A Computational Model of Language Generation Applied to Japanese *Wh*-questions*

Jason Ginsburg

Center for Language Research, University of Aizu,
Tsuruga, Ikki-machi, Aizu-Wakamatsu 965-8580, Japan
jginsbur@gmail.com

Abstract. This paper discusses a computational model of language generation, based on work in Phase Theory, that attempts to shed light on how the human mind generates sentences. This model presents explicit algorithms that a) determine selection and merger of Lexical Items, b) determine the labels of Merged elements, c) account for movement of Lexical Items within a derivation, and d) account for when chunks of a sentence are sent to Spell-Out. We demonstrate how this model accounts for generation of a *wh*-question in Japanese.

Keywords: Phase Theory, computational modeling, *wh*-questions, Japanese

1 Introduction

Examination of a theory via a computational model can be an enlightening exercise. A computer requires precision. Thus, ambiguious and/or conflicting rules can become immediately apparent, since these cannot be processed by a computer. While we acknowledge that the ability to model a theory on a computer does not necessarily correlate with the correctness of the theory, awareness of problems for a computational implementation of a theory and development of solutions to these problems can lead to a more refined and accurate theory. To this end, we developed a computational model of language generation in which sentences are derived from an underlying set of Lexical Items (LIs), via a process of selection and Merge, following a version of Phase Theory (Chomsky, 1999, 2000, 2004, 2006), the most recent implementation of work in the Minimalist Program (Chomsky, 1995b).

In Phase Theory, a sentence is constructed via a bottom-up process whereby LIs are selected from a numeration, which consists of subnumerations, and Merged together. A derivation is broken up into phases - v* (transitive v) and C (also possibly D) are phase heads. A derivation is subject to the Phase Impenetrability Condition (PIC), which determines when chunks of a derivation are sent to Spell-Out (or to Transfer). Crucially, any elements contained within a chunk of a derivation that has been sent to Spell-Out are not accessible to higher operations. The PIC has been given (at least) two formulations (cf. Grewendorf and Kremers (2009)). In one version (Chomsky 1999, 2000), when v* is reached in (1), the complement of v*, which is VP, is sent to Spell-Out, and when C is reached, C's TP complement is sent to Spell-Out. In another version of the PIC (Chomsky 2004), in (1), when C is reached, VP, which is the complement of v*, is sent to Spell-Out.

(1) $[_{CP}$ C $[_{TP}$ T $[_{vP*}$ v* [VP . . .]]

Within Phase Theory, a head with an uninterpretable feature functions as a probe that Agrees with a goal (if present) that has a matching interpretable feature. Crucially, the probe and goal must be accessible to each other - a probe cannot Agree with a goal that has been sent to Spell-Out.

* I thank Sandiway Fong for helpful discussion of various issues presented in this paper. I also thank the anonymous reviewers of this paper for their helpful comments. All errors are my own.

Various complex issues arise in Phase Theory. First of all, elements must be selected from a numeration. For example, it is assumed that v* selects for V and V selects for DP, but why this is the case is not necessarily clear. Also, issues arise with respect to the relationships between phases, subnumerations, and the timing of Spell-Out. A subnumeration can consist of LIs that form a phase, or it can consist of LIs that form a subject or adjunct, which can have a complex structure and must be formed outside the main spine of a derivation (cf. Johnson (2002)). Under the PIC, a phase edge remains visible to a higher phase. Thus, there is an apparent imperfection - the lack of a one-to-one correspondence between sending an element off to Spell-Out and phase-hood. Furthermore, the PIC creates problems for notions of movement. In (2a), the *wh*-phrase 'what' must move through an intervening v*P phase edge. Similarly, in the Japanese (3a), assuming that the Q particle 'no' undergoes movement (see section 2), it too must move through an intervening v*P phase edge. To deal with this issue, Chomsky (1999) suggests that a phase head can optionally have an EPP feature that attracts a *wh*-phrase. This means that an EPP feature must be present in every phase head that intervenes between the base and scope positions of a *wh*-phrase in constructions such as (2-3a). Thus, v* must have an EPP feature, as shown in (2b-3b), that attracts the *wh*/Q-element to the v*P edge (copies of moved phrases are italicized).

(2) (a) What did you eat?

(b) [$_{CP}$ **What** C$_{[EPP|}$ did you [$_{v*P}$ *what you* v*$_{[EPP|}$ [$_{VP}$ eat *what*]]]

(3) (a) *Taro-ga* *nani-o* *tabe-ta* *no?*
Taro-NOM what-ACC eat-PAST Q
'What did Taro eat?'

