Concatenation, $-Theory, and Phonosyntax[1]

Junichi Iida

I propose *phonosyntax* as part of the phonological component, in which phonological strings are recursively merged to build the input for prosody. The proposal is motivated by merge-sensitive phonological processes, including a recursive accent rule and an opaque segmental process, and by a preceding theory of a highly autonomous semantics. I put forth formal definitions of two phonological processes that exemplify this merge-based subcomponent of phonology, namely, concatenation and $-propagation. The former accounts for joining of two phonological strings, a phenomenon found in virtually all languages. The latter accounts for stress assignment of Tokyo Japanese. As a consequence of these proposals, I suggest from both phonological and semantic perspectives that some aspects of what has been called syntax may be better understood as part of phonology.

1. Introduction

The syntactic component of the grammar outlined in *Proposing* describes the process of sentence formation as consecutive merges of syntactic nodes. At the beginning of generation, a sequence of syntactic nodes are assumed. Each node is given, along with its phonological properties, syntactic labels that determine whether a given pair of adjacent nodes can merge. Merging nodes are converted to a new node with new phonological properties and new syntactic labels. The syntactic component of sentence generation completes when all syntactic nodes are merged into one node.

In *Semantics*, I proposed that the semantic component of grammar has a separate base structure from syntax, and lexical items bridge the two structures. Crucially, under this model a lexical item maps a linear string of semantic nodes onto phonological and syntactic properties, without referring to the hierarchical structure of semantics. From two separate linear structures, the grammar generates two separate trees, one for syntax and the other for semantics.

> “*A lexical item is a basic unit of syntax. It is the interface between sound* (*phonology, which then connects to phonetics*) *and meaning* (*semantics*)*. Lexical items are inserted to the base structure.* [...] *Lexical items do not care about the hierarchical structure. Lexical items target an unimpeded partial linear sequence instead of a constituent.*
> (Semantics, p. stardust-surrealism)

The bistratal base model is a natural consequence of earlier observations. Some sort of binary branching tree structure had been evident even before Kyouro had a formal framework of semantics. A good example is $-theory (*Proposing*, p. scholarship). Details aside, $-theory in essence claims that

[1] An official document of Kyouro/教育ローマ字; Version 7.1.3, updated to June 2026. (Original: Version 7.1.0, released in May 2026.)

there exists some feature $ such that the leftmost AP of a constituent containing $ is stressed. It is a constituent-based phonological rule. That is, we already had a tree-based grammar even without a formal framework of semantics. When some particular structure of a sentence is supposed from specifically phonological evidence, there is no reason to believe that semantic evidence supports the exact same structure for the same sentence.[2] As expected, upon closely observing sentence meanings, it is apparent that the tree structure inferred from semantics are often different from that of phonology. (The following is one of such instances.)

Semantics helps us make some discrepancies between syntax as described in *Proposing* and semantics explicit. An easy example is found in the system of what is known as particles of Japanese.[3] According to *Proposing*, the sentence *yama ni wa modotta* ("returned specifically to the mountain") will be analyzed as $\overline{\overline{\overline{yama\ ni}\ wa}\ modotta}$[4] and *yama o kowashita* ("broke the mountain") as $\overline{\overline{yama\ o}\ kowashita}$. In *Semantics*, this *ni* would be described as $\overbrace{to}^{ni}$, *wa* as $\overbrace{X\colon \frac{X}{X'}}^{wa}$, and *o* as $\overbrace{{}_{on}\quad X\colon \frac{X}{X', X''}}^{o}$. This gives *ni wa* as ${}_{to}\quad X\colon \frac{X}{X'}$. The lowered *on* and *to* are objects called *roles*. $X\colon \frac{X}{X', X''}$ and $X\colon \frac{X}{X'}$ are called *parallelizers*. The overbrace denotes lexical mapping relations. A semantic structure x corresponding to a phonological structure y is denoted as $\overbrace{x}^{y}$. Observe that *o* and *ni wa* have analogous structures. They have a role on the left and a parallelizer on the right. In $\overline{\overline{\overline{yama\ ni}\ wa}\ modotta}$, both in syntax and semantics, *ni* (in semantics, it is ${}_{to}$) merges with *yama*. However, in $\overline{\overline{yama\ o}\ kowashita}$, while *o* is a constituent in syntax, it is not a constituent in semantics because the role merges leftward, analogously to $\overline{\overline{\overline{yama\ ni}\ wa}\ modotta}$. In fact, their distributions alone could hint at this. If we suppose that *ni wa* and *o* occupy similar positions, the fact that *ni* merges leftward suggests that, in some representation not purely phonological, a subpart of *o* may merge leftward, leaving the remaining part behind. (Another interesting example of phonology-semantics discrepancy is on p. surrealism of *Semantics*.)

