n\,.\left(\begin{array}{cc} c!(\lambda x\,.\,apple(x)\wedge eat(x)(adam)) & \wedge \\ n > c & \wedge \\ n(\lambda y\,.apple(y)\wedge find(y)(eva)) & \end{array}\right)\right)\right) \tag{43}$$

4.2.3 Incomplete *than* without a Verb

30 *Today Adam saw more apples on the tree than Eva.*

31 *Today Adam saw more apples on the tree than yesterday.*

32 *Today Adam saw more apples on the tree than peaches.*

33 *Today Adam saw more apples on the tree than in the basket.*

34 *Today Adam saw more apples on the tree than Eva yesterday peaches.*

[13] Under the assumption that ambiguities produce exponentially growing workload, to help choosing the right category for a word is an important semantic task by itself.

35 *Adam gave Seth more apples than Eva peaches. (ambiguous)*

36 *Adam gave Seth more apples than Eva. (ambiguous)*

When the verb is missing it seems a mess: anything present in the first statement can be omitted or replaced in the second statement.
Basically the second statement is a copy of the first one with variations to distinguish the standard from the referent. If a noun in the second statement has the same thematic role as the referent in the first statement, it becomes the new standard. Otherwise standard and referent are occupied by the same noun. In [31] the only difference between the referent apples and the standard apples is the time Adam saw them. What really happens in the brain could be any variation of the following scenario:

Imagine the interpretation of the first statement as some structure in the brain where every constituent is linked to the verb. And every constituent has its unique place according to its thematic role.
When *than* is heard, this structure is copied. Next a number of constituents is heard, each with a thematic role. One is a subject, one an object and one describes time or place of action. In the mental structure each one is written on the unique place that is ascribed to its thematic role, overwriting the constituent that originally occupied that place.

There are many variations on which and how many constituents are to be replaced. And it has to be guaranteed that the thematic roles of the constituents in the *than*-statement match with those in the *more*-statement. This is unsolvable even with such generic categories as

$$\textbf{and} : A\backslash A/A \tag{44}$$

where variables stand for unknown categories. In this case not only the categories are unknown but also the number of arguments and the order in which they appear in the first statement.

5 Conclusion

The last example showed the limit of the present type-logical semantics. To me the natural solution is to make use of the semantic output of the first statement in order to interpret the second statement. Admittedly the logical tools used so far are unfit for such a task. But since they were never designed to allow for further processing of the output, this is no argument against recursive interpretation.

Type-logical semantics is still under development. If recursion is to be integrated into type-logical semantics we have to find a better way of coding the semantic output. A problem with the current notation is that as soon as one constituent has been applied to another one the two have become inseparable. Lambda calculus offers no tools to divide them again. There is no way to extract from a structure like $P(x)$ either the P or the x.

[14] Type-Logical Semantics, page 180:
and takes two variable but identical categories as arguments and produces an output of the same category.