THE SYNTAX OF NEGATION AND OPTIMALITY THEORY

Mark Newson

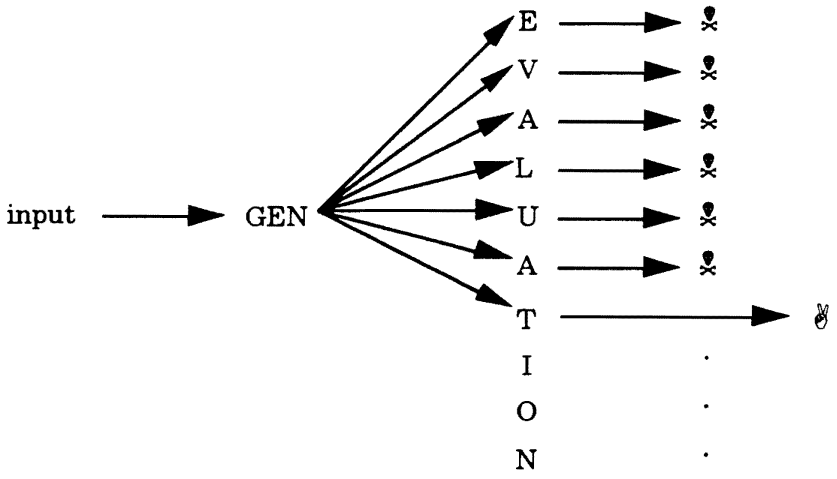
Department of English Linguistics
Eötvös Loránd University, Budapest

1.0 INTRODUCTION

Optimality Theory (OT) is a constraint based theory in which constraints are violable and often in conflict: to conform to constraint A, one has to violate constraint B and *vice versa*. Resolving these conflicts is a matter of deciding which constraint to adhere to and which to violate. This provides an elegant way of capturing the cross linguistic differences, as different languages can be seen as adopting different strategies in the face of these conflicts.

The structure of the theory can be given as in (1):

1



We will take the input to be a set of lexical items from which the sentence is to be built. This notion is similar to Chomsky's (1994) 'Numeration'. The only restriction on the input is that it should be possible to form a full sentence from it and hence the selectional properties of the constituent lexical items should all be 'satisfiable' given the input. GEN is an unconstrained set of linguistically relevant processes which acts on the input to form a potentially infinite number of output structures. The kinds of processes that GEN performs on the input are concatenation, insertion, movement, etc. The set of structures produced by GEN is known as the 'candidate set' and this is evaluated to select the most optimal candidate which will then be the grammatical structure associated with the input. The Evaluation consists of a set of ranked constraints. The ranking of constraints is important for deciding which of any conflicting constraints has primacy: higher ranked constraints are less violable than lower ranked ones. It is at this point that linguistic variation is accounted for: different languages have different constraint rankings. In this talk we will use OT to account for various phenomena concerning the syntax of negation, with particular reference to English, Hungarian and West Flemish.

2.0 THE CONSTRAINTS

Before looking at individual languages, I will first introduce the set of constraints proposed to account for aspects of the syntax of negation. These are the following:

2 Insert
Do not insert any element or structure

3 Move
Do not move elements in a structure

4 Head
All heads must be overt

5 UniSpec (= Unique Specifier)
Any specifier position may only contain one element (no adjunction)

6 LSM (= License *SM*s)
All *SM*s must be licensed

Most of these are self explanatory, though the last needs a few comments. I assume, adapting ideas from Ouhalla, (1990), Haegeman and Zanuttini (1991), Stowell and Beghelli (1994) and Bródy (1995), that scope relations are marked in certain syntactic positions. In particular, the scope position for a negative element is the specifier of a NgP. Scope can be represented with respect to this position in one of two ways: either the negative element itself moves there, or it employs an empty category in SpecNgP to mark its scope. Such an empty category is known as a Scope Marker (*SM*). So, negative scope is marked thus:

7 a ... $[_{NgP}\ Op_i$ Ng ... t_i ...] ...
b ... $[_{NgP}\ SM_i$ Ng ... Op_i ...] ...

