A principled derivation of Harmonic Grammar

Anonymous Author

Affiliation

email

Abstract
The HG implementation of constraint-based phonology is shown to follow axiomatically from the substantive assumption that phonological grammars respect part/whole structure.

In constraint-based phonology, the best surface realization of an underlying form is the one with the smallest vector of constraint violations. **How should constraint violation vectors be ordered to select the smallest?** The recent debate on OT versus HG (e.g. Pater 2016) tackles a special case of this architectural question. Usually, this literature adjudicates the merits of an ordering (e.g., OT’s lexicographic ordering) by testing it on a test case. Yet, the predictions of a specific class of orderings on a specific test case depend on the choice of a specific constraint set. The conclusions reached can thus often be overturned by adopting a different constraint set (e.g. Kawahara 2006).

A more principled approach is thus called for. I propose to start from broad formal properties that a grammar should satisfy to qualify as natural language phonology. And to deduce axiomatically from these desiderata suitable classes of orderings for constraint violation vectors. If this axiomatic deduction holds independently of the choice of a constraint set, we will have circumvented the problem that the constraint set is unknown and can only be figured out once the ordering is settled.

This paper illustrates this research strategy. Section 1 sets the background, by spelling out a broad framework for constraint-based phonology. Sections 2-3 introduce the desideratum that a phonological grammar respect part/whole phonological structure. Sections 4-5 then show that HG can be derived axiomatically from this desideratum, plus the (unmotivated but formally natural) restriction to utility/disharmony-based orders.

1 Constraint-based grammars (CBGs)
We assume a base set $\mathcal{X}$ of underlying forms. We assume next that each underlying form $x$ in this base $\mathcal{X}$ comes with a set $\mathcal{Y}(x)$ of candidate surface realizations. This **representational framework** $(\mathcal{X}, \mathcal{Y})$ makes available a set of candidate phonological mappings $(x, y)$ each consisting of an underlying form $x \in \mathcal{X}$ and a corresponding surface realization $y \in \mathcal{Y}(x)$. A constraint $C$ assigns to each such phonological mapping $(x, y)$ a number $C(x, y)$. This number $C(x, y)$ is interpreted as a count of some problematic, undesirable phonological structure (offending clusters, mismatching corresponding segments, etcetera). This number $C(x, y)$ is thus assumed to be a non-negative integer. This **constraint integrality assumption** will play a crucial role in Section 5. We assume we have a set $C$ consisting of a finite number $n$ of constraints $C_1, \ldots, C_n$.

This constraint set $C$ thus represents a mapping $(x, y)$ as an $n$-dimensional **constraint violation vector** $C(x, y) = (C_1(x, y), \ldots, C_n(x, y))$. Indeed, the core idea here is that phonological mappings are made of discrete objects (strings, trees, etcetera) and therefore come with little mathematical structure. To circumvent this problem, we represent them as vectors of numbers and exploit the rich structure defined on numbers (Haussler 1999).

For instance, numbers can be ordered based on their size. This ordering can be extended from single numbers to vectors in many different ways. Thus, let $\prec$ be some order defined among $n$-dimensional vectors. The inequality $a \prec b$ says that the vector $a$ is smaller than the vector $b$. The **constraint-based grammar** (CBG) corresponding to this order $\prec$ is the function $G_\prec = G_{\mathcal{X}, \mathcal{Y}, C}$ that realizes each underlying form $x$ in the base $\mathcal{X}$ as the surface form $y = G_\prec(x)$ in the candidate set $\mathcal{Y}(x)$ with the smallest constraint violation vector $C(x, y)$, i.e. $C(x, y) \preceq C(x, z)$ for every other candidate $z$ in $\mathcal{Y}(x)$. Effectively, through the representation of phonological mappings as constraint violation vectors, the order $\prec$ among vectors defines an order among candidates and the CBG $G_\prec$ is the minimum relative to this candidate order.
To illustrate, consider $S \subseteq \{1, \ldots, n\}$ and for any vectors $a = (a_1, \ldots, a_n)$ and $b = (b_1, \ldots, b_n)$, let
\[
a \preceq_S b \text{ iff } a_k \leq b_k \text{ for every } k \in S \tag{1}\]
This relation $\preceq$ is a partial order among $n$-dimensional vectors. Let the base set $\mathcal{X}$, the candidacy function $\mathcal{Y}$, and the candidate set $\mathcal{C}$ be those of the Basic Syllable System (Prince and Smolensky 1993/2004), recalled in (2).
\[
\mathcal{X} = \{\text{CV}, /\text{CVC}, /\text{V}, /\text{NC}\}\]
\[
\mathcal{Y}(x) = \{\text{ONSET, [CVC], [V], [NC]}\}
\mathcal{C} = \left\{ \begin{array}{c}
C_1 = \text{ONSET, } C_2 = \text{CODA, } C_3 = \text{DEP, } C_4 = \text{MAX} \end{array} \right\} \tag{2}
\]
The CBG $G_{\mathcal{X}, \mathcal{Y}, \mathcal{C}}$ corresponding to the set $S = \{C_1, C_2\}$ maps all underlying forms to [CV].

