University of Massachusetts Occasional Papers in Linguistics

Volume 2 *u/mass occasional papers in linguistics -- volume II*

Article 5

1976

Some Difficulties with Cresswell's Semantics and the Method of Shallow Structure

John Ruttenberg

some difficulties with cresswell's semantics and the method of shallow structure

john ruttenberg

Cresswell wants to furnish a theory which connects the sentences of English with the sentences of his formal language $\mathcal{L}^{\lambda}$. This is the thrust of the linguistic program of Logics and Languages (Cresswell, 1973). Ideally this theory would assign at least one sentence of $\mathcal{L}^{\lambda}$ to each English sentence. $\mathcal{L}^{\lambda}$ sentences are called "deep structures" of the English sentences to which they are assigned. English sentences would be predicted to have the truth conditions of any of their deep structures. English sentences with two non-equivalent deep structures would be predicted to be ambiguous. In addition, it would be highly desirable if $\mathcal{L}^{\lambda}$ deep structures were assigned only to grammatical -- or perhaps meaningful -- sentences of English. Such a theory, if it could be given, would go a long way toward explaining the relation between the syntax and semantics of English.

As a first step in constructing such a theory, Cresswell suggests that

> "The result of deleting the λ's and the variables [from $\mathcal{L}^{\lambda}$ sentences] gives us something which is very much nearer to the surface structure. It will not, however, quite do as it stands so we shall refer to it as the <u>shallow structure</u> of the λ categorial language," (L&L, p 91).

Though the description just cited contains an inaccuracy, the method Cresswell clearly intends to propose is easy enough to learn.[1] Structure (1) has shallow structure (2), and structure (3) has shallow structure (4):

1. $\langle\langle \underline{\text{John}}, \underline{\text{sings}} \rangle, \underline{\text{and}}, \langle \underline{\text{Arabella}}, \underline{\text{swims}} \rangle\rangle$
2. $\langle \underline{\text{John}}, \underline{\text{sings}}, \underline{\text{and}}, \underline{\text{Arabella}}, \underline{\text{swims}} \rangle$
3. $\langle \underline{\text{John}}, \langle \lambda, X_1, \langle\langle \lambda, Y_1, \langle \underline{\text{loves}}, X_1, Y_1 \rangle\rangle, \underline{\text{Mary}} \rangle\rangle\rangle$
4. $\langle \underline{\text{John}}, \underline{\text{loves}}, \underline{\text{Mary}} \rangle$

(2) and (4) are sentences of neither $\mathcal{L}^{\lambda}$ nor $\mathcal{L}$ (the language just like $\mathcal{L}^{\lambda}$ but without the symbol λ and the variables). They are sequences of symbols of $\mathcal{L}$. If such sequences are not identified with English sentences, it is at least clear that they determine English sentences.[2]

In some cases several nonequivalent $\mathcal{L}^{\lambda}$ sentences will have a single shallow structure. So (5) and (6) both have (7) as shallow structure:

5. $\langle \underline{\text{Every}}, \underline{\text{man}}, \langle \lambda, X_1, \langle\langle \lambda, Y_1, \langle \underline{\text{loves}}, X_1, Y_1 \rangle\rangle, \langle \lambda, X_{\langle 01 \rangle},$
   $\langle \underline{\text{some}}, \underline{\text{woman}}, X_{\langle 01 \rangle} \rangle\rangle\rangle\rangle\rangle$
6. $\langle\langle \lambda, Y_1, \langle \underline{\text{Every}}, \underline{\text{man}}, \langle \lambda, X_1, \langle \underline{\text{loves}}, X_1, Y_1 \rangle\rangle\rangle\rangle, \langle \lambda, X_{\langle 01 \rangle},$
   $\langle \underline{\text{some}}, \underline{\text{woman}}, X_{\langle 01 \rangle} \rangle\rangle\rangle$
7. $\langle \underline{\text{Every}}, \underline{\text{man}}, \underline{\text{loves}}, \underline{\text{some}}, \underline{\text{woman}} \rangle$

This is how the theory is intended to cope with the semantic ambiguity of the sentence determined in English by (7). Since the method of shallow structure maps both (5) and (6) onto (7), (7) is predicted to have (at least) two sets of truth conditions, those associated with (5) and those associated with (6). So (7) is predicted to be semantically ambiguous.