If the latter is selected, the *SM* must be licensed, under LSM, by a 'local' negative element, often the negative head. The condition of licensing that I assume is as follows:

8 α licenses β, β a *SM*, iff
i) α is an overt negative element (or its trace)
ii) α and β are overtly adjacent
iii) β c-commands α

9 α is overtly adjacent to β iff there is no overt element γ such that γ intervenes between α and β in the linear string.

3.0 ENGLISH

We first establish that English makes use of *SM*s to represent the scope of its negative operators and does not employ operator movement for this purpose. Consider a simple negative sentence such as:

10 John does not like insincerity

Putting aside the issue of do-support, which we will not deal with in this paper, we assume that in (10) *not* is the head of NgP and that there is an empty operator base generated in SpecNgP, following the analysis of Ouhalla (1990):

11 John does $[_{NgP}\ Op$ not like insincerity]

The empty operator is like a *SM* in that it has to be licensed. The negative head serves this purpose, hence the ungrammaticality of (12):

12 * John does $[_{NgP}\ Op$ e like insincerity]

Compare this situation to one which concerns an overt negative operator generated outside SpecNgP:

13 I saw no one

On our assumptions, this sentence contains a NgP with a *SM* for the negative operator *no one* in its specifier position. The question is, 'where in the structure of (13) is the NgP?'. Assuming that the *SM* must be licensed by a negative element, and given that the only negative element in the sentence is the operator itself, we must assume that the operator licenses its own *SM*. Therefore, the NgP that houses the *SM* must be local enough to the operator to allow this. I propose the structure (14):

14 I saw $[_{NgP}$ *SM* e no one]

There are a number of considerations which support (14). For example, an object negative operator obligatorily has narrower scope than the subject, whereas a negative above the VP can have wider scope than the subject:

15 a everyone saw no one
b everyone didn't see an aardvark

The difference in the interpretation of these can be accounted for, assuming the structures in (16) which embrace the VP-internal subject hypothesis, under the assumption of the Scope Principle of Aoun and Li (1993) given in its simplest form in (17):

16 a everyone$_i$ $[_{VP}$ t_i saw $[_{NgP}$ *SM* e no one]]
b everyone$_i$ $[_{NgP}$ *Op* not $[_{VP}$ t_i see an aardvark]]

17 *The Scope Principle* (Aoun and Li, 1993, p.11)
A quantifier A may have scope over a quantifier B iff A c-commands a member of the chain containing B.

In (16a) the negative *SM* does not c-command any part of the chain of the subject quantifier, whereas it does in (16b). Also note that (at least the head of the chain of) the quantifier c-commands the negative operator in (16b) and hence there are two possible interpretations for this structure.

Under this analysis, when a negative operator licenses its own *SM*, the head of NgP must be empty:

18 * I saw $[_{NgP}$ SM_i not no one$_i$]

Thus, it seems that the negative head is used only as a 'last resort' licenser in English: it is used only when there is no other possible licenser for a negative operator.

A final point before giving the OT analysis of these facts, is that English has Double Negation (DN) structures. When there is more than one negative element in a clause, each of these retains its negative force in the interpretation and hence we get a cancelling out effect. Thus, consider (19):

19 I did not do nothing

I propose the following structure for this sentence, the reasons for which will become clear later:

20 I did $[_{NgP}$ *Op* not do $[_{NgP}$ SM_i e nothing$_i$]]

First note that this structure conforms to the licensing conditions we have been assuming: the empty operator is licensed by the negative head and the overt operator licenses its own *SM*. That this is not a particularly unusual structure for English is indicated by the fact that English seems capable of having a number of NgPs in one clause:

21 he may not have not been reading in the bath

According to basic structural principles, there will be a phrase for every head and hence the multiple appearance of a negative head in a clause means the multiple appearance of NgP in that clause.