2 Preserving part/whole structure: the case of string concatenation

Strings and part/whole structure

Phonological representations often have part/whole structure. One source of part/whole structure (we will look at another source below in section 3) is some operation $\oplus$ that combines smaller representations into a larger one. To illustrate, in this section we assume that the underlying and surface forms made available by $(\mathcal{X}, \mathcal{Y})$ are strings of segments combined through the operation $\oplus$ of string concatenation.

We consider two underlying strings $x'$ and $x''$ in the base $\mathcal{X}$ such that their concatenation $x' \oplus x''$ belongs to the base $\mathcal{X}$ as well. We assume that the candidate set $\mathcal{Y}(x' \oplus x'')$ of the underlying concatenation $x' \oplus x''$ coincides with the set $\mathcal{Y}(x') \oplus \mathcal{Y}(x'')$ of the concatenations $y' \oplus y''$ of surface strings $y'$ and $y''$ in the candidate sets $\mathcal{Y}(x')$ and $\mathcal{Y}(x'')$.

To illustrate, the underlying string /adta/ can be construed as the concatenation $x' \oplus x''$ in (3) of the underlying strings $x' = /\text{adt}a/ \text{ and } x'' = /\text{d}a/$.
\[
\frac{x' \oplus x''}{\text{adta}} = \text{ad}ta \oplus \text{d}a \tag{3}
\]
We assume that candidates are constructed by systematically modifying obstructing voicing, as in (4).
\[
\mathcal{Y}(/\text{adta}/) = \{ \text{attata, [attata] [attata] [attata]} \}
\mathcal{Y}(/\text{adta}/) = \{ \text{[adta][adta][adta][adta]} \}
\mathcal{Y}(/\text{da}/) = \{ \text{[da] [ta]} \} \tag{4}
\]
These candidate sets (4) satisfy the candidate concatenativity condition $\mathcal{Y}(x' \oplus x'') = \mathcal{Y}(x') \oplus \mathcal{Y}(x'')$.

Preserving part/whole structure

We consider a CBG $G_{\preceq} = G_{\preceq}^{\mathcal{Y}, \mathcal{C}}$ corresponding to some constraint set $\mathcal{C}$ and some vector order $\preceq$. We say that $G_{\preceq}$ respects the part/whole structure induced by the concatenation $x' \oplus x''$ provided it satisfies condition (5). The surface realization $G_{\preceq}(x' \oplus x'')$ of the concatenation of the underlying strings $x'$ and $x''$ is the concatenation of their surface realizations $G_{\preceq}(x')$ and $G_{\preceq}(x'')$. Equivalently, the two operations of string concatenation and surface realization commute.
\[
G_{\preceq}(x' \oplus x'') = G_{\preceq}(x') \oplus G_{\preceq}(x'') \tag{5}
\]
To illustrate, (5) requires the underlying concatenation $x' \oplus x'' = /\text{adta}/ \text{ and } x'' = /\text{d}a/$ as in (6).
\[
G_{\preceq}(/\text{adta}/) = G_{\preceq}(/\text{adta}/) \oplus G_{\preceq}(/\text{d}a/) \tag{6}
\]

Independent sub-strings

“Nothing happens” at the juncture of the concatenation $x' \oplus x''$ of the two underlying strings $x' = /\text{adta}/ \text{ and } x'' = /\text{d}a/$: no relevant phonological structures are created or destroyed by the concatenation. We thus expect the identity (6) to hold. “Something does happen” instead at the juncture of the concatenation $x' \oplus x''$ of the two strings $x' = /\text{ad}/ \text{ and } x'' = /\text{ta}/$: the concatenation creates the potentially dangerous cluster /ad/ (two obstructing disagreeing in voicing) which is otherwise missing from both sub-strings $x' = /\text{ad}/ \text{ and } x'' = /\text{ta}/$. It is thus conceivable that some CBG $G_{\preceq}$ fails at faithfully realizing the concatenated underlying string $x' \oplus x'' = /\text{adta}/$ because of this problematic cluster while succeeding at faithfully realizing the two individual underlying sub-strings $x' = /\text{ad}/ \text{ and } x'' = /\text{ta}/$, as stated in (7).
\[
G_{\preceq}(/\text{adta}/) \neq G_{\preceq}(/\text{ad}/) \oplus G_{\preceq}(/\text{ta}/) \tag{7}
\]
To formalize the difference between (6) and (7), we say that the strings $x'$ and $x''$ are independent relative to a constraint set $\mathcal{C}$ provided condition (8) holds for every constraint $C$ in the set $\mathcal{C}$ and every candidates $y'$ and $y''$ in the candidate sets $\mathcal{Y}(x')$ and $\mathcal{Y}(x'')$. The number of violations of the concatenated mapping $(x' \oplus x', y' \oplus y'')$ is the sum of the numbers of violations of the two mappings $(x', y')$ and $(x'', y'')$. Equivalently, concatenation
does not create nor destroy constraint violations.

