Computational Complexity and Sour-Grapes-Like Patterns

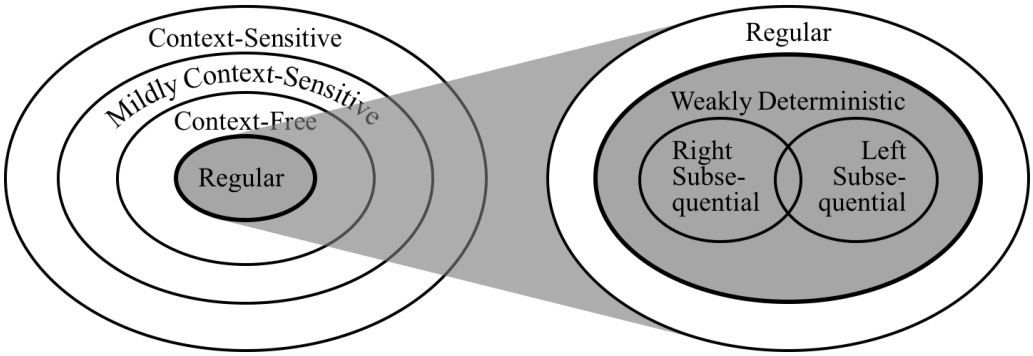
Charlie O'Hara[1], Caitlin Smith[1,2]
[1]*University of Southern California,* [2]*University of North Carolina at Chapel Hill*

1 Introduction

Phonological patterns can be viewed as mappings from underlying (input) to surface (output) forms. The set of attested phonological patterns is smaller than the set of all logically possible input-output mappings; these attested patterns appear to be bounded by computational complexity. One notion of complexity is defined based on the types of dependencies that exist between segments in licit output strings. A foundational claim made by Chomsky (1956) maintains that the computational complexity of languages forms a nested hierarchy. Generally, various syntactic input-output mappings are classified within the relatively complex region of the hierarchy. Phonological patterns, on the other hand, appear to be no more complex than those in the *regular* class of languages.

Phonological input-output mappings may be even less computationally complex than those in the regular class of languages. The Subregular Hypothesis (Heinz 2011) claims that all attested phonological mappings are a proper subset of the class of regular input-output mappings. The formal characterization of that subset is the subject of ongoing debate. In particular, this debate has focused on the *weakly deterministic* class of mappings, pictured in (1). Heinz & Lai (2013) propose that this weakly deterministic class may mark the upper bound on the computational complexity of phonological systems. This proposal is motivated in particular by this class' performance on classifying harmony (feature spreading) patterns. They found that attested unidirectional and bidirectional harmony patterns can be classified as weakly deterministic, while unattested patterns such as the *sour grapes* spreading pathology identified by Wilson (2003) are posited not to be weakly deterministic, but rather more complex.

(1) Chomsky Hierarchy of computational complexity and Subregular Hierarchy



Recently, the status of the weakly deterministic class as the upper bound on the computational complexity of phonological patterns has been called into question. Jardine (2016) identifies several attested *unbounded circumambient* phonological patterns, those in which the potential targets of a process are

* For helpful questions, comments, and discussion we thank Eric Baković, Reed Blaylock, Jeff Heinz, Hayeun Jang, Adam Jardine, Karen Jesney, Andrew Lamont, Anna Mai, Adam McCollum, Eric Meinhardt, Elliott Moreton, Jon Rawski, Arthur Santana, Rachel Walker, Yifan Yang, and audiences at SCAMP 2018, AMP 2018, and SCiL 2019.

Proceedings of *AMP 2018*

dependent on both preceding and following material that may be unboundedly far away. He claims that such unbounded circumambient patterns do not fit within the class of weakly deterministic input-output mappings, but are rather more complex. One major subtype of unbounded circumambient patterns are those that resemble the unattested sour grapes spreading pattern described by Wilson (2003); these *sour-grapes-like* patterns are attested in tonal phonology. In this paper, we claim that such attested sour-grapes-like patterns differ meaningfully from what we call the *true sour grapes* spreading pathology. We propose that the *false sour grapes* processes attested in some tonal systems are computationally less complex than the unattested true sour grapes pathology, due to the presence of *zones of predictability* local to the trigger of spreading. In particular, the false sour grapes patterns can be shown to fall into the class of weakly deterministic mappings, while the true sour grapes spreading pattern cannot.

The paper is organized as follows. In section 2, we define and characterize two different types of sour-grapes-like patterns of featural and tonal spreading. In section 3, we provide background on the definitions of several classes within the Subregular Hierarchy of computational complexity. In section 4, we demonstrate that different types of sour-grapes-like spreading can be distinguished in terms of their computational complexities. Section 5 concludes.