(b) [$_{CP}$ Taro-ga [$_{v*P}$ *Taro* [$_{VP}$[nani-o+*no*] tabeta] v*$_{[EPP|}$ *no*] C$_{[EPP|}$+**no**]

However, if a derivation proceeds in phases, and a lower phase is not 'aware' of the contents of a higher phase, then it is not clear how an intervening phase head can 'know' that it requires an EPP feature (cf. Felser (2004), Bošković (2007)). Furthermore, even if v* were to have an EPP feature, the subject in Spec,v*P should eliminate this EPP feature when it is Merged, thus leaving no EPP to attract the *wh*/Q-element.

In order to develop a more refined version of Phase Theory that eliminates the above mentioned imperfections and accounts for how sentences are generated, we created a computer model, implemented in the Python programming language, that applies a set of algorithms in order to generate a complete derivation of a sentence from an underlying numeration. Notably, we provide an explicit model for how LIs are selected from a numeration and Merged into a derivation. We also present a unique view of the timing of Spell-Out of phases in which a) there is a one-to-one correspondence between phase-hood and being sent to Spell-Out, and b) there is no need for optional EPP features that simply exist to bring a phrase to an edge of a phase. We explain how our model works with respect to a 'simple' *wh*-question in Japanese, and we demonstrate how our model leads to some interesting insights into how language is generated by the human mind.

2 Assumptions

In this section, we explain the basic assumptions that we incorporate into our computational model, regarding Lexical Items (LIs), Merge, features, phrase structure, phases, and *wh*-phrases.

A well-formed sentence is built from LIs, where an LI is a bundle of features (cf. Chomsky (1999)) that determine how an LI is interpreted. For example, various features combine to produce the LI 'dog' which has a particular meaning. In our model, we include only those features that are most relevant for formation of a derivation, such as features that undergo agreement relations.

A numeration in our model is a set that consists of unordered LIs, as well as of subnumerations, where a subnumeration is identical to a numeration (it consists of LIs and possibly also further

subnumerations). For example, (4) is a numeration containing the LIs X and Y, as well as another subnumeration that contains LIs and yet another subnumeration.

(4) {X, Y, {Z, W, {P}}}

A subnumeration corresponds to a phase or to a language chunk (subjects or adjuncts) that must be formed outside the main spine of a derivation. A derivation begins by selecting and Merging elements from the most embedded subnumeration; for example, a derivation will begin with the subnumeration containing P in (4) and once that subnumeration is emptied, the derivation will proceed onto the LIs of the higher subnumeration consisting of Z and W, and so on.

A derivation is built from a process of Merging together elements. There are two types of Merge: a) external Merge, in which an element is selected from a subnumeration and Merged with an already formed syntactic object, and b) internal Merge (often referred to as 'movement') whereby an element that is already part of a syntactic object in the derivational workspace leaves a copy in its original position and is Merged again in a new position (Chomsky, 2004, 2006).

Also of importance is how Merge and labels interact. When two elements Merge, the label of one of these elements becomes the label of the newly formed element. For example, when α and β Merge, the label of the newly formed element is either α or β.

We take the position that there are valued and unvalued features.[1] We divide features up as shown in (5), where there is one type of unvalued feature (5a), and there are two types of valued features (5b-c).

(5)

	Feature	Description
(a)	Unvalued	[Feat:_]
(b)	Valued	[+Feat:X]
(c)	Valued	[Feat:X]

An unvalued feature of type (5a) must be valued by a matching valued feature. For example, T has unvalued phi-features '[Phi:_]' that obtain a phi-feature value from a valued '[Phi:X]'. A valued feature of type (5b) must undergo a feature checking relation in which the valued feature assigns an unvalued feature a value. The '+' in the '[+Feat:X]' signifies that the feature must be assigned. For example, we model a theta-role as a feature of this type, represented as '[+Theta:X]'. If a theta-role is not assigned, a derivation will crash; the sentence '*I mailed' is ill-formed because the '[+Theta:X]' of the verb is unassigned. On the other hand, the valued feature in (5c) need not, but can, undergo a feature checking relation. For example, the LI 'dog' may have a feature for animacy (a feature that helps determine the meaning of 'dog') which does not need to undergo a feature valuation relation with anything else in a derivation. In our model, an N comes with valued phi-features of type (5c). These phi-features can value matching unvalued phi-features on T. However, there is no inherent need of the phi-features on N (unlike a theta-role feature on V) to undergo a feature valuation relation.

In our model, we assume that an EPP feature can be a subfeature of a feature, along the lines of Pesetsky and Torrego (2001). Specifically, a '[+Feat:X]' feature (5b) can have an EPP subfeature that forces an element with a matching unvalued feature to Merge directly with the element containing the '[+Feat:X]'. For example, T may contain a '[+Case_EPP:Nom]' feature, which is a valued nominative case feature with an EPP subfeature. A subject contains an unvalued '[Case:_]' feature. Thus, the EPP subfeature forces a subject, assuming that it is base generated at the edge of a v* phrase, to undergo movement (leave a copy in its base position) and Merge again with T.