So far everything looks rosy for the method of shallow structure. But not all sentences of $\mathcal{L}^{\lambda}$ have shallow structures which determine grammatical

English sentences. For example, (8) has shallow structure (9):

8. $\langle$ *and*, $\langle$ *John*, *swims* $\rangle$, $\langle$ *sings*, *Arabella* $\rangle\rangle$

9. * $\langle$ *and*, *John*, *swims*, *sings*, *Arabella* $\rangle$

Also there are $\mathcal{L}^{\lambda}$ sentences such as

10. $$\langle\langle \lambda, X_{\langle 01\rangle}, \langle\langle \lambda, X_{\langle 0\langle 01\rangle\rangle}, \langle\langle \lambda, X_{\langle 000\rangle}, \langle\langle \text{John}, X_{\langle 01\rangle}\rangle, X_{\langle 000\rangle}, \langle X_{\langle 0\langle 01\rangle\rangle}, \text{sings} \rangle\rangle\rangle, \text{and} \rangle\rangle, \text{Arabella} \rangle\rangle, \text{swims} \rangle$$

(10) has (2) as its shallow structure. (2) does determine a grammatical English sentence. But (10), which λ converts to

(11) $\langle\langle$ *John*, *swims* $\rangle$, *and*, $\langle$ *Arabella*, *sings* $\rangle\rangle$

can by no stretch of the imagination be construed as giving truth conditions for (2). The sentence, *John sings and Arabella swims*, *cannot* be given a reading which is true just in case John swims and Arabella sings.[3] Any theory that gives this sentence the truth conditions associated with (10) is false.

It may seem that (9) and (10) are isolated aberrations, and that most $\mathcal{L}^{\lambda}$ sentences are better behaved *vis-à-vis* their shallow structures. Then it would be an easy matter to add to the theory the definition of a filter on the sentences of $\mathcal{L}^{\lambda}$. This filter would be a function from the sentences of $\mathcal{L}^{\lambda}$ to, say, $\{0,1\}$. Then the method of shallow structure would only be applied to those sentences whose value under this filter was 1.[4] If (9) and (10) were just isolated aberrations, the job of defining the filter would be relatively easy. The value of all sentences not included in a finite list of exceptions would be 1. (9) and (10) could head that list.

But the well-behaved sentences are the exceptions. Throughout the section of *Logics and Languages* called "English as a Formal Language", Cresswell constantly produces $\mathcal{L}^{\lambda}$ sentences whose shallow structures are English sentences. That he can (nearly) always do so is an outgrowth of two rather basic facts about the sentences of $\mathcal{L}^{\lambda}$ and their shallow structures. The first is that the principles of λ conversion ensure that for any sentence S of $\mathcal{L}^{\lambda}$ with shallow structure $\langle \alpha_1, \ldots, \alpha_n \rangle$, and for any permutation $0_1, \ldots, 0_n$ of $1, \ldots, n$, there is a sentence S' of $\mathcal{L}^{\lambda}$ which is equivalent to S and has shallow structure $\langle \alpha_{0_1}, \ldots, \alpha_{0_n} \rangle$. This means that if a sentence can be derived, then so can any of the permutations of its words. Moreover, each of the permutations can be derived with a prediction of semantic equivalence to the original sentence. The proof of this is simple. It depends on the fact that it is always permissible to move a symbol to the end of a $\mathcal{L}^{\lambda}$ sentence. So, by moving the right symbols out in the right order, any order can be achieved.[5] (1) was converted to (10) by this method. Cresswell used essentially this method to derive either (12) or (13) from (14):