2 Sour-grapes-like spreading patterns

Unbounded circumambient patterns, as defined by Jardine (2016), are those in which the output form of some potential target is dependent on both preceding and following material that can be unboundedly far away from it. One subtype of the unbounded circumambient patterns are the sour-grapes-like patterns of featural and tonal spreading. In this section, we discuss two such patterns: the unattested sour grapes spreading pathology, and the attested sour-grapes-like tonal spreading of Copperbelt Bemba.

2.1 *True sour grapes spreading* Wilson (2003) identifies a pathological pattern of feature spreading known as sour grapes spreading. In this pattern, a potential undergoer U that co-occurs at any distance with a trigger T assimilates to that trigger.[1] For example, assuming progressive (rightward) spreading, an underlying form /TUUUU/ surfaces as [TTTTT]. However, if a blocker B appears anywhere after a trigger, any potential undergoers do not assimilate to the trigger; an underlying /TUUBU/ surfaces as [TUUBU], with no spreading, rather than [TTTBU], with partial spreading. In other words, a phonological property borne by the trigger spreads to the edge of a domain or not at all.

The unbounded circumambience of sour grapes spreading is illustrated in (2). In sour grapes spreading, whether a potential undergoer U in the input maps to T or to U in the output depends on the presence or absence of a preceding T, as well as a following B.

(2) Sour grapes spreading as an unbounded circumambient pattern

/T U U B U/ ⟶ [T U U B U]

status as undergoers of spreading determined by material on both sides (T and B)

We define a sour-grapes-like pattern to be any unbounded circumambient process that is blocked from applying to a target when a trigger is present on one side while a blocker is present on its other side. We also claim that there are fundamentally different subtypes of sour-grapes-like spreading patterns. The subtype discussed in this section can be viewed as the prototypical or *true sour grapes* spreading pattern. Another subtype of sour-grapes-like spreading, exemplified by the tone spreading of Copperbelt Bemba, is discussed in the following subsection.

[1] While this pathology owes its name to Padgett (1995), the sour grapes spreading described by Wilson is fundamentally different from the spreading pathology described by Padgett.

2.2 *Copperbelt Bemba tone spreading* Copperbelt Bemba (Bantu; Zambia) exhibits a sour-grapes-like pattern of unbounded progressive (rightward) tone spreading (Bickmore & Kula 2013; Kula & Bickmore 2015; Jardine 2016). The last high tone in the word spreads unboundedly to the end of a word, as illustrated by the data in (3). (An acute accent indicates a high tone; a grave accent indicates a low tone.)

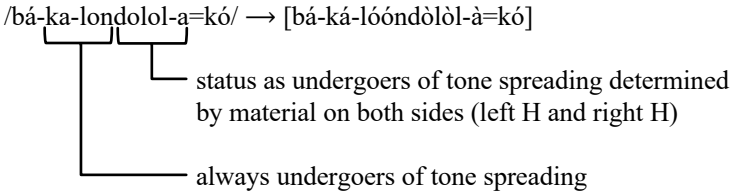
(3)
a.	/bá-ka-fík-a/	[bá-ká-fík-á] 'they will arrive'
b.	/bá-la-mu-luk-il-a/	[bá-lá-mú-lúk-íl-á] 'they weave/plait for him/her'
c.	/ú-ku-londolol-a/	[ú-kú-lóóndólól-á] 'to introduce, explain'
d.	/tu-ka-páapaatik-a/	[tù-kà-páápáátík-á] 'we will flatten'

If another high tone intervenes between a high tone and the end of a word, this unbounded spreading is blocked. However, rather than spreading a high tone onto the tone bearing units preceding the blocker, and rather than not spreading at all, in Copperbelt Bemba the high tone instead spreads onto the two following tone bearing units. This is illustrated in (4).

(4)
a.	/bá-ka-pat-a=kó/	[bá-ká-pát-à=kó] 'they will hate a bit'
b.	/á-mu-luk-il-a=kó/	[bá-mú-lúk-ìl-a=kó] 'they weave/plait a bit for him'
c.	/bá-ka-londolol-a=kó/	[bá-ká-lóóndòlòl-à=kó] 'they will introduce them'
d.	/bá-londolol-é/	[bá-lóóndólòl-é] 'let them explain'

Like the sour grapes spreading pathology described above, this process of high tone spreading in Copperbelt Bemba fits Jardine's definition of unbounded circumambience. Because high tone spreading is only triggered by the final high tone in a word, the status of any low-toned tone bearing unit as a potential target of tone spreading depends on the presence of a preceding high tone, as well as the absence of a following high tone. This is illustrated in (5) for the sequence of potential targets of tone spreading in [bá-ká-lóóndòlòl-à=kó] 'they will introduce them.'