[1] Our valued features are similar to the interpretable features of the Minimalist Program. In the Minimalist Program, uninterpretable features must be eliminated before a derivation can successfully converge (Chomsky 1995b). In our model, any unvalued features must be valued, but they are not eliminated. It is not clear to us that features need to be eliminated. For example, it may be that when phi-features on T (generally considered to be uninterpretable) are valued, they remain to the level of interpretation, where they may actually aid (or at least not hinder) processing. We leave these issues for further investigation.

We assume a bare phrase structure (Chomsky, 1995a) view of a syntactic tree, as in (6).

(6) $[_{C*}$ C* $[_T$ $[_D[_N$I]] T $[_{v*}$ *$[_D[_N I]]$* v* $[_V[_V$ate] $[_D[_N$dinner]]]]]]

In (6), the label of each phrasal projection is represented by the label of the head of the projection. Furthermore, we represent phase heads with the * symbol; thus, v* and C* are phase heads.

We follow the Phase Theory view that a derivation proceeds in phases, and that phases are sent to Spell-Out as chunks. However, we propose a modified form of the PIC (see section 1) in which a phase is sent to Spell-Out as a complete unit. This view of phases has the advantage of sending an entire phase to Spell-Out at once, rather than requiring differing Spell-Out points for different sections of a phase. For example, in (7), when C* is Merged, the lower v* phase is sent to Spell-Out. Thus, v* and anything contained within it (that has not already been sent to Spell-Out) is accessible to higher operations up until C* is Merged.

(7) $[_{C*}$ C* T $[_{v*}$ v* . . .]]

Furthermore, we propose a notion of Last Resort[2] whereby an element that is contained within a phase that is about to be sent to Spell-Out is reinserted into a higher subnumeration as a Last Resort, to save a derivation. In (8), when C* is Merged, the lower v* phase will be sent to Spell-Out. If α contains an unvalued feature '[Feat:_]', then as soon as C* is Merged, α is reinserted into the higher subnumeration. After the Last Resort operation applies, v* is sent to Spell-Out. The LI α then can be selected and re-Merged into a derivation at a later point, such as at the C* edge.[3]

(8) $[_{C*}\ \alpha_{[Feat:\_]}$ C* T $[_{v*}$ v* V $\alpha_{[Feat:\_]}$]]

We demonstrate how this notion of Last Resort does away with the need for extra EPP features (see section 1) that appear to exist solely for the purpose of bringing a phrase to a phase edge.[4]

Lastly, we explain our assumptions about the structures of *wh*-phrases. Cable (2007), following work by Hagstrom (1998) and Kishimoto (2005), argues (based primarily on an analysis of Tlingit) that in a *wh*-construction, a Q morpheme is Merged with a *wh*-phrase. In *wh*-in-situ languages, Q is adjoined to the *wh*-phrase, whereas in *wh*-movement languages, a *wh*-phrase is the complement of Q. Cable argues that in a *wh*-construction, C attracts Q, and not a *wh*-phrase. In a *wh*-movement language, when Q moves, it brings its complement *wh*-phrase with it. In a *wh*-in-situ language, Q, being an adjunct, separates from its associated *wh*-phrase and raises to C. In our model, we adopt Cable's analysis; in the *wh*-in-situ language of Japanese, Q is Merged with a *wh*-phrase and the label of the *wh*-phrase is the label of the Merged syntactic object (i.e., Q is adjoined to the *wh*-phrase). Since Q does not contain the *wh*-phrase, it is able to move without bringing the associated *wh*-phrase along.

In the next section, we demonstrate how these assumptions are incorporated into our model.

3 Implementation

In this section, we demonstrate how our model successfully creates the derivation of the Japanese *wh*-question (9a). To derive this question, the computer model is fed the numeration in (9b).

(9) (a) *Taro-ga* *nani-o* *tabeta* *no?*
Taro-NOM what-ACC ate Q
'What did Taro eat?'

(b) {C_Int, PAST, {v*, taberu, {Taro}, no, nani}}

[2] I thank Sandiway Fong, with whom I developed this idea. Any problems with this notion of Last Resort are my own.

[3] See Bošković (2007) for a similar proposal, but one that assumes the traditional view of the PIC (e.g., VP is sent to Spell-Out when v* is reached in (1) above).

[4] This model eliminates the 'optional' EPP features of Chomsky (1999), but traditional EPP features, such as the EPP feature on T in languages such as English are incoporated into the model as subfeatures of features (as stated above).

The model automatically runs through each step of the derivation, which it prints to a computer screen in the form of subnumerations, Merged syntactic objects, and syntactic trees. Due to lack of space, we only include some of this output in this paper.