12. $\langle \underline{\text{Every}}, \underline{\text{man}}, \langle \lambda, X_1, \ll \lambda, Y_1, \langle \underline{\text{loves}}, X_1, Y_1 \gg, \langle \lambda, X_{\langle 01\rangle}, \langle \underline{\text{a}}, \underline{\text{woman}}, X_{\langle 01\rangle} \rangle\rangle\rangle\rangle\rangle$

13. $\ll \lambda, X_{\langle 01\rangle}, \ll \lambda, X_{\langle 0\langle 01\rangle\langle 01\rangle\rangle}, \langle \underline{\text{every}}, X_{\langle 01\rangle}, \langle \lambda, X_1, \langle X_{\langle 0\langle 01\rangle\langle 01\rangle\rangle}, \underline{\text{woman}}, \langle \lambda, Y_1, \langle \underline{\text{loves}}, X_1, Y_1 \rangle\rangle\rangle\rangle\rangle\rangle, \underline{\text{a}} \gg, \text{man}\rangle$

14. $\langle \underline{\text{Every}}, \underline{\text{man}}, \langle \lambda, X_1, \langle \underline{\text{a}}, \underline{\text{woman}}, \langle \lambda, Y_1, \langle \underline{\text{loves}}, X_1, Y_1 \rangle\rangle\rangle\rangle\rangle$ (cf L&L 138).

(13) has shallow structure

15. $\langle \underline{\text{Every}}, \underline{\text{woman}}, \underline{\text{loves}}, \underline{\text{a}}, \underline{\text{man}} \rangle$

(13) is equivalent to (12) and (14) and is true just in case for each man there is some woman (his wife, say) that he loves. But its shallow structure is (15) which is certainly ambiguous, but which can by no stretch of the imagination be given the truth conditions associated with (13). Of course,

16. $\langle \underline{\text{a}}, \underline{\text{woman}}, \underline{\text{loves}}, \underline{\text{every}}, \underline{\text{man}} \rangle$
17. $\langle \underline{\text{a}}, \underline{\text{man}}, \underline{\text{loves}}, \underline{\text{every}}, \underline{\text{woman}} \rangle$
18. $* \langle \underline{\text{every}}, \underline{\text{a}}, \underline{\text{loves}}, \underline{\text{man}}, \underline{\text{woman}} \rangle$

and every other permutation of (15) are also shallow structures of sentences of $\mathcal{L}^{\lambda}$ which λ convert to (12).

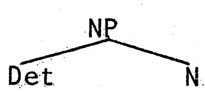
The effect of this upon the filter is that it can no longer be seen as giving 1 to all but a finite number of sentences. For every $\mathcal{L}^{\lambda}$ sentence to which it gives 1, there will be an infinite number of sentences to which it must give 0.[6] The day might still be saved if the method of λ conversion were not so essential to the derivation of $\mathcal{L}^{\lambda}$ sentences which accurately reflect the truth conditions of their shallow structures. Then the filter could give 1 only to least λ converted sentences, perhaps with a finite number of exceptions. But many kinds of English sentences cannot be the shallow structures of least λ converted sentences. λ conversion is essential to proper adverb placement, as in

19. $\langle \underline{\text{Arabella}}, \langle \lambda, X_1, \ll \lambda, X_{\langle 00\rangle}, \ll \lambda, Y_1, \ll \underline{\text{loves}}, X_1, Y_1 \rangle,$
$X_{\langle 00\rangle} \gg, \underline{\text{no}}\ \underline{\text{one}} \gg, \underline{\text{tenderly}} \ggg$

(19), Cresswell says,

> "is a case where λ conversion is essential, for the interpretation of the sentence demands that although the adverb occurs at the end of the sentence, its scope extends only so far as and does not include the penultimate word <u>no</u> <u>one</u> which is adjacent to it in the shallow structure and separates it from its argument," (L&L, p 142).

λ conversion is essential to proper word order in sentences with multiple determiners and/or quantifiers. (12) is a good example of this sort of sentence. It is essential to proper word order in sentences with prepositions whose objects are of the form (20):

20. 