(5) Sour-grapes-like tone spreading in Copperbelt Bemba



However, the presence of bounded ternary spreading as part of the larger tone spreading pattern in Copperbelt Bemba separates this process from the unattested true sour grapes spreading pathology. Crucially, a low tone cannot appear on the two tone bearing units following an underlying high tone within the same word. We refer to this as a *zone of predictability*, a substring whose output form is predictable based on surrounding material. We propose that such a zone of predictability local to the trigger of spreading presents a crucial distinction between the types of sour-grapes-like spreading patterns presented here. In the pathological true sour grapes spreading discussed in section 2.1, no change occurs to the input string when a blocker is present. Some attested sour-grapes-like patterns, such as Copperbelt Bemba tone spreading, are cases of what we refer to as *false sour grapes* spreading, in which the presence of a blocker does not prevent all spreading. In the following sections of this paper, we examine how the distinction between true and false sour grapes is reflected in their different degrees of computational complexity.

3 Computational complexity

3.1 *Background* The complexity of a phonological pattern can be evaluated through the formal machinery necessary to describe it. A large number of phonological patterns (though crucially not unbounded circumambient processes) can be described using either left or right subsequential functions. A subsequential

function can be described by a rule with an unbounded number of segments on at most one side of the rule's context, as in (6).[2]

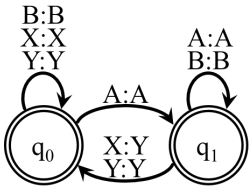
(6) Subsequential input-output functions as rules

a. Left subsequential: $X \rightarrow Y / AB_0$ __
b. Right subsequential: $X \rightarrow Y /$ __B_0A

Functions can also be conceptualized as *finite state transducers*. A finite-state transducer, like a rule, determine a language's licit input-output mappings. If an input-output mapping can be generated by a transducer for a given language, then that mapping is considered grammatical in that language. The transducer is navigated by following the *transitions* (arrows) between *states* (circles). Each transition is associated with an 'Input:Output' pair. To generate an output from an input, an input string is scanned, and for each symbol 'Input' encountered in the input string, a symbol (or string of symbols) 'Output' is added to the output string. An output may consist of a symbol in a language's alphabet, or

The input-output mapping described by the rule in (6a) can be captured by the finite state transducer in (7) by scanning an input string from right to left. The same transducer can also capture the mapping described by the rule in (6b) by scanning an input string from left to right. A subsequential function can be captured using a *deterministic* finite state transducer, that is, a finite state transducer in which each state has only one transition for each symbol in the alphabet, and may serve as a possible end state (represented here by double outlining).

(7) Subsequential input-output function as a deterministic finite state transducer



All regular input-output mappings can be decomposed into one left subsequential and one right subsequential function, which can be applied to a string in either order (Elgot & Mezei 1965). The Subregular Hypothesis (Heinz 2011) proposes that phonological patterns are not as complex as those in the set of regular input-output mappings. Heinz & Lai (2013) also show that phonological mappings can be more complex than those that fall within the class of left and right subsequential mappings. In order to capture attested non-subsequential patterns within a sub-regular class, Heinz & Lai define the *weakly deterministic* class of input-output mappings. Like regular mappings, weakly deterministic mappings can be decomposed into left and right subsequential functions. However, there are restrictions placed on the types of string transformations that these functions may produce; specifically, the left and right subsequential functions of a weakly deterministic mapping must be both alphabet- and length-preserving. That is, these functions cannot introduce new symbols into a language's alphabet (set of symbols), nor change the number of symbols in a string. These restrictions limit the types of input-output mappings that can be captured by such weakly deterministic functions.

3.2 *Complexity of sour-grapes-like patterns* The sour grapes spreading pattern introduced in section 2.1 is an example of a regular input-output mapping that can only be decomposed into a left and a right subsequential function through the use of intermediate mark-up. True sour grapes spreading can be described by the regular rule in (8).