If semantics does not share the same hierarchical structure as syntax, it naturally follows that syntax may be more phonological than we initially thought. The purpose of this manuscript is twofold. First, I will redefine some aspects of our syntactic component as part of *phonosyntax*, a subdivision of phonology, within our grammatical framework. I will discuss what phonosyntax concerns and where it is placed in the process of sentence generation relative to other components of grammar. Second, I will establish two phonological operations that take place in phonosyntax as formal rules, namely,

[2] It is a common observation that semantics and phonology point to different sentence structures. Some of such discrepancies are traditionally handled by movement. I expect the bistratal base model will be able to offer an alternative interpretation of some of the known phenomena commonly considered as evidence of movement, and, ultimately, ease the tension between semantics and syntax that the literature has long been aware of.

[3] In Kyouro, *particle* is not a defined term.

[4] An overline indicates constituency of two overt nodes. In other words, if a sequence of letters has an overline above, it has two overt nodes that merge.

concatenation and $-propagation. The former is a simple concatenation rule where two strings are unified into one. The latter is a reinterpretation of trimming, a syntax-phonology mapping rule partially outlined in *Proposing*. With phonosyntax and some of its internal operations defined, I expect that the present work offers a new outlook on a theory of language.

2. Phonosyntax as a Component of Grammar

The phonological component is subdivided into the phonosyntactic component and the prosodic component. Phonosyntax is a process of building the input of prosody through consecutive merges of phonological objects and accompanying merge-sensitive phonological operations. Prosodic operations are merge-insensitive. It concerns building the prosodic structure. As segments must align with the syllabic template, some segmental operations belong to the prosodic component. The output of prosody is given to phonetics.

The base structure of a sentence is parallel linear structures of phonosyntactic nodes and semantic nodes.[5] On each side, the nodes on the base are the terminal nodes of the sentence. The phonological specifications of a terminal node are at least partially supplied from the lexicon. The specifications contain sufficient information for grammar to generate the correct surface representation (= output of phonology), including its suprasegmental and segmental characteristics. But a terminal node does not necessarily have a corresponding, notationally identical formal expression in the lexicon. As it is often assumed in the phonological literature, I assume that the lexicon is reasonably economical and sufficiently informative. But what exact data are saved in the lexicon in what exact format remains an open question. I will note all relevant phonological information available for each given node, regardless of what part of such information is supplied by lexicon or given by later rules, without worrying about minimizing lexical specifications.

Phonosyntax begins from the base. When two of the nodes merge, it generates a new node. This process of consecutive merges is essentially the same process of what has been described as syntax in *Proposing*. As stated earlier, labels are used to determine which nodes merge next. Upon completion of a merge, the input nodes are no longer available for the grammar. The tree structure we assume is therefore a procedural representation of sentence generation.[6] When the root node is obtained, that is the only node available.

[5] Multiple views are possible as to how this bistratal structure is obtained. One view is that the semantic base is constructed first and lexical phonological forms are inserted to unimpeded partial linear sequences of the semantic base matching their lexical specifications. Under this scenario, if the semantic base cannot be matched to phonological forms without any surpluses or overlaps, the sentence cannot be externalized, while internally the thought may be complete. Another view is that lexical items are laid out linearly to construct both semantic and phonosyntactic bases simultaneously. Under this scenario, a semantic node cannot appear without an already matching phonosyntactic node, and wellformedness is the only constraint of externalization. A scenario where the phonological base is laid out first and then lexicon searches for matching semantic nodes, though unintuitive to the present author, is also conceivable. None of them interferes with our lexically bridged bistratal base model within the scope of this manuscript.

