

## Multiple Multiple Spellout\*

Meaghan Fowlie  
McGill University

### ABSTRACT

It has been posited that Spellout, like other syntactic operations, can occur more than once (Uriagereka 1999). I suggest that what we call Spellout is in fact at least two separate operations: at minimum, one which determines the linear order of Lexical Items (“LINEARIZE”) and another which sends the phonological features of the Spelled-out constituent to the phonological component of the language faculty, rendering that constituent unavailable to the syntactic derivation (“ATOMIZE”). The separate application of these operations can yield phenomena such as Holmberg’s Generalisation and other successive cyclicity effects (Fox & Pesetsky 2005): (LINEARIZE without ATOMIZE) and Scrambling (ATOMIZE without LINEARIZE).

### 1. INTRODUCTION

In the Minimalist Program, Spellout is the syntactic operation that delivers to the phonological component of the language faculty everything it needs in order to form an utterable sentence. This includes all of the phonological features of the terminal nodes – be they lexical items or morphemes – and the order in which they are to appear. Uriagereka (1999) introduced the notion of *Multiple Spellout*: that Spellout can occur several times throughout the derivation. Working within this framework, researchers have been asking what exactly Spellout is, and how a Spelled-out constituent behaves.

Given that phonological features and linear order are quite different animals, I suggest that Spellout is in fact (at least) two operations: one that assigns linear order to the terminal nodes of a constituent, and one that sends phonological features to the phonological component.

### 2. BACKGROUND

#### 2.1 LINEAR CORRESPONDENCE AXIOM (LCA)

X-bar theory offers no mechanical algorithm to map hierarchical structure to the surface linear form of language. Any pair of sisters can be stipulated to appear in either order. In particular, languages have parameters that determine the linear order of specifiers, heads, and complements. In his 1994 monograph *The Antisymmetry of Syntax*, Richard Kayne proposed that linear order *is* in fact derivable from hierarchical structure. He showed that it was possible to derive X-bar assumptions from c-command relations. In particular, he proposed the Linear Correspondence Axiom (LCA):

---

\* I would like to thank the many students and professors who have taken time to advise me on this project, in particular Jon Nissenbaum, Tobin Skinner, and especially my thesis supervisor Lisa De Mena Travis, who was also responsible for pointing out the possibilities this approach presents for Scrambling.

***Linear Correspondence Axiom*** (Kayne 1994): For any pair of non-terminal nodes  $\langle X, Y \rangle$ , if X asymmetrically c-commands Y then each terminal node dominated by X precedes each terminal node dominated by Y. The set of all such correspondences constitutes a total ordering on the terminal nodes.

Kayne assumes irreflexive dominance and that the terminal nodes (e.g. lexical items) project up to a syntactic head without branching. At least one of these two assumptions is necessary to derive a total ordering on the terminal nodes.

Nunes & Uriagereka (2000) propose that the Minimalist assumption of Bare Phrase Structure is correct (i.e. terminal nodes *do not* project up to a syntactic head without branching) and that the LCA is simpler than Kayne's statement. In particular, they remove the notion of dominance from the definition of the LCA.

***Linear Correspondence Axiom*** (Nunes & Uriagereka 2000): A Lexical Item  $\alpha$  precedes a Lexical Item  $\beta$  iff  $\alpha$  asymmetrically c-commands  $\beta$ .

The removal of dominance from the definition of the LCA means that Nunes & Uriagereka are no longer interested in anything but terminal nodes. This is in contrast to Kayne, who uses mathematical relations among non-terminals to determine linear order of terminals. Nunes & Uriagereka determine linear order of terminals directly from the mathematical relations among the terminals themselves.

## 2.2 FOX & PESETSKY

Fox & Pesetsky (2003, 2005) propose that Spellout fixes the relative order of the lexical items in a Spelled-out domain. At the end of the construction of each *Spellout Domain* (SD)  $D$ ,

1. The elements of  $D$  are linearised by some linearisation algorithm
2. Resultant ordered pairs are stored on an Ordering Table

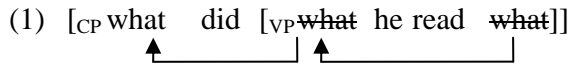
They call this operation Linearize. The rule of *Order Preservation* states that no information is ever lost from the Ordering Table.

When the next Spellout Domain  $D_2$  is Linearized, it is Linearized with respect to the first element of the previously spelled-out SD  $D$ . "First" is defined by the previous application of Linearize.<sup>1</sup> After a Spellout Domain is Linearized, constituents thereof can still move out of the SD, but they cannot change their order. This derives effects such as the Holmberg Generalisation (HG), including aspects of HG that are rarely accounted for.

---

<sup>1</sup> Although it may seem like a lot of extra stipulation that Linearize can tell when something has been spelled out already, the details of Fox & Pesetsky's story are more rigorous than what I have presented here. In particular, the operation Linearize is *always* looking at the "first" and "last" elements of a domain; the difference is that sometimes the domain in question is a single Lexical Item (or possibly morpheme) with a "beginning" and an "end" as expected. See their 2003 handout for details. (A monograph is forthcoming.)