Thus,

21. $\langle \underline{\text{John}}, \underline{\text{serves}}, \underline{\text{in}}, \underline{\text{the}}, \underline{\text{shop}} \rangle$

must come from something at least as complex as

22. $\langle \underline{\text{John}}, \langle \lambda, X_1, \ll \lambda, Y_1, \ll \underline{\text{serves}}, X_1 \rangle, \langle \underline{\text{in}}, Y_1 \ggg, \langle \lambda, X_{\langle 01\rangle},$
$\langle \underline{\text{the}}, \underline{\text{shop}}, X_{\langle 01\rangle} \ggg\gg$

There are many more examples (one, it seems, for practically every one of Cresswell's parts of speech). And as these parts of speech combine into more and more complex sentences, more and more complex λ conversions are required to achieve English word order.

I have argued that the filter cannot operate on the principle of taking the simplest $\mathcal{L}^{\lambda}$ deep structures perhaps with a finite number of exceptions. λ conversion is essential to the derivation of many types of English word orderings. I mentioned above that there are two basic facts about the sentences of $\mathcal{L}^{\lambda}$ and their shallow structures which allow Cresswell to produce $\mathcal{L}^{\lambda}$ sentences for nearly every English shallow structure. I have written at some length about the first fact, but I have not forgotten about the second. If S is a sentence of $\mathcal{L}^{\lambda}$ with shallow structure $< \alpha_1, \ldots, \alpha_i, \ldots, \alpha_j, \ldots, \alpha_n >$ such that $\alpha_i = \alpha_j$, then there is a sentence S' of $\mathcal{L}^{\lambda}$ equivalent to S such that S' has shallow structure $< \alpha_1, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_{j-1}, \alpha_{j+1}, \ldots, \alpha_n, \alpha_i >$. This means that for every English sentence that is derived, all sequences with deletions on identity can also be derived. Again, the method is simple. A variable can always be bound to a word and do its work several times.[7] The principles of λ conversion, as Cresswell says, "allow a single occurrence to do the work of more occurrences," (L&L, p 138).

Of course, this fact is useful in giving $\mathcal{L}^{\lambda}$ sentences for some English sentences. (23) λ conversion to (24), which has (25) as shallow structure:

23. ≪ John, sleep >, and, < Arabella, sleep ≫ [8]
24. ≪ λ, $X_{<01>}$, ≪ John, $X_{<01>}$>, and, < Arabella, $X_{<01>}$>≫, sleep>
25. < John, and, Arabella, sleep > (L&L, p 153)

(26) has shallow structure (27), and is derived from (28) by λ conversion, (L&L, p 154):

26. < John, < λ, X_1, < sleeps, < λ, $X_{<01>}$, ≪ $X_{<01>}$, X_1 >, but,
< Arabella, < does, < λ, Y_1, < not, < $X_{<01>}$, Y_1>>>>>>>>>>
27. < John, sleeps, but, Arabella, does, not >
28. ≪ John, sleeps >, but, < Arabella, < does, < λ, X_1, < not,
< sleeps, X_1 >>>>>>

(29) could come from (30):

29. < Bob, Bill, and, Jimmy, went, to, the, store >

30. $$\langle \underline{\text{Bob}}, \langle \lambda, X_{\langle 0\langle 01\rangle\rangle}, \langle \underline{\text{Bill}}, \langle \lambda, Y_{\langle 0\langle 01\rangle\rangle}, \langle \underline{\text{and}}, \langle \lambda, X_{\langle 000\rangle}, \langle\langle \lambda, X_{\langle 01\rangle}, \langle\langle\langle X_{\langle 0\langle 01\rangle\rangle}, X_{\langle 01\rangle}\rangle, X_{\langle 000\rangle}, \langle Y_{\langle 0\langle 01\rangle\rangle}, X_{\langle 01\rangle}\rangle\rangle, X_{\langle 000\rangle}, \langle \underline{\text{Jimmy}}, X_{\langle 01\rangle}\rangle\rangle\rangle, \langle \lambda, X_1, \langle\langle\ , Y_1, \langle\langle \underline{\text{went}}, X_1 \rangle, \langle \underline{\text{to}}, Y_1 \rangle\rangle\rangle, \langle \lambda, X_{\langle 01\rangle}, \langle \text{the}, \text{store}, X_{\langle 01\rangle}\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle\rangle$$