In the numeration (9b),'C_Int' is an interrogative phase head C*, and 'PAST' is a past-tense T head. The numeration contains a subnumeration that corresponds to a lower v* phase. Note that the subject 'Taro' also has its own subnumeration, since we assume that this is required for subjects, which can have a complex structure (see section 1). The computer program automatically converts each input LI into a list that consists of the label of the LI, the form of the LI, and the relevant features of the LI. For example, *taberu* 'eat' is converted to '[V taberu [+Theta:X, TNS:_]]', in which the label is V, the form is 'taberu', and the relevant features that it has are '[+Theta:X]' (a theta feature that must be assigned) and '[TNS:_]' (a tense feature that must be valued). This is an attempt to model representations of LIs, as they are stored by the human mind.

The derivation utilizes the following algorithm to select a subnumeration to work on.

(10) (a) Find the most embedded subnumeration that contains a phase head (C* or v*).
(b) Search for any subnumerations contained within the selected subnumeration.

The Selector (the element of the model that selects elements from a numeration/subnumeration) searches though (9b) and selects the most embedded subnumeration with a phase-head; in this case, '{v*, taberu, {Taro},no, nani}'. Next, the Selector searches for any subnumeration contained within this phase, and it finds the subnumeration {Taro}. Since there is only one element in this subnumeration,[5] 'Taro', this element is selected. Then, in accord with (11), it is reinserted (renumerated) into the higher v* subnumeration.

(11) If a constituent is formed from a subnumeration that is not a phase (i.e., a subject or adjunct), then reinsert the constituent into a higher subnumeration.

Once renumerated, the subject N simply functions as any other LI, and it can be selected again at a later point.

Next, the Selector searches through the v* subnumeration, which has the form in (12), where each LI has been converted into a list consisting of the label, form, and features of the LI. Note that (12) contains the renumerated subject 'Taro', indicated by the '_R' marking on the label 'N'.

(12) Subnumeration: {[Q, no, [Typ:Int, Scp:_, WH:_]], [v*, [+Theta:X, +Case:Acc, Phi:_]], [V, taberu, [+Theta:X, TNS:_]], {[N_R, Taro, [Theta:_, Phi:X, Case:_]]}, [N, nani, [+WH:X, Theta:_, Phi:X, Case:_]]}

The Selector then utilizes the following algorithm to choose an LI to select from the working subnumeration.

(13) (a) If the label of a constituent contains a '[+Feat:X]', search for an LI in a subnumeration with a matching unvalued feature '[Feat:_]'. If present, select this element.
(b) If nothing is selected, then if the '[+Feat:X]' has an EPP subfeature, '[+Feat_EPP:X]', search for an element in the constituent (in the workspace) that has a matching unvalued feature. If present, select this element and leave a copy in the base position.
(c) Otherwise, select an element from a subnumeration according to a default selection order: N > V > v* > T > C*.
(d) If there is only one element left in a subnumeration, then select that element.

If the label of a constituent (the syntactic object in the derivational workspace) has a '[+Feat:X]', then there is a search in the subnumeration for an element with a matching unvalued feature (13a). If present, this element is selected for external Merge. If nothing is selected, then if the '[+Feat:X]' contains an EPP subfeature, there is a search within the constituent in the derivational workspace

[5] We assume that an N does not require a D in Japanese.

for an LI with a matching feature (13b). If found, this LI is selected 'again' for internal Merge, and a copy is left in the original position, resulting in movement. If (13a-b) are unsuccessful (do not result in selection of an LI), then according to (13c), there is a search for an LI in the subnumeration according a default selection order, which we assume is built into the grammar module of the human mind. For example, the Selector will search first for an LI with label N. If not found, then there is a search for a V, etc.[6] Lastly, if nothing is selected, and there is only one element left in a subnumeration, then that element is selected.[7]

Once an element is selected, the algorithm for Merge applies, as follows.

(14) (a) If there is nothing for a selected LI to Merge with, repeat the selection process.
(b) Otherwise, Merge the selected LI with the already formed constituent.

At the initial stage of a derivation, once an element is selected, then the selection process must be repeated to select one more element (14a), so that there will be two elements that can Merge. If there is already a constituent in the derivational workspace, then the selected element is Merged with the constituent. Once Merge occurs, the label of the newly formed constituent is determined as follows.

(15) (a) A Merged element with a '[+Feat:X]' gives the constituent its label.
(b) Otherwise (no Merged element has a '[+Feat:X]'), the label remains the label of the constituent.