In accounting for the above facts in an optimality framework, it is important to note that of the constraints we suggested earlier, Insert conflicts with all others. For example, a language has the choice of either inserting a *SM* for an operator or moving this operator to a scope position. The first option violates Insert and the second violates Move. English ranks Move above Insert: it will be more optimal to violate Insert than Move. Insert also conflicts with LSM and Head, both of which militate for the insertion of a negative head, in violation of Insert. Obviously, English does not have an obligatory negative head and this argues that Insert is ranked above Head. However, the fact that the head is necessary to licence the empty negative operator suggests that LSM outranks Insert. Finally, Insert conflicts with UniSpec in that the latter forces a language to have a unique scope position and hence a unique NgP for every negative operator in the input. This violates Insert as it forces more structure to be inserted: a language which allowed its specifier positions to be multiply filled would only require a single NgP per clause which is more optimal according to Insert. The fact that English has DN structures therefore argues that UniSpec is ranked higher than Insert. The final ranking proposed is therefore:

22 UniSpec, LSM, Move > Insert > Head

In (22) the relative positions of UniSpec, LSM and Move to each other is of no consequence as these do not conflict and ranking is only important for conflicting constraints.

Consider first a simple negative sentence such as (23a) with its proposed structure (23b):

23 a they did not leave
b they [$_{NgP}$ *Op* not leave]

We will assume that negative sentences differ from positive ones in the inclusion of the negative operator in their inputs. Thus, we are assuming that negative heads are inserted into the structure by GEN. The main issue facing this sort of sentence, therefore, is whether or not to insert the negative head:

24

{they, *Op*, leave}	UniSpec	LSM	Move	Insert	Head
they [$_{NgP}$ *Op* e leave]	✓	*☠			
☞ they [$_{NgP}$ *Op* not leave]	✓	✓			

Obviously, LSM works to rule out the structure without a negative head as the empty operator will be unlicensed. Thus, although the optimal structure violates Insert, all other competing candidates will violate more highly ranked constraints.

Now consider a case of a negative operator in object position. Recall, in this case, that the NgP is inserted low down in the structure and the head is obligatorily missing. There are at least two issues to decide on: whether to insert a negative head to licence an inserted *SM*, or whether to move the operator to avoid having to insert a *SM* at all.

25

{he, saw, no one}	UniSpec	LSM	Move	Insert	Head
☞ he saw [$_{NgP}$ *SM*$_i$ e no one$_i$]	✓	✓	✓	**	
he saw [$_{NgP}$ *SM*$_i$ not no one$_i$]	✓	✓	✓	***☠	
he [$_{NgP}$ no one$_i$ e saw t$_i$]	✓	✓	*☠		
he saw [$_{NgP}$ no one$_i$ e t$_i$]	✓	✓	*☠		

Obviously, moving the operator will be non-optimal as Move outranks Insert. The insertion of a negative head is an unnecessary violation of Insert, the inserted *SM* being already licensed by the quantifier.

Finally, we consider a DN structure. These involve inputs with two negative operators as in (26):

26 I [$_{NgP}$ *Op* not say [$_{NgP}$ SM_i e nothing$_i$]] (I did not say nothing)

27

{I, *Op*, say, nothing}	UniSpec	LSM	Move	Insert	Head
I [$_{NgP}$ SM_i *Op* not say nothing$_i$]	*☹				
☞ I [$_{NgP}$ *Op* not say [$_{NgP}$ SM_i e nothing$_i$]]	✓				

Here the critical decision is how many NgPs to insert. As any structure that has fewer NgPs than negative operators will be forced to violate UniSpec, no matter how many violations of Insert it produces, the optimal candidate will be the one that provides a NgP for all negative operators as UniSpec is dominant.

The basic English facts therefore present very few problems for this OT analysis. There remain one or two outstanding issues which for reasons of time we have not discussed, such as the treatment of a negative operator in subject position. These require a little more argumentation, however they can be handled satisfactorily under the assumptions made so far (see Newson 1994 for a fuller treatment).