Fox & Pesetsky also derive other types of successive cyclicity. Consider, for example, English object-*wh*-movement. The object *wh*-word in its theta-position is after the verb and subject. For example:



Ordering Table (VP)
what < he
he < read
what < read

Ordering Table (CP)
what < he
he < read
what < read
what < did
what < VP

In the final form of the sentence, *what* is before the verb. If *vP* is an SD, *what* must be before the verb when the *vP* is Linearized. Otherwise, the Ordering Table will store the information that *read* precedes *what*, and then when the next SD is Linearized, the Ordering Table will receive the information that *what* precedes *read*. This contradiction will crash the derivation.

In their 2003 handout, Fox & Pesetsky propose a possible linearisation algorithm that relies on stipulated ordering of heads, specifiers, and complements. A major advantage over Nunes & Uriagereka’s version of the LCA is that simple sisters can be linearised (Cf. section 2.2 below). Nevertheless, Fox & Pesetsky comment: “we suspect that all or most of our proposal could be reformulated if c-command, rather than sisterhood, were the central notion for the Laws of Precedence (as in Kayne 1995).” We will not examine their proposal for laws of precedence in any more detail, as ultimately I will take them up on their suggestion that the central idea of their theory is not dependent on any particular linearisation algorithm. In fact, I will be using the LCA as the ordering algorithm in my proposal.

## 2.2 NUNES & URIAGEREKA

Recall Nunes and Uriagereka’s version of the LCA:

**Linear Correspondence Axiom** (Nunes & Uriagereka 2000): A Lexical Item  $\alpha$  precedes a Lexical Item  $\beta$  iff  $\alpha$  asymmetrically c-commands  $\beta$ .

Clearly, this simplified version of the LCA fails to yield a total ordering on the Lexical Items in many sentences. For example, in a phrase with a complex specifier, there is no asymmetric c-command relation between the elements of the specifier and the sister of the specifier.<sup>2</sup> Nunes and Uriagereka seek to derive such an asymmetric c-command relation using Uriagereka’s notion of Multiple Spellout.

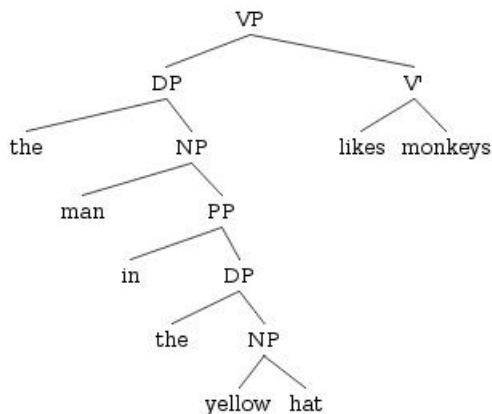
<sup>2</sup> It should be noted that Nunes and Uriagereka are not attempting to solve another problem for bare phrase structure and the LCA: the linearisation of simple sisters. If neither sister is complex, and non-branching nodes are disallowed, no asymmetric c-command relation is possible between them. I have no solution to this problem either, and must put it aside as a matter for further research.

**Multiple Spellout** (Uriagereka 1999): Spellout may occur more than once in the course of a derivation.

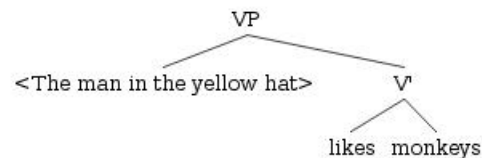
Nunes & Uriagereka propose that Spellout linearises according to the LCA (above). After being spelled out, the structure of the constituent is invisible to syntax. Only the label, which encodes everything the syntax needs to know to manipulate the spelled-out constituent as a single unit, is visible to the derivation. This label, being a terminal node, behaves as a single lexical item. It acts as a “bookmark” to note the location of the spelled-out constituent in the main structure. When PF is generated, the label tells the phonological component where the constituent belongs in the order.

Under this theory, specifiers must be spelled out before merging with the main structure. What is merged is, as far as the syntactic derivation is concerned, only a label, which appears to the derivation as a terminal node. This simple specifier is clearly in an asymmetric c-command relation with the constituents of its sister.

**Figure 1:** Lexical Items of a complex specifier have no c-command relationship with Lexical Items of V'



**Figure 2:** A Spelled-out specifier is a terminal node and is in an asymmetric c-command relation with Lexical Items of V'



### 3. PROPOSAL

Both Fox & Pesetsky and Nunes & Uriagereka propose compelling stories of the mechanisms of the Spellout operation. Fox & Pesetsky are able to account for cyclicity effects and order preservation effects, and simplify and reduce the amount of linearisation calculation that must take place to linearise a complex sentence. Nunes & Uriagereka are able to use the LCA, which *derives* order rather than relying on stipulation. They also account for extraction domain effects such as *wh*-islands and subject islands: since any Spelled-out constituent is inaccessible to the derivation, it is clearly inaccessible for extraction.

It is tempting to claim they are both correct. The problem with this is that Fox & Pesetsky need to be able to access spelled-out constituents to derive ordering effects such as HG, while Nunes & Uriagereka stipulate that this is impossible. A closer look at the problem reveals that Fox & Pesetsky and Nunes & Uriagereka are looking at different domains. Fox and Pesetsky are interested in the “spine” of the tree and its sisters, while Nunes & Uriagereka are looking at the internal structure of the “satellites”. In technical terms, *spine* refers to the

extended projection line of the semantic head (i.e. the verb) of the tree; informally, this is the main trunk of the tree and the heads thereof. The *satellites* are essentially the specifiers and adjuncts: anything but the spine.