[6] In the context of a formal grammar, a procedural account of sentence generation does not in itself entail any neurological claims. Two grammars are notational variants if they make the same predictions about wellformedness. Similarly, a theory of language is distinguished from another by what predictions about wellformedness a grammar can make under those theories. When a grammar describing a language spoken by a particular individual predicts that a certain sentence can be generated, it entails the claim that her mental grammar can generate the same sentence, without committing to how this particular mental grammar is implemented in her brain.

Since, as elaborated in Section 8, the term syntax is reserved for the central component of the grammar, henceforth I will use the adjective *combinatory* to discuss the properties of a phonosyntactic node responsible for merging order or merge-compatibility of two nodes.

Merge is a function that takes two nodes and generates a new one. In phonosyntax, merge generates a new string from two input strings. Concatenation and $-propagation are part of merge. Among other phonological *subroutines* of merge, accent switching and one of the segmental operations are exemplified in Section 7. The subroutines are applied sequentially, and if a subroutine faces a node or nodes that do not meet the input conditions, the nodes will remain intact and be evaluated for the next subroutine.

3. Notations

We reuse the function notation of *Semantics*. It is a notational convention of unnamed functions. The input of a function is placed on the left side of a colon, and the output on the right side. A function f(x) = y is written as *x : y*. An ordered pair is simply denoted with a comma. *a, b* is an ordered pair (a, b), and *a, b : x, y* is a function that takes an ordered pair (a, b) and returns a new ordered pair (x, y).

A subroutine takes one or two phonosyntactic nodes and generates one or two. Each phonosyntactic node has both phonological and combinatory specifications. Hence, some subroutines may refer to or modify combinatory properties. However, since the two subroutines to be formalized in the present work, concatenation and $-propagation, do not need combinatory properties, we simplify the notation by pretending that the input and the output of all functions are either a phonological string or an ordered pair of phonological strings. We do not bother to write an input or output as a full node containing both phonological and combinatory properties.

An AP boundary is represented with a bar ("|"). The letters *X*, *Y*, and *Z* are reserved for sequences of zero or more consecutive symbols; if the same letter is used multiple times in a rule, on all instances it represents the same sequence.

The notation specifies the conditions for the process to be applied to the input. A function *a, b : x, y* requires that the input is an ordered pair for the process of generating *x, y* to be applied. Similarly, a function *X|Y : x* requires that the input is a string containing an AP boundary for generating *x*.

Phonosyntactic trees are written in the same tree notation used in *Proposing*. A parent node is placed above its daughters, shifted to the right from the left daughter. In a tree diagram, the function notation is used in place of the resulting node. If the terminal nodes are X and Y and they merge to generate Z, it is represented as the following:

```
    X, Y : Z
X        Y
```

We only note instances of function applications relevant to the discussion in the context. When we are interested in the resulting form of a merge and not internal subroutines of it, we only note the final result of the merge.

4. Concatenation

Concatenation as a subroutine is defined as *X, Y : XY*, which reads "*take an ordered pair of two strings, and return a new string generated by concatenating them without changing the order.*"

Since concatenation unifies the input, later subroutines of the same merge are likely those that take one string as the input. (Otherwise, there will be a subroutine that split the input.) Conversely, subroutines that precede concatenation are likely those that take two strings and generate two strings.

A sample derivation of $\overline{\textit{yama o}}\,\overline{\textit{mita}}$ ("saw the mountain") is as follows:

```
          |yamao, |mita : |yamao|mita
     |yama, o : |yamao      |mi, ta : |mita
|yama       o               |mi      ta
```

5. $-Propagation

$-propagation is a subroutine where $ assigned to the input string is copied to the first AP of the string. An AP with $ is indicated by placing $ right after an AP boundary. The sequence "|$" will be interpreted in prosody as the left edge of a stressed AP.[7] With this notation, the subroutine is defined as *|X$Y : |$X$Y*, which reads "*take a single string with | at the beginning and $, and return a new string generated by inserting $ right after the first |*." It is also possible to achieve the effectively same prediction under the assumption that this subroutine takes two strings, in which case the function notations should look like *|X, Y$Z : |$X, Y$Z* if the output is two strings.[8] The present definition of $-propagation seems to be the simplest option among those that I considered here.