4.0 HUNGARIAN

Like English, Hungarian also makes use of empty negative operators and *SM*s. A simple negative sentence involves an empty negative operator licensed by the negative head *nem*:

28 Gyula [$_{NgP}$ *Op* nem érti]
Gyula not understand-3s-def
"Gyula doesn't understand it"

However, this is where the similarity between the two languages ends. The Hungarian NgP has a fixed position above the VP, following topic and 'subject' positions (see Kiss 1992 on the basic structure of the Hungarian clause). While this means that a VP internal negative operator is never in a position to license its own *SM*, the licensing of *SM*s is not an important issue as the negative head is obligatory in all negative sentences. Hence a *SM* in SpecNgP will always be licensed no matter what its relationship with its operator is. On the other hand, Hungarian also allows negative operators to move to SpecNgP, thus eradicating the need for a *SM*. But even though in these structures the head is not needed to licence a *SM*, it is still obligatory:

29 a [$_{NgP}$ SM_i nem csinál semmit$_i$]
not do-3s nothing-acc
"he doesn't do anything"
b [$_{NgP}$ semmit$_i$ nem csinál t$_i$]
"he doesn't do anything"
c * [$_{NgP}$ semmit$_i$ e csinál t$_i$]

A final difference between Hungarian and English is that Hungarian allows only Negative Concord (NC) structures. This means that multiple negative elements are used in sentences to express a single negation. This can be seen in sentences such as (30):

30 senki nem látott semmit
no one not saw nothing-acc
"no one saw anything"

Following Haegeman and Zanuttini (1991), we assume that NC arises in situations where multiple negative elements are associated with a single SpecNgP. When more than one negative element or their *SM*s are in or adjoined to SpecNgP there is a 'factorisation' of their negative features and the result is the expression of a single negation. This is very similar to the idea proposed by Higginbotham and May (1981) that in multiple wh-questions there is an absorption of the [+wh] features and such sentences express a single question. Of course, the contrast is with DN structures, which we have described as structures containing more than one NgP. Obviously, when negative elements are associated with different SpecNgPs, there can be no factorisation of their negative features and a DN reading is the result.

The fact that the Hungarian negative head is obligatory in negative sentences indicates that Hungarian ranks Head above Insert. This leads to the fact that LSM will always be adhered to, and hence has very little work to do. For this reason we can place this constraint low in the ranking. That Hungarian has NC instead of DN structures argues that Insert outranks UniSpec: it is more optimal to associate multiple negative operators with a single SpecNgP than it is to insert the extra structure needed to provide each with its own scope position.

It may at first seem problematic that Hungarian allows both operator movement and the insertion of *SM*s optionally: we have dealt with these phenomena in terms of two conflicting constraints (Insert and Move) - how can a language conform to both? The answer I propose is straightforward: conflicting constraints are not always ranked with respect to each other. When such constraints are not ranked, both occupy the same position in the ranking. As these constraints conflict, every relevant structure will violate one or the other and hence every structure represents a violation of one constraint at this rank position. When this happens no candidate is eliminated and all survive to be further evaluated.

The ranking I propose for Hungarian is (31), where equal ranking of conflicting constraints is shown by braces and the ranking of non-conflicting constraints is unimportant:

31 Head > {Insert, Move} > UniSpec, LSM

As we have only so far considered DN structures, I will demonstrate here how placing Insert higher than UniSpec in the ranking leads to NC structures. Consider the sentence:

32 $[_{NgP}$ senki$_j$ SM_i nem csinál semmit$_i$ t$_j$]
no one not does nothing-acc
"no one does anything"

Here there is a single NgP, the specifier of which contains the *SM* for a negative operator in the VP and an operator moved to adjoin to it. In this configuration, the negative features of the moved operator and the *SM* are "factored out", hence only one negation is expressed. The movement is optional and has no bearing on the NC issue, hence we will not discuss it here - we return to the issue below. The question is why is there only one NgP? The table in (33) compares this structure to one in which each operator is associated with a unique SpecNgP:

33

{senki, semmit, csinál}	Head	{Insert	Move}	UniSpec	LSM
☞ $[_{NgP}$ senki$_j$ SM_i nem csinál semmit$_i$ t$_j$]	✓	{***	*}		
$[_{NgP}$ senki$_j$ nem $[_{NgP}$ SM_i nem csinál semmit$_i$ t$_j$]]	✓	{*****	*}☠		

As the table shows, the optimality of the single NgP is decided on the Insert constraint: inserting an extra NgP will always constitute more violations of Insert than structures with only one NgP. Even if the insertion of the head were not necessary, or if the *SM* were not inserted but the second operator were to be moved, the structure with two NgPs involves inserting one more NgP than one

Obviously, moving the operator will be non-optimal as Move outranks Insert. The insertion of a negative head is an unnecessary violation of Insert, the inserted *SM* being already licensed by the quantifier.