\[ C(x' \oplus x'', y' \oplus y'') = C(x', y') + C(x'', y'') \]  

(8)

To illustrate, the strings \( x' = /ad/ \) and \( x'' = /da/ \) are not independent relative to the constraint set \( C \) in (9). In fact, condition (8) fails because, say, the two faithful mappings \( (x', y') = (/ad[ ], /ad[ ]); (x'', y'') = (/da[ ], /da[ ]) \) do not violate \textit{agree} while their concatenation \( (x' \oplus x'', y' \oplus y'') = (/ad/da[ ], /da/ad[ ]) \) does violate it. This failure of condition (8) captures the intuition that “something does happen” at the juncture in this case, namely an offending cluster is created.

\[ C = \begin{cases} 
C_1 = \text{AGREE}, C_2 = \text{NoVoice}, \\
C_3 = \text{IDENT}[\text{voice}] 
\end{cases} \]  

(9)

The strings \( x' = /ad/ \) and \( x'' = /da/ \) are instead independent: condition (8) holds, as verified by the identities in table 1, which formalize the intuition that no constraint violations are created or eliminated, so that “nothing happens” at the juncture.

**Summary**

This section has motivated the following question: under which conditions does a CBG preserve the part/whole structure induced by string concatenation? In other words, it satisfies the commutativity condition (5) whenever the two strings \( x' \) and \( x'' \) are independent in the sense of the constraint condition (8)? Section 4 will address this question.

### 3 Preserving part/whole structure: the case of underspecification

#### Underspecification and part/whole structure

Another source of phonological part/whole structure is \textit{underspecification}: a representation admits sub-representations which are “smaller” because underspecified. To illustrate, the round mid vowel can be represented as the tuple of feature values \([+\text{round} \text{−} +\text{high} \text{−} \text{low}]\). This tuple comes with sub-tuples such as \( x' = [+\text{round}] \) and \( x'' = [−\text{high} \text{−} \text{low}] \). These sub-tuples can be interpreted as sub-representations underspecified for height and for rounding. The original vowel can thus be construed as the combination \( x' \oplus x'' \) of these two underspecified sub-representations \( x' \) and \( x'' \) as in (10), analogous to (3).

\[ \begin{array}{c}
[+\text{rnd} \text{−} \text{high} \text{−} \text{low}] = [+\text{rnd}] \oplus [−\text{high} \text{−} \text{low}] \\
x' \oplus x''
\end{array} \]  

(10)

As another example, the syllable type VC can be represented as the tree on the left hand side of (11). This tree comes with the two sub-trees \( x' \) and \( x'' \) on the right hand side (that we will notate compactly as \( \text{V} \) and \( \text{VC} \)). These sub-trees can be interpreted as sub-representations underspecified for codas and for onsets, respectively. The syllable VC can thus be construed as the combination \( x' \oplus x'' \) of these two underspecified sub-representations \( x' \) and \( x'' \) as in (11), analogous to (3) and (10).

\[ \begin{array}{c}
\text{OSET} \text{RYTHEM} \\
\text{NUC} \text{CODA} \\
0 \text{V} \text{C} \text{NC}
\end{array} = \begin{array}{c}
\text{OSET} \text{RYTHEM} \\
\text{NUC} \text{CODA} \\
0 \text{V} \text{C} \text{NC}
\end{array} \oplus \begin{array}{c}
\text{OSET} \text{RYTHEM} \\
\text{NUC} \text{CODA} \\
0 \text{V} \text{C} \text{NC}
\end{array} \]  

(11)

To systematize these examples, we say that a representational framework \( (X, Y) \) is the \textit{combination} of two sub-frameworks \( (X', Y') \) and \( (X'', Y'') \) of underspecified sub-representations provided each underlying representation in the base set \( X \) admits a unique decomposition \( x' \oplus x'' \) into two underspecified underlying sub-representations \( x' \) and \( x'' \) in the base sets \( X' \) and \( X'' \) and vice versa each composition \( x' \oplus x'' \) in \( X' \oplus X'' \) yields an underlying representation in the
base set $\mathcal{X}$. Analogously, each surface representation in the candidate set $\mathcal{Y}(x' \oplus x'')$ admits a unique decomposition $y' \oplus y''$ into two underspecified surface sub-representations $y'$ and $y''$ in the candidate sets $\mathcal{Y}(x')$ and $\mathcal{Y}(x'')$ and vice versa each composition $y \oplus y''$ in $\mathcal{Y}(x') \oplus \mathcal{Y}(x'')$ yields a surface representation in the candidate set $\mathcal{Y}(x' \oplus x')$.

To illustrate, the representational framework $(\mathcal{X}, \mathcal{Y})$ for syllable types in (2) is the combination of the sub-frameworks $(\mathcal{X}^\prime, \mathcal{Y}^\prime)$ and $(\mathcal{X}''', \mathcal{Y}'''')$ underspecified for codas and for onsets as in table 2a. Analogously, consider the representational framework $(\mathcal{X}, \mathcal{Y})$ consisting of the six vowels $y$, $\partial$, $\epsilon$, $i$, $e$, and $\partial e$. It is the combination of the sub-frameworks $(\mathcal{X}^\prime, \mathcal{Y}^\prime)$ and $(\mathcal{X}''', \mathcal{Y}'''')$ underspecified for height and for rounding as in table 2b.