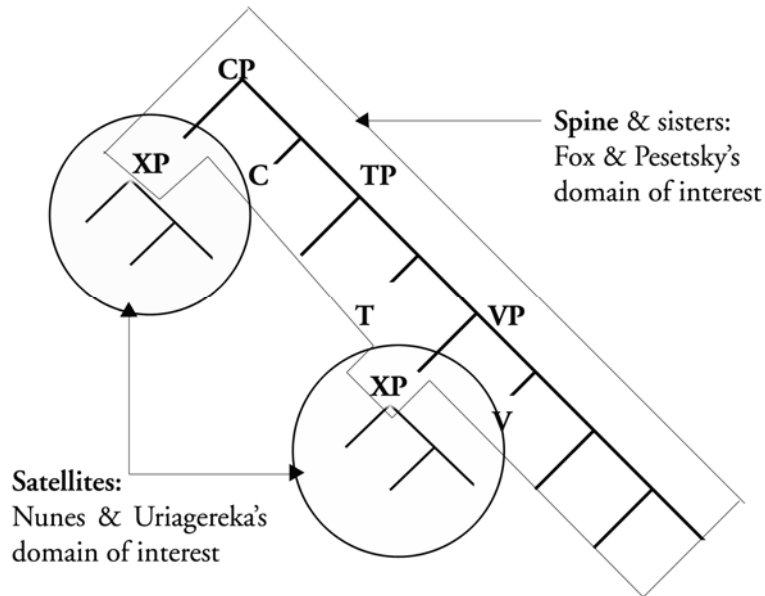


Figure 3: Spine & Satellites

**Claim:** Both conceptions of Spellout are correct but incomplete. Two separate operations apply: LINEARIZE and ATOMIZE.

### 3.1 ATOMIZE

ATOMIZE is essentially Nunes & Uriagereka’s Spellout. It sends the phonological features of the constituent to the phonological component and in so doing renders the constituent inaccessible to the derivation. ATOMIZE applies only when the derivation is “done” with the constituent. The label left behind acts as an atom in the derivation. However, unlike Nunes & Uriagereka’s Spellout, ATOMIZE does not linearise the constituent.<sup>3</sup>

### 3.2 LINEARIZE

LINEARIZE is Fox & Pesetsky’s Spellout/Linearize, with the ordering algorithm specified as Nunes & Uriagereka’s LCA. LINEARIZE takes a “snapshot”<sup>4</sup> of the derivation so far and

<sup>3</sup> Nunes & Uriagereka claim that after Spellout, “there is literally no syntactic object within [the Spelled-out domain]” (N&U 2000 p 25). If this is true, it seems to me that LF receives no structure whatsoever by the end of the derivation. This is clearly impossible: semantics relies heavily on structure. We are left with two possibilities:

- (1) The syntactic structure is still *there*, but simply inaccessible to the syntactic derivation: a closed suitcase that can be moved around but which only LF can open. It is with this model in mind that I have written this paper.
- (2) When a constituent is ATOMIZED it is indeed removed from the derivation, but it goes both to PF and to LF. In other words, there is also LF Spellout. This model is very compelling, but as my research into this is only just beginning, I have chosen to write with option (1) in mind.

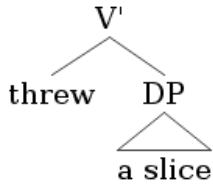
<sup>4</sup> The evocative term “snapshot” comes courtesy of Heather Newell.

applies Nunes & Uriagereka's LCA, storing the resultant ordered pairs on the Ordering Table. LINEARIZE treats an ATOMIZED constituent as a terminal node. This allows satellites to linearise with their sisters.

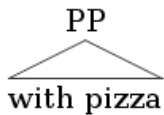
A LINEARIZED constituent is still accessible to the derivation. This solves the problem of accessibility to the spine of the tree. LINEARIZE simply marks the order of Lexical Items.

### 3.3 EXAMPLE: BUILDING THE SENTENCE *THE MONKEY WITH PIZZA THREW A SLICE*.

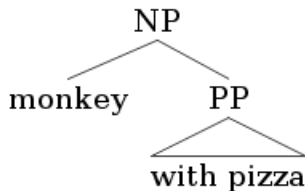
#### Step 1: Merge *threw a slice*



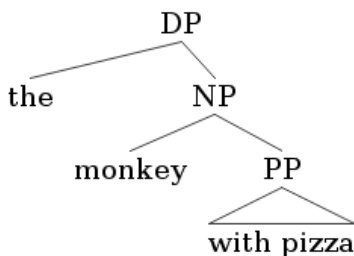
#### Step 2: Meanwhile, Merge *with pizza* in a separate workspace



#### Step 3: Merge *monkey*



#### Step 4: Merge *the*



#### Step 5: LINEARIZE *the monkey with pizza*

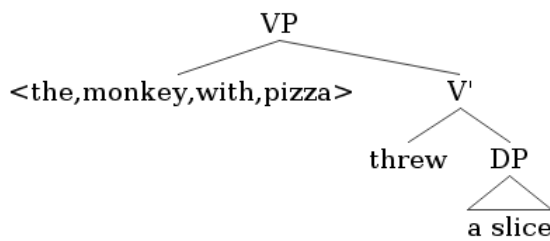
Ordering Table
----------------

with < pizza
monkey < with
monkey < pizza
the < monkey
the < with
the < pizza

**Step 6: ATOMIZE *the monkey with pizza***

The phonological features of *the monkey with pizza* are sent to Phonological Component. Let < **the, monkey, with, pizza** > represent the ATOMIZED constituent.