Finally, we consider a DN structure. These involve inputs with two negative operators as in (26):

26 I [$_{NgP}$ *Op* not say [$_{NgP}$ SM_i e nothing$_i$]] (I did not say nothing)

27

{I, *Op*, say, nothing}	UniSpec	LSM	Move	Insert	Head
I [$_{NgP}$ SM_i *Op* not say nothing$_i$]	*☹				
☞ I [$_{NgP}$ *Op* not say [$_{NgP}$ SM_i e nothing$_i$]]	✓				

Here the critical decision is how many NgPs to insert. As any structure that has fewer NgPs than negative operators will be forced to violate UniSpec, no matter how many violations of Insert it produces, the optimal candidate will be the one that provides a NgP for all negative operators as UniSpec is dominant.

The basic English facts therefore present very few problems for this OT analysis. There remain one or two outstanding issues which for reasons of time we have not discussed, such as the treatment of a negative operator in subject position. These require a little more argumentation, however they can be handled satisfactorily under the assumptions made so far (see Newson 1994 for a fuller treatment).

4.0 HUNGARIAN

Like English, Hungarian also makes use of empty negative operators and *SM*s. A simple negative sentence involves an empty negative operator licensed by the negative head *nem*:

28 Gyula [$_{NgP}$ *Op* nem érti]
Gyula not understand-3s-def
"Gyula doesn't understand it"

However, this is where the similarity between the two languages ends. The Hungarian NgP has a fixed position above the VP, following topic and 'subject' positions (see Kiss 1992 on the basic structure of the Hungarian clause). While this means that a VP internal negative operator is never in a position to license its own *SM*, the licensing of *SM*s is not an important issue as the negative head is obligatory in all negative sentences. Hence a *SM* in SpecNgP will always be licensed no matter what its relationship with its operator is. On the other hand, Hungarian also allows negative operators to move to SpecNgP, thus eradicating the need for a *SM*. But even though in these structures the head is not needed to licence a *SM*, it is still obligatory:

29 a [$_{NgP}$ SM_i nem csinál semmit$_i$]
not do-3s nothing-acc
"he doesn't do anything"
b [$_{NgP}$ semmit$_i$ nem csinál t$_i$]
"he doesn't do anything"
c * [$_{NgP}$ semmit$_i$ e csinál t$_i$]

A final difference between Hungarian and English is that Hungarian allows only Negative Concord (NC) structures. This means that multiple negative elements are used in sentences to express a single negation. This can be seen in sentences such as (30):

30 senki nem látott semmit
no one not saw nothing-acc
"no one saw anything"

Following Haegeman and Zanuttini (1991), we assume that NC arises in situations where multiple negative elements are associated with a single SpecNgP. When more than one negative element or their *SM*s are in or adjoined to SpecNgP there is a 'factorisation' of their negative features and the result is the expression of a single negation. This is very similar to the idea proposed by Higginbotham and May (1981) that in multiple wh-questions there is an absorption of the [+wh] features and such sentences express a single question. Of course, the contrast is with DN structures, which we have described as structures containing more than one NgP. Obviously, when negative elements are associated with different SpecNgPs, there can be no factorisation of their negative features and a DN reading is the result.

The fact that the Hungarian negative head is obligatory in negative sentences indicates that Hungarian ranks Head above Insert. This leads to the fact that LSM will always be adhered to, and hence has very little work to do. For this reason we can place this constraint low in the ranking. That Hungarian has NC instead of DN structures argues that Insert outranks UniSpec: it is more optimal to associate multiple negative operators with a single SpecNgP than it is to insert the extra structure needed to provide each with its own scope position.