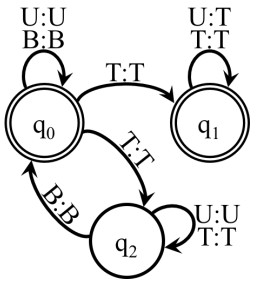
[2] Throughout this paper, all rules within a function are defined to be applied in parallel to all targets within the described environments, with environments always interpreted based on the input to the rule. Thus, there is no feeding or bleeding by any rule application within the application of a single function. We conjecture that based on this type of rule application, rules are cleanly translatable into finite state transducers.

(8) True sour grapes spreading as a rule

$$U \rightarrow T \,/\, T\begin{Bmatrix} U \\ T \end{Bmatrix}_0 \_\_ \begin{Bmatrix} U \\ T \end{Bmatrix}_0 \#$$

Like all regular rules, the true sour grapes rule in (8) can also be conceptualized as a finite state transducer, as in (9). Note, however, that this finite state transducer is not deterministic: state q_2 is not a possible end state, and there are two transitions from q_0 that take T as their input. This reflects that true sour grapes spreading is a more complex mapping than those that fall within either the left or right subsequential classes.

(9) True sour grapes spreading as a non-deterministic finite state transducer



The sour grapes spreading rule in (8) can be decomposed into left and right subsequential functions. This is made possible by using a mark-up strategy to store information about what material occurs unboundedly far away from a potential trigger of spreading on the trigger itself. Assuming a progressive (rightward) spreading process, the presence or absence of a blocker unboundedly far from a potential trigger of spreading can first be marked on the trigger itself by a right subsequential function, as in (10a-b).

(10) True sour grapes spreading decomposed into subsequential rules

Step 1, right subsequential function	Step 2, left subsequential function
a. $T \rightarrow T_B \,/\, \_\_ \begin{Bmatrix} U \\ T \end{Bmatrix}_0 B$	c. $U \rightarrow T \,/\, T_{\neg B} \begin{Bmatrix} U \\ T_{\neg B} \end{Bmatrix}_0 \_\_$
b. $T \rightarrow T_{\neg B} \,/\, \_\_ \begin{Bmatrix} U \\ T \end{Bmatrix}_0 \#$	d. $\begin{Bmatrix} T_B \\ T_{\neg B} \end{Bmatrix} \rightarrow T \,/\, \_\_$

This intermediate mark-up on the trigger eliminates the need for a later rule such as (10c) to include information about the presence or absence of both triggers and blockers unboundedly far from any potential undergoers. In the left subsequential function, only information about the trigger (whether it is a successful trigger $T_{\neg B}$ or unsuccessful trigger T_B) is necessary for the rule in (10c) to determine if assimilation of an undergoer U takes place. The applications of the right and left subsequential functions are illustrated in (11).

(11) Right and left subsequential functions generating true sour grapes spreading

Input	T U U U	T U B U	
Intermediate	$T_{\neg B}$ U U U	T_B U B U	Right subsequential function
Output	T T T T	T U B U	Left subsequential function

While sour grapes spreading fits the definition of a regular pattern, Heinz & Lai (2013) define the sub-regular class of weakly deterministic mappings as those that can be decomposed into a left subsequential and a right subsequential function, such that neither function is string-length-increasing, nor adds additional symbols to the language's alphabet. These restrictions minimize the amount of intermediate mark-up that is available to the two subsequential functions. The mark-up strategy utilized in (10) to capture true sour grapes

spreading does not fit this definition, as it introduces the symbols T_B and $T_{\neg B}$ to the language's alphabet. Heinz & Lai therefore claim that sour grapes spreading is unattested because of its relatively high computational complexity. It is a regular input-output mapping, but does not fall within the more stringently defined weakly deterministic class.

4 Sour-grapes-like patterns and intermediate mark-up

Unbounded circumambient patterns such as sour-grapes-like spreading require some amount of intermediate mark-up in order to be decomposed into left and right subsequential functions. The mark-up strategy implemented in section 3.2 to capture true sour grapes spreading introduced special symbols present only at the intermediate level. However, not all potential mark-up strategies must make use of special symbols. In this section, we propose that false sour grapes spreading patterns can be decomposed into a left subsequential and a right subsequential function that are alphabet- and length-preserving through the use of a *substring mark-up strategy*. As a result, such patterns fit the definition of weak determinism outlined by Heinz & Lai (2013). True sour grapes spreading, on the other hand, can only be captured with the use of the more powerful *special symbol markup strategy*, positioning this pattern outside of the weakly deterministic class of input-output mappings.