**Preserving part/whole structure**

We consider a CBG $G_{\prec} = G_{\prec}^{\mathcal{X}, \mathcal{Y}, C}$ from the underlying representations in $\mathcal{X}$ to the surface representations in $\mathcal{Y}$ corresponding to some constraint set $C$ and some vector order $\prec$. We assume that the representational framework $(\mathcal{X}, \mathcal{Y})$ is the combination of the two sub-frameworks $(\mathcal{X}^\prime, \mathcal{Y}^\prime)$ and $(\mathcal{X}'''', \mathcal{Y}'''')$. We consider some suitable extension of the constraint set $C$ from the mappings $(x, y)$ made available by $(\mathcal{X}, \mathcal{Y})$ to the mappings $(x', y')$ and $(x'', y'')$ made available by $(\mathcal{X}^\prime, \mathcal{Y}^\prime)$ and $(\mathcal{X}'''', \mathcal{Y}'''')$. Using the same vector order $\prec$, we construct the CBGs $G_{\prec}^\prime = G_{\prec}^{\mathcal{X}^\prime, \mathcal{Y}^\prime, C}$ and $G_{\prec}''' = G_{\prec}^{\mathcal{X}''', \mathcal{Y}'''', C}$ between the underlying sub-representations in $\mathcal{X}^\prime$ and $\mathcal{X}'''$ and the surface sub-representations in $\mathcal{Y}^\prime$ and $\mathcal{Y}'''$. We say that $G_{\prec}$ preserves the part/whole structure induced by the decomposition of $(\mathcal{X}, \mathcal{Y})$ into $(\mathcal{X}^\prime, \mathcal{Y}^\prime)$ and $(\mathcal{X}'''', \mathcal{Y}'''')$ iff the identity (12) holds for any underspecified sub-representations $x'$ and $x''$ in the sets $\mathcal{X}^\prime$ and $\mathcal{X}'''$. In this case, we say that the grammar $G_{\prec}$ decomposes into the grammars $G_{\prec}^\prime$ and $G_{\prec}'''$.

$$G_{\prec}(x' \oplus x'') = G_{\prec}^\prime(x') \oplus G_{\prec}'''(x'')$$

(12)

By (12), underlying representation that is the combination $x' \oplus x''$ of two under-specified underlying sub-representations $x'$ and $x''$ admits a surface realization $G_{\prec}(x' \oplus x'')$ that is the combination of the two underspecified surface sub-representations $G_{\prec}^\prime(x')$ and $G_{\prec}'''(x'')$. In other words, the job done by the grammar $G_{\prec}$ can be outsourced to two grammars $G_{\prec}^\prime$ and $G_{\prec}'''$, that each carry out half of it independently from the other.

This identity (12) is analogous to the identity (5). The crucial difference is that in the case of strings considered in section 2, the underlying strings $x', x''$ and their concatenation $x' \oplus x''$ all belong to the base set $\mathcal{X}$. The grammar $G_{\prec}$ is thus defined for all three underlying strings and thus appears at both sides of the identity (5). In the case of underspecification instead, the two underspecified underlying representations $x'$ and $x''$ do not belong to the base set $\mathcal{X}$. Thus, the right hand side of the identity (12) figures not the grammar $G_{\prec}$ but two extensions $G_{\prec}^\prime$ and $G_{\prec}'''$, thereof to the underspecified sub-representations.

To illustrate, we consider again the representational framework $(\mathcal{X}, \mathcal{Y})$ for syllable types and its decomposition into the sub-frameworks $(\mathcal{X}^\prime, \mathcal{Y}^\prime)$ and $(\mathcal{X}'''', \mathcal{Y}'''')$ in table 2a. The grammar $G$ in table 3a (that tolerates empty onsets but deletes codas) decomposes into the grammars $G^\prime$ (that tolerates empty onsets) and $G'''$ (that deletes codas).

**Independent sub-representations**

Consider the grammar $G$ in table 3b. It tolerates empty onsets and filled codas as long as they do not co-occur, as $\text{VC}$ is instead neutralized to $[\text{V}]$ rather than faithfully realized as $[\text{VC}]$. This grammar does not preserve the part/whole structure induced by the decomposition into sub-representations underspecified for codas and for onsets in table 2a. Indeed, the two sub-representations cannot be handled independently, as indicated by the question mark for $G'''$ in table 3b. Yet, this grammar $G$ would be the CBG corresponding to a constraint set containing a constraint that selectively penalizes the doubly-marked syllable type VC. Obviously, such a constraint does
not satisfy the constraint condition (8): it does not penalize the underspecified sub-representations \( x' = V \) and \( x'' = \Box V C \) but it does penalize their combination \( x' \oplus x'' = VC \). The two underspecified sub-representations are therefore not independent relative to such a constraint set. They instead do satisfy condition (8) and thus qualify as independent relative to the classical constraint set \( C \) in (2).