It may at first seem problematic that Hungarian allows both operator movement and the insertion of *SM*s optionally: we have dealt with these phenomena in terms of two conflicting constraints (Insert and Move) - how can a language conform to both? The answer I propose is straightforward: conflicting constraints are not always ranked with respect to each other. When such constraints are not ranked, both occupy the same position in the ranking. As these constraints conflict , every relevant structure will violate one or the other and hence every structure represents a violation of one constraint at this rank position. When this happens no candidate is eliminated and all survive to be further evaluated.

The ranking I propose for Hungarian is (31), where equal ranking of conflicting constraints is shown by braces and the ranking of non-conflicting constraints is unimportant:

31 Head > {Insert, Move} > UniSpec, LSM

As we have only so far considered DN structures, I will demonstrate here how placing Insert higher than UniSpec in the ranking leads to NC structures. Consider the sentence:

32 $[_{NgP}$ senki$_j$ SM_i nem csinál semmit$_i$ t$_j$]
no one not does nothing-acc
"no one does anything"

Here there is a single NgP, the specifier of which contains the *SM* for a negative operator in the VP and an operator moved to adjoin to it. In this configuration, the negative features of the moved operator and the *SM* are "factored out", hence only one negation is expressed. The movement is optional and has no bearing on the NC issue, hence we will not discuss it here - we return to the issue below. The question is why is there only one NgP? The table in (33) compares this structure to one in which each operator is associated with a unique SpecNgP:

33

{senki, semmit, csinál}	Head	{Insert	Move}	UniSpec	LSM
☞ $[_{NgP}$ senki$_j$ SM_i nem csinál semmit$_i$ t$_j$]	✓	{***	*}		
$[_{NgP}$ senki$_j$ nem $[_{NgP}$ SM_i nem csinál semmit$_i$ t$_j$]]	✓	{*****	*}*!		

As the table shows, the optimality of the single NgP is decided on the Insert constraint: inserting an extra NgP will always constitute more violations of Insert than structures with only one NgP. Even if the insertion of the head were not necessary, or if the *SM* were not inserted but the second operator were to be moved, the structure with two NgPs involves inserting one more NgP than one

with only one. Hence having Insert above UniSpec in the ranking will force the language to stack multiple operators or their *SM*s in a single SpecNgP and hence force a NC reading.

Finally, we turn to the optionality of inserting *SM*s or moving negative operators to provide them with their scope interpretations. This follows from the fact that Insert and Move are not ranked with respect to each other. We take a simple example:

34 a $[_{NgP}$ semmit$_i$ nem látok t$_i$]
nothing-acc not see-1s
"I don't see anything"
b $[_{NgP}$ *SM*$_i$ nem látok semmit$_i$]
"I don't see anything"

As both of these structures are grammatical, they should be equally optimal. That no other structure is more optimal than those in (34) follows from what we have discussed above: there must be a single NgP with an overt head otherwise some high ranking constraint will be violated. That both of these structures violate the constraits equally is shown in the following table:

35

{látok, semmit}	Head	{Insert	Move}	UniSpec	LSM
☞ $[_{NgP}$ semmit$_i$ nem látok t$_i$]	✓	{**	*}	✓	✓
☞ $[_{NgP}$ *SM*$_i$ nem látok semmit$_i$]	✓	{***	✓}	✓	✓

The first structure violates Insert only twice (insertion of the NgP and insertion of the head), whereas the second violates this constraint three times. However, the first violates Move once and the second does not violate it at all. Thus both structures violate constraints at this rank position three times each and therefore are equally optimal. It is easy to see that this result carries over to more complex examples, with more negative operators. The difference between the structures will be whether a *SM* is inserted or the operator is moved. If we choose to satisfy one constraint, we violate the other and *vice versa*. The structure with the fewer Insert violations will be the one with the most Move violations, and structures with fewer Move violations will have exactly the same number more Insert violations. Thus, no matter how many operators there are, moving them will always be as optimal as inserting *SM*s for them and, therefore, operator movement is an optional syntactic process.