This function adds one $ at the top of an AP if the input satisfies the conditions, and when a sentence involves multiple merges this can create annoying sequences of multiple $'s. Since, as far as I am aware, the prosodic component does not need to distinguish different numbers of consecutive $'s, for simplicity, I adopt the convention to write a sequence of consecutive $'s as one $. For example, "|$," "|$ $," and "|$$$" are all written as "|$."

[7] Under the prosodic model of *Proposing* and *Version Seven*, an AP is stressed iff it is the first AP of an IP. That is, if the left edge of an AP coincides with an IP boundary, the AP is stressed.

[8] *|X, Y$Z : |$XY$Z* if the output is a single string. Defining $-propagation in this way will eliminate concatenation as a separate process. Therefore such a definition can be considered as a composite notation of the two functions. It seems much more natural to see them as separate subroutines; nearly all or maybe truly all languages seem to have concatenation while whether any non-Japanese language has a subroutine similar to $-propagation is an interesting question.

As a natural consequence of defining \$-propagation as a function of one string, I assume that it follows concatenation, which generates a single string from two.

A sample derivation of $\overline{\textit{yama o}\ \overline{\textit{\$ mita}}}$ is as follows:

```
         |yamao|$mita : |$yamao|$mita
     |yamao                    |$mita
|yama       o             |$mi         ta
```

As it copies \$ only to the first AP, it replicates the prediction of \$-theory. For example, if only the first AP has \$ on the base, no other APs will receive \$, which is exemplified below:

```
         |$yamao|mita
     |$yamao                   |mita
|$yama          o         |mi      ta
```

\$-theory predicts that in a structure of a three APs $\overline{|X \quad \overline{|Y \quad |Z}}$, the rightmost AP being stressed implies that all APs on the left are also stressed while a leftward-merging AP, such as $|Y$ in $\overline{\overline{|X \quad |Y} \quad |Z}$, may evade stress even when an AP on the right is stressed. An example of the former structure is $\overline{\textit{yumi ga}}\ \overline{\overline{\textit{dōnatsu o}}\ \overline{\textit{tabemashita}}}$ ("exactly Yumi ate exactly doughnuts"). If this literal is pronounced with *tabemashita* stressed and *dōnatsu o* unstressed, it will be illformed. These predictions are replicated by the present definition of \$-propagation. Observe that when \$ is placed on *tabemashita*, it propagates to all APs:

```
         |yumiga|$dōnatsuo|$tabemashita : |$yumiga|$dōnatsuo|$tabemashita
                          |dōnatsuo|$tabemashita : |$dōnatsuo|$tabemashita
     |yumiga              |dōnatsuo      |$tabemashita
|yumi       ga       |dōnatsu  o    |$tabe      imashita
```

An example of the latter structure is $\overline{\overline{\textit{oishī dōnatsu o}}\ \textit{tabemashita}}$ ("ate exactly doughnuts which are delicious"). Unlike the previous example, this literal can be pronounced with *tabemashita* stressed and *dōnatsu o* unstressed, and it can be wellformed. Observe that when \$ is placed only on *tabemashita*, the middle AP *dōnatsu o* evades stress:

```
                |oishī|dōnatsuo|$tabemashita : |$oishī|dōnatsuo|$tabemashita
            |oishī|dōnatsuo
        |oishī|dōnatsu
    |oishī                  |$tabemashita
|oishi  i   |dōnatsu  o   |$tabe     imashita
```

6. Tokens With Multiple APs

I suggested in *Proposing* that lexical specifications are necessary to determine stress assignment of multi-AP tokens,[9] but the problem of how such a specification should be made was put aside.