if it were agreed that the English sentence determined by (29) has a reading which does not require that Bob, Bill and Jimmy went together.[9]

The usefulness of the fact about deletion is offset to some degree by its general inapplicability. Replace <u>and</u> with <u>because</u> (also apparently a member of $E_{\langle 000\rangle}$) in (24) and (31) is derived. Similarly, (32) is derived from a $\mathcal{L}^{\lambda}$ sentence just like (30) but with <u>because</u> again substituted for <u>and</u>.

31. $* \langle \underline{\text{John}}, \underline{\text{because}}, \underline{\text{Arabella}}, \underline{\text{sleep}} \rangle$

32. $* \langle \underline{\text{Bob}}, \underline{\text{Bill}}, \underline{\text{because}}, \underline{\text{Jimmy}}, \underline{\text{went}}, \underline{\text{to}}, \underline{\text{the}}, \underline{\text{store}} \rangle$

This is not the only sort of deletion that is automatically carried out by the formation rules of $\mathcal{L}^{\lambda}$. For example, (33) λ converts to (34), which has shallow structure (35):

33. $$\langle \underline{\text{a}}, \underline{\text{man}}, \langle \lambda, X_1, \langle\langle \lambda, Y_1, \langle \underline{\text{loves}}, X_1, Y_1 \rangle\rangle, \langle \lambda, X_{\langle 01\rangle}, \langle \underline{\text{a}}, \underline{\text{woman}}, X_{\langle 01\rangle}\rangle\rangle\rangle\rangle\rangle$$

34. $$\langle \underline{\text{a}}, \langle \lambda, X_{\langle 0\langle 01\rangle\langle 01\rangle\rangle}, \langle X_{\langle 0\langle 01\rangle\langle 01\rangle\rangle}, \underline{\text{man}}, \langle \lambda, X_1, \langle\langle \lambda, Y_1, \langle \underline{\text{loves}}, X_1, Y_1 \rangle\rangle, \langle \lambda, X_{\langle 01\rangle}, \langle X_{\langle 0\langle 01\rangle\langle 01\rangle\rangle}, \underline{\text{woman}}, X_{\langle 01\rangle}\rangle\rangle\rangle\rangle\rangle\rangle\rangle$$

35. $* \langle \underline{\text{a}}, \underline{\text{man}}, \underline{\text{loves}}, \underline{\text{woman}} \rangle$

The fact about word order and the fact about deletion combine to yield a very general result about the sentences of $\mathcal{L}^{\lambda}$ and their shallow structures. If a shallow structure can be derived, then so can any permutation of its elements. If a shallow structure which has $n>1$ occurrences of a word can be derived, then so can any shallow structure just like it but with m $(0<m<n)$ occurrences of that word. So if, for example, (36) is the shallow structure of some sentence of $\mathcal{L}^{\lambda}$, then all of (37) - (40) will be shallow structures of equivalent sentences of $\mathcal{L}^{\lambda}$:

36. < the, man, kissed, the, woman, in, the, store >
37. * < man, kissed, the, store, in, the, woman >
38. < the, store, kissed, the, man, in, the, woman >
39. * < the, the, store, kissed, man, woman, in >
40. * < the, kissed, man, in, the, store, woman >

What effect does all of this have on the problem of defining the filter ? I think it is fair to say that nearly every question about the syntax of English must now be settled by the definition of the filter. Moreover it must settle specifically all questions about the relation between English syntax and semantics. (So although sentence (38) is grammatical, it must not come from a $\mathcal{L}^{\lambda}$ sentence equivalent to the one from which (36) comes.)