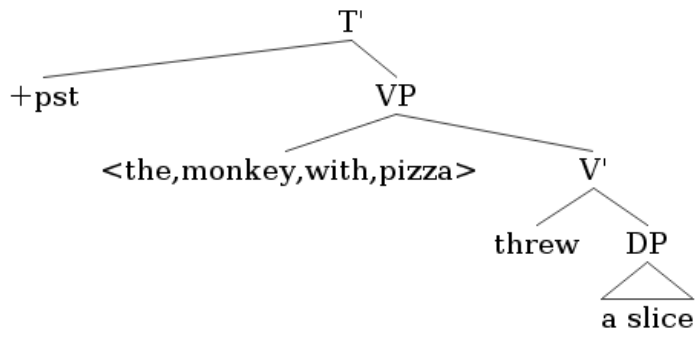
**Step 7: Merge < the, monkey, with, pizza > with *threw a slice***



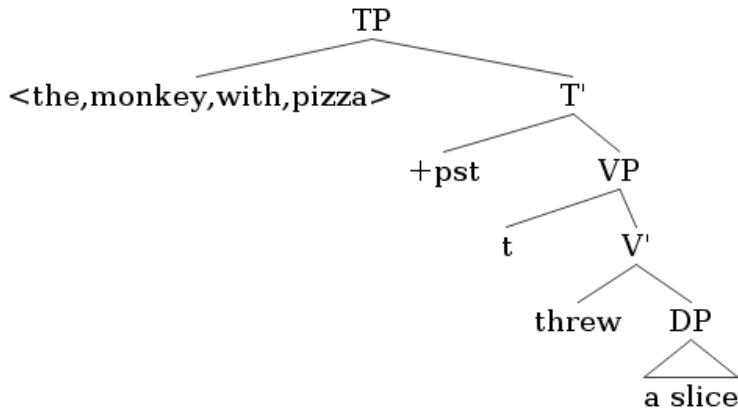
**Step 8: LINEARIZE the VP**

Ordering Table
with < pizza
monkey < with
monkey < pizza
the < monkey
the < with
the < pizza
a < slice
threw < a
threw < slice
the monkey with pizza < threw
the monkey with pizza < a
the monkey with pizza < slice

**Step 9: Merge [+pst] feature**



**Step 10: Move the subject to spec-TP**



**Step 11: LINEARIZE TP**

Ordering Table
with < pizza
monkey < with
monkey < pizza
the < monkey
the < with the < pizza
a < slice
with < pizza
monkey < with
monkey < pizza
the < monkey
the < with the < pizza
a < slice
threw < a
threw < slice
the monkey with pizza < threw
the monkey with pizza < a
the monkey with pizza < slice
+pst < VP = threw a slice
The monkey with pizza < +pst
The monkey with pizza < VP = threw a slice

**Step 12: ATOMIZE whole sentence**

Now we have total ordering on the LIs in the sentence:

The < monkey < with < pizza < +pst < threw < a < slice.

All phonological features are sent to Phonological Component.

Notice also that because *the monkey with pizza* moved before LINEARIZE was applied, the fact that it was once below [+pst] is irrelevant. The phonological component was never told that *the monkey with pizza* was below [+pst].

### 3.4 WH-MOVEMENT

Consider a pair of wh-questions:

- (1) a.  $[_{CP} \text{What}_j \text{ did } [_{TP} \langle \text{the, monkey, with, pizza} \rangle]_i [_{VP} t_i \text{ throw } t_j ]]$   
 b. \*  $[_{CP} \text{What}_j \text{ did } [_{TP} \langle \text{the, monkey, with, } t_j \rangle]_i [_{VP} t_j \text{ throw a slice}]]$

(1b) is ungrammatical because it is not possible to extract anything from an ATOMIZED domain. (1a) is acceptable since VP was not ATOMIZED.

### 4. WHAT DO WE GAIN BY SEPARATING LINEARIZE AND ATOMIZE?

We have seen that separation of LINEARIZE and ATOMIZE is possible. Now we will see why it is advantageous. First, a simple conceptual argument: Spellout seems to be performing several tasks, so perhaps it is more than one operation.

Now let us consider what is predicted when each operation acts without the other.

#### 4.1 LINEARIZE WITHOUT ATOMIZE

A Spellout operation that allows for accessibility after application allows for such phenomena as Object Shift, Quantifier Movement and *wh*-movement to occur across linearisation domains. These are the primary applications in Fox & Pesetsky's 2005 paper on linearisation. LINEARIZE constrains movement across linearisation domains by requiring that any moving elements not change their relative order.

Late Adjunction (Newell 2005, Nissenbaum 2000, Stepanov 2001) may be able to occur after LINEARIZE but before ATOMIZE. This would allow Late Adjunction to occur after Spellout (of a kind) but still allow ATOMIZE to render its domain inaccessible to the derivation. However, this may be a problem for some kinds of Late Adjunction, because it is hard to imagine how the Late-Adjoined element can be assigned a place on the ordering table. A similar situation arises when we consider Lowering (Embick & Noyer 2001, Skinner (in press)) to a Spelled-out domain: a Linearized domain is accessible but presumably not linearisable. This is a topic I will be looking into further in the near future.