**Summary**

This section has motivated the following question: under which conditions does a CBG preserve the part/whole structure induced by the decomposition into under-specified sub-representations? In other words, it satisfies the commutativity condition (12) whenever the two underspecified underlying sub-representations \( x' \) and \( x'' \) are independent in the sense of the constraint condition (8)? We are now ready to address this question.

4 When do CBGs preserve structure?

**CBGs can fail to preserve part/whole structure**

We consider again the strings \( x' = /\text{ad}t\text{a}a/ \) and \( x'' = /\text{d}a/ \). As verified in table 1, they satisfy the constraint condition (8) and thus count as independent relative to the constraint set \( C \) in (9) and the candidacy function \( Y \) in (4). Yet, a CBG can flout condition (6) and thus fail to preserve the part/whole structure of the concatenation \( x' \oplus x'' = /\text{ad}t\text{a}a/>. 

To build a counterexample, let the sum of weighted squares of a 3-dimensional vector \( \mathbf{a} = (a_1, a_2, a_3) \) be the sum \( \text{SWS}(\mathbf{a}) = w_1 a_1^2 + w_2 a_2^2 + w_3 a_3^2 \) of its squared components rescaled by some weights \( w_1, w_2, \) and \( w_3 \). Two vectors \( \mathbf{a} \) and \( \mathbf{b} \) can be ordered based on their SWSs as in (13).

\[
\mathbf{a} \prec \mathbf{b} \text{ iff } \text{SWS}(\mathbf{a}) < \text{SWS}(\mathbf{b}) \quad (13)
\]

For concreteness, we choose the weights as \( w_1 = w_2 = 1 \) and \( w_3 = 1/2 \). It is easy to verify that the CBG \( G_{\prec} \) corresponding to the vector order \( \prec \) in (13) realizes the two underlying strings \( x' = /\text{ad}t\text{a}a/ \) and \( x'' = /\text{d}a/ \) as \( y' = [\text{atta}] \) and \( y'' = [\text{ta}] \), respectively. This CBG flouts condition (6) because it realizes the underlying concatenation \( x' \oplus x'' = /\text{ad}t\text{a}a/ \) faithfully as [adta] rather than as the concatenation \( y' \oplus y'' = [\text{atta}] \) of the realizations \( y' \) and \( y'' \) of the individual underlying strings \( x' \) and \( x'' \).

**Additive orders**

This counterexample shows that some restrictive assumption on the vector order \( \prec \) is needed for the corresponding CBG \( G_{\prec} \) to preserve part/whole structure in the sense of conditions (6) and (12). To this end, let us say that an order \( \prec \) among \( n \)-dimensional vectors is additive provided it satisfies the implication (14) for any three vectors \( \mathbf{a}, \mathbf{b}, \) and \( \mathbf{c} \) (Anderson and Feil 1988). This implication (14) captures the intuition that, if \( \mathbf{a} \) is smaller than \( \mathbf{b} \) and if the same quantity \( \mathbf{c} \) is added to both, the resulting sum \( \mathbf{a} + \mathbf{c} \) ought to be smaller than the sum \( \mathbf{b} + \mathbf{c} \) (all vector sums are component-wise).

\[
\text{If } \mathbf{a} \prec \mathbf{b}, \text{ then } \mathbf{a} + \mathbf{c} \prec \mathbf{b} + \mathbf{c}. \quad (14)
\]

To illustrate, this additivity condition (14) is satisfied by the vector order \( \prec_S \) in (1), for any choice of the set \( S \subseteq \{1, \ldots, n\} \). Although this additivity condition (14) feels intuitive, it can be shown to fail for instance for the order \( \prec \) defined in (13) in terms of SWSs. We will now see that this is expect, as we have just verified that the corresponding CBG does not preserve part/whole structure.

**Additive orders and part/whole structure**

The following proposition says that additivity of a vector order is sufficient to ensure that the corresponding CBG preserves phonological part/whole structure (see Prince 2015 for a special case of this result). Additivity can also be shown to be necessary, in the sense that for any order which is not additive we can construct a corresponding CBG that fails to preserve part/whole structure. Additivity thus provides a complete answer to the problem of characterizing preservation of part/whole structure posed in sections 2 and 3.

**Proposition 1 (A)** Consider a representational framework \( (X, Y) \) for strings. We focus on two
underlying strings \(x'\) and \(x''\) in the base set \(X\) such that their concatenation \(x' \oplus x''\) also belongs to \(X\) and its candidate set \(\mathcal{Y}(x' \oplus x'')\) is the concatenation \(\mathcal{Y}(x') \cdot \mathcal{Y}(x'')\) of the two candidate sets \(\mathcal{Y}(x')\) and \(\mathcal{Y}(x'')\). We assume that the two sub-strings \(x'\) and \(x''\) are independent relative to some constraint set \(C\), namely satisfy the constraint condition (8). Then, the CBG \(G_{x' \oplus x''} < C\) corresponding to any additive order \(\prec\) satisfies condition (5) and thus preserves part/whole structure. (B) Consider a representational framework \((X, \mathcal{Y})\) that is the combination of two factor representational frameworks \((X', \mathcal{Y}')\) and \((X'', \mathcal{Y}'')\) of underspecified sub-representations. We consider a constraint set \(C\) for the representational framework \((X, \mathcal{Y})\) that can be extended to the two factor representational frameworks \((X', \mathcal{Y}')\) and \((X'', \mathcal{Y}'')\) in such a way that they are independent, namely satisfy the constraint condition (8). Then, the CBG \(G_{x' \oplus x''} < C\) corresponding to any additive order \(\prec\) satisfies condition (12) and thus preserves part/whole structure.