But perhaps the fact that the filter operates on sentences of $\mathcal{L}^{\lambda}$ would make the task of giving its definition somewhat simpler than that of giving some other theory of English syntax and semantics ? This is a hard question. It is hard both because of the amount of work that needs to be done before some clear sense of the nature of the filter emerges, and because it is difficult to compare two highly complex theories for simplicity. Nevertheless, I think it will be a good conclusion to compare briefly Cresswell's system with Montague-Partee grammar. Montague has formation rules for English. These rules are of course far more complex than the formation rules of $\mathcal{L}^{\lambda}$ (or of Montague's own intensional logic). Cresswell has no such complex rules. But the linguist who seeks to add a filter to Cresswell's system must have formation rules for $\mathcal{L}^{\lambda}$ sentences whose shallow structures are to determine English sentences.[10]

The Montague system makes syntactic distinctions among lexical items of the same type. The filter on Cresswell's system must have a way of making the same distinctions. It must, for example, be able to distinguish nouns from verbs and plural noun phrases from singular noun phrases. Because and and both have as their ultimate semantic values functions of the type <000>. But the filter must be able to distinguish them in order to give (24) a different value than it gives to the $\mathcal{L}^{\lambda}$ sentence which would yield (31).

Partee adds labelled bracketing to the Montague system. This enables syntactic rules to operate not only on the basis of the type of a derived expression, but also on the basis of its constituent structure. If some equivalent device is not added to Cresswell's system, then it is hard to see how the filter could be made to work.[11] Just what is there about runs that distinguishes it from < does, run > and thus distinguishes (41) to which the filter should assign the value 1 from (42) which must be assigned 0 ?

41. < John, < does, run >>
42. * < John, < does, < do, run >>>

Both run and < does, run > are present tense expressions of $E_{<01>}$. They are even assigned identical values by the semantics.

How can the filter distinguish (43) from (44) ?

43. < The, boy, runs >
44. * < and, < John, swims >, < Arabella, sings >>

Both are three-membered sequences and members of E_0. The difference between them lies in the difference in type and constituent structure of their members. To distinguish them on this basis the filter must have some way of reconstructing their formation sequences. One way to do this is to have the filter's definition include a list of the lexical items and their types. This is a device as strong as the labelled bracketing of the Montague-Partee system (see footnote 11). And some device at least as strong as labelled bracketing must be added to the definition of the filter in order to make many other necessary distinctions. Without such a device it is very hard to see how any degree of syntactic generality can be achieved.

In sum, Cresswell has defined a very elegant and powerful formal language. But very much more needs to be said before it can be seen as playing a vital part in an important theory about English. Without a great deal of linguistic work, the method of shallow structure offers at best only a device for deciphering the sentences of $\mathcal{L}^{\lambda}$ which appear in *Logics and Languages*. On the other hand, the elegance of $\mathcal{L}^{\lambda}$ and the rest of the linguistic structure offered by Cresswell make it desirable that this work be done.

Footnotes.

[1] And difficult to give a formal definition. Every n-word shallow structure must be an n-membered sequence. The difficulty (and the inaccuracy of Cresswell's statement) comes in because , for example, *Mary* is not a member of $<$ *John*, $< \lambda, X_1, \ll \lambda, Y_1, <$ *loves*, $X_1, Y_1 \gg$, *Mary* $\ggg$. This is a two-membered sequence. It has *John* as its first member and $< \lambda, X_1, \ll \lambda, Y_1, <$ *loves*, $X_1, Y_1 \gg$, *Mary* $\gg$ as its second member. Neither of these is *Mary*. Also, neither one is a variable nor a λ. So the result of deleting the variables and λ's from the sequence is $<$ *John*, $< \lambda, X_1, \ll \lambda, Y_1, <$ *loves*, $X_1, Y_1 \gg$, *Mary* $\ggg$. The definition of a function which takes sentences of $\mathcal{L}^{\lambda}$ onto their shallow structures can be given, but the task is not so obvious as it may at first appear. I leave it as homework. (Remember, $<$ *John*, $<$ *loves*, Mary $\gg \neq <$ *John*, *loves*, *Mary* $>$. The second, not the first, is the required shallow structure.)