> “*When the first AP of a multiple-AP word has R, whether the remaining AP's have R is not completely predictable from the positions of $. When the word* nintokutennō (*Emperor Nintoku*) *receives R, it is given only to the first AP* nintoku *while for the word* tōkyōdizunīsī (*Tokyo Disney Sea*) *R is given to each of the two AP's. It seems that the different distribution patterns of R for these words need to be characterized lexically. Some common expressions that seem to come in both forms interchangeable* (*at least for some speakers*)*, such as* arigatōgozaimasu*, may receive the same treatment: the different forms are the results of different lexical properties, not of different information structures.*
>
> (Proposing, p. semantician)

In *Version Seven*, I proposed a partial solution. Phonological-$ is placed on a certain AP per *$-span*, including when the specified AP is one of multiple APs of the same lexical item, and that the AP that receives phonological-$ is specified in the lexicon as an *$-slot*:

[9] Our stress is defined in terms of contour tones: R, Lv, and F. These contour tones are originally defined in Kodama (2008), and in *Proposing* (p. bunnies), a modified definition is given to R. The definition of stress is given in *Version Seven*: “A syllable is stressed iff it bears its AP's final R.” (p. 1).

“ *We assume that the “slot” of phonological-$ is specified (if any) as one of the APs of the lexical item in question. We call this slot the* $-slot *of the lexical item. A token may have up-to one $-slot.*

We assume that internal APs of an multi-AP token belong to the phonological tree of the sentence as terminal nodes.
(Version Seven, p. 1)

“ *The linear sequence of the terminal nodes of a phonological tree is a linear sequence of a whole number of* $-spans.

An AP of a phonological tree belongs to exactly one $-span.

An AP without a $-slot belongs to the same $-span *as the first (structurally closest) AP with a $-slot it merges with.*

A $-span has exactly one $-slot.
(Version Seven, p. 2)

In the context of the present phonosyntactic architecture, these hypotheses about $-slot and multi-AP tokens imply that in phonosyntax a multi-AP token is treated as multiple terminal nodes, with each terminal node having at most one AP. An inter-token structure $\overline{x \quad y}$ may be phonosyntactically $\overline{\overline{X \quad X'} \quad Y}$ or $\overline{X \quad \overline{Y \quad Y'}}$, for example, where $\overline{X \quad X'}$ belongs to x and $\overline{Y \quad Y'}$ to y.

AP-sensitive phonosyntax is motivated by $-theory. Stress has been observed to be sensitive to both inter-token combinatory relations and AP boundaries. *Trimming* (*Proposing*, pp. ruth-sanctuary) was supposed in part for this reason as a process of converting an inter-token combinatory structure into a purely phonological structure. However, once we accept the idea that the token combinatory mechanics and phonology overlap, i.e., some aspects of what has been known as syntax are better understood as part of phonology, it is natural to think that trimming may be an epiphenomenon of a merge-sensitive phonological rule that is also sensitive to APs. Further, if one AP is the max size of a phonosyntactic terminal node, an AP-sensitive subroutine can be naturally conceived, as it is then guaranteed that all AP boundaries appear as an edge of an input string of a subroutine at least once during the derivation, assuming that merge occurs at least once.

I argue that internal merging relations of a multi-AP token is part of its lexical specification. Under this assumption, our $-propagation rule naturally replicates the effect of trimming. Consider a token $\overline{|X \quad \overline{|Y \quad |{}^\wedge\mathrm{Z}}}$, where the substrings X, Y, and Z do not contain an AP boundary and the $-slot is indicated by the circumflex. If $|{}^\wedge Z$, marked as a $-slot, receives $, $-propagation predicts that, in addition to $|{}^\wedge\mathrm{Z}$ itself, $|X$ and $|Y$ will also have $. On the other hand, a token $\overline{\overline{|X \quad |Y} \quad |{}^\wedge\mathrm{Z}}$ with all else being equal will only have $ on $|{}^\wedge\mathrm{Z}$ and $|X$.