In order for an intermediate mark-up strategy to successfully capture a sour-grapes-like spreading pattern, the difference between successful and unsuccessful triggers of spreading must somehow be marked up within a string by one subsequential function in such a way that the next subsequential function can distinguish them. In addition, whatever mark-up is used must *uniquely* represent successful (spreading) triggers and unsuccessful non-spreading triggers. In this section, we differentiate between two types of intermediate mark-up strategies: *special symbol mark-up,* and *substring mark-up*.

Special symbol mark-up refers to the mark-up strategy used for true sour grapes spreading in section 3.2. This strategy may introduce symbols not already present in a language's alphabet. Substring mark-up, on the other hand, relies on symbols already present in the alphabet in order to perform mark-up on an intermediate string. Crucially, substring markup requires the presence of a zone of predictability within a string. Because the presence of a zone of predictability within a string is the result of two or more underlying substrings neutralizing to one output substring, the underlyingly contrastive substrings can be utilized to uniquely mark up the intermediate representation within the zone of predictability.

4.1 *False sour grapes spreading* In this section, we demonstrate that the tone spreading pattern in Copperbelt Bemba introduced in section 2.2 can be captured using substring mark-up, suggesting that this pattern falls into the weakly deterministic class of input-output mappings. Recall that in Copperbelt Bemba, the last high tone in a word spreads unboundedly to the right edge of the word, while any other high tone spreads only onto the two following tone bearing units. The rules in (12) describe this high tone spreading pattern.

(12) Copperbelt Bemba high tone spreading as rules

a. $L \longrightarrow H \,/\, HL_0\_\_L_0\#$
b. $L \longrightarrow H \,/\, H(L)\_\_$

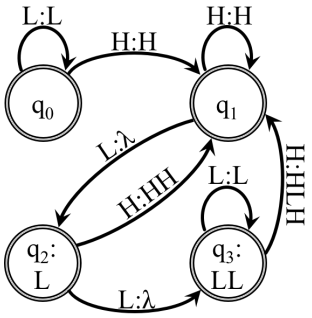
There are two components to Copperbelt Bemba tone spreading: unbounded and bounded spreading. According to the unbounded spreading rule in (12a), a low tone will surface as a high tone after an input high tone unboundedly far away, provided no other high tone follows. The rule in (12b) produces bounded spreading, which Bickmore & Kula (2013) and Kula & Bickmore (2015) refer to as 'ternary spreading.' According to this rule, all low tones that appear within two tone bearing units after any high tone will surface as high.

The unbounded circumambient nature of unbounded high tone spreading in Copperbelt Bemba is reflected in the non-subsequentiality of the rule in (12a). Nevertheless, this mapping can be decomposed into alphabet- and length-preserving right and left subsequential functions via the use of substring mark-up. This is possible due to the zone of predictability local to any underlying high tone in Copperbelt Bemba. These zones of predictability are the result of the ternary spreading produced by the rule in (12b). The finite state

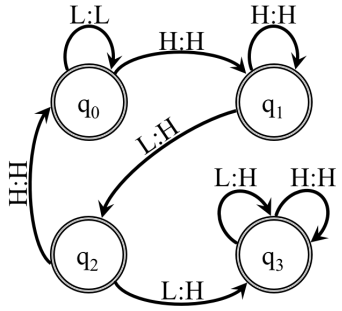
transducers associated with the two subsequential functions that describe Copperbelt Bemba tone spreading using substring mark-up are provided in (13). In these transducers, an output of λ indicates that no symbol is output when some input symbol is observed; a symbol or string of symbols within a state indicates what is output if the string-scanning procedure ends at that state.

(13) Copperbelt Bemba high tone spreading as finite state transducers

a. Right subsequential function (right to left scan of string)



b. Left subsequential function (left to right scan of string)



This substring mark-up strategy can also be conceptualized as the output of the sets of subsequential rules in (14), decomposed from the rules in (12).

(14) Copperbelt Bemba high tone spreading decomposed into subsequential functions

Step 1, right subsequential function
a. $L \longrightarrow H / H\_\_H$
b. $L \longrightarrow H / HL\_\_L_0H$

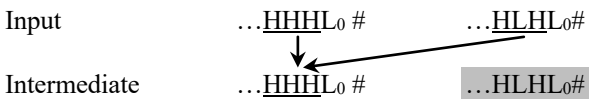
Step 2, left subsequential function
c. $L \longrightarrow H / HLL_0\_\_$
d. $L \longrightarrow H / HH\_\_$
e. $L \longrightarrow H / LLH\_\_$
f. $L \longrightarrow H / \#(L)H\_\_$

Instead of marking up the final high tone in a word, the successful trigger of unbounded tone spreading, as $H_{\#}$ and any non-final (unsuccessfully triggering) high tone as H_H in the first subsequential function, the rules in (14) mark up these tones using predictable substrings of symbols already in the alphabet. We use the intermediate substring HLH to mark up a non-final high tone, while a final high tone maps faithfully to HLL. To ensure that any intermediate HLH substring is uniquely present in the intermediate stage as the result of mark-up of a non-final high tone in an input string, the right subsequential function also maps underlying HLH to HHH (14b). Next, the left subsequential function spreads H unboundedly rightward from intermediate substring HLL (14c), and spreads from any other intermediate H (found in substring HLL and the initial H in substring HLH) to the next two tone bearing units (14d-f).