[2] In some cases shallow structures will have to be altered slightly before the identification can be made. This is why Cresswell distinguishes shallow from surface structure. For the most part, this will have no bearing here.

[3] Compare this case with Cresswell's examples on pp 90 - 92. No amount of pseudo Old English verse will save the day here.

[4] Cresswell does indeed state that such a filter will be needed, (op cit, pp 92, 155, 224 and elsewhere). The job of defining it he calls "giving acceptability principles". He proposes that this job be left to the "empirical linguists".

5 Formally, if α is a sentence of $\mathcal{L}^{\lambda}$ with shallow structure $< \alpha_1, \ldots, \alpha_n >$, such that $\alpha_1, \ldots, \alpha_n \in E_{\sigma_1}, \ldots, E_{\sigma_n}$, and if $0_1, \ldots, 0_n$ is an ordering of $1, \ldots, n$, then $\alpha' = \ll \lambda, X_{\sigma_{o_n}} <, \ldots, \ll \lambda, X_{\sigma_{o_1}}, \alpha[X_{\sigma_1}/\alpha_1] \ldots [X_{\sigma_n}/\alpha_n] >, \alpha_{o_1} >, \ldots \gg, \alpha_{o_n} >$. α' has shallow structure $< \alpha_{o_1}, \ldots, \alpha_{o_n} >$ and λ converts to S by 6.21, (cf L&L, p 89).

6 Infinite because if α is a sentence of $\mathcal{L}^{\lambda}$ with shallow structure $< \alpha_1, \ldots, \alpha_n >$ such that $\alpha_1, \ldots, \alpha_n \in E_{\sigma_1}, \ldots, E_{\sigma_n}$, then $\ll \lambda, X_{\sigma_n} <, \ldots, \ll \lambda, X_{\sigma_1}, \alpha [X_{\sigma_1}/\alpha_1] \ldots [X_{\sigma_n}/\alpha_n] > \alpha_1 > \ldots \gg, \alpha_n >$ is also a sentence of $\mathcal{L}^{\lambda}$ and λ converts to α.

7 Formally, if α is a sentence of $\mathcal{L}^{\lambda}$ with shallow structure $<\alpha_1, \ldots, \alpha_i, \ldots, \alpha_j, \ldots, \alpha_n >$ such that $\alpha_i = \alpha_j$ and $\alpha_i \in E_{\sigma}$, then $\alpha' = \ll \lambda, X_{\sigma}, \alpha[X_{\sigma}/\alpha_i] >, \alpha_i >$. Since $\alpha_i = \alpha_j$, $\alpha[X_{\sigma}/\alpha_i]$ has X_{σ} in place of both occurrences of α_i.

8 Note, however, that the shallow structure of (23) is < <u>John</u>, <u>sleep</u>, <u>and</u>, <u>Arabella</u>, <u>sleep</u> > which must either be respelt (by I know not what rule) or must get 0 from the filter. There may be some rudiments of an argument here for having the respelling rules (Cresswell's amalgamation function) operate on sentences of $\mathcal{L}^{\lambda}$ instead of their shallow structures.

9 See L&L, p 161 for Cresswell's handling of sentences like this when they do require that eg, Bob, Bill and Jimmy went together.

[10] Or perhaps de-formation rules, though I suspect that it comes to the same thing.

[11] I consider the inclusion of a list of lexical items together with their types in the definition of the filter to be at least an equivalent device since it allows a recursive function to be defined on the sentences of $\mathcal{L}^{\lambda}$ which maps them onto sentences with labelled bracketing.

Bibliography.

Cresswell, M, Logics and Languages, 1973, London: Methuen Press.