To illustrate, consider the vector order \(\prec_S\) in (1). As noted in section 4, it is additive for any set \(S\). Proposition 1 thus ensures that the corresponding CBG \(G_{x \prec_S}\) preserves part/whole structure.

**Proof of proposition 1**

We focus on (A), as the proof of (B) is identical. Let us suppose that the CBG \(G_{\prec}\) realizes the underlying strings \(x'\) and \(x''\) as the surface strings \(y'\) and \(y''\) in the candidate sets \(\mathcal{Y}(x')\) and \(\mathcal{Y}(x'')\), namely \(G_{\prec}(x') = y'\) and \(G_{\prec}(x'') = y''\). The concatenation \(y' \oplus y''\) belongs to the candidate set \(\mathcal{Y}(x' \oplus x'')\) because of the inclusion \(\mathcal{Y}(x' \oplus x'') \supseteq \mathcal{Y}(x') \cdot \mathcal{Y}(x'')\). We need to show that the CBG \(G_{\prec}\) realizes the underlying concatenation \(x' \oplus x''\) as the surface concatenation \(y' \oplus y''\), namely \(G_{\prec}(x' \oplus x'') = y' \oplus y''\).

To this end, let us consider a candidate \(z\) in the candidate set \(\mathcal{Y}(x' \oplus x'')\) different from the candidate \(y' \oplus y''\). This candidate \(z\) must be the concatenation \(z = z' \oplus z''\) of some candidates \(z'\) and \(z''\) from \(\mathcal{Y}(x')\) and \(\mathcal{Y}(x'')\), because of the inclusion \(\mathcal{Y}(x' \oplus x'') \subseteq \mathcal{Y}(x') \cdot \mathcal{Y}(x'')\). The assumption \(z' \neq y'\) or \(z'' \neq y''\) (or both). Without loss of generality, we assume \(z' \neq y'\).

Since \(z' \neq y'\), the assumption \(G_{\prec}(x') = y'\) says that the constraint violation vector \(C(x', y')\) of the winner \((x', y')\) is smaller than the constraint violation vector \(C(x', z')\) of the loser \((x', z')\), as in (15). \(C(x', y') \prec C(x', z')\) (15)

Let us now turn to the other two candidates \(y''\) and \(z''\). If they are different as well, we reason analogously that the constraint violation vector \(C(x'', y'')\) of the winner \((x'', y'')\) must be smaller than the constraint violation vector \(C(x'', z'')\) of the loser \((x'', z'')\), as in (16). \(C(x'', y'') \prec C(x'', z'')\) (16)

If instead these two candidates \(y''\) and \(z''\) are identical, their constraint violation vectors \(C(x'', y'')\) and \(C(x'', z'')\) coincide, as stated in (17). \(C(x'', y'') = C(x'', z'')\) (17)

Since the order \(\prec\) satisfies the additivity condition (14), the inequality (15) and the identity (17) can be summed together into the inequality (18). \(C(x', y') + C(x'', y'') \prec C(x', z') + C(x'', z'')\) (18)

Suppose instead that it is the inequality (16) that holds rather than the identity (17). In this case, we note that the additivity condition (14) entails the variant in (19) for any four vectors \(a, b, c, d\). In fact, the assumption \(a \prec b\) in the antecedent of (19) ensures that \(a + c \prec b + c\) through the additivity condition (14). Analogously, the assumption \(c \prec d\) ensures that \(b + c \prec b + d\). The consequent \(a + c \prec b + d\) then follows by transitivity of \(\prec\).

If \(a \prec b\) and \(c \prec d\), then \(a + c \prec b + d\) (19)

Since the vector order \(\prec\) satisfies condition (19), the inequalities (15) and (16) can be summed together yielding once again the inequality (18).

By assumption, the two sub-strings \(x'\) and \(x''\) are independent in the sense of the constraint condition (8). Thus, the sum of the constraint violation vectors \(C(x', y')\) and \(C(x'', y'')\) on the left hand side of the inequality (18) coincides with the constraint violation vector \(C(x' \oplus x'', y' \oplus y'')\) of the concatenated mapping \((x' \oplus x'', y' \oplus y'')\). Analogously for the right hand side, whereby the inequality (18) can be rewritten as (20). \(C(x' \oplus x'', y' \oplus y'') \prec C(x' \oplus x'', z' \oplus z'')\) (20)

By (20), the candidate \(y' \oplus y''\) is less bad than any competing candidate \(z' \oplus z''\). The CBG \(G_{\prec}\) thus realizes the concatenated underlying form \(x' \oplus x''\) as the concatenated surface form \(y' \oplus y''\), namely \(G_{\prec}(x' \oplus x) = y' \oplus y''\) as desired.