With the present model of AP-sensitive phonosyntax, the aforementioned problem of stress configurations of multi-AP items are automatically explained. This includes the problem of *nintoku tennō* and *tōkyō dizunīshī* raised in *Proposing*. They are specified in the lexicon as |^ *nintoku* |*tennō* and |*tōkyō* |^ *dizunīshī*. APs in an $-span that evade stress are also naturally explained in the same fashion. Trivially, APs on the right side of the $-slot will not receive stress. Similarly, $-theory predicts that APs that merge leftwards will not receive stress unless $ is assigned there on the base.

7. Empirical Motivations of Phonosyntax

Within the Kyouro project, the basic motivation behind phonosyntax is most concisely expressed in Chapter Three of *Proposing*:

> “*Some phonological processes are applied to a local syntactic structure. Accent switching occurs recursively as suffixes merge recursively with the base* (»*globalist*)*. Accessing local structures is also necessary to determine the positions of R* (*Chapter Six*)*. These facts suggest that the precedence of syntax over phonology only holds for each subprocess and it is not the case that everything in syntax completes before phonology begins.*
>
> (Proposing, p. easter)

That is, some phonological operations seem to be merge-sensitive. An example of a merge-sensitive phonological operation is accent switching. It is demonstrated by the derivation of a Japanese verb form *warawaserareru* (“will be made laugh”), described in p. glucose, where on each merge until *warawaserare* is obtained, the same accent switching rule is applied, reproduced below:[10]

waraˌw·ˈasėˌr·ˈaˈrėˈr·u
waraˌw·ˈasėˌr·ˈarėˈ
waraˌw·ˈasėˈ

warawˈ	s·ˈasėˌ	r·ˈarėˌ	ˈ(_)r·u
“laugh”	“make”	“be done”	“will”

Other examples include some opaque segmental operations. It is well-observed that the system of verbal inflections of Japanese shows alternations between *t* (anterior coronal stop) and *ch* (posterior coronal affricate):

ka<u>t</u>eba (supposedly, *kat* + *reba*) — if (one) wins

ka<u>ch</u>imasu (supposedly, *kat* + *imasu*) — will win.

A phonologist may hypothesize that *t* becomes *ch* before *i*. But the sequence *ti* occurs on the surface, e.g., *insentibu* (“incentive”). One may point out that the sequence *ti* occurs mostly or entirely in loan words. While it may be tempting to ascribe lexical strata in phonology, for this particular case it will be

---

[10] Kyouro spelling is used to describe accent switching. While Kyouro is not a pure phonological notation, it is a fairly faithful notation of the underlying phonological representation of accent of Tokyo Japanese. See p. ebb of *Proposing* for a relevant discussion.

simpler to state that the rule turning *t* into *ch* is merge-sensitive. Lexical strata may allow a greater number of phonological features left underspecified in the lexicon.[11] But the strata specifications themselves are added phonological information, and filling feature values based on lexical strata requires additional rules.

A simple solution to account for these observations is to assume that merge is a phonosyntactic operation. Isolating some phonological operations as phonosyntactic operations eliminates the necessity for additional theoretical arsenals or lexical burdens explaining why certain rules are only applicable to certain grammatical contexts.

8. Issues of Terminology

It might be argued that what is here called phonosyntax is simply a relabeling of (part of) syntax, and that the operations proposed are not genuinely phonological. Such an objection is not unexpected since merge and concatenation are conventionally not considered as part of phonology (e.g., Scheer 2016). But the core claims around phonosyntax are not just about nomenclature or taxonomy. They are substantive architectural claims. In this section I emphasize that the proposed terminological choice is not arbitrary but is rather grounded on principled ideas of what separates phonology and syntax.

The proposed phonosyntactic component does not entail any arguments for eliminating the central system that feeds both semantics and phonology, and it is not about simply calling this central system by a new name, either. The proposed phonosyntactic operations and other descriptions laid out in Section 7 in sum entail the claim that phonology needs a tree structure resembling what is commonly supposed in syntax. This claim is purely about phonology, and it is not new nor surprising. SPE, for example, relies on the syntactic structure to describe stress rules and some segmental rules of English. Later phonological literature likewise supposes that phonology relies on the syntactic tree, whether it refers to a structure isomorphic to syntax or instead operates on a different structure obtained through mapping rules (Elordieta 2008). In later cases, e.g., Nespor & Vogel (2007), phonology notationally and substantively contains a hierarchical structure derived yet distinct from syntax. In sum, it is uncontroversial that certain empirical phonological facts across languages suggest some non-flat structure. After all, T-model expects that the syntactic tree serves not just LF but also PF. The present proposal is for treating the phonology-serving hierarchical structure within phonology proper, without necessarily eliminating or relabeling the central system.