According to (15a), if the constituent Y, which is already in the derivational workspace, Merges with the LI X, if X has a '[+Feat:X]' feature, the label of the constituent becomes X, thus resulting in a structure of the form $\{_X$ X Y$\}$ (head-complement structure). On the other hand, also in accord with (15a), if the constituent Y has a '[+Feat:X]', then after Merge with X, the label is that of Y, resulting in a structure of the form $\{_Y$ X Y$\}$ (spec-head structure). Lastly, according to (15b), if the constituent is Y and the selected LI is X, and neither element has a '[+Feat:X]' feature, which could be the case if all '[+Feat:X]' features on the label of Y have been eliminated prior to Merge, then the label remains Y (adjunction structure).

We demonstrate how this selection and Merge process works for the current example. Initially, since nothing is Selected, the Selector searches for an element according to the default selection order (13c), and the N *nani* 'what' is selected. Since only one element has been selected, the selection process is repeated. The selected element N is the label of the constituent, and this label contains a '[+WH:X]' feature. Thus, in accord with (13a), the Selector searches in the subnumeration for an element with a matching unvalued feature. In this case, it finds '[Q no [Typ:Int, Scp:_, WH:_]]', which contains the matching unvalued feature '[WH:_]'. The Q and N Merge, and the label of the constituent remains N, as determined in accord with (15a), since N contains a '[+Feat:X]' feature. The Q element *no* contains a valued '[Typ:Int]' feature that we assume is reponsible for typing a clause as an interrogative when it values a matching '[Typ:_]' feature on C. Q also contains an unvalued '[Scp:_]' feature that must be valued by a matching '[Scp:X]' feature on C, thereby giving an associated *wh*-phrase scope. This feature valuation relation between Q and C occurs at a later point in this derivation.

After Merge, there is a process of feature checking, described in (16).

(16) (a) An unvalued '[Feat:_]' feature is valued by Agreeing with a valued feature '[+Feat:X]' or '[Feat:X]'.
(b) A feature valuation relation can only occur once.

[6] This proposed default selection order is suitable for the current example. However, we note that this proposal most certainly requires further motivation and elaboration, which we leave for future work.

[7] This is why 'Taro' is selected at the initial stage of the derivation of (9b) - it is the only element in its subnumeration.

A '[Feat:_]' and a'[+Feat:X]' can function as probes, when contained within the label of a constituent. In the current case of Merge between N and Q, the '[+WH:X]' feature of the *wh*-phrase probes for and Agrees with the unvalued '[WH:_]' of N, resulting in '[cWH:X]', as shown in the Merged constituent (17), which is automatically produced by the computer model.

(17) {N, [N, nani, [cWH:X, Theta:_, Phi:X, Case:_]], [Q, no, [Typ:Int, Scp:_, cWH:X]]}

The completed v* phrase, shown below in (18), is built as follows. Since the label of the constituent (17), in this case N, no longer contains a '[+Feat:X]' feature, the default selection order applies (13c) and the V *taberu* 'eat' is selected from the subnumeration and Merged. The label of the newly formed object is V, since V has a '[+Feat:X]' feature (15a). Feature checking applies, and in accord with (16a), the '[+Theta:X]' feature of V values the '[Theta:_]' feature of N, resulting in the checked features '[cTheta:X]' on both V and N. Next, the Selector selects v* from the subnumeration, again following the default selection order (13c). The v* contains two '[+Feat:X]' features, '[+Theta:X]' and '[+Case:Acc]', as well as an unvalued '[Phi:_]' feature. Merge results in a syntactic object with the label v*, in accord with (15a). Feature checking results in the '[Phi:_]' of v* being valued by the '[Phi:X]' of N, and the unvalued Case feature '[Case:_]' of N being valued by the '[+Case:X]' of v*. Lastly, the renumerated N 'Taro' is selected in accord with (13a), since the label of the v* constituent contains a '[+Theta:X]' feature and 'Taro' contains a matching unvalued '[Theta:_]' feature. After Merge, the label remains v*, in accord with (15a), since v* contains a '[+Feat:X]'. Feature checking occurs, and the '[Theta:_]' feature of N is valued, as shown in (18).

(18) {v*, {v*, [v*, [cTheta:X, cCase:Acc, cPhi:X]], {V, [V, taberu, [cTheta:X, TNS:_]], {N, [N, nani, [cWH:X, cTheta:X, cPhi:X, cCase:Acc]], [Q, no, [Typ:Int, Scp:_, cWH:X]]}}}, {N, Taro, [cTheta:X, Phi:X, Case:_]}}

At this point, the current subnumeration has been emptied and the Selector works on the higher subnumeration, from (9b), of the C* phase, shown in (19).