5. HG and structure preservation

**Disconnection functions**

Let us consider a particularly natural way of ordering numerical vectors. We start from a function \(H\) that assigns to each vector \(a\) a number \(H(a)\) called
its disharmony (or its utility). Any two vectors $a$ and $b$ can then be ordered based on the size of their disharmonies $H(a)$ and $H(b)$ as in (21).

$$a \prec_H b \iff H(a) < H(b)$$

(21)
The disharmony function $H$ thus effectively defines a partial strict order $\prec_H$ among vectors.

To illustrate, let us consider again the numerical order $\prec$ defined in (13) in terms of SWSs. Obviously, it is the numerical order $\prec_H$ corresponding to the disharmony function $H$ that assigns to each vector $a$ its SWSs, namely $H(\cdot) = SWS(\cdot)$. Crucially, there exist numerical orders that are not induced by any disharmony function $H$. In the sense that condition (21) fails for some vectors, no matter how the disharmony function $H$ is chosen. For instance, that can be shown to be case for the vector order $\prec_S$ defined in (1), whenever the set $S$ has cardinality larger than one. The restriction to vector orders that are induced by disharmony functions is therefore substantive.

**Additive disharmony functions**

Section 4 has shown that the condition (14) that a vector order $\prec$ be additive is phonologically substantive because it ensures that the corresponding CBG $G_{\prec}$ preserves part/whole phonological structure. We thus tackle the following question: which assumptions on the disharmony function $H$ suffice to ensure that the corresponding vector order $\prec_H$ defined through (21) satisfies this phonologically substantive additivity condition (14)? We will now see that it suffices to assume that the disharmony of the sum $a + b$ of two vectors $a$ and $b$ is equal to the sum of their disharmonies, as stated in (22).

$$H(a + b) = H(a) + H(b)$$

(22)

Indeed, let us assume that the numerical order $\prec_H$ induced by a disharmony function $H$ satisfies the antecedent of the additivity implication (14), namely $a \prec_H b$. By definition (21), this means in turn that the disharmony $H(a)$ of the smaller vector $a$ is smaller than the disharmony $H(b)$ of the larger vector $b$, as in (23a). Let $H(c)$ be the disharmony of the vector $c$. Whatever this number $H(c)$ looks like, it can be added to both sides of the disharmony inequality $H(a) < H(b)$ without affecting it, yielding (23b). By the assumption (22) that the disharmony of a sum is the sum of the disharmonies, we can rewrite our inequality as in (23c). Finally, we use again the connection (21) between the disharmony function $H$ and the corresponding numerical order $\prec_H$ to reinterpret the disharmony inequality $H(a + c) < H(b + c)$ as the vector inequality $a + c \prec_H b + c$ required by the consequent of the additivity implication (14).

$$a \prec_H b \iff$$

$$\quad \iff H(a) < H(b)$$

$$\quad \iff H(a) + H(c) < H(b) + H(c)$$

(23)

$$\iff H(a + c) < H(b + c)$$

$$\iff a + c \prec_H b + c$$

(d)

**Finite representation of additive disharmonies**

The two preceding subsections have motivated numerical orders defined though disharmony functions which satisfy the identity (22) whereby the disharmony of a sum of vectors is the sum of their disharmonies. We now explore the phonological implications of this assumption (22). To this end, we consider an arbitrary underlying form $x$ in some base set $X$ and an arbitrary surface form $y$ in the corresponding candidate set $Y(x)$. And we compute the disharmony of the corresponding constraint violation vector $C(x, y)$ as in (24).

In step (24a), we have recalled that the components of the constraint violation vector $C(x, y)$ are the $n$ constraint violations $C_1(x, y), \ldots, C_n(x, y)$. In step (24b), we have baroquely rewritten this constraint violation vector $C(x, y)$ as the sum of many vectors: the vector with the 1st component equal to one and all other components equal to zero, repeated $C_1(x, y)$ times; the vector with the 2nd component equal to one and all other components equal to zero, repeated $C_2(x, y)$ times; and so on, down to the vector with the $n$th component equal to one and all other components equal to zero, repeated $C_n(x, y)$ times. We now make the crucial assumption that the disharmony function $H$ is additive. This means in particular that the disharmony of a sum of vectors is the sum of their disharmonies (the identity (22) extends trivially from two to an arbitrary finite number of vectors), yielding the identity (24c). Finally, let us call $w_1$ the disharmony of the vector with the 1st component equal to one and all other components equal to zero; let us called $w_2$ the disharmony of the vector with the 2nd component equal to one and all other components equal to zero; and so on. The disharmony of the constraint vector $C(x, y)$ can thus be described as the sum of the constraint violations $C_1(x, y), \ldots, C_n(x, y)$ rescaled by $w_1, \ldots, w_n$, as stated in (24d).