[11] An argument for lexical strata on the basis of underspecification is generally not justifiable in phonology. In general, the fact that one group of lexical items can be isolated from the rest on the basis of common patterns seen in the group that allow us to underspecify some features does not make any predictions about wellformedness of a sentence. Consider a hypothetical language where exactly six items are known to be in the lexicon: ["pap", "kak", "dim", "gor", "dos", "yap"]. If we isolate the first two items, "pap" and "kak," we can reduce their phonological specifications since they commonly have form of CaC, where the onset and the coda are identical. The only specification needed to distinguish one from the other is for the consonants: if they are "k," then "kak," and "pap" otherwise. Let us say that this group is a stratum and call it "natives." The rest of the lexicon will be "loans." As a result, each item of the natives is specified for just two binary features, one for specifying the stratum, and the other for specifying the consonants. Superficially, it may look like a nice reduction of lexical phonological specifications. But this does not mean that in this language "kos" will be illformed. If "kos" is introduced to or is newly found in the lexicon, it will be one of the "loans," as the phonological shape deviates from the natives. Information reduction as described here can be made independently of descriptive facts about the grammar of a language. How the maximal reduction can be achieved depends purely on the mathematical state of the given lexicon. Such a technique is better called a data compression technique and it does not concern the grammar of a language.

I do not claim that everything that contributes to pronunciation should be part of phonology, either. There is a legitimate reason for considering a tree structure that contributes to pronunciation as part of syntax if the same tree serves semantics. That is, if the rules operate on a structure change both meaning and sound, even if not always simultaneously, it is reasonable to call the structure by a third name and to position it in a third component, that is, syntax. On the contrary, if a structure and the rules that operate on it serve only sound, then they must be phonology[12] (otherwise the terminology would lose common grounds). Our grammatical architecture and the phonosyntactic rules are defined in such a way that the phonosyntactic rules never refer to or modify semantic objects. In this light, placing phonosyntax in phonology proper is a trivial consequence.

In short, proposing phonosyntax and placing it in phonology are a consequence of a trite conclusion that has been expressed repeatedly in the literature: phonology relies on trees. In terms of T-model, we are on the PF branch. It is not a terminological move against "syntax." Perhaps the only true novelty of the present architectural decision made in this manuscript is that it takes evidence of phonology's reliance on a tree structure at face value and puts the tree that phonology refers to within phonology itself.

This train of thoughts depends on the earlier claim that semantics does not refer to the same tree that phonology refers to. One may suspect that the claimed "early branching," i.e., semantics and phonology diverge earlier than conventionally assumed, is a forced decoupling argued only for itself. This is a legitimate concern as one might claim early branching without substance. If you duplicate a file and subsequently apply the same edits on both files, the result will be no different from if you duplicated it after those edits, and claiming that the files branched early makes no predictions about the end result. Similarly, it will be considered a forced decoupling if the same set of operations takes place on both PF and LF. In the present architecture, early branching is substantive. I suppose the fact that our semantics and phonology do not share operations should be evident from the description made in *Semantics.*

As a side note, I would like to verbalize some intuitions on why early branching is a rather natural architectural choice of a framework. In describing a language within any framework of any theory, we face two kinds of puzzles. One concerns the sound patterns of the language, and the other the patterns of meaning. These puzzles, however, are not independent. The pairing of sound and meaning seen in language is not random. Therefore some linking mechanism is necessary, and this becomes a third puzzle. This linking mechanism is traditionally called syntax and lexicon. Here we have a choice of the general direction of the architectural design: loose coupling or tight coupling. A loosely coupled system is where components are less reliant on each other. In such a system, a modification made on one component is relatively less likely to require a modification on other components. In a tightly coupled system, on the other hand, a change in one component relatively more likely disrupts another. In a

[12] Or phonetics, but obviously we are not discussing phonetics.

tightly coupled grammar, phonology and semantics would share a greater amount of the derivational history. In a loosely coupled grammar, a smaller number of derivational steps would be shared. Fixing one component, or more generally, describing components, is thus intuitively expected to be easier with an early branching architecture. If we want to reconcile phonology and semantics, it is natural to lean towards early branching because we would want to avoid modifying what we have already described just to compensate changes made on other parts of the grammar.