(19) {[T, PAST, [+Case_EPP:Nom, Phi:_, Force:_, +TNS:PST]], [C*, [Typ:_, +Scp_EPP:X, +Force:X]]}

Selection and Merge of T and C* results in the Merged constituent in (20). First, T is selected according to the default selection order (13c). T Merges with the v* phrase, and the label of the newly formed phrase is T, in accord with (15a), since T contains the feature '[+Case_EPP:Nom]'. The '[+TNS:PST]' feature of T values the '[TNS:_]' of V and the '[Phi:_]' feature of T is valued by the '[Phi:X]' of the subject 'Taro'. Next, note the presence of the EPP subfeature associated with nominative case on T, '[+Case_EPP:Nom]'.[8] This feature Agrees with the unvalued '[Case:_]' feature of the subject 'Taro', and the EPP subfeature forces the subject to move, in accord with (13b); a copy of 'Taro' is left in the v* edge and 'Taro' is re-Merged at the T edge, where its case feature is valued and the EPP subfeature is eliminated. Next, the Selector finds and selects C* from the subnumeration, in accord with the default selection order (13c). Note that in (20), the features of C* have not yet undergone feature checking relations.

(20) {C*, [C*, [Typ:_, +Scp_EPP:X, +Force:X]], {T, {T, [T, PAST, [cCase:Nom, cPhi:X, cTNS:PST, Force:_]], {v*, {v*, [v*, [cTheta:X, cCase:Acc, cPhi:X]], {V, [V, taberu, [cTheta:X, cTNS:PST]], {N, [N, nani, [cWH:X, cTheta:X, cPhi:X, cCase:Acc]], [Q, no, [Typ:Int, Scp:_, cWH:X]]}}}, {N(Copy), Taro, [cTheta:X, cPhi:X, Case:_]}}}, {N, Taro, [cTheta:X, cPhi:X, cCase:Nom]}}}

As discussed in section 2, we assume that when a phase head is Merged, a lower phase, if present, is sent to Spell-Out. Thus, as soon as the phase head C* is Merged (20), the lower v* phase is sent to Spell-Out. The Spell-Out algorithm applies as follows.

[8] We assume that there is an EPP feature on T in Japanese (cf. Miyagawa (2001)).

(21) (a) Last Resort Check: If an LI with an unvalued feature is contained within a Spell-Out domain, reinsert this LI into the current working subnumeration.

(b) Crash Check: If there are any unvalued features '[Feat:_]' or any unassigned valued features '[+Feat:X]' in the Spell-Out domain, a derivation crashes.

(c) Linearize the Spell-Out domain and apply phonological rules.

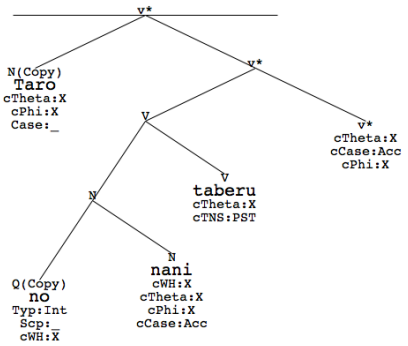
Initially, the Last Resort Check operation (21a) applies. Since the Q element *no*, which is contained within the Spell-Out domain of the v* phase, contains an unvalued '[Scp:_]' feature (see (20)), Q undergoes a Last Resort operation of reinsertion into the current subnumeration, and a copy is left in the base position. Because Q originates in an adjoined position, it undergoes the Last Resort operation without bringing along its associated *wh*-phrase. Crucially, no optional EPP feature is required to bring Q to the v* phase edge. At this point, there is a Crash Check operation (21b), and since there are no unvalued or unassigned features in the v* phrase (copies of moved LIs are invisible to this operation), the lower v* phrase is linearized (21c). We utilize the following linearization algorithm for Japanese.

(22) (a) First Merge: A label LI is Merged to the right when the label LI Merges for the first time.

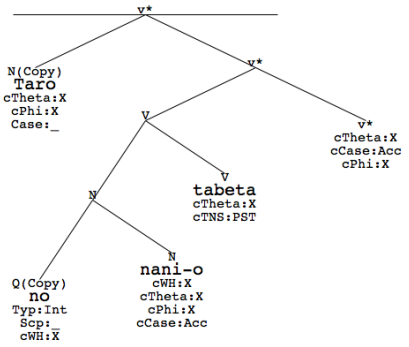
(b) Second Merge: When an LI with label N is Merged to an element that has previously undergone Merge, the N is Merged to the left. Otherwise, the LI is Merged to the right.

With the exception of second Merge of an N to a constituent, Merge is to the right.[9] The computer model linearizes the lower v* phrase as shown in (23a-b), which are trees that are automatically constructed by the computer model.[10]

(23) (a)



(b)



(23a) shows the resulting v* phrase immediately after it is linearized and (23b) shows the v* phrase after phonological rules have applied, resulting in *nani* 'what' taking on the accusative case particle 'o' and the verb *taberu* 'eat' taking on the past tense form *tabeta* 'ate'. Note that the trees contain copies of the subject 'Taro' (which has been re-Merged at the T edge) and of the Q particle 'no' (which has been reinserted into the higher subnumeration). The line through the v* edge signifies that the v* phrase has been linearized.[11]

Once linearization of the v* phase is complete, the derivation continues, resulting in (24). We assume that C* has two '[+Feat:X]' features; a '[+Force:X]' feature and a '[+Scp_EPP:X]' feature. The Force feature, a modified version of Force argued for by Rizzi (1997), is associated with determining whether or not a clause is matrix or embedded. The '[+Scp]' feature is associated

[9] This algorithm most likely requires further refinement with respect to Second Merge.