In conclusion, the disharmony function $H$ is
the one assumed in HG (Legendre et al. 1990b,a; Smolensky and Legendre 2006). And the HG weights \( w_1, \ldots, w_n \) can be interpreted as the disharmony of the base vectors that have one component equal to one and all other components equal to zero. These base vectors have no phonological meaning (they cannot be interpreted as constraint violation vectors). The reasoning in (24) thus illustrates the advantage of construing CBGs rather abstractly as in section 1, in terms of orders defined among arbitrary vectors.

The role of constraint integrality

As anticipated in section 1, the constraints used in phonology are assumed to only take (nonnegative) integer values, interpreted as numbers of violations. This assumption formalizes the intuition that the properties relevant to phonology are discrete—contrary to the properties relevant to phonetics, which are instead continuous and thus cannot be quantified through just integers. This constraint integrality assumption yields a number of finiteness effects when coupled with plausible assumptions on numerical orders. For instance, Magri (2019) shows that (when coupled with a restriction to vector orders that are monotone), constraint integrality ensures that all candidate sets can be assumed to be finite without loss of generality. The reasoning in (24) illustrates another finiteness effect of the constraint integrality assumption. In fact, this reasoning crucially relies on the fact that the constraint violation vector \( C(x, y) \) can be expressed as a sum of a certain number of base vectors. Obviously, this decomposition is only possible because the components \( C_k(x, y) \) of a constraint violation vector are integers but would fail otherwise.\(^1\) The reasoning in (24) can thus be interpreted as another finiteness effect of the constraint integrality assumption: when this constraint integrality assumption is coupled with a restriction to numerical orders defined through additive disharmony functions, it ensures that the disharmony function admits a finite representation in terms of a finite number \( n \) of weights \( w_1, \ldots, w_n \).

6 Conclusions

This paper has shown that the HG implementation of constraint-based phonology follows from the desideratum of part/whole structure preservation plus the restriction to disharmony-based orders. The latter assumption does not seem to admit a phonological justification but it is quite natural from a formal perspective. The former assumption of structure preservation has instead been phonologically motivated in sections 2-3 from the perspectives of string concatenation and underspecification. In other words, HG admits an axiomatic derivation from axioms which are phonologically or formally motivated. The derivation crucially relies on the constraint-integrality assumption that phonologically relevant properties are discrete in nature.

\(^1\) This reasoning (24) is essentially the proof of the Fundamental Theorem of Linear Algebra (Strang 2006), whereby a linear function between finite dimensional spaces admits a matrix representation. The only twist is that we do not need linearity (namely additivity plus homogeneity) but additivity suffices, because we are only dealing with integral vectors.
References

Marlow Anderson and Todd Feil. 1988. *Lattice Ordered Groups: an Introduction*. D. Reidel Publishing Company.

David Haussler. 1999. Convolution kernels on discrete structures. Technical report, Technical report, Department of Computer Science, University of California at Santa Cruz.

Shigeto Kawahara. 2006. A faithfulness ranking projected from a perceptibility scale: The case of [+voice] in Japanese. *Language*, 82:536–574.

Géraldine Legendre, Yoshiro Miyata, and Paul Smolensky. 1990a. Harmonic Grammar: A formal multi-level connectionist theory of linguistic well-formedness: An application. In *Proceedings of the 12th annual conference of the Cognitive Science Society*, pages 884–891, Hillsdale, NJ. Lawrence Erlbaum Associates.

Géraldine Legendre, Yoshiro Miyata, and Paul Smolensky. 1990b. Harmonic Grammar: A formal multi-level connectionist theory of linguistic well-formedness: Theoretical foundations. In *Proceedings of the 12th annual conference of the Cognitive Science Society*, pages 388–395, Hillsdale, NJ. Lawrence Erlbaum.

Giorgio Magri. 2019. Finiteness of optima in constraint-based phonology. Manuscript, CNRS.

John J. McCarthy. 2003. OT constraints are categorical. *Phonology*, 20(1):75–138.

Joe Pater. 2016. Universal grammar with weighted constraints. In Joe Pater and John J. McCarthy, editors, *Harmonic Grammar and Harmonic Serialism*, pages 1–46. Equinox, London.

Alan Prince. 2015. One tableau suffices. Manuscript (Rutgers University). Available from the Rutgers Optimality Archive as ROA 1250.

Alan Prince and Paul Smolensky. 1993/2004. *Optimality Theory: Constraint Interaction in generative grammar*. Blackwell, Oxford. Original version, Technical Report CU-CS-696-93, Department of Computer Science, University of Colorado at Boulder, and Technical Report TR-2, Rutgers Center for Cognitive Science, Rutgers University, April 1993. Available from the Rutgers Optimality Archive as ROA 537.

Paul Smolensky and Géraldine Legendre. 2006. *The Harmonic Mind*. MIT Press, Cambridge, MA.

Gilbert Strang. 2006. *Linear Algebra and its applications*. Thomson Brooks/Cole.