5.0 WEST FLEMISH

The final language we will consider is West Flemish. All data are taken from Haegeman and Zanuttini (1991) and Haegeman (1992). West Flemish (WF) differs from English and Hungarian in that it has no empty negative operators or *SM*s. The operator used in simple negative sentences is overt, and other negative operators always move to SpecNgP, thus there is no insertion of *SM*s:

36 a da ze nie ketent van eur werk (en-)was
that she *Op* contented with her work ng-was
"that she was not pleased with her work"
b da ze me niks ketent (en-)was
that she with nothing contented ng-was
"that she was not pleased with anything"
c * da ze ketent me niks (en-)was

(WF is a V2 language and to avoid complicating issues concerning word order we will only consider subordinate clauses.) Haegeman and Zanuttini (1991) argue that the NgP in WF sits above the VP but below the inflectional nodes. The verb moves out of VP, passing through the NgP head and picking it up as it does so, and ends up in AGR (in subordinate clauses). The overt operator *nie* stays in SpecNgP. Thus, in (36a), it is the operator which indicates the position of the NgP, not the negative head. A more detailed analysis is given in (37):

37 da ze $[_{NgP}$ nie t$_i$ $[_{VP}$ t$_i$ ketent van eur werk]] en-was$_i$

Note that the position of the PP complement of the adjective *ketent* is normally following it. (36b and c) demonstrate the situation with a negative operator generated outside NgP: these must move to SpecNgP to be properly interpreted. The ungrammaticality of (36c) indicates that the insertion of *SM*s, leaving the negative operator *in situ*, is prohibited. This leads us to the conclusion that Insert must be more highly ranked than Move.

Because there are never any empty negative operators to licence, LSM does no work in WF. Moreover, the negative head is never required for licensing purposes. Thus, the appearance of the negative head in negative sentences must be entirely due to Head. However, in all relevant cases the negative head is optional in WF, indicating that Head and Insert are equally ranked.

An interesting fact about WF is that it has both DN and NC structures. However, both are associated with different surface orders. This provides us with further support for our analysis of the difference between DN and NC structures. Consider the following data:

38 a da ze $[_{NgP}$ me niemand$_j$ nie t_i $[_{VP}$ t_i ketent $t_j]]$ (en-)was$_i$
that she with no one *Op* pleased ng-was
"that she wasn't pleased with anyone"
b da ze nie me niemand ketent (en-)was
that she *OP* with no one pleased ng-was
"that she wasn't pleased with no one"

(38a) demonstrates that a NC structure is achieved by moving the negative operator to the left of *nie* generated in SpecNgP, presumably adjoining to it. We have claimed that this is the configuration under which the negative features of the operators will be factored out and a NC reading results. This is confirmed by this datum. It is at first puzzling why, when the operator is moved to the right of SpecNgP, a DN reading should result. Note that the negative PP complement does move from its base generated position to the right of the adjective, in line with our claims that WF is unable to insert *SM*s. But this argues that the operator moves to some SpecNgP in order to receive its scope interpretation. If this movement were to right adjoin the PP to *nie* we would have no account of why there should be a DN reading: there is no reason to believe that the process of negative factorisation should be affected by which side an element is adjoined to another. Furthermore, Haegeman and Zanuttini (1990) point out that when the PP moves to the right of *nie* it has narrower scope than when it moves to the left. Again, if both movements are to the same position then we have no account for this difference, under the assumption that scope is determined under c-command. However, if we assume the structure in (39), then all these puzzles are satisfactorily answered:

39 da ze $[_{NgP}$ nie t_i $[_{NgP}$ me niemand$_j$ t_i $[_{VP}$ t_i ketent $t_j]]$ (en-)was$_i$

Assuming a second lower NgP providing a unique scope position for the negative PP accounts for both the scope interpretation and the DN reading: the PP, having a lower scope position, will naturally have a narrower scope and as it is not associated with the same SpecNgP as the negative operator, the factorisation of their negative features cannot take place. This adds strong support for our analysis of DN as involving more than one NgP.