[10] The model produces one tree for each step of linearization. Due to lack of space we do not include all of the trees.

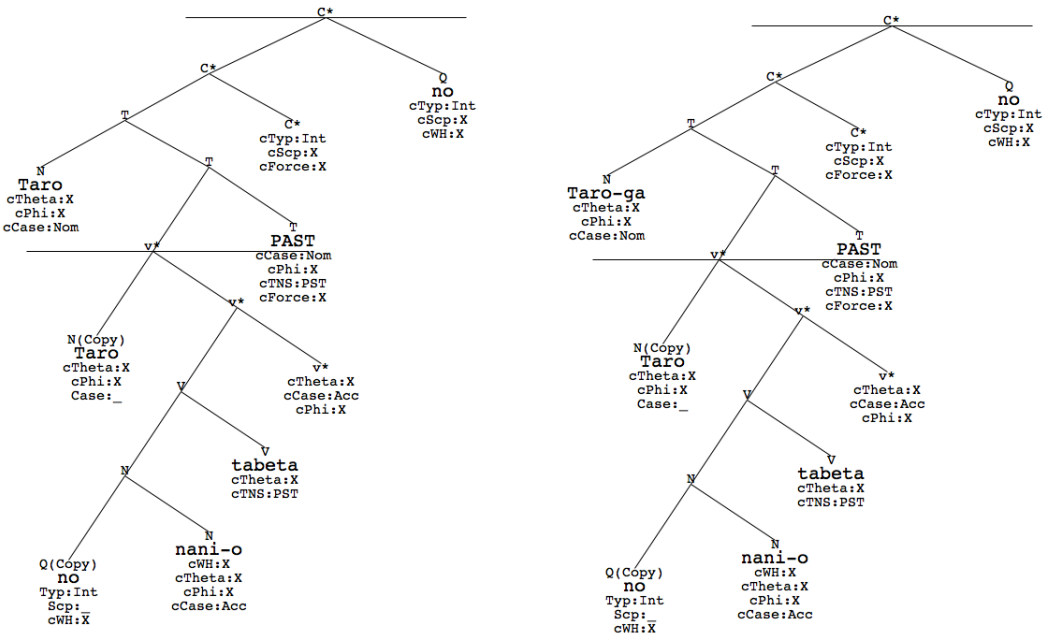
[11] At this point, the complete v* phase chunk is essentially set aside in a form in which it can be pronounced once the derivation is completed.

with giving an element, such as a *wh*-phrase, scope. C* also contains an unvalued '[Typ:_]' feature that is responsible for determining clause type. The current subnumeration now contains the renumerated Q element, which is selected by the Selector, since it contains an unvalued '[Scp:_]' feature that needs to be valued by the '[+Scp_EPP:X]' of C*, in accord with (13a). The Q element *no* Merges with C. Then feature checking occurs. The '[+Scp_EPP:X]' of C values the '[Scp:_]' feature of *no* 'Q'. We assume that this feature on C* contains an EPP subfeature[12] that is eliminated since *no* 'Q' is Merged locally. Furthermore, C* contains an unvalued '[Typ:_]' feature that is valued by the '[Typ:Int]' feature of Q, thereby typing the clause as an interrogative. Note that the resulting Merged object (24) contains the linearized v* phrase, represented as a list.

(24) {C*, {C*, [C*, [cTyp:Int, cScp:X, cForce:X]], {T, {T, [T, PAST, [cCase:Nom, cPhi:X, cTNS:PST, cForce:X]], [v*, [N(Copy)_AdjNd, Taro, [cTheta:X, cPhi:X, Case:_]], [v*, [V, [N, [Q(Copy), no, [Typ:Int, Scp:_, cWH:X]], [N, nani-o, [cWH:X, cTheta:X, cPhi:X, cCase:Acc]]], [V, tabeta, [cTheta:X, cTNS:PST]]], [v*, [cTheta:X, cCase:Acc, cPhi:X]]]]}, {N, Taro, [cTheta:X, cPhi:X, cCase:Nom]}}}, [Q, no, [cTyp:Int, cScp:X, cWH:X]]}

Since the entire numeration has been emptied, the Spell-Out operation (21) applies. There is a Last Resort Check operation (nothing is found in the Spell-Out domain that contains an unvalued feature), and then there is a Crash Check operation (no unvalued or unassigned features are present). The periphery of the clause is linearized (25a) and phonological rules apply (25b), resulting in addition of the nominative case particle *ga* to the subject 'Taro'.