#### 4.2 ATOMIZE WITHOUT LINEARIZE

Everything in the above section is really just a consequence of Fox & Pesetsky's conception of Spellout. The following, however, is predicted only by combining Fox & Pesetsky with Nunes & Uriagereka, not by either paper alone.

Normally, we would expect that ATOMIZE can only apply to fully LINEARIZED constituents; otherwise, elements of that constituent would not be on the Ordering Table. Indeed, Kayne's Linear Correspondence Axiom is precisely that his linearisation function is a total ordering on the terminal nodes of the sentence. Suppose, though, that ATOMIZE applies to an UNLINEARIZED domain. Then no information about the order of those constituents will be given to the phonological component. I predict then that *Scrambling* occurs when objects that have not been ordered by LINEARIZE are instead assigned a (random) order by the phonological component.<sup>5</sup>

<sup>5</sup> Or, given that not all grammatical orders in a scrambling language are equal (e.g. Dixon 1972), pragmatics or even sociolinguistic forces may be involved in the choices between grammatical orders.

### 5. SCRAMBLING TYPOLOGY

Because it is possible to separately LINEARIZE and ATOMIZE different parts of the sentence, combinations of these two operations in various sections should derive many types of Scrambling. Consider the following simplified model:

Suppose there are three basic Spellout Domains:

- D** – Satellites (e.g.: DPs)
- V** – Some subpart of the spine (perhaps *vP*)
- S** – The whole sentence

Suppose further that each of these domains can be LINEARIZED or not, and ATOMIZED or not. Call these combinations *Scrambling Types*, as follows:

**D**[±L±A] **V**[±L±A] **S**[±L±A]

For example, Scrambling Type **D**[-L+A] **V**[+L-A] **S**[-L+A] has un-LINEARIZED but ATOMIZED satellites, LINEARIZED but un-ATOMIZED subpart of the spine and un-LINEARIZED but ATOMIZED whole sentence.

Now, in this simplified schema, there are 64 logically possible Scrambling Types:

$$2^2 2^2 2^2 = 64$$

However, three things can make a scrambling type impossible:

1. The final sentence is not ATOMIZED and therefore PF never receives its phonological features.
  2. Linearisation problems occur when satellites (**D**) are un-ATOMIZED but the sentence is LINEARIZED. LINEARIZE requires that the LCA be used, but un-ATOMIZED complex satellites make that impossible.
  3. Not Scrambling: everything is LINEARIZED before being ATOMIZED.
- 1-3 above eliminate numerous types, leaving us with 19 basic types of scrambling:

<b>D</b> [-L+A] <b>V</b> [+L+A] <b>S</b> [+L+A]	<b>D</b> [-L+A] <b>V</b> [-L+A] <b>S</b> [-L+A]
<b>D</b> [+L+A] <b>V</b> [-L+A] <b>S</b> [+L+A]	<b>D</b> [+L-A] <b>V</b> [-L+A] <b>S</b> [-L+A]
<b>D</b> [-L+A] <b>V</b> [-L+A] <b>S</b> [+L+A]	<b>D</b> [-L-A] <b>V</b> [-L+A] <b>S</b> [-L+A]
<b>D</b> [+L-A] <b>V</b> [-L+A] <b>S</b> [+L+A]	<b>D</b> [+L+A] <b>V</b> [+L-A] <b>S</b> [-L+A]
<b>D</b> [-L-A] <b>V</b> [-L+A] <b>S</b> [+L+A]	<b>D</b> [-L+A] <b>V</b> [+L-A] <b>S</b> [-L+A]
<b>D</b> [-L+A] <b>V</b> [+L-A] <b>S</b> [+L+A]	<b>D</b> [+L+A] <b>V</b> [-L-A] <b>S</b> [-L+A]
<b>D</b> [-L+A] <b>V</b> [-L-A] <b>S</b> [+L+A]	<b>D</b> [-L+A] <b>V</b> [-L-A] <b>S</b> [-L+A]
<b>D</b> [+L+A] <b>V</b> [+L+A] <b>S</b> [-L+A]	<b>D</b> [+L-A] <b>V</b> [-L-A] <b>S</b> [-L+A]
<b>D</b> [-L+A] <b>V</b> [+L+A] <b>S</b> [-L+A]	<b>D</b> [-L-A] <b>V</b> [-L-A] <b>S</b> [-L+A]
<b>D</b> [+L+A] <b>V</b> [-L+A] <b>S</b> [-L+A]	

#### 5.1 A SAMPLING OF TYPES

1. **D**[-L+A] **V**[±L±A] **S**[+L+A]  
→ Scrambling within satellites only

2. **D[±L+A] V[-L-A] S[-L+A]**  
 → Scrambling of ATOMIZED satellites within full sentence
  3. **D[-L-A] V[-L-A] S[-L+A]**  
 → Totally free word order
  4. **D[±L+A] V[+L-A] S[-L+A]**  
 → Set relative order for a lower SD (**V**), but these LINEARIZED LIs would be able to Scramble with constituents in the rest of the sentence.  
 → Resultant sentences would have same order for vP-satellites, but everything merged above vP can appear in any order, even intermingled with vP elements, whether they've moved out of vP or not.
- Type (4) above is an interesting prediction this approach makes. Let us examine it a little more closely in figure 2 below.