This substring mark-up strategy is successful because every high tone in Copperbelt Bemba spreads onto at least the two following tone bearing units, neutralizing the contrast between high and low tones in those positions. As a result, there is a zone of predictability local to any potential trigger of tone spreading. This zone of predictability allows for the underlying contrast in the tone bearing units following an underlying trigger to be eliminated by the first subsequential function, making space for a new contrast (one between successful and unsuccessful triggers of spreading) to be generated using the substrings that would otherwise not persist to the intermediate form. This new contrast allows the first subsequential function to mark up information about the presence or absence of blockers that may be unboundedly far from a potential trigger on substrings that are local to that trigger. This mark-up can carry the same type of information as mark-up symbols $T_{\neg B}$ and T_B in the discussion of true sour grapes spreading in section 3.2, while using no special symbols that are not included in a language's alphabet. As a result, the input-output mapping for Copperbelt Bemba tone spreading can be classified as weakly deterministic.

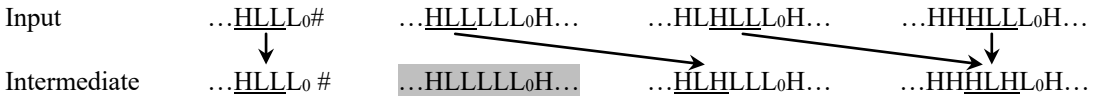
The substring mark-up strategy for Copperbelt Bemba tone spreading is illustrated below. The neutralization of input HLH and HHH by the right subsequential function is shown in (15); this allows the intermediate substring HLH to be used to uniquely represent an unsuccessful trigger of unbounded spreading in the intermediate stage.

(15) Neutralization of input substrings HHH and HLH by right subsequential function



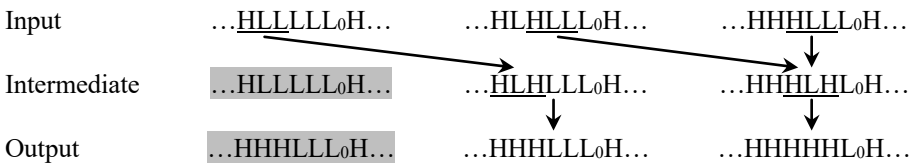
The mark-up of input HLL sequences is shown in (16). If the high tone of a substring HLL is the final high tone in a word (and thus a successful trigger of spreading), that substring is mapped faithfully from the input to the intermediate stage. However, any substring HLL containing a non-final high tone is marked up as HLH. Thus, the intermediate string generated by the right subsequential function stores information about whether a high tone is followed by any other high tones via mark-up. However, this substring mark-up has not neutralized any contrasts that would have otherwise remained in the output.

(16) Mappings from input substring HLL by right subsequential function

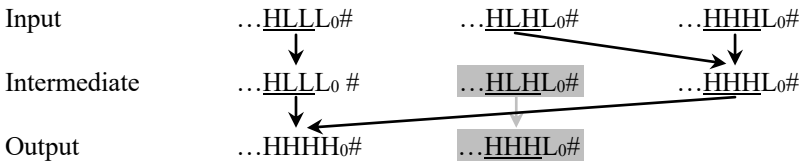


In the second (left) subsequential function, we can use the substring trigger mark-ups produced by the right subsequential function to determine whether or not to apply unbounded tone spreading. Intermediate substring HLH, which represents an unsuccessful trigger of unbounded spreading, maps to output substring HHH as a result of bounded ternary spreading, but does not spread a high tone any further. However, intermediate substring HLL, whether preceded by an arbitrary string (on the left) or high tone (on the right), represents a successful trigger of unbounded spreading, and therefore maps to output substring HHH in accordance with ternary spreading, followed by high tones unboundedly to the end of a word, in accordance with unbounded spreading.