(25) (a) (b)



4 Conclusion

We have demonstrated how our computational model succesfully derives the syntactic representation of the Japanese *wh*-question (9a). We leave for further work discussion of how this model can be applied to a variety of examples in Japanese and other languages. A simple English *wh*-question can be modeled in a manner very similar to (9a), Whereas in Japanese, Q alone moves,

[12] Note that it is not crucial to this derivation for the '[+SCP:_]' feature on C* to have an EPP subfeature. However, if there is an EPP feature associated with T, then in constructions with a subject *wh*-phrase, the subject must move to the T edge, in which case the Last Resort process cannot apply to Q. Movement of Q from a subject *wh*-phrase to C can be accounted for straightforwardly if an interrogative C contains an EPP (notably, not an optional EPP feature) that attracts Q. Due to lack of space, we leave aside the details of this type of construction.

in English, a null Q carries its associated *wh*-phrase with it. Furthermore, various refinements to the model are required to account for the complex set of facts concerning multiple *wh*-questions and island effects in Japanese and other languages.

In conclusion, there are several ways in which our model helps to clarify our understanding of how language is generated, from the perspective of Phase Theory. This model a) provides a specific algorithm for selecting elements from a numeration/subnumeration, b) provides a specific algorithm for determining the label of a Merged syntactic object, c) incorporates a one-to-one correspondence between phase-hood and the timing of Spell-Out, and d) eliminates the need for 'optional' EPP features - via a Last Resort process. This model thus resolves some troublesome issues that arise in Phase Theory, and makes progress towards discovering precisely how the human mind generates sentences.

References

Bošković, Ž. 2007. On the Locality and Motivation of Move and Agree: An Even More Minimal Theory. *Linguistic Inquiry*, 38, 589-644.

Cable, S. 2007. *The Grammar of Q: Q-Particles and the Nature of Wh-Fronting, as Revealed by the Wh-questions of Tlingit*. Ph.D. thesis, Massachusetts Institute of Technology.

Chomsky, N. 1981. *Lectures on Government and Binding*. Dordrecht: Forix.

Chomsky, N. 1995a. Bare Phrase Structure. In H. Campos and P. Kempchinsky, eds., *Evolution and Revolution in Linguistic Theory*, pp. 51-109. Washington, DC: Georgetown University Press.

Chomsky, N. 1995b. *The Minimalist Program*. Cambridge: MIT Press.

Chomsky, N. 1999. Derivation by phase. *MIT Working Papers in Linguistics: MIT Occasional Papers in Linguistics Number 18*.

Chomsky, N. 2000. Minimalist Inquiries: The Framework. In R. Martin, D. Michaels and J. Uriagereka, eds., *Step by Step*, pp. 89-155. Cambridge: MIT Press.

Chomsky, N. 2004. Beyond explanatory adequacy. In A. Belletti, ed., *Structures and Beyond: The Cartography of Syntactic Structures, Volume 3*, pp. 104-131. Oxford: Oxford University Press.

Chomsky, N. 2006. On Phases. In R. Freidin, C. Otero and M. Zubizaretta, eds., *Foundational Issues in Linguistic Theory; Essays in Honor of Jean-Roger Vergnaud*, pp. 133-166. Cambridge: MIT Press.

Felser, C. 2004. *Wh*-copying, Phases, and Successive Cyclicity. *Lingua*, 114, 543-574.

Grewendorf, G. and J. Kremers. 2009. Phases and Cyclicity: Some Problems with Phase Theory. *The Linguistic Review*, 26, 385-430.

Hagstrom, P. 1998. *Decomposing Questions*. Ph.D. thesis, Massachusetts Institute of Technology.

Johnson, K. 2002. Towards an Etiology of Adjunct Islands. Ms., University of Massachusetts at Amherst.

Kishimoto, H. 2005. *Wh*-In-Situ and Movement in Sinhala Questions. *Natural Language and Linguistic Theory*, 23, 1-51.

Miyagawa, S. 2001. EPP, Scrambling and Wh-in-situ. In M. Kenstowicz, ed., *Ken Hale: A life in Language*, pp. 293-338. Cambridge: MIT Press.

Pesetsky, D. and E. Torrego. 2001. T-to-C Movement: Causes and Consequences In M. Kenstowicz, ed., *Ken Hale: A Life in Language*, pp. 355-426. Cambridge, MA: MIT Press.

Rizzi, L. 1997. The Fine Structure of the Left Periphery. In L. Haegeman, ed., *Elements of Grammar*, pp. 281-337. Elements of grammar. Dordrecht: Kluwer.