9. Conclusion

I proposed that syntactic merge described in *Proposing* is better understood as part of a phonological subcomponent called phonosyntax. This hypothesis is based on some earlier considerations of descriptive phonological facts and a preceding theory of a highly autonomous semantic component that does not rely on the same hierarchical structure that the phonological component refers to. Based on this phonosyntactic hypothesis, I defined two subprocesses of phonosyntax: concatenation and $-propagation. The former is a simple process of unifying two phonological strings that is extremely commonly observed (and is rather taken for granted) in human languages. The latter replaces an earlier proposal of a syntax-phonology mapping rule explaining stress configurations and its apparent sensitivity to syntactic structures observed in Tokyo Japanese.

A theoretical implication of phonosyntax is a potentially smaller syntax. If merge is part of phonology, phonology and semantics may part ways earlier than previously thought during the course of sentence generation. Syntax as the sound-meaning interface or the core component of language may be shorter without merge.

The motivations to relocate the borderline between syntax and phonology are rather trivial ones. Phonology seems to need the familiar tree structure, particularly, merge, for itself while semantics does not need the same structure. Therefore what has been considered syntactic merge in *Proposing* should now be considered phonology. If it is phonology, then syntax will be small.

Within the Kyouro project, it is probably more important that we now have a partially formalized expression of phonosyntax and that we have replaced the earlier description of trimming. The train of thought I laid out here is mostly a restatement of the simple intuition that part of phonology accompanies syntax, as expressed in Chapter Three of *Proposing* (quoted in Section 7). The replacement of trimming highlights a key feature of our revised framework: no more syntax-phonology mapping puzzles because they are phonology throughout.

Finally, I would like to leave a comment on concatenation. Concatenation as a phonological operation is often taken for granted, and in Kyouro as well it has been implicit until today. I suspect part of the reason why it is often left unexplained is because a tree structure is obtained by splitting a phonological string, and merge can be intuitively seen as the reverse of branching. It is easy to think that if you get two things by breaking one, you will get the original one by putting them together. But phonological concatenation is a crucial part of the nature of human language. It is imperative for any serious theory of human language to give an explicit status to concatenation and discuss its ontological standing within the model.

References

Chomsky, N. & Halle, M.	1968	The Sound Pattern of English. Harper & Row, Publishers.
Elordieta, G.	2008	An Overview of Theories of the Syntax-Phonology Interface. Anuario Del Seminario De Filología Vasca "Julio De Urquijo" 42 (1):209-86. https://doi.org/10.1387/asju.2309.
Iida, J.	2026	Proposing a New Romanization System of Japanese, ver. 6.5.4. https://nihongotopics.github.io/Kyouro/Proposing a new romanization system of Japanese (book).pdf.
	2026	Semantics of Tokyo Japanese, ver. 6.5.3. https://nihongotopics.github.io/Kyouro/Semantics of Tokyo Japanese.pdf.
	2026	Version Seven, ver. 7.0.7. https://github.com/NihongoTopics/Kyouro/blob/master/Version Seven.pdf.
Kodama, N.	2008	曲線声調と日本語韻律構造. 『ありあけ』熊本大学言語学論集 巻 7, p. 1-40. https://kumadai.repo.nii.ac.jp/records/25636.
Nespor, M. & Vogel, I.	2007	Prosodic Phonology: With a new foreword. Mouton de Gruyter.
Scheer, T.	2016	WHY PHONOLOGY IS FLAT: THE ROLE OF CONCATENATION AND LINEARITY. http://tscheer.free.fr/papers/Hdt%20Scheer%20Cambridge%202-16.pdf