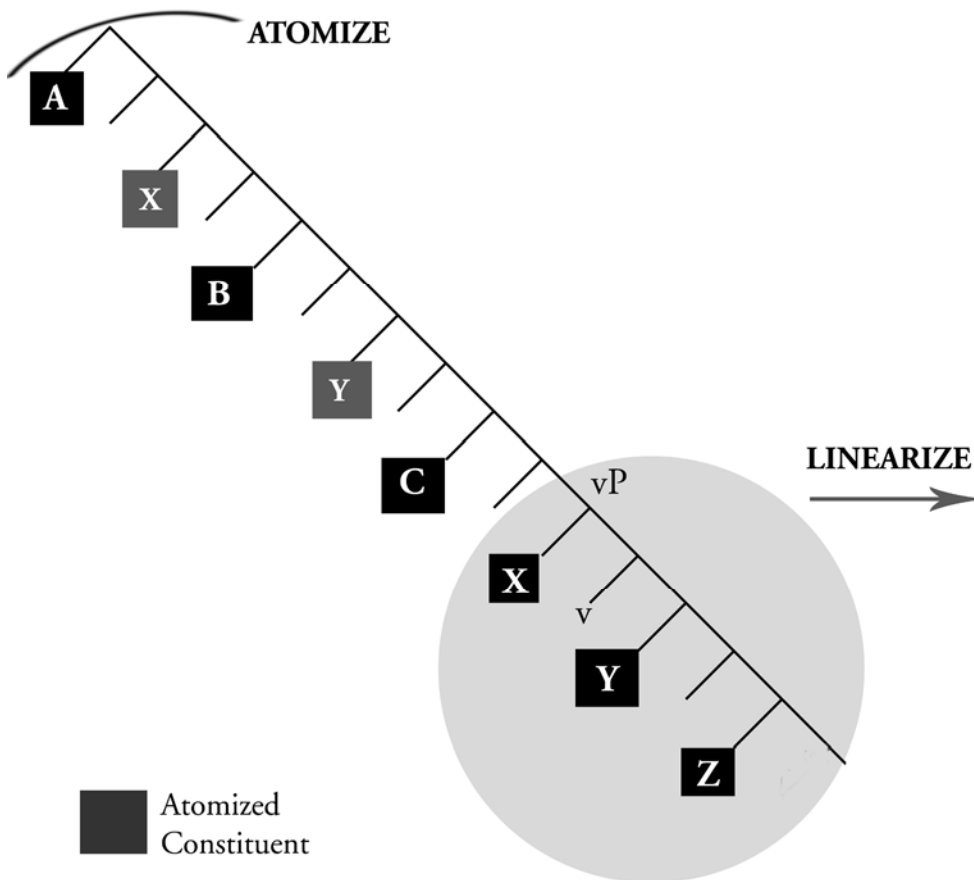


Figure 4

Ordering Table
X < Y
Y < Z
X < Z

LINEARIZATION of  $vP$  determines the order  $X < Y < Z$ . Because the CP is ATOMIZED without LINEARIZATION, A, B, and C have no place on the Ordering Table. Therefore A,B,C,X,Y,Z can order freely as long as  $X < Y < Z$ .

## 6. SCRAMBLING DATA – SOME EXAMPLES

### 6.1 TAGALOG

I propose that Tagalog is of type  $D[\pm L+A]$   $V[-L-A]$   $S[-L+A]$  (Type 2 above): i.e. its ATOMIZED satellites Scramble.<sup>6</sup> Consider the following data:

- (2) a. Nagbigay ng-libro sa-babae ang-lalaki<sup>78</sup>  
       gave GEN-book DAT-woman NOM-man  
       ‘The man gave the woman a book.’
- b. Nagbigay ng-libro ang-lalaki sa-babae  
       gave GEN-book NOM-man DAT-woman
- c. Nagbigay sa-babae ng-libro ang-lalaki  
       gave DAT-woman GEN-book NOM-man
- d. Nagbigay sa-babae ang-lalaki ng-libro  
       gave DAT-woman NOM-man GEN-book
- e. Nagbigay ang-lalaki sa-babae ng-libro  
       gave NOM-man DAT-woman GEN-book
- f. Nagbigay ang-lalaki ng-libro sa-babae  
       gave NOM-man GEN-book DAT-woman

To describe these data, we must leave aside aspects of the oversimplification assumed in the typology in section 5 above. The “full sentence”  $S$  must be slightly less than the full sentence. Perhaps  $S$  is TP, embedded in a CP.<sup>9</sup> This will explain the verb always appearing at the beginning of the sentence.

To build these sentences:

1. DPs are ATOMIZED before being Merged.
2. V moves from its theta-position to somewhere high, say to C
3. TP is then ATOMIZED without being LINEARIZED
4. CP is LINEARIZED and then ATOMIZED

<sup>6</sup> Whether DPs are LINEARIZED before being merged is unclear for this example, as the DPs are single words. How and whether the nouns themselves are linearised (e.g. Distributed Morphology) is beyond the scope of this paper.