The fact that WF allows both DN and NC structures leads to the assumption that UniSpec and Insert are not ranked with respect to each other and thus, the final ranking I will propose for WF is as in (40):

40 {Insert, UniSpec, Head} > Move, LSM

For reasons of space, here we present only an example of how the non-ranking of Insert and UniSpec produce optional DN and NC structures for the same inputs. Consider again the structures (38a) and (39), given here as (41a) and (b) respectively:

41 a da ze [$_{NgP}$ me niemand$_j$ nie t_i [$_{VP}$ t_i ketent t_j]] (en-)was$_i$
b da ze [$_{NgP}$ nie t_i [$_{NgP}$ me niemand$_j$ t_i [$_{VP}$ t_i ketent t_j]] (en-)was$_i$

The crucial issue is how many NgPs to insert into the structure: one or two (any more will give rise to unnecessary Insert violations and any fewer will not provide the structure with an interpretation). That these options are equally optimal is shown in table (42):

42

{... ze, nie, was, ketent, me, niemand}	{Insert	UniSpec	Head}	Move	LSM
☞ da ze [$_{NgP}$ me niemand$_j$ nie t_i [$_{VP}$ t_i ketent t_j]] (en-)was$_i$	{*	*	-}	*	✓
☞ da ze [$_{NgP}$ nie t_i [$_{NgP}$ me niemand$_j$ t_i [$_{VP}$ t_i ketent t_j]] (en-)was$_i$	{**	✓	-}	*	✓

Here we ignore the issue of the optional head: if the head is inserted there will be one more violation of Insert in both cases and if it is not, there will be a violation of Head in both cases. This makes no difference to the issue of the optimality of the DN and NC structures. The NC structure inserts only one NgP and hence violates Insert once. However, as a result, both operators have to share the same specifier position, in violation of UniSpec. For the DN structure, both operators are provided with their own scope position, but at the expense of an extra Insert violation. Once more, it is obvious how the more violations of UniSpec there are, the fewer violations of Insert there will be, and *vice versa*, and therefore this analysis extends to more complex cases where there are more negative operators.

6.0 CONCLUSION

Under the assumptions of OT that constraints are not inviolable but are ranked with respect to each other to determine which violations are acceptable and which are not, I have proposed an analysis that elegantly captures certain negative phenomena in three diverse languages. The main advantage of this analysis over other possible accounts is precisely that it allows constraints to be violated in certain circumstances: if constraints were inviolable either more complicated constraints would have to be proposed or we would have to invent complicated conditions on when such constraints are applicable. Optionality is another problem with non-violable constraints: how can constraints be both applicable and non-applicable in any one language? As we have seen OT provides a very simple way of accounting for optionality through the (non)ranking of the constraints.

REFERENCES

Aoun, J. and Y.-H. A. Li 1993 *Syntax of Scope*, MIT Press, Cambridge, Massachusetts.

Bródy, M. 1995 *Lexico-Logical Form: A Radically Minimalist Theory*, MIT Press, Cambridge, Massachusetts.

Chomsky, N. 1994 'Bare Phrase Structure', *MIT Occasional Papers in Linguistics* 5, MIT, Cambridge, Massachusetts.

Haegeman, L. 1992 'Negative Heads and Negative Operators, the Neg Criterion', ms. University of Geneva.

Haegeman, L. and R. Zanuttini 1990 'Negative Heads and the Neg Criterion', paper presented at GLOW.

Haegeman, L. and R. Zanuttini 1991 'Negative Heads and Negative Concord', ms. University of Geneva.

Higginbotham, J. and R. May 1981 'Questions, Quantifiers and Crossing', *Linguistic Review* 1, 41-79.

Kiss, K. 1992 'Move-alpha and Scrambling in Hungarian', in Approaches to Hungarian: Vol 4 The Structure of Hungarian, Kenesei, I. and Cs Pléh (eds.), JATE, Szeged.

Newson, M. 1994 'Negation, Double Negation and Optimality Theory', ms. Eötvös Loránd University, Budapest.

Ouhalla, J. 1990 'Sentential Negation, Relativized Minimality and the Aspectual Status of Auxiliaries', *Linguistic Review* 7, 183-231.

Stowell, T. and F. Beghelli 1994 'The Direction of Quantifier Movement', paper presented at GLOW, Vienna.