(17) Mappings from intermediate HLH by left subsequential function



(18) Mappings from other intermediate substrings by left subsequential function



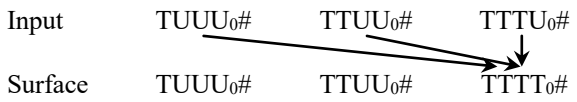
As an unbounded circumambient pattern, false sour grapes spreading requires the power of intermediate mark-up in order to be captured by two deterministic functions. However, the ternary spreading exhibited within Copperbelt Bemba tone spreading creates a zone of predictability following all potential targets of spreading, making it possible for the tone spreading pattern as a whole to be captured using substring mark-

up. The substring mark-up strategy is alphabet- and length-preserving; as a result, the success of this mark-up strategy shows that Copperbelt Bemba tone spreading can be classified as a weakly deterministic pattern.

4.2 *True sour grapes spreading* While the substring mark-up strategy deployed in section 4.1 can be used to identify a weakly deterministic mapping for Copperbelt Bemba tone spreading, the same strategy cannot be applied to the true sour grapes spreading pathology. In this section, we demonstrate why a substring mark-up cannot capture true sour grapes spreading. Crucially, this results from the lack of a zone of predictability in true sour grapes spreading.

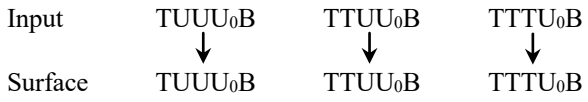
In true sour grapes spreading, there is only a loss of input contrast when no blocker is present in a string. As illustrated in (19), the contrast between a trigger T and a undergoer U is lost between the trigger and the right word edge.

(19) True sour grapes: neutralization when no blocker is present



Before a blocker, however, triggers T and undergoers U always contrast on the surface, as in (20).

(20) True sour grapes: no neutralization when blocker is present



In this subsection, we demonstrate that no successful substring mark-up strategy for true sour grapes exists. Any successful mark-up strategy for sour grapes spreading must create a contrast in the intermediate stage between triggers preceding a blocker T_B and triggers not preceding a blocker $T_{\neg B}$. Let substring M be the mark-up substring for potential triggers not preceding a blocker ($T_{\neg B}$ in the special symbol mark-up strategy). Substring M is a specific substring of symbols present in the alphabet, which we here will assume is of length *n* and made up of symbols T and U. The first function to apply, the right subsequential function, scans a string and finds any input T that is not followed by a blocker B. The function maps the T and the *n-1* following symbols T and U to the intermediate mark-up substring M (21a).

However, in order for the use of substring M for intermediate mark-up to be a successful strategy, it is crucial that substring M never appear before a blocker B in the intermediate stage. Otherwise, the next (left) subsequential function to apply will be unable to distinguish a substring M that precedes a blocker from one that does not. In (21b), an undergoer U that is between mark-up substring M and a blocker B always surfaces faithfully. On the other hand, in (21c) an undergoer U that is between substring M with no blocker B following (17c) maps to T.

(21) True sour grapes spreading rules attempting to utilize substring mark-up

Step 1, right subsequential function

a. $T\begin{Bmatrix}U\\T\end{Bmatrix}_{n-1} \longrightarrow M \,/\, \_\_\begin{Bmatrix}U\\T\end{Bmatrix}_0\#$

Step 2, *not* left subsequential function

b. $U \longrightarrow U \,/\, M\begin{Bmatrix}U\\T\end{Bmatrix}_0\_\_\begin{Bmatrix}U\\T\end{Bmatrix}_0B$

c. $U \longrightarrow T \,/\, M\begin{Bmatrix}U\\T\end{Bmatrix}_0\_\_\begin{Bmatrix}U\\T\end{Bmatrix}_0\#$

Note that these rules are not subsequential. In order to make the second function left subsequential, the intermediate stage must be defined such that the environment for the rule (21b) is never encountered, and that the rule never applies. In that case, all substrings M present in the intermediate stage are known to not be followed by a blocker, rendering the right side of the rule's environment redundant.

To achieve this, the first subsequential function must be defined such that it maps any pre-blocker input substring M unfaithfully, resulting in no intermediate M…B sequences. If we assume that input substring M is mapped to some other arbitrary substring N of equal length by the first (right) subsequential function (22a), the next (left) subsequential function must map intermediate substring N to output substring M, because all input material that precedes a blocker must surface faithfully. However, this in turn means that input substring N cannot map to itself in the intermediate stage, or else it would also map to substring M in the output. Input substring N must instead map to some other arbitrary intermediate substring O (22b). There are only a finite number of substrings of length *n*; if we assume that they must be made up only of symbols T and U, there are only 2^n such substrings. Thus, we can repeat this mapping procedure 2^n times, but in order to maintain a contrast between all 2^n substrings before a blocker, one of these substrings must map to substring M intermediately.