<sup>7</sup> Data from Kroeger 1993

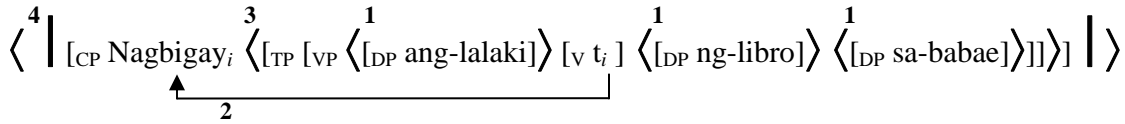
<sup>8</sup> Abbreviations are as follows:

NOM = nominative case

GEN = genitive case

DAT = dative case

<sup>9</sup> This may be the better assumption conceptually as well as empirically. Nissenbaum (2000) for example, describes multiple Spellout in terms of a head sending its complement to PF. This idea provides a trigger for (what I call) ATOMIZE. Intuitively, it also seems reasonable to imagine that LINEARIZE, being merely a “snapshot” of the derivation at a given time, applies to a full domain (e.g.  $vP$ , CP, DP).



$\langle \dots \rangle$     ATOMIZED constituent  
 $\left[ \dots \right]$     LINEARIZED constituent

<b>Ordering Table</b>
Nagbigay < <ang-lalaki, ng-libro, sa-babae>

The result is scrambled satellites with the verb in front.

**6.2 DYIRBAL**

Dyirbal is a non-configurational language. This means it has free word order, syntactically discontinuous expressions, and extensive use of null anaphora (Hale 1983). Hale considered these three properties aspects of a single parameter. If Dyirbal is of type **D[-L-A] V[-L-A] S[-L+A]** (type 3 above), the first two properties are easily explained as being aspects of a single parameter. Consider the following data (Dixon 1972):<sup>101112</sup>

- 3. a. bayi            wangal                    bangul yarangu bulganu banggun  
       the.NOM boomerang.NOM the.GEN man.GEN big.GEN the.ERG  
       dugumbiru buran  
       woman.ERG see.PRES/PAST  
       ‘The woman saw the big man’s boomerang’
  
- b. bayi            yarangu dugumbiru buran            wangal  
       the.NOM man.GEN woman.ERG see.PRES/PAST boomerang.NOM  
       banggun bangul bulganu  
       the.ERG the.GEN big.GEN  
       ‘The woman saw the big man’s boomerang’

Nor are these the only possible orders: Dixon states clearly that all word-orders are grammatical. Example (3b) was chosen because this order was “made up” by Dixon to illustrate dramatic scrambling. He writes:

<sup>10</sup>Orthography is slightly modified to use ASCII characters. *ng* = velar nasal, *r* = “semi-retroflex continuant”, *d* = “lamino-palatal/alveolar” d (Dixon 1972 p. 2).  
<sup>11</sup> *the* in the glosses is my approximation of a more complex, but not relevant, noun-marker system  
<sup>12</sup> Abbreviations are as follows:  
 NOM = nominative case  
 GEN = genitive case  
 ERG = ergative case  
 PRES/PAST = present or past tense

A well-known linguist took exception to this, categorically denied that freedom of word-order of this magnitude was possible in any language, and accused the writer of exaggerating. (321) [3b] was put to informants at the next opportunity, and they castigated the writer for asking a trivial and unnecessary question – “you know that’s alright!” (Dixon 1972 p. 107-8)

Aside from the free word order, the thing to notice is that the noun markers can be separated from the nouns they modify. I propose that this occurs when satellites are not ATOMIZED before being merged to the tree. This is possible if and only if the constituent of which noun and noun marker are a part is ATOMIZED without being LINEARIZED; otherwise the un-ATOMIZED constituent would cause problems for the LCA.

Non-configurational languages are often assumed to have a flat structure for at least part of the sentence (Hale 1983). The advantage of my approach is that the absolute randomness of order can be achieved without assuming very different structures for configurational and non-configurational languages.

## 7. CONCLUSIONS

### 7.1 SUMMARY

Syntactic Spellout is at least two separate operations: LINEARIZE defines the linear order of terminal nodes; ATOMIZE sends phonological features to the phonological component, rendering the ATOMIZED domain inaccessible to the syntactic derivation.

If LINEARIZE applies without ATOMIZE, phenomena such as Holmberg’s Generalisation, *wh*-movement, Lowering, and Late Adjunction are able to occur. LINEARIZE allows movement out of a LINEARIZED constituent, while constraining final order.

If ATOMIZE applies without LINEARIZE, Scrambling occurs within the ATOMIZED constituent. The ATOMIZATION without LINEARIZATION of different constituents accounts for and predicts different scrambling types.

### 7.2 FURTHER RESEARCH

The preliminary nature of this research means there are plenty of unanswered questions; I will list only a few here.

Narrowing the set of possible ATOMIZATION and LINEARIZATION domains would not only make this theory more concrete, but also set a research path for seeking real examples of predicted Scrambling types; conversely, mining Scrambling data may help determine the set of possible ATOMIZATION and LINEARIZATION domains. A major question to examine is whether ATOMIZATION and LINEARIZATION domains are the same domains, and if not whether the domain of one operation is predictable from the domain of the other (e.g. footnote 9). Also of great importance is whether ATOMIZATION and LINEARIZATION domains are parameterised.