(22) Failed attempt to make intended mark-up substring M appear only when no blocker is present

a. $M \longrightarrow N \,/\, \_\_\begin{Bmatrix}U\\T\end{Bmatrix}_0B$

b. $N \longrightarrow O \,/\, \_\_\begin{Bmatrix}U\\T\end{Bmatrix}_0B$

…

z. $X \longrightarrow M \,/\, \_\_\begin{Bmatrix}U\\T\end{Bmatrix}_0B$

Substring M fails as a successful mark-up because it cannot uniquely represent successful triggers of spreading, i.e. those that are not followed by a blocker. From this result, we claim that true sour grapes spreading cannot be captured using the substring mark-up strategy. The crucial difference between true and false sour grapes spreading patterns lies in the absence of any pre-blocker zone of predictability in cases of true sour grapes spreading. Because all input substrings remain contrastive in the output when they appear before blockers, there is no substring that can be used for mark-up to create an intermediate contrast between successful and unsuccessful triggers of spreading.

5 Conclusion

This paper identifies a distinction in computational complexity between attested sour-grapes-like patterns of spreading, and the unattested true sour grapes spreading pathology. The attested false sour grapes spreading patterns differ from the true sour grapes spreading pattern in the presence of a zone of predictability local to a potential trigger of spreading. Cases of false sour grapes spreading, exemplified by Copperbelt Bemba tone spreading, involve some type of surface neutralization that generates such zones of predictability, even in environments in which unbounded spreading does not occur. Though all unbounded circumambient patterns require some sort of mark-up to be used in the application of a pair of subsequential mappings, the true and false sour grapes spreading patterns differ in the types of mark-up required. False sour grapes spreading need only rely on substring mark-up, a less powerful type of mark-up strategy than is required to capture true sour grapes spreading. This substring mark-up strategy allows attested false sour grapes spreading patterns to be classified as weakly deterministic input-output mappings according to the definition put forward by Heinz & Lai (2013).

This paper leaves open several issues for further reflection and research, particularly with respect to the characterization of the weakly deterministic class of mappings. The weakly deterministic class was originally defined with the intention of banning all mark-up strategies, and therefore excluding all unbounded circumambient processes. However, the substring mark-up strategy introduced and utilized in this paper reveals that there are loopholes available in the definition of weak determinism. Whether the intended character of the weakly deterministic class can be captured through independent restrictions on the left and right subsequential functions that make up a weakly deterministic mapping remains to be seen. Such restrictions may need to make crucial reference to the way the subsequential functions interact in composition; see McCollum et al. (2018) for discussion.

References

Bickmore, Lee S., & Kula, Nancy C. (2013) Ternary Spreading and the OCP in Copperbelt Bemba. Studies in African Linguistics, 42(2), 101–132.

Chomsky, Noam (1956) Three Models for the Description of Language. IRE Transactions on Information Theory, 2(3), 113–124.

Elgot, C. C., & Mezei, J. E. (1965) On relations defined by generalized finite automata. IBM Journal of Research and Development, 9, 47–68.

Heinz, Jeffrey (2011) Computational Phonology – Part I: Foundations. Language and Linguistics Compass, 5/4, 140–152.

Heinz, Jeffrey, & Lai, Regine (2013) Vowel Harmony and Subsequentiality. In A. Kornai & M. Kuhlmann (Eds.), Proceedings of the 13th Meeting on the Mathematics of Language (MoL 13) (pp. 52–63). Association for Computational Linguistics.

Jardine, Adam (2016) Computationally, tone is different. Phonology, 33(2), 247–283.

Kula, Nancy C., & Bickmore, Lee S. (2015) Phrasal phonology in Copperbelt Bemba. Phonology, 32, 147–176.

McCollum, Adam G, Baković, Eric, Mai, Anna, & Meinhardt, Eric (2018) The expressivity of segmental phonology and the definition of weak determinism. Unpublished ms., University of California San Diego.

Padgett, Jaye (1995) Partial Class Behavior and Nasal Place Assimilation. In K. Suzuki & D. Elzinga (Eds.), Proceedings of the Arizona Phonology Conference: Workshop on Features in Optimality Theory (pp. 145–183). Tucson: University of Arizona.

Wilson, Colin (2003) Analyzing unbounded spreading with constraints: marks, targets, and derivations. Unpublished ms., University of California Los Angeles.