Lowering and Late Adjunction are interesting research directions, as both may occur post-Spellout. As LINEARIZE allows accessibility to its domain, this interpretation of Spellout may be of interest. However, the question of ordering the Lowered or Late-Adjoined element is a serious issue, and may even lead to a modification of this entire theory. Also in this domain of research is how these operations might interact with Distributed Morphology (Cf: Skinner, in press).

I would like to examine how it is possible to LINEARIZE simple sisters. Possible solutions are that Bare Phrase Structure is too bare, or that the LCA is not the ordering algorithm LINEARIZE uses.

When do LINEARIZE and ATOMIZE apply? Do they both apply directly after the constituent is merged, or is it possible to wait and apply an operation to a previously merged constituent (Cf: Chomsky 2001)?

What is the relationship between LINEARIZE, ATOMIZE and LF Spellout, if such a thing exists? Are there more than two PF-Spellout operations?

If my proposal is correct, the originally simply-shaped model of syntax is still more complex. Not only are there multiple branches from the derivation to PF, but there are (at least) two different types of branches to PF: one for LINEARIZE and one for ATOMIZE.

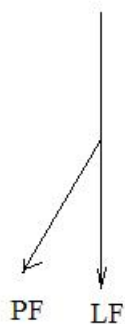


Figure 5: Y-model

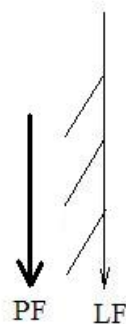


Figure 6: Multiple Spellout

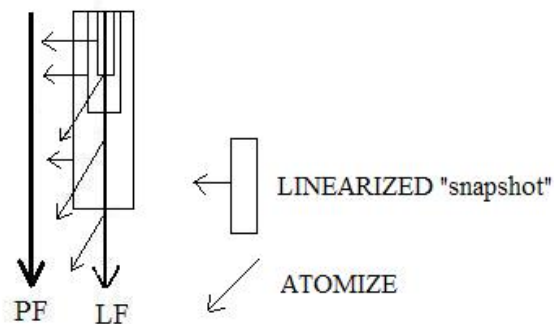


Figure 7: Multiple Multiple Spellout

Different types of information are given to the phonological component at different times. It seems to me that the recipient of the information LINEARIZE and ATOMIZE generate is better thought of as the *phonological component* rather than the *phonological form (PF)*, given how much processing the phonological component is assumed to be doing in order to generate the Phonological Form. If I am correct, the phonological component is processing not only phonological features, but also reinsertion of previously ATOMIZED constituents in the linear order, interpretation of the Ordering Tables, and order-assignment to un-LINEARIZED domains. With further research into how morphology might fit into this model, the distinction between the syntax as a process and PF as a form may continue to weaken. Whether this will prove beneficial or detrimental to the pursuit of a minimalist, computational system of language is only one of many exciting questions to explore.

## REFERENCES

- Chomsky, Noam. 1995. *The Minimalist Program*. Cambridge, MA: MIT Press.
- Chomsky, Noam. 2001. Derivation by Phase. In M. Kenstowicz, ed., *Ken Hale: A Life in Language*, 1-52. Cambridge, MA: MIT Press.
- Dixon, R.M.W. 1972. *The Dyirbal language of North Queensland*. Cambridge: Cambridge University Press.

- Embick, David & Rolf Noyer. 2001. Movement operations after syntax. *Linguistic Inquiry* 32.4: 555-595.
- Fox, Danny and David Pesetsky. 2005. Cyclic linearization of syntactic structure. *Theoretical Linguistics* 31: 1-45.
- Fox, Danny and David Pesetsky. 2003. Cyclic linearization and the typology of movement (handout). MIT, July 2003  
[http://web.mit.edu/linguistics/people/faculty/fox/July\\_19\\_handout.pdf](http://web.mit.edu/linguistics/people/faculty/fox/July_19_handout.pdf).
- Hale, Ken. 1983. Warlpiri and the grammar of non-configurational languages. *Natural Language & Linguistic Theory* 1.1: 5-47.
- Kayne, Richard. 1994. *The Antisymmetry of Syntax*. vol. 25 of *Linguistic Inquiry Monograph*. Cambridge, MA: MIT Press.
- Kroeger, Paul. 1993. *Phrase Structure and Grammatical Relations in Tagalog*. Stanford, CA: CSLI Publications.
- Heather Newell. 2005. Bracketing Paradoxes and Particle Verbs: A Late Adjunction Analysis. In Sylvia Blaho, Luis Vicente & Erik Schoorlemmer (eds.). *Proceedings of Console XIII*. University of Leiden.
- Nissenbaum, Jonathan. 2000. Investigations of covert phrase movement. PhD dissertation, MIT.
- Nunes, Jairo & Juan Uriagereka 2000. Cyclicity and extraction domains. *Syntax* 3.1: 20-43.
- Simpson, Jane. 1983. Aspects of Warlpiri morphology and syntax. PhD dissertation, MIT.
- Skinner, Tobin R. in press. A Post-Spell-out explanation of variation in reduplication. In Eva Dobler & Moti Lieberman, eds., *McGill Working Papers in Linguistics* 22.
- Stepanov, Arthur. 2001. Late adjunction and minimalist phrase structure. *Syntax* 4.2: 94-125.
- Uriagereka, J. 1999. Multiple Spell-out. In S. Epstein and N. Hornstein, eds., *Working Minimalism*, 251-282. Cambridge, MA : MIT Press.